ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE MÉCANIQUE
M. ING.

PAR
YAN BOUTIN

INTÉGRATION DES CYCLES D'USINAGE AVANCÉS DES MACHINES-OUTILS À COMMANDE NUMÉRIQUE AUX LOGICIELS DE FAO

MONTRÉAL, le 22 JANVIER 2002

INTÉGRATION DES CYCLES D'USINAGE AVANCÉS DES MACHINES-OUTILS À COMMANDE NUMÉRIQUE AUX LOGICIELS DE FAO

Yan Boutin

Sommaire

L'utilisation efficace des machines-outils à commande numérique (MOCN) exige un haut niveau d'intégration entre le logiciel FAO et la machine. La dernière génération de contrôleur de MOCN offre des routines avancées propriétaires qui permettent une augmentation significative des possibilités et de la flexibilité des machines.

Le présent projet définit une méthodologie d'intégration des modules FAO et de post-processeur de manière à rendre disponible à même le logiciel de FAO les cycles avancés propriétaires des MOCN. Un exemple d'application à été réalisé dans le système CFAO Pro/ENGINEER 2000i^2 ainsi que dans le logiciel de post-processeur CAM-POST.

Des essais d'usinage effectués en utilisant le module développé ont permis de prouver que l'intégration des cycles avancés permettait d'obtenir des programmes d'usinage de taille réduite offrant une meilleure lisibilité. Suite à l'usinage, le modèle CFAO est mis à jour automatiquement par la lecture du programme, qui peut avoir été modifié par l'opérateur de la MOCN.

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

- M. Jean-François Chatelain, directeur de mémoire
  Département de génie mécanique à l'École de technologie supérieure

- M. Roland Maranzana, professeur
  Département de génie de la production automatisée à l'École de technologie supérieure

- M. Stéphane Chalut
  Spécialiste en développement CAO-FAO, Bombardier Aéronautique

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET UN PUBLIC

LE 13 DÉCEMBRE 2001

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

INTÉGRATION DES CYCLES D'USINAGE AVANCÉS DES MACHINES-OUTILS À COMMANDE NUMÉRIQUE AUX LOGICIELS DE FAO

Yan Boutin

Sommaire

L'utilisation des machines-outils à commande numérique (MOCN) et des logiciels de conception et de fabrication assistées par ordinateur (CAO / FAO) est devenue omniprésente dans le domaine de la fabrication. Ces technologies exigent un haut niveau d'intégration de manière à permettre le transfert d'une information de qualité entre la plate-forme logicielle et le contrôleur de la machine, assurant ainsi un degré de performance et de rentabilité maximisés. Les derniers développements technologiques au niveau des contrôleurs de MOCN font en sorte que l'on y retrouve maintenant des routines avancées, permettant une augmentation significative des possibilités et de la flexibilité des machines les utilisant.

Toutefois, le développement de ces nouvelles fonctionnalités est effectué de manière propriétaire par chaque fabricant de contrôleur. Ceci a comme conséquence que la majorité des fonctionnalités avancées actuellement offertes avec les MOCN ne peuvent pas être utilisées par les systèmes de FAO lors de la génération des trajets d'outils.

Le présent projet a pour but de définir une méthodologie d'intégration des modules FAO et de post-processeur de manière à rendre disponible à même le logiciel de FAO les cycles avancés des MOCN. Le projet a permis de réaliser une telle intégration à l'aide du développement logiciel d'un module spécialisé dans le système CFAO Pro/ENGINEER 2000i^2 de la firme PTC ainsi que dans le logiciel de post-processeur CAM-POST de ICAM. Il a été prouvé qu'une telle intégration était possible, des essais d'usinage ont été effectués à titre comparatif en utilisant le module qui a été développé et avec l'approche traditionnelle.

Les résultats des essais comparatifs d'usinage avec et sans l'intégration proposée ont permis de prouver que l'intégration des cycles avancés permettait d'obtenir une meilleure lisibilité des programmes d'usinage ainsi qu'une plus large possibilité de modifications du programme par l'opérateur de la MOCN. De plus, les programmes de taille réduite ainsi obtenus, une fois modifiés par l'opérateur de la MOCN, peuvent être réacheminés au département d'ingénierie afin d'assurer la mise à jour automatique des modèles CFAO. Les temps d'usinage lors de l'utilisation des cycles avancés sont comparables à ceux qui résultent de l'emploi des capacités natives du système FAO.

CNC MACHINE TOOL ADVANCED CANNED CYCLES INTEGRATION TO CAM SOFTWARE

Yan Boutin

Abstract

The manufacturing industry is taking advantage of the latest developments in computer aided design (CAD), computer aided manufacturing (CAM) and computer numerical control (CNC). The quality of the information transferred between the CAM system and the machine tool is highly dependant on the level of integration that exist between those components of the manufacturing system. The newest CNC controllers are packed with features like advanced canned and probing cycles, increasing the possibilities and flexibilities of the machines using those controllers.

However, the development of those new functionalities is made on a proprietary basis by each and every controller builder. The consequence of this fact is that the vast majority of those functionalities is not available through the CAM software used to generate the toolpaths.

The goal of the present project is to define an integration methodology between the CAM, post-processor and CNC machine controller to allow the CAM software to take full advantage of the advanced canned cycles available in the machine-tool controller. An actual application of the developed methodology has been accomplished in Pro/ENGINEER 2000i^2 CAM module and ICAM's CAM-POST post-processor to prove the possibilities of the concept. It has been proven that such an integration was both possible and practical. Actual part cutting was accomplished using the developed integration and results were compared to those of a part machined using regular toolpath generation techniques.

The results obtained from the machining experiments have proven that the integration of the advanced canned cycles were resulting in a better lisibility of the part programs while allowing a wider possibility of modifications of the program from the machine's operator. The resulting programs, of smaller size, can be sent back to the engineering department after usage to allow an automatic update of the CAD/CAM models following the modifications that might have been done by the operator. The cutting times resulting from the usage of the advanced canned cycles are about the same than those of programs using the native functionalities of the CAM system.

REMERCIEMENTS

Bien que le travail présenté ici porte mon nom, je n'en suis pas moins redevable a plusieurs personnes, que je prendrai ici le temps de nommer. Je tiens tout d'abord à remercier M. Jean-François Chatelain qui a rempli à pleine mesure son rôle de directeur de maîtrise grâce à sa disponibilité et à ses judicieux conseils. Son appui technique et moral durant la réalisation de mes recherches ont permis de mener à terme le présent travail.

Parmi les membres du personnel de l'école qui m'ont grandement aidé par leur support, je tiens particulièrement à mentionner les professeurs Christian Masson et Louis Lamarche, qui ont su me prodiguer de judicieux conseils et qui ont cru en moi lors de la réalisation de mes études et de ma recherche.

Je tiens également à remercier mes parents sans qui je n'aurais jamais possédé la force de caractère et les outils nécessaires à la réalisation d'un tel travail. Je leurs dédie ce travail.

De plus, je me dois de mentionner la participation financière du Fond pour la Formation de Chercheurs et l'Aide à la Recherche (FCAR) et du Programme de bourses institutionnelles de l'École de technologie supérieure. Les fonds qui m'ont été gracieusement offerts durant les deux années d'études supérieures m'ont permis de me concentrer sur la réalisation de ce projet.

A tous ceux que je ne mentionne pas ici et qui ont participé directement ou indirectement à la réalisation de ce travail, sachez que votre contribution a été grandement appréciée.

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS ET SIGLES

ANSI	American National Standards Institute
API	Application Programmation Interface
APT	Automatically Programmed Tools
BCL	Basic Control Language (Initialement Binary Cutter Location)
B-Rep	Boundary Representation
CAD	Computer aided drawing
CAM	Computer aided machining
CAO	Conception assistée par ordinateur
CCA	Cycles Cannés Avancés
CFAO	Conception et fabrication assistée par ordinateur
CL-File	Cutter Location File
CNC	Computer Numerical Control
CSG	Constructive Solid Geometry
DIN	Deutsches Institut für Normung
DNC	Direct Numerical Control
DXF	Drawing eXchange Format
EIA	Electronic Industries Alliance
FAO	Fabrication assistée par ordinateur
FTP	File Transfer Protocol
IGES	Initial Graphic Exchange Specification
ISO	International Standards Organisation
MCD	Machine Control Data
MICCA	Module d'Intégration des Cycles Cannés Avancés
MOCN	Machine-Outil à Commande Numérique
NASA	National Aeronautics & Space Administration
NC	Numerical Control
NCITS	National Committee for Information Technology Standards
NURBS	Non-Uniform Rational B-Spline
PC	Personal Computer
PCMCIA	Personal Computer Memory Card International Association
PLC	Programmable Logic Controller

PPI	Post-Processeur Inversé
STEP	Standard for the Exchange of Product model data
STEP-NC	STEP - Numerical Control
STL	Stereolithography file format
UHV	Usinage à Haute Vitesse

INTRODUCTION

Il va sans dire que l'apparition des technologies électroniques et informatiques a bouleversé la façon de vivre de toutes les sociétés actuelles. Il demeure très peu de domaine où l'intégration de l'électronique ne se soit pas effectuée d'une manière directe ou indirecte. Le domaine de l'usinage par enlèvement de copeaux ainsi qu'à un niveau plus étendu, celui de la fabrication mécanique n'ont pas échappé à cette révolution.

Les avantages de l'intégration de la puissance de calcul offerte par les processeurs aux différents procédés de mise en forme ont été rapidement reconnus. Des besoins de fabrication de pièces mécaniques plus précises, présentant un meilleur fini de surface et dont les profils sont décrits par des équations mathématiques ont été parmi les éléments qui ont menés à la création des machines-outils à commande numérique (MOCN).

C'est précisément dans le but de répondre aux besoins de fabrication de pièces aéronautiques que le Massachusetts Institute of Technology (MIT) a procédé, au début des années 50, à la première intégration de l'électronique à une machine-outil, à la demande du Département de la Défense des États-Unis. Cette union entre une machine physique et un contrôle asservi a résulté en la première MOCN, capable de déplacer en simultané 3 axes cartésiens selon un programme pré-établi. Il était dorénavant possible de fabriquer avec précision des pièces comportant des surfaces complexes.

Suite à la reconnaissance des possibilités offertes par de telles machines, l'intégration de l'électronique aux procédés d'usinage évoluait à la vitesse des découvertes effectuées dans le domaine de l'électronique. L'augmentation rapide de la puissance des processeurs au cours des années 70 et 80 a résulté en une multitude de nouvelles applications dans le domaine de l'usinage. Les systèmes de Conception Assistée par Ordinateur (CAO) et de Fabrication Assisté par Ordinateur (FAO) sont apparus, les contrôleurs de MOCN sont devenus plus rapides, performants et simples d'utilisation et les procédés d'usinage ont augmenté en productivité.

L'évolution de ces technologies s'est effectuée à un rythme effréné et parfois dans des directions différentes, en se basant en partie sur la technologie déjà implantée. Ainsi, aujourd'hui, le langage utilisé afin de communiquer l'information aux contrôleurs de MOCN est basé sur un standard qui date d'environ 30 ans, qui diffère légèrement de machine en machine et qui est différent du langage de sortie des systèmes de FAO. Il existe de plus une multitude de formats de fichiers pour les systèmes de CAO et FAO, qui sont en grande majorité complètement incompatibles.

Ces incompatibilités mineures ou majeures entre les différentes composantes d'un système entier de fabrication résultent en une baisse de productivité, une hausse de coûts et une augmentation des risques d'erreurs dans le procédé de fabrication.

Puisque les incompatibilités actuellement existantes dans un système de fabrication complet incluant les logiciels et les contrôleurs de MOCN sont réparties sur plusieurs niveaux et sont différentes les unes des autres, une solution unique à l'ensemble de ces problèmes nécessite la refonte complète de la structure du système. La présente recherche vise donc à proposer une solution à l'incompatibilité existant entre l'information générée par les systèmes de FAO et l'information que nécessite en entrée le contrôleur de MOCN. Plus précisément, une solution est proposée afin de résoudre les problèmes d'intégration des cycles d'usinage avancés des MOCN aux systèmes de FAO.

Ainsi, le but du présent travail vise à développer une méthodologie générale d'intégration des modules MOCN et FAO au niveau des fonctionnalitées avancées, selon une méthode définie suite à une analyse des possibilités d'intégration. De plus, un exemple d'intégration entre le module FAO du logiciel Pro/Engineer et un contrôleur de MOCN Fanuc sera présenté.

Afin de pouvoir évaluer la solution qui sera proposée, le chapitre 1 fournit les informations nécessaires à la compréhension des problèmes qui existent présentement au niveau de l'intégration des différents modules matériels et logiciels constituant un système com-

plet de conception et de fabrication. Une revue de l'état de l'art est également incluse afin de pouvoir orienter la présente solution en tirant parti des recherches et expérimentations qui ont déjà été réalisées. La méthodologie d'intégration est développée et présentée au chapitre 2. Afin de valider les hypothèses posées lors de la définition de la méthodologie d'intégration, un exemple d'application de cette intégration est réalisé entre le logiciel CFAO Pro/Engineer 2000i^2 conçu par PTC, un post-processeur développé à l'aide de CAM-POST de la firme ICAM et un contrôleur de MOCN Fanuc 16i - Seicos monté à une MOCN Hitachi-Seiki VS-50. Le chapitre 3 décrit en détail cette intégration ainsi que les difficultés rencontrées lors de son élaboration. Les expérimentations qui ont été réalisées à l'aide de cet exemple d'application sont décrites au chapitre 4 et les résultats des essais sont analysés au chapitre 5. Finalement, un résume de l'ensemble du travail et les horizons à explorer afin de pouvoir élargir les applications de la présente recherche sont cités en guise de conclusion et de recommendations.

CHAPITRE 1

REVUE DE LA LITTÉRATURE

La revue de littérature débute en présentant les différents modules qui composent un système complet intégré de conception et de fabrication assistée par ordinateur. Le cheminement que doit effectuer l'information à chaque étape comprise entre la conception d'une pièce jusqu'à l'obtention de son modèle physique est également présenté. Par la suite, une analyse des différentes méthodes en cours de développement pour chacun des modules du système est présenté, et chacune des solutions est évaluée.

1.1 Cheminement de l'information

Afin de permettre une compréhension des problèmes d'intégration des différents modules d'un système complet de fabrication, il convient tout d'abord de présenter le cheminement de l'information entre l'étape initiale du concept de la pièce à fabriquer jusqu'à l'obtention de la pièce physique. L'origine des problèmes d'intégration peut ainsi être mise à jour, ce qui permet d'adopter une meilleure approche à leur résolution.

Un système intégré de fabrication est un système qui comporte tous les modules nécessaires pour passer du concept d'une pièce à l'obtention de son modèle physique usiné. Ce système comporte différents modules logiciels et physiques, représentés par des rectangles à la figure 1, tel le module de conception assistée par ordinateur (CAO) ou encore la machine-outil à commande numérique (MOCN). Une flèche reliant deux modules à la figure 1 indique un transfert d'information entre les modules. Une flèche pleine indique la direction conventionnelle de transfert d'information alors qu'une flèche vide représente une direction possible que peut emprunter l'information selon les capacités des modules liés par les flèches.

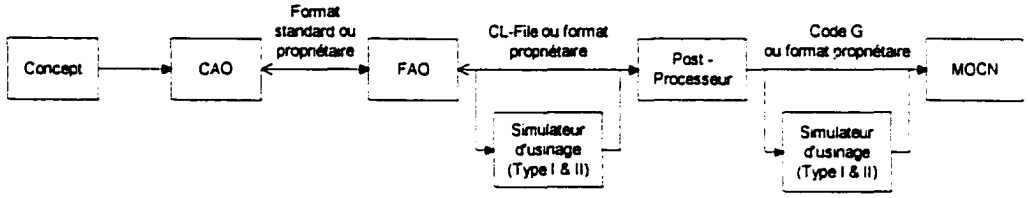


Figure 1 Cheminement de l'information Concept – MOCN

On remarque qu'un format propriétaire ou standardisé est utilisé entre chacun des modules utilisés pour traiter l'information. Dans le cas du format propriétaire, une diminution de la flexibilité du système en entier en résulte.

L'information décrivant la pièce passe au format électronique dans le logiciel de CAO, pour ensuite être transférée au module de fabrication assistée par ordinateur (FAO). L'information qui en résulte est ensuite traduite par le module de post-processeur afin de produire les données qui seront envoyées au contrôleur de la machine-outil à commande numérique (MOCN). Des étapes de simulation d'usinage peuvent être optionnellement incluses avant le module de post-processeur et avant le transfert à la MOCN. Les prochaines sections décrivent en détail les problèmes d'intégration présents lors du transfert d'information entre chacun des modules illustrés.

1.1.1 Cheminement Concept — CAO

Initialement, les caractéristiques de la pièce à fabriquer sont déterminées par le rôle de la pièce en service. On détermine alors des caractéristiques telles que le fini de surface, la précision requise, le matériel à utiliser, les traitements thermiques et de surface à réaliser, pour ne citer que celles-ci. À partir de ces informations, la conception de la pièce à l'aide d'un logiciel de CAO peut être effectuée. C'est à cette étape que les informations qui définissent la pièce sont transformées sous une forme électronique qui sera sauvegardée dans

un fichier, soit sous un format propriétaire au logiciel utilisé pour effectuer la conception ou encore sous un format neutre.

Il est toutefois courant que la compagnie qui effectue la conception de la pièce relègue la fabrication à un sous-traitant qui possède un logiciel CAO différent de celui qui a servi a réaliser la conception. Puisque les formats de fichiers des logiciels sont propriétaires et non divulgués, ceci force l'utilisation d'un format neutre et standardisé pour effectuer le transfert entre les deux plate-formes.

Toutefois, l'utilisation d'un format standardisé résulte immanquablement en une perte d'information, et l'ampleur de cette perte d'information est dépendante à la fois des capacités du format neutre choisi et des différents types d'information qui peuvent être représentées par le logiciel CAO utilisé. La majorité des formats standardisés actuels ne permettent de représenter que la géométrie et la topologie de la pièce, mis à part le format STEP (Standard for the Exchange of Product model data) dont il sera question à la prochaine section.

Les logiciels de CAO modernes incluent beaucoup plus d'information dans le modèle de la pièce que la géométrie seule, et toute cette information supplémentaire ne peut transiger par le format neutre et sera donc éliminée. Parmi les informations qui ne sont pas transmises par le format neutre, notons les types suivants :

1. Caractéristiques de la pièce (features)
2. Paramétrisation
3. Propriétés physiques
4. Associativité géométrique

La majorité des logiciels de CAO permettent d'utiliser une approche basée sur les caractéristiques de la pièce afin de réaliser la modélisation "feature based part modeling". Ainsi, le design de la pièce est effectué à l'aide des éléments qui représentent la pièce tels que

des alésages, des poches, des chambrages ou des ilôts. Une telle méthode est très efficace car elle est basée sur la fonctionnalité de la pièce et les caractéristiques à usiner qui sont incluses directement dans le modèle. Une telle représentation est beaucoup plus riche que celle ne contenant que la géométrie.

De plus, les principales plate-formes CAO permettent de décrire paramétriquement les relations entre les éléments géométriques de la pièce ou entre les caractéristiques. On peut ainsi définir mathématiquement les relations fonctionnelles de la pièce à même le modèle. Ceci permet de pousser à un niveau plus élevé l'incorporation de la fonctionnalité de la pièce dans le modèle. Il est donc possible d'établir une relation qui assure, par exemple, que le diamètre d'un alésage est égal à 50% d'un second alésage appartenant à la pièce. Lors de la modélisation d'assemblages de pièces, des contraintes qui dictent la position relative des pièces sont incorporées dans le modèle. De plus, des relations paramétriques peuvent définir les liens entre les différentes pièces. À titre d'exemple, il est possible de spécifier que deux alésages situés sur deux pièces différentes de l'assemblage conserveront le même diamètre.

En addition au paramétrage, les modules de CAO permettent de définir les propriétés physiques et mécaniques de plusieurs matériaux utilisés dans les modèles, de sorte qu'il soit possible d'utiliser ces informations dans les modules d'analyse par éléments finis ou de génération de trajectoires d'outils (FAO). Dans ce dernier cas, ce type d'information pourrait être utilisé afin de suggérer des valeurs par défaut pour les vitesses de coupe et d'avance. Tout ceci n'est possible que si ces informations peuvent être transmises entre les modules par le format de fichier choisi, ce qui n'est pas le cas avec l'utilisation des formats normalisés.

De plus, les formats normalisés utilisés par les logiciels de CAO ne permettent de représenter qu'un nombre fini de types différents d'entités géométriques. Habituellement, les logiciels comportent des type d'entités géométriques dont la représentation est propriétaire et ne possèdent pas de représentation équivalente dans un format neutre. Une

conversion doit alors prendre place afin que les éléments géométriques touchés puissent être approximés à l'aide des types d'entités disponibles dans le format standardisé choisi pour le transfert d'information.

Citons comme exemple le format normalisé IGES (Initial Graphic Exchange Specification) qui est le plus largement utilisé et qui permet la représentation d'un maximum de 50 types différents d'entités géométriques [1]. Ainsi, un fichier converti en IGES et lu à nouveau dans le logiciel CAO l'ayant généré risque de présenter des différences au niveau de la représentation géométrique et jusqu'à un certain point des inexactitudes de topologies lorsque comparé au modèle original avant l'exportation.

Il est en effet courant qu'une réparation des surfaces représentant une pièce soit nécessaire suite à l'importation par l'entremise d'un format standardisé. Ces réparations visent principalement des surfaces initialement adjacentes et dont les arêtes initialement communes s'entrecroisent ou sont espacées.

Il va sans dire que la perte d'informations liée au transfert par l'entremise d'un format neutre résulte en une diminution de productivité et d'efficacité, tout en augmentant les risques d'erreurs et en diminuant la précision.

1.1.2 Cheminement CAO — FAO

Lorsque la conception CAO est achevée, il est nécessaire de générer à l'aide d'un logiciel FAO les trajectoires d'outils qui seront utilisées par la MOCN afin d'effectuer l'usinage requis pour obtenir le produit fini à partir du brut. Il existe alors deux types de cheminements possibles, soit l'emploi d'un format natif commun au module de CAO et de FAO ou l'emploi d'un format neutre normalisé.

Dans le premier cas, le logiciel de FAO est intégré au logiciel de CAO utilisé pour la conception, ce qui permet de lire la géométrie à usiner directement à partir d'un fichier

au format natif du logiciel utilisé pour la conception. Les caractéristiques à usiner sont ainsi transférées à partir du format natif du logiciel au module FAO, ce qui permet d'associer les opérations d'usinage aux caractéristiques de la pièce. Cette approche permet notamment l'obtention d'une associativité entre les trajets d'outils et la géométrie de la pièce, tout en conservant les qualités de paramétrisation et d'associativité géométrique. Une modification des dimensions de la pièce ou de sa géométrie dans le module CAO sera immédiatement répercutée sur les trajectoires d'outils qui seront mises à jour afin de refléter les changements apportés à la pièce.

Bien que ce type de cheminement de données entre les modules CAO et FAO ne présente pas de problèmes d'intégration, il peut dans certains cas comporter des désavantages. En effet, les efforts de développement d'une telle plate-forme CFAO sont distribués sur les différents modules du logiciel et ne sont pas uniquement axés sur l'évolution des stratégies d'usinage et la qualité des algorithmes de génération de trajectoires d'outils. Certains logiciels sont toutefois orientés purement vers la génération de trajectoires d'outils et offrent parfois des capacités supérieures en ce domaine que les plate-formes CFAO.

L'utilisation de logiciels axés sur la FAO représente le deuxième type de cheminement CAO–FAO possible. La géométrie à usiner sera donc créée dans un logiciel de CAO performant, puis elle sera transférée au logiciel de FAO par l'entremise d'un format neutre. Toutefois, l'utilisation de ces formats standardisés de transfert fait en sorte que l'ensemble des caractéristiques composant la pièce est perdu. Seules des représentations de type fil de fer, surfacique ou solide sont transférées. De plus, cette représentation risque de nécessiter fréquemment des réparations de surfaces ou ne représente pas avec exactitude le modèle de départ.

De plus, cette approche n'est pas propice aux modifications de la géométrie de la pièce ou des dimensions. En effet, tout changement exige une modification du modèle dans le système de CAO qui doit de nouveau être transféré par un format neutre au logiciel FAO. Puisqu'il n'existe pas d'associativité entre les surfaces et les trajectoires d'outils lors du

transfert par format neutre, les trajectoires d'outils doivent être de nouveau définies en entier.

1.1.3 Cheminement FAO – Post-Processeur

C'est au niveau du transfert d'information entre les systèmes FAO et les Post-Processeurs qu'existent la majorité des problèmes d'intégration dans tout le système de fabrication intégré. Afin de comprendre la problématique de transfert d'information entre le système FAO et le Post–Processeur, il convient de présenter le langage APT (Automatically Programmed Tools). Ce langage a été créé au MIT en parallèle avec la première MOCN. Les entrées à ce logiciel sont la géométrie à usiner ainsi que les déplacements que l'outil doit effectuer en relation avec cette géométrie. La sortie, appelée CL-File (Cutter Location File), contient les coordonnées des points par lesquels devra passer l'outil de coupe afin d'usiner la géométrie définie et qui a servie d'entrée au programme [5]. Ce langage est standardisé par ANSI NCITS 37 :1999 et est géré par le comité J7 [2] qui est toujours actif et se charge d'étendre les possibilités du langage afin de l'adapter aux nouvelles réalités technologiques du domaine de l'usinage, telles que les courbes NURBS (Non-uniform Rational B-Spline).

Les grandes plate-formes FAO et CFAO telles que Pro/ENGINEER et CATIA utilisent le CL-File comme format de représentation des trajectoires d'outils. Ce format contient les déplacements d'outils, traduit à l'aide d'un sous-ensemble de mots du langage APT, nécessaires à l'usinage de la pièce . Toutefois, le langage d'entrée utilisé par la majorité des MOCN est différent du CL-FILE. Il convient alors d'utiliser un post-processeur, qui va traiter à nouveau cette information et créer un fichier de données au format du langage d'entrée de la MOCN, soit le code G, que ISO 6983 tente de standardiser.

Bien que le format des langages APT et CL-File soit normalisé, il existe des déviations à ces normes. Ainsi, il est possible que certaines commandes présentes dans le CL-File que produit un système de FAO ne soient pas utilisables par un post-processeur développé pour un autre logiciel de FAO qui utilise également le CL-File comme format de sortie.

De plus, seulement les grandes plate-formes FAO et CFAO utilisent nativement le CL-File comme format de sortie et utilisent le langage APT afin d'obtenir les trajectoires d'outils. Les logiciels FAO de moins grande envergure utilisent un format intermédiaire propriétaire qui est converti par un post-processeur également propriétaire. Ce type de post-processeur utilise le format intermédiaire comme entrée afin de générer le code G requis par la machine. Ceci force l'utilisation du post-processeur livré avec le système de FAO afin de générer le code G nécessaire au contrôle de la MOCN. Généralement, ces post-processeurs intégrés sont beaucoup moins flexibles au niveau de leur capacité de personnalisation que ceux disponibles individuellement.

Tel qu'illustré à la figure 1, il est possible d'effectuer une simulation d'usinage avant d'effectuer le transfert de l'information au module de post-processeur. Cette simulation permet de réaliser un usinage graphique virtuel de la pièce afin de s'assurer que les trajectoires d'outils ainsi que les paramètres associés soient définis de manière convenable dans le module de FAO.

Il existe principalement deux catégories principales de modules de simulation d'usinage. Le premier type effectue une simulation à l'aide de la géométrie de la pièce finie, de la pièce brute et de l'outil. Le second type, en plus de représenter les fonctionnalités du premier type, fournit également une représentation de la MOCN qui effectue l'usinage. Ce second type est plus sécuritaire car il permet de détecter des erreurs d'usinage liées à la configuration et aux limites physiques de la MOCN.

Lors du cheminement d'information entre le module de FAO et le module de post-processeur, l'information décrivant les trajectoires d'outils n'est disponible que sous forme de

CL-File ou un format propriétaire équivalent. C'est donc à partir de ces données que le module de simulation d'usinage pourra effectuer la représentation graphique des déplacements d'outils.

1.1.4 Cheminement Post-Processeur — MOCN

Le module de Post-processeur est utilisé avec chaque système FAO qui doit produire des données pour une MOCN. Le rôle principal du post-processeur consiste à traduire l'information contenue dans le CL-File en code G pour une machine spécifique. On parle ici de traducteur car le post-processeur ne distille pas l'information contenue dans le CL-File mais se contente d'en transformer le format et la syntaxe de manière à produire de l'information directement utilisable par la MOCN.

Un post-processeur particulier est défini en fonction du format de code G que nécessite la MOCN pour laquelle il à été créé. Afin de pouvoir traiter correctement les données reçues sous forme de CL-File, l'information dont doit disposer le post-processeur inclut, entre autres, le type de MOCN, la configuration de celle-ci ainsi que les particularités du contrôleur de cette machine. Le type de MOCN indique au post-processeur sur quelle machine sera effectué l'usinage, tel un tour, une fraiseuse ou encore une rectifieuse. C'est avec cette information qu'une interprétation correcte des informations contenues dans le CL-File est réalisée.

Le post-processeur doit également connaître la configuration de la machine. Ainsi, des informations concernant le nombre d'axes de déplacement ainsi que leur type, linéaire ou rotatif, doivent être fournies lors de la mise en place du module. Les limites de déplacement de chacun des axes doivent également être spécifiées, ainsi que l'orientation relative de chacun de ces derniers. Puisque chaque contrôleur possède des variantes au format de

code G utilisé, l'interprétation des codes G par la MOCN doit être fournie afin que le post-processeur puisse générer les codes requis par la machine, et ce, au format requis.

Puisque le post-processeur dépend des particularités de l'ensemble incluant la machine, sa configuration et son contrôleur, un post-processeur différent doit être créé pour chaque modèle et marque de MOCN. De plus, le format d'entrée de données aux contrôleurs de MOCN, soit le code G, dévie largement des standards définis par ISO, et ce, d'un fabricant de contrôleur à un autre et pour les différents modèles de contrôleur d'un même manufacturier. Ces écarts à la norme sont dûs au marché hautement compétitif qui existe dans ce domaine. Lorsque des innovations technologiques permettent l'implantation de nouvelles fonctionnalités à même les contrôleur de MOCN, elles sont incluses immédiatement et de nouveaux codes G sont créés en conséquence. Cet ajout est effectué sans attendre la normalisation de ces fonctionnalités, ce qui exigerait trop de temps. Ceci permet aux fabricants de conserver un avantage technologique sur leurs concurrents. Puisque chaque compagnie qui produit des contrôleurs agit de la sorte, il existe présentement une multitude de fonctionnalités présentes dans les contrôleurs de MOCN et régies de manière propriétaire.

Ces différents formats de code G nécessitent la création d'un post-processeur par modèle de MOCN que possède une compagnie, ce qui entraîne des coûts élevés de mise en place, sans compter l'entretien des post-processeurs requis lorsque la plate-forme FAO est mise à jour ou modifiée. De plus, il existe des formats propriétaires autres que le code G utilisé afin de contrôler les MOCN. Par exemple, les contrôleurs Heidenhain acceptent à la fois le code G, de manière limitée, et un langage conversationnel propriétaire à Heidenhain [3]. Il en est de même avec les MOCN Mazak dont le contrôleur peut interpréter, outre le code G, le langage conversationnel propriétaire Mazak, le Mazatrol [4].

Dans le cas où un module de simulation d'usinage est utilisé entre le module de post-processeur et celui de la MOCN, il doit utiliser l'information sous forme de code G afin d'illustrer les déplacements d'outils et accessoirement, ceux de la MOCN en entier.

Il importe de préciser que l'exécution du post-traitement ne requiert pas d'entrée de données de la part de l'utilisateur. On peut concevoir qu'il serait possible d'intégrer le post-processeur au contrôleur de la MOCN et d'envoyer directement les données à celle-ci sous forme de CL-File, afin d'éliminer entièrement l'étape de post-traitement.

1.1.5 Module de simulation d'usinage

Il existe principalement deux types de simulateur d'usinage. Le premier type (Type I) simule uniquement l'outil et la pièce. Il permet d'évaluer qualitativement l'efficacité en usinage des trajectoires générées en permettant une observation graphique des déplacements d'outils dans le brut et en présentant l'état de la pièce suite à chaque séquence d'usinage. Il permet également de détecter les collisions en avance rapide entre l'outil et le brut ou encore les brides. Un tel simulateur est souvent intégré aux logiciels de FAO afin de vérifier les trajectoires d'outils pendant leur création.

Les simulateurs du second type (Type II) offrent l'ensemble des possibilités des simulateurs de Type I en plus d'offrir une simulation complète des déplacements des composantes de la MOCN, ce qui permet de détecter les erreurs de dépassement de course des axes ou les collisions machine lors de l'usinage 5 axes ou autres.

Ces deux types de simulateur peuvent réaliser l'usinage virtuel à partir de deux types de données, soit le CL-File (ou son équivalent) et le code G. Il va sans dire que la simulation basée sur le CL-File ne permet pas une vérification complète car la MOCN utilise l'information sous forme de code G pour effectuer l'usinage, et non le CL-File. Il est ainsi possible que la simulation basée sur le CL-File ne présente pas d'erreurs alors que celle basée sur le code G en contienne. Il convient donc, lorsque possible, d'effectuer la simulation à l'aide des données qui seront directement envoyées à la MOCN.

La création du simulateur d'usinage pour chaque MOCN exige de définir une réplique virtuelle de la géométrie de la machine, de ses axes de déplacement et de leurs limites ainsi que du contrôleur de la MOCN. Toute cette information a toutefois déjà été utilisée afin de construire le post-processeur. Il s'agit donc d'une double entrée de données qui est inutile et peut mener à des erreurs résultant de la mise à jour des informations de la machine d'une manière différente dans le post et dans le simulateur.

1.1.6 Module de la MOCN

La vaste majorité des contrôleurs de MOCN en utilisation dans les industries et offerts par les fabricants sont basés sur un contrôleur à logique programmable (PLC). Ces contrôleurs sont caractérisés par une architecture relativement fermée, ce qui devient problématique lorsqu'ils doivent être intégrés à un système de fabrication complet. Habituellement, la principale information qui peut être transférée à ces contrôleurs est le programme d'usinage en format code G ou conversationnel. Afin d'offrir une meilleure vue d'ensemble des problèmes liés au transfert de données vers les contrôleurs de MOCN, nous présenterons en détail dans les sections qui suivent les fonctionnalités des MOCN ainsi que les interfaces permettant le transfert des programmes d'usinage au contrôleur.

1.1.6.1 Lien de donnés à la MOCN

Le lien de données entre le contrôleur du module MOCN et le reste du système de fabrication est assuré soit par un lien série sur port RS-232 ou encore par une interface Ethernet qui permet le transfert des programmes d'usinage, souvent par protocole FTP (File Transfer Protocol). Il existe encore des MOCN utilisées en entreprises qui utilisent des rubans perforés afin de permettre l'entrée des programmes d'usinage au contrôleur.

La majorité des MOCN utilisées actuellement en entreprise se servent d'un lien série RS-232 afin de transférer les programmes entre un ordinateur et le contrôleur. Selon la mémoire disponible à même le contrôleur de la MOCN à laquelle sera envoyée le programme, il est possible de l'emmagasiner en entier avant de débuter l'usinage. Dans le cas où l'espace de stockage n'est pas suffisant pour contenir le programme en entier et/ou le fichier contenant le programme est trop volumineux, celui-ci sera envoyé bloc par bloc au contrôleur pendant l'usinage, en mode DNC (Direct Numerical Control).

Cependant, la vitesse d'un lien RS-232 est relativement faible, et si le programme à transférer présente une combinaison de vitesses d'avance élevées et de segments de lignes de déplacement courts, il est possible que le lien ne puisse fournir à la machine l'information à la vitesse nécessaire lors de l'utilisation du mode DNC. Cet état de manque de données est appelé "data starvation" et résulte en un arrêt des déplacements de la machine jusqu'à ce que l'information nécessaire à la poursuite de l'usinage soit disponible. Une telle situation d'arrêt des déplacements va créer une marque d'outil sur la pièce car la déflection de l'outil causée par les efforts de coupe n'est plus présente lors de l'arrêt des mouvements.

Ainsi, pour ce type de transfert sur lien RS-232, il est courant en industrie de diminuer les vitesse d'avance et de tenter de diminuer au maximum la quantité d'information à transférer à la MOCN. Ces actions résultent en une diminution de la productivité et une hausse des coûts de production. Il est évident que les liens Ethernet sont une solution à ce problème, mais ceux-ci ne sont disponibles que sur les MOCN récentes et ne sont donc pas encore largement implantés. De plus, l'architecture fermée des PLC des contrôleurs de MOCN fait en sorte qu'il est souvent impossible d'adapter un lien Ethernet à une machine utilisant nativement un lien RS-232

1.1.6.2 Fonctionnalités des contrôleurs de MOCN

Mis à part certains fabricants de MOCN qui développent eux-même leurs contrôleurs, il est courant qu'un fabricant achète le contrôleur d'une compagnie se spécialisant dans ce domaine et l'adapte à sa propre MOCN. Les contrôleurs qui sont achetés de compagnies spécialisées présentent déjà des fonctionnalités de base et certaines fonctionnalités avancées.

Parmi les fonctionnalités de base, on compte les interpolateurs linéaires et circulaires qui permettront à la machine de déplacer ses axes selon les données présentes dans le programme de code G. Une MOCN ne peut déplacer l'outil de coupe que sur des segments linéaires commandés par un G01 et des segments circulaires commandés par G02 (interpolation horaire) et G03 (interpolation anti-horaire). Les interpolations linéaires peuvent généralement être exécutées dans l'espace 3D par un déplacement simultané et coordonné des axes cartésiens de la MOCN. Les interpolations circulaires ne peuvent être exécutés que dans les plans cartésiens XY, XZ et YZ du système de coordonnées cartésien de la machine. L'usinage de toute trajectoire qui ne peut être réalisée directement par ces types de déplacement est approximé par une suite de déplacements linéaires dans l'espace. Donc, plus la précision d'usinage exigée pour des surfaces complexes est élevée, plus le nombre de segments de lignes nécessaires à l'usinage de ces surfaces est important, ce qui résulte en un programme d'usinage volumineux, créant des problèmes de transfert sur les liens RS-232. Il existe donc un compromis entre la précision des trajectoires d'outils utilisées pour l'usinage des surfaces complexes 3D et la taille des fichiers d'usinage nécessaires.

Il existe toutefois une autre manière de spécifier les déplacements que doit réaliser une MOCN, mis à part les interpolations linéaires et circulaires. En effet, des cycles standardisés d'usinage sont également généralement intégrés aux contrôleurs. Par exemple, l'ensemble des cycles concernant les opérations de perçage sont standardisés et intégrés à pratiquement tous les contrôleurs de machine pouvant physiquement réaliser ce type

d'opération. Ces cycles respectent habituellement le standard ANSI NCITS 37 :1999 régissant le code G. Ceci permet entre autres d'utiliser ces cycles à partir du module FAO, ce qui n'est pas possible lorsque les cycles ne sont pas conformes au standard, car ils ne sont pas "connus" du logiciel FAO.

Du point de vue de la MOCN, ces cycles d'usinage ne sont que des macros qui utilisent des fonctions de déplacement et de commande en vitesse d'avance (G01, G02 et G03), de départ de broche (M03), de positionnement rapide (G00) et autres. Par exemple, un cycle de perçage avec bris de copeaux sur une fraiseuse va être commandé en règle générale par le code G73 suivi d'arguments définissant les caractéristiques du perçage. Lorsque le contrôleur traite ce code, il appelle la macro associée qui effectue les appels aux codes G et M (codes divers, Miscellaneous) définissant l'action à entreprendre.

En fait, la majorité des contrôleurs de MOCN offrent la possibilité d'effectuer une programmation paramétrique simple à même un fichier de code G. Des instructions de branchement simples tels IF ou GOTO et des variables sont disponibles, ce qui permet d'augmenter la flexibilité limitée du code G et ouvre la voie à l'écriture de macros selon les besoins.

Parmi les fonctionnalités avancées qui peuvent être présentes dans un contrôleur lors de son acquisition, nous pouvons noter l'interpolation NURBS (Non-Uniform Rational B-Spline), qui permet à la MOCN des déplacements d'outils selon une équation de courbe NURBS en plus des interpolations linéaires et circulaires. Toutefois, l'implantation des fonctionnalités NURBS n'est pas standardisée et la syntaxe ainsi que le format des codes G associés à ce type de déplacement varie de manufacturier en manufacturier [6] [7]. De plus, afin d'utiliser cette fonctionnalité, il faut posséder un module FAO capable de générer ce type de trajectoire d'outils ou encore un post-processeur capable d'interpoler une courbe NURBS à partir d'une suite de segments de ligne.

De plus, lorsqu'un fabricant de MOCN fait l'acquisition d'un contrôleur afin de l'incorporer à ses MOCN, il a la possibilité d'ajouter des fonctionnalités supplémentaires à même le contrôleur afin d'augmenter la flexibilité et les capacités de la machine. Le type d'éléments ajoutés aux contrôleurs inclus des macros d'usinage avancé qui permettent, par exemple, l'usinage d'une pochette circulaire en entier à l'aide d'un code G propriétaire. Des macros de mesure automatisée de pièce à l'aide d'une sonde font également partie des options souvent disponibles.

La majorité des fonctionnalités avancées ajoutées aux contrôleurs par les fabricants de MOCN et/ou de contrôleur ne sont pas standardisées et ne sont donc pas accessibles par l'entremise des système FAO. Le seul moyen alors disponible pour utiliser ces fonctionnalités consiste à ajouter les codes G associés de manière manuelle directement dans le fichier du programme d'usinage. Une telle approche est associée à un risque d'erreurs élevé. De plus, l'aspect propriétaire de ces cycles fait en sorte que les simulateurs d'usinage ne peuvent les traiter de manière native et un développement logiciel doit être réalisé afin de rendre l'émulation de ces cycles possible.

1.2 Recherche en cours et solutions

Maintenant que les problèmes de compatibilité entre les différents modules d'un système de fabrication intégré ont été présentés et que les détails du fonctionnement des principaux modules on été donnés, il est possible d'analyser les solutions courantes à ces problèmes d'intégration. Cette section présente les solutions disponibles commercialement afin de tenter d'augmenter le niveau d'intégration entre les différentes composantes du système ainsi que les recherches en cours ayant pour but d'améliorer la fluidité du transfert d'information lors de son transit entre le module CAO et la MOCN. Une analyse des différentes solutions est également présentée.

1.2.1 Transfert d'informations CAO – FAO

Tel qu'abordé aux sections 1.1.1 et 1.1.2, si les logiciels de CAO et de FAO ne sont pas des modules de la même plate-forme CFAO, un format neutre doit être employé afin de transférer la géométrie entre ces modules. La perte de toutes les caractéristiques de la pièce ainsi que des informations autres que géométriques résulte de ce transfert. De plus, il est impossible d'effectuer une mise à jour de la géométrie dans le module de FAO et de refléter les changements dans le module CAO.

Ceci a pour conséquence que l'information voyage unidirectionnellement du système de CAO au système de FAO. Des problèmes de mise à jour des données peuvent résulter d'une telle situation car des pièces ayant des géométries ou des dimensions différentes peuvent exister en parallèle si les changements ne sont pas reflétés à tous les modules. Il existe deux solutions à cette problématique, la première est d'acquérir un logiciel de CFAO intégré. Toutefois, cette méthode radicale peut résulter en la création d'autres problèmes. En effet, mis à part l'investissement élevé nécessaire à l'achat d'un tel logiciel, une formation doit être dispensée aux employés afin qu'ils puissent l'utiliser efficacement.

La seconde solution consiste à apporter les modifications nécessaires dans le module CAO, exporter à nouveau le modèle par l'entremise d'un format neutre, et générer une seconde fois les trajets d'outils sur la nouvelle pièce. Ce qui constitue également une solution inefficace car la perte de l'associativité liée à l'utilisation du format neutre exige de générer à nouveau les trajectoires.

1.2.1.1 Modèle STEP

L'ensemble des problèmes cités ci-dessus sont causés par la présence de formats propriétaires ainsi que par l'inhabileté des formats neutres de représenter tous les types d'entités

géométriques, la paramétrisation et les caractéristiques de la pièce présents au niveau des modules FAO. C'est afin de remédier à ces lacunes et plus spécifiquement aux limitations de IGES que l'organisation responsable de cette norme a développé STEP en tant que successeur à IGES [1]. Cette nouvelle norme régie par ISO 10303 est maintenant devenue un effort international visant à offrir, entre autres choses, un format de fichier informatique standardisé capable de représenter un produit et l'ensemble des données le décrivant, et ce, pendant son cycle de vie en entier.

Une différence fondamentale existe entre les formats neutre de représentation de géométrie tel IGES ou DXF (Drawing Exchange Format) et STEP. En effet, les formats standardisés offerts jusqu'ici ne sont que des spécifications de formats de fichiers d'échange de géométrie. Pour sa part, STEP propose une architecture complète de représentation d'information d'un produit incluant un format neutre d'échange [1]. Ce dernier format permet, entre autres choses, de contenir les informations représentant la géométrie et la topologie des pièces, la paramétrisation, les caractéristiques ainsi que les contraintes d'assemblages, pour ne nommer que celles-ci.

En fait, STEP peut être défini comme un standard interprétable des caractéristiques physiques et fonctionnelles d'un produit durant son cycle de vie entier, incluant le design, la fabrication, l'utilisation, la maintenance et la mise au rebut . Il permet entre autres d'échanger de manière transparente les données entre les différents environnements d'un système intégré de conception et de fabrication. STEP est composé d'une série d'items publiés séparément et utilise le langage EXPRESS afin de décrire les informations du produit représenté [8], [9].

Puisque STEP représente un standard complexe, il a été divisé en différentes sections, appelées protocoles d'applications "Application Protocol" (AP) qui visent différentes parties du cycle de vie d'un produit. Ainsi, AP202 couvre le dessin associatif (associative draughting), AP203 la configuration contrôlée du design 3D des pièces et assemblages mécaniques et AP213 le contrôle de procédé pour l'usinage de pièce par commande nu-

mérique. Il existe présentement environ 30 protocoles d'application actifs. Chacun de ces protocoles d'application contient plusieurs classes de conformité qui sont des sous éléments d'un protocole qui peuvent être implantés dans une application et être fonctionnels sans que l'ensemble des classes de conformité formant un protocole d'application ne soient mises en place.

Ainsi, STEP compte parmi ses objectifs de résoudre la problématique du transfert de données entre les différents modules d'un système de fabrication, et ce, peu importe le fabricant des logiciels utilisés pour chacun des modules. De plus, il permet un échange de données bi-directionnel entre les modules de CAO et de FAO.

Bien que ce format est extrêmement attrayant comparativement à l'utilisation des autres formats standardisés et même, jusqu'à un certain niveau, à l'emploi des formats propriétaires des logiciels de CAO et de FAO. Toutefois, son implantation est relativement lente. La majorité des modules de CAO et de FAO offrent présentement des fonctionnalités d'importation et d'exportation au format STEP, mais les parties de la norme qui sont définies ne sont pas complètement implantées, et souvent, seule la géométrie est transférée [8]. Bien que plusieurs systèmes CAO, FAO et CFAO indiquent posséder une conformité STEP, elle n'inclut que quelques protocoles d'application, et pour chacun d'eux, seuls quelques classes de conformité sont implantées, ce qui en réduit les fonctionnalités.

La lenteur de l'implantation est principalement due à 2 facteurs. Tout d'abord, STEP est une collaboration entre près de 200 comités internationaux d'utilisateurs et autant de comités d'industries privées. Compte tenu du nombre élevé de partis impliqués et de leur dispersion géographique, l'établissement de la norme en est ralenti. De plus, il faut obtenir l'approbation des comités lors de l'établissement des détails de la norme.

Puisque STEP propose de permettre un échange de données presque transparent entre différents modules d'un système de fabrication, et ce, entre des logiciels de différents fabricants, il y a un risque économique ressenti par les concepteurs de logiciels. En effet, il

est présentement courant qu'une compagnie qui reçoit un important contrat de conception ou de fabrication d'une multinationale fasse l'acquisition du même logiciel CFAO que celle-ci, afin de permettre l'échange de données. Le format STEP permettrait de travailler en collaboration sans avoir à changer de plate-forme.

En fait, l'implantation de STEP est tellement lente qu'un consortium international de gouvernement et d'industries, "PDES, Inc.", à été fondé de manière à accélérer le développement et l'implantation de ce standard [10]. Parmi les industries participant à ce groupe, on note plusieurs multinationales dont Boeing, la NASA, General Motors, Ford et Lockheed Martin.

1.2.1.2 Accès au modeleur solide

Il existe principalement six modeleurs solides qui sont utilisés au coeur des logiciels CAO et FAO afin de représenter la géométrie solide de la pièce et de permettre d'effectuer des actions sur cette géométrie.

Ces modeleurs, présentés à la table I peuvent être offerts aux développeurs de systèmes CAO et FAO, comme c'est le cas pour ACIS et Parasolid, qui constituent les modeleurs les plus utilisés. Les quatre autres modeleurs présentés sont propriétaires et inaccessibles aux développeurs externes de modules CAO et FAO. Toutefois, la firme PTC offre depuis peu aux développeurs externes d'acquérir les droits d'utilisation de leur modeleur Granit One afin de permettre l'utilisation du format natif de Pro/Engineer et Pro/Desktop par d'autres logiciels, afin d'améliorer le niveau d'intégration avec d'autres modules d'un système de fabrication intégré [11].

Cette ouverture de PTC face à l'utilisation de leur modeleur par des developpeurs externes, ainsi que leur implication dans la mise en place de la norme STEP démontre un changement de mentalité des compagnies qui développent les systèmes CFAO. Une tendance à

Tableau I

Modeleurs solide

Compagnie	Modeleur	Remarques
Spatial Technology, Inc.	ACIS	Utilisé par de nombreux logiciels CAO et FAO tel AutoCAD, Mechanical Desktop, Design Studio et autres.
UGSolution	Parasolid	Utilisé par de nombreux logiciels CAO et FAO tel Unigraphics, Solid Edge, SolidWorks et autres.
Co-Create	Modeleur propriétaire	Modeleur utilisé par SolidDesigner.
Dassault	Modeleur propriétaire	Modeleur utilisé par CATIA.
PTC	Modeleur propriétaire	Modeleur utilisé par Pro/ENGINEER.
SDRC	Modeleur propriétaire	Modeleur utilisé par I-DEAS.

délaisser lentement les interfaces propriétaires en faveur d'une meilleure intégration des modules utilisés dans la conception et la fabrication des pièces mécaniques prend place.

Le logiciel de FAO EdgeCAM de la compagnie Pathtrace utilise le modeleur Granit One de manière à pouvoir utiliser nativement les fichier du système CAO de Pro/Engineer afin de générer des trajectoires d'outils qui sont associées à la géométrie et aux caractéristiques de la pièce. Cette amélioration de l'intégration est toutefois limitative lorsque comparée à STEP car Granit One est développé et maintenu par PTC, comparativement à STEP qui est développé de manière internationale.

1.2.2 Module FAO

Puisque les implantations de STEP présentement disponibles à même les modules CAO et FAO sont limitées au transfert d'information géométrique et topologique, tout modèle

transféré au module FAO par un format autre que le format natif du logiciel n'est qu'un ensemble de fils de fer, de surfaces et de solides, selon le format neutre utilisé pour le transfert. Ceci pose un sérieux inconvénient au niveau FAO car toutes les caractéristiques (features) composant la pièce sont perdues. Il devient alors impossible, par exemple, de spécifier l'élément géométrique à usiner par un cycle de poche en choisissant celle-ci, il faut alors sélectionner l'ensemble des surfaces constituant la poche. Il en résulte une perte de performance et de flexibilité d'utilisation du modèle associé.

1.2.2.1 Extraction de caractéristiques

C'est afin de résoudre cette lacune qu'un important effort de recherche est orienté vers la reconnaissance des caractéristiques géométriques contenues dans un modèle transféré par l'entremise d'un format neutre. Les algorithmes de reconnaissance ont pour objectif de reconstituer la fonctionnalité de la pièce à partir d'un ensemble de surfaces la modélisant. Des stratégies d'usinage peuvent ensuite être liées aux différentes caractéristiques qui ont été extraites.

Il existe trois types d'extraction de caractéristiques applicables à un modèle obtenu d'un logiciel CAO. L'extraction supervisée laisse l'utilisateur du système définir manuellement les caractéristiques que comporte la pièce. L'extraction automatique utilise des algorithmes qui vont permettre d'extraire les caractéristiques à partir du modèle CAO de la pièce. Le design par caractéristique permet de définir le modèle CAO par l'entremise de caractéristiques qui sont alors inhérentes au modèle.

Un système d'extraction de caractéristiques pour les pièces cylindriques à été élaboré par Chandrasekaran [12] afin d'effectuer l'usinage de pièces par tournage. Ce système, relativement restreint, présente également un module de CAO où le profil 2D de la pièce est créé. Cette information géométrique est ensuite transférée au module d'extraction de ca-

ractéristiques proposé , ce qui permet à l'utilisateur de définir manuellement la géométrie qui constitue chacune des caractéristiques d'usinage parmi les types suivants : alésage, trou, retraits, chariotage et surfaçage.

L'extraction de caractéristiques par le logiciel consiste alors, suite à une définition manuelle de ces caractéristiques par l'utilisateur, à enlever de la pièce brute les caractéristiques afin de vérifier si la pièce peut être usinée à l'aide de l'ensemble d'éléments définis par l'utilisateur. Dans le cas où la pièce peut correctement être usinée à l'aide des caractéristiques définies manuellement et analysées par le logiciel, elles sont sauvegardées dans un modèle séparé entièrement constitué des caractéristiques qui ont été ainsi extraites. Le logiciel propose alors l'ensemble des stratégies d'usinage qui, appliquées aux caractéristiques qui ont été extraites, permettent d'usiner la pièce. L'utilisateur peut alors choisir la stratégie la plus efficace afin de produire la pièce.

Les applications de ce système sont relativement limitées du fait que l'extraction des caractéristiques est effectuée de manière manuelle. Une telle méthode d'extraction des caractéristiques ne présente pas d'avantage majeur sur la sélection manuelle des surfaces à usiner, telle qu'effectuée lors de la création d'une séquence d'usinage régulière à l'aide d'un modèle transféré au module FAO par l'entremise d'un format neutre. De plus, le système est limité au traitement de géométrie 2D pour les applications de tournage.

Un modèle d'usinage est souvent considéré comme un modèle basé sur des caractéristiques d'usinage . Pour une seule pièce, il peut donc exister plusieurs modèles de caractéristiques qui peuvent permettre l'usinage de celle-ci. Gupta et Nau [13] ont élaboré une méthodologie qui permet de générer l'ensemble des modèles de caractéristiques alternatifs pour une pièce à partir d'un modèle de caractéristiques de base. Pour chaque modèle, des règles de précédence sont générées à partir des interactions entre les caractéristiques, puis le modèle de caractéristiques obtenu est évalué afin de déterminer son usinabilité. Les alternatives possibles qui demeurent peuvent alors être évaluées par un utilisateur qui

choisit celle qui est la plus efficace selon des critères relatifs à la pièce et à l'équipement disponible afin de réaliser l'usinage.

De même, Regli, Gupta et Dana [14] proposent un système d'extraction de caractéristiques d'usinage selon une librairie pré-établie qui contient des éléments de perçage, de fraisage, de chanfreins et de rayons. L'algorithme d'extraction développé permet d'extraire avec succès les caractéristiques appartenant à la librairie définie, et ce, même si les caractéristiques s'intersectent. Les différentes stratégies d'usinage applicables aux caractéristiques extraites sont ensuite présentés à l'utilisateur. Afin de rendre ce système d'extraction accessible à toute la communauté, les caractéristiques extraites sont exprimables en format MRSEV, une librairie de caractéristiques d'usinage basée sur PDES/STEP. Un modèle fonctionnel de leur travail a été implanté dans le système IMACS (Interactive Manufacturability Analysis and Critiquing System) de l'Université de Maryland.

Une multitude d'autres travaux de recherche ont été générés afin de réaliser des algorithmes d'extraction plus performants et flexibles, ainsi que des systèmes de génération de modèles basés sur les caractéristiques alternatifs mieux adaptés à la réalité de l'usinage. Ils ne seront toutefois pas mentionnés ici car ils sont hors du contexte de la présente recherche. Il est toutefois bon de remarquer qu'une partie des systèmes d'extraction va perdre de son attrait si la norme STEP est un jour complètement implantée.

1.2.2.2 STEP-NC

Bien que la norme STEP soit attrayante, elle serait de peu d'utilité à la fabrication des pièces mécaniques si elle ne pouvait contenir les trajectoires d'outils ou les informations permettant de les générer. Ainsi, STEP comprend parmi ses nombreux protocoles d'application, un élément appelé STEP-NC et défini par ISO 14649. Cette extension de la norme STEP permet d'inclure dans le modèle les informations permettant de générer les trajec-

toires d'outils, soit les paramètres d'usinage, l'outil à utiliser ainsi que la géométrie qui doit être usinée avec celui-ci.

La norme STEP-NC (AP238) actuelle ne couvre que les opérations de fraisage, de tournage et de découpage au fil par électro-érosion. Des expériences de mise en place de ce standard sont présentement en cours afin de démontrer les possibilités de ce standard. Le but ultime de STEP-NC consiste à transférer un modèle STEP du système CAO au système FAO. Ce modèle sera alors enrichi des informations d'usinage STEP-NC et les données résultantes seront envoyées à une MOCN dont le contrôleur peut lire nativement le format STEP-NC et réaliser l'usinage en calculant les trajectoires d'outils directement à partir des informations contenues dans le fichier STEP/STEP-NC [15].

Un projet de trois ans appelé "Super Model Project" est présentement en cours et a pour but de démontrer la flexibilité et la puissance qu'il est possible de retirer de l'utilisation de la combinaison STEP/STEP-NC dans un processus de conception et de fabrication intégrées [16].

L'utilisation de STEP-NC permet d'éliminer le post-processeur car les données générées par le système FAO ne seraient plus conservées sous un format APT mais bien par un "super modèle" STEP contenant l'ensemble des informations du produit à réaliser, incluant la géométrie de la pièce, ses propriétés, ses caractéristiques ainsi que les données d'usinage. Eventuellement, ces donnés pourraient être transférées directement à une MOCN capable de les interpréter, ce qui rend toutefois désuet tous les contrôleurs existant.

L'utilisation d'un nouveau type de post-processeur capable de traiter le STEP-NC afin d'en tirer le code G nécessaire à l'utilisation des contrôleurs existant sera alors requis. La désuétude des MOCN causée par STEP-NC diminue considérablement l'intérêt porté à ce standard par bon nombre de compagnies manufacturières.

1.2.3 Transfert d'informations FAO – Post-processeur

L'information transmise au Post-Processeur par le système FAO, sous format propriétaire ou de CL-File, est constitué d'une information de piètre qualité. En effet, le système FAO qui possède l'ensemble de l'information géométrique et topologique de la pièce à usiner ainsi que les paramètres associés à l'usinage ne transmet au module de post-processeur que la position des points par lesquels devra passer la pointe de l'outil pendant l'usinage.

1.2.3.1 STEP-NC

Une des solutions présentement en développement afin d'éviter cette dégradation de la qualité de l'information est la norme STEP-NC. Le transfert entre le système de FAO et le post-processeur destiné à générer l'information d'usinage pour une MOCN avec un contrôleur basé sur le code G permettrait de conserver toute l'information jusqu'à l'étape du post-traitement. Ceci permettrait au post-processeur d'agir plus intelligemment en se basant sur l'information de haute qualité contenue dans STEP-NC. Ainsi, une meilleure exploitation des capacités des MOCN serait rendue possible, malgré les limitations inhérentes au code G qui date de près de 50 ans.

1.2.3.2 BCL

Une seconde solution au transfert d'information entre le module FAO et le post-processeur consiste à utiliser le langage BCL (Basic Control Language) normalisé par EIA/ANSI 494-C. Ce langage présente une syntaxe essentiellement semblable à celle du langage utilisé dans les CL-Files, soit un sous ensemble des commandes APT [17]. Son développement a débuté en 1975 alors que Rockwell désirait pouvoir utiliser les CL-Files générés par les

systèmes de FAO directement avec les MOCN, sans utiliser de post-processeur. Le langage BCL est donc directement dérivé du CL-File mais a été adapté de manière à pouvoir être interprété directement par des contrôleurs de MOCN spécifiques. De plus, le BCL est complètement indépendant du système avec lequel il est généré (mainframe, Unix, Windows, etc...), ce qui n'est pas le cas du CL-File [18].

La syntaxe du BCL est plus stricte et la sémantique est plus rigoureuse que celle du APT et des CL-Files. Ainsi, le BCL vise à éliminer les écarts à la norme APT qui s'étaient lentement établis dans l'industrie entre les différentes plates-formes de FAO qui utilisent le CL-File comme format de sortie.

Dans une implantation complète du langage BCL (Native BCL Control), le code BCL généré par le système de FAO est envoyé directement à un contrôleur de MOCN qui peut lire nativement ces données afin d'effectuer l'usinage. Ce mode d'utilisation du BCL permet d'éliminer l'utilisation d'un post-processeur tout en permettant d'utiliser le programme d'usinage ainsi généré sur n'importe quelle MOCN sans avoir à le convertir en un format spécifique à cette machine.

Toutefois, très peu de logiciels de FAO peuvent générer du BCL nativement et un nombre également très restreint de MOCN possèdent des contrôleurs capables d'interpréter le BCL. Afin de survenir à ces problèmes, un convertisseur qui permet de générer du BCL à partir d'un CL-File provenant d'un système FAO doit être utilisé. Puis le BCL doit être post-traité en code G, ou plus précisément en MCD (Machine Control Data) afin de pouvoir être interprété par une MOCN conventionnelle. Le post-traitement qui permet de passer du format BCL au code G peut être effectué sur la station qui sert à générer les trajectoires d'outils et on parle alors de "Basic System Implementation". Si le post-traitement BCL-code G est effectué en temps réel par une station située près de la MOCN, on parle alors de "Front End Control" [19].

Il est évident que toute implantation du BCL hors du "Native BCL Control" présente le désavantage de nécessiter deux étapes de post-traitement, soit CL-File – BCL puis BCL – Code G. Ceci limite fortement les avantages de l'utilisation du BCL et ne peut être justifié que si la majorité des MOCN d'un site de production utilisent une implantation native du langage.

1.2.4 Module Post-processeur

L'existence du module de post-processeur n'est due qu'aux différences entre les formats utilisés à la sortie des systèmes FAO et à l'entrée des contrôleurs de MOCN. Le BCL en implantation native et le format STEP-NC permettent d'effectuer l'usinage sans avoir recours au module de post-processeur qui est voué à disparaître dans quelques décennies. Ceci explique que peu de recherche est effectuée afin d'améliorer l'intégration du post-processeur au reste du système intégré de fabrication.

Il existe de nombreux problèmes liés à l'utilisation des post-processeurs et des générateurs de post-porcesseurs avec les MOCN sophistiquées. En effet, depuis quelques années, des tours à commande numérique comportant une broche de fraisage montée au porte-outil et capable de réaliser des déplacements selon trois axes cartésiens et un axe rotatif sont maintenant disponibles. Des tours comportant deux tourelles portes-outils et deux broches opposées qui permettent de l'usinage en simultané et le transfert de la pièce d'une broche à l'autre ont également fait leur apparition.

La majorité des modules FAO ne peuvent générer en une seule opération les CL-Files nécessaires à l'utilisation optimale de l'ensemble des nouvelles stratégies d'usinage offertes par de telles machines. De plus, lorsque les CL-Files sont obtenus, la majorité des post-processeurs intégrés aux modules FAO ne peuvent prendre en charge de telles configurations de MOCN. Les solutions de post-traitement qui supportent de telles configurations

de machine le font d'une manière souvent sommaire et peuvent présenter certains problèmes lorsque la vitesse d'avance de l'outil dans la pièce doit être conservée constante pendant un déplacement utilisant un ou plusieurs axes rotatifs en simultané [20].

De plus, les post-processeurs n'offrent aucun support natif permettant d'utiliser les fonctionnalités avancées disponibles sur plusieurs MOCN, tels des cycles d'usinage de géométrie prismatique et des cycles de mesure de pièce. Certains environnements de développement de post-processeurs offrent un langage de programmation de base permettant de mettre en place un support pour ces routines avancées.

1.2.4.1 Décentralisation

Ainsi, Bruce [21] propose de décentraliser le post-traitement afin de le réaliser dans l'usine, sur une station reliée directement à la MOCN qui effectue l'usinage. Ceci permet d'augmenter la flexibilité du système d'usinage, car un programme qui est post-traité dans les bureaux d'ingénierie est destiné à une machine en particulier. Des changements dans la production peuvent nécessiter de déplacer l'usinage sur une autre MOCN. Le programme doit alors être retourné à l'ingénierie pour être post-traité à nouveau pour la nouvelle machine, ce qui augmente le temps mort dans le système de production. Les avantages à utiliser le BCL comparativement au CL-File de manière à augmenter la portabilité de l'information d'usinage sont également abordés.

1.2.4.2 Intégration

Blumfield, Shpitalni et Lenz [23] ont proposé un générateur utilisé pour créer des post-processeurs adaptatifs. Ce type de post processeur dispose, en plus de l'information sur les trajectoires d'outils contenues dans le CL-File, d'un accès à différentes bases de don-

nées contenant des informations supplémentaires. La géométrie de la pièce est accessible par l'entremise du modeleur du système de CAO, des tables technologiques contenant des données d'usinage sont également accessibles, de même que des informations sur les fonctionnalités avancées de la MOCN.

La disponibilité de l'information supplémentaire accessible par le post-processeur adaptatif lui permet de prendre de meilleures décisions qui résultent en un code G plus performant. Le post-processeur permet entre autre d'utiliser des cycles d'usinage avancé disponibles à même la MOCN. Toutefois, cette fonctionnalité est limitée car elle est basée sur la reconnaissance automatique de caractéristiques effectuée par le post-processeur en extrayant l'information géométrique du modeleur CAO, sans intervention humaine. Aussi, l'implantation proposée se limite des applications de découpage au laser sur 2 axes.

1.2.4.3 Élimination du post-processeur

Mis à part l'utilisation du BCL et du STEP-NC, plusieurs autres projets de recherche ont pour but principal ou secondaire d'éviter la présence d'un module de post-processeur dans le système intégré de fabrication. Suh, Noh et Choi [22] ont réalisé une intégration entre les modules CAO, FAO et MOCN d'un système de production. Le modeleur du système de CAO ainsi que le module de FAO ont été programmés de manière à utiliser la même base de données, ce qui élimine les problèmes d'intégration entre ces modules. Les informations d'usinage générées par le module FAO utilisent le format du CL-File, qui est directement envoyé au contrôleur à architecture ouverte de la MOCN, développé à partir d'un ordinateur personnel. Ainsi, le module de post-processeur est complètement éliminé du procédé de fabrication.

De même, Wright, Schofield et Wang [24] traitent de l'implantation d'un contrôleur de MOCN à architecture ouverte basé sur une plate-forme UNIX en temps réel qui possède

un interpolateur capable de traiter directement les courbes Spline. Le système MOSAIC (Machine-tool Open System Advanced Intelligent Controller) proposé permet d'éliminer entièrement le module post-processeur et effectue l'interpolation des axes en temps réel à partir des informations géométriques obtenues du module CAO ainsi que des paramètres d'usinage fournis. De plus, le système offre la possibilité d'intégrer des capteurs de force, de vision et autres de manière à offrir un usinage adaptatif.

Bien qu'intéressant, le projet MOSAIC, une extension au projet conjoint IMW (Intelligent Machining Workstation) de la U.S. Air Force, de Cincinnati Milacron et de l'université Carnegie Mellon n'as pas débouché sur des applications industrielles.

De même, il existe maintenant plusieurs contrôleurs de MOCN basés sur un PC disponible. Ces derniers permettent le contrôle des axes de la MOCN directement à partir du modèle de la pièce et des informations d'usinage éliminant du même coup la nécessité d'un module de post-processeur dans la chaîne de fabrication.

1.2.5 Transfert d'informations Post-processeur – MOCN

L'information qui doit être transférée à une MOCN est habituellement sous la forme de code G. Tel que mentionné à la section 1.1.6.1, l'emploi d'un lien RS-232 est relativement lent et peut causer des problèmes lors de l'usinage en mode DNC. Toutefois, l'interface RS-232 est utilisée sur la vaste majorité des MOCN présentement en utilisation aujourd'hui, et l'architecture fermée de ces MOCN ne permet pas d'utiliser un autre type de méthode de transfert. La seule solution possible consiste à remplacer le contrôleur de la MOCN en entier par un contrôleur plus récent qui supporte un lien Ethernet avec le réseau de la compagnie, ce qui est peu viable économiquement.

Ainsi, il est nécessaire d'optimiser l'utilisation du lien RS-232 en n'envoyant que les données nécessaires à la MOCN. Bien que les arcs de cercle, commandés par G02 et G03

puissent être exécutés dans les trois plans cartésiens d'une MOCN, certains systèmes de FAO ne les génèrent que dans le plan perpendiculaire à la broche. Pour les deux autres plans, les interpolations circulaires seront approximées par une série de déplacements linéaires.

Cet état de fait, en plus de résulter en un usinage plus saccadé, nécessite de transférer plus d'information par le lien entre l'ordinateur hôte et le contrôleur. Un arc de cercle peut nécessiter des dizaines de segments de ligne afin d'être approximé correctement. Ainsi, certains post-processeurs vont tenter de faire passer des segments d'arcs par les séries de segments de ligne afin de générer des G02 et G03 au lieu d'une succession de G01, tentant ainsi de recréer de l'information qui a été éliminée par le système FAO lors de la génération du CL-File à l'aide d'une algorithme de "arc fitting".

De plus, certains logiciels, axés vers l'optimisation des fichiers de code G, vont réaliser du "arc fitting" en plus d'éliminer toute information superflue contenue dans le fichier, dans le but d'accélérer le transfert par lien RS-232. Ainsi, les décimales inutiles seront tronquées des valeurs numériques et les espaces seront éliminés du ficher, diminuant sa lisibilité mais augmentant la vitesse de transfert. Le code "G01 X1.2000" deviendrait alors "G1X1.2". Ceci permet d'éliminer en partie les problèmes de "data starvation" et d'augmenter les vitesses d'avance utilisées pendant l'usinage.

Les machines plus modernes utilisant un lien Ethernet permettent des vitesses de transfert suffisamment élevées qui peuvent permettre un transfert en temps réel de l'information pendant l'usinage sans danger d'interruption du flot de données. De plus, les nouvelles machines comportent souvent un disque dur d'une capacité dépassant le giga-byte, qui permet de stocker tout les programmes d'usinage nécessaires à la réalisation d'une pièce complexe.

1.2.6 Module simulateur d'usinage

Les coûts engendrés par une éventuelle erreur peuvent être très élevés. C'est pourquoi l'utilisation de simulateurs d'usinage devient de plus en plus populaire. La majorité des modules FAO offrent un simulateur d'usinage intégré, habituellement de type I, qui permet de visualiser l'enlèvement de matière de manière virtuelle avant que le programme d'usinage ne soit acheminé à la MOCN.

Toutefois, un tel type de simulateur ne permet souvent pas de détecter les collisions avec le porte outil et le montage d'usinage. Même si une telle simulation est incluse dans un simulateur de type I, la visualisation de l'ensemble de la MOCN est absente. De même, une collision entre le broche et la pièce ou les éléments de montage n'est pas détectable. La hausse importante des vitesses de calcul des processeurs d'ordinateur et de cartes graphiques pousse de plus en plus de compagnies à employer des simulateurs d'usinage de type II qui permettent, outre les possibilités du type I, de simuler la machine en entier et d'effectuer une simulation d'usinage virtuelle complète.

Les principales recherches en cours sur les simulateurs d'usinage portent sur les algorithmes permettant d'obtenir le modèle après usinage à partir du fichier de code G. Plusieurs méthodes de simulations sont présentées dans la littérature, et la plus utilisée est celle de la coupe de vecteurs. Ainsi, Chappel [25] a développé cette méthode qui crée sur les surfaces du modèle un ensemble de vecteurs qui sont coupés par l'outil lors de la simulation. Cette méthode permet de calculer directement les "under-cuts" et "over-cuts" à partir de la longueur finale des vecteurs. Toutefois, elle ne génère pas de modèle solide de la pièce obtenue par simulation, ce qui empêche d'obtenir une intégration entre le module de simulation d'usinage et le module CAO pour fins de vérification. La majorité des simulateurs d'usinage de type I présentent cette limitation.

Afin de combler cette lacune de la méthode de coupe de vecteurs, une méthode de triangulation à été appliquée à la coupe de vecteurs. Des surfaces triangulées sont utilisées afin de représenter la pièce, à la manière d'un modèle STL, et les vecteurs sont créés sur ces faces triangulaires. Lorsque les vecteurs sont coupés par le passage de l'outil, de nouvelles faces triangulaires sont générées à la pointe des vecteurs afin de remplacer les anciennes facettes. Cette méthode permet d'obtenir une représentation des frontières de la pièce "Boundary Representation" (B-Rep), qui permet l'intégration au module CAO pour vérification. Toutefois, l'aspect surfacique triangulé de la pièce obtenue présente des inconvénients face à la précision de la pièce, inhérente à ce type de représentation.

Afin d'améliorer la précision de la pièce obtenue suite à l'usinage, des méthodes de simulation par géométrie solide constructive "Constructive Solid Geometry" (CSG) ont été proposées. Wang [26]présente cette méthode qui crée les volume à soustraire de la pièce brute pour chaque mouvement de l'outil dicté par le programme d'usinage. Le solide obtenu est de haute précision et peut être transféré directement au modeleur solide du système CAO. Toutefois, cette approche est extrèmement exigeante au niveau des calculs, et ce, particulièrement pour des surfaces complexes 3D. Ce type de simulation d'usinage peut, dans certains cas, être plus long que l'usinage sur la MOCN.

Ainsi, Roy et Xu [27] ont élaboré des algorithmes de simulation d'usinage basés sur un modèle de type octree étendu "extended octree" avec représentation B-Rep afin de pouvoir obtenir la pièce usinée à partir du programme d'usinage, de la géométrie des outils de coupe et de celle de la pièce brute. La méthode octree consiste à discrétiser le modèle à l'aide de cubes initiaux. Les cubes complètement à l'intérieur ou à l'extérieur du modèle sont conservés et ceux qui sont sur la frontière sont recoupés en huit. La discrétisation continue jusqu'à ce qu'un niveau souhaité de précision soit atteint. Afin d'augmenter d'avantage la précision, le modèle octree avancé permet d'inclure de l'information concernant la surface de la pièce qui traverse les cubes situés sur la frontière. Suite à la simulation, un modèle B-Rep précis est obtenu à l'aide d'un algorithme de conversion

à partir du modèle étendu octree. Ceci permet d'intégrer le modèle final directement au module FAO.

1.2.7 Module MOCN

Mis à part le module de CAO, tous les modules d'un système intégré de fabrication n'ont pour but principal que de produire l'information que nécessite la MOCN afin de produire la pièce. Il existe un volume important de recherches portant sur l'intégration des MOCN aux autres modules du système de fabrication. Le BCL présenté à la section 1.2.3.2 et le STEP-NC introduit à la section 1.2.1.1 proposent d'augmenter le niveau d'intégration avec la MOCN en éliminant le post-processeur et en transférant l'information d'usinage directement au contrôleur de la MOCN. Toutefois, ces solutions exigent une architecture de contrôleur complètement différente à celle implantée présentement, ce qui diminue fortement les avantages de telles solutions.

En effet, l'investissement initial nécessaire à l'acquisition d'une MOCN est élevé et inclut, mis à part le coût de la MOCN elle-même, celui de l'infrastructure, de la formation des employés, de l'intégration au système de fabrication de l'usine et de l'outillage. Puisque la durée de vie moyenne d'une MOCN varie de 5 à 30 ans selon l'application de la machine, un remplacement complet du parc de machine d'une compagnie n'est pas envisageable. Les solutions radicales du type BCL et STEP-NC ne seront donc implantées qu'à très long terme.

Les limitations à l'augmentation du niveau d'intégration entre le module MOCN et les autres modules d'un système de fabrication sont principalement dues à trois facteurs. Tout d'abord, les contrôleurs de MOCN ne peuvent traiter que des interpolations linéaires et circulaires. Récemment, des interpolateurs NURBS qui utilisent des codes G propriétaires ont été introduits.

La seconde limitation concerne le langage utilisé pour transférer l'information à la MOCN, soit le code G, qui est maintenant désuet et ne permet qu'une représentation ponctuelle des points par lesquels l'outil devra se déplacer afin de réaliser l'usinage. Cette information appauvrie ne permet pas d'exploiter l'ensemble des capacités qui seraient disponibles à même les contrôleurs de MOCN.

Finalement, la méthode dont est obtenue le programme d'usinage constitue en elle-même une importante limitation. Lors de la génération des trajectoire d'outils dans le module de FAO, une représentation paramétrique précise des surfaces à usiner est disponible par l'entremise du modèle de CAO. Puisque le CL-File à obtenir, qui doit éventuellement être transformé en code G, ne peut contenir que des déplacements linéaires ou circulaires, une approximation des trajectoires d'outils est obtenue à l'aide de ce type d'éléments. Une dégradation importante de la qualité de l'information est ainsi créée. De plus, un compromis doit être effectué entre la précision des trajectoires d'outils et la taille du programme d'usinage résultant.

1.2.7.1 Interpolateurs

Lors de l'usinage de surfaces complexes, la majorité du programme d'usinage est constitué de déplacements linéaires. Puisque ces segments ne sont pas tangents les uns aux autres, lors du passage d'un segment au suivant, la MOCN subit des accélérations brutales afin de tenter de suivre le profil approximé commandé. L'augmentation des vitesses d'avance accentue la déviation que subit l'outil par rapport à la trajectoire souhaitée.

Afin de permettre des vitesses d'avance élevées sur ces types de déplacements, un effort de recherche a été déployé de la part des fabricants afin d'augmenter la vitesse à laquelle leurs contrôleurs peuvent traiter l'information contenue dans le programme d'usinage. Des algorithmes de "look-ahead" qui ajustent la vitesse et l'accélération des axes de la machine

de manière à assurer un déplacement en douceur à haute vitesse sur des segments de lignes non-tangents ont été développés.

Toutefois, cette approche n'aborde pas la cause du problème, soit les limitations des interpolateurs et de l'information rendue disponible à la MOCN pendant l'usinage. Certains contrôleurs présentent des interpolateurs NURBS et cubique qui utilisent un format propriétaire de code G afin de permettre un usinage plus précis et en douceur à haute vitesse pour les surfaces complexes. Toutefois, peu de modules de CAO peuvent générer du NURBS au niveau du CL-File, et ceux qui le font appliquent un algorithme de "NURBS fitting" à un trajet d'outil qui a d'abord été discrétisé en segments de lignes et d'arcs de cercle, ce qui diminue fortement les avantage de l'utilisation des NURBS qui sont ainsi obtenues.

Ainsi, Ippolito, Iuliano et Vezzetti [28] ont proposés un algorithme qui traite les information appauvries contenues dans un CL-File de manière à y adapter des courbes de Bézier, une forme de courbe NURBS. Les courbes ainsi obtenues sont transformées en code G au format propriétaire du contrôleur d'une MOCN possédant un interpolateur NURBS. La méthode employée est fortement limitée car les NURBS utilisées sont reconstruites à partir d'un CL-File régulier contenant seulement des interpolations linéaires et circulaires, n'adressant pas la problématique initiale.

Afin de permettre une amélioration de la qualité de l'information transmise à la MOCN, Farouki, Manjunathaiah et Yuan [29] ont proposé une méthode permettant d'obtenir des trajectoires d'outils sous forme de courbe "Pythagorean-hodograph" dérivées directement à partir de la définition mathématique du modèle disponible dans le module de FAO. Bien que le type d'interpolation proposée résulte en des pièces de plus haute précision et présentant un meilleur fini de surface, l'utilisation des trajectoires obtenues exige un contrôleur possédant un interpolateur de ce type de courbes, ce qui limite les applications de cette méthode.

1.2.7.2 Architecture Ouverte

L'architecture des contrôleurs de MOCN basée sur un PLC et qui équipent la majorité des machines-outils est à architecture fermée. Toutefois, depuis quelques années, plusieurs contrôleurs équipant les MOCN sont présentés comme contrôleurs à architecture ouverte. Tout contrôleur est composé d'un niveau logiciel et d'un niveau matériel, il convient de définir ce que représente l'architecture ouverte pour ces deux niveaux.

Une architecture ouverte exige que les interfaces d'entrée et de sortie des niveaux logiciel et matériel soient définies et accessibles par les développeurs. L'ensemble modulaire formant ainsi le contrôleur est directement accessible par l'utilisateur qui peut le modifier et en améliorer les fonctionnalités. Ceci est réalisé en modifiant le code du niveau logiciel à l'aide d'une interface de programmation d'application "Application Programmation Interface" (API) ou encore en changeant des composantes matérielles.

Toutefois, un tel accès ouvert peut résulter en des bris majeurs ainsi que des blessures si les avantages de cette architecture sont exploités par un personnel peu expérimenté. La responsabilité des bris et blessures pourrait alors être attribuée au fabricant du contrôleur. Afin d'obtenir une certaine protection face à ce type de risques, l'ouverture des contrôleurs offerte actuellement par les fabricants est limitée.

Lorsqu'un contrôleur est acheté par un fabricant de MOCN afin de l'intégrer à l'une de leur machine, une interface de programmation permet, à un certain niveau, de modifier le comportement du contrôleur et d'y ajouter certaines fonctionnalités, tels des cycles d'usinage avancé. Toutefois, lorsque la MOCN est livrée à son acheteur, l'interface de programmation est verrouillée, empêchant ainsi toute modification logicielle au contrôleur. De plus, l'ensemble des composantes matérielles des contrôleurs basés sur PLC est propriétaire et ne peut être modifiée.

À titre d'exemple, la compagnie Hitachi-Seiki propose des MOCN dotées d'un contrôleur à "architecture ouverte" permettant une connectivité réseau de la MOCN [30]. Le lien Ethernet UUP (Universal User Port) proposé permet, entre autres, d'obtenir en temps réel des informations sur les paramètres du procédé d'usinage ainsi que sur les outils installés sur la machine. Le UUP est développé sur des contrôleurs Fanuc ou Yasnac, puis l'interface de développement est bloquée avant la livraison de la machine au client. Ce type d'ouverture est limité car il ne permet qu'un accès aux données présentes, sans permettre de modifications au contrôleur lui-même.

Les contrôleurs basés sur des ordinateurs personnels (Personal Computer, PC) sont également devenus populaires car ils présentent un certain niveau d'ouverture inhérent au PC lui-même. La compagnie Mazak propose un contrôleur dans lequel un PLC gère les déplacements de la machine et un PC, lié au PLC, permet de générer les programmes d'usinage en langage conversationnel. De plus, le système d'exploitation Windows installé sur le PC permet d'effectuer un lien entre la MOCN et le réseau, tout en rendant possible l'intégration d'autres logiciels directement sur la machine.

Toutefois, Hillaire [31] indique que de telles architectures ouvertes offertes par les fabricants de contrôleurs et de MOCN ne sont ouvertes que dans le contexte propriétaire du contrôleur. Il mentionne que l'architecture ouverte souhaitée devrait encourager l'utilisation d'une modularité des composantes matérielles et logicielles indépendantes des vendeurs.

Parmi les efforts de recherche axés vers l'architecture ouverte, Altintas et Erol [32] proposent une librairie de fonctions logicielles qui peuvent être utilisées afin de designer un contrôleur de MOCN. La modularité des composantes logicielles permet de reconfigurer ou de modifier rapidement le contrôleur qui est utilisé sur une plate-forme PC utilisée en temps réel à l'aide d'un processeur de signaux digitaux. L'ensemble de l'application est employée sous Windows NT. Les avantages d'un tel contrôleur et de son ouverture ont

été démontrés en laboratoire ainsi qu'en industrie en permettant, entre autres, un usinage adaptatif.

Zhang, Deng et Chan [33] proposent un contrôleur à architecture ouverte qui utilise des unités de caractéristiques d'usinage (NC Feature Unit, NCFU). Le système proposé fait appel au langage EXPRESS de STEP afin de contenir les informations concernant les unités de caractéristiques d'usinage, soit la géométrie à usiner et les paramètres associés. Ces unités sont ensuite transférés au contrôleur qui permet de générer en temps réel les trajectoires d'usinage liées aux caractéristiques. Puisque les déplacements de l'outil sont générés en temps réel, des paramètres d'usinage tels la profondeur de passe ou l'espacement entre les trajets d'outil peuvent être modifiés pendant l'usinage, contrairement aux contrôleurs conventionnels qui ne permettent que l'ajustement des vitesses d'avance et de coupe.

Afin d'augmenter la vitesse d'évolution des contrôleurs, Zuo et al. [34] proposent de réutiliser les modules logiciels des contrôleurs actuels afin de produire de nouveaux contrôleurs. La réutilisation des composantes logicielles permet d'accélérer le développement des contrôleurs en permettant l'utilisation des fonctionnalités de base des modules logiciels actuels qui serviront de fondation aux nouveaux contrôleurs ainsi développés. Bien que cette approche, si utilisée par des développeurs de contrôle, permettrait de concentrer les efforts de recherche sur de nouvelles fonctionnalités des MOCN, elle présente le désavantage de se baser sur les fondations existantes qui sont déficientes au niveau des performances et de l'intégration avec les autres modules d'un système intégré de fabrication. En effet, le système proposé utilise le code G comme langage de transfert d'informations à la MOCN.

1.3 Choix de la méthode d'intégration

La description du cheminement de l'information entre la conception et l'usinage sur MOCN ainsi que la revue de l'état de la recherche en cours présentés aux sections précédentes permet d'apprécier l'ampleur des besoins en intégration de haut niveau entre les divers modules d'un système intégré de fabrication. La revue de littérature présentée n'est pas exhaustive mais vise à indiquer les directions vers lesquelles les solutions aux problèmes d'intégration sont orientées.

À partir des informations présentées aux sections précédentes, on remarque qu'il existe une sous-utilisation des capacités des MOCN lorsque les programmes d'usinage sont réalisés à l'aide d'un module de FAO. En fait, Zietarski [35] mentionne ces problèmes de sous-utilisation dûs à un manque d'intégration entre les nouvelles capacités des MOCN et les systèmes de FAO. Il propose une intégration qui permet d'utiliser les possibilités de tournage et de fraisage disponibles sur une configuration relativement récente de MOCN, les tours-fraiseuses, qui permettent de réaliser les deux types d'usinage sur la même machine.

L'intégration qu'il propose est constituée d'un module spécialisé développé à même un logiciel CFAO ainsi que la modification appropriée du module de post-processeur qui permet alors de gérer les trajectoires d'outils générées et d'en produire le programme d'usinage qui sera acheminé à la MOCN. La solution proposée est toutefois orientée seulement vers l'intégration des capacités des tours-fraiseuses aux systèmes CFAO, ce qui en réduit fortement les possibilités d'extension et d'adaptation à des types différents de MOCN ainsi qu'à de nouvelles possibilités des MOCN.

Il est toutefois clair, suite à l'examen des informations fournies aux section précédentes, qu'il existe une sous-utilisation des possibilités offertes par la puissance de calcul disponible à même les contrôleurs de MOCN. Parmi les possibilités des MOCN qui sont lar-

gement sous-utilisées figurent les cycles cannés avancés (CCA). En effet, tous les cycles cannés avancés (CCA) disponible dans le contrôleur de la MOCN sont rendus inutilisables dans le module FAO par manque de standardisation des codes G utilisés pour appeler ces cycles. Ces cycles avancés peuvent inclure, mais ne sont pas limités à :

- **Cycles avancés d'usinage prismatique**
  - Surfaçage
  - Ébauche et finition de pochettes rectangulaires et circulaires en fraisage
  - Ébauche et finition d'îlots circulaires et rectangulaires en fraisage
  - Ébauche et finition de parois verticales en fraisage
  - Ébauche et finition de profils en tournage
  - Usinage hélicoïdal en fraisage
  - Usinage de rainures en fraisage
  - Motifs d'application des cycles cannés standardisés en fraisage et tournage
- **Cycles de mesure de pièce**
  - Coordonnées d'un point
  - Mesure de diamètres intérieurs et extérieurs
  - Distance entre deux points
  - Distance entre deux plans
  - Angle entre deux plans
  - Position d'un point mediant entre deux points
- **Cycle de positionnement du zéro pièce**
  - Coordonnée en Z d'une surface
  - Coordonnées du centre d'un diamètre intérieur ou extérieur
  - Coordonnées du centre d'une rainure
  - Coordonnées d'une paroi verticale
  - Coordonnées de l'intersection de deux plans
  - Positionnement angulaire d'un plan

De telles fonctionnalités sont couramment incluses à même les contrôleurs et leur utilisation présente plusieurs avantages. Le programme d'usinage qui utilise ces fonctionnalités présente une plus grande lisibilité pour l'opérateur de la MOCN. Afin d'assurer une productivité maximale, il convient de faire en sorte que l'opérateur de la machine puisse identifier les actions de la MOCN associées au programme d'usinage.

L'utilisation de tels cycles avancés permet de réaliser, par exemple, l'usinage d'une pochette rectangulaire à l'aide d'un seul code G. Les arguments associés au code en question vont définir les dimensions de la pochette et sa position ainsi que les paramètres d'usinage. L'usinage de la même pochette à l'aide des fonctions présentes dans le module FAO va résulter en plusieurs dizaines ou quelques centaines de lignes de code G n'utilisant que des déplacements linéaires ou circulaires afin de décrire l'ensemble des mouvements nécessaires pour accomplir l'usinage.

Ces cycles permettent à l'opérateur, lors de l'usinage de pièces d'essai ou de pièces unitaires, d'éditer directement sur le contrôleur les dimensions ou encore les paramètres d'usinage de ces cycles avancés puisque chaque caractéristique de l'usinage du cycle est un paramètre du code G associé. Cette flexibilité de modification du programme n'est pas possible dans le cas ou le programme de code G à été entièrement généré par le module FAO et où l'usinage de la pochette est représenté par des dizaines de lignes de code G régulier.

De plus, ces CCA ont été introduits dans le contrôleur par son fabricant ou encore par le fabricant de la MOCN qui utilise ce contrôleur. Ainsi, les capacités de la MOCN et de l'architecture de son contrôleur sont utilisées au maximum car les routines des cycles sont intégrées à même le contrôleur.

Puisque les codes G définissant les CCA sont représentés par une seule ligne de code G, une partie des problèmes associés à l'état de "data starvation" des MOCN utilisant un mode de transfert de programme par interface RS-232 peuvent ainsi être résolus. De même, des

programmes d'usinage réguliers nécessitant l'emploi d'un transfert en mode DNC sont réduits en taille par l'utilisation des CCA, ce qui peut permettre de les transférer en entier dans la mémoire du contrôleur avant même de débuter l'usinage.

Il est évident que la majorité des CCA d'usinage sont axés vers la géométrie prismatique, il est donc simple de faire un lien entre ces routines d'usinage, la modélisation par caractéristiques et l'extraction de caractéristiques. En effet, l'ensemble des efforts de recherche développés autour des modèles basés sur les caractéristiques et sur leur extraction à partir d'un format neutre peuvent être réutilisés afin de permettre l'intégration de ces CCA directement au module FAO.

La présente recherche vise donc à proposer une méthodologie d'intégration bi-directionnelle flexible et personnalisable des CCA aux contrôleurs de MOCN, ce qui permet de tirer avantage des travaux réalisés sur la modélisation par caractéristiques et leur extraction tout en permettant de rentabiliser les CCA dont les coûts de développement sont inclus dans le coût d'achat de toute MOCN.

CHAPITRE 2

MÉTHODOLOGIE D'INTÉGRATION

Puisque les cycles cannés avancés disponibles sur les différentes marques et les différents modèles de MOCN sont extrêmement variés, tant au niveau de leurs capacités que de la syntaxe employée pour les utiliser, il convient de définir une méthodologie d'intégration à la fois performante et flexible. Ceci est nécessaire afin de permettre l'adaptation de cette méthodologie au plus vaste éventail possible de modules FAO ou CFAO, de modules post-processeurs et de contrôleurs de MOCN.

2.1 Intégrations envisageables

Il existe plusieurs niveaux d'intégration possible de manière à rendre les CCA des MOCN disponibles à même le module FAO, et chacun de ces types d'intégration présente des avantages et des inconvénients qui vont maintenant être explorés de manière à cerner la méthodologie qui sera mise en application. La démonstration des possibilités d'intégration offertes permet de justifier le choix qui sera effectué subséquemment.

2.1.1 Intégration au niveau du code G

L'intégration au niveau du code G est celle qui est habituellement utilisée en industrie lorsque l'utilisation des CCA est nécessaire ou souhaitée. L'introduction des codes G associés aux CCA est effectuée manuellement par édition du fichier du programme d'usinage. Les informations géométriques nécessaires à la description des paramètres des codes G associés aux cycles cannés avancés qui désirent être utilisés sont extraites du modèle présent dans le système FAO à l'aide des fonctions de mesure disponibles par l'interface usager de ce module.

La syntaxe des codes G utilisés pour représenter les CCA est rébarbative et laborieuse à introduire dans le programme d'usinage. À titre d'exemple, le code G utilisé pour réaliser l'usinage d'une pochette rectangulaire intérieure disponible sur les contrôleurs Seicos / Fanuc 16i des MOCN Hitachi-Seiki est :

G328 X x_pos **Y** y_pos **Z** depth **R** retract **I** x_dim **J** y_dim **K** step_over **Q** z_step **A** corner_radius **D** tool_offset **E** finish_feedrate **U** finish_rpm **V** z_feedrate **F** feedrate

Les risques d'erreurs associés à l'introduction manuelle d'une telle instruction sont élevés et le temps requis afin d'ajouter une telle ligne de code au programme de code G diminue l'attrait d'utilisation d'une telle méthode qui n'est conséquemment utilisée qu'en cas de nécessité, ne permettant pas une rentabilisation des CCA dont le coût est inclus dans celui de la MOCN.

2.1.2 Intégration au niveau du post-processeur

Cette méthode d'intégration est relativement rapide à mettre en place mais peu performante. Elle consiste à utiliser les possibilités d'édition manuelle du CL-File qui est présente dans la vaste majorité des modules FAO ou CFAO qui utilisent un CL-File standardisé comme format de sortie. Avec cette méthode, illustrée à la figure 2, seule une personnalisation au niveau du post-processeur est nécessaire de manière à permettre l'utilisation des CCA directement à partir du système FAO.

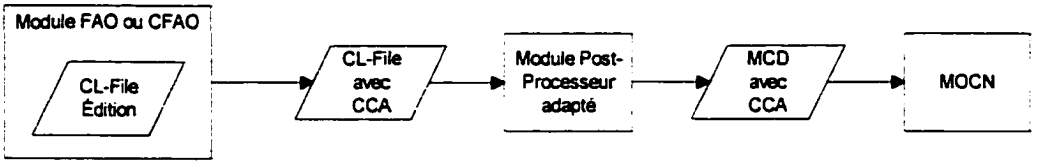


Figure 2 Intégration CCA au post-processeur.

L'utilisation de cette intégration au niveau du post-processeur consiste à éditer manuellement le CL-File à même le module de FAO afin d'y inclure, à l'endroit approprié, une commande personnalisée qui décrit à l'aide de mots tirés du langage APT, toute l'information nécessaire à la génération du code G associé au CCA associé. Le CL-File complet, incluant ou non des séquences d'usinage créées à l'aide du moteur de génération de trajectoires d'outils du système FAO, est par la suite envoyé au module de post-processeur.

Lorsque les lignes du CL-File contenant les commandes personnalisées des CCA seront lues par le post-processeur, les macro correspondantes seront appelées afin de produire le code G associé au CCA dans le format requis par la MOCN pour laquelle le CL-File est post-traité. Ainsi, le nombre de macros spécialisées qui doivent être ajoutées au post-processeur se doit d'égaler le nombre de CCA différents disponibles sur l'ensemble du parc de MOCN pour lesquelles les fonctionnalités des CCA désirent être utilisées. Le programme de code G ainsi obtenu peut être transféré au contrôleur de la MOCN afin de procéder à l'usinage de la pièce.

Le principal avantage de cette méthode d'intégration est sa simplicité. En effet, seul le module de post-processeur nécessite une personnalisation afin de pouvoir tirer avantage de l'ensemble des CCA d'un parc de machines-outils.

Toutefois, les désavantages associés à ce type d'intégration sont nombreux. Tout d'abord, l'utilisateur du module FAO doit effectuer manuellement l'entrée de données dans le CL-File afin de pouvoir en tirer le code G associé par l'entremise du module de post-processeur. Mis à part les risques d'erreurs associés à l'entrée manuelle de données, la syntaxe nécessaire à la description en langage APT d'un CCA est relativement rébarbative. Par exemple, la ligne de code APT nécessaire à la définition d'un CCA permettant l'usinage d'une pochette rectangulaire intérieure peut ressembler à :

CYCLE/POCKET, RCTNGL, IN, XAXIS, x_pos, **YAXIS**, y_pos,
DEPTH, depth, **CLEAR**, clear, **XDIM**, x_dim, **YDIM**, y_dim,

STPOVR, step_over, **ZSTEP**, z_step, **FEDRAT**, feedrate,
WIDTH, width, **RADIUS**, corner_radius, **ZFEED**, z_feedrate,
FINFED, finish_feedrate, **FINRPM**, finish_rpm

où les éléments en caractères gras sont des mots de base du langage APT qui ont été rajoutés afin de représenter le nom des arguments nécessaires à la définition du CCA. Les éléments en caractères normaux représentent les valeurs de chacun des paramètres. Le type de paramètre ainsi que leur nombre dépend du CCA visé et des options qui peuvent y être appliqué mais cet exemple démontre bien la tâche que représente la définition manuelle d'un tel CCA et le risque d'erreurs y étant associé. Il faut de plus obtenir à l'aide de fonctions présentes dans le système de FAO toutes les informations géométriques nécessaires à l'ajout de la commande APT du CCA, telles les coordonnées de l'élément à définir ou encore ses dimensions.

Afin que l'utilisateur du système FAO puisse utiliser les CCA, il doit disposer d'un document qui indique la syntaxe associée à chacun d'eux ainsi que la définition de chacun des arguments le définissant. La gestion des informations requises liée à la complexité de la syntaxe de définition des CCA en APT résulte en une probabilité élevée d'erreurs. Des collisions machines, des bris d'outils et de pièces peuvent résulter de l'entrée manuelle de données qui ne présente alors aucune vérification informatisée de la validité de la syntaxe et des valeurs numériques utilisées.

La personnalisation du post-processeur consiste à définir une macro pour chaque CCA qui désire être utilisé. Cette macro utilise comme entrée la ligne de CCA du fichier APT et génère en sortie la ligne en code G associée au CCA. Les macros de CCA seront appelées lorsqu'une association sera effectuée par le post-processeur entre une ligne du CL-File et le format d'entrée d'une des macros de CCA. Tout ceci nécessite que l'environnement de développement de post-processeur utilisé permette l'emploi de macros et que le langage de programmation fournit dans cet environnement soit suffisamment flexible pour permettre une telle intégration.

Dans le cas où le module de FAO utilise un format intermédiaire propriétaire différent du CL-File, les possibilités d'édition de ce fichier sont habituellement réduites, ce qui est également le cas du module de post-processeur habituellement intégré à de telles plates-formes FAO. L'aspect limité des possibilités de développement de ces post-processeurs peut proscrire le développement des macros nécessaires à la génération des codes G représentant les CCA.

Un autre désavantage est que la simulation virtuelle d'usinage offerte en tant que solution intégrée aux logiciels de FAO, basée sur la lecture du CL-File, ne permet généralement pas la simulation native des CCA. La majorité de ces simulateurs sont de type I, ne permettant que la visualisation de l'outil, et présentant des possibilités de développement minimes ou encore nulles face à l'intégration de routines telles les CCA.

De plus, une telle méthodologie ne permet pas de tirer avantage d'un modèle basé sur les caractéristiques qui peut être disponible à même le modeleur solide du module FAO utilisé. De même, tout algorithme d'extraction de caractéristiques est inutilisable lors de l'emploi de ce type d'intégration. Puisque l'introduction de données est manuelle, il n'existe pas d'associativité entre la géométrie et le CCA introduit dans le CL-File, et toute mise à jour du CL-File par le système FAO suite à l'ajout du code APT définissant le CCA risque d'éliminer les données entrées manuellement, selon le logiciel de FAO utilisé.

L'ensemble des désavantages que présente ce type d'intégration, plus particulièrement les risques d'erreurs et la syntaxe utilisée, font de ce mode d'intégration un choix peu intéressant. À l'ère de la paramétrisation et de l'associativité des modules de CAO et de FAO, cette avenue revêt donc très peu d'avantages.

2.1.3 Intégration complète au module de FAO

La majorité des logiciels de CAO / FAO de haut niveau offrent une interface de programmation d'application (API) constituée d'une librairie de fonctions qui permettent de créer de nouvelles fonctionnalités à même le logiciel et dont l'utilisation sera transparente du point de vue de l'utilisateur. Les langages de développement utilisés par les diverses plates-formes FAO et CFAO varient selon la compagnie qui développe les logiciels, mais le C, le C++ ainsi que le Visual Basic sont habituellement utilisés pour les logiciels exploités sous Windows.

Les possibilités de développement sont limitées par l'étendue des API offerts avec le logiciel. En effet, bien que l'ensemble du langage disponible pour le développement dans un logiciel soit disponible, le seul moyen d'interfacer les fonctionnalités développées au coeur du système FAO s'effectue à l'aide des API fournis, limitant ainsi les capacités de développement. Il est habituellement possible d'accéder par les API aux fonctionnalités du modeleur solide afin d'extraire de l'information de la géométrie existante, de la modifier ou encore de créer de la nouvelle géométrie. De plus, une interface utilisateur peut être créée à l'aide de l'environnement de développement utilisé ou encore des API fournis.

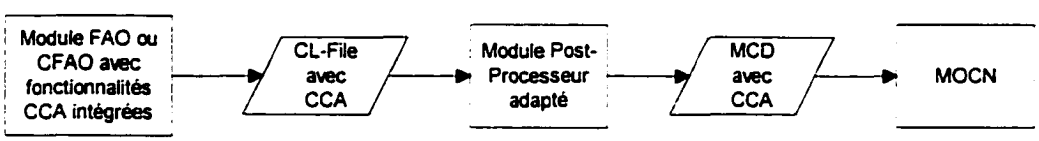


Figure 3 Intégration CCA au module FAO.

Une intégration des CCA directement au logiciel de FAO devient alors possible par l'utilisation de ces API. Une telle intégration, illustrée schématiquement à la figure 3, permet entre autre d'utiliser les caractéristiques comprises dans le modèle dans le cas où il a été créé par une approche basée sur les caractéristiques ou encore d'utiliser les algorithmes

d'extraction de caractéristiques qui peuvent être présentes, pour autant que les API fournis avec la plate-forme FAO ou CFAO permette un tel accès à l'information.

Cette approche permet ainsi de tirer avantage de l'information riche contenue dans les caractéristiques d'usinage du modèle qui sont particulièrement bien adaptées aux CCA d'usinage et à certains CCA de mesure. L'associativité entre les caractéristiques d'usinage et les données des CCA peut ainsi être maintenue dans le cas ou le module développé à l'aide des API a également intégré cette fonctionnalité.

En conservant ainsi l'associativité, des modifications de la géométrie dans le module de FAO ou même dans le module CAO seront répercutées sur les paramètres des CCA qui portent sur la géométrie. Le CL-File associé sera ainsi mis à jour automatiquement, permettant une intégration de haut niveau entre les CCA et les modules de FAO et de CAO.

Dans le cas où les caractéristique sont inaccessibles par une absence des API appropriés ou des fonctionnalités requises, une sélection graphique des surfaces à usiner par l'entremise des CCA disponibles est envisageable. Une telle approche représente une productivité élevée lorsque comparée à l'intégration au post-processeur et elle permet de maintenir une associativité entre les surface à usiner et les CCA qui ont étés définis.

L'information recueillie lors de la création des CCA dans le système de FAO à l'aide des fonctionnalités créées avec les API permet de générer un CL-File où les lignes définissant les CCA en langage APT ont été générées automatiquement. Afin de compléter l'information requise à la définition du CCA, les paramètres d'usinage peuvent être introduits par l'utilisateur à l'aide d'une interface personnalisée créée pour chaque type de CCA.

Le CL-File ainsi obtenu peut par la suite être acheminé au module de post-processeur personnalisé qui va générer les codes G appropriés tel que décrit à la section 2.1.2. Les programmes d'usinage résultants peuvent ensuite être transférés au contrôleur de la MOCN afin d'usiner la pièce. L'utilisation de ce type d'intégration devient alors complètement

transparente du point de vue de l'utilisateur du système de FAO qui perçoit le module de CCA comme une partie native du logiciel de FAO utilisé.

De plus, ce type d'intégration permet d'éliminer les erreurs associées à l'introduction manuelle de données nécessaire à la réalisation des intégrations présentées aux sections 2.1.1 et 2.1.2. L'utilisation d'un simulateur d'usinage qui offre un environnement de développement permettant de rendre possible la simulation des CCA est ainsi envisageable au niveau du CL-File et du fichier de code G, permettant ainsi une simulation de vérification dans l'environnement FAO et une vérification finale suite au traitement des données par le module de post-processeur.

Il est à mentionner qu'une bonne partie des logiciels de CAO, FAO ou CFAO de haut niveau présentant une interface de développement avec API utilisent des simulateurs qui contiennent une interface de développement qui peut permettre de développer les routines de simulation des CCA. Habituellement, les compagnies qui font l'acquisition de systèmes FAO de haut niveau fabriquent des pièces à valeur élevée, sur des MOCN de valeur tout aussi élevée. Ainsi, des simulateurs indépendants de type II sont habituellement utilisés tant au niveau de la vérification du CL-File que de la simulation finale effectuée sur les fichiers de code G. De tels simulateurs proposent presque tous des interfaces de développement utilisant ou non un langage de programmation propriétaire mais qui permet du moins d'émuler les déplacements associés à l'exécution des CCA.

Toutefois, une telle implantation exige que le logiciel FAO ou CFAO visé présente un environnement de développement et que celui-ci soit suffisamment puissant pour permettre la réalisation d'une telle intégration. Cette limitation est toutefois amoindrie par le fait qu'un nombre croissant de logiciels FAO et CFAO offrent des API et que les possibilités offertes aux développeurs par l'entremise de ces API soit en hausse.

2.1.3.1 Associativité bi-directionnelle

On peut remarquer en observant la figure 3 que le cheminement de l'information résultant d'une intégration des CCA au système FAO par l'entremise des API est unidirectionnelle du système FAO à la MOCN. Il en est de même du transit d'information existant dans tout système intégré de fabrication constitué de modules disponibles commercialement. Cette unidirectionnalité est problématique car il est courant que l'opérateur modifie manuellement les programmes d'usinage afin d'adapter, par exemple, les vitesses d'avance et de rotation de broche aux caractéristiques physiques du montage et des outils utilisés.

De telles modifications ne sont jamais réacheminées aux bureaux d'ingénierie de manière électronique car les modules compris entre le module de FAO et celui de la MOCN ne permettent pas un tel cheminement de l'information. Il est de même rare qu'une procédure soit implantée en industrie de manière à indiquer aux programmeurs de contrôle numérique (NC) les modifications qui ont été réalisées par l'opérateur de la MOCN. Une telle situation résulte en une baisse de productivité associée directement au temps mis par l'opérateur à effectuer les modifications suite à la résolution des problèmes résultant des paramètres d'usinage initialement inclus dans le programme d'usinage.

Une baisse indirecte de la productivité est également résultante de l'absence du cheminement bi-directionnel de l'information. En effet, les connaissances et l'expérience de l'opérateur de la MOCN qui a réalisé les modifications ne sont jamais ré-acheminées à l'ingénierie qui pourrait tirer avantage des paramètres d'usinage optimisés qu'a introduit l'opérateur de la MOCN dans le programme d'usinage.

Cette conséquence indirecte prend de l'ampleur dans le cas où la compagnie utilise une banque de données d'usinage suggérant des paramètres d'usinage par défaut selon le matériau usiné et l'outil utilisé dans le système FAO pour réaliser l'usinage. L'absence de retour d'information à partir de la MOCN signifie une diminution importante de la qualité

de l'information contenue dans la base de données d'usinage utilisée par la compagnie. Ainsi, lors d'un prochain usinage dans les mêmes conditions, le programme sera à nouveau généré avec des paramètres d'usinage erronés et l'opérateur devra à nouveau prendre le temps de modifier ces paramètres.

L'ensemble des problèmes en lien avec l'unidirectionnalité de cheminement de l'information sont amplifiés lors de l'intégration des CCA au système de FAO. En effet, un des avantages de l'emploi des CCA est de permettre à l'opérateur de réaliser, outre la modification des vitesses d'avance et de rotation de broche, des modifications à la géométrie usinée. Les paramètres du code G d'un CCA peuvent être altérés directement au contrôleur de la MOCN afin d'adapter le programme à l'usinage de la pièce en cours.

Dans le cas des modifications de paramètres de vitesses d'avance et de coupe ou encore de mise en route du liquide d'arrosage, des problèmes de vibration et d'usure d'outils peuvent être résolus. Toutefois, la modification des paramètres des CCA au niveau du contrôleur peuvent résulter en des différences dimensionnelles et même géométriques de la pièce usinée lorsque comparée au modèle CAO.

Ces modifications se doivent d'être transférées au modèle CAO suite à l'usinage de manière à ce que le modèle CAO conservé pour des fins d'archivage ou de réutilisation concorde avec la pièce physique qui a été usinée à partir des informations de ce modèle. Il convient donc d'inclure à une intégration des CCA la possibilité de faire voyager l'information bi-directionnellement entre le module FAO et le module MOCN selon le diagramme de la figure 4. Ceci permet de tirer avantage des possibilités de modification des CCA à même le contrôleur sans rendre le modèle CAO et l'information du modèle FAO invalides.

La nécessité d'incorporer la communication bi-directionnelle FAO – MOCN afin de conserver la validité des informations des modules de CAO et de FAO permet d'éliminer les intégrations limitées au code G et au post-processeur citées dans les sections précédentes.

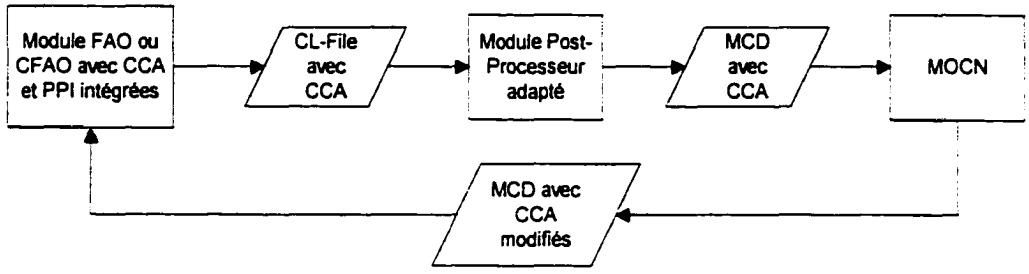


Figure 4 Cheminement bi-directionnel FAO – MOCN

En effet, seul le développement à l'aide des API d'un logiciel de FAO ou de CFAO permet de mettre à jour le modèle et les paramètres d'usinage à l'aide des informations contenues dans un fichier externe, soit le programme d'usinage en code G.

Cette communication à sens inverse MOCN – FAO doit inclure une interface de post-processeur inversé (PPI) afin d'extraire les informations des CCA présentées sous forme de code G. Les informations mises à jour ainsi obtenues peuvent par la suite être acheminées à des fonctions du module d'intégration développé dans le système de FAO. Ces fonctions ont pour rôle de mettre à jour la géométrie ainsi que les paramètres d'usinage du modèle afin de refléter les changements apportés au programme d'usinage au niveau du contrôleur.

Bien qu'il soit possible d'intégrer le module de post-processeur inversé au module de post-processeur ou encore d'en constituer un module indépendant, ces solutions ne sont respectivement pas praticables ou non-efficaces. En effet, l'intégration au module de post-processeur n'est pas possible car les environnements de développement de post présentent une architecture native conçue spécifiquement pour recevoir un CL-File ou son équivalent propriétaire en entrée et générer le code G en sortie. Les possibilités de programmation offertes au coeur des environnements de développement de post les plus évolués ne permettent pas de recevoir en entrée un programme d'usinage en code G. En plus des problèmes dûs à l'architecture interne des post-processeurs, la vérification de syntaxe qui est effectuée automatiquement sur le fichier d'entrée ne peut être modifiée de manière à accepter un programme d'usinage en code G.

La création d'un module de PPI indépendant des modules de post-processeur et de FAO n'est pas attrayante car l'exécution de ce module afin d'extraire du programme en code G les informations pertinentes exige des actions supplémentaires. De plus, les informations ainsi obtenues doivent tout de même être mises sous forme de fichier, qui doit à son tour être lu par le système FAO afin de permettre la mise à jour du modèle.

2.2 Caractéristique de l'intégration des CCA

Les informations contenues dans la section 2.1 permet de conclure que la seule méthode d'intégration complète et permissible est celle de l'intégration des CCA au module de FAO par l'entremise des API disponibles dans ce module. De plus, l'intégration doit présenter des fonctionnalités de post-processeur inversé afin de mettre à jour le modèle suite aux modifications possiblement apportées au programme d'usinage par l'opérateur de la MOCN.

Toutefois, ces informations concernant l'intégration projetée ne sont pas suffisantes pour définir les caractéristiques complètes de la méthodologie d'intégration. Il convient donc de définir quelles sont les capacités recherchées par la définition de l'intégration afin de mieux cerner les détails de la fusion des CCA avec un module de FAO. Ces capacités vont cependant engendrer des limitations aux possibilités de l'intégration, et celles-ci seront exposées de manière à identifier les applications possibles de l'intégration ainsi décrite.

Tel qu'il a pu en être jugé jusqu'à présent, le but du présent travail est de produire des résultats applicables dans un environnement industriel pouvant générer des avantages pratiquement immédiats aux éventuelles entreprises qui souhaitent procéder à l'intégration des CCA à leur logiciel de FAO. Cette approche à été privilégiée de manière à permettre des applications immédiates de la recherche en cours. Afin de réaliser cet objectif, la méthodologie proposée se doit d'être basée sur la technologie présentement disponible sans

s'y limiter et se veut donc une extension des capacités actuelles des systèmes de fabrication intégrés.

Une telle approche pratique et applicable à court terme à été privilégiée de manière à permettre l'utilisation de la technologie développée en son entier. En effet, des efforts tels que la norme STEP et de son sous ensemble STEP-NC offrent de grandes promesses, mais leurs réalisation, dans toute son ampleur, en milieu industriel, risque d'exiger plusieurs années ou même des décennies, ce qui peut résulter en une désuétude partielle des possibilités ou des applications de telles normes.

Ainsi, les efforts du présent travail sont orientés de manière à pouvoir produire un résultat applicable industriellement de la façon la plus directe possible. Le développement proposé se doit donc d'être réalisé de manière à être extensible, dans les limites du présent travail, aux éventuelles évolutions technologiques des systèmes FAO, des MOCN, de leur contrôleur ainsi que des CCA.

Les caractéristiques présentées aux sections suivantes servent de fondement au développement de l'intégration proposée. Elles permettent entre autre d'assurer une application de la méthodologie au maximum de systèmes de FAO et de post-processeurs disponibles commercialement.

2.2.1 Intégration au module FAO

Afin de pouvoir intégrer les CCA de manière complète à un système intégré de fabrication, il est nécessaire d'apporter des modifications au niveau du module de post-processeur et du module de FAO. C'est toutefois au niveau de ce dernier module que les efforts de développement seront les plus importants afin de produire une intégration d'apparence native pour l'utilisateur. De plus, l'intégration au logiciel de FAO doit être réalisée de manière à permettre une associativité entre les CCA et le modèle de la pièce, incluant la géométrie,

les dimensions ainsi que les autres données nécessaires contenues dans le modèle. Les sections suivantes illustrent les lignes directrices qui doivent guider la procédure d'intégration afin qu'elle soit flexible dans son utilisation et facilement modifiable.

2.2.1.1 Flexibilité

Bien que le code nécessaire à l'intégration des CCA soit différent selon le logiciel de FAO pour lequel il est spécifiquement créé, il n'en demeure pas moins que la structure utilisée afin de coder l'application se doit d'être flexible. On entend par "flexible" la capacité de pouvoir ajouter relativement facilement de nouveaux CCA ou de nouvelles MOCN à l'intégration sans devoir recoder une partie majeure de l'application d'intégration. Pour ce faire, les données concernant les MOCN et les CCA leur étant associés doivent être conservées dans une base de données où seront présentes toutes les informations nécessaires à la génération du CL-File. La figure 5 présente le modèle retenu où l'information des MOCN est contenue dans une base de données qui gère également les CCA disponibles pour chacune des MOCN intégrées. L'information ainsi rendue disponible au module d'intégration des CCA du système FAO permet la génération des CCA au format APT dans le CL-File.

Un tel modèle peut être mis en place selon deux méthodes. En effet, lors de la définition de l'usinage d'une pièce dans un logiciel FAO, la procédure habituelle consiste à définir sur quelle MOCN sera réalisé l'usinage. Pour la majorité des logiciels, la définition des séquences d'usinage ne peut prendre place qu'à la suite du choix de la MOCN pour laquelle sera créé le CL-File. Ainsi, lors de la définition des séquences d'usinage basées sur les CCA, la MOCN sur lesquelles elles seront utilisées est définie. Il est alors possible de créer la base de données qui contient les informations sur les MOCN et les CCA en liant les CCA directement avec les MOCN qui les contiennent.

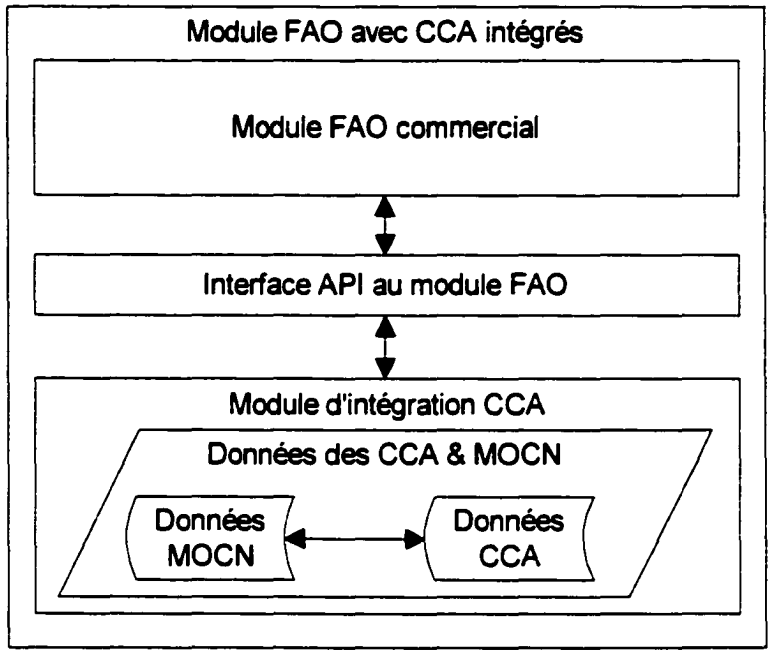


Figure 5 Schémas de l'intégration flexible

Une telle approche permet de ne traiter que les CCA qui sont applicables à la MOCN choisie pour réaliser l'usinage. Ce système est performant car les CCA disponibles pour réaliser l'usinage ainsi que les paramètres qu'ils nécessitent peuvent être obtenus directement de la base de données. Toutefois, si on désire changer la MOCN sur laquelle sera effectué l'usinage suite à la définition des séquences, il faut mettre à jour l'ensemble des séquences qui utilisent les CCA. Il est alors possible qu'une des séquences qui utilisait un CCA disponible sur la MOCN originale ne puisse être mise à jour par l'absence d'un CCA correspondant pour la nouvelle MOCN qui à été sélectionnée. Cette problématique est illustrée à la table II qui présente la structure schématique de la base de données nécessaire afin de lier les CCA aux MOCN.

La section des paramètres des CCA de la table II doit contenir l'ensemble de l'information que nécessite l'application, incluant entre autres les champs nécessaires dans l'interface

Tableau II

Base de donnée des CCA liés aux MOCN

MOCN	Données CCA	
	CCA	**Paramètres CCA**
MOCN A	CCA A1	Par. CCA A1
	CCA A2	Par. CCA A2
	⋮	⋮
	CCA An	Par. CCA An
MOCN B	CCA B1	Par. CCA B1
	CCA B2	Par. CCA B2
	⋮	⋮
	CCA Bn	Par. CCA Bn
⋮	⋮	
MOCN N	CCA N1	Par. CCA N1
	CCA N2	Par. CCA N2
	⋮	⋮
	CCA Nn	Par. CCA Nn

usager afin de définir le CCA, les paramètres nécessaires à la définition du CCA dans le CL-File et les intervalles de valeurs des paramètres des CCA.

Le second type de mise en place du modèle présenté à la figure 5 consiste à définir seulement les CCA communs à l'ensemble des MOCN sur lesquelles de telles fonctionnalités désirent être utilisées. Les CCA communs sont alors représentés dans la base de données en ne tenant pas compte des MOCN auxquelles ils correspondent, tel que l'illustre la table III.

Cette approche permet d'éliminer les problèmes associés au changement de MOCN dans le module FAO suite à la génération des séquences d'usinage. Cependant, un nombre relativement élevé de CCA disponibles sur des machines spécifiques sont ainsi rendus inutilisables, ce qui diminue l'intérêt de ce type d'intégration. Plus le nombre de CCA non

Tableau III

Base de données des CCA communs

CCA	Paramètres CCA
CCA 1	Par. CCA 1
CCA 2	Par. CCA 2
CCA 3	Par. CCA 3
⋮	⋮
CCA n	Par. CCA n

communes à l'ensemble des MOCN est élevé, moins le nombre de CCA disponibles dans le module FAO est élevé.

Pour un groupe de CCA représentant des fonctionnalités similaires sur des MOCN différentes, il est probable que les paramètres utilisés par ces CCA diffèrent. Par exemple, un cycle canné avancé d'usinage qui permet d'enlever la matière située autour d'un îlot cylindrique peut exiger la définition des diamètres initial et final d'usinage pour une MOCN. Sur une seconde MOCN, le CCA correspondant peut exiger la définition du diamètre final ainsi que la largeur de matériel à usiner. Il convient donc de construire la base de données des CCA en tenant compte des différences de syntaxes qui peuvent exister entre des cycles de fonctionnalités similaires.

La différence de syntaxe entre des CCA similaires sur des MOCN différentes ainsi que le fait que certains types de CCA ne soient pas disponibles sur l'ensemble des MOCN pose des problèmes au niveau des deux types de bases de données proposées. Toutefois, la majorité des contrôleurs de MOCN permettent de réaliser de la programmation paramétrique en code G, permettant ainsi de définir de nouveaux CCA à même le contrôleur. La programmation paramétrique utilise des instructions de branchement utilisables dans un programme ou une macro qui résident dans le contrôleur. Un certain nombre de variables qui peuvent contenir des valeurs numériques sont disponibles de manière à stocker des

informations temporaires lors de l'exécution de la macro. De plus, un accès à certains des paramètres de la MOCN est disponible par l'entremise de la programmation paramétrique. Des valeurs telles que le décalage en rayon ou en longueur de l'outil courant, la vitesse de la broche et la position du zéro pièce sont accessibles.

La définition en programmation paramétrique de CCA permet ainsi d'uniformiser la disponibilité de ces cycles cannés avancés pour un parc donné de MOCN. Il est alors possible de changer la MOCN choisie dans le module FAO sans engendrer de problèmes dûs à des CCA exclusifs à certaines machines. De même, la programmation paramétrique permet d'éliminer les problèmes de limitations associés à l'utilisation des CCA communs dans la base de données.

Toutefois, il peut parfois être laborieux de tenter d'émuler les CCA d'une MOCN sur une seconde qui ne possède pas nativement de CCA correspondant. Cette réplique par programmation paramétrique des CCA est d'autant plus complexe que le code des CCA intégrés aux machines-outils n'est pas disponible publiquement, et qu'une méthode d'essais et erreurs est nécessaire à la réalisation paramétrique d'une émulation de ces CCA.

Il est donc proposé d'utiliser la structure de base de données présentée à la table II en y incluant que les cycles qui sont les plus susceptibles d'être employés, selon le type de produits fabriqués par l'entreprise qui effectue l'intégration de ceux-ci au module FAO. Les cycles qui ne sont pas disponibles sur l'ensemble des MOCN visées par l'intégration peuvent être émulés par programmation paramétrique dans le contrôleur des machines déficientes. Cette approche permet une meilleure gestion des CCA qui sont alors associés aux MOCN tout en éliminant les problèmes associés aux cycles spécifiques à seulement quelques machines-outils.

2.2.1.2 Modularité

L'approche utilisée lors de la programmation du module d'intégration des CCA au module FAO se doit de résulter en une architecture modulaire. Le respect de ce critère permet de diminuer les efforts requis lors de l'ajout éventuel de possibilités et de fonctionnalités additionnelles au module d'intégration. Le choix de la division modulaire du module d'intégration des cycles cannés avancés (MICCA) doit être selon l'interface au logiciel de FAO, le format de la base de données des CCA ainsi que des fonctionnalités du MICCA. Bien que le découpage modulaire de l'application d'intégration désirée dépende du logiciel de FAO utilisé ainsi que des possibilités offertes par les API et du langage de programmation utilisé, les modules suivants doivent être présents afin d'assurer l'évolution de l'application :

- Interface API au modeleur solide.
- Interface API à l'interface utilisateur
- Interface API à la création de CL-Files
- Interface API aux données du modèle (non-géométriques)
- Générateur de CCA dans le CL-File
- Post-processeur inversé

Chacune des interfaces API permet de faire le lien entre le coeur de l'application programmée et le module de FAO. Les deux derniers modules représentent les principales composantes du MICCA. La figure 6 permet d'apprécier les relations qui existent entre ces différents modules ainsi que le cheminement de données associé.

En ce qui concerne l'interface API au modeleur solide, celle-ci contient les fonctions qui permettent d'obtenir les informations géométriques et dimensionnelles du modèle à partir du modeleur solide. Cette information est utilisée par le générateur de CCA afin de permettre à l'utilisateur de sélectionner les caractéristiques ou les surfaces de la pièce qui

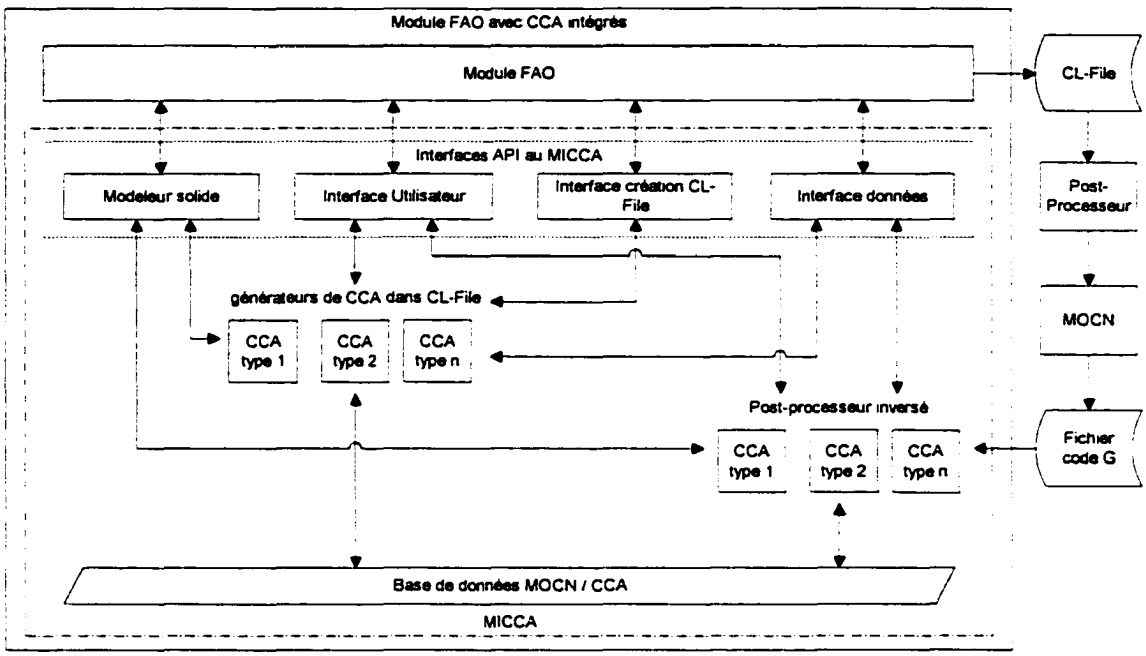


Figure 6 Schémas de l'intégration modulaire

seront utilisées par les CCA à définir. De plus, cette interface est également utilisée par le module de post-processeur inversé de manière à mettre à jour la géométrie du modèle lors de la lecture du fichier de code G. En effet, l'opérateur de la MOCN peut avoir modifié certains des paramètres d'un ou de plusieurs CCA du programme d'usinage afin de modifier les dimensions ou les positions des caractéristiques usinées par ces CCA. Le post-processeur inversé utilise alors l'API au modeleur solide afin de mettre à jour le modèle CAO de la pièce à partir du programme en code G modifié.

Afin de permettre l'affichage des boîtes de dialogue nécessaires à la sélection des entités graphiques et des paramètres des CCA, une interface au système de FAO est nécessaire. Le module API d'interface utilisateur permet d'utiliser les fonctionnalités d'affichage du logiciel de FAO afin de faire apparaître les messages, menus et dialogues nécessaires à la création des CCA. Ce type d'interface est préférable à l'utilisation des possibilités graphiques offertes par le langage de développement utilisé. En effet, les API d'affichage

des logiciels de FAO, si existants, permettent de conserver la même apparence d'interface qui caractérise l'ensemble du logiciel, permettant une intégration du MICCA d'apparence plus native. De même, cette interface API graphique est utilisée lors de la modification des CCA déjà créés.

Il faut considérer que l'information nécessaire à la définition complète d'un CCA varie selon le type de géométrie qui est usinable par chaque CCA. Ainsi, l'interface API d'affichage doit comporter autant de sous modules que de types de géométries usinables différentes. En effet, certains CCA de fonctionnalités similaires au niveau de la MOCN peuvent utiliser les mêmes boîtes de dialogues dont certains éléments seront désactivés afin de représenter le CCA précis pour lesquelles elles ont été appelées. Toutefois, certains type de CCA ne peuvent tout simplement pas utiliser une interface graphique commune, ce qui exige la programmation de plusieurs interfaces graphiques différentes. Par exemple, un champ diamètre n'est pas significatif à l'intérieur d'une boîte de dialogue servant à définir un CCA de surfaçage.

Puisque le CL-File est généré par le système de FAO et que celui-ci peut inclure ou bien l'ensemble des séquences d'usinage de la pièce ou uniquement quelques séquences présélectionnées, il convient d'utiliser les API du logiciel afin d'intégrer les CCA au CL-File produit. Ainsi, l'interface API qui gère la création du CL-File est utilisée par le module de génération de CCA afin d'ajouter au CL-File créé par les fonctionnalités natives du système de FAO les commandes résultantes de l'ajout des CCA.

Mis à part les données géométriques et dimensionnelles qui sont obtenues par l'entremise de l'interface API au modeleur solide, d'autres types d'information doivent être extraits du modèle afin de permettre la création des CCA. En effet, il est essentiel de pouvoir obtenir, par exemple, la MOCN qui à été sélectionnée dans le module de FAO afin de pouvoir obtenir les caractéristiques des CCA associés à partir de la base de donnée MOCN – CCA. De même, il est essentiel de pouvoir accéder à la géométrie des outils qui sont définis dans le modèle afin de réaliser l'usinage.

Lorsque la définition des CCA est complétée et que le CL-File est généré, il est nécessaire de pouvoir conserver les paramètres et caractéristiques entre chacun de ces CCA dans le fichier du modèle de la plate-forme FAO. Ceci est nécessaire de manière à ce que l'information soit disponible à nouveau dans le cas où la session de travail a été interrompue et que le modèle doit être lu à nouveau. De plus, une telle sauvegarde d'information dans le fichier du modèle FAO au format natif permet de créer une associativité entre les CCA définis, la géométrie du modèle et les autres informations pertinentes. Cette capacité de sauvegarder de l'information dans le fichier du modèle nécessite absolument l'utilisation des API afin de pouvoir accéder le format propriétaire utilisé dont les détails internes ne sont pas divulgués.

Le module de génération des CCA dans le CL-File peut présenter différents types d'architecture interne, le choix du type approprié doit être effectué selon le langage de programmation utilisé, les API disponibles ainsi que du niveau d'intégration désiré entre les CCA et le module de FAO. Il doit toutefois être lié aux modules de modèle solide, d'interface utilisateur, de création de CL-File ainsi qu'à la base de données de CCA.

Lorsque l'utilisateur effectue un choix de menu afin de définir un CCA d'un type particulier, l'interface API graphique liée à cette sélection de menu fait appel au générateur de CCA en lui passant le type de CCA à créer. Le générateur fait afficher la boîte de dialogue correspondante par l'interface API graphique après avoir consulté la base de données de CCA afin d'en retirer les informations associées. Il attend ensuite les sélections de l'utilisateur afin de récolter les données nécessaires à la création du CCA.

Dans le cas d'une sélection de surfaces ou de caractéristiques de modèle, le générateur de CCA fait appel à l'interface API du modeleur afin d'obtenir les informations géométriques ou dimensionnelles requises. De plus, le générateur de CCA doit recueillir de l'information par l'entremise de l'interface API aux données du modèle afin de connaître, par exemple, l'outil de coupe à utiliser pour l'usinage. Chacune des donnés recueillies suite aux actions de l'utilisateur est transférée à l'interface API graphique afin de faire afficher dans les

champs correspondants de la boîte de dialogue de définition du cycle les éléments qui sont sélectionnés.

Lorsque l'entrée d'information est complétée dans l'interface graphique, le générateur de CCA vérifie la validité des informations recueillies à l'aide des données retirées de la base de données de CCA, tel qu'illustré à la figure 7. Dans le cas où l'information est incorrecte, l'utilisateur en est averti à l'aide de l'interface API graphique. Si l'information est acceptable, le générateur de CCA fait appel à l'interface API de création de CL-File afin d'y ajouter les données selon le format contenu dans la base de données de CCA, puis sauvegarde l'information du CCA créé ainsi que ses liens avec les caractéristiques du modèle par l'entremise de l'interface API aux données. Le contrôle est alors retourné du MICCA au logiciel FAO.

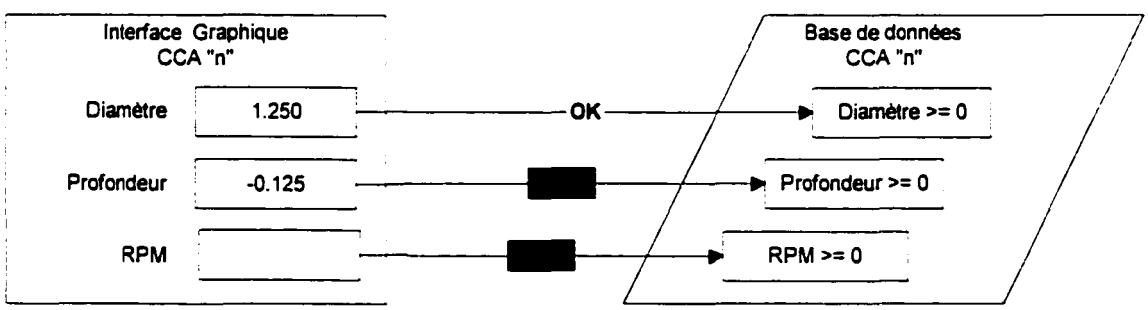


Figure 7 Validation de l'information d'un CCA

Les fonctionnalités du MICCA doivent permettre minimalement à l'utilisateur de créer, de modifier et de supprimer des CCA, et ce, au niveau du CL-File et des données conservées dans le modèle. De plus, une fonctionnalité de post-processeur inversé (PPI) doit être ajoutée de manière à fermer la boucle d'informations FAO – CCA – MOCN. Lorsque l'utilisateur du système de FAO désire effectuer la lecture du fichier de code G afin de mettre à jour le modèle FAO, il effectue des sélections de menu qui indiquent à l'interface API graphique d'appeler le module PPI. Une boîte de dialogue d'ouverture de fichier est ainsi affichée et permet la sélection du programme d'usinage à lire.

Lorsque le fichier de code G a été sélectionné, son contenu est par la suite envoyé au PPI qui effectue une vérification de la concordance de l'ordre des CCA qu'il contient avec le modèle FAO. Le nombre de CCA présent dans le programme d'usinage, leur ordre et leur type sont obtenus du fichier de code G par le PPI à l'aide de l'information obtenue de la banque de données de CCA. De même, les informations sur ces cycles définis dans le module de FAO sont extraites par l'entremise de l'interface API aux données. Les CCA du module de FAO sont par la suite comparés à ceux du programme d'usinage afin d'assurer que leur nombre, ordre et type concordent.

Dans le cas où aucune concordance n'est détectée, l'opération de post-processeur inversé est abandonnée. Dans le cas contraire, les paramètres de chaque CCA sont extraits à tour de rôle du programme en code G par le PPI. Chacun des paramètres des CCA sont comparés à ceux contenus dans le modèle FAO, obtenus par l'interface API de données. Les différences sont notées et chaque élément géométrique ou dimensionnel pour lequel les données du modèle nécessitent une mise à jour par rapport au programme en code G est modifié, tel que l'illustre la figure 8.

Lorsque l'ensemble des CCA du programme d'usinage ont été traités, l'interface API graphique est appelée afin d'afficher un sommaire des modifications qui ont été apportées au modèle.

2.2.1.3 Extensibilité

Il est évident que les efforts exigés par le développement d'un MICCA sont relativement élevés, particulièrement dans le cas de compagnies peu familières avec le développement avancé de fonctionnalités à l'intérieur d'un système de FAO par le biais d'API. Afin de permettre l'ajout de fonctionnalités au module développé de manière efficace, il est nécessaire de concevoir les modules constituants le MICCA de manière à ce que leurs in-

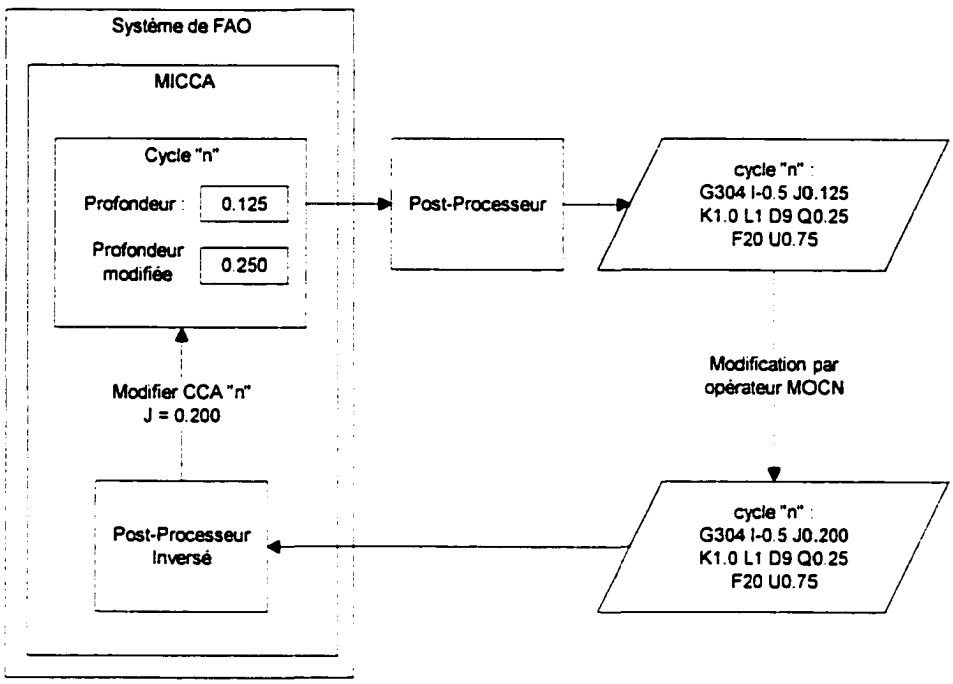


Figure 8 Post-traitement inversé.

terfaces permettent des possibilités d'expansion. Ces extensions pourront permettre d'exploiter d'avantage les particularités et fonctionnalités des MOCN disponibles.

À titre d'exemple, le module de PPI peut être étendu de manière à permettre la mise à jour des vitesse de broche et d'avance des séquences d'usinage natives du système de FAO ayant déjà fait l'objet de modifications par l'opérateur de MOCN. De même, il a été mentionné que la procédure utilisée par bon nombre de systèmes intégrés de fabrication pour utiliser les fontionnalités d'interpolation NURBS des MOCN est réalisée au niveau du post-processeur. Ce module tente de reconstruire des courbes NURBS à partir d'une succession de segments linéaires générés par le logiciel de FAO. Il en est de même des interpolations circulaires qui sont parfois approximées par une succession de lignes lors de la création du CL-File. Il est toutefois possible d'accéder à la définition mathématique des éléments géométriques d'un modèle CAO à l'aide des API du modeleur géométrique. Ceci ouvre la voie à la génération directe de NURBS ou d'arcs de cercles à partir des équations

définissant le modèle, ce que peu de systèmes FAO font directement. Il en résulte une élimination des erreurs associées à une conversion NURBS ou en arcs de cercles générés par le post-processeur à partir de segments de lignes.

2.2.2 Intégration au post-processeur

Les informations ajoutées au CL-File par le module de génération de CCA doivent être traitées par le post-processeur afin de générer dans le programme d'usinage les codes G correspondants. Puisque les post-processeurs ne supportent pas nativement les cycles cannés avancés, il est nécessaire de développer des macros qui vont effectuer la gestion des cycles afin d'en tirer les codes G à partir de leur description en langage APT. La personnalisation du post-processeur doit être réalisée en liaison avec les informations qui ont été entrées dans la base de données de CCA du module de FAO et qui dictent le format APT de représentation des cycles.

Pour un parc de machines-outils donné, il existe autant de post-processeurs que de machines-outils différentes. Pour chacun de ces post-processeurs, il faut développer autant de macro de CCA que le nombre qui est rendu accessible au module de FAO. Par exemple, si la base de données dans le module FAO offre "n" CCA pour la MOCN "X", il faut développer "n" macros de CCA dans le post-processeur de la MOCN "X", et ce, pour chaque MOCN intégrée au logiciel de FAO. La structure générale résultante est illustrée à la figure 9 pour "N" MOCN et "n" CCA pour chaque MOCN.

Chacune des macros développées qui définit les CCA d'une MOCN doit intercepter celles-ci en format APT présents dans le CL-File et générer les codes G associés aux CCA dans le programme d'usinage. Une vérification de la valeur des arguments des CCA présents dans le CL-File doit également être effectuée de manière à assurer leur validité. Une conversion des valeurs peut être nécessaire dans le cas où les arguments que nécessite la représentation

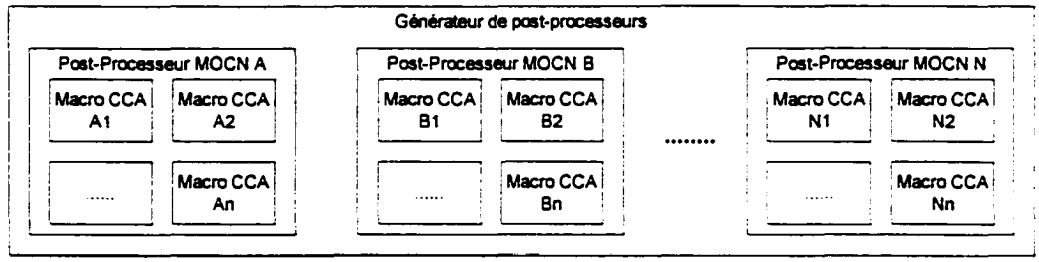


Figure 9 Schémas de personalisation du post-processeur

en code G d'un CCA pour une MOCN en particulier ne sont pas les mêmes que ceux utilisés dans le format unifié de représentation des CCA au niveau du CL-File.

Par exemple, l'ensemble des CCA d'usinage extérieur d'îlots rectangulaires peut avoir une représentation unifiée en APT qui indique les dimensions finale de l'îlot ainsi que la largeur de matériel à enlever tout autour de cette géométrie finale. Pour une certaine MOCN, le même type d'arguments peut être nécessaire alors que pour une autre machine, il peut être nécessaire d'indiquer les dimensions initiales et finales de l'îlot. Il relève du rôle de la macro de CCA d'obtenir les bons arguments nécessaires au code G à partir de l'information contenue dans le CL-File.

Les macros de CCA doivent également obtenir les états modaux qui seront actifs dans le contrôleur de la MOCN au moment de la lecture des CCA, afin de prendre des actions en conséquence. Il est par exemple possible qu'un certain CCA ne puisse être utilisé sur une MOCN que si la compensation d'outil en diamètre est inactive. Si une telle compensation a été activée par une opération d'usinage située avant le CCA dans le CL-File, la macro doit alors désactiver la compensation, générer le CCA dans le programme en code G puis procéder à une ré-activation de la compensation.

Il est évident que la manière de générer les macros nécessaires ainsi que leur architecture est fortement liée au générateur de post-processeur utilisé, du langage de programmation associé et des limitations inhérentes à cette plate-forme particulière.

CHAPITRE 3

APPLICATION MICCA

Afin de permettre une validation des concepts et de la méthodologie d'intégration des cycles cannés avancés présentée au chapitre 2, un exemple d'application à été réalisé. Les sections suivantes présentent le développement complet d'un module d'intégration des CCA au niveau d'un logiciel de FAO et d'un générateur de post-processeur. Les limitations d'intégration rencontrées lors de l'intégration logicielle, dûes à la nature propriétaire des environnements de développement utilisés, seront illustrées.

3.1 Intégration FAO

Afin de réaliser l'intégration FAO, le logiciel Pro/Engineer de Parametric Technology Corporation (PTC) à été utilisé. Ce système CFAO complètement intégré offre les avantage d'incorporer deux modules de génération de post-processeurs, CAM-POST de ICAM ainsi que G-POST de Intercim. De plus, deux simulateurs d'usinage sont inclus avec Pro/Engineer, soit PRO/NC-CHECK de PTC, un simulateur de type I, ainsi que Vericut de CG-TECH, un simulateur également de type I mais qui peut être être converti en type II lorsque celui-ci est acheté indépendamment.

Le logiciel Pro/Engineer est un système CFAO de haut niveau utilisé par bon nombre de multi-nationales telles que Caterpillar, Pratt & Whithey, Lockheed Martin, Volvo ainsi que la NASA, pour ne citer que celles-ci. La vaste librairie d'API disponibles sous l'appellation Pro/TOOLKIT permet un développement logiciel présentant des possibilités relativement larges. Il est entre autres possible d'accéder le modeleur solide, le générateur de CL-File, l'interface graphique ainsi que le reste des données du modèle FAO qui ne sont pas accessibles par l'entremise du modeleur solide. De plus, il est possible d'inclure aux fichiers qui contient le modèle FAO des informations qui sont générées par le MICCA. L'ensemble

des fonctionnalités nécessaires à l'intégration des CCA sont ainsi présentes, ce qui permet la réalisation de l'intégration.

De plus, Pro/Engineer utilise un modeleur solide propriétaire basé sur les caractéristiques, permettant ainsi de simplifier l'intégration des CCA ainsi que leur associativité à la géométrie. Plus précisément, le module Pro/MANUFACTURING de la version $2000i^2$ de Pro/Engineer a été utilisé afin de réaliser, à l'aide des API de Pro/TOOLKIT, l'intégration proposée. Le positionnement de l'application développée à l'intérieur de Pro/ENGINEER est présenté à la figure 10 et permet d'illustrer les liens entre cette application et les autres modules du logiciel.

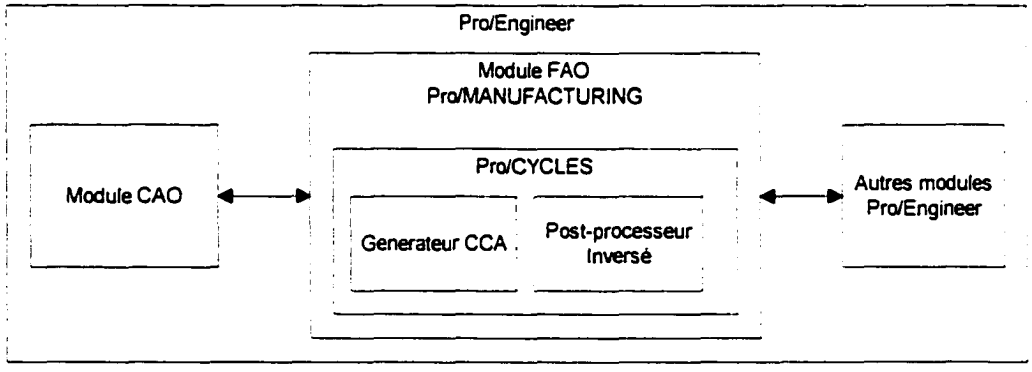


Figure 10 Positionnement du module d'intégration dans Pro/Engineer.

L'ensemble des CCA qui sont utilisés sont ceux d'une MOCN Hitachi-Seiki modèle VS-50 à laquelle est intégrée une table à deux axes rotatifs Nikken permettant de réaliser de l'usinage 5 axes simultané. Le contrôleur monté à cette machine est un Fanuc 16i, un des plus performants actuellement disponibles en industrie. Il offre nativement des fonctionnalités d'usinage à haute vitesse, de "look-ahead" ainsi que d'interpolation NURBS. De plus, il offre la possibilité de transférer les programme d'usinage à l'aide d'une interface Ethernet utilisant un protocole FTP en plus du transfert série RS-232. Les programmes ainsi transférés sont stockés dans la mémoire du contrôleur ou encore sur un disque dur intégré à ce même contrôleur.

De plus, le contrôleur peut intégrer, en option, un processeur Pentium qui permet d'opérer des logiciels divers à travers l'interface de la MOCN, ouvrant la voie au développement d'un degré d'intégration encore plus élevé. Le Fanuc 16i présente également une interface PCMCIA qui permet, outre le transfert de programmes d'usinage, la mise à jour du contrôleur et la sauvegarde des paramètres de la MOCN. Une interface de programmation conversationnelle utilisée afin de générer les programmes d'usinage est également disponible. Afin d'assurer une ouverture sur l'évolution des CCA et de l'intégration FAO – MOCN, ce contrôleur représente un choix judicieux.

Il est toutefois nécessaire de s'intéresser aux CCA offerts par ce contrôleur et qui pourront être intégrés au module de FAO de Pro/Engineer. Un nombre élevé de cycles cannés avancés ont étés intégrés par Hitachi-Seiki au Fanuc 16i, renommé Seicos, afin d'augmenter les possibilités de la MOCN VS-50. Les possibilités de développement en langage C offertes nativement par Fanuc ont été utilisées par Hitachi-Seiki afin d'inclure des CCA qui permettent la prise de mesures de pièce, le positionnement du zéro pièce ainsi que l'usinage de géométries prismatiques. Toutefois, l'environnement en langage C n'est accessible que par la compagnie qui achète ce contrôleur afin de l'intégrer à sa MOCN, limitant du même coup l'évolutivité du contrôleur et de la MOCN en milieu industriel.

3.1.1 Détails généraux de l'intégration

L'accessibilité offerte par les API d'un logiciel de FAO est toujours limitée par la compagnie qui développe ce logiciel, et ce, en restreignant les fonctionnalités des API offerts. Ainsi, Pro/Engineer ne permet pas la création de séquences d'usinage différentes de celles offertes nativement dans le module de FAO. Afin de pouvoir générer des séquences de CCA, il a donc été nécessaire de modifier un type existant de séquence d'usinage afin de permettre l'inclusion des CCA au logiciel.

Il convenait donc de choisir un type de séquence spécifique et d'en modifier les données afin d'en obtenir une séquence de CCA. Le type "customize" de Pro/MANUFACTURING se prête particulièrement bien à une modification pour les séquences de type CCA car il permet de choisir l'outil utilisé pour l'usinage, le système de coordonnées ainsi que le plan de retrait de l'outil, tout en générant l'information associée dans le CL-File. L'utilisation primaire de ce type de séquence est de définir par édition manuelle les trajectoires d'outils dans le CL-File "vide" ainsi généré.

Le déroulement général de la génération d'une séquence de CCA consiste ainsi à définir une séquence de type "Customize" en utilisant les fonctionnalités natives du logiciel. Cette séquence est par la suite convertie en CCA à l'aide des fonctionnalités du MICCA. Cet état de fait constitue la première limitation résultant de l'utilisation de Pro/ENGINEER afin de développer une interface d'intégration de CCA. En effet, deux étapes sont nécessaires à la définition de tout cycle canné avancé, soit la création d'une séquence d'usinage de type "Customize" suivie d'une conversion en séquence CCA qui a pour but ultime de produire le code nécessaire dans le CL-File.

Suite à la création de la séquence "Customize", la conversion de cette séquence en CCA est permise par l'entremise de choix de menus rendus disponibles par le module développé à l'aide de Pro/TOOLKIT, nommé Pro/CYCLES. Ce choix de menu permet de sélectionner, parmi les types de CCA disponibles, celui auquel sera converti la séquence. Selon le type de CCA choisi par l'utilisateur, une boîte de dialogue correspondante permet l'entrée des paramètres nécessaires à la définition du cycle. Suite à l'entrée des paramètres nécessaires, l'information résultante est ajoutée au CL-File appartenant à la séquence de type "Customize" correspondante. De plus, les données définissant le CCA généré sont incorporées au modèle CAO à l'aide des API permettant l'accès au fichier de sauvegarde du modèle en cours d'édition.

La définition de cette démarche sommaire de création de CCA permet de mieux comprendre les détails de chaque étape de création d'un CCA dans Pro/Engineer.

3.1.2 Particularités de l'intégration

Afin de cerner de manière acceptable et compréhensible l'intégration des CCA réalisée au niveau du logiciel Pro/ENGINEER, il est nécessaire de s'attarder à certains aspects de Pro/CYCLES. Les sections suivantes ont pour but de présenter certains détails sur les principaux modules de Pro/CYCLES qui sont nécessaires à la compréhension du fonctionnement de ce MICCA. De plus, les limitations de l'intégration proposée pourront être mieux cernées.

3.1.2.1 Base de données CCA

Bien qu'une seule MOCN soit utilisée dans cette implantation de CCA dans le logiciel de FAO, il convient tout de même de représenter l'ensemble des CCA disponibles dans une base de données tel que présenté à la section 2.2.1.1. L'environnement de développement Pro/TOOLKIT est basé sur du C ANSI qui est également utilisé afin de créer l'application MICCA Pro/CYCLES. Ainsi, les informations concernant les CCA du contrôleur de la MOCN VS-50 ont été décrites dans un ficher d'entête ".h" afin d'être compilées à même l'application créée.

Cette approche a pour but de prévenir la modification des informations des CCA par les utilisateurs du MICCA. Ainsi, seuls les programmeurs de Pro/CYCLES ont accès aux définitions des cycles cannés avancés, ce qui permet d'éliminer les risques de corruption de données et d'erreurs associés à l'accès libre aux données.

De plus, puisque la programmation de l'application Pro/CYCLES est vouée à des fins de démonstration de la fonctionnalité d'un tel type d'intégration, seuls les CCA d'usinage de géométries cylindriques ont étés inclus. En effet, cette intégration est suffisante pour permettre la réalisation d'essais de génération de trajectoires d'outils et d'usinage compa-

ratifs. L'implantation des quelques 45 CCA offerts sur la MOCN VS-50 n'est donc pas nécessaire dans le cadre de ce travail. Le tableau IV présente les cycles faisant l'objet d'implantation.

Tableau IV

CCA de la MOCN Hitachi-Seiki implantés

Géométrie	Type
Pochette Circulaire	Niveaux en Z Niveaux en Z et îlot central Pleine Profondeur Hélicoïdale Hélicoïdale conique
Îlot Circulaire	Niveaux en Z Pleine profondeur

Les informations incluses dans la base de données des CCA comprennent les paramètres nécessaires à la génération des codes G associés aux cycles intégrés à Pro/CYCLES ainsi qu'au post-traitement inversé.

3.1.2.2 Interface graphique

L'interface graphique nécessaire à l'utilisation des fonctionnalités de Pro/CYCLES est divisée en deux sections, soit les menus et les boîtes de dialogue. L'accès aux fonctionnalités du MICCA est réalisé par l'ajout de nouveaux choix de menus à ceux qui sont présents dans la version native du module Pro/MANUFACTURING. L'interface originale des menus de Pro/MFG est présentée à la partie gauche de la figure 11 alors que les menus incluant l'accessibilité à Pro/CYCLES est illustrée dans la partie de droite de cette même figure. Les choix additionnels de menus permettent de réaliser la conversion d'une séquence de type "customize" en une séquence de CCA, la redéfinition d'une séquence de

CCA existante, la conversion inverse d'une séquence de CCA en séquence de type "customize" ainsi que la réalisation du post-traitement inversé.

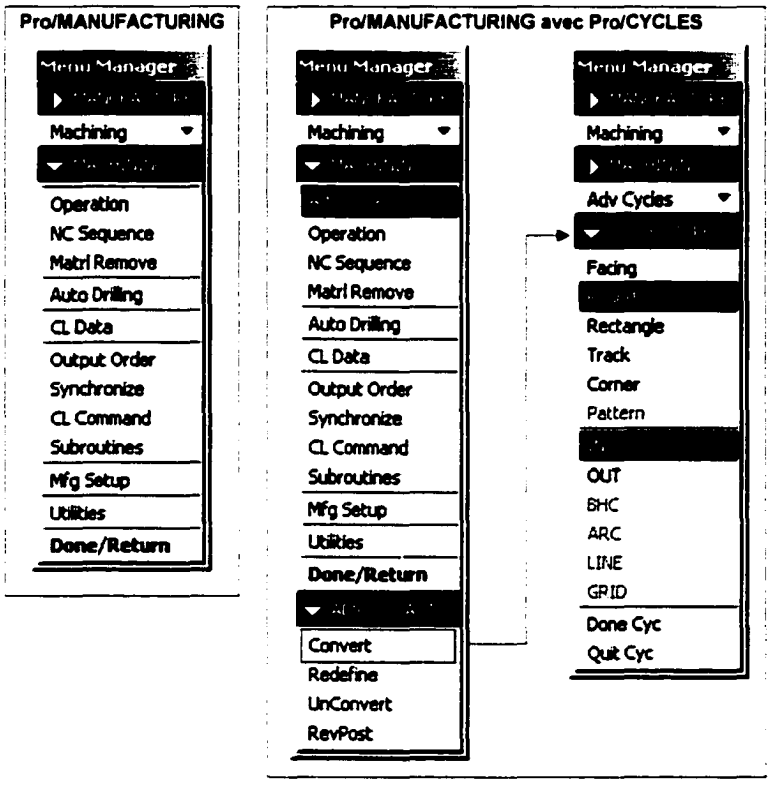


Figure 11 Menus du module FAO Pro/Manufacturing.

Dans le cas d'une conversion en CCA d'une séquence customize, l'utilisateur doit choisir le type de géométrie à usiner de manière à accéder les CCA disponibles sur la MOCN sélectionnée lors de la définition de l'opération d'usinage dans Pro/MFG. La géométrie à usiner à été choisie comme critère initial de définition de la conversion en CCA car une telle approche permet à l'utilisateur, lors d'une redéfinition d'une séquence de CCA, de changer le CCA pour tout autre permettant d'usiner le même type de géométrie, augmentant ainsi la flexibilité de l'intégration. De plus, une telle architecture de menu permet à l'utilisateur d'utiliser une approche à la génération des séquences de CCA basée sur

la géométrie à usiner sans avoir à mémoriser le nom des CCA associés, ce qui est plus efficace.

Lors de la sélection du type de géométrie à usiner lors de la conversion d'une séquence "customize" en CCA ou lors de la redéfinition d'une séquence CCA existante, une boîte de dialogue, telle que celle illustrée à la figure 12, est utilisée afin de permettre l'introduction des paramètres d'usinage ainsi que des éléments géométriques de référence du CCA en cours de définition. Une boîte de dialogue par type de géométrie à usiner est nécessaire, soit l'usinage circulaire intérieur et l'usinage circulaire extérieur dans le cas de l'intégration courante.

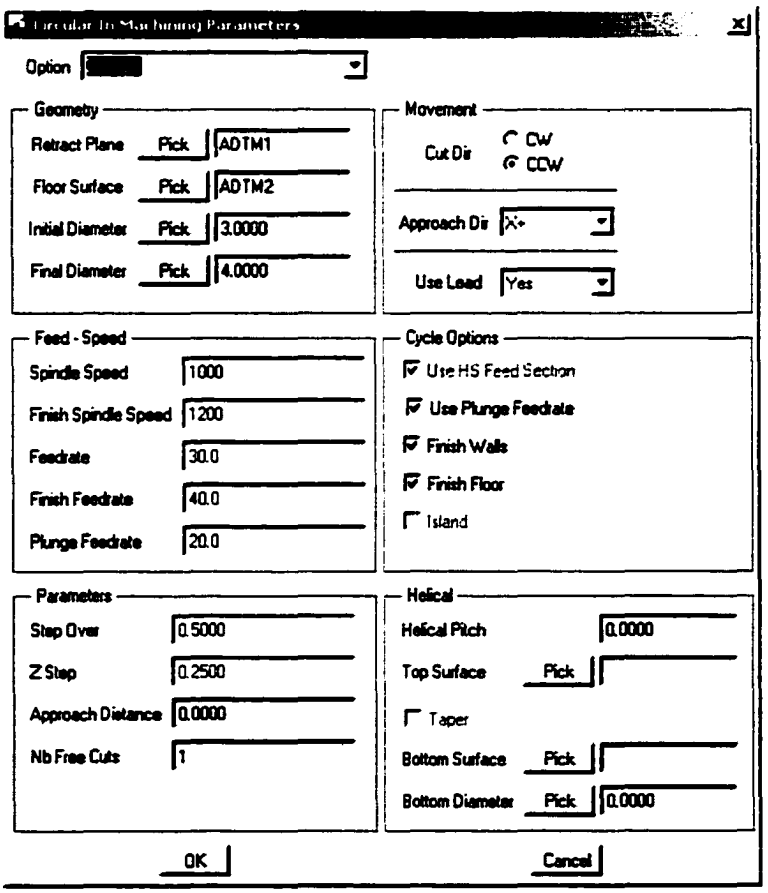


Figure 12 Boîte de dialogue associée à un usinage circulaire intérieur

L'approche utilisée afin de définir les boîtes de dialogues à l'aide des API de Pro/TOOL-KIT permet d'indiquer le type de données que peut contenir chaque champ ainsi que les limites permises. Ceci permet d'inclure une vérification primaire des données à même la définition de la boîte de dialogue.

Lors de la sélection graphique des éléments géométriques tels que le plan de rétraction ou le diamètre initial à usiner, une vérification des éléments sélectionnés est effectuée selon les informations de la base de données de CCA de manière à permettre la sélection d'éléments qui sont consistants avec le type de CCA en cours de définition. Par exemple, le plan de retrait doit être normal à l'axe Z de l'opération d'usinage courante et les diamètres initiaux et finaux d'usinage doivent posséder un axe parallèle à ce même axe Z du système de coordonnées courant.

La validation de toutes les autres données est effectuée lorsque l'utilisateur appuie sur le bouton "OK" de la boîte de dialogue afin d'indiquer la fin de la saisie des données. Les informations sont alors validées selon le contenu de la base de données des CCA, et toute incompatibilité des données entrées est signalée à l'utilisateur par l'entremise d'un message. Dans le cas où les données sont satisfaisantes, les informations contenues dans la boîte de dialogue sont transférées au modèle FAO, les lignes nécessaires au CCA sont ajoutées automatiquement au CL-File de la séquence "customize" qui devient alors de type CCA et l'opération est complétée. La figure 13 présente les ajouts qui sont effectués au CL-File d'une séquence "customize", initialement vide.

Lors d'une opération de redéfinition d'une séquence de CCA, les paramètres du CCA visé sont lus à partir du modèle FAO dans lequel ils avaient été sauvegardés, puis sont inscrits dans les champs correspondants de la boîte de dialogue qui avait servi à la définition initiale du CCA. La boîte de dialogue contenant les données est alors affichée afin de permettre à l'utilisateur d'apporter les modifications nécessaires. La validation des informations utilisées lors de la création du CCA est également appliquée lors de la redéfinition

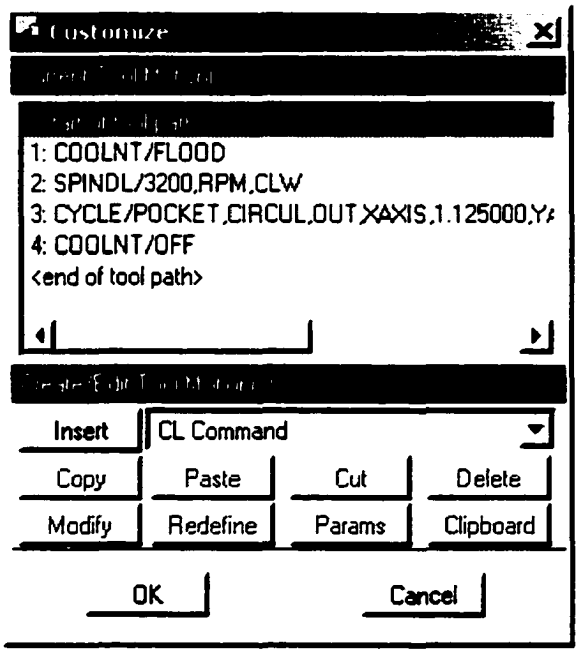


Figure 13 Séquence "Customize" modifiée.

afin d'assurer l'intégrité des données passées au modèle FAO et utilisées afin de générer les commandes APT du CL-File.

Afin de permettre la suppression des séquences d'usinage à l'aide des fonctionnalités de Pro/MANUFACTURING, il est nécessaire d'utiliser le choix de menu de conversion inverse de Pro/CYCLES. En effet, Pro/MFG ne peut supprimer directement une séquence de CCA car le logiciel ne connaît pas l'ensemble des données qui ont été ajoutées au modèle FAO lors de la création de la séquence CCA. La fonctionnalité de conversion inverse a pour rôle de supprimer du modèle FAO toutes les informations de CCA de la séquence sélectionnée ainsi que d'effacer les commandes ajoutées au CL-File. Suite à la conversion inverse, une séquence de CCA retrouve l'état d'une séquence "customize" tel que présente avant la conversion initiale. Il est par la suite possible d'effacer la séquence "customize" à l'aide des fonctions natives de Pro/MFG. Les deux étapes nécessaires à la réalisation d'une suppression de CCA découlent de la restriction des API de Pro/TOOLKIT qui ne

permettent pas de créer des séquences d'usinage différentes de celles disponibles nativement dans le logiciel Pro/ENGINEER.

3.1.2.3 Ajout de données au modèle FAO

Tel que mentionné dans les sections précédentes, il est nécessaire de conserver les données associées au CCA définies directement dans le fichier du modèle FAO afin d'assurer l'existence de ces séquences au delà de la séance de travail dans laquelle elles ont été définies. L'accès au modèle FAO afin d'y ajouter des données est relativement restreint par les API offerts par la majorité des logiciels CFAO, et Pro/ENGINEER ne fait pas exception à cette règle. Ainsi, les seuls types de données qu'il est possible d'ajouter aux modèles par l'entremise de Pro/TOOLKIT sont des entiers, des doubles, des chaînes de caractères et des trains de bits. Ce dernier type de donnée à été utilisé afin d'emmagasiner les informations des CCA dans le modèle Pro/ENGINEER car il permet de transférer l'information des séquences de CCA directement au modèle dans la forme o ;u elle est utilisée dans Pro/CYCLES.

Puisque le langage de programmation C ANSI est utilisé par Pro/TOOLKIT, les informations entrées dans la boîte de dialogue de définition d'une séquence de CCA sont emmagasinées dans une structure de données en langage C. Un seul type de structure pouvant convenir à tous les types de CCA est défini dans Pro/CYCLES et passé directement au modèle FAO sous forme d'un train de bits dont la principale limitation concerne la taille maximale, qui ne peut excéder 512 kilo-octets par train de bit défini. Ainsi, l'ensemble des informations représentant un CCA ne peut dépasser cette taille, ce qui peut devenir problématique dans le cas de CCA complexes et peut forcer l'usage de plus d'un type de structure de données afin d'optimiser l'utilisation de l'espace alloué par Pro/TOOLKIT pour conserver de l'information externe.

L'interface aux données illustrée à la figure 6 utilise les API de Pro/TOOLKIT de manière à inclure les informations des séquences de CCA au modèle Pro/ENGINEER lors de la définition de ces séquences. De même, les API sont utilisées pour aller lire le train de bit associé à une séquence CCA qui doit être redéfinie afin de permettre d'afficher les paramètres du CCA existant dans la boîte de dialogue correspondante. Lors d'une conversion inverse, l'interface aux données permet l'effacement du train de bit d'une séquence CCA du modèle FAO.

Finalement, lors d'une opération de post-traitement inversé, le module assurant ce rôle utilise l'interface aux données afin de mettre à jour les informations des séquences de CCA à partir du fichier de code G qui peut avoir été modifié à même le contrôleur de la MOCN utilisée pour réaliser l'usinage.

3.1.2.4 Ajout de données au CL-File

Le module de post-processeur génère les codes G des CCA définis dans Pro/ENGINEER à partir des informations qui sont présentes dans le CL-File traité. L'utilisation des API de Pro/TOOLKIT permet d'ajouter les lignes de code APT qui définissent les CCA pour chaque séquence de ce type. C'est à partir des paramètres et de la géométrie d'un CCA présents dans une boîte de dialogue que Pro/CYCLES génère les commandes APT selon les informations présentes dans la base de données de CCA.

Puisque le format standardisé d'un CL-File exige qu'aucune ligne d'un tel fichier ne possède plus de 80 caractères, une fonction permettant de briser la commande APT d'un CCA en lignes ne dépassant pas cette limite est nécessaire. Suite à l'obtention des lignes de code de longueur convenable à partir de la commande APT, l'information est inscrite dans le CL-File de la séquence de CCA qui pourra alors être acheminée vers le post-processeur afin d'obtenir le programme d'usinage.

3.1.2.5 Post-processeur inversé

Lors de la sélection du choix de menu de post-processeur inversé, une boîte de dialogue permettant la sélection du fichier de code G à traiter est présentée à l'utilisateur. Le fichier choisi doit être celui qui contient l'ensemble du programme d'usinage défini dans Pro/MFG et il doit inclure toutes les séquences d'usinage définies dans ce modèle FAO. Une vérification est effectuée afin de déterminer si le fichier choisi contient des séquences de type CCA. Si c'est le cas, le nombre de séquences de type CCA, leur type ainsi que leur ordre est comparé avec ce que contient le modèle FAO actif. Si les données concordent, le processus de post-traitement inverse débute.

Chacun des CCA définis dans le fichier de code G est transféré dans une structure de données utilisée pour représenter les séquences de CCA lors de leur définition initiale. Puis ces données sont transférées au modèle FAO sous la forme d'un train de bits afin de mettre à jour les informations des séquences de CCA sauvegardées dans le fichier Pro/MFG. Par l'entremise des informations de la base de données de CCA, le CL-File de chaque séquence de type CCA est redéfini par les informations du fichier de code G contenues par les structures de données.

Dans le cas où le fichier de code G choisi ne concorde pas sur un ou plusieurs critères avec les informations de CCA contenues dans le modèle FAO, un message indique à l'utilisateur du logiciel que le processus de post-traitement inversé ne peut pas être réalisé. Ceci permet d'éviter la corruption de données qui résulterait du traitement d'un programme d'usinage provenant d'un autre modèle FAO que celui en cours d'édition. De plus, un programme de code G qui aurait été fortement modifié par l'opérateur de la MOCN ne pourra pas être traité. Ceci constitue une sécurité dans le cas où l'ordre des CCA aurait été altéré dans le programme d'usinage.

3.2 Module Post-Processeur

Le module de post-processeur utilisé afin de compléter la réalisation de Pro/CYCLES et de compléter le lien FAO – MOCN est basé sur le générateur de post-processeur CAM–POST de la compagnie ICAM. Ce module à été préféré à G-Post, également offert avec Pro/Engineer $2000i^2$, car il présente des possibilité de développement et de programmation plus vastes. De plus, la solution CAM–POST est une plate-forme de haut niveau capable de gérer des MOCN comportant jusqu'à 15 axes de déplacement et représente le standard en terme d'environnement de développement de post-processeur utilisé en industrie.

L'intégration des CCA aux post-processeurs des MOCN est relativement plus simple et directe que le développement Pro/TOOLKIT présenté à la section 3.1. Il suffit en effet de créer des macros associées à la définition APT de chaque CCA utilisé sur une MOCN donnée. Ces macros sont appelées lorsque le post-processeur interceptera une ligne de CCA dans le CL-File qui correspond à une macro. Cette dernière se charge alors d'inscrire dans le fichier de code G en cours de génération le code associé au CCA pour la MOCN à laquelle appartient le post-processeur. Une partie de l'interface de développement de macros de CAM-POST est présentée à la figure 14, on peut y remarquer les macros de CCA qui ont été créées.

À titre d'exemple, lors de la création d'un CCA d'usinage hélicoïdal de pochette circulaire intérieure dans le module Pro/CYCLES, la syntaxe suivante est générée par le MICCA dans le CL-FILE :

COOLNT/THRU
SPINDL/5300,RPM,CLW
CYCLE/HELICL,CCLW,HSM,DEPTH,-1.562500,DIAMET,1.750000,ZSTEP,0.079000,
FEDRAT,30.000000,XAXIS,2.750000,YAXIS,2.000000,CLEAR,-0.312500,
HIGH,-0.562500,BTMDIA,1.250000,LOW,-1.562500,FINCUT,1
COOLNT/OFF

Figure 14 Interface de développement de post-processeurs CAM-POST

Le post-processeur, à l'aide des macros qui ont été développées, va produire les commandes en code G associées à ce CCA spécifique à la MOCN VS-50 Hitachi-Seiki :

M50

M3S5300

G813 X1.1250 Y2.8750 Z-0.8125 I0.7500 J0.7500 K-0.8125 R-0.3125 L1 D3.

Q0.136 F21., R-0.5625

G80

M09

Il doit donc exister autant de macros dans un post-processeur spécifique que le nombre de CCA qui sont accessibles pour cette machine dans le logiciel de FAO. Les CCA d'usinage de forme circulaire qui ont été intégrés à Pro/CYCLES sont présentés au tableau V.

Tableau V

Détails des CCA circulaires implantés

Géométrie	**Type**	**Sous-Type**	Direction	Code G
Pochette	Pleine profondeur		CW	G302
	Pleine profondeur		CCW	G303
	Niveaux en Z		CW & CCW	G327
	Niveaux en Z	Approche Tangentielle	CW & CCW	G333
	Hélicoïdal	Conique ou cylindrique	CW	G812
	Hélicoïdal	Conique ou cylindrique	CCW	G813
Îlot circulaire	Pleine profondeur		CW	G304
	Pleine profondeur		CCW	G305
	Niveaux en Z		CW & CCW	G330

Il doit donc exister 9 macros de CCA dans le post-processeur de la MOCN VS-50, à laquelle appartiennent ces cycles. Chacune des macros vérifie la validité des arguments de la définition APT des CCA avant de générer le code G correspondant dans le programme d'usinage résultant. Dans le cas où une erreur est détectée dans un CCA du CL-File, un message est affiché afin d'indiquer à l'opérateur la source de l'erreur, ce qui permet de prendre les actions correctrices correspondantes dans le module Pro/CYCLES de Pro/MANUFACTURING.

CHAPITRE 4

EXPÉRIMENTATION

Afin d'apprécier les avantages offerts par l'intégration des CCA réalisée avec Pro/ CYCLES, il est nécessaire de réaliser des essais d'usinage desquels seront tirés des valeurs qualitatives et quantitatives qui permettront de détailler les performances du MICCA réalisé. Puisque le module développé dans le cadre de ce projet permet seulement la définition de CCA permettant l'usinage de géométries prismatiques circulaires, la pièce d'essai proposée est limitée à des caractéristiques qui présentent ce type de géométrie.

4.1 Détails de l'expérimentation

Une pièce comportant diverses caractéristiques de géométrie circulaire a été modélisée à l'aide du module CAO de Pro/ENGINEER, tel qu'illustré à la figure 15. La stratégie d'expérimentation proposée consiste à effectuer la génération des trajectoires d'outils nécessaires à l'usinage de cette pièce sur la MOCN VS-50 à l'aide de Pro/MANUFACTURING dans un premier temps puis à l'aide de Pro/CYCLES par la suite. L'usinage de la pièce d'essai modélisée sera effectué à l'aide des deux programmes d'usinage résultants afin de pouvoir effectuer une comparaison de la performance associée à l'utilisation des deux méthodes de génération des trajectoires. Il est à noter que les caractéristiques circulaires constituant la pièce d'essai permettent d'utiliser l'ensemble des CCA qui sont couramment disponibles dans Pro/CYCLES.

Afin d'assurer une comparaison équitable lors de la génération des trajectoires d'outils et de l'usinage des pièces d'essai, les mêmes outils de coupe ainsi que les mêmes paramètres d'usinage sont employés pour chacune des deux pièces. Un premier ensemble de trajets d'outils réalisés avec les fonctionnalités natives de Pro/MANUFACTURING servira d'étalon, puis un second ensemble de trajets d'outils faisant appel aux cycles avancés

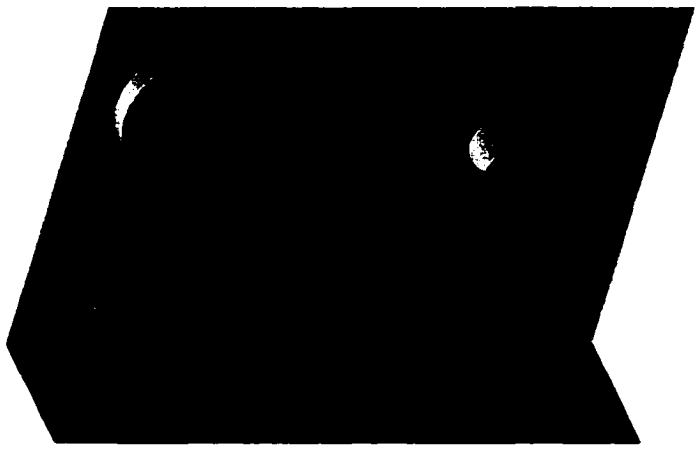


Figure 15 Pièce utilisée pour les essais d'usinage

et utilisant Pro/CYCLES sera réalisé en utilisant les paramètres retenus pour l'usinage du premier.

Les gains qui peuvent être obtenus par l'utilisation d'un MICCA sont présents à plusieurs niveaux du système de fabrication intégré. Il est donc nécessaire de définir l'ensemble des gains possibles qui deviendront ainsi des critères d'évaluation du module Pro/CYCLES créé. Les résultats obtenus pourront ainsi être extrapolés, avec une certaine limite, à toute intégration de CCA à un système FAO.

4.1.1 Critères d'évaluation

Bien que le but final de l'intégration d'un MICCA soit de produire une pièce physique par usinage ou de la mesurer, la majorité des critères d'évaluation permettant d'obtenir les gains de performance d'une telle intégration sont à caractère qualitatif.

En effet, le seul critère de type quantitatif concerne les gains possibles en temps. On peut à ce niveau comparer les temps nécessaires à la programmation des trajectoires d'outils

à l'aide des séquences d'usinage présentes nativement dans Pro/MANUFACTURING à ceux nécessaires à la génération des trajectoires en utilisant le module Pro/CYCLES développé. Les temps d'usinage sur la MOCN peuvent également être comparés. Dans le cas des temps de génération des trajectoires d'outils, les gains associés à l'utilisation du MICCA, si ils existent, vont permettre de témoigner de l'efficacité de l'application des CCA à la génération des programmes d'usinage. En ce qui concerne les temps d'usinage, un temps moindre pouvant résulter de l'utilisation du programme d'usinage avec CCA témoignerait alors de l'efficacité des algorithmes utilisés à même le contrôleur afin de générer les trajectoires d'outils associées au CCA.

Les autres critères d'évaluation sont purement qualitatifs et seront évalués suite à la réalisation de l'expérimentation. Il convient toutefois de les énumérer avant même que ne prennent place les essais d'usinage. Ainsi, les points suivants seront évalués :

- Facilité d'utilisation de Pro/CYCLES versus Pro/MANUFACTURING.
- Lisibilité du programme d'usinage en code G résultant
- Possibilités de modifications du programme d'usinage en code G
- Taille du programme d'usinage en code G
- Flexibilité du transfert de données FAO – MOCN
- Entretien FAO & Post-processeur avec et sans MICCA

4.1.2 Programmes d'usinage

L'ensemble des trajectoires d'outils qui permettent d'obtenir les programmes d'usinage de la pièce d'essai sont réalisées avec le module FAO de Pro/ENGINEER. Le post-processeur de la MOCN VS-50 développé dans l'environnement ICAM de CAM-POST est par la suite utilisé afin d'obtenir les fichiers de code G qui seront envoyés à la MOCN. Afin de permettre une comparaison valable des résultats, la stratégie d'usinage employée lors de

la création des deux ensembles de trajectoires d'outils, avec et sans Pro/CYCLES, se doit d'être similaire.

Ainsi, en plus d'employer les mêmes outils de coupe pour l'usinage des caractéristiques correspondantes des deux copies des pièces, les paramètres d'usinage employés, tels la vitesse d'avance et la profondeur de passe, sont similaires. Finalement, le type et l'ordonnancement des séquences qui ont été utilisées lors de la réalisation du programme d'usinage utilisant les fonctions natives de Pro/MFG ont été choisies de manière à ressembler à l'usinage qui est réalisé à l'aide des CCA de Pro/CYCLES. Ceci est rendu nécessaire de manière à fournir une base commune permettant la comparaison des performances des deux types de génération de trajectoires d'outils.

Les sections suivantes détaillent les stratégies utilisées pour la création des trajectoires d'outils, et ce, pour chacune des deux copies des pièces d'essai.

4.1.2.1 Programme d'usinage avec Pro/MANUFACTURING

Le premier programme d'usinage réalisé afin d'obtenir la pièce d'essai de la figure 15 utilise uniquement les fonctionnalités et les séquences d'usinage qui sont natives au module Pro/MANUFACTURING de Pro/ENGINEER. Les opérations qui permettent l'usinage de cette pièce sont décrites dans la liste suivante et réfèrent aux caractéristiques de la pièce d'essai définie à la figure 16. Pour chaque séquence d'usinage de la liste, le type associé qui est utilisé dans Pro/MANUFACTURING est inscrit entre parenthèses suite à la description de la séquence.

- 1a : Surfaçage entier du brut au niveau de la surface supérieure de l'ilot #1. (Face Milling)
- 2a : Dégrossisage du volume compris entre le niveau de surfacage et la surface #6, autour de l'îlot #1. (Volume Milling)

- 3a : Dégrossisage de la pochette #4. (Volume Milling)
- 4a : Dégrossisage de la pochette conique #3. (Volume Milling)
- 5a : Finition de la pochette conique #3. (Pocket Milling)
- 6a : Dégrossisage de la pochette #2. (Pocket Milling)
- 7a : Finition de la pochette #2. (Profile Milling)
- 8a : Finition du diamètre externe de l'îlot #1. (Profile Milling)
- 9a : Finition du diamètre externe de l'îlot #5. (Profile Milling)
- 10a : Finition de la pochette #4. (Pocket Milling)

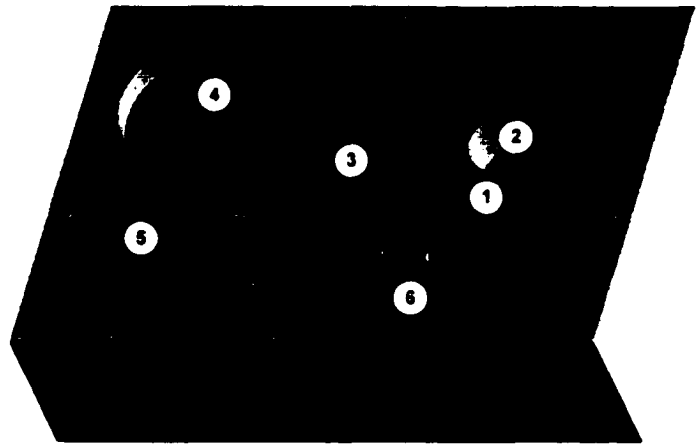


Figure 16 Caractéristiques de la pièce d'essai

Le type de séquences utilisées ainsi que leur ordonnancement est représentatif de ce qui pourrait être utilisé en industrie afin d'accomplir l'usinage d'une telle pièce. Un programme d'usinage de 1190 lignes résulte du post-traitement de ces séquences d'usinage.

4.1.2.2 Programme d'usinage avec Pro/CYCLES

Le second ensemble de trajectoires d'outils réalisées utilise, lorsque possible, les CCA implantés dans Pro/CYCLES. En effet, il convient de surfacer la pièce avant de débu-

ter l'ébauche des caractéristiques prismatiques circulaires, et le CCA de surfaçage de la MOCN VS-50 n'est pas implanté dans le MICCA développé. Une séquence de surfaçage native de Pro/MANUFACTURING, identique à celle utilisée lors de l'usinage de la première pièce est alors programmée. Une telle action ne fausse pas les résultats qui seront obtenus de l'expérimentation. De même, il est nécessaire de dégrossir le volume situé au dessus des pochettes #3, #4 ainsi que de l'îlot #5, tel que référencé à la figure 16. Une séquence d'usinage de type "Volume Milling" est alors utilisée afin de dégrossir cette région. Le reste de la pièce est entièrement usiné à l'aide des CCA d'usinage implantés.

La liste suivante décrit la stratégie d'usinage avec les CCA utilisés afin de réaliser l'usinage de la seconde pièce. Les identificateurs de caractéristiques font référence aux éléments de la figure 16.

- 1b : Surfaçage entier du brut au niveau de la surface supérieure de l'ilot #1. (Face Milling)
- 2b : Dégrossisage au niveau supérieur de l'îlot #5, dans la zone couvrant l'îlot #5 et les pochettes #3 et #4. (Face Milling)
- 3b : Dégrossissage du diamètre externe de l'îlot #1. (CCA Circulaire extérieur avec niveaux en Z, G330)
- 4b : Dégrossissage du diamètre externe de l'îlot #5. (CCA Circulaire extérieur avec niveaux en Z, G330)
- 5b : Dégrossissage de la pochette #4. (CCA Circulaire intérieur hélicoïdal, G813)
- 6b : Dégrossissage de la pochette #3. (CCA Circulaire intérieur hélicoïdal, G813)
- 7b : Finition de la pochette #3. (CCA Circulaire intérieur hélicoïdal, G813)
- 8b : Dégrossissage de la pochette #1. (CCA Circulaire intérieur hélicoïdal, G813)
- 9b : Finition de la pochette #1. (CCA Circulaire intérieur pleine profondeur, anti-horaire, G302)
- 10b : Finition du diamètre externe de l'îlot #1. (CCA Circulaire extérieur pleine profondeur, horaire, G304)

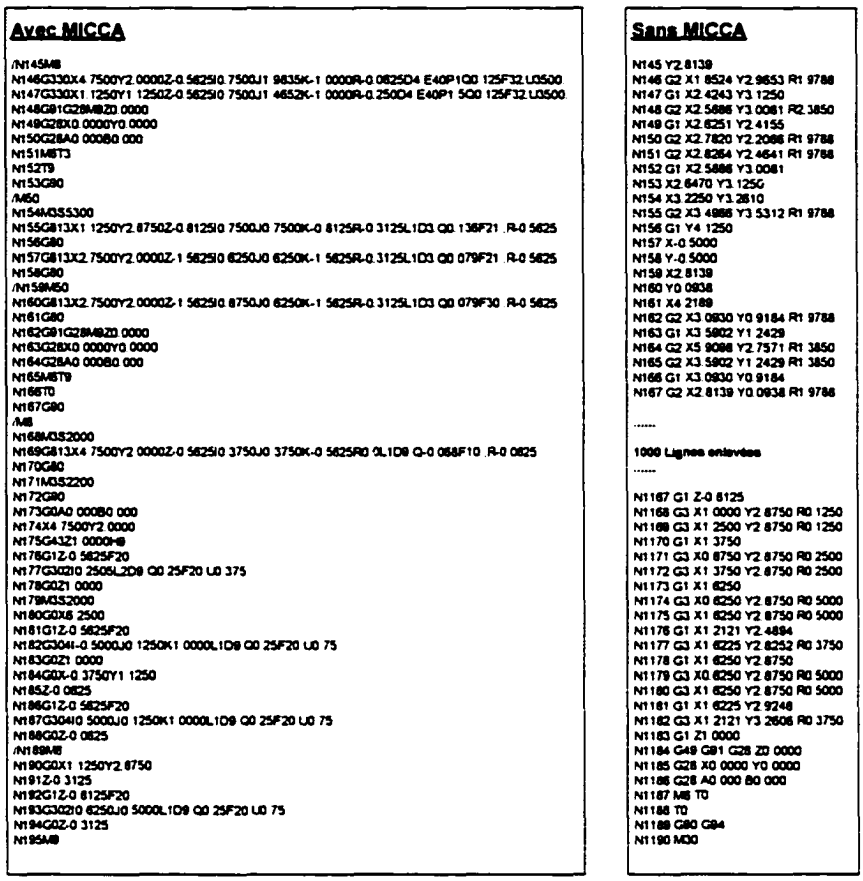
- 11b : Finition du diamètre externe de l'îlot #5. (CCA Circulaire extérieur pleine profondeur, horaire, G304)
- 12b : Finition de la pochette #4. (CCA Circulaire intérieur pleine profondeur, antihoraire, G302)

L'ordonnancement des séquences ainsi que la stratégie d'usinage utilisée permet de minimiser l'utilisation des séquences natives de Pro/MANUFACTURING. De plus, l'ensemble des types de CCA implantés dans Pro/CYCLES est utilisé. Un programme d'usinage de 202 lignes résulte du post-traitement des séquences d'usinage décrites ci-dessus. De ces lignes de code G, 157 sont obtenues à partir des séquences d'usinage décrites par les deux premiers éléments de la liste ci-dessus, réalisés à l'aide des séquences natives de Pro/MANUFACTURING.

Afin de bien illustrer les différences existant au niveau du contenu des programmes d'usinage obtenus, la figure 17 présente dans le cadre de gauche le code G en format CCA qui est nécessaire à l'usinage entier de la pièce mis à part les séquences 1b et 2b. Le cadre de gauche présente le code G généré à l'aide des séquences natives de Pro/MFG qui est nécessaire à l'usinage de la pièce équivalente. Il est à noter que 1000 lignes de code ont été enlevées du cadre de gauche.

4.1.2.3 Usinage

L'usinage des deux pièces d'essai a été réalisé sur la fraiseuse Hitachi-Seiki VS-50. La pièce brute, dont toutes les faces étaient déjà surfacées mis à part la face supérieure, est montée en étau sur la table de la MOCN, tel qu'illustré à la figure 18. L'usinage a été effectué avec la vitesse d'avance rapide de la machine ajustée à 100% de sa vitesse maximale (environ 38 mètres/minute).



Avec MICCA

```
/N145M8
N146G330X4 7500Y2.0000Z-0.5625I0 7500J1 9835K-1 0000R-0.0825D4 E40P1D0 125F32.L0500
N147G330X1 1250Y1 1250Z-0.5625I0 7500J1 4652K-1 0000R-0.250D4 E40P1 5D0 125F32.L0500
N148G81G28M8Z0.0000
N149G28X0.0000Y0.0000
N150G28A0 000B0.000
N151M6T3
N152T9
N153G80
/M50
N154M3S5300
N155G813X1 1250Y2.8750Z-0.8125I0 7500J0 7500K-0.8125R-0.3125L1D3 Q0.136F21 .R-0.5625
N156G80
N157G813X2.7500Y2.0000Z-1.5625I0 6250J0 6250K-1.5625R-0.3125L1D3 Q0.079F21 .R-0.5625
N158G80
/N159M50
N160G813X2.7500Y2.0000Z-1.5625I0.8750J0 6250K-1.5625R-0.3125L1D3 Q0.079F30 .R-0.5625
N161G80
N162G81G28M8Z0.0000
N163G28X0.0000Y0.0000
N164G28A0 000B0.000
N165M6T9
N166T0
N167G80
/M8
N168M3S2000
N169G813X4 7500Y2.0000Z-0.5625I0 3750J0 3750K-0.5625R0.0L1D9 Q-0.058F10 .R-0.0825
N170G80
N171M3S2200
N172G80
N173G0A0 000B0.000
N174X4 7500Y2.0000
N175G4321.0000H9
N176G1Z-0.5625F20
N177G3I0Z0.2505L2D9 Q0 25F20 U0.375
N178G0Z1.0000
N179M3S2000
N180G0X6.2500
N181G1Z-0.5625F20
N182G3I-0.5000J0 1250K1.0000L1D9 Q0 25F20 U0.75
N183G0Z1.0000
N184G0X-0.3750Y1.1250
N185Z-0.0825
N186G1Z-0.5625F20
N187G3I-0.5000J0 1250K1.0000L1D9 Q0 25F20 U0.75
N188G0Z-0.0825
/N189M8
N190G0X1.1250Y2.8750
N191Z-0.3125
N192G1Z-0.8125F20
N193G3I0Z0.6250J0.5000L1D9 Q0 25F20 U0.75
N194G0Z-0.3125
N195M9
```

Sans MICCA

```
N145 Y2.8139
N146 G2 X1.8524 Y2.9853 R1.9788
N147 G1 X2.4243 Y3.1250
N148 G2 X2.5686 Y3.0081 R2.3850
N149 G1 X2.6251 Y2.4155
N150 G2 X2.7820 Y2.2086 R1.9788
N151 G2 X2.8264 Y2.4641 R1.9788
N152 G1 X2.5886 Y3.0081
N153 X2.6470 Y3.1250
N154 X3.2250 Y3.2810
N155 G2 X3.4986 Y3.5312 R1.9788
N156 G1 Y4.1250
N157 X-0.5000
N158 Y-0.5000
N159 X2.8139
N160 Y0.0938
N161 X4.2188
N162 G2 X3.0830 Y0.9184 R1.9788
N163 G1 X3.5802 Y1.2429
N164 G2 X5.9088 Y2.7571 R1.3850
N165 G2 X3.5802 Y1.2429 R1.3850
N166 G1 X3.0830 Y0.9184
N167 G2 X2.8139 Y0.0938 R1.9788

.....

1000 Lignes enlevées
.....

N1167 G1 Z-0.8125
N1168 G3 X1.0000 Y2.8750 R0.1250
N1169 G3 X1.2500 Y2.8750 R0.1250
N1170 G1 X1.3750
N1171 G3 X0.8750 Y2.8750 R0.2500
N1172 G3 X1.3750 Y2.8750 R0.2500
N1173 G1 X1.6250
N1174 G3 X0.6250 Y2.8750 R0.5000
N1175 G3 X1.6250 Y2.8750 R0.5000
N1176 G1 X1.2121 Y2.4894
N1177 G3 X1.6225 Y2.8252 R0.3750
N1178 G1 X1.6250 Y2.8750
N1179 G3 X0.6250 Y2.8750 R0.5000
N1180 G3 X1.6250 Y2.8750 R0.5000
N1181 G1 X1.6225 Y2.9248
N1182 G3 X1.2121 Y3.2606 R0.3750
N1183 G1 Z1.0000
N1184 G49 G81 G28 Z0.0000
N1185 G28 X0.0000 Y0.0000
N1186 G28 A0.000 B0.000
N1187 M8 T0
N1188 T0
N1189 G80 G94
N1190 M30
```

Figure 17 Comparaison des programmes d'usinage

L'utilisation de la vitesse maximale de la MOCN lors des déplacements rapide permet d'augmenter le ratio du temps pendant lequel la machine usine la pièce par rapport au temps utilisé pour le positionnement de l'outil préalablement à l'usinage. Les temps ainsi obtenus sont plus représentatifs des séquences d'usinage et moins liés aux courses maximales des axes de la machine. Les vitesses de broche et d'avance d'usinage sont toutes deux fixées à 100% des valeurs programmées, contenues dans les programmes de code G.

Les programmes d'usinage sont transférés en entier dans la mémoire du contrôleur avant de débuter l'usinage. Ceci permet d'éliminer les effets de ralentissements qui auraient pu être causés par un taux de transfert trop lent entre le contrôleur de la MOCN et l'ordi-

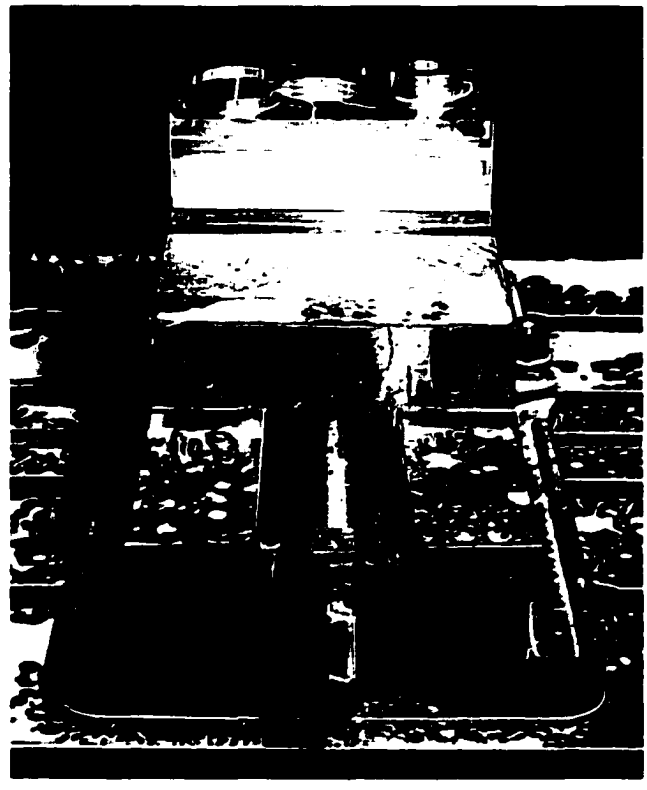


Figure 18 Montage d'usinage des pièces d'essai

nateur duquel est envoyé le programme d'usinage. Les résultats obtenus au niveau des temps d'usinage sont ainsi exempts des effets du lien utilisé pour effectuer le transfert du programme d'usinage entre la MOCN et l'ordinateur hôte.

Les temps de chacune des séquences pour l'usinage des deux pièces ont été relevés durant l'expérimentation pour fins de comparaison de l'efficacité des séquences natives Pro/MANUFACTURING versus les séquences réalisées à l'aide du MICCA Pro/CYCLES. Une des deux pièces expérimentales de géométrie identique est présentée à la figure 19.

Figure 19 Pièce d'essai complétée.

4.2 Résultats de l'expérimentation

Tel que mentionné précédemment, le seul critère quantitatif qui peut être utilisé afin d'évaluer l'implantation du MICCA repose sur les temps relevés durant l'usinage des pièces. Le tableau VI présente les temps d'usinage de chacune des séquences utilisées afin de réaliser l'usinage de la pièce utilisant entièrement les fonctionnalités d'usinage natives de Pro/MANUFACTURING. Le temps d'usinage total pour ce programme est de 15 :51 minutes, excluant les changements d'outils.

Les temps recueillis lors de l'usinage de la pièce utilisant les fonctionnalités du MICCA Pro/CYCLES sont présentés dans le tableau VII. Le temps d'usinage total de la pièce est de 18 :10, excluant également les temps de changement d'outils. Il est à noter que les séquences d'usinage 1b (Surfaçage) et 2b (Dégrossissage Volume) sont des séquences natives de Pro/MANUFACTURING. En effet, le CCA de surfaçage de la MOCN Hitachi-Seiki n'a pas été intégré au MICCA développé, et il n'existe pas de CCA équivalent à la séquence 2b de dégrossissage de région.

Tableau VI

Temps d'usinage sans utilisation du MICCA

Séquence	**Temps**
1a : Surfaçage	2 :27 min
2a : Dégrossissage Volume	7 :53 min
3a : Dégrossissage pochette #4	0 :34 min
4a : Dégrossissage pochette conique #5	1 :34 min
5a : Finition pochette conique #5	0 :38 min
6a : Dégrossissage pochette #2	0 :54 min
7a : Finition pochette #2	0 :11 min
8a : Finition diamètre externe îlot #1	0 :30 min
9a : Finition diamètre externe îlot #5	0 :29 min
10a : Finition pochette #4	0 :41 min
Total	15 :51 min

À partir de ces données quantitatives de temps et des autres données qualitatives qui seront présentées à la section suivante, l'évaluation de la performance offerte par le MICCA Pro/CYCLES est effectuée.

Tableau VII

Temps d'usinage avec utilisation du MICCA

Séquence	**Temps**
1b : Surfaçage	2 :27 min
2b : Dégrossissage Volume	2 :43 min
3b : Dégrossissage diamètre externe îlot #1 (G330)	4 :12 min
4b : Dégrossissage diamètre externe îlot #5 (G330)	2 :55 min
5b : Dégrossissage pochette #4 (G813)	0 :25 min
6b : Dégrossissage pochette conique #3 (G813)	0 :41 min
7b : Finition pochette conique #3 (G813)	1 :01 min
8b : Dégrossissage pochette #1 (G813)	0 :41 min
9b : Finition pochette #1 (G302)	0 :14 min
10b : Finition diamètre externe îlot #1 (G304)	1 :12 min
11b : Finition diamètre externe îlot #5 (G304)	1 :08 min
12b : Finition pochette #4 (G302)	0 :31 min
Total	18 :10 min

CHAPITRE 5

ANALYSE DES RÉSULTATS

Tel que déjà mentionné, un des seuls critères quantitatifs qui permet d'effectuer une comparaison entre l'utilisation du MICCA Pro/CYCLES et le module Pro/MANUFACTURING est celui des temps d'usinage. L'ensemble des autres critères d'évaluation sont à caractère qualitatif et il sera nécessaire de les quantifier de manière objective afin de réaliser une comparaison des avantages et des inconvénients associés au développement du module d'intégration des CCA.

Afin de bien illustrer les gains et les désavantages associés à l'utilisation d'un MICCA, les prochaines sections vont traiter de façon indépendante les différents critères d'évaluation. Cette approche permet d'illustrer plus clairement sur chacun des aspects du processus de génération de trajectoire d'outils et d'usinage les effets résultants de l'utilisation du module Pro/CYCLES. Par la suite, une comparaison et une évaluation globale sera réalisée afin d'illustrer l'utilisation d'une telle intégration dans un milieu industriel.

5.1 Évaluation individuelle des critères

Pour toute entreprise, il est évident que les gains associés à l'utilisation d'un MICCA sont associés à une productivité accrue. Il est toutefois complexe d'évaluer les gains associés à l'utilisation d'une interface graphique différente dans un logiciel de FAO. Il en est de même pour l'évaluation des gains de temps et de productivité obtenus par un programme d'usinage qui présente une lisibilité accrue. De plus, ces gains vont varier d'une compagnie à l'autre selon les produits fabriqués, la philosophie de l'entreprise et le personnel qui est employé afin de réaliser les trajectoires d'outils et l'usinage.

Pour ces raisons, les évaluations des critères qui suivent sont présentées de manière à décrire les gains et avantages possibles sans tenter de les quantifier de manière absolue. En effet, tenter de définir les gains de manière précise ne pourrait que restreindre l'application des résultats de la présente recherche car l'évaluation quantitative ne peut être réalisée qu'en visant un type d'entreprise et de produits manufacturés précis.

5.1.1 Temps d'usinage

Dans le cas des pièces d'essai réalisées et dont les temps d'usinage sont présentés à la section 4.2, on remarque que la pièce usinée à l'aide des CCA a exigé un temps total d'usinage de 18 minutes 15 secondes comparativement à un temps de 15 minutes 51 secondes pour l'utilisation du programme résultant de l'emploi des séquences d'usinage natives de Pro/MANUFACTURING. Les 2 :19 minutes supplémentaires qui ont été nécessaires à l'accomplissement de l'usinage basé sur les CCA sont principalement attributables au manque de contrôle de la direction des stratégies d'approche de ces CCA. Toutefois, ces mêmes stratégies d'approche sont plus efficaces que celles générées par les systèmes de FAO conventionnels.

En effet, les entrées en matières sont plus progressives et moins exigeantes pour l'outil de coupe. Une longévité accrue des outils et un meilleur fini de surface en résultent, ce qui compense pour le temps additionnel utilisé pour réaliser l'usinage.

Il est toutefois évident qu'une partie de l'influence des CCA sur la productivité est liée au temps d'usinage des pièces. L'effet de l'emploi des CCA sur la durée d'usinage dépend principalement de deux facteurs, soit la proportion de caractéristiques prismatiques de la pièce versus la proportion de caractéristiques non-usinables par CCA et le positionnement des caractéristiques prismatiques sur la pièce.

La proportion de caractéristiques prismatiques est liée au temps d'usinage qui sera effectué sur la MOCN par les CCA. Toutefois, l'efficacité relative de l'usinage effectué par les CCA dépend des caractéristiques et des propriétés du CCA utilisé qui est propriétaire à la MOCN sur laquelle l'usinage est réalisé. De plus, la majorité des CCA n'offrent pas de routines d'approches et débutent directement l'usinage de la caractéristique sans qu'il soit possible de définir le point d'approche de manière avancée.

Ainsi, pour la pièce d'essai qui à été usinée à l'aide des programmes réalisés avec Pro/CYCLES, le dégrossissage du diamètre externe de l'îlot #1 (G302) ne peut être programmé qu'avec une approche par Y+. Afin d'assurer que l'outil effectue l'approche hors du brut, il faut augmenter la distance d'approche du cycle de manière à ce que l'outil effectue sa descente en Z hors du brut. Le trajet d'approche est ensuite parcouru à la vitesse d'avance programmée pour l'usinage, ce qui cause des pertes de temps.

En fait, la comparaison des gains ou pertes de temps liés à l'utilisation des CCA dépend fortement des CCA utilisés ainsi que de leurs algorithmes de déplacement. De plus, la géométrie de la pièce a une forte influence sur les temps d'usinage, car certaines caractéristiques peuvent entraver l'approche de l'outil pour l'usinage d'un élément particulier. Ainsi, il est difficile de quantifier de manière absolue les gains de temps qui peuvent résulter de l'utilisation des CCA d'une MOCN particulière.

Il est évident qu'une augmentation des temps d'usinage est inacceptable dans le cas où une production en série de pièces est visée. Toutefois, une grande partie des avantages offerts par un MICCA pour la production de pièces unitaires ou prototypes est inexistante dans le cas de la production en série. À titre d'exemple, une augmentation de la lisibilité du programme n'apporte aucun gain pour une pièce qui sera usinée à quelques milliers d'exemplaires à l'aide du même programme d'usinage. Ainsi, l'utilisation d'un MICCA est axé vers les productions de faibles quantités de pièces, cas dans lequel une hausse du temps d'usinage qui peut ou non résulter de l'utilisation des CCA est compensée par les autres avantages de l'utilisation de ces cycles.

5.1.2 Facilité d'utilisation du MICCA

La facilité d'utilisation du module MICCA ne peut être évaluée qu'en comparaison à l'utilisation du logiciel dans lequel le MICCA à été développé, Pro/MANUFACTURING, dans le cas présent. Cette facilité d'utilisation est fortement basée sur l'interface graphique qui a été créée afin d'exploiter les capacités offertes par l'intégration proposée, elle est même limitée par les possibilités de programmation offertes par l'environnement de développement de la plate-forme de FAO utilisée pour réaliser l'intégration.

Toutefois, le développement d'un module d'intégration des CCA doit être réalisé par la compagnie qui désire intégrer les cycles de ses MOCN à son logiciel de FAO. Ainsi, le développement de l'interface graphique et de la méthodologie d'intégration peut être fortement orienté vers les produits fabriqués. Ceci ouvre la porte à la création d'un module personnalisé présentant une haute performance d'utilisation.

L'intégration actuelle est basée sur le module Pro/MANUFACTURING de Pro/ENGINEER 2000i^2. La définition des séquences d'usinage native dans ce logiciel utilise une méthodologie de description des informations d'usinage qui est peu efficace. En effet, plusieurs boîtes de dialogue successives doivent être utilisées de manière à définir l'ensemble de l'information nécessaire à la création d'une seule séquence d'usinage, tel qu'il en fut question à la section 3.1. Ceci résulte en une difficulté accrue pour l'utilisateur du système à faire la synthèse des informations concernant une séquence d'usinage en particulier.

Afin d'améliorer l'interface et d'augmenter la clarté des informations concernant les séquences d'usinage créées, le module Pro/CYCLES utilise une interface unique, sous la forme d'une seule boîte de dialogue qui renferme toute l'information décrivant un CCA. Ainsi, les paramètres d'usinage, tels que la vitesse de broche et d'avance ainsi que le nombre passe de finition, sont présentés dans la même interface que la géométrie à usi-

ner. Ceci permet une utilisation plus efficace du module en augmentant la lisibilité pour l'utilisateur et en diminuant les actions nécessaires à l'entrée des données.

Il est ainsi possible d'éliminer certaines lacunes que la plate-forme FAO présente au niveau de l'interface et d'adapter cette dernière aux besoins spécifiques de l'entreprise, ce qui résulte en une productivité accrue. De plus, si le MICCA est développé à l'interne, la formation des utilisateurs du MICCA peut être dispensée de manière efficace et le support offert aux utilisateurs de ce module est disponible à l'intérieur de l'entreprise.

On peut donc affirmer que, du point de vue de la programmation, le développement du MICCA résulte en un logiciel FAO offrant une productivité accrue car l'interface et le mode d'utilisation du module est créé sur mesure pour l'entreprise pour lequel il est créé.

De plus, l'intégration réalisée dans le cadre de ce projet traite des CCA d'usinage. Toutefois, l'usinage demeure possible en utilisant les fonctionnalités du logiciel de FAO en question. Il importe de mentionner qu'il existe également des CCA de mesure de pièce et de positionnement du zéro pièce sur le brut. De tels CCA ne possèdent pas d'équivalent dans les systèmes FAO commerciaux, et le seul moyen de réaliser ces opérations consiste à coder manuellement au niveau du CL-File ou du programme de code G les instructions de la MOCN afin de réaliser la mesure des pièces. Dans ce cas, l'utilisation d'un MICCA facilite grandement la réalisation de telles opérations en plus de diminuer les risques d'erreurs associés à l'édition manuelle des programmes.

5.1.3 Lisibilité du programme d'usinage

Pour toute entreprise qui effectue de la fabrication unitaire de pièces ou de prototypes, la lisibilité du programme d'usinage revêt une importance capitale afin que l'opérateur de la MOCN puisse prévoir tout problème qui pourrait résulter de la fabrication de la pièce. En effet, la connaissance des capacités des machines-outils et des différents outils

de coupe est détenue par l'opérateur de la MOCN, mais le programme d'usinage est réalisé par un programmeur qui connaît généralement moins bien les capacités et les limites des équipements.

Un programme offrant une lisibilité accrue peut être plus facilement analysé par l'opérateur de la MOCN, ce qui permet de détecter tout problème ou erreur avant qu'ils ne surviennent. L'emploi des CCA d'une MOCN permet d'augmenter fortement la lisibilité du programme en permettant de synthétiser par un seul code G l'usinage d'une zone complète de la pièce, comparativement à l'utilisation de quelques dizaines, voire plusieurs centaines de lignes de code G associées à l'emploi des séquences d'usinage natives des systèmes de FAO.

Dans le cas des pièces qui ont été usinées dans le cadre de l'expérimentation décrite au chapitre précédent, l'utilisation des séquences d'usinage natives de Pro/MANUFACTURING a résulté en un programme d'usinage en code G de 1190 lignes. Ce programme ne contient que des déplacements linéaires G01 et circulaires G02 et G03 afin d'accomplir l'usinage. Le programme obtenu à l'aide de l'utilisation du MICCA Pro/CYCLES ne contient quant à lui que 57 lignes de code G, en excluant les 145 lignes reliées aux séquences utilisées pour le surfaçage et le dégrossissage de la zone située au dessus de l'îlot #4.

En effet, chaque séquence d'usinage qui utilise un CCA de la MOCN utilise une seule ligne de code G afin de décrire l'ensemble de l'usinage réalisé par cette séquence. À titre d'exemple, le dégrossissage de la pochette conique #3 qui est réalisé de manière hélicoïdale est décrit par la ligne suivante sous forme de CCA :

G813 X2.7500 Y2.0000 Z-1.5625 I0.6250 J0.6250 K-1.5625 R-0.3125 L1 D3. Q0.079 F21. ,R-0.5625

L'usinage équivalent réalisé à l'aide des séquences natives de Pro/MANUFACTURING utilise 392 lignes composées de G01, G02 et G03. Il devient alors évident que l'opéra-

teur de la MOCN ne peut pas extraire les principaux paramètres d'usinage, tels que le pas de l'hélice et le diamètre qui est usiné, à partir d'un nombre si élevé de lignes de codes qui servent à décrire les déplacements d'outils. Toutefois, la description de l'usinage sous forme de CCA offre une lisibilité beaucoup plus élevée car le cycle utilisé décrit la géométrie à usiner ainsi que les paramètres d'usinage. Les positions ponctuelles par lesquelles l'outil doit passer afin de réaliser l'usinage sont déterminées par le contrôleur de la MOCN durant l'usinage de manière transparente pour l'utilisateur.

Il est alors évident que la lisibilité du programme d'usinage est significativement améliorée lors de l'utilisation des CCA afin de réaliser l'usinage. Les avantages qui découlent de cette lisibilité accrue sont présents à plusieurs niveaux et sont principalement appréciables lors de la fabrication de pièces unitaires. En effet, si l'utilisateur peut prévoir et interpréter les déplacements qui seront effectués par la MOCN à partir de la lecture du programme d'usinage pendant son exécution, ce dernier est plus en confiance face à la réalisation de l'usinage. Les temps d'hésitation et de vérification s'en trouvent réduits et l'utilisation de vitesses d'avances plus élevées sera adoptée. L'opérateur n'aura pas en effet à réduire la vitesse d'usinage qui serait nécessaire afin de lui permettre d'arrêter l'usinage si la machine exécute un mouvement que l'opérateur n'avait pas prévu.

5.1.4 Possibilités de modifications du programme d'usinage

Lors de la définition d'une séquence d'usinage à l'aide des fonctionnalités natives d'un système de FAO, l'utilisateur précise les valeurs des paramètres d'usinage et la géométrie à usiner. À partir de ces informations riches, le système de FAO génère le programme d'usinage qui contient des informations appauvries limitées aux coordonnées des points par lesquels l'outil doit passer afin de réaliser l'usinage. Ainsi, un paramètre tel que la profondeur de passe associée à une séquence d'usinage est dilué par le système de FAO dans l'ensemble des lignes du programme d'usinage sous forme des coordonnées X, Y

et Z représentant les trajectoires d'outils. Une modification subséquente de ce paramètre exige d'effectuer à nouveau la création des trajectoires d'outil dans le système FAO, et de traiter le CL-File résultant à l'aide du post-processeur afin d'obtenir la version corrigée du programme.

Les modifications, même mineures, qui doivent être apportées à ce type de programme exigent donc un temps élevé. Les corrections qui doivent être apportées au programme doivent en effet être acheminées jusqu'au département des méthodes où la personne qui a réalisé initialement le programme doit alors modifier les données afin de produire le nouveau programme qui doit être à nouveau acheminé à la MOCN qui est en attente de la nouvelle version du programme. Des coûts élevés sont donc associés à de telles modifications.

Dans le cas de l'emploi de CCA dans le programme d'usinage, l'opérateur peut apporter des modifications aux paramètres d'usinage du CCA, tel que la profondeur ou encore la largeur de passe, si il juge que les paramètres qui ont étés sélectionnés résulteront en une vibration excessive pendant l'usinage ou seront trop peu efficaces, par exemple. Ceci est rendu possible par le type de définition qui est utilisée par les CCA, qui contiennent directement les dimensions des caractéristiques à usiner ainsi que les valeurs des paramètres d'usinage. Des modifications rapides et efficaces des trajectoires d'outils peuvent être réalisées en quelques minutes directement au contrôleur de la machine, favorisant la réduction des temps associés aux altérations des données d'usinage.

Les avantages résultants de cette possibilité de modification des programmes sont plus importants pour la fabrication de pièces prototypes, unitaires ou de pièces d'essai pour des productions en série. De telles pièces sont en effet susceptibles de nécessiter des modifications au programme d'usinage. Dans le cas de la fabrication de série, lorsque les essais sont complétés sur les pièces prototype, les avantages des possibilités de modification du programme sont réduits.

5.1.5 Taille du programme d'usinage en code G

Plusieurs MOCN présentent une faiblesse au niveau du lien utilisé pour transférer le programme d'usinage d'un ordinateur jusqu'au contrôleur tout comme au niveau de la mémoire disponible à même le contrôleur pour entreposer ce programme. En effet, tel que mentionné à la section 1.1.6.1, si le lien utilisé pour transmettre le programme est trop lent et que la mémoire de la MOCN présente une taille trop faible pour permettre d'emmagasiner le programme en entier, ce dernier est transféré par blocs durant l'usinage, au fur et à mesure que la MOCN nécessite l'information d'usinage.

Il existe plusieurs solutions à ce problème. L'une d'entre elles consiste à ajouter de la mémoire supplémentaire à même le contrôleur de la MOCN afin de pouvoir y loger le programme d'usinage en entier. Cette solution est limitée par l'aspect propriétaire des contrôleurs de MOCN. En effet, les rares contrôleurs qui offrent des possibilités d'expansion de la mémoire nécessitent des investissements excessifs, se chiffrant à plusieurs milliers de dollars pour quelques méga-octets supplémentaires d'espace mémoire. De plus, une limitation d'espace existe toujours suite à l'augmentation de la mémoire disponible.

Une autre solution consiste à changer la technologie du lien qui assure la transmission des données entre l'ordinateur et le contrôleur de la MOCN, de manière à assurer un débit d'information suffisant pour réaliser l'usinage sans qu'il n'y ait de saturation du lien. Cette solution est encore plus limitée que l'ajout de la mémoire, tant au niveau de la disponibilité de telles options de la part des fabricants de contrôleurs que du coût associé à ces options.

La dernière solution consiste à diminuer la taille des données à transférer afin d'assurer que le programme puisse être emmagasiné en entier dans la mémoire du contrôleur ou que la transmission des données en temps réel pendant l'usinage puisse être effectuée sans saturation du lien à basse vitesse qui équipe la majorité des MOCN utilisées en industrie.

L'utilisation des CCA dans les programmes d'usinage est extrêmement avantageuse à ce niveau. En effet, le programme d'usinage utilisant les fonctionnalités natives de Pro/MANUFACTURING présente un taille de 33Ko alors que celui utilisant Pro/CYCLES n'utilise que 4Ko. La diminution de la taille des programmes d'usinage est ainsi un avantage indéniable pour assurer un usinage sans arrêt du flux de données.

Il est toutefois évident que la réduction de taille du fichier dépend de la quantité de CCA qu'il est possible d'utiliser pour réaliser l'usinage d'une pièce en particulier. Pour les pièces qui présentent plusieurs caractéristiques prismatiques, une réduction importante de la taille du fichier peut être atteinte, alors que pour les pièces comportant des caractéristiques qui sont composées principalement de surfaces complexes, les gains sont négligeables ou inexistants.

Pour des entreprises qui usinent des pièces principalement prismatiques sur des MOCN possédant une mémoire limitée et un lien relativement lent, des gains importants en terme de productivité et de qualité de pièces peuvent être atteints par l'utilisation d'un MICCA. Cette solution est également avantageuse économiquement comparativement à l'ajout de mémoire sur les contrôleurs de MOCN ou encore à la modification du type de lien de transfert de données. En effet, la création d'un MICCA permet le gain associé d'être réparti sur l'ensemble du parc de MOCN d'une compagnie, alors que les modifications associées aux contrôleurs des MOCN ne sont bénéfiques que pour les MOCN qui ont subi la modification.

5.1.6 Flexibilité du transfert de données FAO – MOCN

L'une des limitations principales du cheminement de l'information entre le module FAO et le module MOCN réside dans le fait que ce transfert de données est unidirectionnel. Ceci limite fortement l'utilisation et l'exploitation des connaissances techniques que possèdent

les opérateurs de MOCN sur le procédé d'usinage et les capacités des équipements. En effet, il est difficile pour le département de programmation d'inclure de façon efficace ces connaissances présentes sur le plancher de l'usine dans la création des programmes d'usinage. Ceci résulte en une perte de productivité provenant d'une mauvaise gestion des ressources techniques disponibles dans toute industrie qui effectue de l'usinage.

Le seul moyen disponible en industrie pour faire cheminer les informations techniques concernant les paramètres et les stratégies d'usinage de l'usine jusqu'au département d'ingénierie consiste à transmettre les recommandations des opérateurs de MOCN de manière verbale ou encore sur support physique jusqu'au département d'ingénierie. Cette méthodologie nécessite un temps excessif, tel que discuté à la section 5.1.4. De plus, le climat qui règne entre le département de programmation et les opérateurs de MOCN peut faire en sorte que les recommandations exprimées par les opérateurs ne soient pas prises en compte.

L'utilisation d'un MICCA doté d'un module de post-processeur inversé permet de fournir une solution efficace au problème d'unidirectionnalité du transfert d'information FAO – MOCN. En effet, ce module permet d'incorporer les modifications qui ont pu être apportées au programme d'usinage dans le modèle FAO.

Ceci permet une meilleure liaison entre l'ingénierie et l'usine tout en assurant une exploitation maximale des connaissances des opérateurs de MOCN au niveau de l'usinage et des capacités des équipements. Les gains à court terme se traduisent par un usinage plus efficace ainsi que des temps de production diminués. À long terme, l'expérience des opérateurs concernant les équipements et les outils de coupe peut être transférée à l'ingénierie afin d'assurer une amélioration continue de la qualité des programmes d'usinage générés.

5.1.7 Développement et entretien du système FAO & des post-processeurs

Dans un processus de fabrication assistée par ordinateur conventionnel, sans utilisation de MICCA, des coûts d'entretien des systèmes de FAO et des post-processeurs existent. En effet, lors de la mise à jour du système FAO à une nouvelle version, des coûts d'acquisition du logiciel ainsi que d'installation doivent être défrayés. De plus, une formation doit être dispensée aux employés qui utilisent ce logiciel afin de leur permettre d'utiliser efficacement les fonctionnalités offertes dans la nouvelle version.

Les post-processeurs doivent également être maintenus afin d'assurer la compatibilité avec les nouvelles versions des systèmes de FAO. L'achat de nouvelles MOCN exige la réalisation d'un post-processeur pour chacune de ces machines dans le système FAO utilisé.

Dans le cas de l'utilisation d'un MICCA, des coûts additionnels doivent être défrayés. Parmi ces coûts supplémentaires, citons l'achat d'une license de développement pour le système de CAO/FAO dans lequel le MICCA sera développé. De plus, les ressources humaines et techniques doivent être embauchées ou formées afin de pouvoir procéder au développement du MICCA envisagé. Le post-processeur de chaque MOCN doit être adapté de manière à pouvoir traiter les informations des CCA qui sont intégrés au CL-File par le MICCA et générer les codes G correspondants aux MICCA pour chaque MOCN.

De plus, la mise à jour du système de FAO à une nouvelle version peut exiger des modifications au code source de MICCA de manière à l'adapter aux API disponibles dans l'environnement de développement de la nouvelle version du système FAO.

L'ensemble des coûts liés au développement et à l'entretien d'un module MICCA sont difficilement définissables. Les facteurs qui influent sur ces coûts incluent, mais ne sont pas limités à :

- Coût de la license de développement du système de FAO
- Possibilités offertes par les API

- Disponibilité des ressources humaines pour le développement du MICCA
- Disponibilité des ressources techniques pour le développement du MICCA
- Qualité de la documentation disponible sur les API du système de FAO.
- Qualité de la documentation disponible sur les CCA des MOCN
- Niveau d'intégration souhaité des CCA
- Nombre de CCA à intégrer
- Nombre de MOCN à intégrer
- Différences entre les contrôleurs de MOCN dont les CCA sont à intégrer
- Nombre de post-processeurs à adapter
- Importance des changements à apporter au MICCA entre les versions du logiciel de FAO
- Possibilités de développement offertes par le post-processeur

Il est toutefois évident que la réalisation d'un MICCA n'est économiquement viable que pour une compagnie qui produit des pièces qui présentent une proportion relativement élevée de pièces prismatiques, afin que les économies associées aux gains de productivité du MICCA réalisé puissent rentabiliser le développement et l'entretien de ce MICCA.

5.2 Analyse globale des résultats

À partir des analyses individuelles des différents critères d'évaluation du MICCA présentées aux sections précédentes, il convient de réaliser une analyse globale des résultats obtenus lors de l'expérimentation de manière à pouvoir évaluer l'ensemble de la performance du module Pro/CYCLES développé, et à une plus grande échelle, d'un MICCA.

Il ressort de l'expérimentation que la réalisation d'une telle intégration permet d'obtenir un meilleur accès à l'information d'usinage, et ce, à deux niveaux. Premièrement, l'interface utilisateur qui est développée pour le MICCA dans le module de FAO est adaptée aux

besoins des utilisateurs du système. Ceci permet entre autres de combler des lacunes au niveau de l'interface native de ce module, et d'ajouter des fonctionnalités qui n'existent pas dans le logiciel initial. De plus, des programmes d'usinage présentant une lisibilité élevée résultent de l'emploi d'un MICCA, permettant une vérification du programme et des paramètres d'usinage de la part de l'opérateur, qui peut également modifier les paramètres qu'il considère incorrects.

Cette augmentation de l'accès aux données d'usinage résulte en des gains de temps de programmation et d'usinage dont l'importance varie selon les CCA disponibles, le niveau d'intégration des CCA dans le MICCA, et la mentalité de l'entreprise face aux modifications des paramètres d'usinage par les employés d'usine.

De plus, les programmes d'usinage plus compacts qui résultent de l'utilisation des CCA permettent d'utiliser des machines possédant peu de mémoire et des liens de transfert de données lents pour effectuer de l'usinage relativement complexe à des vitesse supérieures à ce qui était atteignable avec des programmes conventionnels. On peut utiliser à pleine capacité les MOCN qui pouvaient être considérées comme désuètes jusqu'alors.

Toutefois, le développement d'une MICCA n'est justifiable économiquement que dans le cas où une entreprise fabrique des pièces pour lesquelles les CCA pourraient facilement être appliqués, et ce, en proportion suffisante. De plus, la production se doit d'être majoritairement unitaire ou de faible série afin de pouvoir profiter de l'ensemble des avantages offerts par un MICCA.

CONCLUSION

Les incompatibilités et les problématiques qui existent au niveau du transfert d'information entre les modules de FAO, de post-processeur et de MOCN d'un système intégré de fabrication ont été décrits. Plusieurs solutions en cours de développement qui tentent d'augmenter, de différentes façons, le niveau d'intégration entre ces diverses composantes ont été analysées afin de démontrer qu'une lacune existait au niveau de l'utilisation de l'ensemble des capacités des MOCN.

En effet, plusieurs fonctionnalités avancées offertes par les MOCN sont sous exploitées ou encore inutilisées. Cet état de fait est principalement dû à un manque de normalisation au niveau du langage utilisé pour contrôler les MOCN, le code G. Afin de pouvoir tirer avantage de ces fonctionnalités avancées, une méthodologie d'intégration à été développée. Celle-ci présente les détails nécessaires à la création d'un lien efficace entre les cycles avancés des MOCN et les systèmes FAO.

Un exemple d'application de la méthodologie d'intégration à été réalisé en intégrant des cycles cannés avancés d'usinage disponibles à même le contrôleur Fanuc - Seicos d'une MOCN au logiciel de FAO Pro/MANUFACTURING du système Pro/ENGINEER. L'intégration a permis de démontrer la faisabilité du projet selon la méthodologie d'intégration établie. Des pièces d'essai ont été usinées afin de permettre une comparaison des possibilités et des avantages de l'usinage à l'aide des cycles avancés comparativement à l'usinage référant aux fonctionnalités de base des systèmes de FAO.

Il a été démontré lors de l'analyse des résultats qu'une exploitation maximale des avantages offerts par le développement d'un module d'intégration des CCA à une plate-forme FAO ne peut être réalisée que pour des productions unitaires ou des séries de faibles envergures, et ce, pour des pièces présentant une géométrie majoritairement prismatique. Dans ce cas, les principaux avantages associés à l'utilisation de tels cycles d'usinage sont représentés par un meilleur accès à l'information d'usinage dans le système de FAO,

une meilleure lisibilité du programme d'usinage, et la possibilité de le modifier à même le contrôleur de la MOCN. De plus, un accroissement des performances des MOCN désuettes peut être obtenu par une réduction de la taille des programmes d'usinage. Pour des productions en série ou pour l'usinage de pièces composées principalement de surfaces et de géométries complexes, les avantages offerts par l'utilisation des CCA sont moins nombreux.

De plus, les efforts nécessaires à la réalisation des CCA à un logiciel de FAO ont été évalués considérables. Ils ne sont donc justifiables que dans le cas où plusieurs des avantages possibles liés à l'utilisation des CCA peuvent être tirés de l'intégration.

Ce type d'intégration n'est donc efficace que pour des cas où les pièces produites sont représentatives des capacités des CCA disponibles sur les différentes MOCN sur lesquelles l'usinage peut être réalisé.

RECOMMANDATIONS

Le travail présenté ici s'est limité, dans ses parties application et expérimentation, à l'intégration des CCA d'usinage fournis avec les contrôleurs de MOCN aux systèmes de FAO. Toutefois, il est possible de programmer paramétriquement des CCA d'usinage directement dans les contrôleurs de MOCN, de manière à créer des CCA qui sont orientés vers le type de produits manufacturés par une entreprise spécifique. L'étude de ce type d'intégration pourrait élargir les possibilités d'application de la méthodologie développée au cours du présent travail.

De plus, une des limitations de l'intégration proposée est qu'il est présentement impossible d'effectuer une simulation virtuelle native de l'usinage réalisé par ces CCA. L'établissement d'une méthodologie d'intégration des CCA à un module de simulation d'usinage permettrait de diminuer les risques d'erreurs liés à l'usinage tout en maximisant la productivité des MOCN qui ne serviraient plus alors de plateforme de simulation pour les programmes contenant des CCA.

Finalement, le sous-module de post-processeur inversé proposé dans l'intégration des CCA pourrait être étendu de manière à pouvoir être utilisé afin de traiter les modifications qui peuvent être apportées aux sections du programme d'usinage qui ne sont pas constituées de CCA. Ainsi, des modifications de vitesses de rotation d'outil et d'avance pourraient être intégrées au modèle FAO suite à leur modification à même le contrôleur de la MOCN, permettant ainsi de mieux fermer la boucle bi-directionnelle de transfert d'information FAO – MOCN permise par la création du MICCA.

ANNEXE 1

Boîtes de dialogue Pro/CYCLES

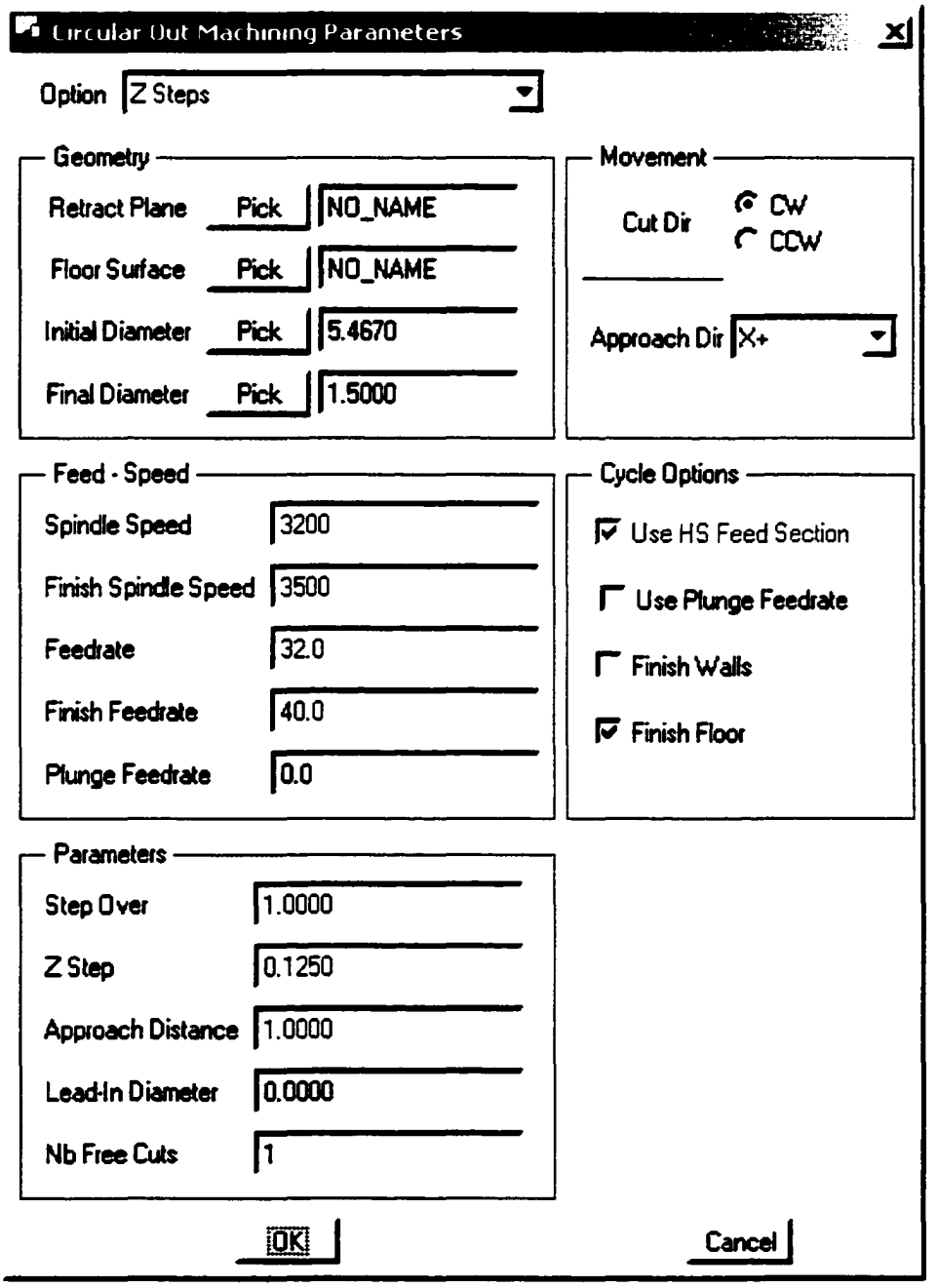


Figure 20 Boîte de dialogue - usinage circulaire externe.

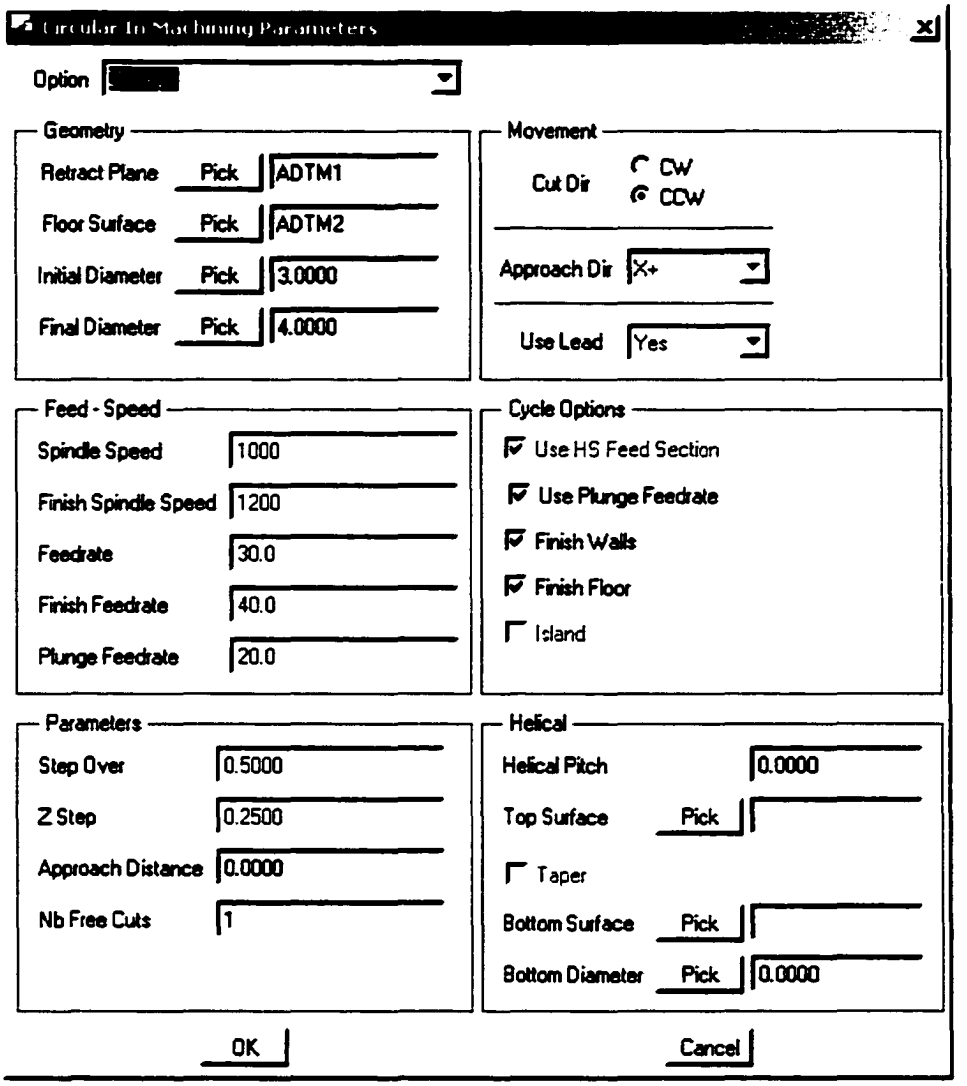


Figure 21 Boîte de dialogue - usinage circulaire interne.

ANNEXE 2

Description des cycles avancés intégrés à Pro/CYCLES

Pochette circulaire pleine profondeur

Ce cycle permet de réaliser l'usinage d'une pochette de géométrie intérieure circulaire sur la MOCN Hitachi-Seiki VS-50. L'usinage est réalisé en une seule passe en Z et les déplacements d'outils associés à ce cycles sont présentés à la figure 22.

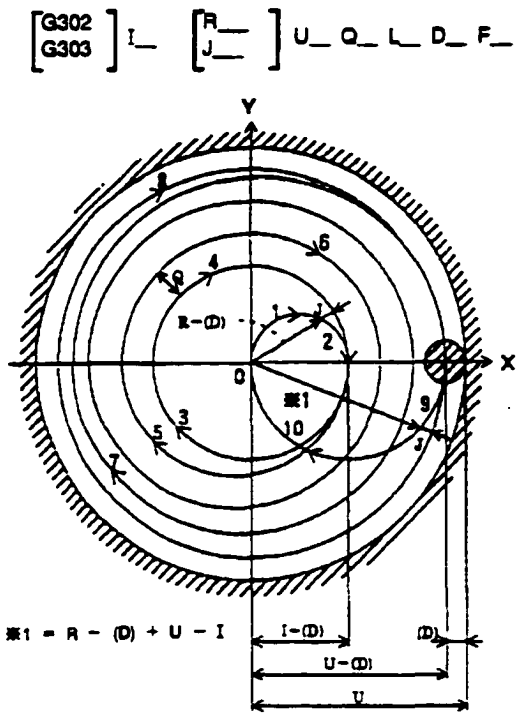


Figure 22 Cycle d'usinage de pochette circulaire G302 et G303. Tiré de [36].

Ce cycle est commandé soit par le code G302, résultant en un déplacement horaire de l'outil, ou par le code G303, résultant en un déplacement anti-horaire. Les éléments entre parenthèses carrées représentent des paramètres optionnels du cycle.

La description détaillée des paramètres d'usinage de ce cycle en code G est présentée ci-dessous.

Syntaxe du code G

I : Rayon de l'avant-tour utilisé pour effectuer la descente de l'outil.

I+ : Approche par la direction X+.

I- : Approche par la direction X-.

R : Distance d'approche à haute vitesse. Représente la distance entre le périmètre de l'outil et le rayon final de la pochette.

J : Distance d'approche à haute vitesse. Représente la distance entre le centre de la pochette et le centre de l'outil selon l'axe X.

U : Rayon final de la pochette circulaire.

Q : Pas de la spirale de déplacement.

L : Nombre de répétition de la passe de finition.

D : Numéro de la compensation en diamètre de l'outil.

F : Vitesse d'avance de l'outil.

Syntaxe du CL-File Pro/CYCLES

La syntaxe générée dans le CL-File par Pro/CYCLES et qui sera transformée en G302 ou G303 par le post-processeur est présentée ci-dessous. Les éléments entre parenthèses carrées sont optionnels.

Usinage Horaire (G302) :

CYCLE / CIRCUL, IN, **CLW**, BORE, $P1, DIAMET, $P2, STPOVR, $P3, FEDRAT, $P4, [HSDIST, $P5], [FINCUT, $P6]

Usinage Anti-horaire (G303) :

CYCLE / CIRCUL, IN, **CCLW**, BORE, $P1, DIAMET, $P2, STPOVR, $P3, FEDRAT, $P4, [HSDIST, $P5], [FINCUT, $P6]

La correspondance entre les paramètres en syntaxe CL-File et la syntaxe en code G est la suivante :

BORE : Correspond au double du paramètre I du G302 et G303.

DIAMET : Correspond au double du paramètre U du G302 et G303.

STPOVR : Correspond au paramètre U du G302 et G303.

FEEDRAT : Correspond au paramètre F du G302 et G303.

FINCUT : Correspond au paramètre L du G302 et G303.

HSDIST : Correspond au paramètre J du G302 et G303

Îlot circulaire pleine profondeur

Ce cycle permet de réaliser l'usinage extérieur d'un îlot de géométrie circulaire sur la MOCN Hitachi-Seiki VS-50. L'usinage est réalisé en une seule passe en Z et les déplacements d'outils associés à ce cycles sont présentés à la figure 23.

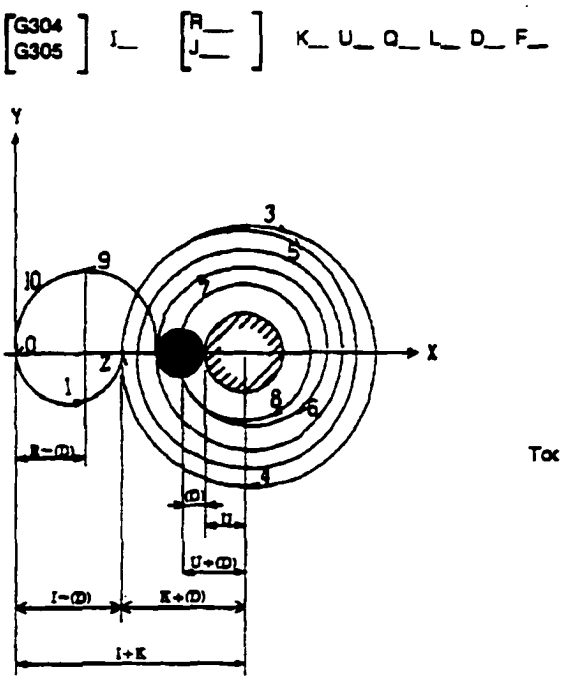


Figure 23 Cycle d'usinage d'îlot circulaire G304 et G305. Tiré de [36].

Ce cycle est commandé soit par le code G304, résultant en un déplacement horaire de l'outil, ou par le code G305, résultant en un déplacement anti-horaire. Les éléments entre parenthèses carrées représentent des paramètres optionnels du cycle.

La description détaillée des paramètres d'usinage de ce cycle en code G est détaillée ci-dessous.

Syntaxe du code G

I : Diamètre du cercle d'approche de la trajectoire d'outil.

I+ : Approche par X+.

I- : Approche par X-.

R : Distance d'approche à haute vitesse. Représente la distance entre le point de départ de l'outil et le périmètre de l'outil lors de son approche au diamètre initial de l'îlot.

J : Distance d'approche à haute vitesse. Représente la distance entre le point de départ de l'outil et le centre de l'outil selon l'axe X.

K : Rayon initial de l'îlot.

U : Rayon final de l'îlot.

Q : Pas de la spirale de déplacement.

L : Nombre de répétition de la passe de finition.

D : Numéro de la compensation en diamètre de l'outil.

F : Vitesse d'avance de l'outil.

Syntaxe du CL-File Pro/CYCLES

La syntaxe générée dans le CL-File par Pro/CYCLES et qui sera transformée en G304 ou G305 par le post-processeur est présentée ci-dessous. Les éléments entre parenthèses carrées sont optionnels.

Usinage Horaire (G304) :

CYCLE / CIRCUL, OUT, **CLW**, LEAD, $P1, DIAMET, $P2, BOSS, $P3, STPOVR, $P4, FEDRAT, $P5, [HSDIST, $P6], [FINCUT, $P7]

Usinage Anti-horaire (G305) :

CYCLE / CIRCUL, OUT, **CCLW**, LEAD, $P1, DIAMET, $P2, BOSS, $P3, STPOVR, $P4, FEDRAT, $P5, [HSDIST, $P6], [FINCUT, $P7]

La description détaillé des paramètres est la suivante.

LEAD : Correspond au paramètre R du G304 et G305.

DIAMET : Correspond au double du paramètre K du G304 et G305.

BOSS : Correspond au double du paramètre U du G304 et G305.

STPOVR : Correspond au paramètre Q du G304 et G305.

FEEDRAT : Correspond au paramètre F du G304 et G305.

HSDIST : Correspond au paramètre J du G304 et G305.

FINCUT : Correspond au paramètre L du G304 et G305.

Pochette circulaire avec niveaux en Z

Ce cycle permet de réaliser l'usinage interne d'une pochette de géométrie circulaire sur la MOCN Hitachi-Seiki VS-50. L'usinage est réalisé en plusieurs niveaux en Z et les déplacements d'outils associés à ce cycles sont présentés à la figure 24.

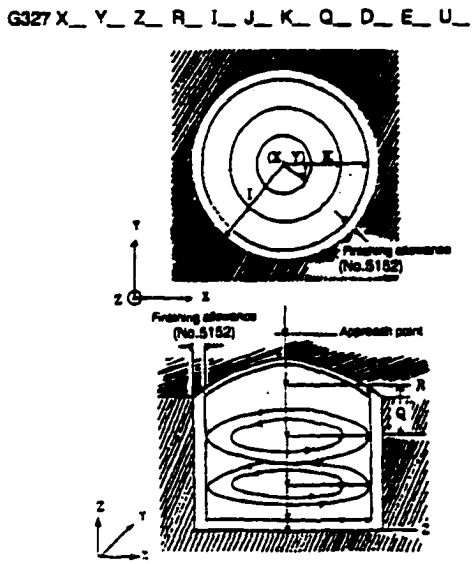


Figure 24 Cycle d'usinage de pochette circulaire G327. Tiré de [36].

La description détaillée des paramètres d'usinage de ce cycle en code G est détaillée ci-dessous.

Syntaxe du code G

X, Y : Position du centre de la pochette.

Z : Coordonnée du fond de la pochette finie.

R : Position d'approche de l'outil selon Z.

I : Rayon final de la pochette.

I+ : Usinage en direction horaire.

I- : Usinage en direction anti-horaire.

J : Rayon de l'avant-trou utilisé pour effectuer la descente de l'outil.

J+ : Usinage régulier.

J- : Un îlot de rayon J sera laissé au centre de la pochette.

K : Largeur de coupe de l'outil pendant l'usinage.

K+ : Finition effectuée sur les parois verticales.

K- : Pas de finition effectuée sur les parois verticales.

Q : Profondeur de passe de l'outil.

Q+ : Finition effectuée sur le fond de la pochette.

Q- : Pas de finition effectuée sur le fond de la pochette.

D : Numéro de la compensation en diamètre de l'outil.

E : Vitesse d'avance de l'outil pour la passe de finition.

U : Vitesse de la broche pour la passe de finition.

V : Vitesse d'avance de l'outil lors des déplacements selon Z.

F : Vitesse d'avance de l'outil.

Syntaxe du CL-File Pro/CYCLES

La syntaxe générée dans le CL-File par Pro/CYCLES et qui sera transformée en G327 par le post-processeur est la suivante :

CYCLE / POCKET, CIRCUL, IN, XAXIS, $P1, YAXIS, $P2, DEPTH, $P3, CLEAR, $P4, DIAMET, $P5, BORE, $P6, STPOVR, $P7, ZSTEP, $P8, FEDRAT, $P9, [ZFEED, $P10], [FINFED, $P11], [FINRPM, $P12]

La description détaillé des paramètres est présentée ci-dessous.

XAXIS : Correspond au paramètre X du G327.

YAXIS : Correspond au paramètre Y du G327.

DEPTH : Correspond au paramètre Z du G327.

CLEAR : Correspond au paramètre R du G327.

DIAMET : Correspond au double du paramètre I du G327.

BORE : Correspond au double du paramètre J du G327.

STPOVR : Correspond au paramètre K du G327.

ZSTEP : Correspond au paramètre Q du G327.

FEDRAT : Correspond au paramètre F du G327.

ZFEED : Correspond au paramètre V du G327.

FINFED : Correspond au paramètre E du G327.

FINRPM : Correspond au paramètre U du G327.

Îlot circulaire avec niveaux en Z

Ce cycle permet de réaliser l'usinage externe d'un îlot de géométrie circulaire sur la MOCN Hitachi-Seiki VS-50. L'usinage est réalisé en plusieurs niveaux en Z et les déplacements d'outils associés à ce cycles sont présentés à la figure 25.

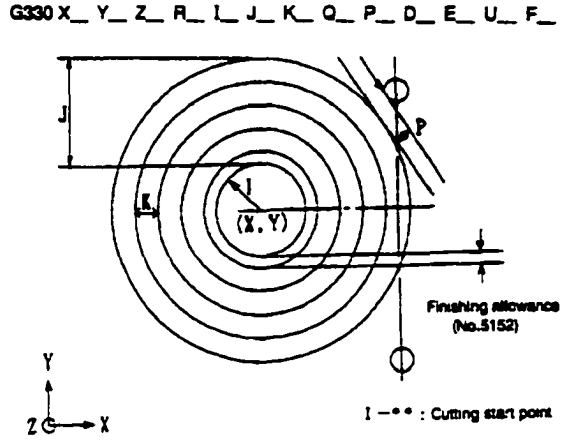


Figure 25 Cycle d'usinage d'îlot circulaire G330. Tiré de [36].

La description détaillée des paramètres d'usinage de ce cycle en code G est détaillée ci-dessous.

Syntaxe du code G

X, Y : Position du centre de l'îlot.

Z : Coordonnée du fond de l'îlot fini.

R : Position d'approche de l'outil selon Z.

I : Rayon final de l'îlot.

I+ : Usinage en direction horaire.

I- : Usinage en direction anti-horaire.

J : Largeur de matériel à usiner en périphérie de l'îlot.

K : Largeur de coupe de l'outil pendant l'usinage.

K+ : Finition effectuée sur les parois verticales.

K- : Pas de finition effectuée sur les parois verticales.

Q : Profondeur de passe de l'outil.

Q+ : Finition effectuée sur le fond de l'îlot.

Q- : Pas de finition effectuée sur le fond de l'îlot.

P : Distance d'approche de l'outil.

D : Numéro de la compensation en diamètre de l'outil.

E : Vitesse d'avance de l'outil pour la passe de finition.

U : Vitesse de la broche pour la passe de finition.

F : Vitesse d'avance de l'outil.

Syntaxe du CL-File Pro/CYCLES

La syntaxe générée dans le CL-File par Pro/CYCLES et qui sera transformée en G330 par le post-processeur est la suivante :

CYCLE / POCKET, CIRCUL, OUT, XAXIS, $P1, YAXIS, $P2, DEPTH, $P3, CLEAR, $P4, DIAMET, $P5, WIDTH, $P6, STPOVR, $P7, ZSTEP, $P8, APRDST, $P9, FEDRAT, $P10, [FINFED, $P11], [FINRPM, $P12]

La description détaillé des paramètres est la suivante.

XAXIS : Correspond au paramètre X du G330.

YAXIS : Correspond au paramètre Y du G330.

DEPTH : Correspond au paramètre Z du G330.

CLEAR : Correspond au paramètre R du G330.

DIAMET : Correspond au double du paramètre I du G330.

WIDTH : Correspond au double du paramètre J du G330.

STPOVR : Correspond au paramètre K du G330.

ZSTEP : Correspond au paramètre Q du G330.

APRDST : Correspond au paramètre P du G330.

FEDRAT : Correspond au paramètre F du G330.

FINFED : Correspond au paramètre E du G330.

FINRPM : Correspond au paramètre U du G330.

Pochette circulaire avec niveaux en Z et approche tangentielle

Ce cycle permet de réaliser l'usinage interne d'une pochette de géométrie circulaire sur la MOCN Hitachi-Seiki VS-50. L'usinage est réalisé en plusieurs niveaux en Z et des approches et sorties tangentielles sont utilisées. Les déplacements d'outils associés à ce cycles sont présentés à la figure 26.

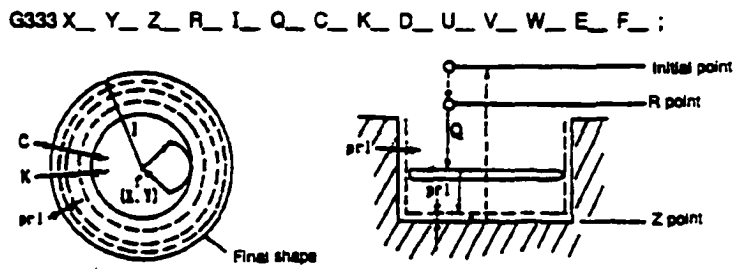


Figure 26 Cycle d'usinage de pochette circulaire G333. Tiré de [36].

La description détaillée des paramètres d'usinage de ce cycle en code G est détaillée ci-dessous.

Syntaxe du code G

X, Y : Position du centre de la pochette.

Z : Coordonnée du fond de la pochette finie.

R : Position d'approche de l'outil selon Z.

I : Rayon final de la pochette.

I+ : Usinage en direction horaire.

I- : Usinage en direction anti-horaire.

Q : Profondeur de passe de l'outil.

Q+ : Finition effectuée sur le fond de la pochette.

Q- : Pas de finition effectuée sur le fond de la pochette.

C : Largeur de matériel à usiner.

K : Largeur de coupe de l'outil pendant l'usinage.

K+ : Finition effectuée sur les parois verticales.

K- : Pas de finition effectuée sur les parois verticales.

D : Numéro de la compensation en diamètre de l'outil.

U : Vitesse de la broche pour la passe de finition.

V : Vitesse d'avance de l'outil lors des déplacements selon Z.

W : Vitesse d'avance de l'outil lors des approches et sorties tangentielles.

E : Vitesse d'avance de l'outil pour la passe de finition.

F : Vitesse d'avance de l'outil.

Syntaxe du CL-File Pro/CYCLES

La syntaxe générée dans le CL-File par Pro/CYCLES et qui sera transformée en G333 par le post-processeur est présentée ci-dessous.

CYCLE / POCKET, CIRCUL, IN, LEAD, LEAD, XAXIS, $P1, YAXIS, $P2, DEPTH, $P3, CLEAR, $P4, DIAMET, $P5, ZSTEP, $P6, WIDTH, $P7, STPOVR, $P8, FEDRAT, $P9, [ZFEED, $P10], [FINFED, $P11] ,[FEDTO, $P12], [FINRPM, $P13]

La description détaillé des paramètres est la suivante.

XAXIS : Correspond au paramètre X du G333.

YAXIS : Correspond au paramètre Y du G333.

DEPTH : Correspond au paramètre Z du G333.

CLEAR : Correspond au paramètre R du G333.

DIAMET : Correspond au double du paramètre I du G333.

ZSTEP : Correspond au paramètre Q du G333.

WIDTH : Correspond au paramètre C du G333.

BORE : Correspond au double du paramètre J du G333.

STPOVR : Correspond au paramètre K du G333.

FEDRAT : Correspond au paramètre F du G333.

ZFEED : Correspond au paramètre V du G333.

FINFED : Correspond au paramètre E du G333.

FEDTO : Correspond au paramètre W du G333.

FINRPM : Correspond au paramètre U du G333.

Pochette circulaire hélicoïdale CW

Ce cycle permet de réaliser l'usinage interne d'une pochette de géométrie circulaire conique ou non sur la MOCN Hitachi-Seiki VS-50. L'usinage est réalisé de manière hélicoïdale et les déplacements d'outils associés à ce cycles sont présentés à la figure 27.

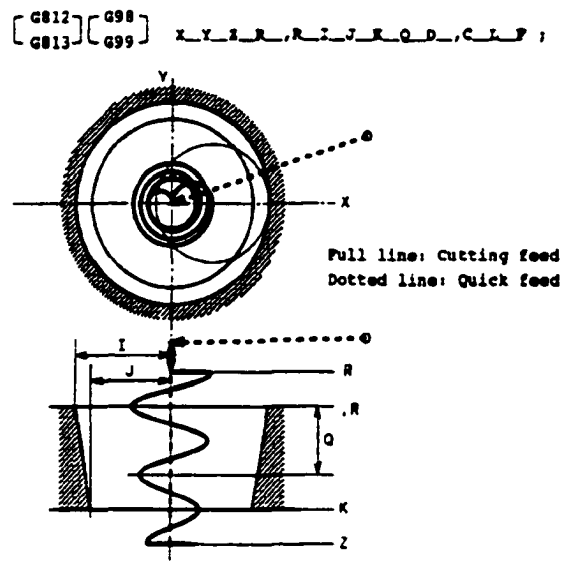


Figure 27 Cycle d'usinage de pochette conique G812 et G813. Tiré de [36].

Ce cycle est commandé soit par le code G812, résultant en un déplacement horaire de l'outil, ou par le code G813, résultant en un déplacement anti-horaire.

La description détaillée des paramètres d'usinage de ce cycle en code G est détaillée ci-dessous :

Syntaxe du code G

X, Y : Position du centre de la pochette.

Z : Coordonnée minimale atteinte par l'outil durant l'usinage.

R : Position d'approche de l'outil selon Z.

,R : Position selon Z de la face supérieure de la matière à usiner

I : Rayon final de la pochette au niveau ,R.

I+ : Usinage conventionnel.

I- : I est appliquée au niveau de R et non ,R.

J : Rayon final de la pochette au niveau K. Par défaut J = I.

K : Position selon Z de la face inférieure de la matière à usiner.

Q : Pas de l'hélice de déplacement de l'outil.

Q+ : Usinage d'un cercle complet au fond de la pochette.

Q- : Pas d'usinage d'un cercle complet au fond de la pochette.

D : Numéro de la compensation en diamètre de l'outil.

,C : Rayon de la fraise en bout sphérique utilisée pour l'usinage. Si omis, une fraise en bout cylindrique est supposée.

L : Nombre de répétition de la passe de finition au fond de la pochette.

F : Vitesse d'avance de l'outil.

Syntaxe du CL-File Pro/CYCLES

La syntaxe générée dans le CL-File par Pro/CYCLES et qui sera transformée en G812 ou G813 par le post-processeur est la suivante :

Usinage Horaire (G812) :

CYCLE / HELICL, **CLW**, HSM, DEPTH, $P1, DIAMET, $P2, ZSTEP, $P3, FEDRAT, $P4, [XAXIS, $P5, YAXIS, $P6], [CLEAR, $P7], [HIGH, $P8], [BTMDIA, $P9], [LOW, $P10], [TYPE, $P11], [FINCUT, $P12]

Usinage Horaire (G813) :

CYCLE / HELICL, **CCLW**, HSM, DEPTH, $P1, DIAMET, $P2, ZSTEP, $P3, FEDRAT, $P4, [XAXIS, $P5, YAXIS, $P6], [CLEAR, $P7], [HIGH, $P8], [BTMDIA, $P9], [LOW, $P10], [TYPE, $P11], [FINCUT, $P12]

La description détaillé des paramètres est la suivante.

DEPTH : Correspond au paramètre Z du G812 et G813.

DIAMET : Correspond au double du paramètre I du G812 et G813.

ZSTEP : Correspond au paramètre Q du G812 et G813.

FEDRAT : Correspond au paramètre F du G812 et G813.

XAXIS : Correspond au paramètre X du G812 et G813.

YAXIS : Correspond au paramètre Y du G812 et G813.

CLEAR : Correspond au paramètre R du G812 et G813.

HIGH : Correspond au paramètre ,R du G812 et G813.

BTMDIA : Correspond au paramètre J du G812 et G813.

LOW : Correspond au paramètre K du G812 et G813.

TYPE : Correspond au paramètre ,C du G812 et G813.

FINCUT : Correspond au paramètre L du G812 et G813.

BIBLIOGRAPHIE

[1] McMahon, C., Browne, J. (1998). *CADCAM : principles, practice and manufacturing management.* (2e éd.). Essex : Addisson–Wesley.

[2] Technical Committee J7, Programming Language APT. *Background of technical Committee J7.* "http ://user.icx.net/ bwilson/apt/j7backgr.htm". Consulté le 12 décembre 2000.

[3] Heindenhain. (1993). *User's manual Heidenhain Conversational Programming TNC 425 TNC 415B TNC 407.* Traunreut : Heidenhain.

[4] Mazak. (1992) *Mazak Mazatrol Programming Manual.* Yamazaki Mazak Corporation.

[5] Houtzeel Manufacturing Systems. (1994). *APT Language Reference Manual.* Waltham : Houtzeel Manufacturing Systems, Inc.

[6] Modern Machine Shop Online. *Understanding NURBS Interpolation.* "http ://www.-mmsonline.com/articles/079901.html". Consulté le 14 avril 2001.

[7] Modern Machine Shop Online. *Interpolating Curves.* "http ://www.mmsonline.com-/articles/109704.html". Consulté le 14 avril 2001.

[8] NASA - National Aeornautics & Space Administration, NASA STEP Central. *STEP Application handbook.* "http ://step.nasa.gov/docs/-STEP_Application_Handbook.pdf". Consuté le 20 juillet 2001.

[9] Modern Machine Shop Online. *STEP NC - The End Of G-Codes ?.* "http ://www.mmsonline.com/articles/070001.html". Consulté le 28 mai 2001.

[10] PDES, Inc. *PDES Program Overview.* "http ://pdesinc.aticorp.org/whatsnew/flier/-index.html". Consulté le 12 juillet 2001.

[11] PTC, Parametric Technology Corporation. *Introducing Granit One.* "http ://www.-ptc.com/products/granite/index.htm". Consulté le 4 avril 2001.

[12] Chandrasekaran, N. (1994). *Generation of Feature-Based Models.* Technical Research Report, University of Maryland, Maryland.

[13] Gupta, S.K. (1992). *Generation of Alternative Feature-Based Models and Precedence Ordering for Machining Applications.* Technical Research Report, University of Maryland, Maryland.

[14] Regli, W.C., Gupta, S.K., Nau, D.S. (1995). *Extracting Alternative Machining Features : Algorithmic Approach.* Technical Research Report, University of Maryland, Maryland.

[15] Hardwick, M., Loffredo, D. (2001). *STEP into NC Manufacturing Engineering.* 126(1), pp. 38-50.

[16] STEP Tools, Inc. *Super Model Project.* "http ://www.steptools.com/library/stepnc/-smp.html". Consulté le 20 novembre 2000.

[17] Electronic Industries Association, EIA Standard. (1992). *32 Bit Binary CL (BCL) and 7 Bit ASCII CL (ACL) Exchange Input Format for Numerically Controlled Machines, EIA-494-B* Washington : Electronic Industries Association.

[18] Numerical Control BCL Standards Association, NCBSA. *History of BCL.* "http ://www.ncbsa.org/bcl/history/history.htm". Consulté le 10 décembre 2000.

[19] Numerical Control BCL Standards Association, NCBSA. *BCL Implementation.* "http ://www.ncbsa.org/bcl/implementation/implementation.htm". Consulté le 10 décembre 2000.

[20] Modern Machine Shop Online. *What's Wrong With Postprocessors ?.* "http ://www.mmsonline.com/articles/mtg9808.html". Consulté le 10 novembre 2000.

[21] Bruce, R.G. (1990) NC Program Protability Through Decentralized Postprocessing. *Autofact '90* Society of Manufacturing Engineering, pp. 11-31 – 11-40.

[22] Suh, S.H., Noh, S.K., Choi, Y.J. (1995). A PC-Based Retrofitting Toward CAD/CAM/CNC Integration. *Computer and Industrial Engineering*, 28(1), pp. 133-146.

[23] Blumfield, A., Shpitalni, M., Lenz, E. (1992) A Generator for Creating Adaptive Post Processors. *Annals of the CIRP*, 41(1), pp. 527-530.

[24] Wright, P., Schofield, S., Wang, F.C. (1996). *Open Architecture Control for Machine Tool.* The University of California, Berkeley.

[25] Chappel, IT. (1983). The use of vectors to simulate material removed by numerically controlled milling. *Computer-Aided Design*, 15(3), pp. 156-158

[26] Wang, WP. (1988). Solid Modeling for optimizing metal removal of three-dimensionnal NC end milling. *Journal of manufacturing Systems* 7(1)

[27] Roy, U., Xu, Y. (1999). Computation of a geometric model of a machined part from its NC machining programs. *Computer-Aided Design*, 31 pp. 401-411.

[28] Ippolito, R., Iuliano, L., Vezzetti, E. (1999). A New CAM/CNC Interface for High Speed Milling. *Advanced Manufacturing Systems and Technology, CISM Courses and Lectures*, 406, pp. 231–240.

[29] Farouke, R.T., Manjunathaiah, J. Yuan, G.F. (1999). G Codes for the specification of Pythagorean-hodograph tool paths and associated feedrate functions of open-architecture CNC machines. *International Journal of Machine Tools & Manufacture*, 39, pp. 123–142.

[30] Modern Machine Shop Online. *Hitachi Seiki Introduces Open CNC/PC Network Connectivity.* "http ://www.mmsonline.com/articles/0699scan2.html". Consulté le 12 février 2001.

[31] Hillaire, R. (2000). Whatever Happened to Open Controls ? *Manufacturing Engineering* 124(6), pp. 80–88.

[32] Altintas, Y., Erol, N.A. (1998). Open Architecture Modular Tool Kit for Motion and Machining Process Control. *Annals of the CIRP*, 47(1), pp. 295–300.

[33] Zhang, L., Deng, J., Chang, S.C.-F. (2000). A Next Generation NC Machining System Based on NC Feature Unit and Real-Time Tool-Path Generation. *The International Journal of Advanced Manufacturing Technology*, 2000(16), pp. 889–901.

[34] Zuo, J., Chen, Y.P., Zhou, Z.D., Nee, A.Y.C., Wong, Y.S., Zhang, Y.F. (2000). Building Open CNC Systems with Software IC Chips Based on Sortware Reuse. *The International Journal of Advanced Manufacturing Technology*, 2000(16), pp. 643–648.

[35] Zietarski, S. (2001). System integrated product design, CNC programming and postprocessing for three-axis lathes. *Journal of Material Processing Technology*, 109(2001), pp. 294–299.

[36] Hitachi Seiki Corporation Limited. (1998). *Seiki-Seicos* $\sum$ *10M/16M/18M Instruction Manual Programming*