数値風洞用 MSP ビュー・ユーザ・インターフェースの開発

土　屋　雅　子*1

The Development of MSP View User Interfaces for a Numerical Wind Tunnel*

Masako TSUCHIYA*1

ABSTRACT

In this paper we present the concepts and detailed functions of the MSP View User Interfaces developed for the Numerical Wind Tunnel(NWT), a distributed memory parallel computer with UNIX operating systems.

Users at the NAL however were very skilled in the use of MSP operating systems for general purposes, and it was necessary therefore to ensure continued usability by users and of user assets.

For this reason we made Interfaces in NWT in MSP View. Catalogued procedures(NS-CATAPRO) and command procedures(NS-COMMAND) were developed as MSP View User Interfaces. In job processings with these Interfaces a number of the differences between UNIX and MSP can be discounted, and NWT used in the same manner as MSP. As a result, users can use NWT easily and with confidence. Moreover, there is no need to change resources from MSP to UNIX.

Experiments have shown that the intended effectiveness of MSP View User Interfaces have been achieved.

Key Word : NWT, MSP, UNIX, operating system, user interface, parallel computer, program, file, catalogued procedure, command, job

概　　要

主記憶分散型並列コンピュータである数値風洞 (NWT) のジョブ処理に必要なユーザインターフェースを開発した。NWT のオペレーティング・システム (OS) は UNIX であるが，航技研ユーザは長年運用されてきた汎用 OS の MSP システムの利用に長けていた。したがって，NWT においても，ユーザ利用性の継続とユーザ資産の継承を図る必要があった。この観点から，NWT のユーザ・インターフェースは MSP ビューとした。本稿では，MSP ビュー・ユーザ・インターフェースとして開発されたジョブ制御言語マクロ形式の NS カタプロと会話型マクロ・コマンド形式の NS コマンドを利用したジョブ処理について述べる。これらのユーザ・インターフェースを利用したジョブ処理の中では，二つの異種 OS の相違が自動吸収され，UNIX システムの NWT を従来 OS の MSP と等価な利用とすることが実現した。この結果，NWT の運用において，ユーザ利用性の継続とユーザ資産の継承という所期の目標を達成することができた。

＊　平成 11 年 11 月 1 日　受付　(received 1 November 1999)

＊1　計算科学研究部　(Computational Science Division)

This document is provided by JAXA.

1．はじめに

平成５年２月，航技研に導入された科学技術用超高速計算機システムの数値風洞（以降，NWTと略記）は，要素計算機にベクトル計算機を配置する分散主記憶型の並列計算機システムである[1),2)]。また，そのオペレーティングシステム（以降，OSと略記）は富士通（株）製のUXP/M（以降，UXP/Mと略記）[3),4)]というUNIXベースのシステムである。

一方，航技研では汎用大型電子計算機システムのOSとして富士通独自のOSⅣ／MSP[5)]（以降，MSPと略記）を昭和57年より運用している。その結果，ジョブ処理方式やジョブストリーム記述言語等，ユーザ側から見たシステムの使い勝手等に対するノウハウを含めたMSP資産の蓄積は膨大であった。

この観点から，NWTの利用におけるユーザインターフェースはMSPビューのユーザ・インターフェースとし得る運用システムを構築した。

MSPとUNIXではジョブ処理の体系からファイルの形式等，ならびに各種のユーザインターフェースに関しても相違点が多数ある。先ず，MSPではシステムへの大規模計算や長時間計算のための処理の投入から実行および実行結果の取り出しに至るまで，複数のプロセスはジョブという処理単位で統合管理される。一方，UNIXにはジョブという概念はない。NWTをMSPと同様なジョブ管理方式で利用すること，すなわちMSPビューのユーザインターフェースで利用するためには，ジョブ処理過程の多くのフェーズで二つの異種OSの相違点を吸収し，ユーザがそれぞれのOSの違いを意識せずに，かつ容易にNWTを利用するためのシステム運用機能が必要となる。

本稿は，このような目的からNWTのジョブ処理用に開発されたMSPビュー・ユーザ・インターフェースについて述べる。

なお，MSPビュー・ユーザ・インターフェースとは，これを一口で言うならば，UNIXシステムであるNWTの利用をMSPのそれと等価にするものである。すなわち，MSPビュー・ユーザ・インターフェースには，システム利用性の継続およびユーザ資産の継承という二つの目標を叶える機能が具備されなければならない。この目標達成のために，ジョブストリーム記述用カタログド・プロセジャ[6)]，および会話型処理（以降，TSSと略記する）用コマンド・プロセジャ[8)]というMSPのマン・マシーン・インターフェース機能を駆使するNSカタプロ[7)]とNSコマンド[9)]というユーザ・インターフェースを開発し，それぞれにNWTジョブ処理用に必要な諸機能を集約した。MSPビュー・ユーザ・インターフェースの開発により，長年，MSPの文化に慣れ親しんできたユーザは，全く馴染みのないUNIXという異種OS文化を容易に利用することが可能となった。この結果，航技研のMSPユーザはUNIX学習にかける時間と，何時のシステムリプレースにも不可避なシステム移行処理に対するユーザ作業を大幅に削減することができた。

2．NWTのハードウェア構成概要

NWT導入時から大型電子計算機システムが更新される平成８年１月までの運用システムを旧運用システム，平成８年１月以降，現在までの運用システムを現運用システムと呼称し，それぞれのシステム中核部のハードウェア概念図を図2.1および図2.2に示す。図2.1に示すとおり，旧運用システムのNWTは既設の大型電子計算機システムをフロントエンドシステムとして有機的に結合

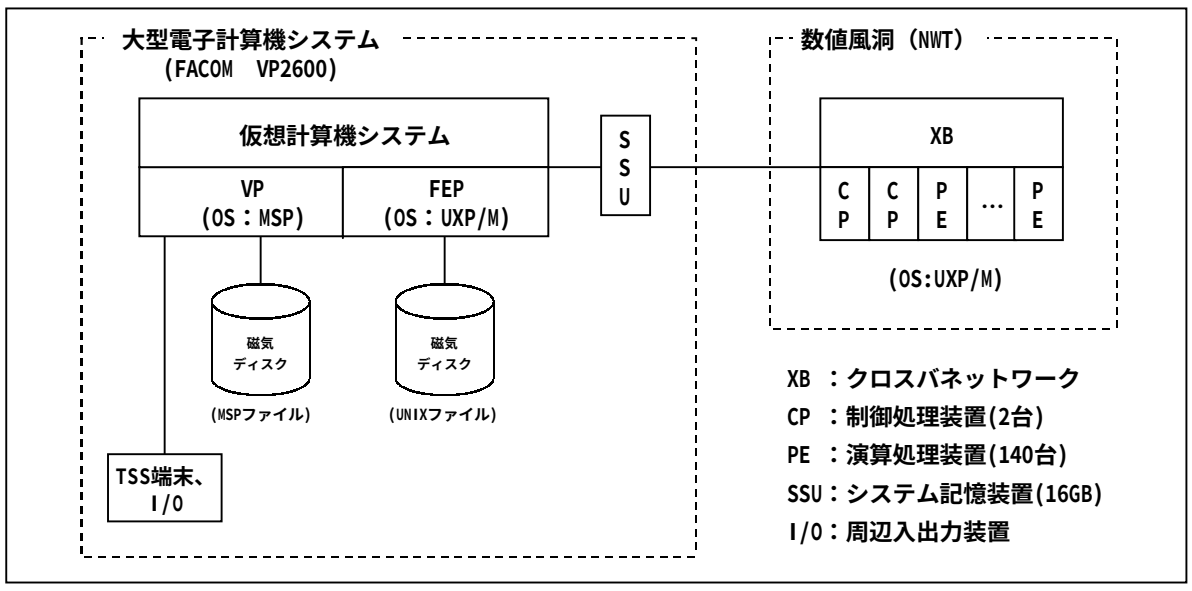


図2.1　中核部のハードウェア構成概念図（旧運用システム）

This document is provided by JAXA.

した複合計算機システムを構成している。NWTの中核部は256MBメモリを実装した要素計算機（PE）140台，OS処理のみを実行する制御処理装置（CP）２台，およびそれらを相互接続するクロスバネットワーク（XB）より構成される。CPはシステム記憶（SSU）を介して大型電子計算機システムと接続される。NWTのハードウェアが有する超高速処理性能を十分に発揮するため，NWTには直接的には入出力装置を接続していない。したがって，ジョブ入出力のフロントエンド処理やNWTで実行する大規模数値シミュレーションジョブ（以降，NWTジョブと呼称する）の先行する翻訳処理，結合・編集処理ならびにTSS処理等はNWTのフロント・エンド・プロセッサ（FEP）として大型電子計算機システムが分担し，NWTでは数値シミュレーション実行処理に専念するシステムとしている。

各システムに搭載しているOSについては，次のとおりである。NWTにはUNIX System Ⅴ Release4.1に準拠したUXP/Mを搭載している。また，大型電子計算機システムのOSは従来OSのMSPと，一方，NWTとFEPの緊密な連携をとるため，親和性の観点から，UXP/Mの二つのOSを搭載し，仮想計算機システムを構築した。なお，大型電子計算機システムでは，UXP/MでNWTのFEPとしての役割を遂行するとともに，MSPにおいては，大規模ベクトル計算ジョブ処理の実行システム（以降，VPと呼称する）として，従来からのシステム運用をも継続した。

平成８年１月には大型電子計算機システムは更新され，図2.2に示す現運用システムのハードウェア構成となった。同図に示すとおり，NWTのPEは166台構成（256MBメモリ実装PE：162台，１GBメモリ実装PE：４台）となり，SSUの容量は16GBより24GBに拡張された。システム更新により，二つのOSを有した大型電子計算機システムはNWT用のフロントエンドプロセッサ専用計算機システム（以降，NWT－FEPと呼称する）として富士通FACOM　VPX－220システムにリプレースされ，OSはUNIXのUXP/Mのみが搭載された。このときからNWTのジョブ処理では，NWT－FEPのUXP/MからUNIXを直接利用できる運用を開始した。また，MSPビュー・ユーザ・インターフェースは平成６年２月末にファイルサーバシステムとして導入された汎用大型電子計算機システムFACOM　VP2100システムに移植され，ネットワークを介してNWT－FEPと通信することにより，NWTジョブ処理の運用を継続した。なお，本稿は旧運用システムにおけるMSPビュー・ユーザ・インターフェースに照準を合わせて記述しているが，その機能や形式においては，現運用システムのそれとほとんど差異はない。

3. ユーザビューにおけるMSPとUNIXの主な相違点

大型かつ超高速のスーパコンピュータから個人ユースのパソコンにいたるまで，本来，計算機システムはOSが異なると，システムの利用性から走行させるプログラムもOSに依存し非互換であったり，互換性はとれても大幅な変更を強いられることがある。本章では，VPのOS「MSP」およびFEPやNWTのUNIXベースOS「UXP/M」という二つの異種OSについて，そのユーザビューにおける主な相違点を列挙する。

3.1　ジョブの記述形式

MSPで実行処理するジョブは実行プログラムごとにジョブステップという処理単位で定義され，一つまたは複数のジョブステップ列で構成される。一方，UNIXでは，

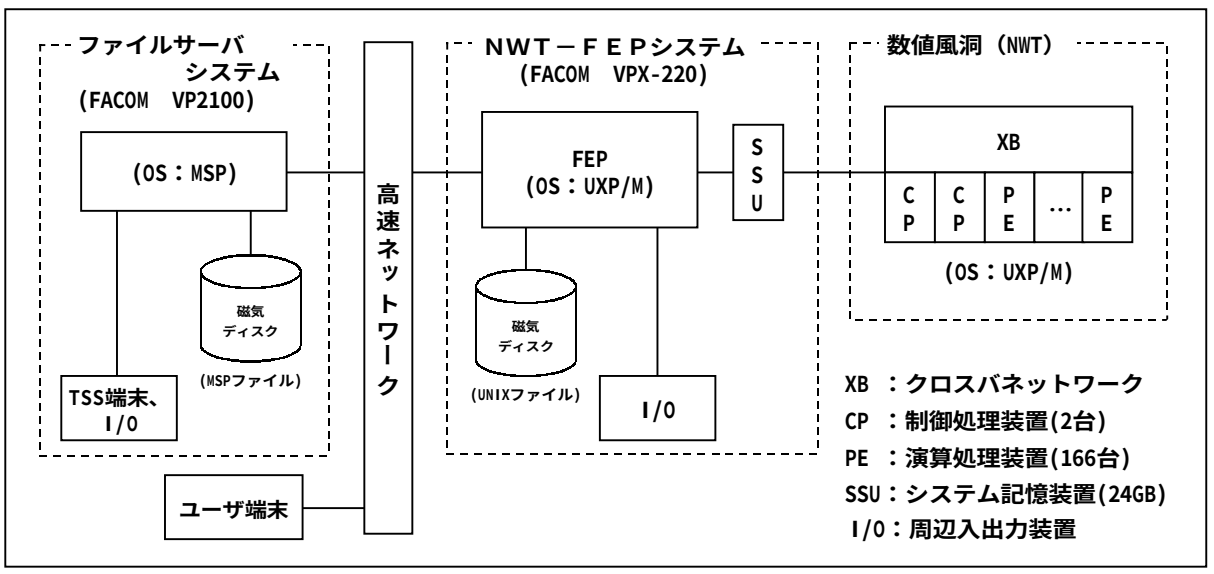


図2.2　中核部のハードウェア構成概念図（現運用システム）

This document is provided by JAXA.

MSPのジョブに対応する処理単位の概念はなく，バッチリクエストというプロセスの集合が，MSPのジョブステップに対応している。また，バッチリクエストはUNIXシステム固有のシェルスクリプト[10]で記述される。さらに，UNIXには個々のユーザごとに多数発生し，処理待ちキューに並ぶバッチリクエストに関して，その実行順序を保証するというジョブ処理には必須の機能が欠けている。

3. 2 ファイル・システム

NWTジョブのユーザプログラムは，その殆どがFORTRANプログラムで記述されている。表3.1に標準的なNWTジョブの処理において入出力されるファイルの種類を示す。同表に示されるように，各種のファイルはUNIXの各処理フェーズにおいて入出力される。表中のファイルの保存システムではファイルはそのシステムに有効なファイルの形式でなければならない。これらのファイルの形式に関しても，MSPとUNIXの二つのOSの違いによる差異は大きい。先ず，MSP形式のファイルは区分編成や順編成等，種々のファイル編成と任意の長さのブロック長をもつ。一方，UNIX形式のファイルは単なるバイト列であり，固定長ブロックのデータの集まりである。また，実行時使用入出力ファイルであるFORTRANレコード[11]の各種レコード形式におけるデータフォーマットも異なる。図3.1は異種OSを遷移するNWTジョブに主要な入出力データについて，MSPとUNIXにおけるデータフォーマットの相違点を示す。同図に示すとおり，異種OS間のデータフォーマットの違いは顕著である。図中のテキストデータは表３．１の印刷出力ファイル，実行時標準入力ファイルおよび小規模実行時使用入出力ファイルの内容に対応する。また，FORTRANレコードは大規模および小規模実行時使用入出力ファイルの内容に対応する。なお，プログラムの各種モジュールはシステム固有の機械語であり，全く異なるフォーマットを有している。その比較には意味がないので省略する。MSPビュー・ユーザ・インターフェースとするためには，これらのファイルがMSP − UNIX間を遷移する都度，MSP形式からUNIX形式に，またはUNIX形式からMSP形式へのデータフォーマットを変換処理するための工夫が必要となる。

3. 3 NWTジョブ実行結果の取り出し方式

MSPでは通常，ジョブを構成する各ジョブステップの実行結果として印刷イメージのファイルを標準出力する。NWTでは，この標準出力ファイルを出力されてから７日間は保存状態とする期限管理の運用を行っている。ユーザはこの結果を必要時に，期限内に印刷出力装置よりプリントとして取り出す。MSPではこの標準出力ファイルもジョブとして管理され，ジョブ単位に統合され，１つの出力待ちジョブとして，ユーザファイルとは異なるシステムのスプール・ファイルに出力される。一方，UNIXの場合には標準印刷出力結果はプロセス単位に通常の入出力ファイルと同様にユーザファイルとして出力される。複数ステップからなるFORTRANプログラムのジョブをUNIXで実行すると，標準印刷出力結果として，翻訳処理結果，結合・編集処理結果および実行処理結果等，多数の印刷イメージファイルがそれぞれ関連付けなく作成される。これをMSPビューのユーザインターフェースとするためには，これらのファイルをジョブとして実行順に統合して扱うための工夫が必要となる。

3. 4 文字データの内部コード

ソースプログラムやソースデータ等のカードイメージのデータにおける文字データの内部コードはMSPではEBCDICコードであるが，UNIXではASCIIコードである。MSPビュー・ユーザ・インターフェースとするため

表3.1 NWTジョブの入出力ファイル

処理フェーズ	ファイルの種類	入出力モード	ファイルの保存システム
翻訳処理	FORTRANソースプログラム	入力	VP(MSP)
	コンパイラの印刷出力ファイル	出力	
	オブジェクトモジュール	出力	FEP(UXP/M)
結合・編集処理	オブジェクトモジュール	入力	FEP(UXP/M)
	ユーザライブラリ	入出力	
	システムライブラリ	入力	
	ロードモジュール	入出力	
	リンケージエディタの印刷出力ファイル	出力	VP(MSP)
数値シミュレーション実行処理	ロードモジュール	入力	FEP(UXP/M)
	実行時標準入力ファイル	入力	VP(MSP)
	実行時標準印刷出力ファイル	出力	
	実行時使用入出力ファイル（大規模）	入出力	
	実行時使用入出力ファイル（小規模）	入出力	

This document is provided by JAXA.

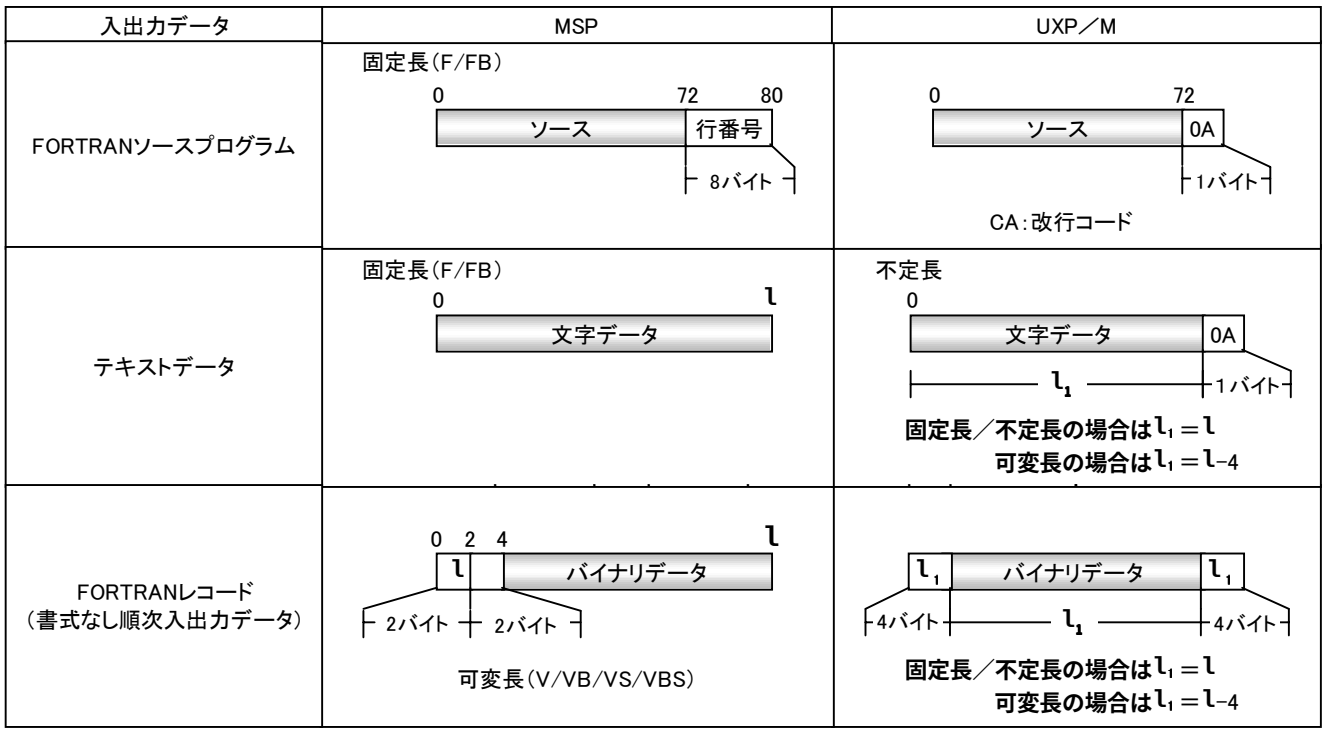


図 3.1　主要な入出力データのフォーマット

には，文字データが MSP − UNIX 間を遷移するためには，EBCDIC コードから ASCII コードへ，または ASCII コードから EBCDIC コードへ，コードを変換処理するための工夫が必要となる。

4．MSP ビュー・ユーザ・インターフェースの機能概要

第 3 章に示した MSP と UNIX におけるユーザ・インターフェースの相違点を吸収し，UNIX という異種 OS の利用をユーザに意識させずに NWT ジョブ処理を可能にするためには，以下に列挙する機能要件を実現する必要がある。

(1) MSP からジョブを投入できること。

(2) ジョブを構成するジョブステップの実行順序を違えることなく，FEP や NWT において順番にバッチリクエストとして処理できること。

(3) MSP からジョブの実行結果の検索ができること。また，ジョブの実行結果の取り出し，ならびにキャンセルができること。

(4) NWT ジョブについて，システム内の実行待ちジョブ混雑状況および実行中ジョブ状況を確認できること。

上記要件を実現するために，MSP ビュー・ユーザ・インターフェースとして NWT ジョブ処理用の NS カタプロと NS コマンドを開発し，必要となる機能を集約した。本章では，NWT ジョブ処理用に開発された NS カタプロと NS コマンドの機能概要について記述する。これらのユーザ・インターフェースを介し，システムに発生した NWT ジョブは本来のユーザプログラム処理と，ユーザ・インターフェース中には明確に現れない多くの OS 処理フェーズを経由しながら実行処理されていく。なお，MSP ビュー・ユーザ・インターフェースによる NWT ジョブ処理の実現方式については，第 5 章「NWT ジョブ処理実現方式の概要」と第 6 章「ファイル転送実現方式の概要」で説明する。

4.1 NS カタプロの機能概要

VP における MSP の運用では，MSP が有するカタログド・プロシジャ機能を駆使した航技研独自の NS カタプロを多数用意している。NS カタプロとは実行プログラム名や入出力ファイル定義文等のジョブを記述する制御文をマクロ化した定義文である。通常，ジョブの記述にはシステムプログラムや入出力ファイル等のユーザには明解でない多数のファイル文や実行定義文が必要である。このため，ジョブの記述を正確に行うためには，システムの構成，システムパラメータならびにシステム固有のジョブ制御言語の文法を熟知していなければならない。また，ジョブの記述を完成させるまでには，ジョブコンエラーなる記述上のシンタックスエラーを取り除くために，何度もジョブを投入しなければならず，目的のユーザプログラムの実行処理に到達するまでには，非常に煩雑な作業と多大な時間を費やすことになる。

一方，NS カタプロでは大方の処理に共通な手続きはカタプロ展開時に自動展開されるので，ユーザは複雑な定義は不要となり，ユーザジョブに固有の定義につい

This document is provided by JAXA.

て，パラメータで簡単に指定できる仕組みになっている。また，NS カタプロはジョブ実行時の処理装置使用時間，メモリおよびファイル等のシステム資源量についてシステム標準値やセンタが決める省略値を有し，ユーザがパラメータの指定を省略すると，所定の標準値または省略値が設定されるようになっている。この標準値および省略値にはシステムが効率的に運用管理できるようにチューニングされた値が定義されている。したがって，NSカタプロを活用すれば，目的とするジョブの記述が非常に容易に，かつ確実となり，さらにシステム資源の無駄等も省けるので，ユーザのみならずシステム運用管理上も利点が大きい。以下に NWT ジョブ処理用のMSPビューインターフェースとして開発されたNSカタプロの機能概要を示す。なお，各項に付記する参考の表については文末に一括して配置する。また，参考の表は現運用システムの形式で記述する。

（1）NJOB 文

NWTで実行するジョブの先頭を示す文である。NJOB 文では，ユーザのシステム使用権，ユーザ属性ならびにシステム内投入ジョブ数等を確認するためのユーザ登録名，およびジョブクラス等が記述できる。表4.1にNJOB 文の記述形式および機能詳細を示す。

（2）NFORTC 文

NWT ジョブの FORTRAN プログラムの翻訳処理を FEP で実行することを指示する文である。NFORTC 文では，FORTRANコンパイルオプションおよび入力するソースプログラムのファイル名等が記述できる。表４．２に NFORTC 文の記述形式および機能詳細について示す。

（3）NLIED 文

NFORTC 文による先行の FORTRAN コンパイル処理が出力するオブジェクトモジュールを入力し，FEPでプログラムの結合・編集処理することを指示する文である。NLIED 文では，結合・編集時のオプションと実行結果としてのロードモジュールをFEPに作成保存する場合のファイル名が記述できる。表4.3にNLIED 文の記述形式および機能詳細について示す。

（4）NGO 文

NLIED 文による結合・編集処理で作成保存されたロードモジュールを入力し，NWT で数値シミュレーションプログラムの実行処理を指示する文である。NGO 文では，実行時のオプション，CPU打ち切り時間および既存の実行プログラムとしてユーザ保存ファイル名等の指定が可能であるとともに，実行時使用標準入力ファイルとしてMSP上のユーザ保存ファイルが指定可能である。表 4.4 に NGO 文の記述形式および機能詳細について示す。

（5）NUSDKR 文

NWT における数値シミュレーション実行時使用入出力ファイルの内，入力ファイルの使用を指示する文である。NUSDKR 文では，MSP 上のユーザ保存ファイルとともに，FEP 上のユーザ保存ファイルを指定することも可能にしている。表4.5にNUSDKR文の記述形式および機能詳細について示す。

（6） NUSDKRW 文

NWT における数値シミュレーション実行時使用入出力ファイルの内，入力と出力をともに行うファイルの使用を指示する文である。NUSDKRW 文では，MSP 上のユーザ保存ファイルとともに，FEP上のユーザ保存ファイルを指定することが可能である。表4.6にNUSDKRW 文の記述形式および機能詳細について示す。

（7）NUSDKW 文

NWT における数値シミュレーション実行時使用入出力ファイルの内，出力ファイルの使用を指示する文である。NUSDKW 文では，MSP 上のユーザ保存ファイルとともに，FEP上のユーザ保存ファイルを指定することが可能である。表 4.7 に NUSDKW 文の記述形式および機能詳細について示す。

（8）NXY 文

NWT における数値シミュレーション実行時入出力ファイルの内，静的図形データファイルの出力を指示する文である。MSP のセッション画面で NXY 文による出力ファイルを XY プロッタイメージで表示し，数値シミュレーション実行処理の検証を行うことができる。また，実際に XY プロッタ装置から図形出力を行うこともできる。表4.8にNXY 文の記述形式および機能詳細について示す。

（9）XMTON 文

MSP のファイルを UNIX のファイルに形式変換することを指示する文である。XMTON 文は MSP で実行するジョブ処理の中でファイル形式の変換を可能にするものである。表 4.9 に XMTON 文の記述形式および機能詳細について示す。

（10）XNTOM 文

UNIX のファイルを MSP のファイルに形式変換することを指示する文である。XNTOM 文は MSP で実行するジョブ処理の中でファイル形式の変換を可能にするものである。表 4.10 に XNTOM 文の記述形式および機能詳細について示す。

4. 2　NS コマンドの機能概要

MSP のシステム運用では，MSP が有するコマンド・プロシジャ機能を駆使した航技研独自の NS コマンドを多数，用意している。NS コマンドとは，TSS のセッションから投入可能な各種のコマンドをマクロ化したコマ

This document is provided by JAXA.

ンドである。NSカタプロによるジョブの記述と同様に，大方のコマンド処理に共通な手続きはコマンド展開時に自動展開されるので，ユーザのコマンド実行に必要な固有の定義について，コマンドのオプションで簡単に指定できる仕組みになっている。また，コマンド実行に必要な各種のシステム資源量についてもNSカタプロと同様，運用効率を高めるようなシステム標準値や省略値を定義している。したがって，NS コマンドを活用すると，会話型処理におけるジョブの操作やファイルの作成・編集処理等が容易に，かつ効率的に行える。以下に MSP ビュー・ユーザ・インターフェースとして開発されたNWTジョブ処理用 NS コマンドの機能概要について述べる。なお，各項に付記する参考の表については文末に一括して記述する。また，参考の表は現運用システムの形式で記述する。

（1）NSUB コマンド

NS カタプロで記述された NWT ジョブの投入を指示するコマンドである。NSUBコマンドでは，MSPセッションのコマンド入力画面でジョブストリームが格納されているユーザ保存ファイル名を入力し，投入できるとともに，ジョブ記述のための NS カタプロを編集するファイル編集メニュー画面のサブコマンド状態で投入することもできる仕組みになっている。表4.11にNSUBコマンドの入力形式および機能詳細について示す。

（2）NFORT コマンド

NWTで実行するFORTRANプログラムの翻訳処理をＦＥＰ で実行することを指示するコマンドである。NFORTコマンドでは，MSPセッションのコマンド入力画面において，FORTRANソースプログラムが格納されているユーザ保存ファイル名を入力可能であるとともに，ファイル編集メニュー画面においてサブコマンドとして入力することも可能な仕組みになっている。表4.12にNFORT コマンドの入力形式および機能詳細について示す。

（3）NCAN コマンド

MSPのセッションからNWTジョブのキャンセルを指示するコマンドである。NCANコマンドでは，MSP，FEPおよびNWTで実行待ち，実行中等，各種の状態における NWT ジョブについてキャンセルを指示できる。表4.13 に NCAN コマンドの入力形式および機能詳細について示す。

（4）NS コマンド

MSPのセッションからユーザのNWTジョブ処理状況表示を指示するコマンドである。NS コマンドでは，実行待ち，実行中および実行結果出力待ち等，各種の状態におけるNWTジョブについてその状況表示を指示できる。表4.14 に NS コマンドの入力形式および機能詳細について示す。

（5）ACTJOB コマンド

MSPのセッションからNWTジョブの実行状況の表示を指示するコマンドである。ACTJOB コマンドでは，FEPおよびNWTで実行中の全ユーザジョブについて割当 PE 台数，CPU 使用時間ならびに経過時間等の詳細なジョブ実行状況を表示する。表 4.15 に ACTJOB コマンドの入力形式および機能詳細について示す。

（6）WAITJOB コマンド

MSPのセッションから実行待ちNWTジョブの混雑状況の表示を指示するコマンドである。WAITJOB コマンドでは，NWT における数値シミュレーション実行処理を起動待ちしている全ユーザジョブについて要求 PE 台数，要求CPU使用時間ならびに起動優先権等，詳細なジョブ実行待ち状況を表示する。表4.16はWAITJOBコマンドの入力形式および表示結果を示す。

（7）XMTON コマンド

MSP で作成されたファイルを UNIX のファイルに形式変換することを指示するコマンドである。表 4.17 に XMTONコマンドの記述形式および機能詳細について示す。

（8）XNTOM コマンド

UNIX システムで作成されたファイルを MSP のファイルに形式変換することを指示するコマンドである。表4.18にXNTOM文の記述形式および機能詳細について示す。

5. NWTジョブ処理実現方式の概要

MSP ビュー・ユーザ・インターフェースによるNWTジョブ処理の流れの概念図を図 5.1 に示す。図 5.1 では，MSP ビュー・ユーザ・インターフェースを介し，MSP上のセッション処理の中から発生したNWTジョブがVP上のMSPシステムとFEPならびにNWTにおけるUXP/M の三つの OS を遷移し，各種の処理フェーズを経由しながら処理されていく過程を示している。同図にもとづいて NWT ジョブ処理実現方式の概要を以下に示す。

5. 1 NWT ジョブ投入処理

MSP が管理する各種の TSS 端末のセッションから，NWT ジョブを投入する処理フェーズである。ジョブ投入処理フェーズを MSP ビュー・ユーザ・インターフェースとするためには，また，各種ファイルの作成・編集等の会話型処理におけるシステムの快適なレスポンスを保証する観点からも，NWT ジョブの実行に必要なファイルはすべて MSP 配下の磁気ディスク装置に格納する方式とした。

5. 2 MSP 受付および終了処理

本処理フェーズは，ユーザが定義する NWT ジョブの

This document is provided by JAXA.

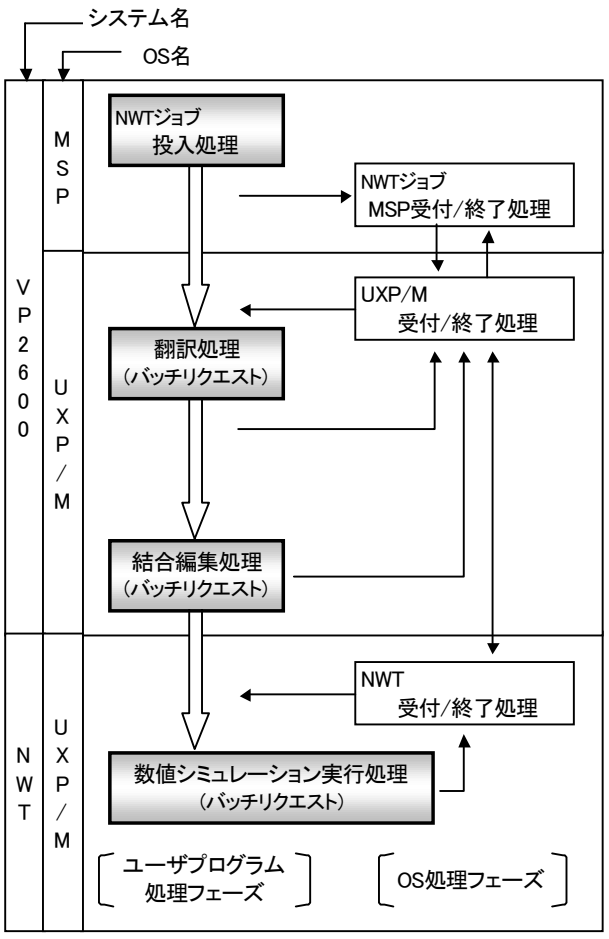


図 5.1　NWT ジョブ処理の流れの概念図

手続きには直接には現れないMSPのOS処理フェーズである。この処理フェーズでは，ジョブ投入のNSUBコマンドを受けて，NWT ジョブの受付処理を行う。また，FEPおよびNWTで終了したバッチリクエストの実行結果として出力された各種のファイルを受信し，MSPのジョブとして復元する処理を行う。第 3.3 節の NWT ジョブ実行結果取り出し方式におけるMSP－UNIX異種OS間の相違については，本処理フェーズで吸収する。以下に本処理フェーズの主な処理概要を示す。

（1）NSカタプロ確認／検査処理

NWT ジョブのジョブストリームを記述する NS カタプロについて，指定必須なパラメータと構文，および全入出力ファイルの割当の確認／検査を行う。NWT ジョブ記述上のエラーによるジョブ再投入の回数を少なくするため，本処理フェーズでは，MSPでなければ確認できない処理のみを行い，エラー処理はUXP/M 受付処理で集約する。NS カタプロの記述内容にエラーがある場合にはその旨のメッセージをユーザごとのメッセージ・ファイルに出力する。

（2）NSカタプロ変換処理

NS カタプロ記述形式を UXP/M 上で確実に UNIX シェルスクリプトに変換するために，MSP 上ではNSカタプロとシェルスクリプトの中間の形式を採った中間シェルスクリプトのジョブ記述形式に変換する。図5.2にNSカタプロの中間シェルスクリプトへの変換例を示す。同図に示すとおり，NSカタプロは，余分な記述子を取り，カタプロ形式の名残を幾分残した中間シェルスクリプトなる形式に変換される。なお，第3.1節に示したMSPとUNIX の異種 OS 間におけるジョブ記述形式の相違は二つの処理フェーズに分けて吸収し，第一段階の吸収を本処理フェーズで実行する。

NSカタプロの記述形式

```
//X01PASS NJOB XXXX
//     EXEC NFORTC,PARAM="-Os,-i,-Pt",
//          SF='X01.A.FORT77',SINC='X01.A.INCLUDE'
//     EXEC NLIED,OEF='X01.A.LOAD(MAIN)'
//     EXEC NGO,EF='X01.A.LOAD',EMEM=MAIN,DF='X01.A.DATA'
//  EXPAND NUSDKR,RNO=01,FILE='X01.FT01.INPUT',CONV=YES
//  EXPAND NUSDKW,RNO=02,FILE='X01.FT02.OUTPUT',ORG=PO,
//          DISP=NEW,SPACE='(2,2)'
```

中間シェルスクリプトの記述形式

```
njob   -r X01S767 -q XXXX -m X01S767
        _NWT.NWJ.X01S767.SH_nb
nfortc -p'-Os,-i,-Pt' -I $TMP/nwj/X01S767/include
        X01S767_X01.A.FORT77(A1)_nn.f
        X01S767_X01.A.FORT77(B1)_nn.f
nlied  -o X01S767_X01.A.LOAD(MAIN)_nb
ngo    -f X01S767_X01.A.LOAD(MAIN)_nb
       <  X01S767_X01.A.DATA_nn
nusdkr -n 01   X01S767_X01.FT01.INPUT_nb
nusdkw -n 02   X01S767_X01.FT02.OUTPUT_nb
nuswk  -n 08
ngoend
```

図 5.2　NS カタプロの中間シェルスクリプトへの変換例

（3）NWT ジョブの転送

MSPで正常に受け付けたNWTジョブをFEPのUXP/M に自動転送する。

（4）ファイル送／受信のセットアップ処理

NS カタプロ中に指定された MSP 形式のファイルを UXP/M に転送するためのセットアップ処理を行う。また，バッチリクエスト実行終了時に UXP/M から送信された各種のファイルを展開し，必要ならばテキスト変換を行う。さらに，表 3.1 のファイルの保存システムに示すように MSP 上の所定の展開先ファイルに格納する。

(5) ジョブ終了メッセージ，エラーメッセージ等の情報を MSP の関連ファイルに格納する。

5．3 UXP/M 受付および終了処理

本処理フェーズは FEP における UXP/M の OS 処理フ

This document is provided by JAXA.

中間シェルスクリプトの記述形式

```
njob   -r X01S767 -q XXXX -m X01S767
          NWT.NWJ.X01S767.SH_nb
nfortc -p '-Os -Pt' -I
          X01S767_X01.A.INCLUDE
          X01S767_X01.A.FORT77(A1)_nn.f
nlied  -o X01S767_X01.A.LOAD(MAIN)_nb
ngo    -f X01S767_X01.A.LOAD(MAIN)_nb
       -t 18000
nusdkr -n 01  X01S767_X01.A.INPUT_nb
nusdkw -n 02  X01S767_X01.A.OUTPUT_A0_nn
ngoend
```

UNIXシェルスクリプトの記述形式

```
#!/bin/sh
# /usr/NWT/bin/nwjqsub <<"!NWJ_FEP"
qsub <<"!NWJ_FEP"
##
#@$-s /bin/sh
#@$-q FEPX
#@$-r X01S767
#@$-lt 3600
#@$-ln 4
#@$-o /nwj/X01S767/acct_X01S767_01
#@$-g x
#@$-eo
#@$
#!/bin/sh
    cd /nwj/X01S767
    # compile
    f77nwt  -c -Os -Pt -I /nwj/X01S767/X01.A.INCLUDE \
                          /nwj/X01S767/X01.A.FORT77/A1_nn.f \
                    >>    /nwj/X01S767/jesf_X01S767_01 2>&1
    rtncode=$?
    if [ $rtncode -ne 0 ] ; then
        echo "frt/frtpx error" > /nwj/X01S767/nwj_qsuberr
        exit 2
    fi
    # link
    f77nwt  -o /nwj/X01S767/X01.A.LOAD/MAIN_nb  *.o \
           >> /nwj/X01S767/jesf_X01S767_01  2>&1
    rtncode=$?
    if [ $rtncode -ne 0 ] ; then
        rm *.o
        echo "lied error" > /nwj/X01S767/nwj_qsuberr
        exit 2
    fi
    rm *.o
    # execution
#   /usr/NWT/bin/nwjqsub <<"!NWJ_NWT01"
    qsub <<"!NWJ_NWT01"
##
#@$-s /bin/sh
#@$-r X01S767
#@$-q XXXX
#@$-lPv 001
#@$-lt 18000
#@$-o /nwj/X01S767/acct_X01S767_02
#@$-g x
#@$-eo
#@$
#!/bin/sh
    cd /nwj/X01S767
    if [ -f /nwj/X01S767/nwj_qsuberr ] ; then
        exit
    fi
    fu01=/nwj/X01S767/X01.A.INPUT_nb; export fu01
    fu02=/nwj/X01S767/X01.A.OUTPUT/A0_nn; export fu02
         /nwj/X01S767/X01.A.LOAD/MAIN_nb \
       > /nwj/X01S767/jesf_X01S767_02  2>&1
    rtncode=$?
    if [ $rtncode -ne 0 ] ; then
        echo "go error" > /nwj/X01S767/nwj_qsuberr
        exit 2
    fi
!NWJ_NWT01
if [ $? -ne 0 ] ; then
    echo "qsub error" > /nwj/X01S767/nwj_qsuberr
    exit 2
fi
!NWJ_FEP
```

図 5.3　中間シェルスクリプトからシェルスクリプトへの変換例

This document is provided by JAXA.

ェーズである。MSPから転送されたNWTジョブを確認／検査し，バッチリクエストを生成するための受付処理を行う。また，UXP/M で実行したバッチリクエストの終了処理を行う。本処理フェーズの主な処理概要を以下に示す。

（1）シェルスクリプトの展開

中間シェルスクリプトをUNIXシェルスクリプトに自動展開し，同時に構文の確認／検査を行う。図 5.3 に中間シェルスクリプトからシェルスクリプトへの変換例を示す。同図に示されるとおり，NS カタプロや中間シェルスクリプトからは簡単に対応がとれないほど，全く異なる記述形式をもつUNIX本来のシェルスクリプト記述形式に変換される。同図において，第3.1節で示したMSPとUNIXにおけるジョブ記述形式の相違が顕著に現れている。この異種 OS 間におけるジョブ記述形式の相違を吸収するための第二段階の処理を本処理フェーズで実行する。この結果，NS カタプロは最終的な UNIX のシェルスクリプトの形式に生成される。UNIX システムにおいては，シェルスクリプトの記述に従ってバッチリクエストのプロセスが順次，実行していく。また，本処理フェーズではカタプロの記述内容にエラーがある場合には，その旨のメッセージをユーザごとのメッセージ・ファイルに出力する。さらに，UXP/M および NWT で標準入出力するデータファイルや自動呼び出しライブラリの定義ならびにシステムパラメータ等の設定を行う。

（2）バッチリクエストのスケジューリング処理

NWT で実行する数値シミュレーション実行処理を除く全てのバッチリクエストは VP の UXP/M で実行されるようにスケジューリングする。

（3）ファイル送／受信のセットアップ

バッチリクエストが入出力するファイルを MSP と送／受信するためのファイル転送をセットアップする。なお，転送ファイルは MSP と UXP/M で形式が異なるので，転送規約を設けてブロックデータとして転送する。

（4）バッチリクエストの転送

NWT で実行するバッチリクエストの数値シミュレーション実行処理を NWT に転送する。

5.4 翻訳処理

本処理フェーズは，NWT 用の並列 FORTRAN 言語で記述されたユーザプログラムの翻訳処理をFEPのUXP/Mで行う処理フェーズである。通常の翻訳処理ではソースプログラムファイルを入力し，実行結果として構文チェック，エラーメッセージならびにソースプログラムリスト等の標準印刷結果であるコンパイラ印刷出力ファイルとオブジェクトモジュールを出力する。

5. 5 結合・編集処理

本処理フェーズは，翻訳処理フェーズに続くユーザプログラムの結合・編集処理をFEPのUXP/M 上で行う処理フェーズである。通常の結合・編集処理では先行の翻訳処理フェーズが出力したオブジェクトモジュール，およびユーザライブラリならびにシステムライブラリ等を入力し，実行結果としてエラーメッセージならびにリンケージエディタのリスト等の標準印刷結果であるリンケージエディタ印刷出力ファイルとロードモジュールを出力する。

5. 6 NWT 受付／終了処理

本処理フェーズは，NWT で実行する数値シミュレーション実行処理のバッチリクエスト受付および終了処理を，NWT の構成要素である制御処理装置（CP）で実行する OS 処理フェーズである。本処理フェーズにおける主な処理概要を以下に示す。

（1）バッチリクエストのスケジューリング処理

実行待ちバッチリクエストをいかなる順序で，どの様にNWTで実行させるか，また，要求PE台数をいかに割り当てるか等のバッチリクエストのスケジューリング処理を行う。

（2）ファイルのステージング処理

NWT の数値シミュレーション実行処理時に入出力される大規模データを高速に処理するため，NWT では磁気ディスク装置に比べ格段に高いデータ転送性能を有するSSUを磁気ディスクファイルのキャッシュとして位置づけている。本処理フェーズでは，NWT ジョブの実行処理の直前に，入力データを磁気ディスク装置からSSUへ転送するプレステージング処理のスケジューリングと実行を行う。また，NWT実行時にSSUに出力されたデータを磁気ディスク装置に退避するディスステージング処理のスケジューリングを行う。

5. 7 数値シミュレーション実行処理

本処理フェーズは並列ベクトル演算の数値シミュレーション実行処理をNWTで処理するユーザプログラムの処理フェーズである。実行時には，SSU 上の各種入出力ファイルをアクセスするが，この入出力処理の中心となるのは書式なし順次入出力データのFORTRANレコードである。NWT と VP2600（VP および FEP）では書式なし FORTRAN レコードの内部表現形式が異なる。NWT の内部表現形式はIEEE形式であるが，VP2600ではIBM形式である。この内部表現形式の相違は実行時に FORTRAN システムのライブラリが吸収する。

6．ファイル転送実現方式の概要

実際のところ，如何に超高速な計算機システムの利用といえども，その処理の内容はファイルを入力し，結果としてファイルに出力するという，いわばファイリング処理にすぎないといっても過言ではない。MSPビュー・

This document is provided by JAXA.

NSUB NWT

MSP

NS NS 送信データ作成ジョブは,

M M MSP UXP/M

NWT UXP/M

ファイル受信デーモンは, UXP/

OS M UXP

NWT

2

NWT

M M MSP

3. 3.4

OS 後処理パッチリクエストは, UXP

6. 1 翻訳処理と結合･編集処理の実行時入出力ファイル転送処理

ファイル受信ジョブは, MSP

6.1

MSP UXP/M

UXP 6.2

NWT

UXP

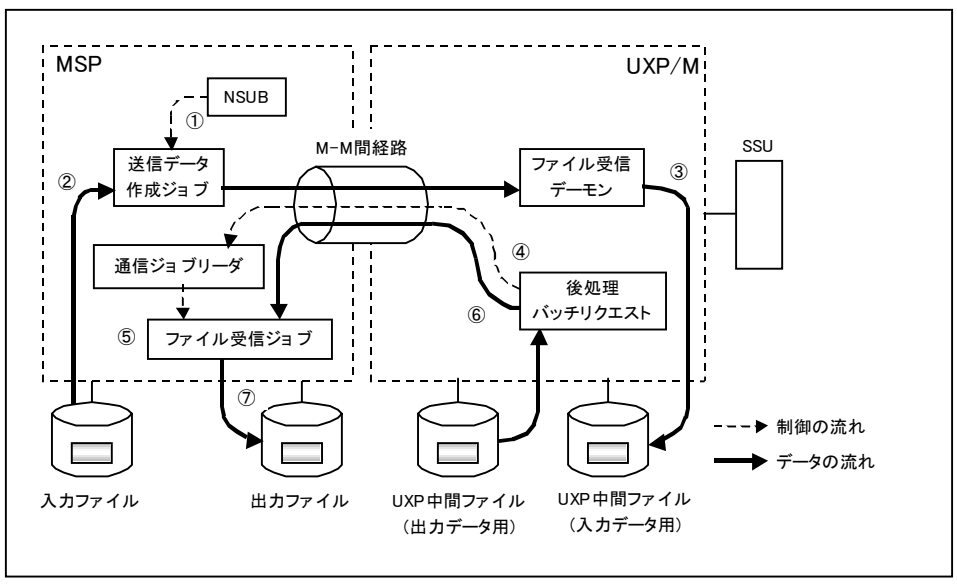
MSP

UXP MSP

6.1 SSU

SSU

1 MSP SSU



6.1

6. 2 数値シミュレーション実行時の入出力ファイル転送処理

This document is provided by JAXA.

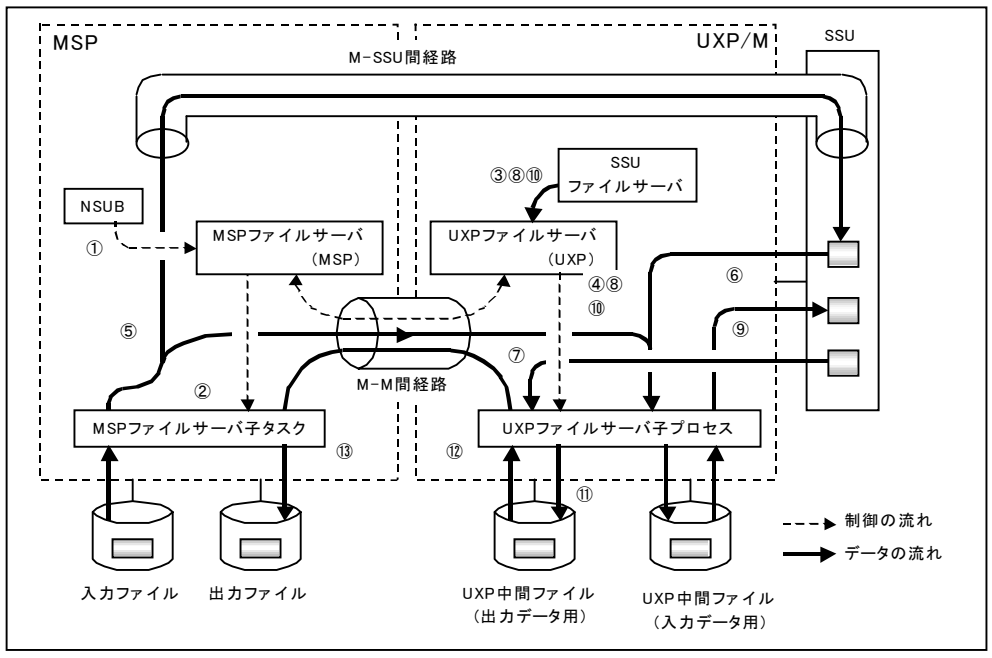


図 6.2　数値シミュレーション実行時入出力ファイルの転送方式

から退避するディスステージング処理に必要なファイル転送の実現方式の概要は以下のとおりである。

(1) ステージング処理

① NSUB コマンドにより NWT ジョブを受け付けると，MSPにおけるファイル転送処理を制御するMSPファイルサーバに通知する。

②MSPファイルサーバは，NWTジョブごとに通信経路を指定し，MSPファイルサーバ子タスクを生成する。

③ SSU におけるステージング処理を制御する SSU ファイルサーバよりステージング要求があると，UXPにおけるファイルの転送処理を制御する UXP ファイルサーバに通知される。

④ UXP ファイルサーバは，通信経路を指定して UXP ファイルサーバ子プロセスを起動する。UXPファイルサーバ子プロセスはMSP ファイルサーバ子タスクにデータ転送を要求する。

⑤ MSP ファイルサーバ子タスクは，入力ファイルを読み込み，（MSPメモリ－SSU間転送経路）を利用して，UXP/M配下のSSUに転送する。なお，M－SSU間経路については実運用のNWTジョブ処理の中では利用されていない。

⑥UXPファイルサーバ子プロセスは，⑤の処理により転送された SSU 上のデータを UXP 中間ファイル（入力データ用）に複写する。

⑦ SSU が使用できない状態になったら，MSP ファイルサーバ子タスクは，経路をM－M間経路に変更し，UXP ファイルサーバ子プロセスに直接データ転送する。このデータを，UXPファイルサーバ子プロセスは，⑥のデータに続けて格納する。

⑧SSUファイルサーバより入力要求があると，UXPファイルサーバは，UXPファイルサーバ子プロセスにデータ転送処理を依頼する。

⑨ UXP ファイルサーバ子プロセスは，UXP 中間ファイル（入力データ用）よりデータを読み込み，SSUに転送する。

（2）ディスステージング処理

⑩SSUファイルサーバより出力要求があると，UXPファイルサーバは，UXPファイルサーバ子プロセスにデータ転送処理を依頼する。

⑪ UXP ファイルサーバ子プロセスは，SSU よりデータを読み込み，UXP中間ファイル（出力データ用）に転送する。

⑫ UXP ファイルサーバ子プロセスは，UXP 中間ファイル（出力データ用）よりデータを読み込み，M－M間経路を利用して，MSPファイルサーバ子タスクにデータを転送する。

⑬ MSP ファイルサーバ子タスクは受信したデータを格納ファイルに出力する。

7. MSP ビュー・ユーザ・インターフェースの検証と考察

MSPビュー・ユーザ・インターフェースの諸機能の概要は第４章で示したとおりである。MSP ビュー・ユーザ・インターフェースには，異種 OS としての UNIX システム利用において，ユーザ利用性の継続とユーザ資産の継承という二つの目標実現を叶えるために多数の機能

This document is provided by JAXA.

```
//X01S*** NJOB C004                              ……①
//        EXEC NFORTC,PARAM='-Wp -Ps',           ……②
//        SF='X01.NWJTEST.FORT77',ELM='JST01'
//        EXEC NLIED,PARAM='-Wp'                 ……③
//        EXEC NGO,TIME=3400                     ……④
//                                               ……⑤
```

図 7.1 NWT ジョブのジョブストリーム記述例

が用意されており，それらの機能はNSカタプロおよびNSコマンドに集約されている。本章では，MSPビュー・ユーザ・インターフェースが所期の目標どおりに所有の機能を確実に遂行しているかどうかを検証するとともに，その有効性について考察する。なお，検証ジョブについては，FORTRAN プログラムの翻訳処理，結合・編集処理ならびに実行処理の 3 ジョブステップで構成される一般的なジョブの例である。

7. 1 NWT ジョブの投入

一般的なNWTジョブのジョブストリーム記述例を図7.1に示す。同図に示すとおり，ジョブストリーム記述法は今まで慣れ親しんできた従来形式のNSカタプロ形式である。すなわち，MSPのユーザインターフェースと等価であり，MSPで実行処理するジョブと同様にNSUBコマンドを入力すると，あたかもMSPへジョブを投入するかのごとくNWTへのジョブ投入が行える。このことから，第4章に示したMSPビュー・ユーザ・インターフェースの機能要件（1）が実現しているといえる。図7.1のNSカタプロで指示する内容は以下のとおりである。

①のNJOB文はNWTジョブの先頭を意味する制御文である。要素計算機を4台使用する並列ジョブのジョブクラス（C004）を利用することを意味している。

②のNFORTC文はFEPでFORTRAN翻訳処理を指示する制御文である。PARAMはFORTRANコンパイラに渡すオプションを指定するパラメータである。なお，オプションの形式はFEPのコンパイラオプション形式を指定する。SFは翻訳処理するソースプログラムが格納されているMSP上のファイル名を指定するパラメータである。また，ELMは区分編成ファイルのメンバ名を指定するパラメータである。なお，区分編成ファイルはMSPが有するファイル編成の種類の一つであり，多数のメンバを含むことができる。

③のNLIED文はFEPで結合・編集処理を指示する制御文である。PARAMはリンケージエディタに渡すオプション形式を指定するパラメータである。なお，オプションはFEPのリンケージエディタオプション形式を指定

```
①  NWJSNDSB: X01S001 MSP->UXP TRANSFER COMPLETE:MSP
②  NWX101I 93/ 2/ 3 15:46:55 X01S001.@@/02,1041.fepuxp,afterq,S,ACCEPT:UXP
③  NWX101I 93/ 2/ 3 15:47:13 X01S001.01/02,1043.fepuxp,C004,L,ACCEPT:UXP
④  NWX201I 93/ 2/ 3 15:47:21 X01S001.01/02,1043.fepuxp,C004,START:UXP
⑤  X01S001: nfortc(01) return value is 0.:UXP
⑥  X01S001: nlied(ld) return value is 0.:UXP
⑦  X01S001: Size of a.out -> 419936 + 69376 + 4328 = 493640:UXP
⑧  NWX301I 93/ 2/ 3 15:49:37 X01S001.01/02,1043,fepuxp,END,91/100:UXP
⑨  NWX101I 93/ 2/ 3 15:51:38 X01S001.02/02,1045,fepuxp,C004,L,ACCEPT:NWT
⑩  NWX201I 93/ 2/ 3 15:51:52 X01S001.02/02,1045,fepuxp,C004,START:NWT
⑪  X01S001: ngo(01)  return value is 0.:NWT
⑫  NWX301I 93/ 2/ 3 16:52:30 X01S001.02.02,1045,fepuxp,END,1409600/100:NWT
⑬  NWX201I 93/ 2/ 3 16:52:40 X01S001.@@/02,1041,fepuxp,aftermost,START:UXP
⑭  Request 1041.fepuxp submitted to queue: afterq.:UXP
⑮  Request 1043.fepuxp submitted to queue: C004.:UXP
⑯  you have mail:UXP
⑰  Request 1045.fepuxp submitted to queue: NC004.:UXP
⑱  NWX302I 93/ 2/ 3 16:53:16 X01S001,JOBEND:UXP
⑲  NWJRCVSB JOB=S FILE TRANSFER COMPLETE :MSP
```

図 7.2 NWT ジョブの LOG 情報

This document is provided by JAXA.

する。

④ NGO 文は NWT における数値シミュレーション実行処理を指示する制御文である。TIMEはCPU打ち切り時間（単位：秒）を指定するパラメータである。

⑤はジョブの区切りを　　パラメータを有しない空文である。

7. 2　NWT ジョブ実行経過状況の確認

NWT ジョブは処理の経過とともに，その実行状況を示す各種のログ情報を，ジョブ投入元である MSP のセッションに時々刻々，通知する。図7.1で例示したNWTジョブのログ情報の通知内容を図 7.2 に示す。また，図7.2の各メッセージの意味を図7.3に示す。これらの図に示すとおり，UNIX システムである FEP と NWT におけるジョブの実行処理状況が，MSP上のジョブの所有者であるユーザのセッションで確認できる。また，図 7.2 からユーザが指示するとおり，NWT ジョブを構成するジョブステップが正確な実行順序で実行されていく様子が確認できる。このことから，第 4 章に示した MSP ビュー・ユーザ・インターフェースの機能要件 (2) が実現しているといえる。

7. 3　NWT ジョブ実行結果の検索

NWT ジョブでは，ほとんどのジョブが印刷イメージの実行結果をファイルに標準出力する。かなり前の大型電子計算機システムでは，ジョブの実行結果というとプ

```
①  NWJSNDSB: X01S001 MSP->UXP TRANSFER COMPLETE:MSP
   MSPのソースプログラムおよび実行時標準入力ファイル等のFEPへの転送が正常に終了した。
②  NWX101I 93/ 2/ 3 15:46:55 X01S001.@@/02,1041.fepuxp,afterq,S,ACCEPT:UXP
   FEPへ後処理用のジョブがジョブ待ちキューに受付られた。
③  NWX101I 93/ 2/ 3 15:47:13 X01S001.01/02,1043.fepuxp,C004,L,ACCEPT:UXP
   FEPへジョブがクラスC004でジョブ待ちキューに受付られた。
④  NWX201I 93/ 2/ 3 15:47:21 X01S001.01/02,1043.fepuxp,C004,START:UXP
   翻訳処理および結合・編集処理を開始した。
⑤  X01S001: nfortc(01) return value is 0.:UXP
   FEPで翻訳処理は正常終了した（ value is 0 ）。
⑥  X01S001: nlied(ld) return value is 0.:UXP
   結合・編集処理は正常終了した（ value is 0 ）。
⑦  X01S001: Size of a.out -> 419936 + 69376 + 4328 = 493640:UXP
   実行時のサイズ（ text + data + bss )を表示している（単位:バイト）。
    bss: 非初期化データサイズ（僅少なｼｽﾃﾑﾗｲﾌﾞﾗﾘｻｲｽﾞを含む）
⑧  NWX301I 93/ 2/ 3 15:49:37 X01S001.01/02,1043,fepuxp,END,91/100:UXP
   FEPでの翻訳処理および結合・編集処理が終了した（ CPU 91/100秒 ）。
⑨  NWX101I 93/ 2/ 3 15:51:38 X01S001.02/02,1045,fepuxp,C004,L,ACCEPT:NWT
   NWTへジョブがクラスC004でジョブ待ちキューに受付られた。
⑩  NWX201I 93/ 2/ 3 15:51:52 X01S001.02/02,1045,fepuxp,C004,START:NWT
   NWTで実行処理を開始した。
⑪  X01S001: ngo(01) return value is 0.:NWT
   実行は正常終了した（ value is 0 ）。
⑫  NWX301I 93/ 2/ 3 16:52:30 X01S001.02/02,1045,fepuxp,END,1409600/100:NWT
   NWTでの実行処理が終了した（ CPU 6/100秒 ）。
⑬  NWX201I 93/ 2/ 3 16:52:40 X01S001.@@/02,1041,fepuxp,aftermost,START:UXP
   FEPで後処理が開始した。
⑭  Request 1041.fepuxp submitted to queue: afterq.:UXP
   FEPへ後処理用ジョブの投入を行った。
⑮  Request 1043.fepuxp submitted to queue: C004.:UXP
   FEPへジョブの投入を行った。
⑯  you have mail:UXP
   FEPよりMAILが届いた。
⑰  Request 1045.fepuxp submitted to queue: NC004.:UXP
   FEPへジョブの投入を行った。
⑱  NWX302I 93/ 2/ 3 16:53:16 X01S001,JOBEND:UXP
   NWTにおけるジョブが全て終了した。
⑲  NWJRCVSB JOB=S FILE TRANSFER COMPLETE :MSP
   NWTで作成されたプログラムリストおよび実行結果等のMSPへの転送が正常に終了した。
```

図 7.3　図 7.2 のメッセージの意味

This document is provided by JAXA.

リンタ用紙の出力を指した。事実，プリント出力は一番重要であった。現在では，本来の実行結果はファイルに出力し，印刷イメージの結果としては計算処理上の各種パラメータ，有用なメッセージや処理の指標となる少量のメッセージを出力することが一般的である。また，印刷イメージ結果の内容もファイルの中で検索することが通常となり，実際のプリンタ用紙に出力することは稀になった。この検索処理を実現するMSPビュー・ユーザ・インターフェースとして，MSPの「PFD-OUTLISTユーティリティ」において「NWTジョブ実行結果検索メニュー」機能を追加した。図7.4はNWTジョブの各ユーザプログラム処理フェーズで出力された印刷イメージの実行結果をジョブとして統合し，検索可能にする「PFD-OUTLISTユーティリティ」のメニュー表示画面である。このユーティリティでは，ジョブステップごとに出力された印刷イメージの実行結果を検索することができ，検索処理の中から必要となった実行結果をプリンタ用紙や指定ファイル等に出力起動できるとともに，不要となったファイルを削除することも指示できる。以上より，印刷結果検索処理についても，あたかもMSPで実行したジョブと同様にMSPのセッションでNWTジョブの検索処理やキャンセルが行える。このことから，第4章に示したMSPビュー・ユーザ・インターフェースの機能要件（3）が実現しているといえる。

7．4　実行待ちジョブおよび実行中ジョブ状況の確認

ユーザがNWTジョブの処理を効率的に進める上で，また，プログラムの生産性をより高めるためにもジョブ投入から終了までに要する経過時間，すなわちターン・アラウンド・タイムの予測は重要である。MSPのセッションからWAITJOBコマンドを投入すると，図7.5に示すNWTジョブの実行待ちジョブ状況の表示情報が得られる。同図から，実行待ちしているジョブについて，属するキュー名，実行順序，バッチリクエスト名およびID，

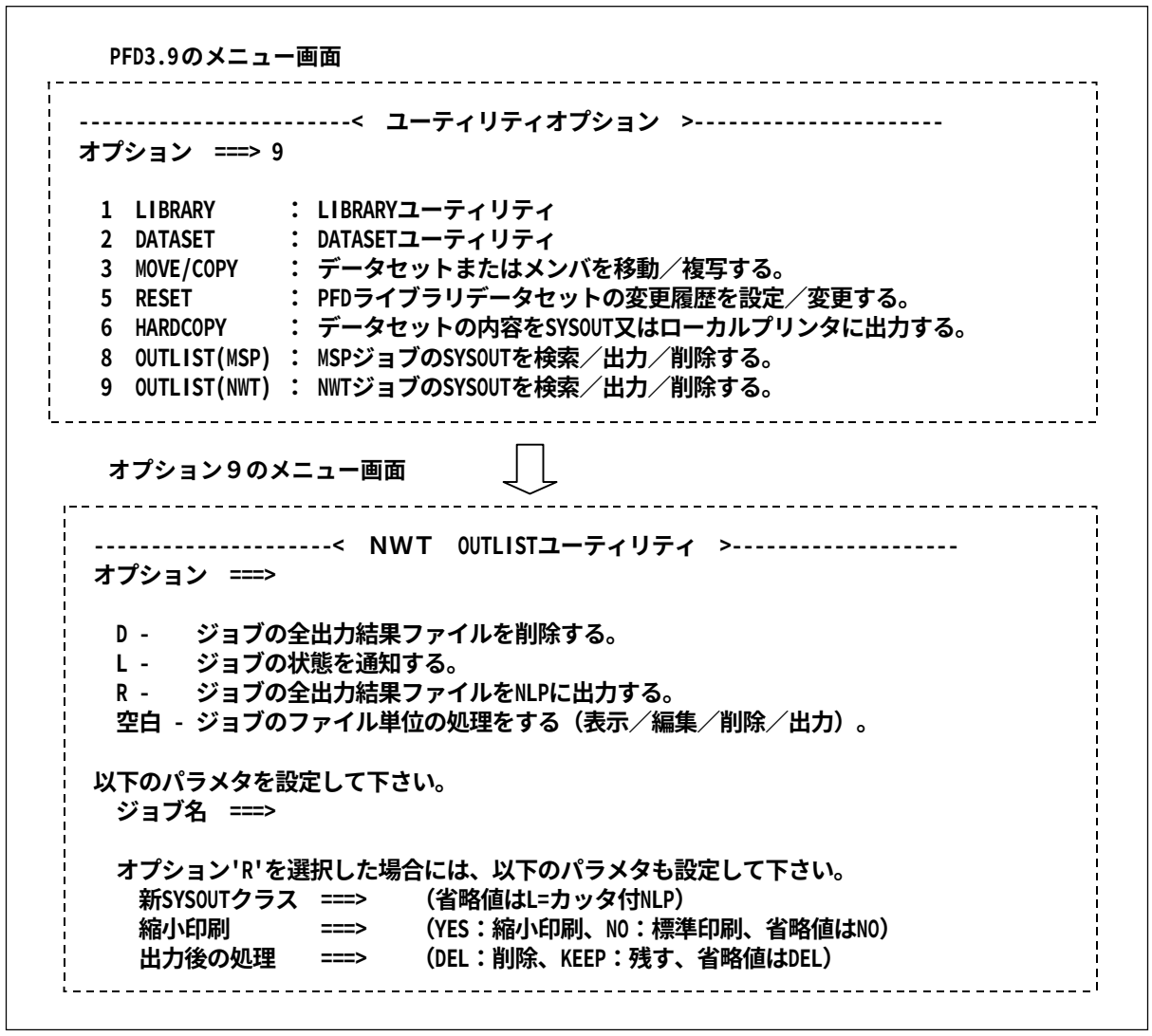


図7.4　ＮＷＴジョブ実行結果検索ユーティリティのメニュー表示画面

This document is provided by JAXA.

```
ｷｭｰ 順番    ﾘｸｴｽﾄ名         ﾘｸｴｽﾄID       要求    受付時間    優先度
 Q   No.  REQUEST-NAME    REQUEST-ID   PE   CPU ACCEPT-TIME PRIORITY  STATUS
 u0   1. a12+15GTO.01/01 97020.aoi     16   600 15-16:29:27        0 pe-busy
      2. l32+12MBO.08/11 96101.aoi     48 20000 12-22:12:09        0 next-step
         l32+12MBO.09/11 96102.aoi     48 20000 12-22:12:12        0 next-step
         l32+12MBO.10/11 96103.aoi     48 20000 12-22:12:15        0 next-step
         l32+12MBO.11/11 96104.aoi     48 20000 12-22:12:19        0 next-step
      3. l19-J0749.06/07 96353.aoi     30 18000 13-16:43:18        0 next-step
         l19-J0749.07/07 96354.aoi     30 18000 13-16:43:21        0 next-step
      4. a12+14Abj.04/05 96474.aoi     16 15000 14-10:38:04        0 next-step
         a12+14Abj.05/05 96475.aoi     16 15000 14-10:38:08        0 next-step
                    .                   .                .
                    .                   .                .
                    .                   .                .
                    .                   .                .
                    .                   .                .

     60. p41+15F3p.01/01 96949.aoi      1  6400 15-15:03:54        0 resv-time
     61. p41+15F44.01/01 96950.aoi      1  6400 15-15:04:07        0 resv-time
     62. a76+15C3l.01/01 96874.aoi      1 18000 15-12:04:20        0 resv-time
     63. p73+15Eim.02/02 96929.aoi      1 18000 15-14:44:55        0 resv-time
 l0   1. n91+12IL5.08/09 96012.aoi      1 18000 12-18:21:54        0 next-step
         n91+12IL5.09/09 96013.aoi      1 18000 12-18:21:57        0 next-step
      2. n92+12IMk.08/09 96021.aoi      1 18000 12-18:23:36        0 next-step
         n92+12IMk.09/09 96022.aoi      1 18000 12-18:23:40        0 next-step
      3. n92+12INR.08/09 96030.aoi      1 18000 12-18:24:17        0 next-step
         n92+12INR.09/09 96031.aoi      1 18000 12-18:24:38        0 next-step
 -----
 INFO: 12pe reserved for j06+14IUT.03/10.
 INFO: The job will be started at 16:38:22 restricted to 12000sec-cpu time.
 INFO: But job(pe<=1 & cpu<360sec) is excutable now.
 -----
 Q   NO.  REQUEST-NAME    REQUEST-ID    STATUS
 uc   1. j65+15GUA.@X/00 97021.aoi     RUNNING
      2. p51+15GW8.@X/01 97022.aoi     RUNNING
 INFO: AOI REQUEST(routing) NOT FOUND.
```

図 7.5　実行待ち NWT ジョブの情報

要求 PE 台数，要求 CPU 使用時間，バッチリクエスト受付時間，優先度ならびにバッチリクエストの状態等が確認できる。なお，連続する実行待ちジョブの表示に続けて次回ジョブスケジューリング契機に実行起動を予定しているジョブについて，要求 PE 台数と実行開始予定時刻等を表示する。

同様に ACTJOB コマンドを投入すると，図 7.6 に示す NWT 実行中ジョブ状況が確認できる。同図に示すとおり，NWT で実行しているジョブについて，キュー名、バッチリクエスト名および ID，割当 PE 台数，実使用 PE 台数，要求 CPU 使用時間，ジョブ経過時間等を表示する。なお，最下行の表示は，PE 台数の情報について，運用台数，ONLINE／OFFLINE 台数，実行／非実行台数，および 1 GB のジョブ用に割り当てている PE 台数を表示している。ユーザは以上の情報をもとに，MSP ビュー・ユーザ・インターフェースを介し，新たなジョブ投入，実行結果の検索ならびにジョブキャンセル等のジョブ操作を，あたかも UNIX システムのセッションで行うかのように，MSP のセッションにおいて操作することができる。上記より，第 4 章に示した MSP ビュー・ユーザ・インターフェースの機能要件（4）を実現しているといえる。

7. 5　考察

以上に述べたとおり，MSP ビュー・ユーザ・インターフェースは，UNIX システムである NWT の利用を従来システムの MSP の利用と等価にし得た。また，MSP ビュー・ユーザ・インターフェースの開発により NWT ジョブ処理において得られた利点を列挙すると以下のとおりである。

（1）通常，OS の異なる計算機システムへの移行に際し，ジョブの投入手続きを規定するジョブ制御文は移行せざるを得ない。このことは今まで慣れ親しんできたジ

This document is provided by JAXA.

```
QUE   REQUEST_NAME   REQUEST_ID  STATUS ALLOC USE    REQUEST_CPU     TIME
jq01 a12+14ARn.03/03 96469.aoi      RUN    16  16  15000( 4:10:00) 1:10:53
jq01 a31+149io.02/05 96436.aoi      RUN    16  16  17500( 4:51:40) 1:06:01
jq01 a73+15EnL.01/01 96935.aoi      RUN    32  32    600( 0:10:00) 0:01:23
jq01 l19-J0749.05/07 96352.aoi      RUN    30  30  18000( 5:00:00) 1:12:08
jq01 l32+12MBO.07/11 96100.aoi      RUN    48  48  20000( 5:33:20) 1:07:06
jq01 l63+15Fu0.01/01 96999.aoi      RUN     1   1    600( 0:10:00) 0:01:21
jq01 l92+14BnX.02/04 96505.aoi      RUN    18  18  18000( 5:00:00) 1:06:52
jq02 n91+12IL5.07/09 96011.aoi      RUN     1   0  18000( 5:00:00) 1:30:16
jq02 n92+12IMk.07/09 96020.aoi      RUN     1   0  18000( 5:00:00) 1:30:14
jq02 n92+12INR.07/09 96029.aoi      RUN     1   0  18000( 5:00:00) 1:30:12
jq02 n93+14BRd.02/02 96498.aoi      RUN     1   0  18000( 5:00:00) 1:30:12

TOTAL_PE   ONLINE  OFFLINE   RUNNING[OFFLINE]   IDLING[ONLINE]  1G[ONLINE]
    166      166        0       165       0         1      1     4      4
```

図 7.6　実行中 NWT ジョブの情報

ョブストリーム記述法に代えて，新たなジョブストリーム記述法の習得を強いることを意味する。MSP ビュー・ユーザ・インターフェースの開発により，新たなジョブストリーム記述法の習得を不要にし得た。

(2) MSPの長年運用により，ユーザ資産としての磁気ディスクファイルは膨大に蓄積されている。ユーザはジョブ処理に必要なリソースであるファイルを新しいシステムに移行するためには，先ず，ファイルのバックアップに始まり，レコード形式の変換，コード変換ならびに新磁気ディスク装置への格納処理等，非常に煩雑で膨大なシステム移行作業を強いられることになる。しかし，MSP ビュー・ユーザ・インターフェースの開発により，このシステム移行処理に伴うユーザ負担を最小限にとどめることができた。

(3) 新 UNIX システムの利用，すなわち，プログラムの作成・編集方式ならびに各種のコマンド利用等，新たなシステム利用方式を習得するためには根気のいる長大な時間を消費する必要がある。しかし，MSP ビュー・ユーザ・インターフェースを利用することにより，これらのユーザ負担を激減することができた。

8．おわりに

航技研計算機システムの運用において，MSPの利用は非常に長期間継続している。この間に航技研独自のシステム運用機能の開発を多く積み重ね，ユーザのシステム利用性を一層向上させてきた。MSP ビュー・ユーザ・インターフェースの開発により，このシステム利用環境をハードウェア，OS ならびに FORTRAN コンパイラまでも更新された NWT の運用にも活かすことができた。

なお，航技研における NWT 用に開発された本 MSP ビュー・ユーザ・インターフェースの有効性が確認され，富士通（株）は異種オペレーティングシステム連携機能「FUJITSU M-VPP/CF」を MSP に標準インストール可能なソフトウェアとして製品化した。

平成８年１月の FEP の更新に際し，MSP ビュー・ユーザ・インターフェースは多少の変更が加えられ，現システム構成の NWT の運用においても継続利用されている。このとき，本稿で示した MSP ビュー・ユーザ・インターフェースを実現する OS 処理フェーズについては，標準ソフトウェア「FUJITSU M-VPP/CF」の機能に置き換えられた。また，このような MSP と UNIX ベースの異種OSを連携するシステム機能の有用性については，他の大型電子計算機センタ等においてもニーズがあり，「FUJITSU M-VPP/CF」は広く利用されていることから明らかである。

おわりに当たり，NWT 用 MSP ビュー・ユーザ・インターフェースの開発に必要な資料の提出を頂いた富士通（株）に対し，特に多大なご協力と多くの討論を頂いた森重博司氏，山口靖氏，矢澤克己氏，軽部行洋氏，藤田信英氏に対して，末筆ながら感謝の意を表する。

参考文献

1) 三好，吉岡，他：“数値風洞のハードウェ ア”第 9 回航空機計算空気力学シンポジウム論文集，SP-16，1991 年 12 月
2) 福田，末松，他：“数値風洞のオペレーティングシステム”，第 9 回航空機計算空気力学シンポジウム論文集，SP-16，1991 年 12 月
3) FACOM UXP/M 使用手引き書（システム管理者編）
4) FACOM UXP/M 使用手引き書（ユーザ編）
5) FACOM OS Ⅳ /F4 MSP 制御プログラム解説書
6) FACOM OS Ⅳ /MSP ジョブ制御言語文法書

This document is provided by JAXA.

7) 土屋，末松，畑山：“次期計算機システム用ジョブ制御マクロの設計”，航技研資料，TM-444
8) FACOM OS Ⅳ /MSP TSS／Eコマンド文法書
9) 畑山，土屋，末松，他：“NSシステム利用の手引き”，航技研技術資料，N-41
10) FACOM UXP/M VPP FORTRAN77 EX/VP 使用手引き書
11) FACOM OS Ⅳ /MSP FORTRAN77 EX 使用手引き書

This document is provided by JAXA.

表 4.1　NJOB 文の記述形式

```
//ジョブ名　NJOB　ジョブクラス　[ ,NOTIFY=ユーザ名 ]
```

パラメータ	省略値	パラメータの説明
ジョブ名	―――	ユーザ名＋英数字からなる 7 文字以内の文字列を指定する。
ジョブクラス	―――	NWT 用ジョブクラスを指定する。
NOTIFY	―――	実行終了メッセージを指定 TSS 端末ユーザに通知する。

＜特記事項＞

(1) ジョブクラスには，U001～U162(256MB 用)，L001～L004 (1GB 用)の範囲で指定できる。

＜記述例＞

(1) ユーザ名 X01 で，256MB 用 を 4 台使用のジョブクラス U004 を使用する。

```
//X01S001  NJOB  U004,NOTIFY=X01
```

＜ジョブ使用例＞

(a) FORTRAN プログラムを翻訳／結合／編集を行い，実行する。

```
//X01S001   NJOB  U001
//  EXEC    NFORTC,SF='X01.TEST10.FORT77',SINC='X01.TEST10.INCLUDE'
//  EXEC    NLIED
//  EXEC    NGO,TIME=60
//
```

(b) FORTRAN プログラムを翻訳／結合／編集を行い，ロードモジュールを作成する。

```
//X01S002   NJOB  U001
//  EXEC    NFORTC,PARAM='DOUBLE',SF='X01.TEST20.FORT77',ELM='*'
//  EXEC    NLIED,@OEF='/home/x/x01/test20.load'
//
```

(c) ロードモジュールを使用する。

```
//X01S003   NJOB  U001
//  EXEC    NGO,TIME=240,@EF='/home/x/x01/test30.load'
//
```

(d) 入出力ファイルを使用する。

```
//X01S003   NJOB  U001
//  EXEC    NGO,PARAM='-W1,-C10,-C20',@EF='/home/x/x01/test30.load'
//  EXPAND  NUSDKR,RNO=10,FILE='X01.TEST30.DATA10',TYPE=T
//  EXPAND  NUSDKR,RNO=20,FILE='X01.TEST30.DATA20',TYPE=T
//  EXPAND  NUSDKW,RNO=30,@FILE='/large/x/x01/test30.data30'
//  EXPAND  NUSDKW,RNO=40,@FILE='/large/x/x01/test30.data40'
//
```

(e) カードイメージデータを使用する。

```
//X01S004   NJOB  U008
//  EXEC    NFORTC,SF='X01.TEST40.FORT77',ELM='PARALLEL'
//  EXEC    NLIED
//  EXEC    NGO,TIME=300
 &NAME40
     A=40.1, B=40.2, C=40.3, D=40.4
 &END
//  EXPAND  NUSDKR,RNO=10,FILE='X01.TEST40.DATA10',TYPE=B
//  EXPAND  NUSDKW,RNO=20,FILE='X01.TEST40.DATA20',TYPE=B
//
```

This document is provided by JAXA.

表 4.1　NJOB 文の記述形式 (続き)

(f) 図形データをファイルに作成する。

```
//X01S003   NJOB  U001
//  EXEC    NFORTC,SF='X01.TEST50.FORT77'
//  EXEC    NLIED
//  EXEC    NGO,TIME=180
//  EXPAND  NUSDKRW,RNO=10,FILE='X01.TEST50.DATA10',TYPE=B
//  EXPAND  NXY
//
```

表 4.2　NFORTC 文の記述形式

```
//  EXEC NFORTC  [ ,PARAM='コンパイラオプション' ]
                 [ ,SF='MSP ファイル名[(メンバ名)]' ]  [ ,ELM='メンバ名' ]
                 [ ,SINC='MSP ファイル名' ]
```

パラメータ	省略値	パラメータの説明
PARAM	システム標準値	MSP 形式のコンパイラオプションを指定する。
SF	―――	MSP ファイルのソースプログラムを指定する。 ファイルは，SF，SF1，･･･，SF10 の 11 個まで指定できる。
ELM	―――	SF に区分ファイルを指定した場合のメンバ名を指定する。
SINC	―――	INCLUDE 文のメンバが含まれる MSP ファイルを指定する。 ファイルは，SINC，SINC1，･･･，SINC4 の５個まで指定できる。

＜特記事項＞

(1) 本カタプロ文は，複数指定できない。

(2) ソースプログラムは，NWT/FORTRAN77/VP または NWT/FORTRAN77/VPP しか記述できない。

(3) MSP 形式のコンパイラオプションには，並列処理用オプションが存在しない。ジョブクラスに，
U001 または L001 を指定した場合には逐次処理，それ以外が指定された場合には並列処理とする。

(4) ELM には，１メンバまたは全メンバの『＊』しか指定できない。

＜記述例＞

(1) コンパイラオプションを指定して翻訳する。

```
//  EXEC  FORTC,PARAM='DOUBLE',SF='X01.TEST10.FORT77'
```

(2) インクルードとともに翻訳する。

```
//  EXEC  FORTC,SF='X01.TEST20.FORT77',SINC='X01.TEST20.INCLUDE'
```

(3) ２つの区分ファイルの全メンバを翻訳する。

```
//  EXEC  FORTC,SF='X01.TEST30.FORT77',SF1='X01.TEST31.FORT77',ELM='*'
```

This document is provided by JAXA.

表 4.3　NLIED 文の記述形式

```
//  EXEC  NLIED  [ ,@OEF='UXP ファイル名' ]
```

パラメータ	省略値	パラメータの説明
@OEF	―――	UXP ファイルにロードモジュールを保存する場合に指定する。

＜特記事項＞
(1) 本カタプロ文は，複数指定できない。
(2) UXP ファイルを指定する場合に，相対パスはホームディレクトリを示す。
＜記述例＞
(1) FORTRAN プログラムの結合／編集を行い，ロードモジュールを保存する。

```
//  EXEC  NLIED,@OEF='/home/x/x01/test10.load'
```

表 4.4　NGO 文の記述形式

```
//  EXEC  NGO  [ ,PARAM='実行時オプション' ]  [ ,TIME=分 ]  [ ,@EF='UXP ファイル名' ]
                [ { ,@DF='UXP ファイル名'                    } ]
                  { ,DF='MSP ファイル名[(メンバ名)]'         }
```

パラメータ	省略値	パラメータの説明
PARAM	システム標準値	UXP 形式の実行時オプションを指定する。
TIME	10	実行打ち切り時間を分単位で指定する。
@EF	―――	XP ファイルのロードモジュールを指定する。省略時には，NLIED で作成さ U れたロードモジュールを使用する。
@DF	―――	カードイメージで入力する NWT ファイルを指定する。
DF	―――	カードイメージで入力する MSP ファイルを指定する。

＜特記事項＞
(1) 実行時オプションには『 -Wl 』を必ず指定し，続けてオプションをカンマで区切って指定する。
(2) 実行打ち切り時間は，最大で 300 分 を指定できる。
(3) 浮動小数点の内部表現において，M形式のデータを入出力する際に，実行時オプションに変換対象の
ファイル識別番号を指定する。
・全ファイルの場合　　：　-C
・特定ファイルの場合　：　-Cnn,-Cnn,･･･
＜記述例＞
(1) 実行時間を 180 分 指定して実行する。

```
//  EXEC  NGO,TIME=180
```

(2) ロードモジュールを使用して実行する。

```
//  EXEC  NGO,@EF='/home/x/x01/test20.load'
```

(3) ファイル識別番号 30，31 をデータ変換しながら実行する。

```
//  EXEC  NGO,PARAM='-Wl,-C30,-C31'
```

This document is provided by JAXA.

表 4.5　NUSDKR 文の記述形式

```
//  EXPAND  NUSDKR,RNO=ファイル識別番号  [ ,DUMMY=ON ]
               ┌,@FILE ='UXP ファイル名'            ┐          ┌ BINARY ┐
              [┤                                     ├ ]  [ ,TYPE=┤        ├ ]
               └, FILE='MSP ファイル名[(メンバ名)]' ┘          └  TEXT  ┘
```

パラメータ	省略値	パラメータの説明
RNO	―――	プログラム内入出力文のファイル識別番号を指定する。
DUMMY	―――	実際の入出力動作が不要である場合に指定する。
@FILE	―――	使用する UXP ファイル名を指定する。
FILE	―――	使用する MSP ファイル名を指定する。
TYPE	BINARY	MSP ファイルを指定した場合の変換を指示する。 ・BINARY ： ファイルがバイナリ形式の場合に，フォーマット変換を指示する（ 省略形は B ）。 ・TEXT ： ファイルがテキスト形式の場合に，コード変換とフォーマット変換を指示する（ 省略形は T ）。

＜特記事項＞
(1) 本文で指示するファイルは既存ファイルに限る。
＜記述例＞
(1) 実行時に MSP ファイルを参照する。

```
//  EXPAND  NUSDKR,RNO=10,FILE='X01.TEST10.DATA',TYPE=B
```

表 4.6　NUSDKRW 文の記述形式

```
//  EXPAND  NUSDKRW,RNO=ファイル識別番号  [ ,DUMMY=ON ]
               ┌,@FILE ='UXP ファイル名'            ┐          ┌ BINARY ┐
              [┤                                     ├ ]  [ ,TYPE=┤        ├ ]
               └, FILE='MSP ファイル名[(メンバ名)]' ┘          └  TEXT  ┘
```

パラメータ	省略値	パラメータの説明
RNO	―――	プログラム内入出力文のファイル識別番号を指定する。
DUMMY	―――	実際の入出力動作が不要である場合に指定する。
@FILE	―――	使用する UXP ファイル名を指定する。
FILE	―――	使用する MSP ファイル名を指定する。
TYPE	BINARY	MSP ファイルを指定した場合の変換を指示する。 ・BINARY ： ファイルがバイナリ形式の場合に，フォーマット変換を指示する（ 省略形は B ）。 ・TEXT ： ファイルがテキスト形式の場合に，コード変換とフォーマット変換を指示する（ 省略形は T ）。

＜特記事項＞
(1) 本文で指示するファイルは既存ファイルに限る。
＜記述例＞
(1) 実行時に MSP ファイルを参照・更新する。

```
//  EXPAND  NUSDKRW,RNO=10,FILE='X01.TEST10.DATA',TYPE=B
```

This document is provided by JAXA.

表 4.7　NUSDKW 文の記述形式

```
//  EXPAND  NUSDKW,RNO=ファイル識別番号  [ ,DUMMY=ON ]
          [ { ,@FILE ='UXP ファイル名'                          } ]  [ ,TYPE={ BINARY } ]
          [ { , FILE='MSP ファイル名[(メンバ名)]'               } ]  [ ,TYPE={ TEXT   } ]
          [ ,DISP={ NEW } ]   [ ,ORG={ PS } ]
          [ ,DISP={ OLD } ]   [ ,ORG={ PO } ]
          [ ,SPACE='([トラック数][,[増分][,ディレクトリブロック数]])' ]
          [ ,BSIZE=ブロック長, SIZE=レコード長, RECFM=レコード形式 ]
```

パラメータ		省略値	パラメータの説明
RNO		――――	プログラム内入出力文のファイル識別番号を指定する。
DUMMY		――――	実際の入出力動作が不要である場合に指定する。
@FILE		――――	使用する UXP ファイル名を指定する。
FILE		――――	使用する MSP ファイル名を指定する。
TYPE		BINARY	MSP ファイルを指定した場合の変換を指示する。 ・BINARY ： ファイルがバイナリ形式の場合に，フォーマット変換を指示する(省略形は B)。 ・TEXT　： ファイルがテキスト形式の場合に，コード変換とフォーマット変換を指示する（省略形は T)。
DISP		OLD	MSP ファイルの処置を指定する。
ORG		PS	MSP ファイルを新規作成する場合のファイル編成を指定する。
SPACE	PS	'(5,3)'	MSP ファイルを新規作成する場合の領域を指定する。
	PO	'(5,3,5)'	
BSIZE, RSIZE, RECFM		システム標準値	MSP ファイルを新規作成する場合のブロック長，レコード長およびレコード形式を指定する。

＜特記事項＞

(1) 本文で指示する UXP ファイルは既存ファイルに限る。

(2) DISP=OLD を指定し，さらに SPACE で増分値を指定しても増分は有効とはならない。

＜記述例＞

(1) 実行時に MSP ファイルを更新する。

```
//  EXPAND  NUSDKW,RNO=10,FILE='X01.TEST10.DATA',TYPE=B
```

This document is provided by JAXA.

表 4.8　NXY 文の記述形式

```
//  EXPAND  NXY  [ ,DUMMY=ON ]
```

パラメータ	省略値	パラメータの説明
DUMMY	―――	実際の入出力動作が不要である場合に指定する。

＜特記事項＞

(1) 本カタプロ文は，複数指定できない。

(2) 出力時の原点は，原点を基準にしてユーザが原点を定める必要がある。

(3) 出力はベクトルイメージで格納され，１ベクトル格納するのに約５バイトを必要とする。図形データは最大 160 トラック(約４MB)の容量があり，約 80 万ベクトルが格納できる。

(4) 図形データのファイルは，TSS でモニタ処理することができる。

表 4.9　XMTON 文の記述形式

```
//  EXEC  XMTON,IF='MSP ファイル名[(メンバ名)]',OF='MSP ファイル名'
                  [ ,TYPE={ BINALY [,RECFM=DIRECT] } ]  [ ,DISP={ NEW } ]
                          { TEXT                   }          { OLD }
                  [ ,SPACE='([トラック数][,増分])' ]
```

パラメータ	省略値	パラメータの説明
IF	―――	変換元の MSP ファイル名を指定する。
OF	―――	変換先の MSP ファイル名を指定する。
TYPE	BINARY	変換を指示する。 ・BINARY ： ファイルがバイナリ形式の場合に、フォーマット変換を指示する(省略形は B)。 直接編成ファイルの場合には，RECFM=DIRECT を指定する。 ・TEXT ： ファイルがテキスト形式の場合に，コード変換とフォーマット変換を指示する(省略形は T)。
DISP	OLD	変換先の MSP ファイルの処置を指定する。
SPACE	'(5,3)'	変換先の MSP ファイルを新規作成する場合の領域を指定する。

＜特記事項＞

(1) 変換先ファイルのファイル編成，レコード形式およびブロック長などについては，最適な DCB 属性を自動的に設定する。

＜記述例＞

(1) バイナリファイルを変換する。

```
//  EXEC  XMTON,IF='X01.TEST10.MSP',OF='X01.TEST10.NWT',TYPE=B
```

(2) テキストファイルを変換する。

```
//  EXEC  XMTON,IF='X01.TEST20.MSP(SOURCE)',OF='X01.TEST20.NWT',TYPE=T
```

This document is provided by JAXA.

表 4.10 XNTOM 文の記述形式

```
//  EXEC  XNTOM,IF='MSP ファイル名',OF='MSP ファイル名 [(メンバ名)]'
                    [ ,TYPE={ BINALY [,RECFM=DIRECT] } ] [ ,DISP={ NEW } ] [ ,ORG={ PS } ]
                            { TEXT                   }         { OLD }         { PO }
                    [ ,SPACE='([トラック数][,増分][,ディレクトリブロック数])' ]
```

パラメータ		省略値	パラメータの説明
IF		―――	変換元の MSP ファイル名を指定する。
OF		―――	変換先の MSP ファイル名を指定する。
TYPE		BINARY	変換を指示する。 ・BINARY ： ファイルがバイナリ形式の場合に，フォーマット変換を指示する(省略形は B)。 直接編成ファイルの場合には，RECFM=DIRECT を指定する。 ・TEXT ： ファイルがテキスト形式の場合に，コード変換とフォーマット変換を指示する(省略形は T)。
DISP		OLD	変換先の MSP ファイルの処置を指定する。
ORG		PS	変換先の MSP ファイルを新規作成する場合の編成を指定する。
SPACE	PS	'(5,3)'	変換先の MSP ファイルを新規作成する場合の領域を指定する。
	PO	'(5,3,5)'	

＜特記事項＞

(1) 変換先ファイルのファイル編成，レコード形式およびブロック長などについては，最適な DCB 属性を自動的に設定する。

＜記述例＞

(1) バイナリファイルを変換する。

```
//  EXEC  XNTOM,IF='X01.TEST10.NWT',OF='X01.TEST10.MSP',TYPE=B
```

(2) テキストファイルを変換する。

```
//  EXEC  XNTOM,IF='X01.TEST20.NWT',OF='X01.TEST20.MSP(SOURCE)',TYPE=T
```

(a) MSP 形式を NWT 形式にバイナリファイルを変換する。

```
//X01S001  NJOB HANJ,NOTIFY=X01
//  EXEC   XMTON,IF='X01.TEST10.NWT',OF='X01.TEST10.MSP',TYPE=B
//
```

(b) NWT 形式を MSP 形式にテキストファイルを変換して，ファイルを新規作成する。

```
//X01S002  NJOB HANJ,NOTIFY=X01
//  EXEC   XNTOM,IF='X01.TEST20.NWT',OF='X01.TEST20.MSP(SOURCE)',TYPE=T,
//           DISP=NEW,ORG=PO,SPACE='(20,10,5)'
```

This document is provided by JAXA.

表 4.11　NSUB コマンドの入力形式

コマンド名	オペランド
$NSUB	['MSP ファイル名[(メンバ名)]']

①機能概要

ジョブを起動する。

②オペランド

(1) MSP ファイル名[(メンバ名)]

ジョブストリームの格納されているファイル名を指定する。ＰＦＤ上のサブコマンドとして使用する場合にはファイル名を省略する。なお，区分ファイルの場合には，メンバ名を指定する。

③使用例

(1) ジョブを起動する。

```
$NSUB  'X01.TEST10.CNTL(PARALLEL)'
```

表 4.12　NFORT コマンドの入力形式

コマンド名	オペランド
$NFORT	['MSP ファイル名[(メンバ)]'] [コンパイラオプション]

①機能概要

FORTRAN プログラムの翻訳を行います。結果は，ファイル名 'uid.NFORT.LIST' に出力する。

②オペランド

(1) MSP ファイル名[(メンバ)]

プログラムが格納されているファイル名を指定する。なお，区分ファイルの場合は，メンバ名を省略すると全メンバを入力する。メンバ名を指定する場合には，１メンバ名のみを指定できる。

(2) コンパイラオプション

NWT 形式のコンパイラオプションを指定する。

③使用例

(1) 逐次 FORTRAN プログラムの翻訳を実行する。

```
$NFORT  'X01.TEST10.FORT77'
```

(2) 並列 FORTRAN プログラムの翻訳を実行する。

```
$NFORT  'X01.TEST20.FORT77(PARALLEL)'  -Wx
```

表 4.13　NCAN コマンドの入力形式

コマンド名	オペランド
$NCAN	ジョブ名

①機能概要

ジョブをキャンセルする。

②オペランドの説明

(1) ジョブ名

キャンセルするジョブ名を指定する。

③使用例

(1) ジョブをキャンセルする。

```
$NCAN  X01S001
```

This document is provided by JAXA.

表 4.14　NS コマンドの入力形式

コマンド名	オペランド
$NS	なし

①機能概要

ジョブの状況を表示する。

②表示例

```
KKA400I 10.54.36 ACTIVE NWT JOBS
ジョブ名   クラス   ステップ名    システム名     経過時間       ファイル転送
JOB NAME   CLS      STEPNAME      MSP FEP    NWT ELAPTIME    DDNAME   BYTE(KB)
X01S001    X        NGO           --         EX   00.02.21                  0
X01S002    X        NFORTC        --         EX   00.01.43                  0
```

表 4.15　ACTJOB コマンドの入力形式

コマンド名	オペランド
$ACTJOB	なし

①機能概要

ジョブの処理状況を表示する。

②表示例

```
キュー名   リクエスト名   リクエスト ID   状態      PE 数       要求 CPU 時間   経過時間
  QUE      REQUEST_NAME    REQUEST_ID    STATUS  ALLOC USE     REQUEST_CPU     TIME
  jq01   a12+13A5b.03/03   96171.aoi       RUN      16  16   15000( 4:10:00)  0:39:31
  jq01   a12+13GKB.01/03   96341.aoi       RUN      16  16   15000( 4:10:00)  2:56:22
  jq01   a31+137J0.01/05   96119.aoi       RUN      16  16   17500( 4:51:40)  2:31:27
  jq01   a31+13JGd.01/05   96383.aoi       RUN      16  16   17500( 4:51:40)  2:26:42
  jq01   a67+13HKM.05/05   96375.aoi       RUN      15  15   10000( 2:46:40)  2:09:00
  jq01   i33+14AiY.01/02   96480.aoi       RUN       2   2   20000( 5:33:20)  2:25:24
  jq01   j16-w11cn.01/01   96547.aoi       RUN       1   0    4200( 1:10:00)  0:36:53
  jq01   l32+12MBO.02/11   96095.aoi       RUN      48  48   20000( 5:33:20)  2:28:38
  jq01   l92+14A5I.01/03   96444.aoi       RUN       3   3   18000( 5:00:00)  2:25:22
  jq01   p42+148he.02/05   96424.aoi       RUN      19  19    3300( 0:55:00)  0:39:31
  jq02   n91+12IL5.02/09   96006.aoi       RUN       1   0   18000( 5:00:00)  3:42:05
  jq02   n91+14BOC.01/02   96493.aoi       RUN       1   0   18000( 5:00:00)  2:33:34
  jq02   n92+12IMk.02/09   96015.aoi       RUN       1   0   18000( 5:00:00)  2:01:43
  jq02   n92+12INR.02/09   96024.aoi       RUN       1   0   18000( 5:00:00)  1:16:42

  TOTAL_PE     ONLINE   OFFLINE     RUNNING[OFFLINE]        IDLING[ONLINE]    1G[ONLINE]
    166          166        0        156       0             10       10       4       4
```

This document is provided by JAXA.

表 4.16　WAITJOB コマンドの入力形式

コマンド名	オペランド
$WAITJOB	なし

①機能概要

ジョブの処理待ち状況を表示する。

②表示例

```
    キュー 順番    リクエスト名  リクエストID    要求        受付時間      優先度
     Q     No.    REQUEST-NAME   REQUEST-ID   PE   CPU  ACCEPT-TIME  PRIORITY  STATUS
     u0     1.  a12+15GTO.01/01  97020.aoi    16   600  15-16:29:27         0  pe-busy
            2.  l32+12MBO.08/11  96101.aoi    48 20000  12-22:12:09         0  next-step
                l32+12MBO.09/11  96102.aoi    48 20000  12-22:12:12         0  next-step
                l32+12MBO.10/11  96103.aoi    48 20000  12-22:12:15         0  next-step
                l32+12MBO.11/11  96104.aoi    48 20000  12-22:12:19         0  next-step
            3.  l19-J0749.06/07  96353.aoi    30 18000  13-16:43:18         0  next-step
                l19-J0749.07/07  96354.aoi    30 18000  13-16:43:21         0  next-step
            4.  a12+14Abj.04/05  96474.aoi    16 15000  14-10:38:04         0  next-step
                a12+14Abj.05/05  96475.aoi    16 15000  14-10:38:08         0  next-step
                      :                          :                          :
           60.  p41+15F3p.01/01  96949.aoi     1  6400  15-15:03:54         0  resv-time
           61.  p41+15F44.01/01  96950.aoi     1  6400  15-15:04:07         0  resv-time
           62.  a76+15C31.01/01  96874.aoi     1 18000  15-12:04:20         0  resv-time
           63.  p73+15Eim.02/02  96929.aoi     1 18000  15-14:44:55         0  resv-time
     l0     1.  n91+12IL5.08/09  96012.aoi     1 18000  12-18:21:54         0  next-step
                n91+12IL5.09/09  96013.aoi     1 18000  12-18:21:57         0  next-step
            2.  n92+12IMk.08/09  96021.aoi     1 18000  12-18:23:36         0  next-step
                n92+12IMk.09/09  96022.aoi     1 18000  12-18:23:40         0  next-step
            3.  n92+12INR.08/09  96030.aoi     1 18000  12-18:24:17         0  next-step
                n92+12INR.09/09  96031.aoi     1 18000  12-18:24:38         0  next-step
    ----
    INFO: 12pe reserved for j06+14IUT.03/10.
    INFO: The job will be started at 16:38:22 restricted to 12000sec-cpu time.
    INFO: But job(pe<=1 & cpu<360sec) is excutable now.
    ----
    Q       NO.    REQUEST-NAME    REQUEST-ID    STATUS
    uc       1.   j65+15GUA.@X/00  97021.aoi     RUNNING
             2.   p51+15GW8.@X/01  97022.aoi     RUNNING
    INFO: AOI REQUEST(routing) NOT FOUND.
```

This document is provided by JAXA.

表 4.17　XNTOM コマンドの入力形式

コマンド名	オペランド
$XMTON	変換元 MSP ファイル名[(メンバ名)]　変換先 MSP ファイル名 [TYPE { (BINALY) [RECFM(DIRECT)] / (TEXT) }] [DISP { ('NEW[(初期量,増分量)]') / ('OLD') }]

①機能概要

MSP 形式ファイルを NWT 形式ファイルに変換する。

②オペランドの説明

(1) 変換元 MSP ファイル名[(メンバ名)]

変換元のファイル名を指定する。

(2) 変換先 MSP ファイル名

変換先のファイル名を指定する。

(3) TYPE { (BINALY) [RECFM(DIRECT)] / (TEXT) }

変換を指示する。

・BINARY ： バイナリ形式の場合に, フォーマット変換を指示する(省略形は B)。
直接編成ファイルの場合には, RECFM(DIRECT) を指定する。

・TEXT ： テキスト形式の場合に, コード変換とフォーマット変換を指示する(省略形は T)。

(4) DISP { ('NEW[(初期量,増分量)]') / ('OLD') }

変換先のファイルの処置を指定する。

・NWT ： 新規ファイルの場合に指定する。
初期量 ： 一次割当量をトラック単位で指定する。
増分量 ： 二次割当量をトラック単位で指定する。

・OLD ： 既存ファイルの場合に指定する。

③使用例

(1)MSP 形式のバイナリデータを NWT 形式に変換する。

```
$XMTON 'X01.MSP10.DATA' 'X01.NWT10.DATA' TYPE(B) DISP('OLD')
```

(2)MSP 形式のテキストデータを NWT 形式に変換する。

```
$XMTON 'X01.MSP20.DATA(CARD)' 'X01.NWT20.DATA' TYPE(T) DISP('NEW(10,10)')
```

This document is provided by JAXA.

表 4.18　XNTOM コマンドの入力形式

コマンド名	オペランド
$XNTOM	変換元 MSP ファイル名　変換先 MSP ファイル名[(メンバ名)] [TYPE { (BINALY) [RECFM(DIRECT)] / (TEXT) }] [DISP { ('NEW[(初期量,増分量[,ディレクトリ量])]') / ('OLD') }]　[ORG { (PO) / (PS) }]

①機能概要

NWT 形式ファイルを MSP 形式ファイルに変換する。

②オペランドの説明

(1) 変換元 MSP ファイル名

変換元のファイル名を指定する。

(2) 変換先 MSP ファイル名[(メンバ名)]

変換先のファイル名を指定する。

(3) TYPE { (BINALY) [RECFM(DIRECT)] / (TEXT) }

変換を指示する。

・BINARY ： バイナリ形式の場合に，フォーマット変換を指示する(省略形は B)。
　　直接編成ファイルの場合には，RECFM(DIRECT) を指定する。

・TEXT　： テキスト形式の場合に，コード変換とフォーマット変換を指示する(省略形は T)。

(4) DISP { ('NEW[(初期量, 増分量[,ディレクトリ量])]') / ('OLD') }

変換先のファイルの処置を指定する。

・NWT　： 新規ファイルの場合に指定する。
　　初期量　： 一次割当量をトラック単位で指定する。
　　増分量　： 二次割当量をトラック単位で指定する。
　　ディレクトリ量： ディレクトリ量をトラック単位で指定する。

・OLD　： 既存ファイルの場合に指定する。

(5) ORG { (PO) / (PS) }

変換先のファイルが新規作成の場合に指定する。

・PO　： 区分編成を指定する。

・PS　： 順編成を指定する。

③使用例

(1)NWT 形式のバイナリデータを MSP 形式に変換する。

```
$XNTOM 'X01.NWT10.DATA' 'X01.MSP10.DATA' TYPE(B) DISP('OLD')
```

(2)MSP 形式のテキストデータを NWT 形式に変換する。

```
$XNTOM 'X01.NWT20.DATA' 'X01.MSP20.DATA(CARD)' TYPE(T) DISP('NEW(10,10,5)') ORG(PO)
```

This document is provided by JAXA.