OSEC-II プロジェクト報告書（要約版）
OSEC仕様オープンコントローラの開発

OSE協議会

＜目次＞

＜図リスト＞

＜表リスト＞

巻頭言

東芝機械、豊田工機、ヤマザキマザックの工作機械3社と日本アイ・ビー・エム、三菱電機および情報システム開発会社エス・エム・エルの6社は1994年12月に次世代の産業オートメーション構築の要となるCNC等のFA制御機器装置一般のオープンアーキテクチャ化を目指したOSE (Open System Environment)研究会を結成致しました。その後このFAオープン化活動に賛同、協力する企業及び団体が新たに加わり、現在は18社1団体の参加メンバが活発な推進活動を続けております。

ご承知のように既に欧米においてはCNCを中心とした広範なFA制御機器（NC工作機械、ロボット等の制御機器が対象）に柔軟に対応できる幾つかのオープンアーキテクチャの提案が成されております。

OSEC (Open System Environment for Controllers)は前記OSE研究会の成果を纏めて新たに提案する制御装置の略称であります。OSE研究会の精神は『オープン』を目的にしておりますので、そこで議論されたアーキテクチャおよびインタフェース仕様書は一般に公表、提案すると同時に将来のCALS端末用FA制御機器の標準化にも些かでもお役に立てることを願っております。

今回のドキュメントは、1995年9月に公表したOSEC Version 1.0 (OSEC-I)についでOSEC Version 2.0(OSEC-II)に相当するものでありますが、このドキュメントも『オープン』の性格上一般に広く公開することと致しました。われわれの成果はOSECメンバで独占することなく、多くのユーザ各位およびベンダ各位に開示し、その広範なご意見を頂戴してより完全を期したいと考えております。

各位がOSEC仕様に準じた制御装置をお使い戴ければ、よりユーザオリエンテッドで、よりユーザフレンドリなオープンCNCが自由に取捨選択できる真のマルチベンダ環境が創出されるものと確信しております。OSE研究会は今後ともその活動の透明性を確保しつつ、企業や団体・個人等が自由に参画して具体的活動が継承できるように、関係諸機関とも協力して開発体制を整備して参る所存であります。

我々の今回のOSEC仕様公開がわが国の次世代コントローラ開発の積極的な競合開発環境の整備や関連業界の活性化に少しでもお役にたてば誠に幸甚であります。

OSE研究会（音順）

株式会社エス・エム・エル、株式会社唐津鉄工所、株式会社小松製作所、
株式会社ツガミ、 株式会社新潟鐵工所、株式会社日平トヤマ、
株式会社安永、
クボテック株式会社、財団法人機械振興協会技術研究所、
ソニー・プレシジョン・テクノロジー株式会社、東芝機械株式会社、
豊田工機株式会社、日本アイ・ビー・エム株式会社、
富士機械製造株式会社、富士電機株式会社、ブラザー工業株式会社
三菱電機株式会社、ヤマザキマザック株式会社、ユアサ商事株式会社

1 OSE研究会の活動の概要

1.1 目的

CNCの基本機能とは予め与えられた空間形状情報を制御対象の駆動装置を動かすための制御情報に変換するための１種の情報変換装置である（図1-1）。工作機械はCNCの基本機能に加工作業を行うための種々の支援機能が集約されたものである。図1-2に示すように従来のCNCはその機能がブラックボックスになっていた。

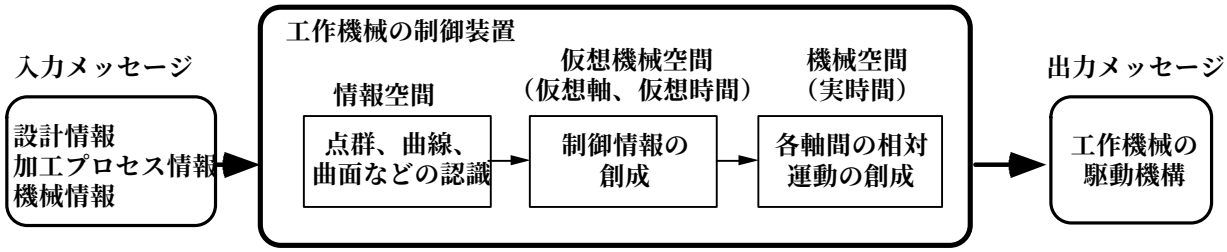

図 1-1　CNCの基本機能

OSE研究会の使命は、コントローラの原点に立ち返って、ネットワークをベースとする将来の生産システムのオープン化のために、産業機械のコントローラのアーキテクチャ及び標準I/Fに関する研究開発を行うことである。そしてその結果として、ユーザの生産環境や目的に応じて産業機械を自由に再構築可能で、保守や保全が容易で、ライフサイクルコストを最小化できる、オープンな生産システム環境を実現することである。

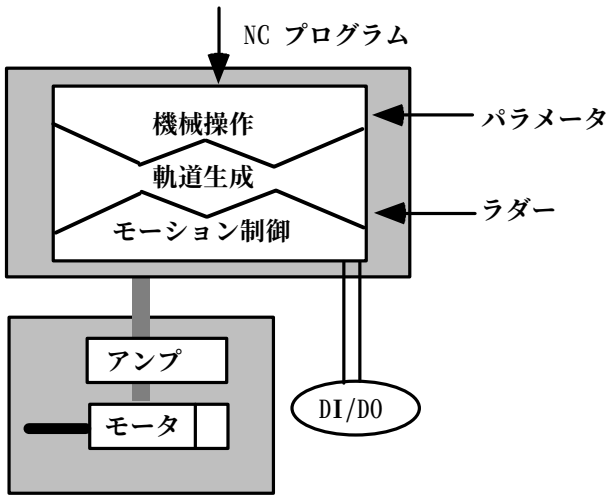

図 1-2　従来のCNC

1.2　スケジュールと成果

OSE研究会のこれまでの主な活動経過は以下のとおりである。

■ 1994年

12月　　OSE研究会発足（OSEC-Iプロジェクト開始）

■ 1995年

9月25日 OSEC-I発表と公開デモ（於・刈谷市豊田工機）
OSEC Architecture Version 1.0 (OSEC-I)
OSEC-I準拠の実証システム2台とCAD/CAMステーションによる公開デモ
インターネットによるOSEC-I仕様公開
ホームページ http://www.sml.co.jp/OSEC/
11月 OSEC-IIプロジェクト開始、OSE研究会メンバ公募
（1997年4月現在の参加メンバ：18社1団体）

■ 1996年
8月26日 OSEC-II発表（於・東京都芝公園機械振興会館）
OSEC Architecture Version 2.0 (OSEC-II)
8月27～28日 OSEC-II実証システム公開デモ
（於・東久留米市機械振興協会技術研究所）
OSEC-IIに準拠した実証システム3台、CAD入力から加工までのOSEL処理システム、生産ネットワークによる稼動監視などによる公開デモ
11月 日本国際工作機械見本市(JIMTOF)におけるOSEC-II準拠システム展示

1.3 組織

OSEC-IIにおけるOSE研究会の組織構成を図1-3に示す。運営委員会はOSE研究会の運営方針や対外的な活動を決定する最高機関であり、実行委員会はその下で実務的な活動を行う。制御、MMI、OSEL、ネットワークの各WGは、オープンコントローラに対してそれぞれ機械、操作員、CAD系、生産ネットワークからみた付加価値のあり方を中心に討論してきた。アーキテクチャWGはOSECアーキテクチャを大局的な視点から確認する場であり、各WGから提案される仕様の整合性を検証することと取りまとめを中心に行ってきた。

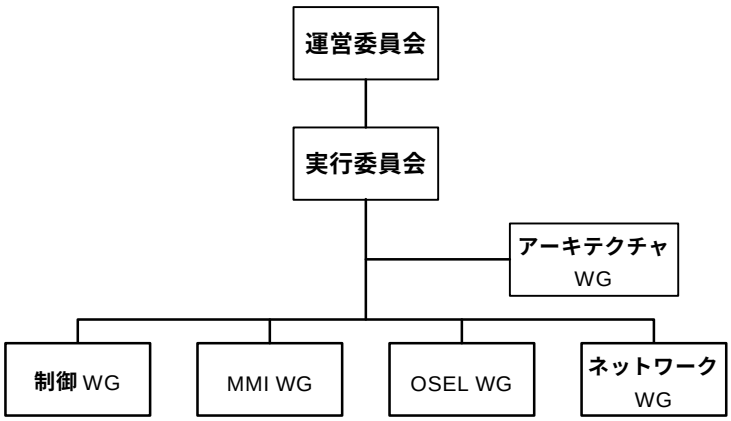

図 1-3 OSE研究会の組織構成

1.4 OSEC-Iの概要

OSEC-Iでは図1-4に示す参照モデルを規定することで真の『オープン』コントローラのためのアーキテクチャ策定を目指した。このOSEC-Iアーキテクチャでは、コントローラは生産システムの一部として定義され、設計データから加工までの情報の処理手順が明確にされている。具体的には、今までブラックボックスだったCNCを含めた生産システム

を、処理階層にまとめて、入力処理を主に行う階層、演算を行い工具軌跡を創成する階層、制御を主とする階層などの７階層に分類し、各階層で処理すべき内容を定義した。

またOSEC-Iでは、1960年代から使われてきたが柔軟性、拡張性、可搬性などの点において問題が数多く指摘されるEIAコード（Gコード）に代わるものとして、FA機器記述言語も提案した。FA機器記述言語は式、繰り返し、条件分岐といったプログラムの制御構造を持つスクリプト記述型言語をベースにした。さらに、工作機械の加工プログラムのための拡張として、工作機械の仕様/工具/治具/入出力接点など機械依存情報の抽象化、自由曲線/自由曲面の数式による直接表現、原始的な軸動作記述などの機能を追加している。マクロや関数からなるライブラリによって加工プログラムのモジュール化をすすめるという効果を目指した。

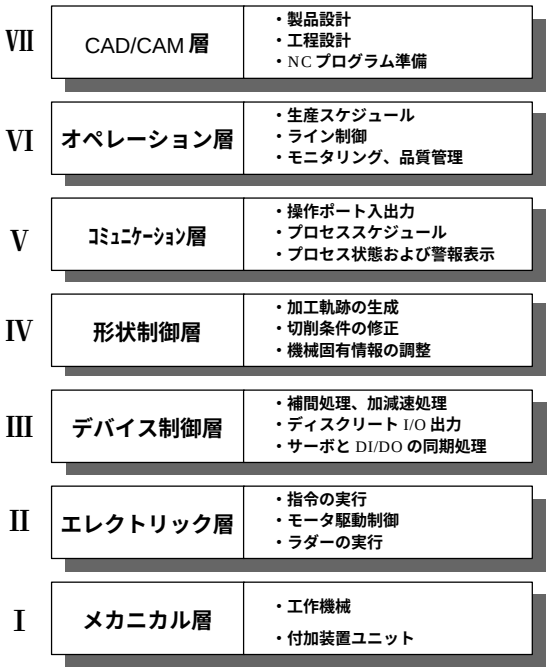

図 1-4　参照モデル

1.5　OSEC-IからOSEC-IIへ

OSEC-Iにおけるアーキテクチャ参照モデルは設計データから加工までの情報の処理手順を明確にしたもので、生産システムにおける処理を７階層に分類している。今までブラックボックスだったCNCを含めての処理階層の分類であり、各階層で処理すべき内容が定義された。しかし実際のコントローラを実装するにはあまりに抽象度の高いモデルである。そこでOSEC-IIでは、抽象的な参照モデル(OSEC-I)、機能とインタフェースが明確に定義されたアーキテクチャモデル、そして実世界のハードウェアとソフトウェアに直接マッピング可能な実装モデルという３種類の記述レベルを設定した。制御系ソフトウェアの部品化や操作系ソフトウェアから見た機械資源のモデル（マシンリソースオブジェクト）はこのアーキテクチャモデルから考えられたものである。

OSEC-IにおけるFA機器記述言語は産業機械の主として制御を目的に考えられた言語で、従来のEIAコードに欠けていた表現の柔軟性や拡張性と、加工プログラムの可搬性の向上を狙っていたものである。OSEC-IIにおける加工法記述言語OSELはさらに一歩踏み

込んで、工作機械を活用していく上での加工のノウハウがソフトウェア部品として流通・調達できるよう、加工法のクラスライブラリ化に焦点を絞った。CAD/CAMとのインタフェースを考えて、穴・溝といった加工特徴に対しての加工法の与え方の議論がなされた。コンピュータのプログラミング言語への類似を求めるなら、EIAコードは低級なアセンブラ言語、FA機器記述言語が柔軟性・拡張性に富んだC言語、OSELはより抽象度が高められ、さらなるソフトウェアの部品化が可能なオブジェクト指向言語C++、といったように位置づけられる。

2 OSECアーキテクチャの理念

2.1 アーキテクチャのねらい

OSECアーキテクチャは、産業機械のコントローラを標準的なプラットフォームとして定義し、それを広く世の中に開放することを目指して考案されている。 そのねらいは、エンドユーザ、機械メーカ、コントローラベンダ、ソフトウェアベンダ、システムインテグレータなどが、それぞれ独自の付加価値を産業機械に容易に加える事がきるようにすることで、産業機械の技術的、商業的な発展を加速することである。
産業機械のオープンコントローラに対する付加価値を考察するに当たり、コントローラを取り巻く環境として機械、操作員、CAD系、生産ネットワーク、という４つの側面を考え（図2-1)、これらの側面から見て付加価値をいかに与えやすくするかという点からOSECアーキテクチャを議論した。逆の見方をすれば、生産に関わる機械、操作員、CAD系、生産ネットワーク、という４つの事物を有機的に結合し、融合する仕組みが、OSECアーキテクチャの目指すオープンコントローラに他ならない。

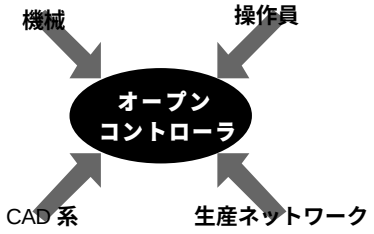

図 2-1 オープンコントローラに対する４つの側面

（1）機械から見たオープンコントローラ
制御系のソフトウェアやハードウェアを交換可能にする一つの重要な鍵は、制御ソフトウェアの部品化である。機能毎および実時間性への依存度毎に部品化された制御ソフトウェアがメッセージの授受という形で一つのシステムにまとめられるというシステム概念を基本として、標準的なインタフェース（OSEC API）をメッセージの形式で定義した。
（2）操作員から見たオープンコントローラ
操作盤などのソフトウェアを作り、改良する効率を上げるという観点から、アプリケーションフレームワークというソフトウェア手法を取り入れる事にした。これにより様々なマンマシンインタフェース機能を必要に応じて取捨選択して実装する事が可能となる。操作の対象となる産業機械は、標準的な機械資源（マシンリソースオブジェクト）として規定する。操作系のソフトウェアをこの標準的なオブジェクトに対して開発することにより、その可搬性も保証される。
（3）CAD系から見たオープンコントローラ
CAD系で定義された形状を加工するためのソフトウェアの開発効率を高めるという観点から、加工ノウハウのソフトウェアを部品化し、それに基づいて加工法を記述するという新しい加工記述言語OSEL (OSE Language)を提案している。OSELにおける加工ノウハウのソフトウェア部品化の考え方は、現場における固有の生産技術をソフトウェア部品化し、ネットワークを利用して流通、調達する新たなビジネス形態へのパラダイムシフトを促すものでもある。

（4）生産ネットワークから見たオープンコントローラ

特に生産ラインの稼働監視という観点に注目して、稼働情報の送出およびその収集、集計を行うインタフェースプロトコルOFMP (OSE Floor Management Protocol)を定義し、これをインターネット応用技術で取り扱う仕組みを提案している。なおプロジェクトの進行状況により、本ドラフトでは限定的な開示となる。

2.2 アーキテクチャ策定における基本方針

- 既存技術に新技術を漸次追加して発展させる進化的(Evolutionary)なシステムとする。ソフトウェアを指向したアーキテクチャとし、機能や目的に応じてソフトウェアを部品化することでアーキテクチャを開放型にする。
- マルチプラットフォームに対応したアーキテクチャとする。
- 業界で標準化がすすむ制御用コンポーネントへの対応を考慮する。また、CAD/CAMを含めた開発から設計への垂直的な業務の流れにおける位置づけも考慮する。
- 分散環境下にある生産現場で利用されることを想定する。CALSなどの統合化された開発生産環境における生産用端末として利用されることも考慮する。

2.3 基盤となる技術

オープンシステムでは、システムの全体とその構成要素とを分離して開発することになる。その場合、目標とする全体システムの技術的なレベルはそれを支える構成要素技術に強く依存するが、逆に、基盤となる要素技術がある程度変化してもシステム全体としては本質的な変化をせずに済むように出来ることもオープンシステムの特徴である。このことは現在非常に大きな変革期を迎つつあるデジタル情報処理システムおよびそれに関するソフトウェア技術を利用しようとする際に特に重要な意味を持ってくる。

本プロジェクトでは、産業機械制御システムをオープンシステムとして構成する際に現時点で考慮しておかなければならない基盤技術として、以下の項目を考慮にいれた。このリストは、時とともに変化して行くはずのものである。

■ ソフトウェア開発手法	オブジェクト指向技術
	アプリケーションフレームワーク
	プラットフォームに独立の処理体系
■ 制御システム技術	デジタル制御
	分散制御
■ ネットワーク技術	インターネット／イントラネット
■ CAD/CAM技術	フィーチャーベースモデリング

3 OSECアーキテクチャの概要

本章では、第2章で述べたオープンコントローラに対する4つの側面を支える機能要素をOSECではどのように規定しているかについて述べる。

3.1 制御系ソフトウェアの部品化: OSEC API

機械制御系ソフトウェアの世界は、これまでブラックボックスといわれてきた。本研究会では先ずNC装置の内部の制御系ソフトウェアを現在のソフトウェア開発手法を意識して一旦全て明示的に細部にわたって明らかにし、その上で制御系ソフトウェアを整理、部品化する作業を行って、制御用ソフトウェア構成のアーキテクチャを明らかにする試みを行った。具体的には、制御階層を実装可能なレベルまでブレークダウンし、目的、処理内容、要求される実時間性などの点から機能毎に分類し、一つ一つの機能のまとまりを『機能ブロック』と呼ばれるソフトウェア部品として再構成した。こうしてソフトウェアシステム全体の「基本設計図」を明らかにすることにより、制御系ソフトウェアが機能ブロック単位でプラグアンドプレイできることを目指している。またOSEC-Iの階層に対応するものとして、機能ブロックのうち同じレベルにあるものをまとめた『機能群』を規定した(表3-1参照)。

表 3-1 OSECアーキテクチャにおける制御系ソフトウェアの機能群と機能ブロック

機能群	機能ブロック	要求実時間性	サービスの概略
リソース管理	リソース制御	(32～100msec)	機械資源および操作系ソフトウェアからのイベントを処理する。
動作生成	OSEL実行	(8～100msec)	OSELによる加工プログラムの最終出力であるOSEL-Xを解釈し動作生成を行う。
	EIAコードデコーダ		EIAコードによる加工プログラムを解釈し動作生成を行う。
	手動制御		ジョグやダイアルなどマニュアル入力による動作生成を扱う。
機械制御	マシン制御	8～16msec	補間と加減速処理、サーボとI/Oの同期などを行う。
デバイス制御	サーボ制御	1～2msec	各軸のサーボ制御を行う。
	PLC DI/DO	20～50msec	DIからの信号入力を受けてラダープログラムを解釈しDOに信号を出力する。

表3-1は各機能群および機能ブロックのサービスの概略をまとめたものである。参考のため、各機能ブロックを再構成する上で想定した、要求される実時間性も併せて表す。

概念的には、各機能ブロックをオブジェクトの形でソフトウェア部品化し、これらオブジェクトに対するメッセージをインタフェイスプロトコル(OSEC API)として定義するという方針をたてた（付録参照)。こうすることで、各機能ブロックをオブジェクトとしてカプセル化出来、アーキテクチャとしては機能ブロックがメッセージを受けてそれに対して内部で処理をどのように行うかには具体的には立ち入らなくて良いことになる。つまり処理の実装に関しては機能ブロックの開発者が設計、制作の自由度を保証されている。こうしたカプセル化により、企業や個人が個別に独自の付加価値を加えやすくなる。また処理方法を規定しないことは、各機能ブロック間の結合関係を規定しないことも意味する。OSECアーキテクチャではあくまで論理的な機能ブロックのサービスの内容とそれら機能ブロックに対する論理的なメッセージを定めているだけである。このためOSECアーキテクチャモデルとしては、各機能ブロックは階層構造にはならず、各機能ブロックに対するメッセージ(OSEC API)を介して並列に接続される形となる（図 3-1)。

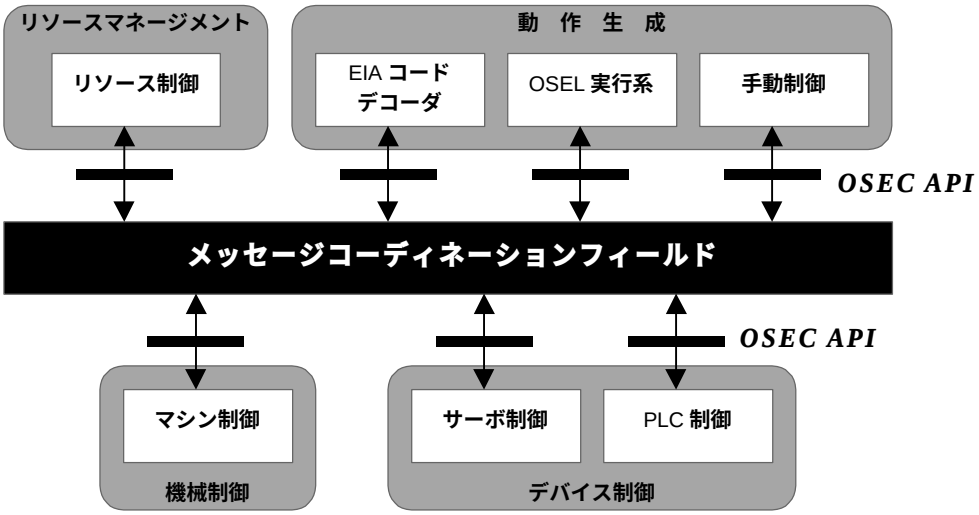

図 3-1　OSECアーキテクチャモデル

またモータ、I/Oといった制御コンポーネントハードウェアのプラグアンドプレイを可能にするために、各ハードウェアはデバイス制御機能群の中に隠蔽される。サーボ制御、PLC制御などの対応した機能ブロックへのメッセージがこうしたハードウェアへの唯一のインタフェースとなっている。

OSECアーキテクチャは、出来る限り広い範囲の制御システムを包含できるような概念として一般化を行った結果策定されたものなので、それ自体は抽象的なものである。したがって、OSECアーキテクチャは論理的には一意的に機能ブロック構成が規定されているが、これらの各機能ブロックを具体的に実装するときには、デバイスドライバ、プロセス、プロセス間通信といったシステムの仕組、静的なライブラリやDLLといった組込の手法、利用する制御カードなどのハードウェア要因、あるいは各種ソフトウェアの実行制御/監視用に追加されるソフトウェアモジュールなどの実装方式について多くの選択肢が存在し、一意的にシステムが決定、もしくは限定されることはない。つまり、アーキテクチャを実装するための実装モデルは1つには限定されず、システムの規模や構成するハードウェアなど実装の形態あるいは使われ方に応じて自由な発想がそこに盛り込める余地が保証されることになる。4章の図4-1は、特に各機能ブロック間の結合関係を明示的にした実装モデルである。

3.2　操作系ソフトウェアから見た機械の標準モデル: マシンリソースオブジェクト

コントローラのマンマシンインタフェースは、それぞれの機械に適した操作環境を実現するという点で機械メーカやエンドユーザなどにとって非常に関心の高い部分である。オープンコントローラでは機械に対する標準的なインタフェースを明確に定義することによりソフトウェアの独立性を保証することを目指しているので、機械メーカやエンドユーザなどが自由に操作系ソフトウェアを構築し改良できるものと期待を集めている。
機械メーカやエンドユーザなどが自由に操作系ソフトウェアを構築したり改良していくためには、以下の事項を考慮する必要がある。

- ●ソフトウェア部品開発ツール
- ●GUI部品など構成要素の再利用性

●既存のアプリケーションソフトウェア製品との相互連携
●最新のソフトウェア技術への対応
●開発した操作系ソフトウェアの可搬性

操作系ソフトウェアに関するOSECアーキテクチャでは、こうした要求事項を満たすための手段として「アプリケーションフレームワーク」開発手法を全面的に取り入れた。
一般に、操作画面に関するソフトウェアのひな型としては『モデル(Model)・ビュー(View)・コントロール(Control)』という3つの部分(MVC)に分けて考えると漏れなく機能を網羅しやすい。機械類の操作系ソフトウェアの場合をこれにあてはめてみると、『モデル』は操作する対象である機械のモデルになり、『ビュー』は画面およびGUI部品など画面要素になり、『コントロール』はモデルとビューとの関わり方を規定するもので、GUI部品のボタンが押されたときの処理や機械の状態が変わったときの表示変更の処理などをまとめたものになる。OSECでは、機械のモデルをオブジェクトの形で規定して『マシンリソースオブジェクト』と呼んでいる。

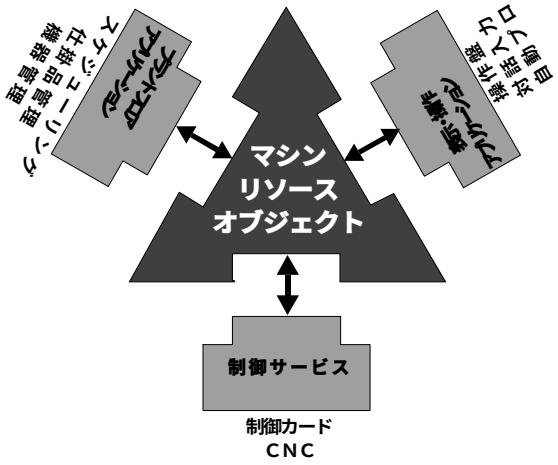

図 3-2　マシンリソースオブジェクトの概念

このマシンリソースオブジェクトは、操作系のアプリケーションから見たときの機械を抽象化することによってそれに対するインタフェースを標準化し、アプリケーションの可搬性、移植性、再利用性、拡張性の向上を狙ったものである。マシンリソースオブジェクトは、

■表示操作関係のアプリケーションインタフェース
■生産管理など上位プラントフロアアプリケーションとのインタフェース
■NC制御部（NC制御カード、PLC、サーボなど）とのインタフェース

という3つの標準的なインタフェースによってカプセル化される(図3-2参照)。
軸、スピンドル、ツール、PLCなど工作機械の持つ様々なリソースを論理的に分析し、各リソースの属性と機能を明示的に定義する。操作系ソフトウェアとしては、機械の各リソースオブジェクトにメッセージを投げる形で機械に指令を与えたり機械の状況を知ったりする。

このマシンリソースオブジェクトは工作機械をオブジェクトの概念でモデル化したものであるが、同じ工作機械といっても機種の違いや構成の違いなどを反映してマシンリソースオブジェクトにもバリエーションを持たせる必要がある。そこでマシンリソースオブジェクトとしては抽象的な工作機械の一般的なモデル（図4-4参照）を一つ用意し、機種や構成の違いをあらわすためには、個別の機械に対してその一般モデルから各個別機械のオブジェクト群を派生させて対応出来るようにした。こうすることによりオブジェクト指向での継承の概念が利用できるので、各機械固有のマシンリソースオブジェクトを作る効率が高められる。

3.3 加工法記述言語：OSEL

現在のNC装置で一般的に利用されているEIAコード（Gコード）の体系は1960年代に、ISOなどにより工作機械のための標準データフォーマットとして国際的に標準化が行われた。現在では、次のような問題点が指摘されようになっている。

- 工具や機械指令などに関して、個別の機械に依存したデータとなってしまう。
- NC装置メーカ間の互換性がない。
- 曲線曲面加工のためのデータ量が膨大になる。
- ユーザレベルでの拡張や改良が困難である。

こうした問題点の解決に加え、エンドユーザや機械メーカなどにあまた蓄積されている固有の生産や加工の技術を、ソフトウェアの形でパッケージ化し、再利用可能にするための手法を提案することが加工法記述言語OSEL (OSE Language)の目的である。これはさらに、ネットワークを中心とするであろう将来の産業社会の中で、ソフトウェア化された固有の生産技術が流通し調達できるような新たなビジネス環境への転換という大きなパラダイムシフトへの提言をも行うものである。

OSELにおける処理の流れの概略を図3-3に示す。

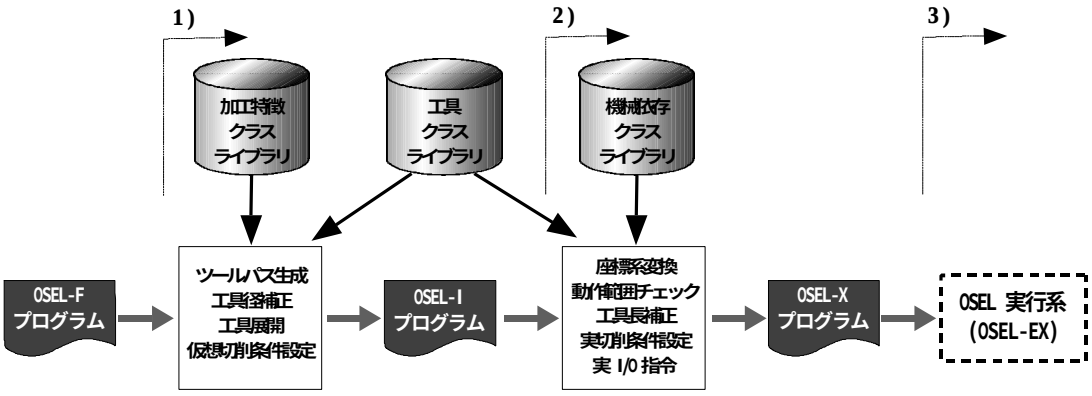

図 3-3 OSELにおける処理の流れと実装のレベル

最上位レベルでの記述であるOSEL-F (Feature OSEL)では、加工プログラムは穴や溝といった加工特徴の記述、例えば「貫通穴をドリルであけよ」といった形で入力される。OSEL-Fは加工特徴クラスライブラリと工具クラスライブラリを利用して、個別の機械とは独立した中間記述である OSEL-I (Intermediate OSEL)に展開される。OSEL-Iは機械クラスライブラリを利用して、 機械に依存した実行フォーマットであるOSEL-X (eXecutable OSEL)に展開される。実時間で処理する必要のあるのは、このOSEL-Xプロ

グラムの解釈処理部（OSEL実行系：OSEL-EX）以降である。

加工特徴クラスライブラリでは、機械に依存しない加工方法が、穴、溝、側面といった加工特徴に基づくクラス毎のメソッドの形で定義されている。機械メーカやエンドユーザなどは、基底クラスの派生クラスを自由に加えることによって、独自の加工ノウハウを加えていくことが可能である。機械クラスライブラリでは仮想的なプログラム座標系から物理的な機械座標系への変換や実I/O指令など、機械に依存した処理への展開がメソッドの形で記述されている。

オープンコントローラへのOSEL処理系の実装としては、コントローラのシステム構成やユーザ要求などに応じて、図3-3の実装レベル1)、2)、3)と示された3つのモデルから選択できる。実装モデル1)では、コントローラは一番抽象度の高いOSEL-Fプログラムを受け取り、軸制御まで一貫して行う。CAD/CAMの機能を備えた高度なコントローラではこの構成になる。実装モデル2)では、コントローラは機械独立な中間記述である OSEL-Iを受け取る。機械依存部分以下をコントローラが受けもつ形態である。実装モデル3)では、コントローラは実行フォーマットであるOSEL-Xプログラムを受け取る。低機能なシステムでも実装可能な構成である。

OSELの概念的な規定は上記のようになるが、JAVAを用いて具体的に実装してみることによって検討を行なった。

3.4 生産ネットワークにおける工作機械へのインタフェース

従来のCNCと生産管理システムとの接続では、CNC機種ごとにインタフェースが異なり、また生産システム側の標準インタフェースも存在していないので生産ネットワーク系のアプリケーションに工作機械を直接接続することは難しい。そのため工作機械の外部に配置されたセンサーやPOPターミナルなどを使って、工作機械とネットワーク系のアプリケーションとのインタフェースをとるといった作り込みを行なってきた。

工作機械を含んだ生産ネットワークを構築するために重要なことは、ネットワークのハードウエア、ソフトウエアが複数の企業から調達でき、マルチベンダ環境を実現できることであり、またネットワークから見て工作機械に対し、種類によらず同一のインタフェースでアクセスできることである。

マルチベンダ環境を実現するために、ネットワークのプロトコルとしてTCP/IP、ネットワーク上のアプリケーションのGUIとしてWebブラウザを用いることにする。また工作機械に同一のインタフェースでアクセスするために、工作機械側にネットワークオブジェクトを用意した。工程管理アプリケーションはネットワークオブジェクトと通信することにより、工作機械の相違を意識する必要はなくなる(図3-4)。ネットワークオブジェクトは3.2節で述べたリソースオブジェクトとメッセージの交換を行なう。

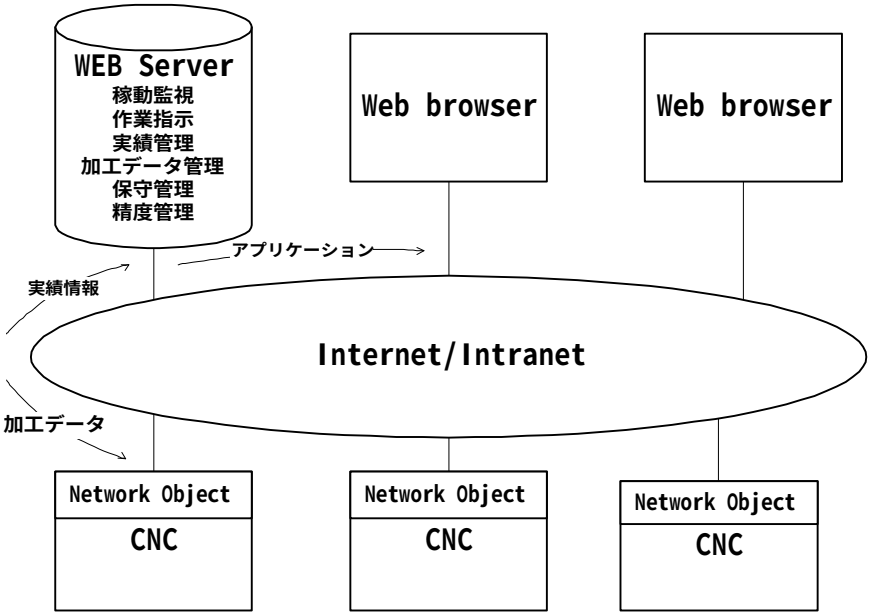

図 3-4　生産ネットワークとCNCとの直接接続

4　機能群とOSEC-APIから構成される実装モデル

本章は、ある目的の下で定義された機能群に基づくアーキテクチャから実装モデルを考え、最終的にマイクロコンピュータシステムであるコントローラのH/W及びS/Wに実装していく過程について述べる。

4.1　実装モデル

OSEC-IIではオープンCNCを構成する機能群を定義し、この機能群によりアーキテクチャを定義した。

実装モデルは、

- ■MMI機能群
- ■リソース管理機能群
- ■動作生成機能群
- ■マシン制御機能群
- ■デバイス制御（サーボ制御・PLC-DIDO）機能群

の５つの機能群から成るものとして定義した。

図4-1は機能群のプロトコルの流れを含む実装モデルである。

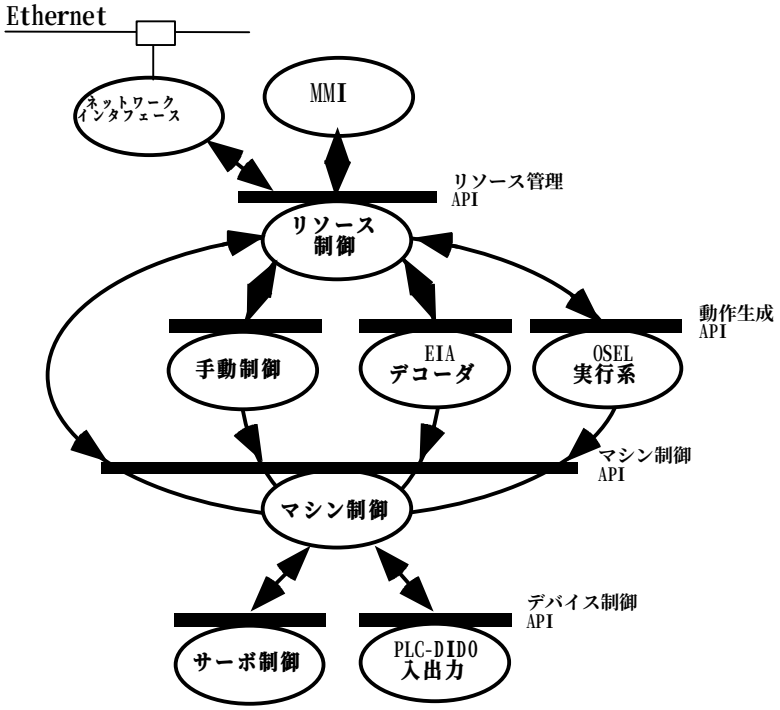

図 4-1　CNCの実装モデル

4.2　MMI（操作運用系）の実装

MMI（操作運用系）は、外部プロセスからオープンCNCの操作運用環境を介して工作機械の持つ色々な資源（リソース）に働きかけるインタフェース部分となっている。
画面作成用アプリケーションフレームワークにより、マンマシンインタフェースソフトウエアの部品化を行った。このために、工作機械のもつ色々な資源（マシンリソース）をク

ラス分けされた抽象的なオブジェクトモデルで定義し、各オブジェクトに対して属性とメソッドを割付け、抽象的な形で工作機械の資源への働きかけが行えるようにした。

実証システムの構築では、マシンリソースオブジェクトの実装と同時に基本的な操作盤機能を有するアプリケーションを作成し、マシンリソースオブジェクトのソフトウェア部品としての有効性について検証した。

図4-2は、主軸及び送り軸の情報を表示するパーツ（MpsAxesView）とオブジェクトモデルの結合の様子を示している。

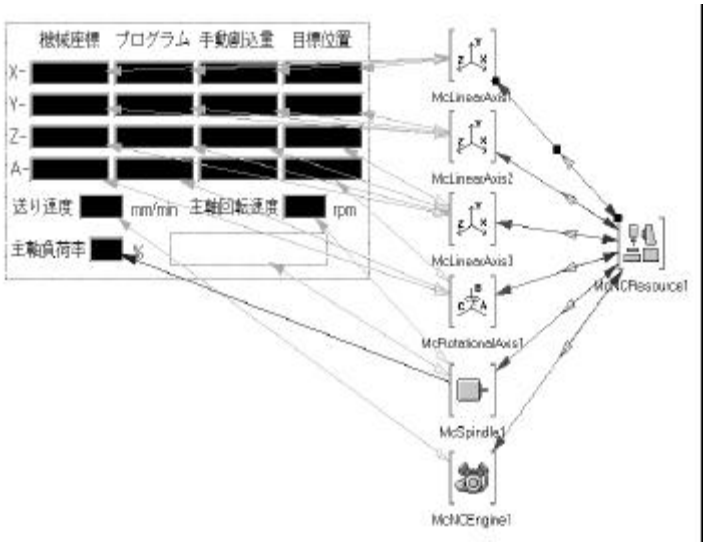

図 4-2　MpsAxesViewパーツとオブジェクトモデルの結合

表示パーツとオブジェクトモデルの接続を示す「線」は、オブジェクト間のメッセージを表している。例えば、機能モデルの送り軸の現在位置属性と表示パーツの機械座標表示ボックスが接続されている。これは現在位置の値が変更されたときに機械座標表示ボックスの内容を更新することを示している。

実証システムの開発では、MMI-WGのメンバ各社が異なる3つの操作盤アプリケーションを開発した。これらは同一の実証システム（東芝機械製マシニングセンタ）を対象としたものであったが、実際に開発された操作盤アプリケーションは、各社の操作盤機能に対する考え方を反映した、それぞれに異なる特徴を持つものとなった。

4.3　リソース管理

リソース管理は、工作機械をモデル化したマシンリソースオブジェクトと、それを現実の制御系と接続して整合をとる機能の総体としてアーキテクチャ上に設けられた。
マシンリソースオブジェクトはアプリケーションと制御機能を分離し、両者の間のインタフェースを明確に定義する。これによって工作機械メーカや工作機械ユーザが、独自の機能を持つアプリケーションを容易に組み込める環境を提供することが、マシンリソースオブジェクトの重要な役割である。

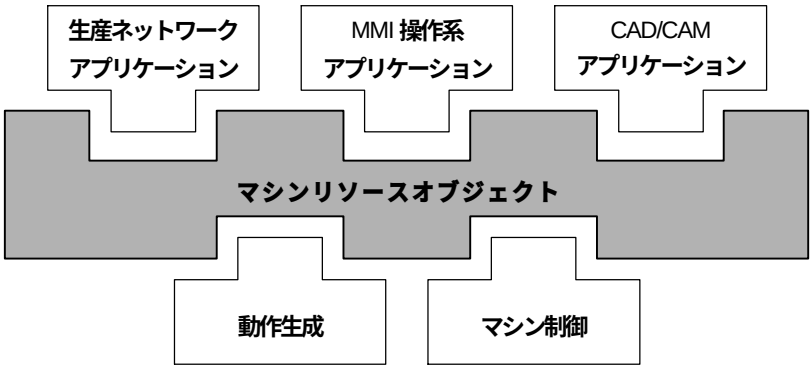

図 4-3　マシンリソースオブジェクトを中心としたリソース管理

（1）マシンリソースオブジェクトの概要

マシンリソースオブジェクトのモデル化の際には、コントローラの制御機能をドメインと呼ぶいくつかの機能グループに分けてモデル化をおこなった。
以下のようなドメインが定義されている。

- **■基本機能ドメイン**
- **■機械操作ドメイン**
- **■制御サービスドメイン**
- **■基準データ管理ドメイン**
- **■プラントオペレーションサービスドメイン**

■ 基本機能ドメイン

基本機能ドメインは工作機械およびアラームのクラスを定義している。

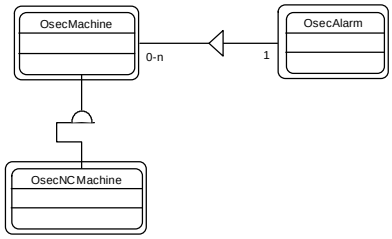

図 4-4　基本機能ドメインのオブジェクトモデル(Coad-Yourdonの記法による)

■ 機械操作ドメイン

機械操作ドメインは工作機械の運転のためのインタフェースを提供するものである。このドメインに含まれるオブジェクトの多くは、送り軸や主軸など工作機械を構成する機械部品をモデル化したものである。

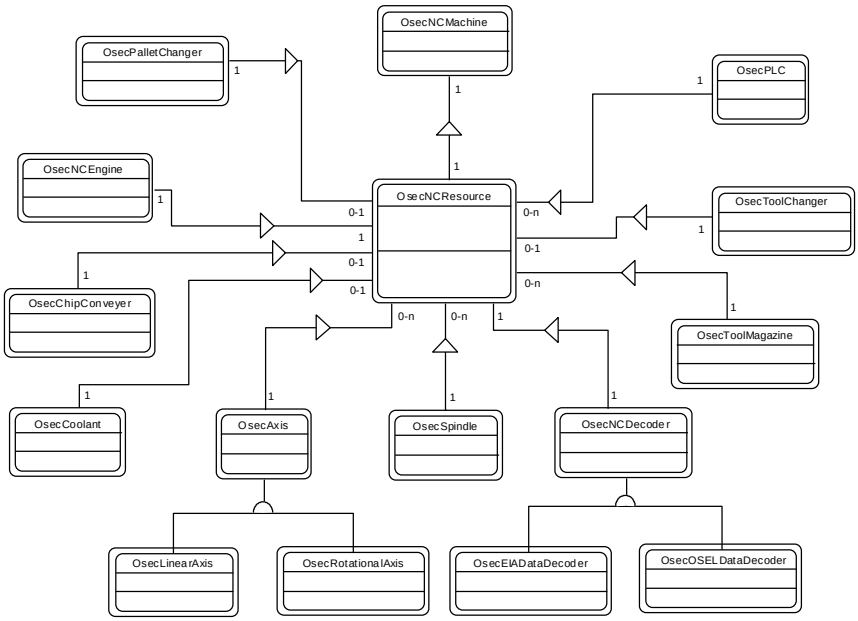

図 4-5　機械操作ドメインのオブジェクトモデル

■ 制御サービスドメイン

制御サービスドメインはマシンリソースオブジェクトとコントローラの制御機能（動作生成機能群）との結合を行う。OSEC-IIでは「リソース制御」と呼ぶ機能ブロックを設けて、マシンリソースオブジェクトから動作生成機能群へのマッピングを行うことになっているが、このリソース制御機能は制御サービスドメインの実装形態の一つとして考えることもできる。

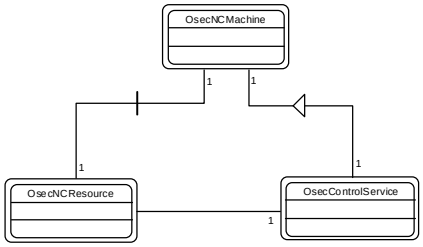

図 4-6　制御サービスドメイン・オブジェクトモデル

（2）実装

MMIやネットワークアプリケーションは標準化したマシンリソースオブジェクトに接続することにより標準化を実現している。

一方、現実の制御系では、機械制御・動作生成の実装に様々な形態があり、一意の形態の定義では困難である。このため、加工プログラムの選択、手動操作などMMI操作によって起動された操作を動作生成、機械制御への指令に変換する部分はリソース制御に組み込んでいる。また現在位置などのフィードバック情報や実行コマンド、目標位置などの指令情報、機械の稼動状況の現実の情報はマシンリソースオブジェクトへ反映され、これをリソース制御で処理している。

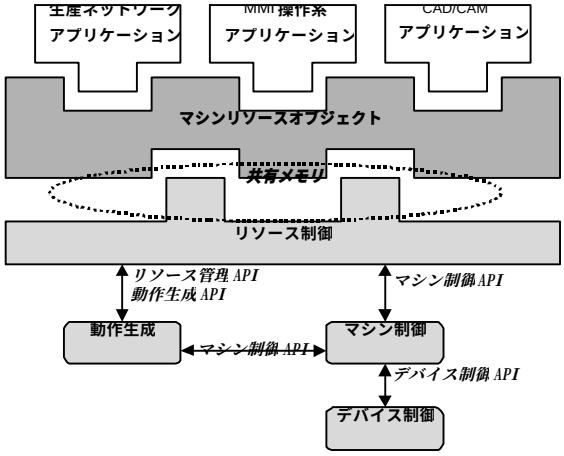

図 4-7　リソース制御 共有メモリとAPIによる実装

マシンリソースオブジェクトへの接続は、すべてオブジェクトモデルのインタフェースとして実現できる。しかし、実際には現実の制御系の機械制御、動作生成のインタフェースはオブジェクトモデルに対応していない。そこでリソース制御を、マシンリソースオブジェクトにオブジェクトモデルでインタフェースし、動作生成、機械制御とは静的なAPIで接続することにより、マシンリソースオブジェクトのリソース管理を可能にする部分として実装した。

■ マシンリソースオブジェクト

マシンリソースオブジェクトは開発ツール（VisualAge for C++）上で利用可能なソフトウェア部品として実装されている。工作機械の機能モデルは、これによって提供されるソフトウェア部品を使って構築されている。

仮想NCカード

仮想NCカード（VNCCard）は、リソース制御とのインタフェースするために設けられた。今回は、共有メモリを用いて実装されている。

共有メモリ

共有メモリでは次の2種類のデータがやり取りされている。

- ■コマンドやアラームのようにマシンリソースオブジェクトとリソース制御機能の間でその都度確実に受け渡される必要のあるデータ。
- ■送り軸の現在位置のように必要に応じてその監視を行えばよいデータ。
- ■これらは周期監視データ領域にマッピングされる。

リソース管理API

リソース管理のAPIは、共有メモリを経由したマシンリソースオブジェクトへのAPIとして定義されている。マシンリソースオブジェクトの資源と属性はこのAPIによってアクセスされる。

（3）検討課題

■ OSEC-IIのオブジェクトモデルは、システムの静的な構造を定義したものである。システムの仕様を確定するには、これに加えてシステムの動的モデルを定義する必要がある。

■ マシンリソースオブジェクトは工作機械全体に適用できるモデルを目指したものであるが、実証システムではマシニングセンタおよび3次元測定器に対する適用のみが検証された。旋盤、レーザ加工機といったマシニングセンタとは異なる構成の工作機械に対しても、モデルの適用を行っていく必要がある。

4.4 動作生成

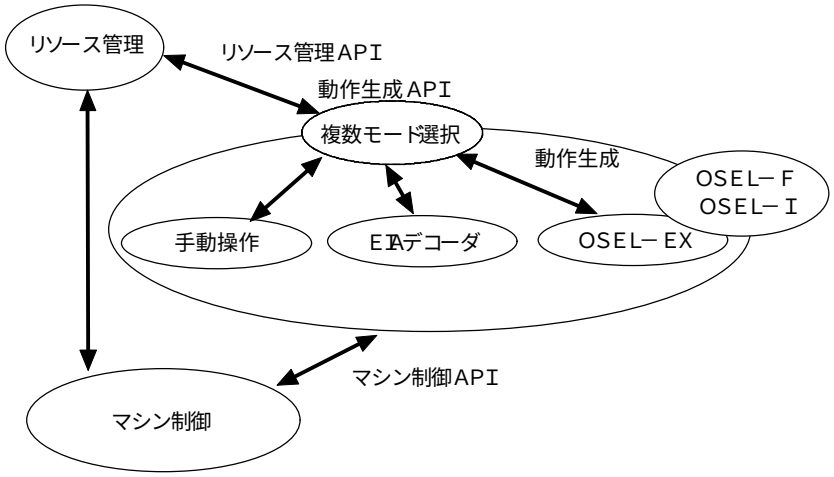

図 4-8 リソース管理、動作生成、マシン制御とAPI

動作生成は、プログラム運転や手動操作等、リソース管理からの加工に関わる指令を解釈して加工軌跡を生成し、マシン制御機能への一連の動作に変換する。リソース管理からの指令としては、CNCの加工プログラム、手動操作指令に対応している。マシン制御への指令としては、直線や円弧などの動作指令や、座標系の操作などがある。また、加工プログラムは、複数の言語に対応する。

（1）動作生成API

動作生成部では、リソース管理からの指令を受けて、マシン制御APIを介して工作機械の現実の動作を生成する。動作生成部のプログラム実行を管理するためのAPIである。初期化、プログラムの起動・停止・再開、プログラム名の設定等の機能を持つ。全ての言語処理系への指示はこのAPIを介して処理するため、APIに従った複数の言語処理系の実装が単一種類のAPIで行える。指定された加工プログラムの属性により言語処理系を選択し、起動する。今回は、OSEL言語とEIA（Gコード）デコーダを実装した。

（2）OSELデコーダ（OSEL-EX）

OSELプログラム実行を管理するためのAPIである。OSELデコーダの初期化、プログラムの起動・停止・再開、プログラム名の設定等の機能を持つ。OSELプログラムを実行する場合に、動作生成APIを介して呼び出される。

（3）EIAデコーダ

EIAプログラム実行を管理するためのAPIである。EIAデコーダの初期化、プログラムの起動・停止・再開、プログラム名の設定等の機能を持つ。EIAプログラムを実行する場合に、動作生成APIを介して呼び出される。

4.5 マシン制御

4.5.1 機能

マシン制御は、上位層から与えられる操作系の指示を契機にして、機械の送り軸・主軸などのサーボモータへの駆動指令と、バルブ・ランプなどのI/Oデバイスへの離散的な指令を生成する。さらに、サーボ系の状態(サーボモータの位置・角度・速度情報等)やPLCの情報(I/Oの接点情報とシーケンス演算処理結果)を取り込み、自身の信号同期処理で参照するとともに、上位系に伝達する機能を持つ。

4.5.2 目標と方針

アーキテクチャを規定して機能群のインタフェースを決めた後、それを具体化して実証する段階においては、パソコンにソフトウェアとして実装することがオープン指向という観点からは最もふさわしいと考えられる。その作成に当っては、参考にできる公開例がないため、今後のモデルとなるように努めた。

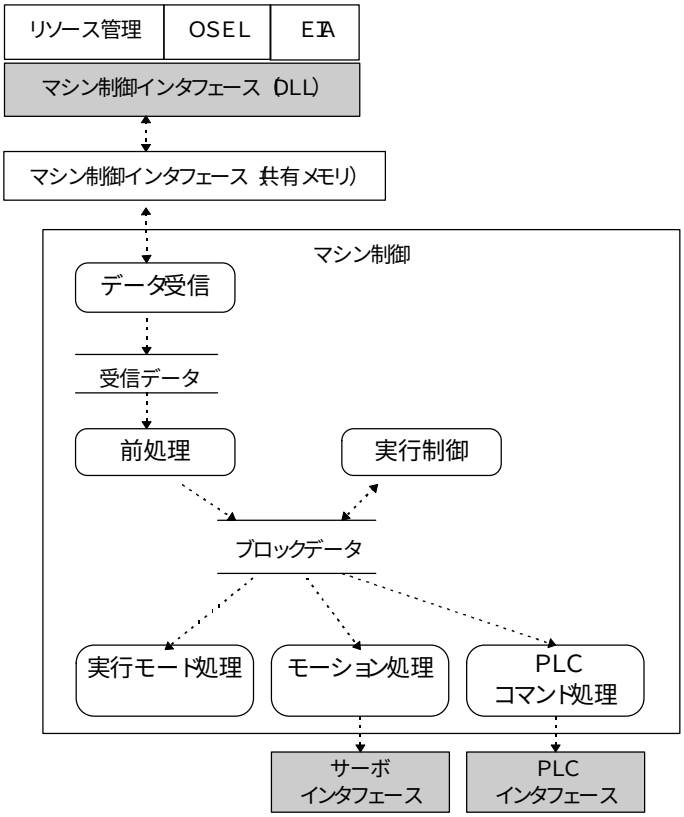

図 4-9 マシン制御の実装例

パソコンの汎用OSが実現できるリアルタイム能力は、せいぜい数10ミリ秒である。一方、サーボ軸が必要とするリアルタイム能力は位置制御ループにおいて数ミリ秒であり、現状では埋められないギャップがある。マシン制御とサーボデバイスとの機能的切り分けにはいくつかの案が考えられるが、上記の理由により、マシン制御ではサーボ軸対応での位置･速度等動作指令の生成、サーボデバイスは各サーボ軸単位でのサーボループ処理以下を分担するのが最適と考えられる。また、マシン制御とサーボデバイスは、それらのリアルタイム要求度の差異により、両者を完全に同期させることは困難である。この差異を

吸収するため、マシン制御とサーボデバイスの間にバッファ等の機構を設ける構成とする。

4.5.3 内部機能

マシン制御の内部は、下記の4つの機能に分けた。

（1）上位層とのインタフェース処理部
（2）サーボ系に対する信号生成部およびそのインタフェース処理部
（3）PLC系に対する信号生成部およびそのインタフェース処理部
（4）デバイス制御の同期処理部

その実装例を図4-9に示す。

4.5.4 マシン制御API

マシン制御が持つインタフェースとして、マシン制御APIを定めている。それらは次の6つのグループに大別される。

（1）マシン制御コマンド
（2）モード制御コマンド
（3）軸制御コマンド
（4）DI/DO制御コマンド
（5）情報取得コマンド
（6）アラームコマンド

4.5.5 課題

必要に応じて最適な補間処理や加減速パターンを組込むことができるような機構と補間、加減速処理をモジュール化することが必要である。さらにATC動作に関わるアーム旋回・主軸の制御など機械固有の動作についてもサブシステム化し、機能的切り分けができるような構成をとる必要がある。

これら機能的分類の観点とは別に各種のサーボシステムやPLCシステムを共通のインタフェースで扱える環境の検証も必要と考える。さらにマシニング系以外の機械に適用することにより、各種機械への適応性を検証していかなければならない。

4.6 デバイス制御

デバイス制御とは、計算機内部の情報と外界との接続部分を司る部分である。本仕様では、デバイスとしてサーボおよびPLCを対象としている。

デバイス制御インタフェースは下記の方針で設計された。

（a）標準化レベル

デバイス制御機能ブロックの部品が豊富に市場に出回る状況を作り出すために、H/W部分ではなく、CまたはC++の関数レベルでのインタフェースを規定する。

（b）リアルタイム処理

パソコン上での処理ではリアルタイム性を保証できない。そのため、ハードリアルタイム処理部との橋渡しをする機能を組み込み可能とする。

（c）マルチベンダ対応

複数ベンダの供給する部品を単一の部品として利用する機構を設ける。

（d）モデル

移植性を考慮し、物理的な構成と論理的なモデルを分離する。

4.6.1 サーボ制御

（1）サーボデバイスの概要

サーボ処理は、アクチュエータを目標に追従させるための処理である。このためにはハードリアルタイム処理が必要であるが、パソコンではリアルタイム性の保証ができないため、リアルタイム処理の実現方法はサーボデバイスの提供する側に任せている。

（2）サーボデバイスモデル

サーボデバイス機構としてサーボ軸単体、サーボグループ、サーボ管理、の３階層に分かれているモデルを想定した。このモデルを図4-10に示す。

（3）サーボ制御API

サーボ制御APIは、サーボデバイスモデルの各部分の管理と、サイクリックなサーボ処理機能をサポートする。

(a)インタフェース管理機能：　デバイス制御部分との接続や初期化、構成情報管理
(b)グループ管理機能：　サーボグループ単位での初期化、構成情報の管理
(c)サーボ軸管理機能：　軸単位での初期化、構成情報の管理
(d)サーボ制御処理機能：　オンライン時サーボ制御用機能で、サーボ指令機能、フィードバック情報取得機能、同期処理機能、サーボ周りの諸機能を含む

（4）今後の課題

(a)マルチベンダインタフェース

複数ベンダ製品の同時使用を可能とする部分の仕様の検討を行う

(b)高機能指令

リアルタイムフィードバックの活用など、高度な処理を行うため、高機能指令を受け付けられるインタフェース仕様を検討する。

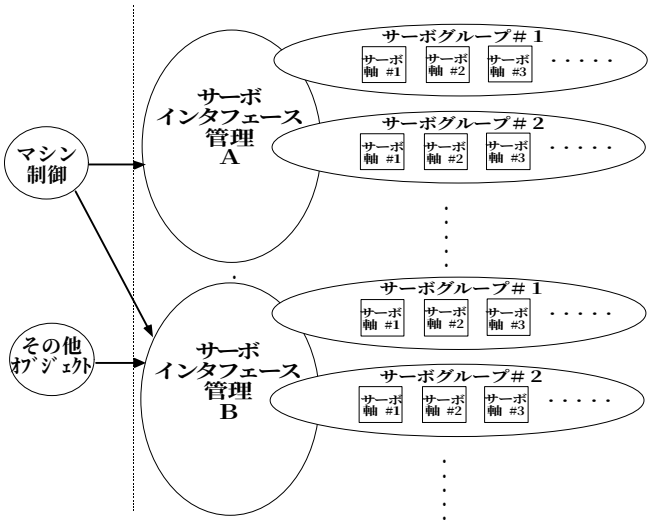

図 4-10　OSECサーボデバイスモデル

4.6.2　PLC入出力

（1）PLCデバイスの概要

PLCはシーケンスロジックを制御する方法として、従来より工作機械に使用されている。本仕様では、主に入出力機能の実現のために利用するPLCを対象に、CNCからアクセスする際のインタフェースを規定している。また、プログラムの互換性の観点から、入出力データを接点番号ではなくシンボル名で取扱っている。

（2）PLCデバイスモデル

PLCデバイスのモデルを図4-11に示す。上位プログラムから認識できるPLCデバイスと実際のPLCとは同一であるとは限らない。

（3）PLC入出力API

このAPIは上層の機能がPLCを使用するためのインタフェースである。

(a)機構管理：　デバイス制御部分との接続関連APIを用意している。

(b)パラメータ操作：　デバイス制御部分のパラメータの操作を行う。

(c) PLC状態チェック：　PLCの実行状況情報を得るための機能。

(d)データコントロール：　PLCとのデータの授受に関する機能で、データ授受機能、ID管理機能、同期処理機能を含む。

（4）今後の課題

- マルチベンダインタフェース：複数ベンダ製品の同時使用を可能とする部分の仕様の検討を継続して行う。
- データの識別子の規定：PLCで標準的に使用するデータに関する識別子（シンボル名およびID）の検討を継続する。
- PLCプログラム開発を含めた部分の規定：現行仕様では、PLCメーカ毎に提供される開発環境を使用する為に、作成されたアプリケーションは可搬性・移植性・再利用性に制限がある。そのため、開発・メンテナンス環境の仕様を検討する。

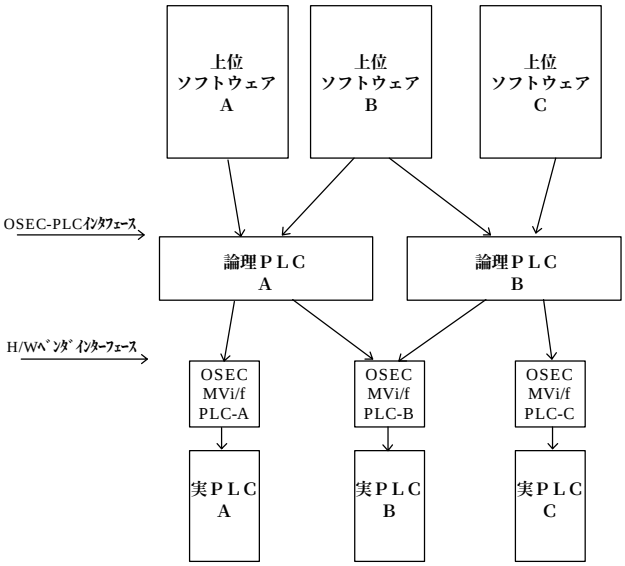

図 4-11　OSEC PLCデバイスのモデル

5 OSEL

5.1 目的

OSELは、加工に関するノウハウをソフトウェア部品として記述できる枠組みを、オープン化された形で提供する。加工に関するノウハウは、技術資料や技能のレベルで工作機械に携わる関連部署や関連する人たちによって生成され、保存されている。この加工ノウハウをオブジェクト指向の対象として捕らえ、クラスライブラリを構築することで、その加工ノウハウを蓄積できる枠組みに焦点を当てる。材質が変わり、加工方法が違い、加工条件も異なる場合は、加工用ライブラリを取り替えることによって、膨大になるライブラリ群をすべて手元に置く無駄を排除できる。他のライブラリは、ネットワークのなかのどこかに存在させ、必要ならばそれと入換える。

我々は、NC言語としての枠組みを作り、OSECアーキテクチャにおける機能モジュールに対してのI/Fを決めていくことに重点を置いている。すべてを網羅する加工用クラスライブラリをここでは提案しない。実用に供する加工用ライブラリを実用化し、提供するのは、興味を持つメーカの役目である。しかし、この枠組みとI/Fの具体性を示すために、デモバージョンを作成し、公開する。

5.2 背景

5.2.1 現在のNC言語・言語処理系の問題点

現在、工作機械を制御するために用いられているNCデータのフォーマットは、ISOでも規定されており工作機械を制御するには簡潔な表現方法である。しかし、この規格は20年以上も前に制定されたものであり、近年の目覚しい技術革新を遂げているコンピュータの性能を十分反映しているとは言い難い。さらに、現在のNCデータは入力データ量を少なくするためにマクロ化され、CNCメーカごとに特徴を出した複雑な機能が追加され、互換性が保証できなくなってきている。現在のＮＣデータに対する問題点を以下に列挙する。

- 表記文字を少なくなるように定義してあるＮＣデータは、メモリが高価だった時代にメモリを浪費しないように設計された体系であり、必ずしも表記が読みやすくない。
- NCデータには自由曲線や自由曲面を扱う記述が用意されていないので、自由曲線、自由曲面といった形状に沿った動作は、細かい点列に分解された動作となる。例えば、金型加工ではみがきを省くため機械加工でできるだけ曲面の面精度を上げたいが、NCの基本機能は工具の直線補間であり滑らかな面の加工は難しい。このような補間は加工精度の劣化とデータ量の増大につながる。
- NCデータは一般的に記述のレベルがまちまちである。複合されたマクロ的で複雑な動作記述もあれば直線運動など単純な動作記述もある。
- カスタムマクロはCNCメーカが独自に拡張を加えたものであり、それぞれ他のCNCメーカのものに対して互換性がない。また、工作機械の種類に依存した部分を含むため、同一のNCデータをそのまま他の機種に横展開できない。
- 工作機械の種類に対する依存性が大きく、また表現の柔軟性・拡張性に乏しいため、ライブラリ化が困難である。

このような問題点に加えて、ここへ来てオープン化への期待がFA制御装置の分野でも大きくなり、上位システムであるCAD/CAMシステムやデータベースシステム、その他各

種情報処理ツールとの統合性の容易さが求められている。これは種々のベンダが提供するパッケージソフトウェアやソフトウェア部品を利用し、それらを統合することによって目的にあったシステムを容易に構築することができることを意味する。

5.2.2 要求機能

以上のような観点から、工作機械を制御するデータに対し、次のものが求められる。

（1）表現に柔軟性・拡張性を持たせるために

- 工作機械の動作記述としてわかりやすいスクリプト記述型の言語体系であること。
- 自由曲線、自由曲面の数式による直接指定できること。Bezierなど曲線･曲面関数があらかじめ定義されることで、今までのNCデータの点列による指定よりも加工の品質が向上し、データ量も少なくできる。
- 拡張性を持たせるためのマクロや関数のライブラリ化ができること。

（2）工作機械の抽象度・一般性を高め、機種依存性を低減させ、可搬性を向上させるために

- 精度や動作速度など工作機械の仕様、ツールハンドリングなどの工具、ワークオフセットなどの治具、工作機械に組込まれたDI/DO信号割当という工作機械依存の情報を動作記述レベルでは抽象化して扱うことができること。
- 計算機ハードウェアやオペレーティングシステムに依存する部分を解消するため、ASC-II表記による記述であること。
- 工作機械の機種に対する依存性を少なくするため、動作記述レベルとしてはプリミティブな処理単位に揃えてある。

（3）既存のデータとの整合性を保証するために

- GコードやCLデータなどで記述された既存のデータを継承するため、NCデータのデコーダを用意し既存のデータとの整合性を保証すること。

5.2.3 経緯

OSEC-Iでは、動作生成部をFADL(Factory Automation equipment Description Language)の名称で、シーケンス制御も含めた加工プロセスを記述したプログラムを解釈して、制御命令に落とすという位置付け(図5-1)で、捕らえていた。

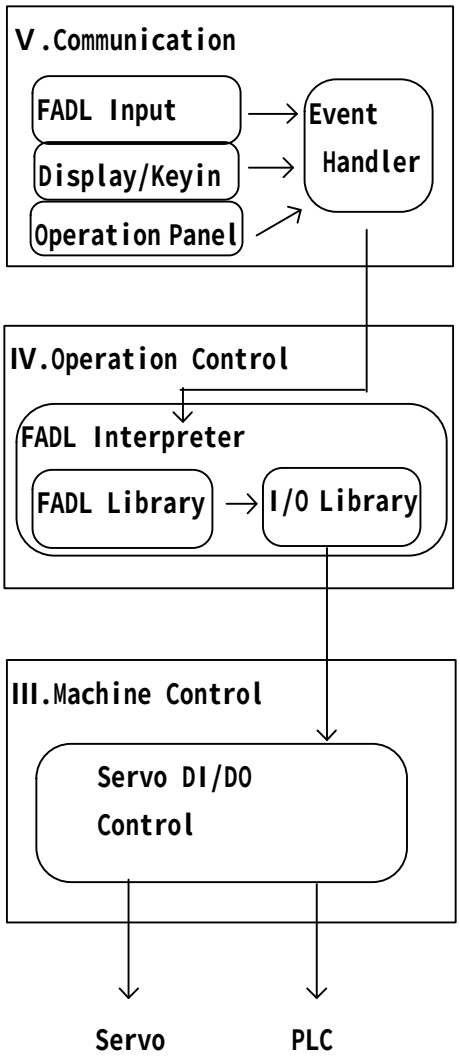

図 5-1　OSEC-ⅠにおけるFADLの位置付け

OSEC-Ⅱでは、名称をOSEL(OSE Language)と改名し、位置付けを変えた。先回のFADLはＮＣ全体を制御している印象を与えている。しかしOSEC-Ⅱでは、NCのリソース全体を監視・制御する“リソース管理”を新たに設けた。従って、OSELは、加工の特徴を記述でき、その記述に基づいての工具軌跡と加工条件を生成する役割に専念する位置付けとした。

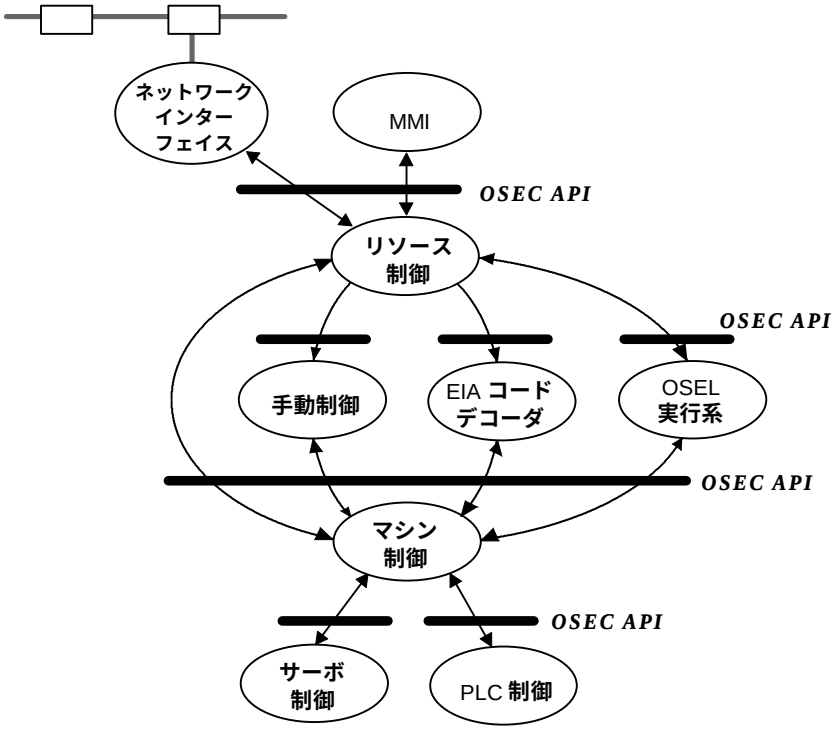

図 5-2　OSEC-ⅡにおけるOSELの位置付け

5.2.4　OSELの範囲

OSEC-Ⅱで検討したOSELでは、加工形状と加工材質、加工順序は定まっているとする。この情報から加工に係る工具、工具経路と加工条件を、機能モジュールの一つであるMC制御に出力するまでを扱う（図5-3）。この処理系は、処理系全体として工作機械から切り離すことができ、必要とあらば交換でき、この処理系の中にある加工クラスライブラリも、対応する加工に合わせて交換できるように考えている。このことによって、H/Wの性能向上や工具の進歩、加工材質の変化に合わせたOSEL処理系を構成できる。

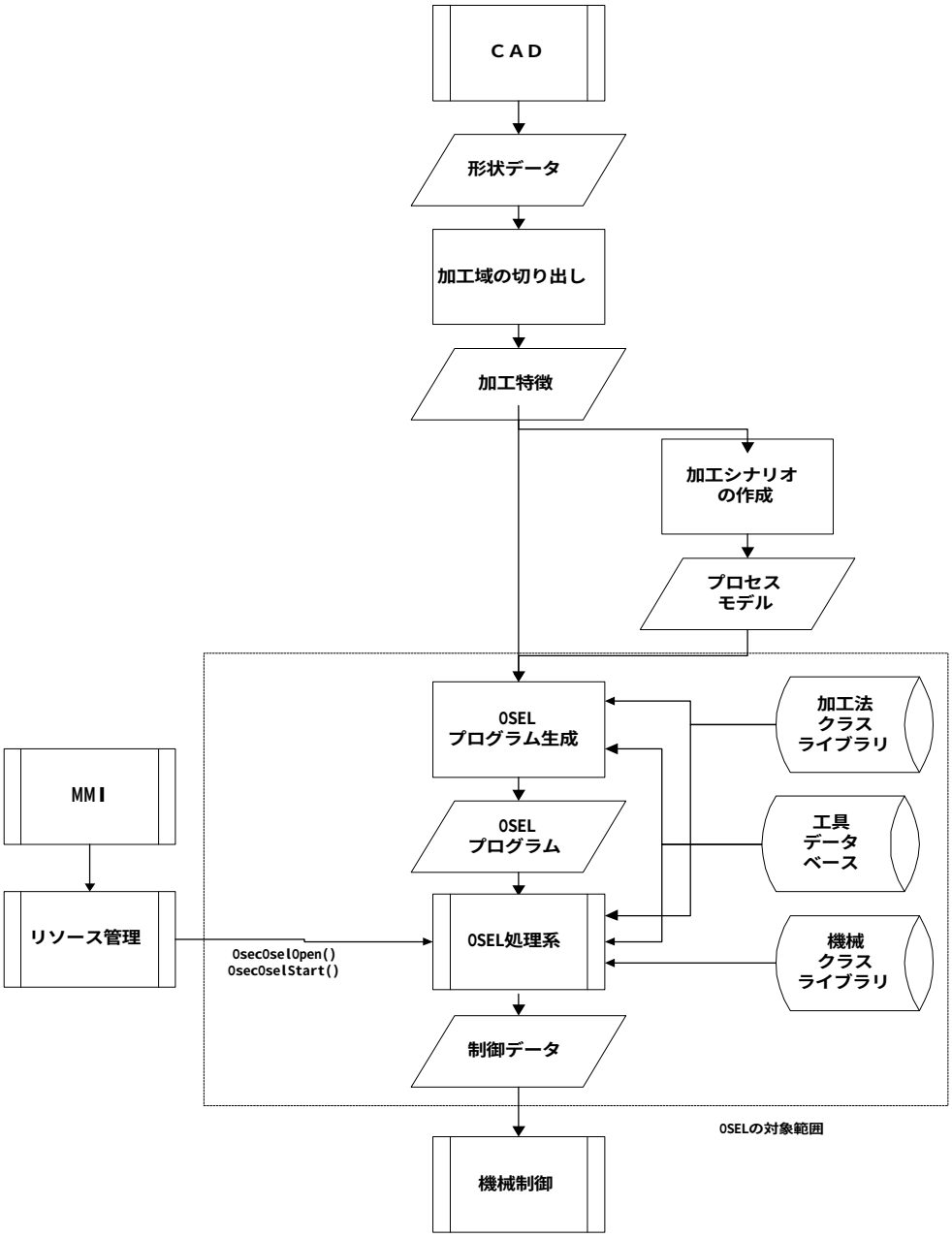

図 5-3　OSELが取り扱う範囲

5.3　コンセプト

NCのためのコンピュータ言語は、人が加工の意図を表現するための記述言語と、加工機械を制御するために制御システムが読みとる制御言語が最低限必要である。

NC記述言語は、加工対象や加工方法を特定することが最優先される言語である。記述に当たっては、加工対象や方法を指定はするが、それが実際にどこのどの機械で加工するかという事は意識したくない。飽くまでもある形をある精度で完成させたいという事を考えて記述したい。作られた記述は、出来る限り多様な加工環境で実行できるように展開してゆけることが望ましい。従って、加工対象や加工方法は記述するが、その順序については細かくは規定しないのが望ましい。

一方、制御言語は、それを直接実行して特定の機械を動作させるのが目的であるので、その機械に対して最適化された動作順序が組み立てられており、また、その機械に固有の工具、軸、工具経路補間、加工速度、補機の制御、などが、その機械のために最適化された順序で、その機械が受取って実行出来る言語で記述できる必要がある。

OSELでは、記述言語をOSEL-F(Feature)、実行言語をOSEL-X (eXecutable) と読んでいる。通常はこの２つが決まっていれば言語処理システムを作ることが出来るが、その場合は作られる言語処理システムが固定化し易く、同じ記述言語プログラムから異なる制御言語プログラムを生成するといった柔軟性を保証することが難しくなる。そのため、これらの組み合わせの柔軟性を保証する仕組みとして中間的なジョイント言語を想定してOSEL-Iと名付けた。OSEL-Iは純粋に言語処理系のために規定される言語であり、必ずしも外から見える必要はないが、OSELの柔軟性と共通性を保証するメカニズムとしては非常に重要な役割を担うはずである。

これらの言語の処理系は、加工機械、加工方法、加工工具、などのバリエーションや進歩に対応できるようにコンポーネントを組み合わせることで実現出来ることが望ましい。OSEC-IIでは、その実現手法として、オブジェクト指向言語の技術の妥当性を検証することとした。

OSELにおける処理の流れの概略を図5-4に示す。

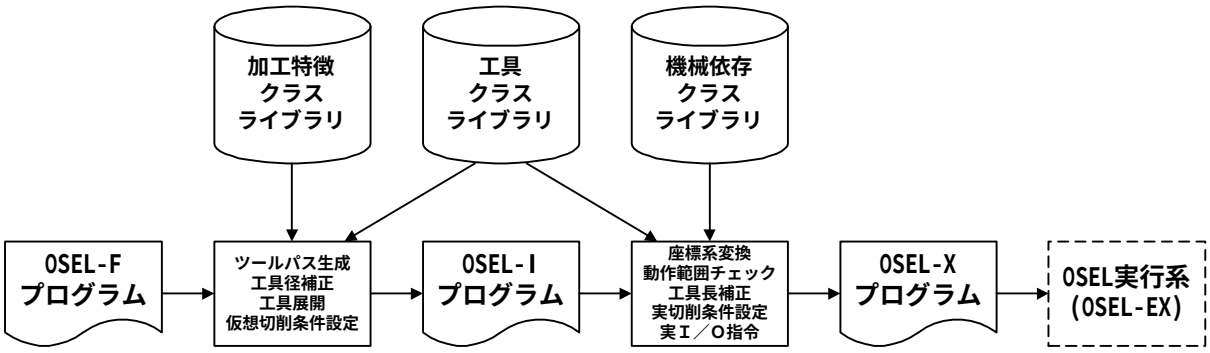

図 5-4　OSELにおける処理の流れ

最上位レベルでの記述であるOSEL-Fでは、加工プログラムは形状特徴に対応した加工特徴の記述、例えば「貫通穴をドリルで開けよ」といった形で入力される。OSEL-Fは加工特徴クラスライブラリと工具クラスライブラリを利用して、機械独立な中間記述であるOSEL-Iに展開される。OSEL-Iは機械依存クラスライブラリを利用して、機械に依存した実行フォーマットであるOSEL-Xに展開される。実時間で処理する必要のあるのは、このOSEL-Xプログラムの解釈処理部（OSEL実行系：OSEL-EX）以降である。

5.4　OSEL-F

本章ではOSELのトップレベルの記述であるOSEL-Fについて説明する。プログラミング言語としての枠組みはJavaの文法に従うので、OSELに固有の部分のみを扱う。

OSEL-F加工プログラムでは以下の記述が行われる。

- 加工対象となる加工特徴の定義
- 個々の加工特徴に対して行われる加工法の指定
- 加工工程（加工の順序）の指定

加工特徴の定義とは、「貫通穴」・「溝」といった加工特徴オブジェクトを定義することである。加工特徴オブジェクトは後述の加工特徴クラスライブラリで定義される加工特徴クラス（「貫通穴クラス」・「溝クラス」など）に対して、その加工特徴のワークにおける位置と幾何形状（例えば「貫通穴クラス」における「直径」・「深さ」など）、交差情報を加工特徴変数（フィールド）として与えることで、オブジェクトとして実体化（インスタンス化）する。形式的にJavaの記法に従えば、概念的には次のような記述になる。

加工特徴クラス 加工特徴オブジェクト ＝ new 加工特徴クラス（加工特徴変数）；

加工法の指定とは、定義された加工特徴オブジェクトに対して、加工法メソッドを呼び出すことで行われる。加工法メソッドはそれぞれの加工特徴クラスにおいてクラスメソッドとして定義されている。概念的には次のような記述になる。

加工特徴オブジェクト．加工法メソッド（加工のための変数, …）；

このような加工法メソッドは加工の順序に従って呼び出される。

5.4.1 OSEL-F加工プログラムの例

穴明け加工を行う加工プログラムOSEL-Fを図5-5に示す。この例では理解を容易にするため一部簡略化してある。

```
public class Sample {                                          // サンプルプログラム
    public static void main(String args[]) {
        //
        // 本来、この部分で位置・直径・深さの定義が行われる
        //

        ThroughHole hole = new ThroughHole(pos, dia, depth);          // 加工特徴『貫通穴』の定義
        hole.centerdrilling();      // センタドリル加工
        hole.drilling();            // ドリリング加工
    }
}
```

図 5-5 OSEL-F加工プログラムのサンプルプログラミング例

5.5 加工特徴クラスライブラリ

本章では、OSELが規定しようとしている加工特徴ベースクラスライブラリおよび形状ベースクラスライブラリについて説明する。ユーザはJAVAの文法にしたがって、自分で導出クラスおよびメソッドを定義することができる。

加工特徴オブジェクトは、加工される幾何形状と公差情報を持ち、その加工方法をメソッドとして定義している。形状表現と幾何計算のために形状オブジェクト、３次元座標と座標・ベクトル計算のためにベクトルオブジェクトがある。公差オブジェクトについては、現段階では十分な議論が行われていない。

加工特徴オブジェクトを生成する段階で幾何形状が定義される。加工メソッドが呼ばれると、まず形状オブジェクトが加工に必要なカッターパスを生成し、加工特徴オブジェクトがそのカッターパスにアプローチ・退避運動を加えた工具先端中心の動作を生成し、さらに工具オブジェクトを参照しつつ加工条件を設定し、実際に軸制御とDI/DOを行なう工

作機械オブジェクトのメソッドを呼び出す。オブジェクト間の呼び出し関係を図5-6に示す。

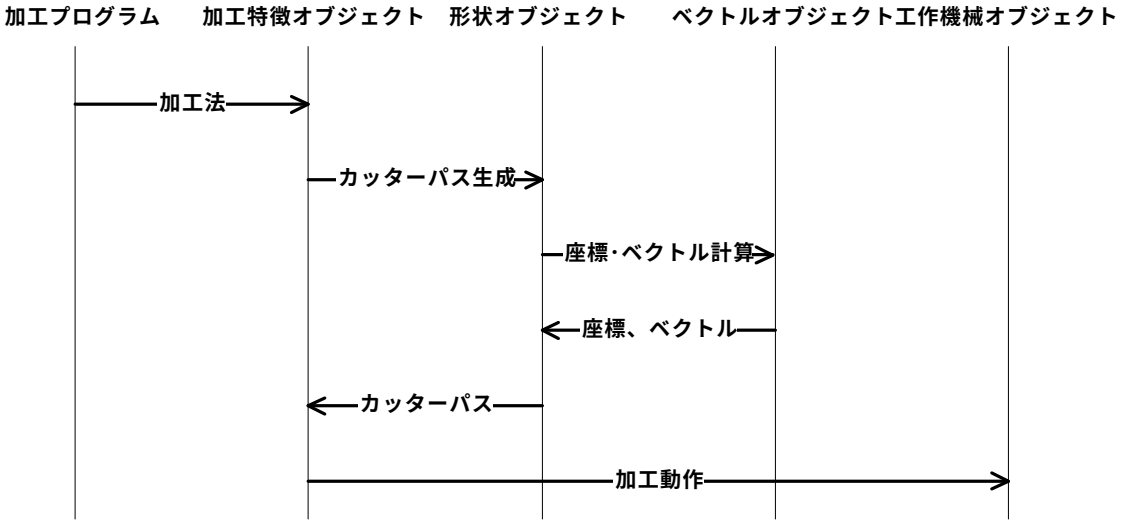

図 5-6　オブジェクト間の呼び出し関係

5.5.1　加工特徴クラス概要

加工特徴クラスは大きく穴、側面、面の3つのクラスに分類される。クラスの属性として加工特徴の形状オブジェクトと公差、メソッドとして加工法を持つ。全体構成を図5-7に示す。

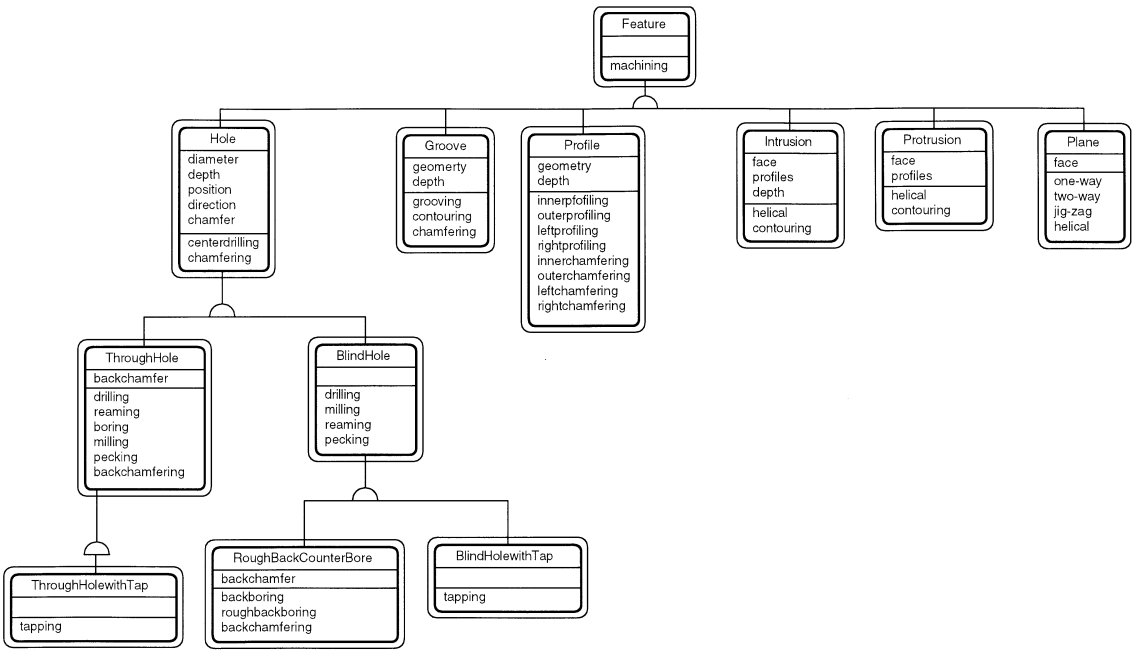

図 5-7　加工特徴クラス階層図

5.5.2 形状関連クラスライブラリ

形状表現と幾何計算のためのクラスである。属性として形状表現のためのパラメータ、メソッドとして工具の軌跡計算を持つ。全体構成を図5-8に示す。

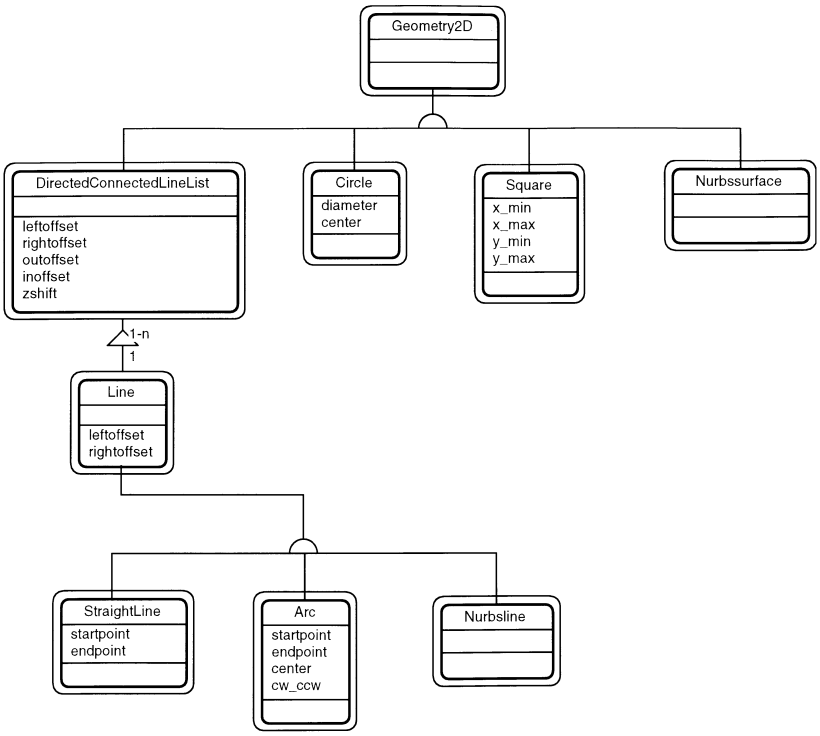

図 5-8 形状表現クラス階層図

5.6 工具クラスライブラリ

工具クラスライブラリは､加工特徴クラスライブラリや機械依存クラスライブラリから参照される。機械依存クラスライブラリは、ATCの工具ステーションに実際に装備されている工具のクラス名を得るために参照される。また、加工特徴クラスライブラリは、機械依存クラスライブラリに登録された工具の中から加工する加工特徴にあった工具を選択し、その工具の寸法や加工条件を工具クラスライブラリより参照する。

工具クラスライブラリが提供するデータは、工具の径や長さ、歯数などの寸法に関するものの他に、被加工物に応じた加工速度や、１パスの切込量などで、いわば詳細な工具カタログである。工具クラスライブラリが提供するデータと一般の工具カタログとの違いは、工具寸法が実際に機上で使用するのと同じ状態、つまり工具ホルダーやチャックなどに装着された状態の寸法である事である。また、加工条件等も実際の作業経験により順次変更が可能で、作業者の経験に基づく加工条件を登録する事も可能である。

工具クラスライブラリは、新しい工具を購入した際にも順次拡張が可能である。また、工具クラスライブラリ本体はオブジェクト指向言語で記述してあるので、例えば最新の切削理論によって最適な送り速度を自動的に計算するなど、新たな機能を持つ工具クラスを作成することも可能である｡その場合でも、既存の工具クラスの中で一番近いものを拡張する事により､新たに付け加えた機能のみをプログラムするだけですむ事となる。

しかし、新しい工具クラスを使用するためには､その機能を活かすことの出来る新たな加工特徴クラスを必要とする場合もある。そのような時は、既存の加工特徴クラスを基に

新たな加工特徴クラスを作成する必要がある。

現時点では、工具クラスライブラリが保持している寸法値は、完全な工具の詳細寸法ではなく、刃部、シャンク部、ホルダー部と大きく３つの部分に分けた寸法値である。そのため完全な工具干渉を計算する事は出来ないが、刃部以外の径寸法は干渉計算に配慮して､各部分の最大値を登録しているため、簡単な干渉チェックに利用する事が可能である。また、工具寸法だけでなく、その工具の３次元データを記載したファイル名を登録出来るようにすることにより、より正確な工具干渉チェックに利用することも可能である。

工具クラスライブラリは工具摩耗補正値の管理も行うが、工具摩耗補正値は他のクラスライブラリからの参照、変更が可能なため、実加工時間による工具摩耗管理や、工具交換時期の予測などの応用が考えられる。

5.7 機械依存加工クラスライブラリ

機械クラスライブラリは、次の機能を有する。

（１）OSEL-Xコマンドの出力メソッド
- ■補助機能コマンドの呼び出し
- ■軸移動コマンドの呼び出し

（２）機械依存メソッド<機械の固有情報による機械制限チェック機能>
- ■ストロークリミットチェック機能
- ■速度指令上限クランプ機能
- ■主軸回転数上限クランプ機能
- ■工具磨耗補正機能

加工特徴クラスライブラリから呼び出された工作機械オブジェクトは、与えられたワーク座標系による工具先端の座標を機械座標系における座標へ変換し、工具長の補正を行なった後、動作範囲と加工条件の制限範囲のチェックを行ない、軸移動コマンドやDI/DOを行なう。加工特徴クラスライブラリで機械に独立した形で記述されている加工法が、機械依存加工クラスライブラリによって、個々の工作機械を用いた場合に最適な加工法へ変換される。

5.8 サンプルコード

```
public class ThroughHole extends Hole {          //加工特徴『貫通穴』クラスの定義
  Vect3 position;                                //属性：位置
  double diameter;                               //属性：直径
  double depth;                                  //属性：深さ
  public ThroughHole(Vect3 pos, double dia, double d) {    //『貫通穴』のコンストラクタ
  position = pos;
  diameter = dia;
  depth = d;
  }

  public void drilling() {                       //加工法『ドリリング』メソッドの定義
  Tool tool = toolfordrilling(diameter);                 //穴明け用工具選定
  machining_center.toolchange(tool);             //工具交換
  Vect3 initial = getinitial();                          //逃げ点
  Vect3 reference = getreference();              //加工開始点
  Vect3 target = gettarget();                    //貫通穴の底の位置
  machining_center.toolmove(initial);            //逃げ点へ移動
```

```
machining_center.toolmove(reference);        //加工開始点へ移動
machining_center.spindle_on(Osel.CW);        //スピンドル・正転
machining_center.coolant_on();               //クーラント・オン
machining_center.toolline(target);           //貫通穴の底へ直線移動
machining_center.toolmove(reference);        //加工開始点へ戻る
machining_center.spindle_off();              //スピンドル・停止
machining_center.coolant_off();              //クーラント・オフ
}
}
```

5.9 他の標準化活動

ここでは具体的な標準化活動として、ESPRITプロジェクト内のOPTIMALと、ISOの審議組織の１つであるTC184/SC1/WG7について概要を紹介する。

（1）OPTIMAL

OPTIMALは、 ESPRITプロジェクト8643、Optimized Preparation of Manufacturing Information with Multi-Level CAM-CNC Couplingの略で、6企業、1研究機関の参加で活動が行われている。製品の幾何形状から制御装置までのつながりを維持し、モジュール構造によるCAD-CAM-CNC間の一貫した流れの実現を目標としている。具体的にはモジュラ設計のプログラミングシステムの開発と、CNCとの間の多階層のデータ交換インタフェースの実現を計画している(図5-9)。

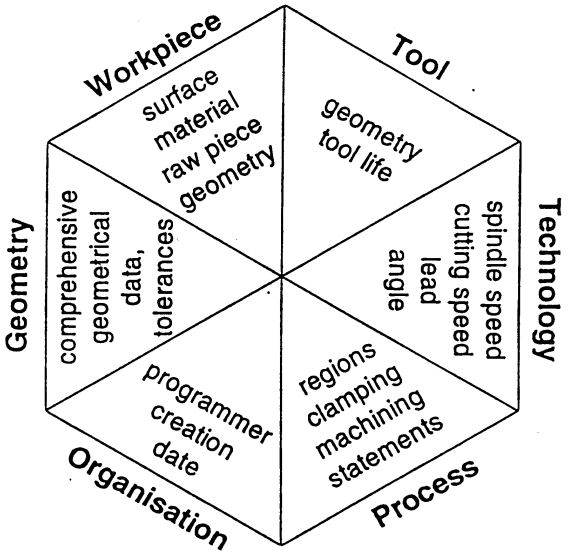

図 5-9 OPTIMAL概念図

（出典：Deliverable 1.2 Definition of the Interface and the Interface Levels）

（2） TC184/SC1/WG7

ISO/TC184/SC1では1994年11月より新たにWG7（Data modeling for integration of physical devices）が組織され、主にESPRITの研究成果をもとに、制御装置で用いるデータ構造のモデル作成とデータ利用に関する標準化の作業（当面の作業タイトルは“ Data model for CNC controllers"）が進められている。規格化の狙いとしては、

■ マルチベンダー化

- CADからNCプログラムまでを統一された手法で取り扱う
- 新しくオープンなNC言語を作成する

などを挙げ、まず

- CNCを対象とする
- フライス盤を想定する
- 2～2.5次元の簡単な形状

の範囲で、STEPなどの既存の国際標準規格により記述された製品データや工具データなどを入力として、EXPRESS-Gで記述した幾何形状、加工方法、加工工具、制御プログラムの4種のデータセットを独立させることによる制御装置の参照モデル（図5-10）を検討している。最終的な出力は数値制御プログラムであり、これは工具データセット、技術データセットを参照し、技術データセットはさらに形状データセットを参照する。形状データセットはSTEPファイル等から直接データを収集する構成である。プログラムデータセットを除いた各データセットは、EXPRESSで表現したデータ群である。現在は、まず静的なモデルを確定することに専念しており、現実の世界に必ずある外乱（例えば加工形状の設計変更や工具・加工方法・使用機械の変更など）があった時に、加工プログラムを再生成するといったリアルタイムな処理が必要なモデルは第2ステップの課題となっている。

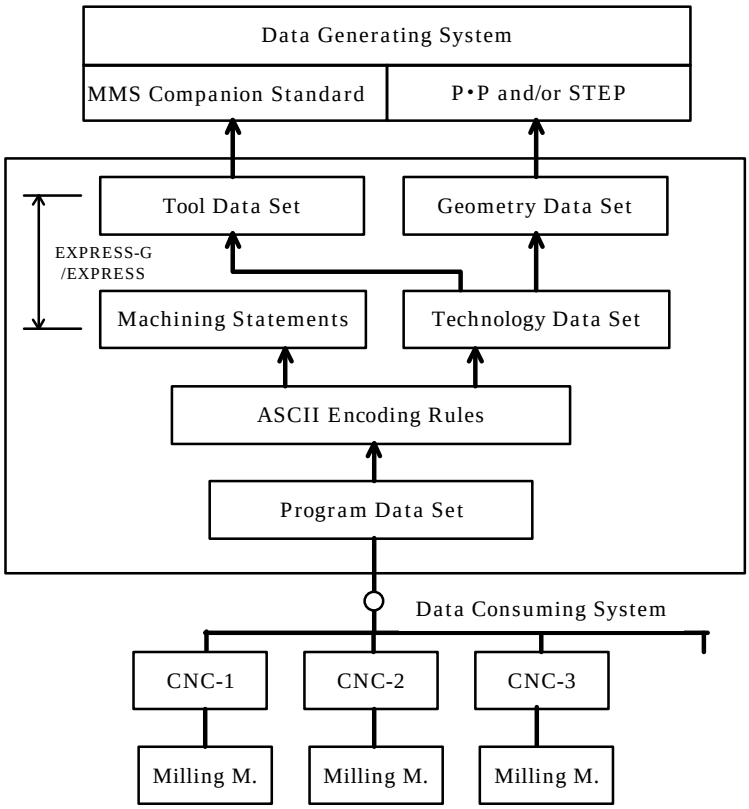

図 5-10　WG7の作業概念図

（出典：日本機械学会[No.96-50]講習会教材
['96-7.24,東京,NCオープン化への取り組みと期待] p-10より）

5.10　今後の課題

OSEC-Ⅱでは主に3軸マシニングセンタを考慮し、穴・軸・平面の加工を処理の対象と

した。これは、いきなり困難な加工を対象にするよりも、現段階では、OSELのコンセプトが正しいことを簡単な例でもいいから、まず実証することが重要であると考えたからである。

この過程で、何を基準に加工特徴クラスの階層構造を定めるか、議論が別れた。一つの立場は、現在現場で用いられている分類を最優先にして、クラスライブラリ化する立場であり、もう一つは、情報処理の立場から可能な限り拡張性に富むクラス構造を取りたいという立場である。例えば、“穴”という形状に対し、単純な穴加工ですむ場合、プロファイル加工する場合、ポケット加工する場合が存在する。加工特徴として加工法が密接に関連すると考える現場を優先するのであれば3種類の加工特徴を用意することになる。情報処理の立場からは、凹形状のサブクラスとして断面が円になるものを定義し、メソッドを３種類用意することになるだろう。これはまず加工形状で分類し、加工法によりメソッドを定義することによって、後のメンテナンスを容易にするという考えに基づく。現在のOSELでは前者の立場が強いが、後者とのメリット・デメリットをまだ十分に考察していない。また両者を共存させる方法を今後考案する必要がある。

残された課題として、

- ポケット加工、ボス加工への対応
- 公差の取り扱い

がある。さらに

- 多軸マシニングセンタを対象とした自由曲面の加工
- 旋盤等の他の工作機械への対応
- 加工法ライブラリを用いて、ユーザとの対話による最適な加工プログラムの生成を支援するCAMの構成法
- ネットワークを用いた加工ソフトウエアの流通、調達法

などを今後の課題として考えている。

（参考文献）

- 精密工学会主催　第２１６回講習会「いよいよ見えてきた工作機械制御装置のオープン化アーキテクチャ」資料
- '95工作機械関連技術者会議（日本能率協会1995.7）資料
- 日本機械学会[No.96-50]講習会教材［'96-7.24、東京、NCオープン化への取り組みと期待］
- Deliverable 1.2 Definition of the Interface and the Interface Levels　11.April. 1995)

6　実証システム

6.1　実証システムの概要

このプロジェクトの目的は、あくまでも現時点の技術によって具体的に実現可能なシステムアーキテクチャを作り上げることである。その意味で、これまでに述べたシステムアーキテクチャとインタフェースを具体的に実装し、技術の実用性と仕様の抜けを確認することは非常に重要である。またCNCにPCを使用することで必然的に可能になるべき他のシステムとの組み合わせによる柔軟な運用が実際に可能かどうかも、具体的なシステム構築の過程で確かめることができる。このような目的のために、本プロジェクトでは実際に以下に述べるようなOSEC実証システムを設営した。今後この実証システム構築の経験から、より具体的なオープン化への指針が導き出されるはずである。

実証システムは、以下の各ステーションをローカルネットワークで結合し、さらにそれをインターネットによってワイドネットに開放していくことを行った（図6-1）。

- ステーションA……OSECアーキテクチャに則ったモジュール構成のMC
- ステーションB……従来のMC構成に近いものにOSECモジュールを追加
- ステーションC……ソフトウェアによるOSECコントローラ実装マシン
- OSELステーション……CADからOSEL処理系への連携を実験する
- ネットワークステーション…OSECコントローラをネットワーク展開

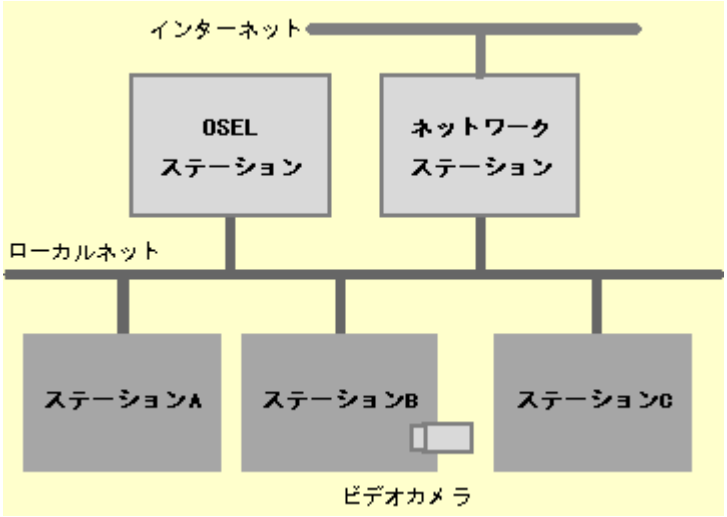

図 6-1　実証システムのネットワーク構成

各コントローラの実装を、図6-2に示す。図では、左部分の実装モデルと実証システムの構成を対比することにより、各ステーションの特徴を明らかしている。

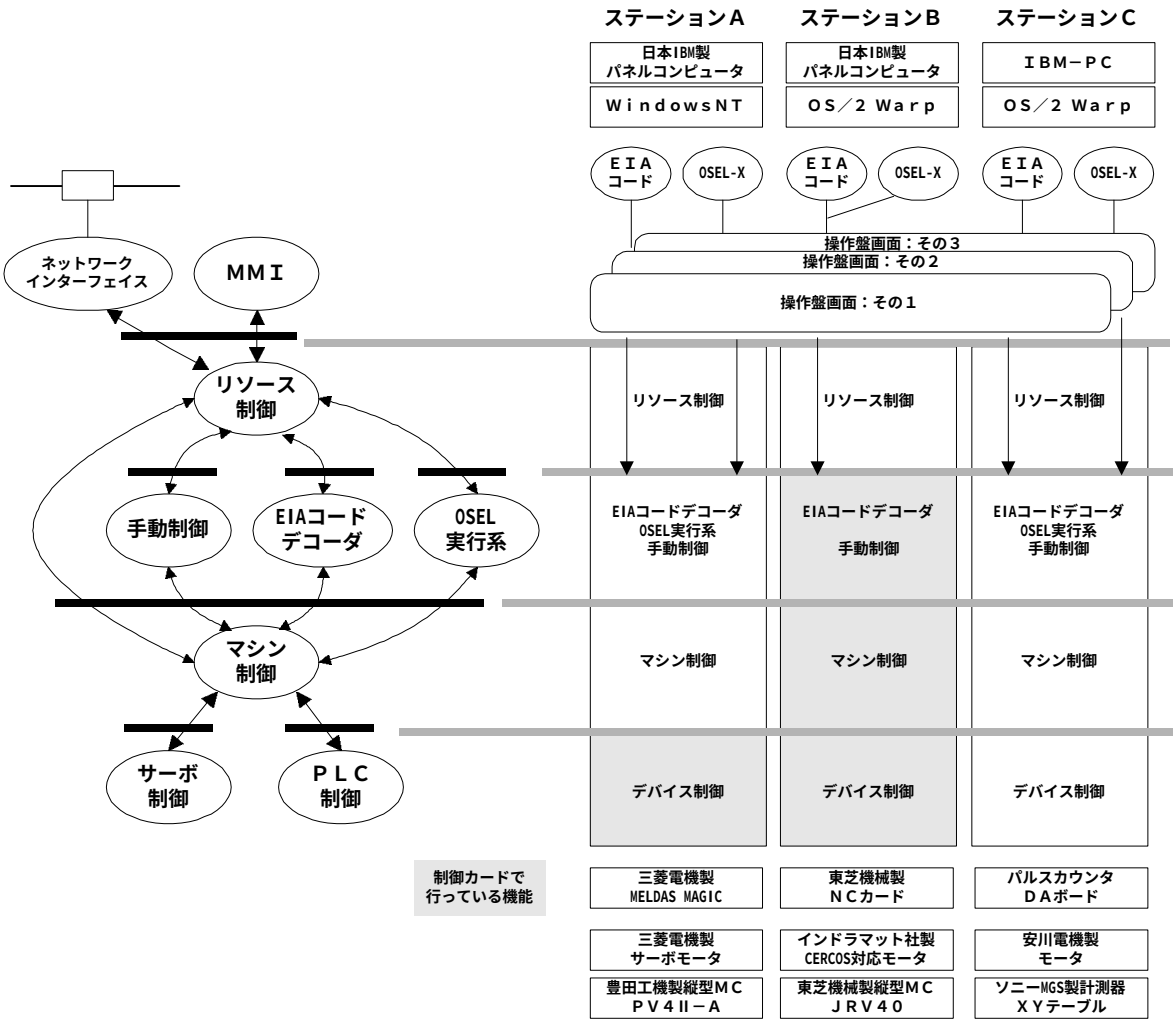

図 6-2　OSEC実装モデルと各ステーションの構成

各ステーションで実証できたことを列挙する。

- NC機能をPCのソフトウェアで実現し、加工運転を行ったことにより、ソフトウェアNCの実現性
- OSECの仕様で操作画面用ソフトウェア、サーボデバイスなどの互換性と部品化の実現性
- 用途によってはOS、ボードなどを汎用品、市販品だけで実時間制御も含めたシステム構築の可能性
- ユーザの持つ加工ノウハウを記述できること及びネットワーク上での流通性の向上
  特別なネットワーク技術なしで、安価なインターネット技術を応用した生産システムのネットワーク化とネットワーク応用の可能性

6.2 ステーションA

ステーションAの狙いは次の2つである。

・PCベースのコントローラを使用して工作機械を構築し、切削加工を行う。

・そのコントローラを実現するにあたってPCの資源を最大限活用する。

従ってコントローラの機能の多くはPC上のソフトウェア部品として実装され、必要とするハードウェアの機能は極めてシンプルなものになる。これにより、オープンで、モジュラでスケーラブルなコントローラを実現することが可能になる。

（1）機能モジュールの実装

コントローラの各機能モジュールはOSE研究会の成果および市販品として市場に出回っている製品を使うこととした。この構成によって機械メーカ、ユーザ、研究者がコントローラにオリジナルな機能を組み込むことが可能になる。

表 6-1 ハードウェア構成（ステーションA）

パソコン本体	IBM-7344	Pentium (100MHz), HDD : 500MB, RAM : 32MB, タッチパネル
サーボ機器	MELDAS MAGIC（三菱電機）	
	送り軸 (X,Y,Z)	駆動アンプ: MDS-A-V type（三菱電機） モータ: HA type（三菱電機）
	主軸 (SP)	駆動アンプ:MDS-A-SP type（三菱電機） モータ :ビルトインモータ（三菱電機）
II/Oユニット	リモートI/O: FCUA-DX（三菱電機）	

図 6-3 ステーションA全景

（2）まとめ

この実証機は先に述べた２つのねらいを十分に達成できた。マシニングの切削加工として、 ATC（工具交換）や主軸制御を行ったことに加え、フライス・ミーリング・ドリル・タップの基本加工を実行した。これらはOSELで記述された加工プログラムにより行われた。またコントローラの機能の多くをソフトウェアで実行できることも確認された。

いくつかの課題も残されている。市販品の組合せで実時間制御を確実に行うための方法の確立が必要である。またNCとPLCでは大量の情報の授受を高速に行う必要があるが、この要求を満たす汎用的な方法がないことなどが挙げられる。従って、NCとPLCが連動して行われる機能（工具選択、工具交換、インタロック処理）は性能を落として行うことになった。

6.3 ステーションB

ステーションBでは、東芝機械製立形マシニングセンタ（JRV40）にＩＢM製のパネルコンピュータ（IBM7344, OS/2）を取り付け、パソコンNCを実現している（図6-4）。
ステーションBの狙いは以下のようになる。

・マシンフレームワークによるヒューマン・マシンインタフェースのオープン化
・IEC規格（SERCOS）に準拠した市販サーボシステムとの接続
・OSEL-EIA変換方式によるOSELプログラムとの接続
・ネットワーク接続によるNCデータのリアルタイムモニタリング

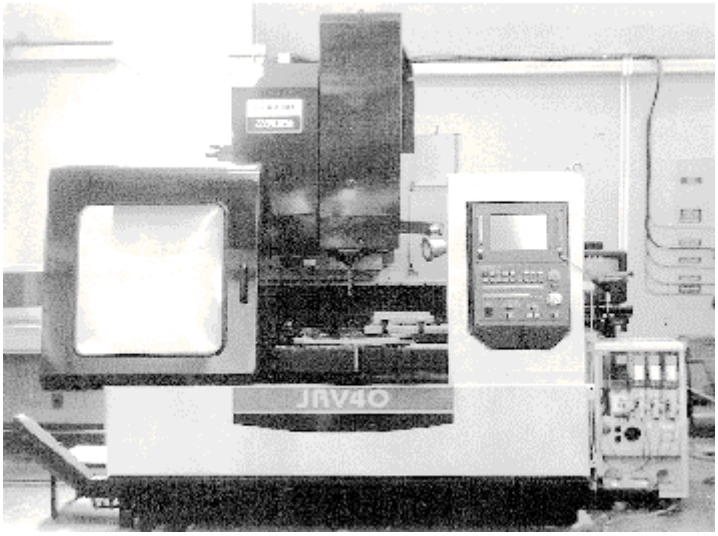

図 6-4　ステーションB全景

（1）実装

NCカードにはSERCOSに準拠したインタフェースボードが実装されていて、光ファイバケーブルにより市販のサーボアンプとモータ（インドラマット社製）を制御している。パネルコンピュータとNCカードはリソースマネージャを介して、マシンツールフレームワークモデルによるデータ交換を行っている。

表 6-2　ハードウェア構成（ステーションB）

パソコン本体	Pentium 100MHz HDD　500MB、　RAM　24MB タッチパネル分解能（800x600）
サーボドライブシステム	IEC1491規格（SERCOS）準拠 通信：　光ファイバ・リング　2MBps サーボアンプ・モータ：インドラマット製
NCカード	東芝機械製NCカード（仮称TOSNUC-P80M） ISAバス接続、PLCコントローラ内蔵

（2）まとめ

複数のメンバ企業がマシンフレームワークにより、別個に独立に作製した独自の操作画面をソフトウェア部品として組み込み、選択して実行可能なことが確認できた。また、ネットワークのサーバへデータを送出しながら、市販サーボを用いて基本的なミーリング加工

をATCなどの動作を含めて実行できることを実証できた。

6.4 ステーションC

本システムは、工作機械以外への応用事例として、ソニー・プレシジョン・テクノロジー社製非接触測定機をOSEC-IIアーキテクチャに基づくオープンコントローラで制御している。 本システムは、５つの機能群を標準インタフェースに準拠したパソコン上のソフトウェアで実現した、全ソフトウェアCNCで制御されている。

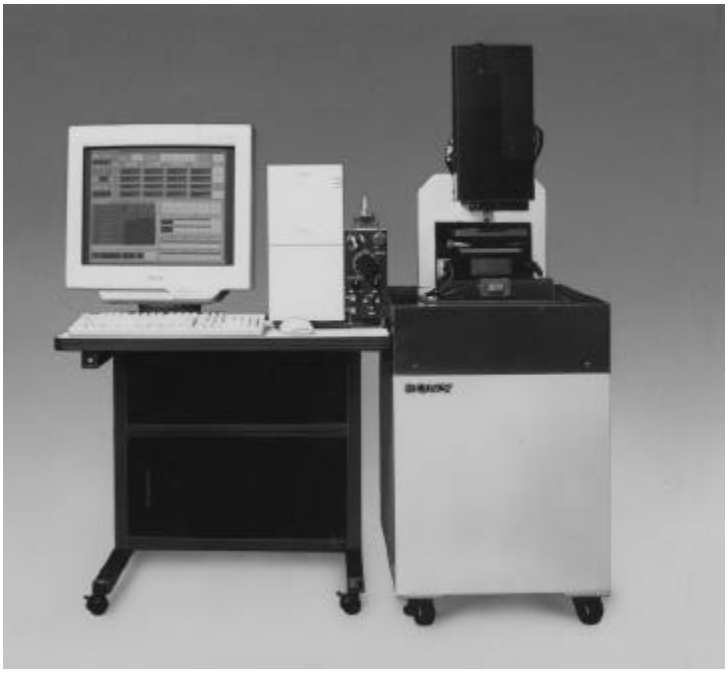

図 6-5 ステーションC全景

（1）実装

センサおよび電気デバイスとのインタフェースに、パソコンに実装した汎用カウンタボード、D/AボードおよびDI/DOボードを使用し、8msで起動されるOS上のプロセスで位置ループに関する演算処理を行う。 また、PLCはIEC-1131に準拠したソフトウェアPLC(ISaGRAF)を実装している。

本システムは、OSにOS/2 Warpを採用し、OSEC-IIアーキテクチャに基づく5つの機能群をソフトウェアで実現した実装モデルとして設計された。

表 6-3 ハードウエア構成（ステーションC）

パソコン本体	Pentium 133MHz RAM 32MB HDD 1.6GB
サーボドライブ	速度制御型 アナログ入力
汎用PCボード	4ch A/B相 Counting board 4ch 16bit D/A board 32 input 32 output Di/Do board

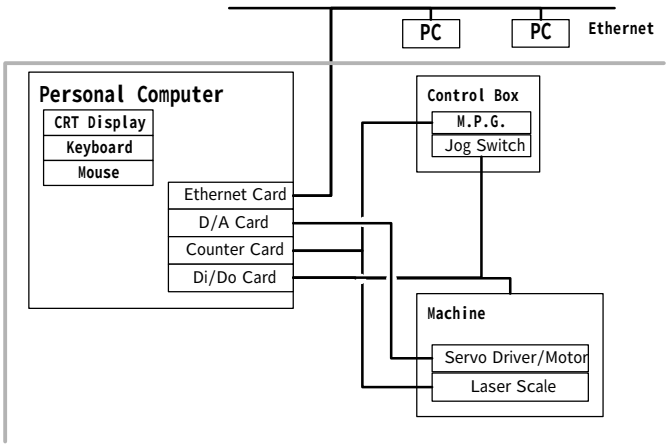

図 6-6　ハードウェア接続ブロック図（ステーションC）

（2）まとめ

OSEC-IIで定義した、5つの機能群と標準インタフェースに基づく実装モデルの設計、試作を通して、コントローラのオープン化による「目的に応じた手段の選択肢を広げる」効果が実証できたといえる。

汎用ボードおよび汎用サーボドライバを用いてOS/2 Warp上に8msecサンプリングのフルクローズド位置ループサーボを構築できたことで、オープン化の実証デモとして可能性を示した。

6.5 OSELステーション

OSELステーションにおける実証デモの目的は、

■ OSELを用いた場合の全体の情報の流れをユーザに理解してもらうこと。
■ OSELの意義を具体的に示すこと。特に加工ノウハウをプログラムに簡単にユーザが反映できること。

を示すことである。

OSEL加工プログラムに含まれる重要な情報として、カッターパス・加工条件・工具・シーケンス情報がある。ユーザにわかりやすくこれらを提示するため、GUIとプロセスシートを用意した。GUIとしてCAD（CADKEY)、プロセスシートとして表計算プログラム（EXCEL）を用いた。CADKEYにはワーク形状、カッターパスを表示し、表には加工条件、工具、シーケンスを表示する。EXCELは工程最適化のツールとしても用いる。

デモのシナリオを次に示す。

(1)CADKEYを用いてワークの図面を作成する。
(2)CADKEY上から加工特徴を順次ユーザが指示していく。
(3)このデータをもとに、OSEL-F加工プログラムを生成する。この段階で、プログラム中には、加工特徴のインスタンス生成と、加工特徴の一般的な加工指示しか記述されていない。

```
public class Sample
 public static void main(String args[])
 // 加工特徴の定義がここに入る
 // m10_4:4-M10タップ、hole20:Φ20貫通穴
 m10_4.machining();
 hole20.machining();
 }
```

図 6-7 CADKEYによって生成されたOSEL-F加工プログラム（抜粋）

(4)加工すべき加工特徴の漏れがないか確認するために、EXCELに加工シートを表示する。
(5)つぎにOSEL処理系が加工特徴クラスライブラリと工具ライブラリを用いて、工具展開を行ない、OSEL-Fによる工程プログラムを生成する。
(6)無駄な工具交換が生じているので、切削を開始するＺ座標と工具の情報を用いて、EXCEL上でシーケンスをソートし、工具交換回数を最小にする。最適化されたシーケンスデータをもとにOSEL-Fの工程プログラムが最適化されたものに変更される。

```
public class Sample
 public static void main(String args[])
 // 加工特徴の定義文がここに入る
 // m10_4:4-M10タップ、hole20:Φ20貫通穴
 // センタドリリングによる加工
 m10_4.centerdrilling();
 hole20.centerdrilling();
 // ドリル D8.6による加工
```

```
Tool drill8 = toolfordrilling(8.6);     // 工具クラスライブラリから選択
m10_4.drilling(drill8);
// ドリル D15.8による加工
Tool drill15 = toolfordrilling(15.8);  // 工具クラスライブラリから選択
hole20.drilling(drill15);
// エンドミルD15による加工
Tool endmill15 = toolforendmilling(15); // 工具クラスライブラリから選択
hole20.milling(endmill15);
// タッピング
m10_4.tapping();
}
}
```

図 6-8　最適化されたOSEL-F工程プログラム（抜粋）

(7)加工特徴クラスライブラリ、工具ライブラリ、機械依存加工クラスライブラリを用いて、OSEL-Xプログラムが生成される。このOSEL-Xを用いてCADKEYでカッターパスの確認を行なう。

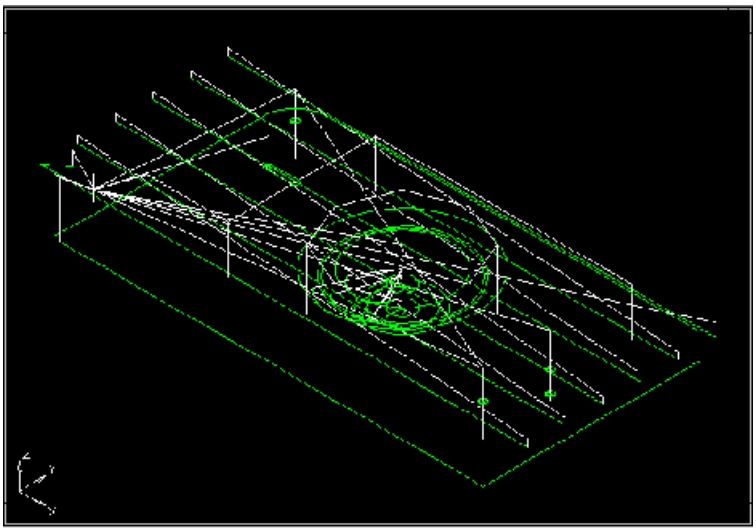

図 6-9　CADKEYによるカッターパスの確認

次に、OSELによる意義を具体的に示す例として、加工に用いる工具の変更が簡単に行なえることを示す。 OSEL は、ユーザが持っているノウハウをプログラムの形で記述し再利用することを可能にしている。たとえば、工具を選択する際には、種々の条件を勘案するが、一度、工具とその使用条件をプログラムしておけば、以後適切な工具が自動的に選択されることになる。また選択した工具に応じ、加工条件やカッターパスが自動的に変更される。実証デモでは、工具の変更をプログラム上で行ない、その工具によってカッターパスが変わることをJAVAアプレットで開発したシミュレータ上で表示した。

このようにOSELを用いることによって、ユーザが持っている加工ノウハウの記述が可能になることを実証し、OSELの実装言語としてJAVAを採用したことから加工ノウハウや

ツールのネットワーク上での流通に道を開いた。

6.6 ネットワークアプリケーション

6.6.1 実装

OSEC-IIでは、イントラネットおよびインターネット双方におけるオープンコントローラの利用について検討を進めており、創出される概念をインターネット電脳工場(Internet Cyber Factories)と呼ぶ。今回の実証システムにおいては、
(1)稼働管理、(2)リモートモニタリング、(3) TV会議応用
などのアプリケーションを試作した。全体のシステム構成の概要を図6-10に示す。これらはインターネット技術を応用し、かつパソコン用の汎用ソフト・ハードを利用している。

図中、各ステーションはネットワークを介してラインデータベースに対し稼働中のステータスを送信し、一方ネットワークパソコン群および携帯端末からWWWブラウザ及び表計算ソフトをデータブラウザとして用いることでライン監視データベースから稼動状況を読み出している。さらにTV会議システムを用いてライン監視TVあるいはフロア間の連絡を行なっている。なお、コントローラからのステータス送信はOFMP(OSEC Floor Management Protocol)による。

6.6.2 まとめ

インターネット技術、パソコン製品の利用により安価かつ開発期間の短いシステムとなった。今後、インターネット電脳工場では以下のような課題を考慮すべきと考えている。

- ■セキュリティ対策
- ■インターネット上の商取引の仕組み
- ■ネットワーク設備のアドミニストレーション
- ■既存設備のネットワーク化

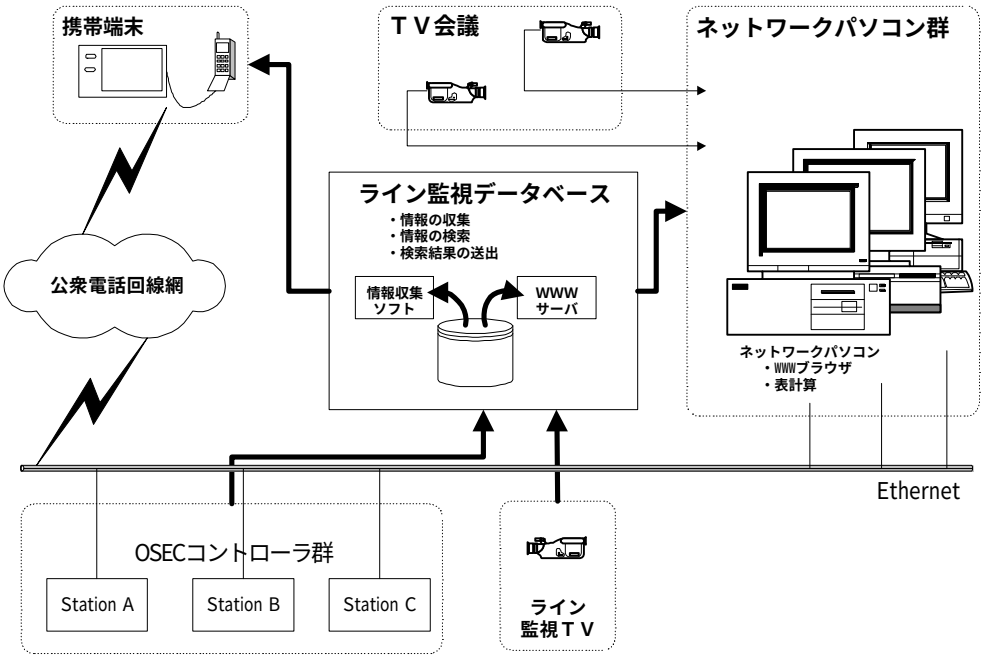

図 6-10 ネットワークシステム全体の構成

7　成果と課題

OSE研究会では４つのワーキンググループを中心に、モジュール構成とそのインタフェースの検討とその実証テストを行い、OSEC-IIとして以下の成果を得ることができた。

（１）アプリケーションフレームによるマンマシンインタフェースシステムの開発

工作機械の各要素をオブジェクト指向関連図(ORD)にまとめ、これらの抽象的なモデル(マシンリソースオブジェクト)をもとに操作・表示系用のクラスライブラリとソフトウェア開発ツールを開発し、機械操作画面を数例試作して実機へ組み込み評価を行った。オブジェクト指向プログラミングの全面採用により、短期間に効率良く表示・操作画面を開発できることが実証された。

（２）OSEL言語の開発

加工ノウハウの組み込みの可能な新しいNC言語OSELの言語仕様について検討し、加工特徴に対応した加工方法を記述するライブラリ・個別の機械に依存するクラスライブラリ・工具ライブラリの３つのクラスライブラリを試作し、実機で加工プログラムを実行して機能を確認した。

（３）マシン制御へのインタフェース仕様と言語プロセッサの作成

軸移動などを生成するためのマシン制御へのインタフェース仕様をOSEC-APIとして定義し、ライブラリの形で実装した。

（４）サーボ制御部へのインタフェース仕様の作成

サーボ制御とのインタフェースをOSEC-APIとして定め、この仕様に合わせてマシン制御とサーボ制御のソフトウェアを開発し、サーボドライブと接続し動作することを確認した。

（５）インターネット応用システムの試作

標準的なパソコンS/Wとインターネット技術だけを使って、リモート監視、ラインの監視とトレンド分析、そしてTV会議応用などのインターネット電脳工場が構築可能であることを示した。

まだ解決すべき課題が残されている。

（１）組み合わせが自由になったがゆえに、マルチベンダから提供された部品・モジュールの組み合わせシステムでの機械停止など、不具合の問題については触れていない。

（２）システムの不具合により引き起こされた事故の責任所在、すなわち製造物責任についても触れていない。

（３）インターネット・イントラネットに接続し、ネットワーク上のリソース(CAD/CAM・生産管理情報・電子商取引き)への自由なアクセスにより、情報有効活用から生まれる生産リードタイムの短縮などのメリットが考えられるが、その反面、外部からの不法侵入やセキュリティに関する問題点には触れていない。

これらは今後の課題ではあるが、OSECで研究を進めているオープンコントローラのアーキテクチャやAPIの標準化が解決に貢献するものと考えている。もちろんそのためには、同様の先端情報処理技術の標準化に取り組んでいる、分散オブジェクト技術のOMG、製品データ交換形式のSTEP、CALS、そして同じオープンコントローラの研究開発を実施している、OMACやOSACAなど幅広い国際共同研究開発作業が不可欠である。

付録　MROとOSEC-IIの機能ブロックAPIリスト

MROと機能ブロックベースOSEC-IIのAPIを以下に示す。APIに関するパラメータや処理内容などの詳細情報については、OSEC-II報告書またはAPI報告書を参照されたい。このAPI仕様書は始めて整理されたものであり、現在進められているOSEC-IIIプロジェクトの進捗状況により修正される可能性がある。OSE協議会としては、今後オープンな議論を行って標準化されるべきもの、そのための基本資料だと考えている。

1 マシンリソースオブジェクト

1．基本ドメインクラスのオブジェクト	
OsecAlarm	機械装置内で発生するアラームを表すクラス。
OsecMachine	汎用的な機械を表すクラス。始動や遮断のためのメソッドをもつ。また、アラームのリストを保持しており、その解除のメソッドを提供する。
OsecNCMachine	NC工作機械を表すクラス。ジョグやプログラム運転などの工作機械固有の動作モードを切り替えるメソッドや、工作機械のロック・アンロックのメソッドなどを提供する。
2．操作系ドメインのオブジェクト	
OsecNCResource	操作系ドメインのリソース・オブジェクトを束ねるためのホルダーの役割をする。派生クラスに複数のリソース・オブジェクトにまたがるような機能を定義することを想定している。また対象の工作機械ごとに実際のリソースの構成を定義することも可能である。
OsecAxis	送り軸を表すクラス。直線軸または回転軸に派生される。
OsecLinearAxis	直線軸を表すクラス。
OsecRotationalAxis	回転軸を表すクラス。
OsecSpindle	主軸を表すクラス。
OsecNCDecoder	NCプログラムのデコーダを表すクラス。
OsecEIADataDecoder	EIAデータのデコーダを表すクラス。
OsecOSELDecoder	OSELプログラムのデコーダを表すクラス。
OsecToolChanger	ツール・チェンジャーを表すクラス。
OsecToolMagazine	ツール・マガジンを表すクラス。
OsecPalletChanger	パレット・チェンジャーを表すクラス。
OsecCoolant	クーラントの供給装置を表すクラス。
OsecChipConveyor	チップ・コンベヤーを表すクラス。
OsecPLC	PLCを表すクラス。PLCプログラムのダウンロードやI/O接点リストの読み書きなどの機能を提供する。標準のオブジェクト・モデルに用意されていない装置を追加する場合などにこのI/O接点リスト上に構築することを想定している。
OsecNCEngine	サーボ系の全体的な制御を行うクラス。合成された送り速度の設定など、複数の軸にまたがるような機能を提供する。
3．基準データドメインのオブジェクト	
OsecRecipeManager	工具テーブル、工具補正値テーブル、NCプログラムなどの基準データを管理するクラス。
OsecNCProgram	NCプログラムを表すクラス。プログラムの格納場所、バージョン、累積実行時間、実行所要時間などの情報を属性として持つ。
OsecTool	工具を表すクラス。工具のID、寿命データ、定格寸法などの情報を属性として持つ。
OsecToolTable	工具テーブルを表すコレクション・クラス。
OsecToolTableItem	工具テーブルの要素を表すクラス。
OsecToolOffsetTable	工具補正値テーブルを表すコレクション・クラス。
OsecToolOffsetTableItem	工具補正値テーブルの要素を表すクラス。工具ID、D/Hコード、補正量などの情報を属性として持つ。

2 リソース管理API

コマンド名	内容
OsecRmOpen()	リソース管理APIのオープン
OsecRmClose()	リソース管理APIのクローズ
OsecRmGetRunCommand()	MROのRUNコマンド取得
OsecRmGetHoldCommand()	MROのHOLDコマンド取得
OsecRmGetEiaProgram()	MROの加工プログラム名取得
OsecRmSetRunStatus()	MROのRUNステータス取得
OsecRmSetAlarm()	MROのアラーム設定
OsecRmGetSingle()	MROのシングルブロックストップ・ステータス取得

OsecRmGetBlockSkip()	MROのオプショナルブロックスキップ・ステータス取得
OsecRmGetOptStop()	MROのオプショナルブロックストップ・ステータス取得
OsecRmSetStopStatus()	MROのストップステータス設定
OsecRmGetReady()	MROのリソース管理機能有効
OsecRmSetFileName()	MROの加工プログラム名設定
OsecRmGetNcAutoStatus()	MROのオートモードステータス取得

3 動作生成 API

コマンド名	内容
OsecMgenOpen()	動作生成APIのオープン
OsecMgenClose()	動作生成APIのクローズ
OsecMgenInit()	動作生成機能の初期化
OsecMgenReset()	動作生成機能のリセット
OsecMgenReady()	動作生成機能の準備完了設定
OsecMgenStart()	加工プログラムの起動
OsecMgenStop()	加工プログラムの停止
OsecMgenStep()	加工プログラムのステップ実行
OsecMgenResume()	加工プログラムの実行再開
OsecMgenSetProgram()	加工プログラム名の設定
OsecMgenGetProgram()	加工プログラム名の取得
OsecMgenSearchBlock()	加工プログラムの内容サーチ

（1）OSEL―EXデコーダAPI例

コマンド名	内容
OsecOselOpen()	OSEL-EXデコーダのオープン
OsecOselClose()	OSEL-EXデコーダのクローズ
OsecOselInit()	OSEL-EXの初期化
OsecOselRun()	OSELプログラムの起動
OsecOselFileOpen()	OSELファイルの読み込み・コマンド解析
OsecOselStep()	OSELプログラムのステップ実行
OsecOselExDebug()	OSELデバッグ出力の設定
OsecOselSetBreakPoint()	OSELブレークポイントの設定
OsecOselReleaseBreakPoint()	OSELブレークポイントのクリア

（2）EIAデコーダAPI例

コマンド名	内容
OsecEiaOpen()	EIAデコーダのオープン
OsecEiaClose()	EIAデコーダのクローズ
OsecEiaInit()	EIAデコーダの初期化
OsecEiaReset()	EIAデコーダ機能のリセット
OsecEiaReady()	EIAデコーダ機能の準備完了設定
OsecEiaStart()	EIAデコーダプログラムの起動
OsecEiaStop()	EIAデコーダプログラムの停止
OsecEiaStep()	EIAデコーダプログラムのステップ実行
OsecEiaResume()	EIAデコーダプログラムの実行再開
OsecEiaSetProgram()	EIAデコーダ加工プログラム名の設定
OsecEiaGetProgram()	EIAデコーダ加工プログラム名の取得
OsecEiaSearchBlock()	EIAデコーダ加工プログラムの内容サーチ

4．マシン制御API

（1）Machine Control Command

コマンド名	内容
OsecMctlOpen()	マシン制御インタフェースのオープン
OsecMctlClose()	マシン制御インタフェースのクローズ
OsecMctlReady()	マシン制御の運転準備状態をON/OFFする
OsecMctlCancelBuf()	コマンドバッファ内のデータ取り消し
OsecMctlStart()	コマンドの解釈処理を開始する
OsecMctlReset()	マシン制御の初期化
OsecMctlStop()	コマンドの解釈処理を停止する
OsecMctlProgramEnd()	コマンドの終了を通知する

（2）Mode Control Command (Option)

コマンド名	内容
OsecMctlModeSingleBlock()	シングルブロックモードのON/OFF
OsecMctlModeDryrun()	ドライランモードのON/OFF
OsecMctlModeExactStop()	イザクトストップモードのON/OFF
OsecMctlModeAxisCancel()	軸動作キャンセルモードのON/OFF
OsecMctlOverCut()	切削速度オーバーライドを設定する
OsecMctlOverPart()	切削速度部分オーバーライド値を設定する
OsecMctlOverMove()	早送り速度オーバーライド値を設定する
OsecMctlAccurate()	切削精度モードを設定する

（3）Axis Control Command

コマンド名	内容
OsecMctlAxisProgramOrigin()	プログラム座標系原点通知
OsecMctlAxisWorkOffset()	ワーク座標系オフセット通知
OsecMctlAxisMove()	目標位置への早送り指令
OsecMctlAxisLine()	直線切削送り指令
OsecMctlAxisCps()	連続位置データ列切削送り指令
OsecMctlAxisCpsAdd()	継続するブロックデータ列を指令
OsecMctlAxisWait()	指定時間停止指令
OsecMctlAxisVelocity()	速度指令
OsecMctlAxisOrigin()	原点復帰指令

（4）Axis Control Command(Option)

コマンド名	内容
OsecMctlAxisArc()	円弧切削指令
OsecMctlAxisArcVector()	円弧切削指令（同時３軸制御）
OsecMctlAxisHelical()	ヘリカル切削指令
OsecMctlAxisTap()	タップ切削指令
OsecMctlAxisSkip()	スキップ機能付き直線送り指令

（5）DI/DO Control Command

コマンド名	内容
OsecMctlSetDIDO()	ＩＯのON/OFF指示

（6）Get Command

コマンド名	内容
OsecMctlGetFreeBufCount()	コマンドバッファの空きサイズを取得
OsecMctlGetUserBufCount()	コマンドバッファのサイズを取得
OsecMctlGetAxisStatus()	軸現在ステータス取得
OsecMctlGetPosition()	現在位置取得
OsecMctlGetProgramOrigin()	プログラム座標系原点取得
OsecMctlGetWorkOffset()	ワーク座標系オフセット取得
OsecMctlGetBlockNo()	ブロック番号取得
OsecMctlGetInfo()	インフォメーション情報取得
OsecMctlGetTargetPosition()	目標位置取得
OsecMctlGetStrokeLimit()	ストロークリミット取得
OsecMctlGetSoftStrokeLimit()	ソフトストロークリミット取得
OsecMctlGetMAXVelocity()	軸最大速度取得
OsecMctlGetDiDoStatus()	ＤＩ／ＤＯステータス取得
OsecMctlGetDiDoData()	ＤＩ／ＤＯデータ取得
OsecMctlGetStatus()	現在ステータス取得
OsecMctlGetTheoryPosition()	理論現在位置取得

（7）Alarm Command

コマンド名	内容
OsecMctlSetAlarm()	アラーム情報通知
OsecMctlGetAlarm()	アラーム情報取得

5．サーボ制御API

（1）インタフェース管理機能

コマンド名	内容
OsecSrvOpen()	サーボ機能のオープン
OsecSrvClose()	サーボ機能のクローズ
OsecSrvInit()	サーボ機能の初期化
OsecSrvSetInfo()	サーボ機能の情報の設定
OsecSrvGetInfo()	サーボ機能の情報の取得

（2）グループ管理機能

コマンド名	内容
OsecSrvGrpInit()	サーボグループ機構の初期化
OsecSrvGrpSetInfo()	サーボグループ機構の情報の設定
OsecSrvGrpGetInfo()	サーボグループ機構の情報の取得

（3）サーボ軸管理機能

コマンド名	内容
OsecSrvAxisInit()	サーボ軸機構の初期化
OsecSrvAxisSetInfo()	サーボ軸機構の情報の設定
OsecSrvAxisGetInfo()	サーボ軸機構の情報の取得

（4）バッファ管理機能

コマンド名	内容
OsecSrvGetFreeBufCount()	空きバッファ数の取得
OsecSrvGetUsedBufCount()	使用バッファ数の取得
OsecSrvCancelBuf()	バッファのキャンセル

（5）サーボ制御処理機能

コマンド名	内容
OsecSrvErrorReset()	エラーリセット
OsecSrvSetSrvOn()	軸のサーボＯＮ
OsecSrvSetSrvOff()	軸のサーボＯＦＦ
OsecSrvSetCyclicCmd()	サーボ指令値の設定
OsecSrvGetCyclicData()	サーボの観測データの読み込み
OsecSrvSyncroCyclicData()	サーボ指令との同期
OsecSrvWaitEvent()	サーボ機能のイベント検出機能

６．PLC入出力API

（１）インタフェース管理

コマンド名	内容
OsecPlcOpen()	OSEC－PLCオープン
OsecPlcClose()	PLCクローズ
OsecPlcInit()	初期化
OsecPlcGetInfo()	情報獲得
OsecPlcSetInfo()	情報設定

（２）PLC処理

コマンド名	内容
OsecPlcGetIOData()	データ取得
OsecPlcSetIOData()	データ設定
OsecPlcGetIDbyName()	ID番号の取得