数値風洞用ジョブスケジューラの開発*

末 松 和 代*1

Development of the Job Scheduler for the Numerical Wind Tunnel

Kazuyo Suematu*1

ABSTRACT

The Numerical Wind Tunnel, NWT is a CFD-oriented vector parallel computer system with distributed memory. We developed a job scheduler to utilize the NWT effectively and put it into operation in October 1994. Since then we have added various functions to the job scheduler to enhance the manageability of the NWT. It now achieves effective use of the processing elements (PEs) and executes tasks with a suitable turn-around time.

This paper reports the functions and availablity of the job scheduler for NWT use.

Keywords : job scheduler, parallel computer system, waiting time, load guantity, scheduler simulator

概　要

数値風洞は，計算空気力学プログラムの超高速処理を目的とした分散メモリ型の並列計算機システムである。数値風洞を効率的に利用するためにジョブスケジューラを開発し，1994年の10月から運用に供している。その際，数値風洞の運用効率を高めるために種々の機能を付加した。その結果，要素計算機の効率的使用，適切なターンアラウンドタイムでのジョブ実行が実現された。

本稿では，数値風洞用ジョブスケジューラの機能および有効性について報告する。

1．はじめに

航技研では，1993年2月に分散メモリ型のベクトル並列計算機システムである数値風洞（図1.1参照）を導入し，稼働を開始した。このシステムは，計算空気力学プログラムの高速実行の実現のために，航技研と富士通㈱との共同研究の結果開発されたシステムである。

数値風洞は，演算処理用の166台のプロセッサエレメント（以下PEと略記する），ジョブおよび入出力制御用の2台のコントロールプロセッサ（以下CPと略記する），PE－CP間およびPE－PE間接続用のクロスバネットワーク，ジョブ実行に必要なファイル転送用の高速システム記憶（以下SSUと略記する），およびフロントエンドプロセッサ（以下FEPと略記する）であるFujitsu - VP2Xから構成されている。なお，プログラムの実行は，翻訳処理，結合・編集処理，実行処理から構成されるが，数値風洞の場合には各PEがピーク性能1.7GFLOPSの高性能ベクトル計算機であることを考慮し，PEでは実

＊　平成10年7月29日受付（received 29, July 1998）

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

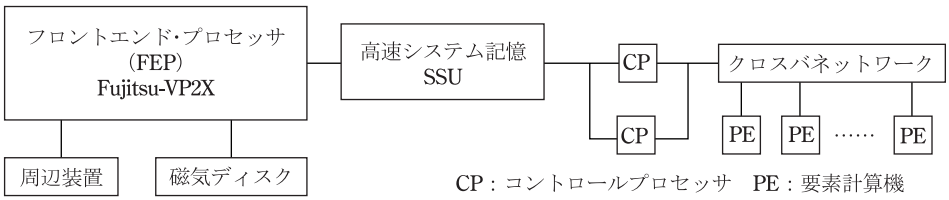

図 1.1　数値風洞のハードウェア構成図

行処理のみを行い，その他の処理は FEP を使用している。

なお，このシステムでは，投入されるジョブの実行の流れを制御する「ジョブスケジューラ」に種々の機能を組み込むことにより，PE の効率的使用や独自の運用方式の実現を図ることができる。従来の汎用計算機システムでは，ジョブスケジューラに計算機の効率的運用を図るためのアルゴリズムを組み込んで使用してきた。しかし，数値風洞は並列計算機であり，ジョブ実行時に割り当てるシステム資源が汎用計算機とは異なることから，汎用計算機システム用のアルゴリズムを利用したのでは空き PE が発生し易い，同時に使用できるジョブクラスに制限がある，ジョブのターンアラウンドタイム（注 1）が適切に制御できない等の問題が発生することが判明した。そのため，システムの有効利用を図るために新たなアルゴリズムの検討を行い，以下の目的を実現する並列計算機用のジョブスケジューラ[1]を考案した。

① PE の高効率利用
② 任意の PE 台数でのジョブ実行
③ システム資源要求量・緊急度に応じた優先度の設定
④ 適切なジョブターンアラウンドタイムの保証

しかし，このジョブスケジューラを実運用に供する際には，システムの効率的利用のための機能だけでなく，そのシステムの運用環境や運用方針，そのシステム独自の要望等を考慮しなければ使い易いシステムを実現することはできない。そのため，並列計算機用に考案したジョブスケジューラの基本機能に，航技研独自の機能を追加したジョブスケジューラを開発し，1994 年 10 月から運用に供している。

本稿では，航技研の数値風洞用に開発したジョブスケジューラ（以下，新スケジューラと呼ぶ）の機能，実現方式，処理構造及び有効性検証結果について報告する。

2．新スケジューラの機能

文献 1 で報告した並列計算機用のジョブスケジューリングアルゴリズムは，スケジューラの最も重要な要素である「システムの有効利用」および「ジョブの適切なターンアラウンドタイムでの処理」に有効である。本章では，新スケジューラの機能として，並列計算機システム用に開発した基本機能と航技研の運用方式・運用環境等を考慮した付加機能について説明する。

2.1　新スケジューラの基本機能

(1) システム資源要求量と待ち時間に応じたジョブ優先度の設定機能

ジョブをシステム資源要求量に応じた適切なターンアラウンドタイムで実行するために，ジョブ優先度はシステム資源要求量や待ち時間等の関数によって任意に設定可能である。数値風洞では，要求 PE 台数，要求 CPU 時間，待ち時間によって優先度を設定している。

(2) ジョブ優先度の動的制御機能

ジョブを待ち時間の限界値を越えた最優先ジョブ，待ち時間が一定値を越えた優先ジョブ，待ち時間が一定値以下の非優先ジョブの 3 つに分類し，それぞれ 3 つのキュー（注 2）で優先度順に管理する。なお，ジョブを分類するためのパラメータは，優先度と同様にシステム資源要求量や待ち時間等の関数によって任意に設定可能である。また，ジョブは常時適切なキュー，適切な優先度で処理するために適時ジョブ優先度や所属すべきキューを見直し，動的にジョブを管理する。

(3) PE の有効利用を考慮したジョブの起動（注 3）機能

PE を有効利用するために，優先度の高いキューの先頭ジョブから順にジョブ起動条件を満たしているすべてのジョブを起動する。処理中に起動条件を満たさないジョブがあっても，割り当て可能な空き PE がある限り後続のジョブの起動を試みる。

(4) 長時間待ちジョブの優先実行機能

システム資源要求量に応じた適切なターンアラウンドタイムでのジョブ処理を図るには，長時間待ちジョブは優先的に実行させる必要がある。そのために，待ち時間

（注 1）ジョブが投入されてから処理が終了するまでに要する時間

（注 2）待ち行列（queue）

（注 3）実行待ちジョブの中から実行すべきジョブを決定すること。また，そのために必要となる処理を行うこと。

の限界値を越えた最優先ジョブを以下の方法により優先的に実行する。

① ジョブ起動時のPEの優先割り当て機能

最優先ジョブの起動時には，実行中ジョブのスワップアウト(注4)により空きPEを生成して優先的にPEを割り当てる。ただし，混雑時など大半のジョブが最優先ジョブになる場合もあるため，特に，実行待ち状態の最優先ジョブの内，最も優先度の高い実行可能なジョブを起動優先ジョブと称し，この対象とする。

② PEリザーブ機能

ジョブ起動時のPEの優先割り当て機能を使用しても起動優先ジョブが起動できない場合には，そのジョブの実行開始までの最短時間を求め，その実行予定時刻から終了予定時刻までの間のPEの割り当てを予約する。この処理をPEリザーブと呼ぶ。なお，PEリザーブ対象となったPEは，他のジョブが起動優先ジョブとなった時点でPEリザーブが解消されるが，そうでない限り他のジョブに割り当てされることはない。

③ スワップアウト抑止機能

実行中の最優先ジョブの処理が滞ることを防止するために，最優先ジョブをスワップアウトの対象から外す。

(5) システムの立ち上げ・停止等を考慮した起動可否判定機能

ジョブ起動の可否を，空きPE台数と要求台数の比較のみで判断した場合には，システム停止等を予定時刻に行えない事態や，起動優先ジョブのPEリザーブに対しPEの割り当てが保証できない事態が発生する。そのため，システムの立ち上げ・停止等を考慮して割り当て可能なPE台数を算出し，起動の可否を判定する際に使用する。

(6) スワップアウトに起因するシステム効率低下回避機能

最優先ジョブを処理する際のスワップアウトは，強制的な空きPE台数増加のための有効な手段である。しかし，スワップアウトが頻発して入出力のためのSSU領域が十分に取れないと，ジョブの入出力時間が極端に長引き，システムオーバヘッドも増加する。したがって，スワップアウトの頻発は好ましくない。また，複数PEを使用するジョブの場合，各PEでの負荷分散が義務づけられている。図2.1に負荷分散の例を示す。例では処理Aに8時間，処理B，処理Cにはそれぞれ2時間，処理Dには4時間かかるプログラムを実行した場合のもので，(1) は1 PEでの実行例である。この場合，全処理を行うのに16時間かかる。この処理を4 PEを使用して最短時間で処理する方法として(2) のような方法が考えられる。ここで処理A1～A4はAの処理を，処理D1，D2はDの処理をそれぞれ均等分割したものであり，実行時間は4時間となる。ただし，処理B，C，D1，D2は処理Aの結果のみを使用して並列に実行可能でなければならない。もし，処理Dが処理B，Cの結果を使用するような場合には，処理D1，D2は，処理B，Cが終わるまで待たされるため(3) のようになり，実行時間は6時間となってしまう。この場合は，処理B，Cをそれぞれ2つの処理に，処理Dは4つの処理に均等分割すれば(4) に示すように実行時間は4時間となる。一般的に，入出力処理は1 PEで行うことが多いため，このように完全に並列化はできないが，できるだけ(2) または(4) のような負荷分散を行うことが要請されている。

したがって，このように負荷分散されているジョブの

(注4) 実行中ジョブのPE内の情報を継続可能な形でSSUやディスクに待避すること。

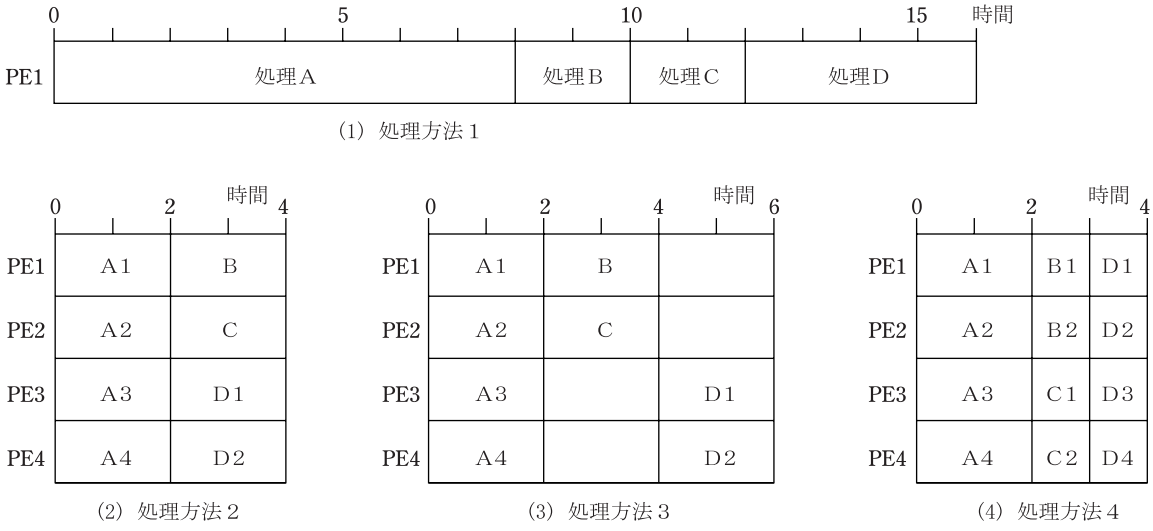

図2.1 並列計算機使用時のプログラムの負荷分散の例

PEが1台でもスワップアウトされるとすべてのPEの処理がその分だけ遅れることになり，スワップアウトに伴うPE利用率の低下の割合は使用PE台数が多いジョブほど顕著となる。

上記理由から，スワップアウトを行う際には，1つのジョブを起動するためのスワップアウトPE台数，システムの総スワップアウトPE台数，スワップアウト可能なジョブの規模をそれぞれ制限し，スワップアウトに起因するシステム効率の低下を回避する。

2.2 航技研運用方式実現のための機能

航技研では，昭和35年のデータトロン205の導入以来，5年から7年の周期でその時代の最新鋭の計算機システムを導入してきた。そして，これらのハードウェア性能を十分に発揮でき，使い易いシステムの構築を目指して，種々の運用管理方式の研究[2)-14)]が行われ，多くの運用管理プログラムが開発された。

本項では，航技研における従来の有効な運用方式を継続すると共に，新たな運用方式の実現のために数値風洞において必要となる機能について説明する。

(1) 起動ジョブの制限機能

数値風洞では，効率的なシステム運用を行うために，ジョブの起動を以下の項目で制限する。

① 1 PEジョブ実行数の制限

数値風洞では，ジョブの優先度はジョブ属性，システム資源要求量および待ち時間で決定する。したがって，1PEのジョブは相対的に短時間で優先度が上がるため，1 PEジョブの実行数を規制しないと実行ジョブがすべて1 PEジョブとなる恐れがある。特に混雑時には，大規模ジョブばかりがシステムに残留するような結果になり兼ねず，システムの効率上も好ましくない。以上の理由から，1 PEジョブの実行数を制限する。

② 同一ユーザジョブの投入ジョブ数,実行ジョブ数,使用PE台数の制限

従来のシステムでは，同一ユーザが一度に大量のジョブを投入した場合，システム内に過大なジョブが滞留したり，特定ユーザのジョブばかり実行される可能性があるため，同一ユーザの投入ジョブ数や実行ジョブ数を制限してきた。数値風洞でも同様に同一ユーザのシステム内ジョブ数や同時実行ジョブ数を制限する。また，同一ユーザが同時に使用できるPE台数も制限する。

③ ジョブの最大使用PE台数およびCPU時間の制限

航技研計算機システムでは，昼間は小規模ジョブやデバッグジョブを優先実行し，ユーザが少ない夜間は大規模ジョブの実行を可能とすることにより，プログラム生産性の向上を図ってきた。数値風洞でもこの方針を継続するため，時間帯毎に実行可能なジョブの最大PE台数やCPU時間を制限する。

(2) ユーザタイムジョブの優先実行機能

汎用計算機の場合には，システム資源使用量が多いためジョブがなかなか処理されず，研究に支障をきたす恐れのあるユーザなどに対し，夜間や週末のように利用者が少ない時間帯を選んでシステムを占有割当したり，優先的に実行したりする「ユーザタイム運用」を行ってきた。数値風洞はPE台数が多く，通常は全PEを割り当てる必要がないため，ユーザタイムユーザのジョブ（以下ユーザタイムジョブと呼ぶ）は一般ジョブより優先度を一定値上げて処理する。しかし，独占使用させることも可能であり，その場合は一般ジョブはキューへ登録されても実行されないか，あるいはキューへの登録もリジェクトするかを指定することができる。

(3) 優先ユーザジョブの優先実行機能

ユーザタイムジョブ以外にも，システム内の緊急性の高いジョブを優先的に処理できるようにするため，「高優先度ジョブ」という概念を設ける。そのため，一般ジョブの高優先度化，高優先度ジョブの非高優先度化を可能とし，一般ジョブの中では，高優先度ジョブ，ユーザタイムジョブ，その他のジョブの順に優先する。なお，優先ジョブはその目的により，優先度化時刻順に優先する方式とする。

(4) 優先度の調整機能

航技研計算機システムは，一般研究者，研修生，システム管理者等多様なユーザが使用するばかりでなく，使用目的も多岐に渡っている。ジョブスケジューラの基本機能では，これらのユーザの優先度をユーザ属性によって変更させることが可能であるが，さらにユーザごとに優先度を調整し，木目細かな制御を目指す。

(5) ジョブステップの逐次実行機能

航技研では，CPU制限時間を越えるプログラム処理を行うために，長時間プログラムの処理を複数ジョブステップに分割して実行可能とする必要がある。この場合，以下の2項目を保証しなければならない。

a) ジョブステップ間での実行開始順序の保証

b) ジョブステップ間での同時実行しないことの保証

しかし，数値風洞では同一ジョブ内の各ジョブステップは先頭から逐次実行するため，項目b)の機能のみ必要となる。この機能は，ジョブ起動時に同一ジョブ内のジョブステップ処理が実行中か否かを判定し，実行中でない場合のみ起動可能とすることにより実現できる。

(6) システム状態情報出力機能

ユーザにとって，システムに投入したジョブがどのような状態になっているのか，システムの混雑度はどの程度かということは重大な関心事である。そのため航技研システムではシステム状態表示機能を設け，ジョブ状態

表示用ディスプレイ装置やワークステーション等にシステムの状態表示を行うための機能を用意し，ジョブの終了予測等を行えるような環境を用意してきた。数値風洞でも，ジョブ受付時，ジョブ実行開始時，ジョブ終了時等に情報を出力し，システムの混雑状態やジョブの処理状態の表示を可能とする。

2.3 運用環境を考慮した機能

本項では，数値風洞の運用環境を考慮した上で必要と考えられる機能について説明する。

(1) 自動化プログラムとの連携機能

数値風洞には，電源制御操作等を自動化するための「自動化プログラム」が，FEPおよび数値風洞のCPに搭載されている。自動化プログラムは，あらかじめ設定されている自動化スケジュールテーブル内のイベント情報を参照して，運用時間毎の運用PE電源グループ（注5）番号，ジョブクラス毎の多重度（同時実行可能な最大ジョブ数）を認識し，システム，冷却設備およびPEの電源操作処理とジョブクラス毎の多重度設定処理等を行う。自動化プログラムの使用するこれらの情報はスケジューラにとっても必須な情報であるが，それぞれ独自の情報を使用した場合には情報の不整合が発生する恐れがあるため，自動化プログラムとの連携を取り，効率的なジョブスケジュールを行う。

(2) 使用可能PE認識機能

一般的に，システム内の故障PEの発生確率は設置PE台数に依存するため，設置PE台数の多い数値風洞では常時使用可能なPE台数を正確に認識することが重要となる。そのため，運用の切替え時には，PEの状態からその時点での使用可能PE台数を認識する。

(3) 故障PE認識機能

ジョブの起動の可否を判定する際に運用時間帯ごとの使用可能PE台数が必要となる。運用時間帯ごとの使用可能PE台数の予測は，まず故障PE認識機能により故障中のPE番号を入手し，運用時間帯に使用する予定の電源グループ番号と，各電源グループを構成するPE番号，故障ＰＥ番号を照合することにより行う。

(4) システム状況に基づくスケジューリング可否判定機能

数値風洞で実行するジョブは，FEPを介して入出力処理等を行っているため，FEPでシステム異常が発生した時には新たなジョブのスケジューリングを停止させる必要がある。また，運用の切替え中も安全のために同様の処置を行う必要がある。この様に，ジョブのスケジューリングを停止または再開させるべき事象が発生した場合には，FEPはスケジューラにその旨を通知し，スケジューラもこれらの情報に基づきスケジューリングの停止および開始を行う。

(5) 複数ジョブグループの設定機能

数値風洞では，システムオペレータや緊急度の高いプロジェクトを処理するユーザを特権ユーザ，特権ユーザの投入したジョブを特権ジョブと呼んで最優先に処理を行うものとする。また，数値風洞を構成する166台のPEのメモリ容量は，162台は256MB，残りの４台は１GBであるため，256MBのPEを要求するジョブと１GBのPEを要求するジョブを別々に管理しなければならない。

このように，緊急度や要求するシステム資源が異なるジョブを効率よく管理するために，複数のジョブグループを設定できるようにする。

(6) 複数スケジューリングアルゴリズムの共存機能

新スケジューラは，ジョブはシステム資源要求量と待ち時間に応じた適切な時間で処理するという考え方に基づいている。そのためには，ジョブは投入されてから実行が終了するまでの間，システム資源要求量と待ち時間に応じた優先度が設定されなければならない。ここで言うシステム資源要求量は，ジョブが翻訳，結合・編集，実行という過程を経る場合には，実行時のものを指す。しかし，現実のシステムでは，FEPで行われる翻訳処理，結合・編集処理時には，実行時のシステム資源要求量が認識できないことから，先着順等の一般的なアルゴリズムで処理せざるを得ない。また，特権ジョブは，その目的からシステム内に複数存在した場合には先着順に処理することが望ましい。

そのため，並列計算機システム用に考案したアルゴリズム（以下，PP方式と呼ぶ）だけでなく，先着順のアルゴリズム（以下，FCFS方式と呼ぶ）を用意し，ジョブグループ毎に，また使用システム毎に指定のアルゴリズムでの処理を可能とする。

３．新スケジューラの実現方式

図3.1は，数値風洞のソフトウェア概念図である。図に示されているように，ＦＥＰおよび数値風洞のコントロールプロセッサＣＰには，第２章で述べた「自動化プログラム」の他に，リクエスト（注6）の流れを管理するための「Network Queueing System（NQS）」が搭載されている。NQSは，アメリカ航空宇宙局の依頼によりSterling Software社が1986年に開発したUNIX上のキ

（注５）PEの電源制御の単位であり，８台または12台のPEから成る15の電源グループがある。

（注６）数値風洞における処理の単位。３ジョブステップから構成されるジョブを投入した場合，数値風洞では３つの独立したリクエストとして処理される。

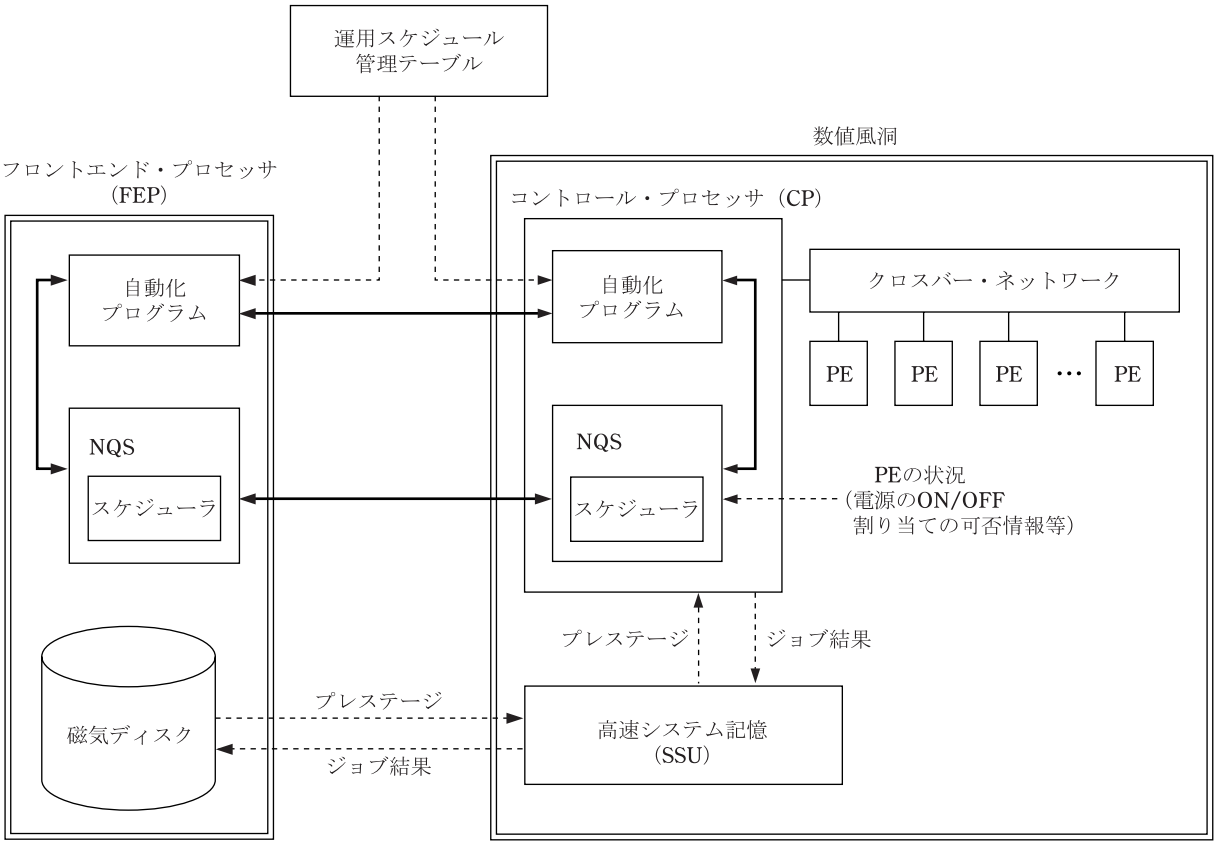

図 3.1　数値風洞のソフトウェア概念図

表 3.1　NQS の主な関数と機能

関数名	機　　能	内容変更の可否
qsub	リクエストの投入	不可
nqs_acceptreq	リクエストのキューイング制御	可能
nqs_closereq	リクエストの後処理	可能
nqs_delreqinfo	リクエストの削除	可能
nqs_init	NQS ブート時の初期化処理	可能
nqs_qcall	qcall コマンドによる NQS デーモン内関数の実行	可能
nqs_schedbatreq	リクエストのスケジューリングと実行	可能
nqs_spawn	NQS リクエストの実行	不可
nqs_vtimer	指定時刻での関数呼び出し	不可
nqs_unsetvtimer	関数呼び出しの設定解除	不可

ューイングシステムであり，システム内のリクエストの管理やスケジュール等を行うための数多くの関数から構成されている[15),16)]。スケジューラに関係する代表的な関数の関数名と機能を表 3.1 に紹介する。これらの関数では，リクエストの受け付け，システム資源要求量が制限値をこえるリクエストのリジェクト，多重度による実行リクエストの制御，実行リクエストの先着順処理等をクラス毎に行うことができる。なお，これらの関数の中にはリクエスト到着時，実行終了時，キャンセル時の手続き付加や独自のスケジュールアルゴリズムの組み込みを行うために内容変更が許されている関数がある。

新スケジューラでは，前章で述べた基本機能，航技研運用方式実現のための機能，運用環境を考慮した機能が必要であるが，既存の NQS では実現できない機能が多いため NQS に足りない機能を追加して，航技研独自のジョブスケジューラを構築する。また，これらの機能は適切なタイミングで確実に実行しなければならないが，各機能の処理手順やタイミングについては次章で述べるものとし，本章では，これらの機能を実現するために必要となるリクエスト管理方式，リクエスト起動方式につ

いて述べる。

3.1 リクエスト管理方式

以下に，リクエストを管理するためのキュー構成とリクエストの分類，リクエストのキュー遷移条件，リクエストのキュー内優先度について説明する。

(1) キュー構成とリクエストの分類

図3.2に，NQSのリクエスト管理方式例を示す。例では，256MBのPEを要求するリクエストの管理方式を示したもので，特権ジョブのリクエスト（以後，特権リクエストと呼ぶ）を処理するJQ00クラスと一般ジョブのリクエスト（以後，一般リクエストと呼ぶ）を処理するJQ01クラスという２つのクラスを定義した場合を示している。

各クラスに投入されたリクエストは，以下に述べるwaitset，queuedset，stageset，runset，holdsetの中のいずれかの状態となり，状態別に優先度順で管理される。なお，queuedset，stagesetのリクエストのみ起動可能である。図3.3に状態遷移図を示す。

waitset：リクエストの実行日時の指定等の条件により待ちの状態となっているリクエスト。待ち条件が解消されれば，queuedsetとなる。

queuedset：stageset，runsetに移行可能なリクエストで，プレステージ(pre-stage)されればstagesetに，実行開始されればrunsetになる。

stageset：プレステージが完了したリクエストで，実行されればrunsetになる。

runset：実行中のリクエスト。実行が終了し，FEPへのファイル転送終了後削除される。

holdset：waitset，queuedset，stageset，runsetへの移行抑制中のリクエスト。

しかし,NQSで設定できる優先度のレベルには限界があり，キューへの登録時に決定した優先度はリクエストが削除されるまで変化しない。したがって，待ち時間に応じて優先度を変化させることはできない。また，リクエスト到着時にはディスク上のすべての情報が参照可能であるが，到着後はメモリ内に展開された一部のデータしか参照できないためリクエストのシステム資源要求量等を参照することはできない。すなわち，リクエスト到着後に随時参照しなければならないデータは別に管理する必要がある。

したがって，新スケジューラではリクエストを任意レベルの優先度で優先度順に管理でき，システム資源要求量等を格納することの可能な管理キューが必要となる。また，次項に述べるように，起動可否判定時にシステム

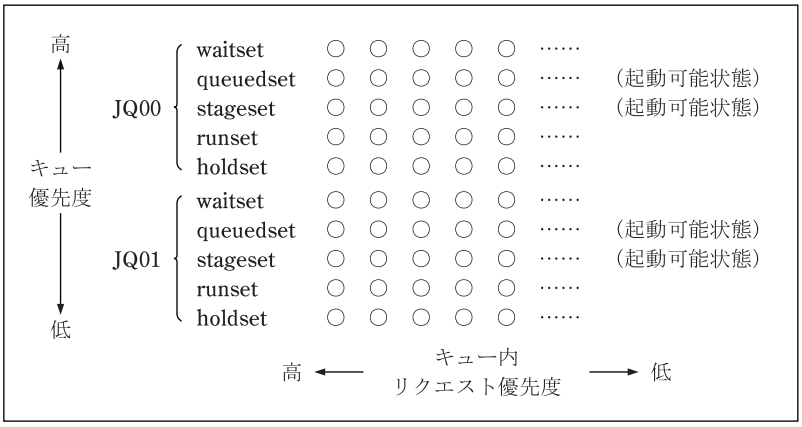

図3.2 NQSのリクエスト管理例

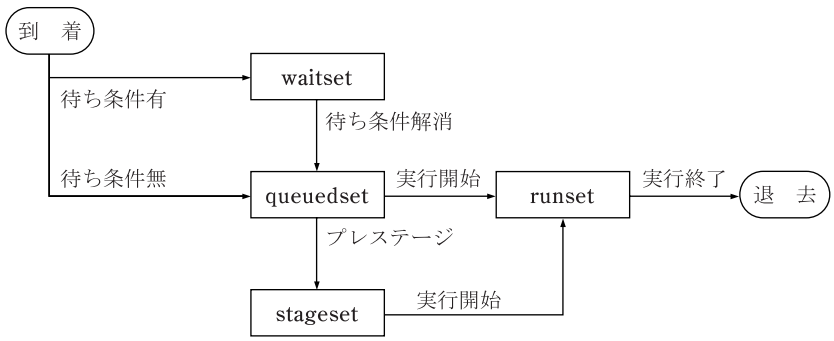

図3.3 NQSにおけるリクエストの状態遷移図

表 3.2　新スケジューラのリクエスト管理キュー

キューの分類	キュー名	取り扱うリクエストの種類	リクエスト順序
特権リクエスト管理キュー	prwait	実行待ちのリクエスト	優先度順
	prexec	実行中のリクエスト	
一般リクエスト管理キュー	gewait[1] gewait[2] gewait[3]	実行待ちの非優先リクエスト 実行待ちの優先リクエスト 実行待ちの最優先リクエスト	優先度順
	geexec[1] geexec[2] geexec[3]	実行待ちの非優先リクエスト 実行待ちの優先リクエスト 実行待ちの最優先リクエスト	
残り実行時間管理キュー	rtime	実行中全リクエスト	終了予定順

の空き PE 台数の変化を予測するために実行中リクエストの残り時間を管理するためのキューも必要である。

以上の結果,新スケジューラの機能を実現するために, NQSのリクエスト管理キューとは別に, 表3.2に述べる５種類のリクエスト管理キューを使用する。これらのキューには, 優先度またはリクエスト残り時間を示す情報の他に起動判定等に必要となるシステム資源要求量等の情報や, リクエストを識別するためのマシン番号とシークエンス番号が格納されている。なお, キュー名の中の pr，ge，r は特権 (privilege) , 一般 (general) , 残り時間 (remain time) の頭文字を取っている。

なお, これらの管理キューは, 優先度または残り時間の順に管理するが, それらのデータは時間に応じて変化する。そのため, 各キューの優先度, または残り時間は, 先頭リクエストはそのままの値を, 後続のリクエストは直前のリクエストとの差を格納する。これにより，t 時間経過した場合の優先度または残り時間は, 先頭のリクエストからのみ t を減算することになるため, 計算の効率化を図ることができる。

また，256MB の PE を要求するリクエストと 1 GB の PE を要求するリクエストのように，複数のジョブグループを取り扱う場合にはその分だけ管理キューの数が増えることになる。例えば，i 種類のジョブグループがある場合には,

prwait[ii], prexec[ii]　(ii=1 〜 i)

gewait[ii][m], geexec[ii][m]　(ii=1 〜 i,m=1 〜 3)

rque[ii]　(ii=1 〜 i)

となる。しかし, 数値風洞では, 以下に示すように, 台数の多い256MBのPEを要求するリクエストのみ特権リクエストの定義を可能にしている。

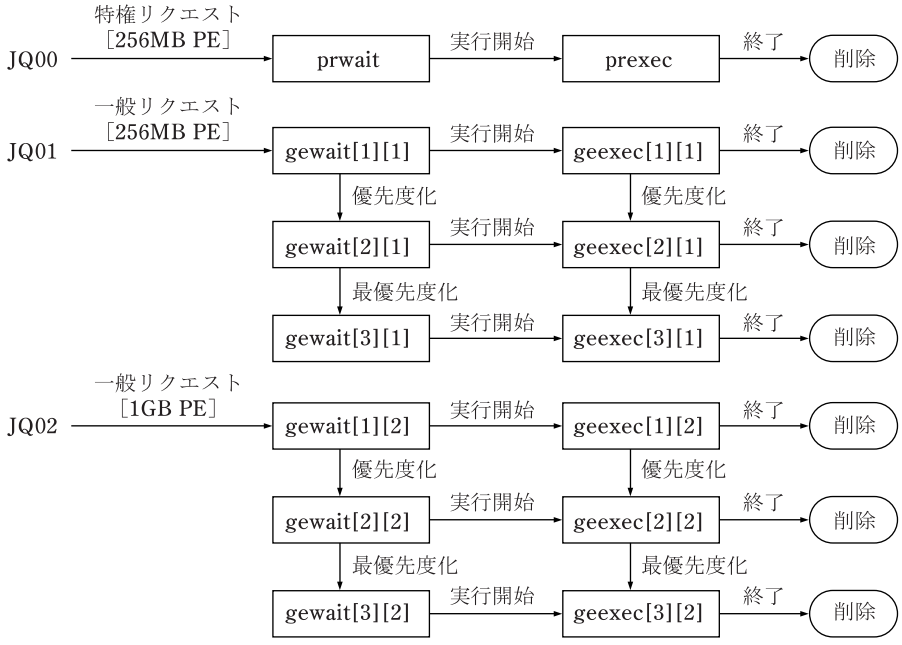

図 3.4　リクエスト遷移説明図

JQ00：256MB の PE を使用する特権クラス
JQ01：256MB の PE を使用する一般クラス
JQ02：１ GB の PE を使用する一般クラス
そのため，
prwait[ii], prexec[ii] (ii=1)
gewait[ii][m], geexec[ii][m] (ii=1 ～ 2,m=1 ～ 3)
rque[ii] (ii=1 ～ 2)
となっている。

図 3.4 に，リクエスト管理キューにおけるリクエストのキュー遷移説明図を示す。同図は，システムに投入されたリクエストが実行終了するまでに遷移するキューを図示したものである。

図中の遷移条件である「優先度化」および「最優先度化」は次項のキュー遷移条件の中の「優先ジョブ」および「最優先ジョブ」への移行条件を示している。

なお，前述したように，特権リクエストを扱う JQ00 クラスは先着順処理を行う FCFS 方式で，一般リクエストを扱う JQ01，JQ02 クラスはシステム資源要求量と待ち時間に応じた優先度順処理を行う PP 方式で処理をする。しかし，FEP では資源要求量が認識できないためにすべてのクラスのリクエストを FCFS 方式で処理をする。

図 3.5，図 3.6 に実行待ちリクエストおよび実行中リクエストの管理キュー説明図を示す。新スケジューラではリクエストを優先度順に管理する。また，特権リクエストを扱う prwait キュー，prexec キューは一般リクエストを扱う gewait キュー，geexec キューより高い優先度を設定している。同図の丸印は管理キューに登録されているリクエストを，中の数字はシステム内の相対的なリクエスト優先順位を示す。

なお，上記処理と平行して，特権リクエストおよび一般リクエストの実行時には，rtime キューに登録し，実行終了時には rtime キューから削除する。

(2) リクエストのキュー遷移条件

PP 方式の場合には，待ち時間が一定値に達したリクエストをより優先度の高いキューに移すために，実行待ちリクエストおよび実行中リクエストの遷移条件を以下のように設定する。

(a) 実行待ちリクエストの遷移条件
gewait[2] への遷移条件：$WTj \geqq WTj_1$
gewait[3] への遷移条件：$WTj \geqq WTj_2$

(b) 実行中リクエストの遷移条件
geexec[2] への遷移条件：$WTj \geqq WTj_1$
geexec[3] への遷移条件：$WTj \geqq WTj_2$

ここで，WTj は，そのリクエストが投入されてから現在に至るまでの待ち時間であるが，高優先度ジョブのリクエスト（以後，高優先度リクエストと呼ぶ）の場合に

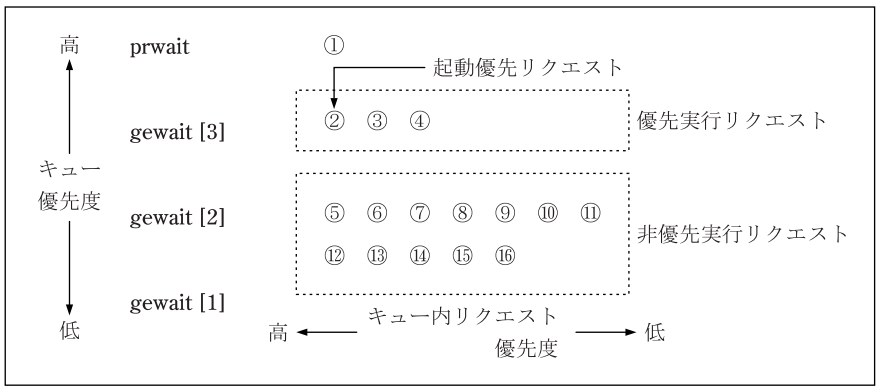

図 3.5　実行待ちリクエスト管理キュー

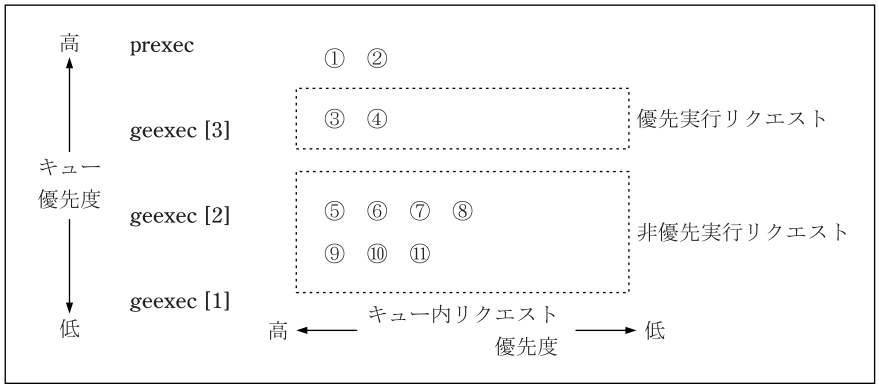

図 3.6　実行中リクエスト管理キュー

は高優先度化されてからの時間，ユーザタイムジョブのリクエスト（以後，ユーザタイムリクエストと呼ぶ）の場合にはユーザタイム開始後の時間となる。また，WTj_1，WTj_2 は以下のように定義する。

・高優先度リクエストの場合

$$WTj_1 = WTj_2 = WTSPF$$

・ユーザタイムユーザのリクエストの場合

$$WTj_1 = WTj_2 = CPUj \times \sqrt{PEj} \times WT2F + WTUTF$$

・その他のリクエストの場合

$$WTj_1 = CPUj \times \sqrt{PEj} \times WT1F + userfact$$
$$WTj_2 = CPUj \times \sqrt{PEj} \times WT2F + userfact$$

ここで，PEj および CPUj はリクエスト j の要求 PE 台数および要求 CPU 時間（単位：秒）を，userfact はユーザ毎の優先度の調整値を表す。また，WT1F，WT2F はシステムの混雑度に対処するための運用パラメータであり，以下の範囲で定義する。

$$0 < WT1F \leqq WT2F < \infty$$

遷移条件の定義からも判るように，数値風洞では「大規模 CFD プログラムの高速実行」という設置目的を考慮して，ＰＥ台数の多いリクエストを優遇している。また，WTSPF，WTUTF は高優先度リクエストやユーザタイムリクエストを優先するためのパラメータであり，

$$WTSPF \ll WTUTF \ll 0$$

とする。したがって，先に述べた遷移条件から，高優先度リクエストやユーザタイムリクエストは，実行可能であれば ppexec[3] へ，実行不可能であれば ppwait[3] に直ちにつながれる。

(3) リクエストのキュー内優先度

FCFS 方式の場合には，到着順にリクエストを管理するため，キュー内優先度（以後，単に優先度と呼ぶ）の設定は行わないが，PP 方式の場合には以下の様に設定する。

gewait[1] および geexec[1] のリクエストの優先度 PR_1，gewait[2] および geexec[2] のリクエストの優先度 PR_2，gewait[3] および geexec[3] のリクエストの優先度 PR_3 は，それぞれ以下の関数に従って決定する。なお，キュー内優先度の値の小さいもの程優先度は高いものとする。

$$PR_1 = WTj_1 - WTj$$
$$PR_2 = WTj_2 - WTj$$
$$PR_3 = WTj_2 - WTj$$

ここで，前項の遷移条件から，PR_1 および PR_2 は常に正となり，PR_3 は常に負となる。また，$PR_1 \neq PR_2$ であることから，gewait[2] または geexec[2] へ遷移したリクエストの優先度はすでにキュー内に存在するリクエストの優先度より高くなることがあるのに対し，$PR_2 = PR_3$ であるため，gewait[3] または geexec[3] キューへ遷移したリクエストの優先度はすでに存在するリクエストの優先度より高くなることはない。

また，gewait[3] および geexec[3] キューの場合，高優先度リクエストの優先度 PR_3(SPECIAL)，ユーザタイムリクエストの優先度 PR_3(USERTIME)，その他のリクエストの優先度 PR_3(NOMAL) の関係は，

PR_3(SPECIAL) ≪ PR_3(USERTIME) ≪ PR_3(NOMAL)

となり，高優先度リクエストはユーザタイムリクエストより常に優先され，ユーザタイムリクエストはその他のリクエストより常に優先される。

3. 2 リクエスト起動方式

リクエストを起動するためには，要求台数の PE の割り当てが必要条件となるが，本スケジューラではリクエストの起動を効率的，効果的に行うために，独自の方式を用いる。以下に，リクエスト起動条件，起動時の PE 割り当て可否判定条件，割り当て可能な空き PE 台数算出方法について説明する。

(1) リクエスト起動条件

新スケジューラでは，時間帯毎にシステム資源要求量による起動リクエスト制限，クラス毎の多重度制限，同一ジョブ内のリクエストの逐次実行を行っていることから，リクエストの起動可否および PE リザーブ実行可否の判定には以下の条件が考えられる。

① システムがスケジュール可能な状態であること。
② ＮＱＳの実行可能キュー（queuedset または stageset）のリクエストであること。
③ 要求台数の PE の割り当てが可能であること。
④ 実行多重度に余裕があること。
⑤ １ PE 使用リクエストの実行制限数以下であること。
⑥ 同一ユーザリクエストの実行制限数以下であること。
⑦ システム資源要求量が起動リクエストの制限値以下であること。
⑧ 同一ジョブ内の他のリクエストが実行中でないこ

と。

しかし，組み込まれる計算機，処理方式（PP 方式／FCFS 方式），処理内容（起動優先リクエストの起動／非起動優先リクエストの起動／起動優先リクエストの PE リザーブ）等により起動判定に必要と考えられる項目が異なることから，それぞれの処理でチェックの必要な項目，不要な項目，意味のない項目に分け，必要な項目のみを条件とすることにした（表 3.3 参照）。

以下に設定理由を簡単に説明する。

CP用のスケジューラでは，PP方式の非起動優先リクエストを起動する際には全ての条件が必要となる。しかし，PP方式の起動優先リクエストの場合には，早急に処理するという目的から運用スケジュールの妨げとならない⑤と⑥の項目を除外した。また，PE リザーブ処理では，PE リザーブ開始予定時刻におけるリクエストの起動条件のうち，起動判定時まで判らない項目は除外したが，リクエストの高優先度化・非高優先度化やユーザタイム運用により PE リザーブ対象のリクエストのシステム内の相対優先順位が変化することを考慮し，PE リザーブ開始予定時刻に起動優先リクエストであることを確認する項目を付加した。また，FCFS 方式のリクエストは特殊用途で用いるために⑤から⑦の起動リクエスト制限を除外した。

一方，翻訳処理，結合処理を実行するFEP用のスケジューラではFCFS方式で処理するが，これらはCPのPP方式で処理する非起動優先リクエストの場合に準ずるものとする。しかし，FEP は並列計算機でないため PE の割り当てに関する項目は除外し，システム資源要求量は実行時のものが認識できないために除外した。

(2) PE 割り当て可否判定条件

リクエスト起動時および PE リザーブ時に必要となる PE 割り当て可否判定条件を以下のように設定する。

(a) 起動優先リクエストの起動の場合

条件１：PEj ≦ free
条件２：(PEj － free) ≦ AVSWAP,
ただし、AVSWAP=MAX(SWAP1,SWAP2,SWAP3-SWAP)

(b) 非起動優先リクエストの起動の場合

条件１：PEj ≦ free

(c) 起動優先リクエストのＰＥリザーブの場合

条件１：PEj ≦ free

ここで使用されている記号の定義は，以下の通りである。

PEj：起動対象リクエストの要求 PE 台数
free：割り当て可能空き PE 台数
SWAP：スワップアウト中の PE 台数
SWMAX：スワップアウト可能なリクエストの使用 PE 台数の上限
SWAP1：スワップアウト可能なリクエスト（実行中の非優先リクエスト，優先リクエストのうち要求PE台数がSWMAX 以下のリクエスト）の総 PE 使用台数
SWAP2：一つのリクエストを起動する際のスワップアウトＰＥ台数の上限
SWAP3：システム全体での総スワップアウトPE台数の上限

表 3.3　リクエストの起動条件・PE リザーブ条件

○：必要　×：不要　―：意味なし

条件の種類	起　動				PE リザーブ
スケジューラの組み込み計算機	ＦＥＰ	ＣＰ			
スケジュール方式とリクエストの種類 ／ 項　目	FCFS 方式	FCFS 方式	ＰＰ方式		
			起動優先	非起動優先	PE リザーブ
①システム状態（スケジュール可否）	○	○	○	○	――
②NQS の実行可能キューのリクエスト	○	○	○	○	○
③PE の割り当て	――	○	○	○	○
④実行多重度制限	○	○	○	○	――
⑤起動制限１：１ PE リクエスト数	――	――	×	○	――
⑥起動制限２：同一ユーザリクエスト数	○	――	×	○	――
⑦起動制限３：システム資源要求量	×	――	○	○	○
⑧ジョブステップの逐次実行	○	○	○	○	――

条件１は，空き PE だけでリクエストを実行するための条件であり，この条件を満たした場合には，要求台数の PE を割り当ててリクエストを実行する。

条件２は，空き状態の PE が要求台数に満たないときに，不足分の PE をスワップアウトにより生成するための条件であり，起動優先リクエスト，すなわち，ppwait [3] キューの先頭リクエストの場合のみに使用する。この条件を満たした場合には，スワップアウト可能なリクエストの中の実行優先度の低いリクエストから順に (PEj-free) 台数分の PE をスワップアウトし，既存の空き PE とスワップアウトによって生じた空き PE を割り当ててリクエストを実行する。

以上に述べたように，起動可否判定時および PE リザーブ可否判定時には，そのリクエストに対する「割り当て可能空き PE 台数」の値が必要となる。次項で，これらの台数の算出方法について説明する。

(3) 割り当て可能空き PE 台数算出方法

起動優先リクエスト起動時には，他の実行待ちリクエストより優先的に起動するため，空き PE 台数をすべて使用できると考えがちであるが，使用可能 PE 台数が変化する場合には，その時刻と変化量を考慮しないとシステム停止等の処理を遅延させる恐れがある。さらに，非起動優先リクエスト起動時には，起動優先リクエストのための PE リザーブ台数をも考慮しなければならない。そのため，リクエスト起動時の割り当て可能な空き PE 台数は以下のように算出する。

リクエストを新たに起動しないと仮定すると，システムで使用可能な空き PE 台数は，①実行終了，②使用可能最大 PE 台数の変更，③ PE リザーブ開始，④ PE リザーブ終了により変化する。

最初に，PE リザーブされていない場合，すなわち，①および②のみを考慮した場合の例を図 3.7 に示す。

図では縦軸に PE 台数を，横軸に時刻を採っている。説明に先立ち，以下の記号を定義する。

jmax：現在実行中のリクエスト数

PEj：ｊ番目に終了予定のリクエストの使用 PE 台数 (j=1 ～ jmax)

nmax：空き PE 台数変更事象の数

t_0 ：現在の時刻

tn ：ｎ番目に発生する空き PE 台数変更事象の発生時刻 (n=1 ～ nmax)

fn：時刻 tn における空き PE 台数 (n=0 ～ nmax)

FPEn：時刻 tn における空き PE 台数の変化量 (n=1 ～ nmax)

PEmax：時刻 to におけるシステムで使用可能な最大 PE 台数

この場合，空き PE 台数の変化事象はリクエストの終了および図中の時刻 t1，t4，t7 で示されるような「利用可能な PE 台数の変更」となる。なお，時刻 tn における空き PE 台数 fn は次式で表される。

$$f_n = \begin{cases} PEmax - \sum_{j=1}^{jmax} PEj & (n = 0 \text{ のとき}) \\ f_{n-1} + FPEn & (n = 1 \sim nmax \text{ のとき}) \end{cases}$$

同図では，起動優先リクエストを時刻 t0 から開始した

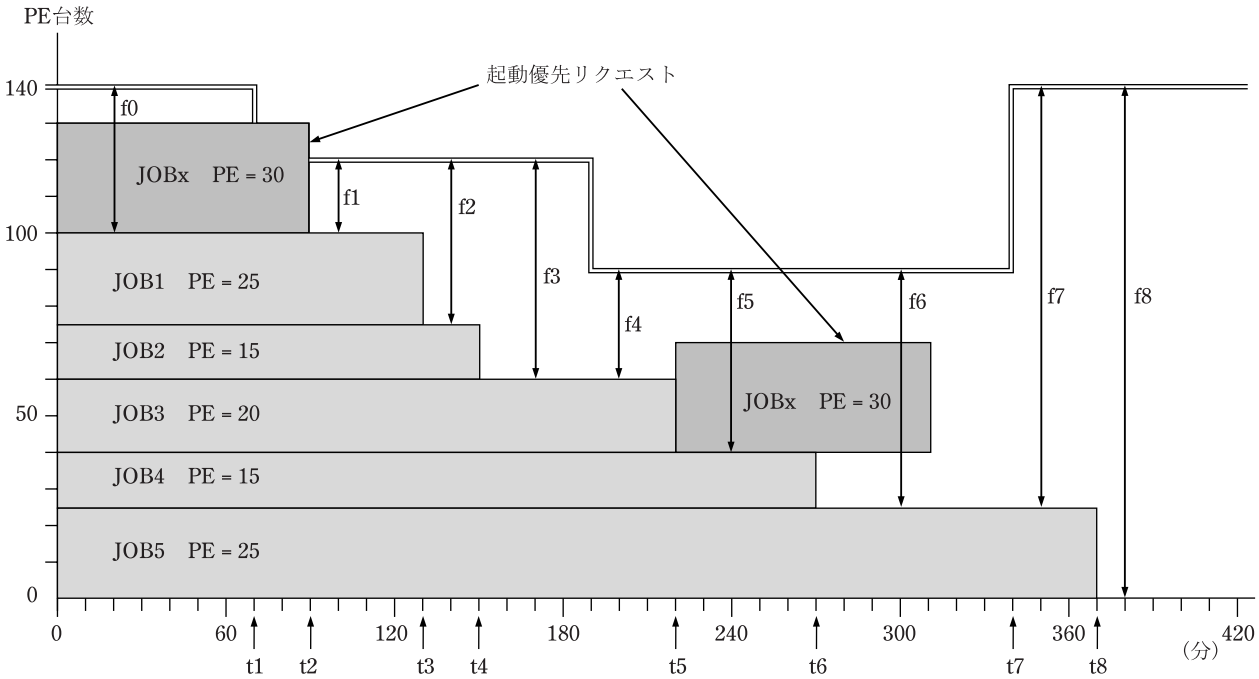

図 3.7 空き PE 台数算出方法説明図

表 3.4　割り当て可能 PE 台数算出表

n	時刻(分)	FPEn	fn	備　考
0	0	——	40	リクエスト終了
1	70	− 20	20	PE 台数削減
2	120	25	45	リクエスト終了
3	150	15	60	リクエスト終了
4	190	− 30	30	PE 台数削減
5	220	20	50	リクエスト終了
6	270	15	65	リクエスト終了
7	340	50	115	PE 台数増加
8	370	25	140	リクエスト終了

場合と時刻t5から開始した場合の例が示されているが，free =「現在の空き PE 台数」と考えて時刻 t1 までに終了しないリクエストをt0に起動した場合には，時刻t1に発生する使用可能 PE 台数の削減処理が阻害されることを示している。すなわち，使用可能最大 PE 台数が変化する場合には，free =「時刻 t0 から時刻 t0+ELAPSj までの間の最小空き PE 台数」としなければならない。ここで，ELAPSj は「起動対象リクエストの経過時間」を表す。実際の起動判定処理では，表 3.4 に示す形式の表を使用して，各時刻における空き PE 台数を算出することができる。また，PE リザーブが行われている場合には，PEのリザーブ開始およびPEリザーブ終了をシステムの使用可能ＰＥ台数の変更事象と同様に考える事により対処できる。

同様に，PE リザーブの場合には，free =「時刻 tn から時刻 tn+ ELAPSj までの間の最小空き PE 台数」とすることにより，時刻tnでの起動の可否を判定することが可能になる。

なお，割り当て可能空き PE 台数の算出にはリクエストの経過時間が必要になるが，数値風洞ではリクエスト情報の経過時間を使用することができない。その理由は以下の通りである。

リクエストの出力はSSUを経由してFEP配下のディスクに書き込まれるが，プログラムはSSUにすべて書き込んだ時点でつぎの処理に移ることが可能である。

したがって，要求経過時間が十分でない場合には，SSUへの書き込みが終了し，リクエストの処理がすべて終了しても，ディスクへの書き込みが終了する前に経過時間でリクエストが打ち切られる事態が発生する可能性がある。したがって，リクエストの出力結果を保証するためには要求経過時間は十分大きめにセットせざるを得ない。

しかし，本スケジューラにおいて，リクエストの経過時間情報は必須であり，文献１で示したように，要求CPU時間からの予測がかなり有効であることが実証されているため，要求 CPU 時間に一定の値を掛けてもとめる方式を採る。

４．新スケジューラの処理構造

数値風洞では，新スケジューラの機能を実現するための新たな関数を作成し，既存の NQS の関数から必要に応じて呼び出している。本章では，NQSの関数を含むこれらすべての関数から成るプログラムを新スケジューリングプログラムと呼ぶ。以下に，新スケジューリングプログラムの処理構造について説明する。

図 4.1 に新スケジューリングプログラム構成を示す。これらの図は，メインプログラムからどの様な機能を持

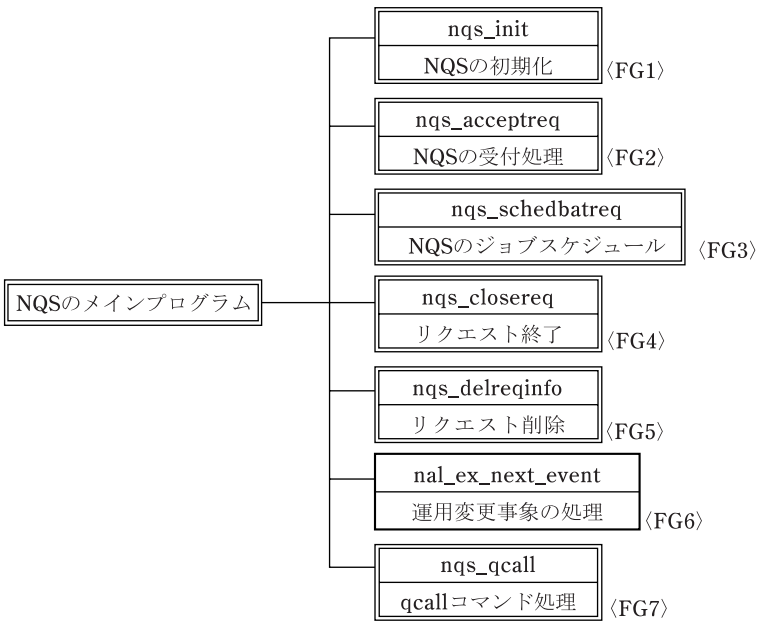

図 4.1　新スケジューリングプログラム構成（メインプログラム関連）

つ関数が呼び出されているかを示している。なお，新スケジューリングプログラムは12個の関数群に分類し，それぞれ FG1 から FG12 という関数群番号を付加している。図 4.2～図 4.9 に各関数群のプログラム構成を示す。図中，二重枠で示されている関数は NQS の関数（関数名は nqs から始まる）であり，すべて関数群の最上位関数となっている。それ以外の関数は航技研計算機システム用の関数（関数名が nwx で始まるものは数値風洞導入時に組み込まれた関数，関数名が nal で始まるものは新スケジューラ用に開発した関数）であり，関数群の最上位関数は太枠で，それ以外は一重枠で示している。

以下に図 4.1 の処理構造に沿って新スケジューリングプログラムの処理内容を簡単に説明する。

4.1 NQS の初期化処理

NQSプログラムの実行開始時に，初期化処理を行うために呼び出される関数群が nqs_init( ) であり，その中から新スケジューラ用の初期化処理を行うための関数群 nal_init( ) を呼び出す。nal_init( ) では，図 4.2 に示される関数を使用して以下の処理を行う。

(1) スケジューラ時刻のセット

現在の日時を調べ，スケジューラ時刻をセットする。スケジューラには，時刻に応じて変化させるべきテーブルがあり，それらのテーブルがいつの時点で更新されたものかを示すためにスケジューラ時刻を使用する。

(2) システム状態情報の初期化

スケジューラでは，FEPが正常に稼働中でない場合や運用切替え中には新たなリクエストを起動しないため，FEPの状態と運用切替え中か否かの情報をもつ。システム状態情報の初期化処理では，両方の状態に起動可能を示す 1 をセットする。

(3) 運用パラメータの読み込み

関数 nal_sys_ param_ read( ) により，変更が許されているスケジューリング変数の内，運用時間帯ごとに設定する必要のない以下の変数を運用パラメータとして読み込む。

① リクエスト管理用パラメータ［第 3 章で述べた WT1F, WT2F, WTSPF, WTUTF の値を指す。］
② 起動リクエスト制限値（1 PE 使用リクエストの同時実行数，同一ユーザリクエストの同時実行数，最大 CPU 時間，最大 PE 台数）［このうち，最大 CPU 時間，最大 PE 台数は，運用時間帯で同項目の制限値が設定されていない場合に使用する。］
③ 経過時間予測ファクタ［この値はリクエストの経過時間を CPU 時間から予測するために使用する。］
④ 数値風洞内投入リクエスト数のユーザ制限値

(4) PCG 情報の読み込み

関数 nal_pcg_ read( ) により，ファイルから運用 PE 電源グループを構成するＰＥ番号の情報を読み込む。この情報を PCG 情報と呼ぶ。

(5) 使用可能運用 PE 台数の設定

関数 nal_pcg_ running_set( ) により，設置されているPEの状態と故障PEの情報を調べ，電源が投入されていて使用できる状態にあり，かつ，故障 PE 情報に登録されていないPE台数を調べ，現在の使用可能運用PE台数を設定する。また，運用 PE 電源グループを構成する

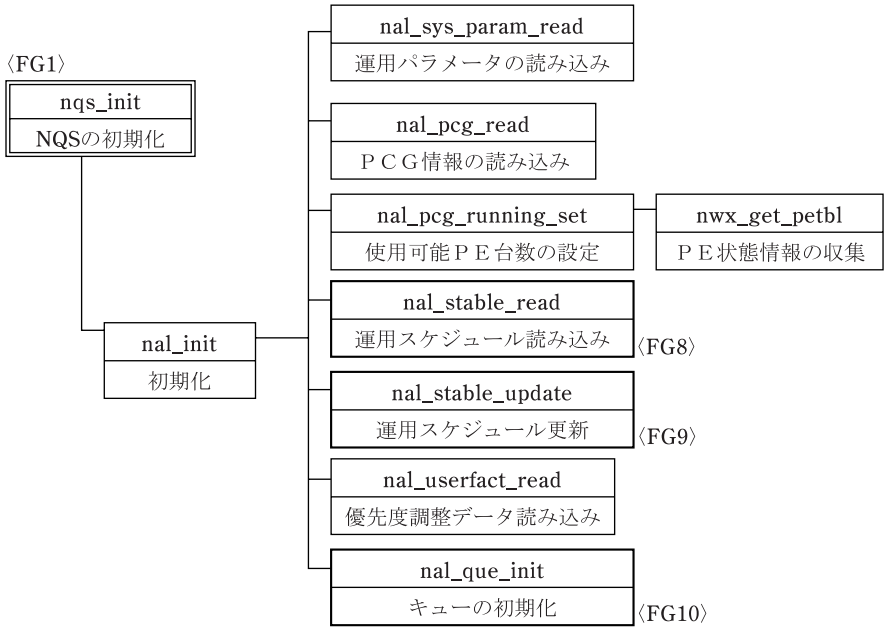

図 4.2 新スケジューリングプログラム構成（初期化関連プログラム）

PE番号と故障PE情報を照合することによりPE電源グループごとの使用可能PE台数を設定する。

(6) 運用スケジュールの読み込み

関数群 nal_stable_read( ) では，以下の処理を行う。なお，処理内容は図 4.9 の FG8 に示す。

① 関数 nal_time_sched_ read( ) により，運用スケジュールファイルから，運用変更日時，起動リクエストの最大CPU時間，最大PE台数，ユーザタイムユーザ名から構成される運用情報を読み込む。数値風洞では，1週間の運用方法がほぼ一定であり，祭日等に特別な運用方法を指定することが多いため，運用時間開始日時は，日付と時刻または曜日と時刻の指定を可能とし，日付情報がある日は日付情報を優先し，日付情報がない日は曜日情報を使用する。

② 関数 nal_sys_ default_ set( ) により，運用情報から，自動化情報の最初のイベント発生時刻における運用情報の値を決定する。これを運用情報初期値と呼ぶ。

③ 関数 nal_sys_ sched_ read( ) により，自動化スケジュールファイルのイベント情報から，自動化スケジュール変更日時，運用 PE 電源グループ番号，クラス毎の多重度を読み込む。また，先に求めたPE電源グループごとの使用可能PE台数から各時間帯の使用可能運用 PE 台数を算出する。これらの情報から自動化情報を作成する。

④ 関数 nal_time_sched_ add( ) により，自動化情報に運用情報を追加し，運用変更日時，使用可能運用ＰＥ台数，クラスごとの多重度，起動リクエスト制限値（最大CPU時間，最大PE台数），ユーザタイムユーザ名から成る運用スケジュールを作成する。なお，運用情報のクラス毎の多重度情報や，自動化情報の起動リクエスト制限等，各情報の未設定の項目は直前の情報により補う。

(7) 運用スケジュールの更新

関数群 nal_stable_update( ) では，図 4.9 の FG9 に示される関数を使用し，以下の処理を行う。

① 前項で作成した運用スケジュール情報を参照して現時点での運用情報をスケジューリングパラメータにセットし，将来の情報をリクエスト起動時等に参照可能な運用テーブルに格納する。

② 関数 nal_ next_event_ set( ) により，ユーザタイムユーザ名の変更のみの運用切替えの事象の予定時刻とその時に呼び出す関数名“nal_ex_next_event”をNQSの関数 nqs_vtimer( ) を使用してセットする。

これは，自動化プログラムの認識する運用切替えの場合には，第4.7項に述べるqcallコマンドを介してスケジューラに制御が渡るが，それ以外はスケジューラに制御が渡らないので，その時刻にNQSから呼び出される様にするためである。すでにセット済の事象予定時刻がある場合には，NQSの関数 nqs_unsetvtimer( ) で予約を取消し，新たな時刻をセットし直す。

(8) 優先度調整データの読み込み

関数 nal_userfact_read( ) により，ユーザ優先度ファイルから優先度を調整するユーザのユーザ名と調整値を読み込む。

(9) 管理キューの初期化

関数群 nal_ que_ init( ) では，図 4.9 の FG10 に示される関数を使用し，以下の処理を行う。

① 既存のwaitキュー，execキュー，rtimeキューの情報をすべてクリアする。

② 関数 nal_special_ info_read( ) により，ファイルに格納されている高優先度リクエスト情報を読み込む。

③ 関数 nal_que_ generate( ) では，関数群 nal_req_add( ) を使用して，NQSに登録されている全てのリクエストを，実行待ちリクエストはwaitキューへ，実行中リクエストはexecキューおよびrtimeキューへ登録する。ここで，高優先度リクエストか否かは高優先度リクエスト情報により判定し，ユーザタイムリクエストか否かは関数 nal_name_chk( ) により判定する。なお，高優先度リクエスト情報には，高優先度リクエストおよびユーザタイムリクエストに関する情報として，高優先度リクエストかユーザタイムリクエストかを判定するための属性，リクエストを識別するためのマシン番号およびシークエンス番号，高優先度化時刻またはユーザタイム開始時刻が格納されている。本処理により高優先度リクエスト情報が変更された場合には，これらの情報がシステムダウン時でも損失する恐れのないように，ファイルへ書き込む。

4. 2 リクエスト受付処理

リクエスト投入時に，NQSのメインプログラムから呼び出される関数群が nqs_acceptreq( ) である。

nqs_ acceptreq ( ) では，図 4.3 に示す関数を使用して以下の処理を行う。

(1) リクエストの受付可否判定

NQSには，投入可能なユーザ名や要求システム資源量制限値を指定し，これにより投入時にリクエストの受付

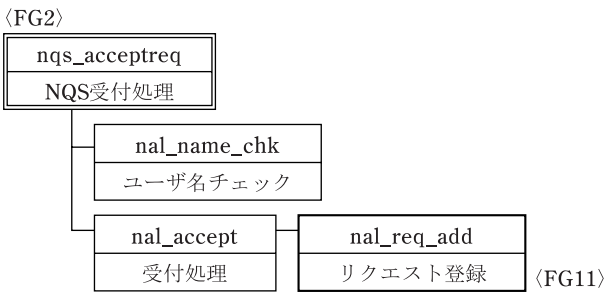

図 4.3 新スケジューリングプログラム構成
（受付処理関連プログラム）

をリジェクトする機能がある。数値風洞では，この機能の他に，ユーザ毎のシステム内投入可能リクエスト数の制限，グループ毎の使用可能ユーザの制限を行っているため，リクエスト投入時にはこれらの制限に基づき受付可否を判定する。受付可能と判断したリクエストのみ以下の受付処理を行う。

(2) リクエストの登録処理

関数nal_accept( )では，数値風洞におけるリクエスト受付に必要な処理を行い，さらに関数群 nal_req_add( ) を使用して wait キューへの登録処理を行う。

(3) 受付リクエストの通知処理

リクエスト情報表示用 TV へ受付リクエスト情報を通知する。

4. 3 リクエストのスケジューリングと実行

スケジューリング契機に，NQSのメインプログラムから呼び出される関数群が nqs_schedbatreq( ) である。NQSでは，システムへの新たなリクエスト投入等によりNQSのキューの状態が変化した場合をスケジューリング契機とみなしている。 nqs_schedbatreq( ) では，図 4.4 に示す関数を使用して以下のリクエストのスケジューリングと実行処理を行う。

(1) 空き PE 台数設定処理

関数 nal_peuse_set( ) では，NQS の実行中キューを参照し，実行中であることが確認された rtime キューのリクエストの PE 台数から現在の空き PE 台数を設定する。リクエスト終了時にはリクエスト終了処理に先立って nqs_schedbatreq( ) が呼び出されるため，本処理が必要となる。

(2) スケジューラ時刻更新処理

スケジューラ内のテーブルは直前の nqs_schedbatreq ( ) が呼ばれた時刻での状態になっている。そのため，関数 nal_time_up( ) では，現在の日時を調べてその差を求め，wait キュー，exec キュー，rtime キューのように時刻で変化するテーブルおよびスケジューラ内の時刻を更新する。

(3) 登録キューの見直し処理

前項の処理によりリクエストの優先度が上がるため，関数 nal_que_change( ) では，各リクエストの所属キューを見直す。本処理では，リクエストがより上位のキューへの遷移条件を満たしたか否かを調べ，条件を満たしたリクエストをつなぎ直す。

(4) スケジュール処理

本処理では，最初にシステムの状態を調べスケジュール可能か否かを判定する。可能と判定した場合には，NQSの全グループに対し，優先度の高いグループから順に実行多重度に余裕があるクラスのリクエストの起動処理を以下のように行う。なお，起動条件および PE リザーブ条件は前章で述べた条件を使用する。

① 関数 nwx_srch_ureq( )，nwx_srch_nreq( )，nal_srch_nreq( ) を使用し，起動条件を満たすリクエストの中で最も優先度の高いリクエストを起動リクエストとして選択する。ここで，PP 方式の場合には優先度順に並んでいるwaitキューを利用し，FCFS 方式の場合には先着順に並んでいるNQSのキューを利用するために，計算機（FEP か CP か）やジョブクラスにより使用する関数を変えている。なお，関数群 nal_exec_chk( ) で数値風洞を使用するリクエストの実行可否を判定するが，本処理においてPP方式の起動優先リクエストが起動不可能と判定された場合には，PE リザーブ条件を満たす最も早い時刻を起動予定時刻とし，その時刻に PE リザーブを行う。

② 前項で，起動可能リクエストが選択された場合には，選択リクエストを起動する。また，当該リクエストを wait キューから exec キューへ移し，さらに実行予定時刻を求め，rtime キューに登録する。

③ リクエスト情報表示用ＴＶへ起動リクエストの情報を通知する。

4. 4 リクエスト終了処理

実行中リクエストの終了時に，NQS のメインプログラムから呼び出される関数群がnqs_closereq( )である。

nqs_closereq( ) では，図 4.5 に示す関数を使用して以下の処理を行う。

(1) リクエスト登録削除処理

正常終了したリクエストの場合には，関数 nal_req_del ( ) を使用して exec キューおよび rtime キューから削除する。また，システム障害等により再実行が必要なリクエストの場合には，関数 nal_que_wait( ) を使用し，exec キューおよび rtime キューから削除し，wait キュー

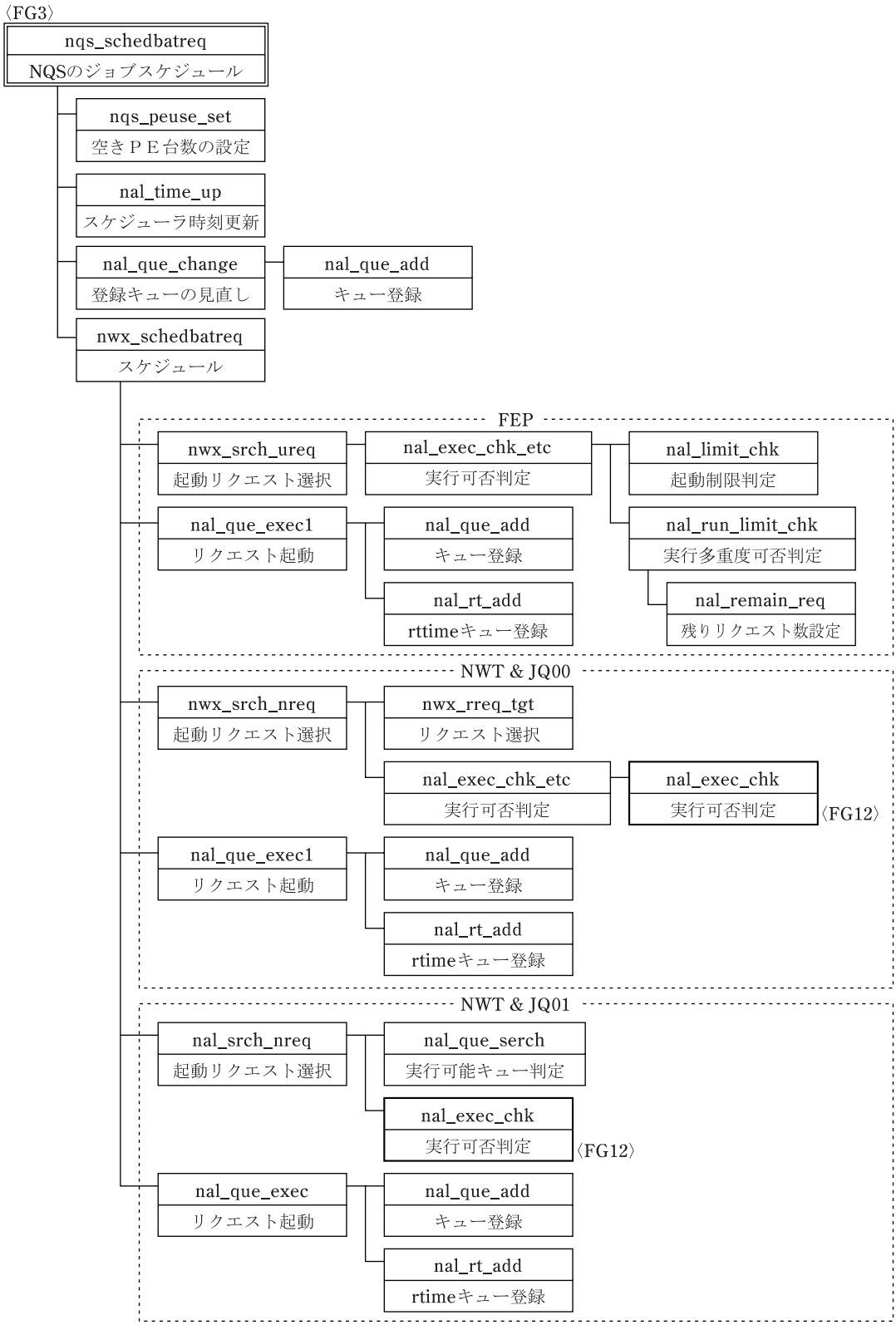

図 4.4　新スケジューリングプログラム構成（スケジュール関連）

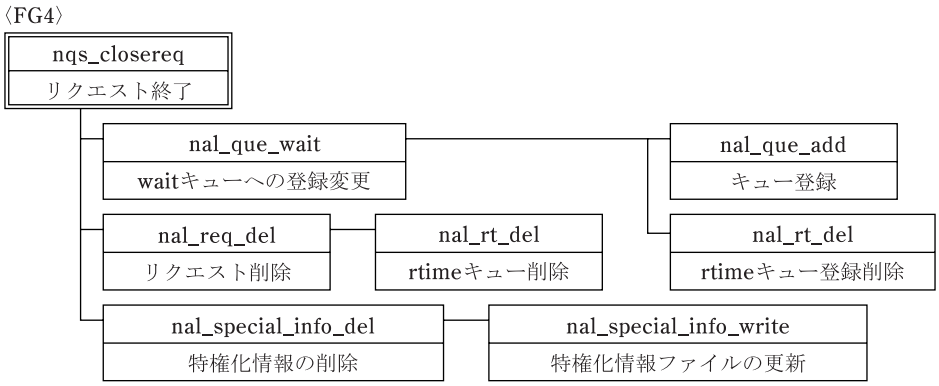

図 4.5　新スケジューリングプログラム構成（リクエスト終了関連プログラム）

につなぎ直す。

(2) 高優先度リクエスト情報の削除処理

高優先度化されたリクエストの情報は「高優先度リクエストテーブル」に格納されているが，nal_special_info_ del( ) では，正常終了したリクエストの情報が高優先度リクエストテーブルにある場合には削除する。

(3) 終了リクエストの通知

リクエスト情報表示用 TV に，正常終了または再実行リクエストの情報を通知する。

4. 5　リクエスト削除処理

リクエストキャンセル時に，NQSのメインプログラムから呼び出される関数群が nqs_delreqinfo( ) である。

nqs_delreqinfo( ) では，図 4.6 に示す関数を使用して以下の処理を行う。

(1) リクエスト登録削除処理

関数 nal_req_del( ) では，実行待ちリクエストの場合にはwaitキューから，実行中リクエストの場合にはexecキューおよび rtime キューから，当該リクエストの情報を削除する。

(2) 高優先度リクエスト情報の削除処理

関数 nal_special_info_del( ) では，高優先度リクエストの情報を格納する高優先度リクエストテーブルの中にキャンセルリクエストの情報がある場合には削除する。

(3) キャンセルリクエストの通知

リクエスト情報表示用 TV に，キャンセルリクエストの情報を通知する。

4. 6　スケジュール事象処理

第 4.1 項 (5) では，ユーザタイムユーザ名の変更のみの運用切替えの事象を処理するために，ＮＱＳ の関数 nqs_vtimer( ) を使用して，事象発生時刻とその時に呼び出す関数名 “ nal_ex_next_event( ) ” をセットすることを説明した。この場合，指定時刻には関数群 nal_ex_next_ event( ) が呼び出される。nal_ex_next_event( ) では，図 4.7 に示す関数を使用して以下の処理を行う。

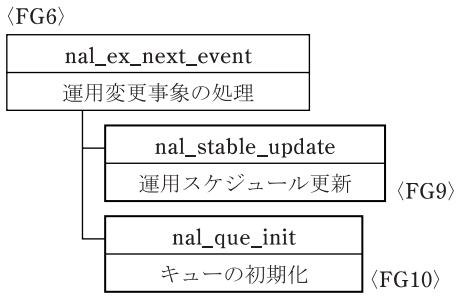

図 4.7　新スケジューリングプログラム構成
（運用変更事象関連プログラム）

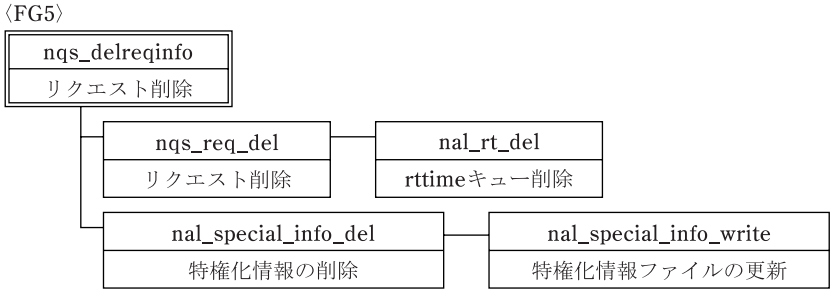

図 4.6　新スケジューリングプログラム構成（リクエスト削除関連プログラム）

(1) 運用スケジュールテーブルの更新処理

関数群 nal_stable_update( ) を使用し，NQSの初期化処理と同様に運用スケジュールテーブルの更新処理を行う。

(2) キューの初期化処理

関数群 nal_que_init( ) を使用し，NQS の初期化処理と同様に管理キューを初期化する。

4. 7　qcall コマンド処理

NQS の関数 nqs_qcall( ) は，図 4.1 から図 4.9 に示した関数からだけでなく，運用管理者がコマンドを投入することにより呼び出すことができる。新スケジューラでは，図 4.8 に示す関数を使用して以下の処理を行う。

(1) システム状況の表示

システムの状態，リクエストの状態が変化した場合にはその都度コンソールに表示することができる。しかし，スケジューラの動きを確認するために，大量の情報の中からそれぞれの最新情報を探し，照らし合わせるのは大変な作業となる。そのため，qcall　コマンドで“ system_ info _list ”と指定した場合には，その時点での種々の情報を指定されたファイルへまとめて出力する。

(2) リクエストの高優先度化処理

qcall　コマンドでリクエスト名を付加して“ chg_special ”と指定した場合には，指定された既存のリクエストを高優先度化する。指示されたリクエストの情報は高優先度リクエストテーブルに追加し，高優先度リクエストとしてwaitキューまたはexecキューにつなぎ直す。

(3) リクエストの非高優先度化処理

qcall　コマンドでリクエスト名を付加して“ chg_general ”と指定した場合には，指定された既存のリクエストを非高優先度化する。指定リクエストがユーザタイムリクエストの場合には，高優先度リクエストテーブルの当該リクエストの情報の属性を高優先度リクエストからユーザタイムリクエストに替え，ユーザタイムリクエストとしてwaitキューまたはexecキューにつなぎ直す。ユーザタイムリクエストでない場合には，指示されたリクエストの情報を高優先度リクエストテーブルから削除し，非高優先度リクエストとしてwaitキューまたはexecキューにつなぎ直す。

(4) ユーザ優先度情報変更処理

ユーザの優先度情報を更新した場合には，変更内容を運用に反映させるために，変更処理が必要となる。qcall コマンドで“ user_fact_read ”と指定すると，NQS の初期化処理の場合と同様に，関数 nal_userfact_read( ) を使用してファイルからユーザ優先度データを読み込み，スケジューラのユーザ優先度情報を更新する。また，関数 nal_que_init( ) を使用してシステム内のすべてのリクエストをこの優先度情報を使用して再登録する。なお，qcall コマンドで“ que_init ”と指定した場合には，システム内のすべてのリクエストの再登録のみを実行することができる。

(5) 運用パラメータ変更処理

qcall コマンドで“ sys_param ”と指定した場合には，nal_sys_ param_ read( )を使用してファイルから運用パラメータを読み込み，スケジューラの運用パラメータを更新する。運用パラメータを変更した場合に使用する。

(6) 運用スケジュール変更処理

qcall コマンドで“ sch_update ”と指定した場合には，NQS の初期化処理に行った運用スケジュールの読み込み，運用スケジュールの更新処理を行う。運用スケジュールを変更した場合に使用する。

(7) 故障 PE 変更処理

qcall コマンドで“ pe_trouble ”と指定した場合には，nal_pcg_running_set( ) を使用した使用可能ＰＥ台数の設定，nal_sctbl_read( ) を使用した運用スケジュールの読み込み，nal_sctbl_update( )を使用した運用スケジュールの更新，nal_que_init( ) を使用したキューの初期化処理を行う。本処理は，故障 PE の発生，故障 PE の回復等，故障 PE の情報が変化した場合に使用する。

(8) システム状態変更処理

スケジューラは，スケジューリングの可否を判定するためにFEPの状態と運用切替え中か否かの情報を持つ。qcall コマンドで“ msp_abnomal ”と指定した場合にはFEPの状態を起動不可を示す０を，“ msp_nomal ”と指定した場合にはFEPの状態を起動可を示す１を，“ chg_start ”と指定した場合には，運用切替え中のための起動不可を示す０をそれぞれ関数 nal_state( ) を使用してセットする。また，“ chg_end ”と指定した場合には運用切替え処理終了とみなし，故障 PE 変更処理と同様の処理を行い，最後に関数 nal_state( ) を使用して運用切替え処理終了による起動可能を示す１をセットする。

５．新スケジューラの有効性検証

文献１で述べたように，新スケジューラの基本機能の開発に際して，シミュレータを作成してその有効性を検証した。しかし，その時に使用したデータはあくまで航技研のジョブミックスから推定したものであり，新スケジューラでは新たな機能を付加したことから，現システムから採取した稼働状況データに基づき実運用における有効性を以下の観点から検証する。

① PE の高効率利用がなされているか。
② パラメータの設定が適切であるか。
③ PE リザーブによる他リクエストへの影響が多くないか。

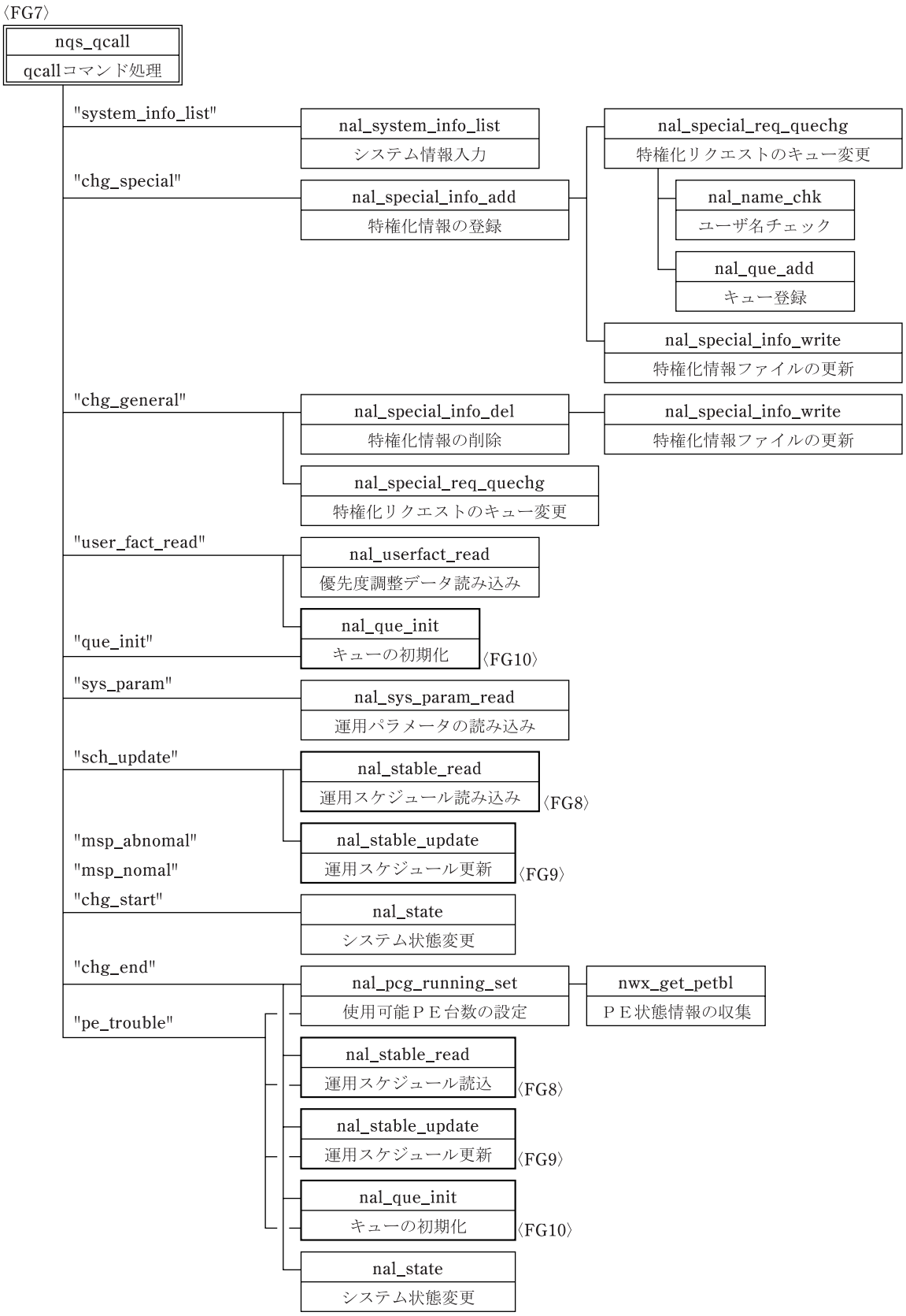

図 4.8　新スケジューリングプログラム構成（qcall 処理関連プログラム）

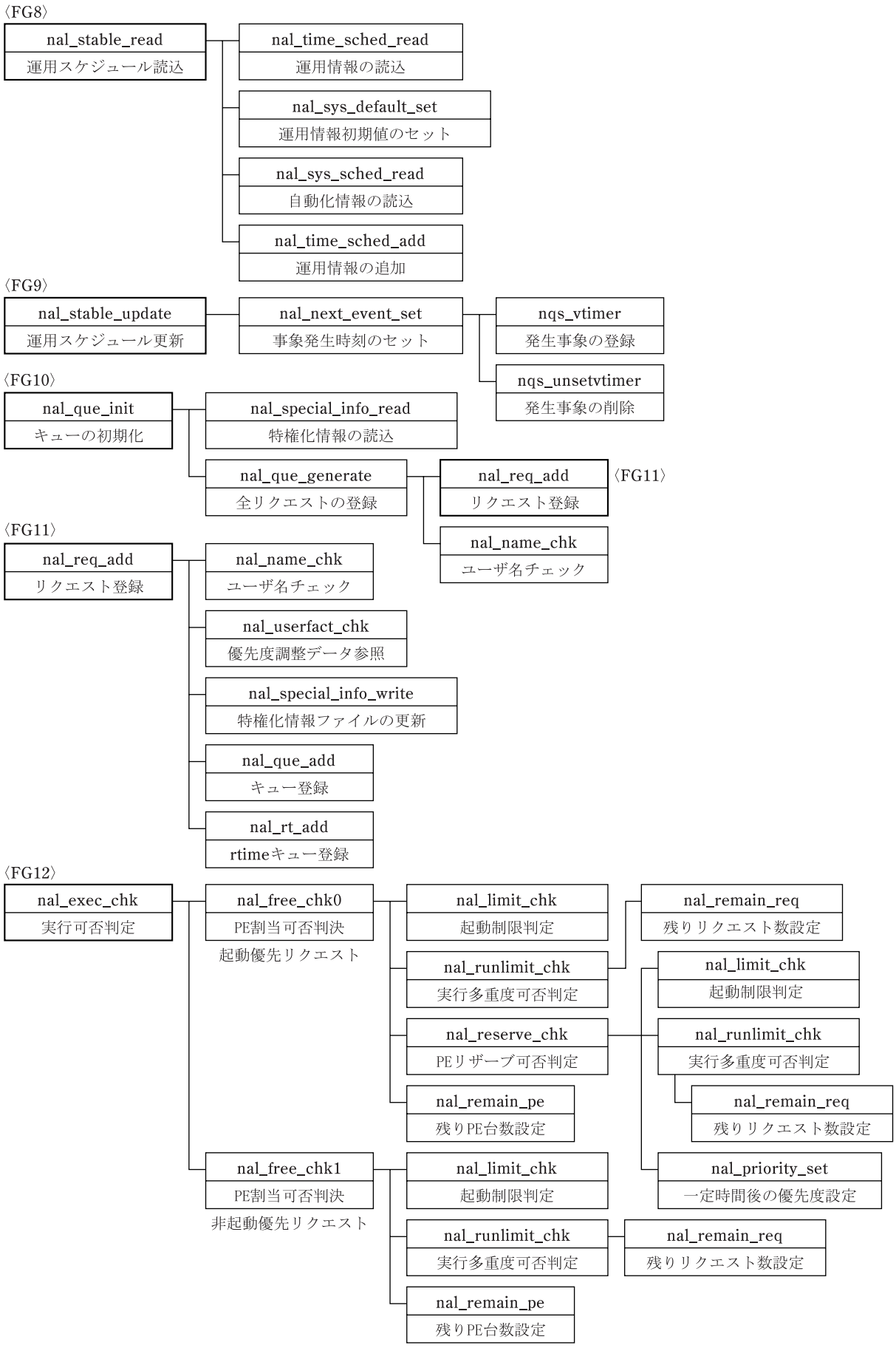

図 4.9　新スケジューリングプログラム構成（その他）

④ 起動優先リクエストの起動までに長時間かかっていないか。
⑤ リミット値による影響は大きくないか。
⑥ スワップアウトの頻発がないか。

なお，1日分のデータは0時から23時までの24回採取するものとし，データ採取期間は平成9年5月15日～5月21日の7日間とした。また，システムの運用PE台数は電源グループ単位で増減できるが，ここではすべてのPEを使用して，JQ01用162台，JQ02用4台，合計166台で運用した。

5.1 現システムの稼働状況

表5.1～表5.7に，7日間の稼働状況を示す。稼働状況は表中の分類に合わせてPE利用状況と実行待ちリクエスト状況に分けて考察する。

5.1.1 PE利用状況

PE使用状況は，使用可能PE台数（運用PE台数から故障等により使用不可能なＰＥ台数を除いたもの）と使用PE台数を“jq01”,“jq02”,“合計”の3つに分けて示した。このうち，“jq01”はメモリ量が256MBのPEを使用するJQ01のPE台数，“jq02”はメモリ量が1GBのPEを使用するJQ02のPE台数，“合計”はJQ01，JQ02の合計値である。表の下段にその日の平均値と平均利用率を示す。

表5.1～表5.7を見ると，システム全体のPE利用率は93.9% ～96.9%と極めて高く，かつ安定しており，新スケジューラの第一目標であるPEの高効率利用が達成されていることが確認できる。クラス毎のPE使用率は，JQ01は95.4%～98.1%と高いものの，JQ02は0.0%～57.3%と低い値を示している。この様な状況から，PE台数の多いJQ01の利用率がシステム全体の利用率を支えていることが分かる。

JQ01のPEが高利用されている要因は，設置PE台数が多く投入されるリクエスト規模の種類も多いためスケジューリングの選択に幅ができたためと考えられる。一方，平均利用率の低いJQ02の場合，使用PE台数0が連続するのは17日の7:00から18日の23:00，19日の23:00から20日の9:00までであり，休日や夜間にJQ02リクエストがなくなったことが原因となっている。

5.1.2 実行待ちリクエスト状況

実行待ちリクエストは，クラスごとに実行不可条件で分類し，リクエスト数，PE台数を示している。以下に項目の説明を行う。

ge1：JQ01の非優先リクエスト数
ge2：JQ01の優先リクエスト数
ge3：JQ01の最優先リクエスト数
min：JQ01の最小PE台数
max：JQ01の最大PE台数
jq02：JQ02の総リクエスト数

以下に実行不可条件を示す。なお，実行不可条件は①から順に判定し，ある条件に該当すればその項目で分類し，すべての条件をクリアしたリクエストは実行に移る。

① 先行ステップ実行待ち：ジョブステップの逐次実行を行っているジョブで，先行リクエストの処理が終了しないため実行不可能と判定されたリクエスト。
② リミット値による実行制限：1 PEリクエストの実行数制限，同一ユーザの使用PE台数制限，リクエストの最大使用PE台数制限等により実行不可能と判定されたリクエスト。
③ PE割り当て不可［空きPE］：空きPE台数が足りないために要求PE台数の割り当てができないと判定されたリクエスト。
④ PE割り当て不可［PEリザーブ］：要求台数以上の空きPEがあるにもかかわらず，PEリザーブリクエストがあるために要求PE台数の割り当てができないと判定されたリクエスト。

また，実行待ちリクエストの表の中で，すべてのリクエスト数が示されていない部分がある。これは，FEP異常時やスケジューリング切り換え時のためスケジューリング処理が停止されている状態か，あるいはスケジューラに制御が渡っているが各リクエストの起動判定処理にまで至っていない状態を示している。データ採取は，スケジューラの状態とは関係なく処理するため，このような状況が発生する。

実行待ちリクエストに関しては,クラス毎に考察する。

(1) JQ01クラスの場合

① 先行ステップ実行待ち

先行ステップ実行待ちの場合には，他の分類に比べてリクエスト数がかなり多いことが特徴となっている。このことは，ジョブステップの逐次実行を行っているジョブが多いということを示している。しかし，これらのリクエストは，先行ステップが終了すればスケジューリング対象になるため，システムの有効利用という観点から考えるとある程度の数があることはむしろ好ましい。ただし，最優先リクエストになった場合には，先行リクエストが終了すれば後続のリクエストは待たずに実行に移る可能性が高くなり,CPU打ち切り値よりも長いジョブを投入したことと同様になる。そのため，最優先リクエスト数が多くなることは好ましくないが，多い日でも平均2本以下，7日間の平均は1本以下であり，極端に多いとは考えられない。

表5.1　数値風洞稼働記録［平成９年５月15日（木)］

時刻	ＰＥ利用状況						実行待ちリクエスト状況																			
	使用可能ＰＥ台数			使用ＰＥ台数			先行リクエスト実行待ち				リミット値による実行制限					PE割り当て不可［空きPE］					PE割り当て不可［PEリザーブ］					
	jq01	jq02	合計	jq01	jq02	合計	ge1	ge2	ge3	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	max	jq02
00:00	162	4	166	120	3	123	188	15	0	3	0	0	0	0	0	5	4	2	49	0	20	2	0	6	24	0
01:00	162	4	166	160	3	163	186	4	4	3	0	0	0	0	0	25	2	0	6	0	0	0	0	0	0	0
02:00	162	4	166	157	3	160	183	4	4	3	0	0	0	0	0	22	2	0	6	0	0	0	0	0	0	0
03:00	162	4	166	161	2	163	178	4	0	2	0	0	0	0	0	18	2	0	6	0	0	0	0	0	0	0
04:00	162	4	166	159	2	161	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
05:00	162	4	166	157	2	159	168	5	0	2	1	0	0	50	0	15	0	0	6	0	0	0	0	0	0	0
06:00	162	4	166	161	2	163	167	5	0	2	1	0	0	50	0	13	0	0	6	0	0	0	0	0	0	0
07:00	162	4	166	160	2	162	163	5	0	2	1	0	0	50	0	12	0	0	6	0	0	0	0	0	0	0
08:00	162	4	166	160	1	161	162	4	0	1	1	0	0	50	0	10	0	0	6	0	0	0	0	0	0	0
09:00	162	4	166	153	2	155	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
10:00	162	4	166	162	3	165	182	2	0	4	0	0	0	0	0	46	2	0	6	0	0	0	0	0	0	0
11:00	162	4	166	162	3	165	217	1	3	4	0	0	0	0	0	28	2	1	1	0	0	0	0	0	0	0
12:00	162	4	166	141	4	145	222	1	5	3	3	0	0	1	0	11	0	1	24	1	13	0	0	10	16	0
13:00	162	4	166	162	3	165	215	1	0	3	9	0	0	14	0	14	2	1	1	0	0	0	0	0	0	0
14:00	162	4	166	159	3	162	243	3	0	2	9	0	0	14	0	26	2	0	6	0	0	0	0	0	0	0
15:00	162	4	166	154	3	157	186	58	0	2	10	10	0	1	0	12	2	0	16	0	0	0	0	0	0	0
16:00	162	4	166	158	2	160	191	54	5	2	9	0	0	1	0	19	14	1	6	0	0	0	0	0	0	0
17:00	162	4	166	160	2	162	185	56	2	2	8	9	0	1	0	18	4	1	6	0	0	0	0	0	0	0
18:00	162	4	166	161	2	163	184	50	10	1	10	11	0	1	0	18	3	1	6	0	0	0	0	0	0	0
19:00	162	4	166	162	1	163	180	54	1	1	11	9	0	1	0	16	2	0	6	0	0	0	0	0	0	0
20:00	162	4	166	159	1	160	176	48	0	1	8	9	0	1	0	11	2	0	16	0	0	0	0	0	0	0
21:00	161	4	165	159	2	161	167	51	0	1	5	9	0	1	0	15	6	0	8	0	0	0	0	0	0	0
22:00	161	4	165	158	2	160	162	53	3	0	2	10	1	12	0	13	8	1	8	0	0	0	0	0	0	0
23:00	161	4	165	151	2	153	155	50	0	4	0	12	0	12	0	11	5	0	16	0	0	0	0	0	0	0
	161.9	4.0	165.9	156.5 96.7	2.3 57.3	158.8 95.7	184.5	24.0	1.7	2.2	4.0	3.6	0.0	11.8	0.0	17.2	2.9	0.4	9.9	0.0	1.5	0.1	0.0	0.7	1.8	0.0

表 5.2　数値風洞稼働記録［平成９年５月 16 日（金)］

時刻	ＰＥ利用状況						実行待ちリクエスト状況																		
	使用可能ＰＥ台数			使用ＰＥ台数			先行リクエスト実行待ち				リミット値による実行制限					PE 割り当て不可［空き PE］					PE 割り当て不可［PE リザーブ］					
	jq01	jq02	合計	jq01	jq02	合計	ge1	ge2	ge3	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	max	jq02
00:00	161	4	165	161	2	163	144	45	0	4	0	12	0	12	0	25	6	0	6	0	0	0	0	0	0	0
01:00	161	4	165	162	2	164	159	42	0	4	0	12	0	12	0	23	4	0	6	0	0	0	0	0	0	0
02:00	161	4	165	157	2	159	153	45	0	4	0	12	0	12	0	17	4	0	6	0	0	0	0	0	0	0
03:00	161	4	165	148	2	150	151	42	0	3	0	10	0	12	0	11	5	3	16	0	0	0	0	0	0	0
04:00	161	4	165	162	2	164	146	40	0	2	0	0	0	0	0	11	12	0	14	0	0	0	0	0	0	0
05:00	161	4	165	158	2	160	141	30	6	2	0	9	0	14	0	16	1	0	6	0	0	0	0	0	0	0
06:00	161	4	165	159	1	160	139	29	4	2	0	9	0	14	0	13	1	0	6	0	0	0	0	0	0	0
07:00	161	4	165	151	1	152	136	28	0	2	0	9	0	14	0	9	1	0	24	0	0	0	0	0	0	0
08:00	161	4	165	156	1	157	135	27	0	1	0	9	0	14	0	8	1	0	24	0	0	0	0	0	0	0
09:00	161	4	165	141	1	142	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
10:00	161	4	165	161	1	162	134	27	0	1	0	9	0	12	0	27	2	1	1	0	0	0	0	0	0	0
11:00	161	4	165	157	4	161	157	28	8	1	0	0	0	0	0	25	17	0	6	0	0	0	0	0	0	0
12:00	161	4	165	158	2	160	159	24	0	1	2	7	0	25	0	21	10	0	6	0	0	0	0	0	0	0
13:00	161	4	165	161	4	165	137	26	0	0	2	7	0	14	0	13	10	0	16	0	0	0	0	0	0	0
14:00	161	4	165	160	2	162	154	23	0	0	3	8	0	14	0	10	4	17	4	0	0	0	0	0	0	0
15:00	154	4	158	144	2	146	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
16:00	162	4	166	160	3	163	147	58	3	0	2	7	0	14	0	30	6	0	6	0	0	0	0	0	0	0
17:00	162	4	166	157	4	161	143	57	3	0	0	0	0	0	0	28	22	0	6	3	0	0	0	0	0	0
18:00	162	4	166	159	3	162	164	54	2	7	0	9	0	14	0	22	16	0	6	1	0	0	0	0	0	1
19:00	162	4	166	141	4	145	121	51	2	0	0	9	0	14	0	17	16	0	6	0	0	0	0	0	0	0
20:00	162	4	166	142	4	146	150	50	2	7	0	9	0	14	0	10	18	0	24	3	0	0	0	0	0	0
21:00	162	4	166	161	1	162	147	47	0	7	2	11	0	14	0	14	4	0	1	0	0	0	0	0	0	0
22:00	162	4	166	148	2	150	143	45	0	6	1	11	0	1	0	10	4	0	24	0	0	0	0	0	0	0
23:00	162	4	166	159	2	161	140	40	0	6	0	8	0	14	0	10	4	0	10	0	0	0	0	0	0	0
	161.0	4.0	165.0	155.1 96.3	2.3 56.3	157.4 95.4	145.5	39.0	1.4	2.7	0.5	8.0	0.0	11.5	0.0	16.8	7.6	1.0	10.2	0.3	0.0	0.0	0.0	0.0	0.0	0.0

表 5.3　数値風洞稼働記録［平成９年５月 17 日（土)］

時刻	ＰＥ利用状況						実行待ちリクエスト状況																		
	使用可能ＰＥ台数			使用ＰＥ台数			先行リクエスト実行待ち				リミット値による実行制限					PE 割り当て不可［空き PE］					PE 割り当て不可［PE リザーブ］					
	jq01	jq02	合計	jq01	jq02	合計	ge1	ge2	ge3	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	max	jq02
00:00	162	4	166	151	2	153	135	39	0	6	0	8	0	14	0	9	4	0	16	0	0	0	0	0	0	0
01:00	162	4	166	160	2	162	130	35	0	6	0	9	0	14	0	19	2	2	4	0	0	0	0	0	0	0
02:00	162	4	166	159	2	161	128	32	0	6	0	9	0	14	0	24	2	0	6	0	0	0	0	0	0	0
03:00	162	4	166	158	2	160	121	28	0	5	0	9	0	14	0	23	1	0	6	0	0	0	0	0	0	0
04:00	162	4	166	158	2	160	119	25	0	5	0	8	0	14	0	22	1	0	6	0	0	0	0	0	0	0
05:00	162	4	166	156	1	157	108	15	7	0	0	8	0	14	0	21	1	0	6	0	0	0	0	0	0	0
06:00	162	4	166	159	1	160	101	15	4	0	0	8	0	14	0	16	1	0	6	0	0	0	0	0	0	0
07:00	162	4	166	161	0	161	97	15	3	0	0	7	0	14	0	11	1	0	6	0	0	0	0	0	0	0
08:00	162	4	166	148	0	148	97	15	0	0	0	6	0	14	0	8	1	0	24	0	0	0	0	0	0	0
09:00	162	4	166	153	0	153	94	12	0	0	0	6	0	14	0	7	1	0	32	0	0	0	0	0	0	0
10:00	162	4	166	134	0	134	94	12	0	0	0	6	0	14	0	7	1	0	32	0	0	0	0	0	0	0
11:00	162	4	166	162	0	162	95	11	0	0	2	6	0	14	0	32	0	0	1	0	0	0	0	0	0	0
12:00	162	4	166	157	0	157	104	11	0	0	2	8	0	14	0	29	0	1	6	0	0	0	0	0	0	0
13:00	162	4	166	157	3	160	100	10	0	0	0	8	0	14	0	22	0	0	6	0	0	0	0	0	0	0
14:00	162	4	166	159	0	159	91	9	0	0	0	5	0	14	0	9	1	0	24	0	0	0	0	0	0	0
15:00	162	4	166	159	0	159	118	9	0	0	0	5	0	14	0	10	0	0	6	0	0	0	0	0	0	0
16:00	162	4	166	162	3	165	117	9	0	0	0	5	0	14	0	22	2	0	6	0	0	0	0	0	0	0
17:00	162	4	166	162	3	165	121	9	2	0	0	5	0	14	0	28	4	0	6	0	0	0	0	0	0	0
18:00	162	4	166	160	0	160	115	21	0	0	0	0	0	0	0	26	7	0	6	0	0	0	0	0	0	0
19:00	162	4	166	160	0	160	111	21	0	0	4	4	0	6	0	13	5	2	4	0	0	0	0	0	0	0
20:00	162	4	166	159	0	159	104	24	0	0	0	4	0	14	0	16	3	0	6	0	0	0	0	0	0	0
21:00	162	4	166	160	0	160	101	19	0	0	0	7	0	14	0	12	0	6	4	0	0	0	0	0	0	0
22:00	162	4	166	159	0	159	99	26	0	0	0	4	0	14	0	28	3	0	6	0	0	0	0	0	0	0
23:00	162	4	166	162	0	162	101	30	0	0	0	7	0	14	0	24	2	9	4	0	0	0	0	0	0	0
	162.0	4.0	166.0	157.3 97.1	0.9 21.9	158.2 95.3	108.4	18.8	0.7	1.2	0.3	6.3	0.0	13.1	0.0	18.3	1.8	0.8	9.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0

表 5.4　数値風洞稼働記録［平成９年５月 18 日（日）］

時刻	ＰＥ利用状況						実行待ちリクエスト状況																		
	使用可能ＰＥ台数			使用ＰＥ台数			先行リクエスト実行待ち				リミット値による実行制限					PE 割り当て不可［空き PE］					PE 割り当て不可［PE リザーブ］					
	jq01	jq02	合計	jq01	jq02	合計	ge1	ge2	ge3	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	max	jq02
00:00	162	4	166	159	0	159	107	25	0	0	1	4	0	6	0	16	7	0	6	0	0	0	0	0	0	0
01:00	162	4	166	157	0	157	105	23	0	0	0	4	0	14	0	10	5	0	20	0	0	0	0	0	0	0
02:00	162	4	166	160	0	160	95	31	0	0	0	5	0	6	0	9	6	11	4	0	0	0	0	0	0	0
03:00	162	4	166	160	0	160	95	31	0	0	0	7	0	14	0	20	2	0	6	0	0	0	0	0	0	0
04:00	162	4	166	160	0	160	91	30	0	0	0	7	0	14	0	13	2	0	6	0	0	0	0	0	0	0
05:00	162	4	166	160	0	160	83	35	0	0	0	6	0	14	0	15	2	0	6	0	0	0	0	0	0	0
06:00	162	4	166	144	0	144	83	32	0	0	0	6	0	14	0	9	2	0	24	0	0	0	0	0	0	0
07:00	162	4	166	162	0	162	83	31	0	0	0	6	0	14	0	18	2	0	6	0	0	0	0	0	0	0
08:00	162	4	166	162	0	162	83	31	0	0	0	6	0	14	0	19	2	0	6	0	0	0	0	0	0	0
09:00	162	4	166	154	0	154	78	28	0	0	0	2	0	14	0	9	7	0	20	0	0	0	0	0	0	0
10:00	162	4	166	158	0	158	78	27	0	0	0	5	0	14	0	28	3	1	6	0	0	0	0	0	0	0
11:00	162	4	166	128	0	128	78	27	0	0	0	2	1	36	0	7	0	1	36	0	17	3	0	6	32	0
12:00	162	4	166	158	0	158	78	26	0	0	0	4	0	20	0	32	1	0	6	0	0	0	0	0	0	0
13:00	162	4	166	158	0	158	74	26	0	0	0	2	0	32	0	24	7	0	6	0	0	0	0	0	0	0
14:00	162	4	166	162	0	162	71	25	0	0	0	2	0	32	0	19	3	0	6	0	0	0	0	0	0	0
15:00	162	4	166	160	0	160	71	23	0	0	1	1	0	6	0	15	2	0	6	0	0	0	0	0	0	0
16:00	160	4	164	148	0	148	71	21	0	0	0	1	0	55	0	8	2	0	24	0	0	0	0	0	0	0
17:00	160	4	164	156	0	156	71	20	0	0	0	1	0	55	0	7	1	0	24	0	0	0	0	0	0	0
18:00	160	4	164	161	0	161	69	19	0	0	0	0	0	0	0	12	3	17	4	0	2	0	0	1	1	0
19:00	160	4	164	161	0	161	67	17	0	0	0	0	0	0	0	24	2	0	6	0	0	0	0	0	0	0
20:00	160	4	164	159	0	159	67	17	0	0	0	1	0	55	0	24	1	0	6	0	0	0	0	0	0	0
21:00	160	4	164	131	0	131	66	15	0	0	3	0	0	6	0	6	1	1	50	0	2	2	0	14	24	0
22:00	160	4	164	161	0	161	66	13	0	0	2	1	0	50	0	14	0	0	6	0	0	0	0	0	0	0
23:00	160	4	164	159	0	159	66	9	0	0	0	1	0	55	0	22	1	0	6	0	0	0	0	0	0	0
	161.3	4.0	165.3	155.8 96.5	0.0 0.0	155.8 94.2	79.0	24.3	0.0	0.0	0.3	3.1	0.0	22.5	0.0	15.8	2.7	1.3	12.3	0.0	0.9	0.2	0.0	0.9	2.4	0.0

表 5.5　数値風洞稼働記録［平成９年５月 19 日（月）］

時刻	ＰＥ利用状況						実行待ちリクエスト状況																		
	使用可能ＰＥ台数			使用ＰＥ台数			先行リクエスト実行待ち				リミット値による実行制限					PE 割り当て不可［空き PE］					PE 割り当て不可［PE リザーブ］					
	jq01	jq02	合計	jq01	jq02	合計	ge1	ge2	ge3	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	max	jq02
00:00	160	4	164	162	1	163	70	8	0	1	0	1	0	55	0	20	1	0	1	0	0	0	0	0	0	0
01:00	160	4	164	160	1	161	73	8	0	3	1	1	0	6	0	7	1	0	24	0	0	0	0	0	0	0
02:00	160	4	164	136	1	137	73	7	0	3	0	1	0	55	0	6	1	0	50	0	0	0	0	0	0	0
03:00	160	4	164	158	1	159	73	7	0	3	0	1	0	55	0	16	1	0	6	0	0	0	0	0	0	0
04:00	160	4	164	128	1	129	72	6	0	3	2	0	0	6	0	6	2	0	50	0	0	0	0	0	0	0
05:00	160	4	164	142	1	143	67	6	0	3	2	0	0	50	0	4	1	0	55	0	0	0	0	0	0	0
06:00	160	4	164	144	1	145	66	0	0	2	0	0	0	0	0	5	0	0	50	0	0	0	0	0	0	0
07:00	160	4	164	162	1	163	66	0	0	2	0	0	0	0	0	19	0	0	6	0	0	0	0	0	0	0
08:00	160	4	164	161	1	162	65	0	0	2	0	0	0	0	0	14	0	0	6	0	0	0	0	0	0	0
09:00	161	4	165	158	1	159	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
10:00	161	4	165	149	1	150	104	0	1	1	1	0	0	50	0	17	0	3	14	0	0	0	0	0	0	0
11:00	161	4	165	151	2	153	108	0	2	1	1	0	0	50	0	17	2	1	14	0	0	0	0	0	0	0
12:00	161	4	165	148	2	150	134	0	0	1	9	0	0	14	0	10	3	0	16	0	0	0	0	0	0	0
13:00	161	4	165	161	3	164	112	18	0	1	11	0	0	14	0	10	0	16	4	0	2	0	0	1	1	0
14:00	161	4	165	158	3	161	103	21	0	1	10	1	0	14	0	28	2	1	6	0	0	0	0	0	0	0
15:00	161	4	165	161	3	164	97	25	0	0	2	9	0	14	0	28	0	0	6	0	0	0	0	0	0	0
16:00	161	4	165	161	1	162	86	28	0	0	2	12	0	14	0	23	0	1	6	0	0	0	0	0	0	0
17:00	161	4	165	159	2	161	83	27	0	0	0	11	0	14	0	16	3	0	6	0	0	0	0	0	0	0
18:00	161	4	165	159	2	161	76	29	0	0	0	11	0	14	0	14	3	1	6	0	0	0	0	0	0	0
19:00	161	4	165	158	1	159	74	25	0	0	1	9	0	12	0	7	6	0	20	0	0	0	0	0	0	0
20:00	161	4	165	161	2	163	89	23	0	0	1	9	0	12	0	21	6	0	6	0	0	0	0	0	0	0
21:00	161	4	165	159	1	160	88	20	0	0	5	9	0	1	0	17	6	1	6	0	0	0	0	0	0	0
22:00	161	4	165	160	1	161	86	20	0	0	3	9	0	12	0	14	6	0	6	0	0	0	0	0	0	0
23:00	161	4	165	160	0	160	69	41	0	0	3	11	0	12	0	9	2	0	6	0	0	0	0	0	0	0
	160.6	4.0	164.6	154.8 96.4	1.4 35.4	156.3 94.9	84.1	13.9	0.1	1.2	2.3	4.1	0.0	20.6	0.0	14.3	2.0	1.0	16.1	0.0	0.1	0.0	0.0	0.0	0.0	0.0

表 5.6　数値風洞稼働記録［平成 9 年 5 月 20 日（火）］

時刻	PE利用状況						実行待ちリクエスト状況																			
	使用可能PE台数			使用PE台数			先行リクエスト実行待ち				リミット値による実行制限					PE割り当て不可［空きPE］					PE割り当て不可［PEリザーブ］					
	jq01	jq02	合計	jq01	jq02	合計	ge1	ge2	ge3	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	max	jq02
00:00	161	4	165	159	0	159	56	46	0	0	2	12	0	12	0	6	1	0	24	0	0	0	0	0	0	0
01:00	161	4	165	158	0	158	52	43	0	0	2	12	0	12	0	20	1	0	6	0	0	0	0	0	0	0
02:00	161	4	165	161	0	161	50	44	0	0	0	14	0	12	0	15	1	0	6	0	0	0	0	0	0	0
03:00	161	4	165	160	0	160	31	59	0	0	0	8	0	12	0	8	6	0	6	0	0	0	0	0	0	0
04:00	161	4	165	152	0	152	30	58	0	0	0	10	0	12	0	5	3	0	16	0	0	0	0	0	0	0
05:00	161	4	165	150	0	150	22	57	0	0	0	9	0	12	0	4	3	0	24	0	0	0	0	0	0	0
06:00	161	4	165	143	0	143	21	56	0	0	0	8	0	12	0	3	3	0	50	0	0	0	0	0	0	0
07:00	161	4	165	141	0	141	21	56	0	0	0	8	0	12	0	3	3	0	50	0	0	0	0	0	0	0
08:00	161	4	165	133	0	133	21	55	0	0	0	4	0	12	0	3	3	0	50	0	0	0	0	0	0	0
09:00	161	4	165	142	0	142	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
10:00	161	4	165	159	1	160	23	44	21	0	7	4	0	12	0	3	3	1	20	0	0	0	0	0	0	0
11:00	161	4	165	161	2	163	35	43	12	0	7	3	1	12	0	4	4	1	8	0	0	0	0	0	0	0
12:00	161	4	165	161	3	164	61	51	8	0	0	9	0	12	0	3	5	7	4	0	8	0	0	1	50	0
13:00	161	4	165	159	3	162	34	86	2	0	1	0	0	64	0	21	13	1	6	0	0	0	0	0	0	0
14:00	161	4	165	161	3	164	39	74	0	0	3	4	0	10	0	16	10	4	1	0	0	0	0	0	0	0
15:00	161	4	165	156	2	158	51	70	1	0	2	6	0	6	0	6	5	0	9	0	0	0	0	0	0	0
16:00	161	4	165	161	1	162	37	75	0	0	0	6	0	14	0	2	6	0	20	0	0	0	0	0	0	0
17:00	161	4	165	160	1	161	52	73	0	0	7	7	0	1	0	3	5	10	4	0	0	0	0	0	0	0
18:00	161	4	165	159	0	159	79	66	0	0	1	6	0	14	0	18	10	0	6	0	0	0	0	0	0	0
19:00	161	4	165	158	4	162	69	71	0	0	3	6	0	6	0	5	10	0	10	6	0	0	0	0	0	0
20:00	161	4	165	148	4	152	75	65	0	0	1	6	0	14	0	5	9	0	18	4	0	0	0	0	0	0
21:00	161	4	165	154	3	157	67	56	0	0	1	6	0	14	0	3	4	0	24	0	0	0	0	0	0	0
22:00	161	4	165	150	2	152	71	52	0	0	1	0	0	55	0	3	10	0	14	0	0	0	0	0	0	0
23:00	161	4	165	142	2	144	59	59	0	0	0	4	0	14	0	4	4	0	20	0	0	0	0	0	0	0
	161.0	4.0	165.0	153.7 95.4	1.3 32.3	155.0 93.9	45.9	59.1	1.9	0.0	1.7	6.6	0.0	15.5	0.0	7.1	5.3	1.0	17.2	0.4	0.3	0.0	0.0	0.0	2.2	0.0

表5.7　数値風洞稼働記録［平成９年５月21日（水）］

時刻	ＰＥ利用状況						実行待ちリクエスト状況																			
	使用可能ＰＥ台数			使用ＰＥ台数			先行リクエスト実行待ち				リミット値による実行制限					PE割り当て不可［空きPE］					PE割り当て不可［PEリザーブ］					
	jq01	jq02	合計	jq01	jq02	合計	ge1	ge2	ge3	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	jq02	ge1	ge2	ge3	min	max	jq02
00:00	161	4	165	159	2	161	53	56	0	0	0	4	0	14	0	18	4	0	6	0	0	0	0	0	0	0
01:00	161	4	165	160	2	162	43	63	0	0	0	4	0	14	0	13	6	0	6	0	0	0	0	0	0	0
02:00	161	4	165	158	2	160	56	62	0	0	0	4	0	14	0	10	6	0	6	0	0	0	0	0	0	0
03:00	161	4	165	158	2	160	54	51	4	0	0	4	0	14	0	5	5	0	16	0	0	0	0	0	0	0
04:00	161	4	165	155	2	157	50	47	4	0	0	4	0	14	0	3	4	0	24	0	0	0	0	0	0	0
05:00	161	4	165	159	2	161	42	51	0	0	0	4	0	14	0	2	4	0	50	0	0	0	0	0	0	0
06:00	161	4	165	158	2	160	40	51	0	0	0	3	0	18	0	19	4	0	6	0	0	0	0	0	0	0
07:00	161	4	165	159	2	161	40	49	0	0	0	0	0	0	0	13	6	1	6	0	0	0	0	0	0	0
08:00	161	4	165	161	2	163	39	48	0	0	1	1	0	18	0	10	4	0	6	0	0	0	0	0	0	0
09:00	161	4	165	148	2	150	－	－	－	－	－	－	－	－	－	－	－	－	－	－	－	－	－	－	－	－
10:00	161	4	165	161	1	162	41	45	0	0	0	1	0	50	0	21	4	0	1	0	0	0	0	0	0	0
11:00	161	4	165	160	1	161	95	43	4	0	0	2	0	18	0	11	3	0	6	0	0	0	0	0	0	0
12:00	161	4	165	156	1	157	62	77	0	0	0	2	0	18	0	11	3	0	9	0	0	0	0	0	0	0
13:00	161	4	165	161	0	161	71	71	0	0	0	2	0	18	0	24	3	0	1	0	0	0	0	0	0	0
14:00	161	4	165	159	4	163	71	66	0	0	0	2	0	18	0	17	4	0	6	7	0	0	0	0	0	0
15:00	161	4	165	156	4	160	88	63	0	0	0	1	0	55	0	12	8	0	6	4	0	0	0	0	0	0
16:00	161	4	165	160	1	161	91	64	1	0	3	1	0	1	0	9	10	0	16	0	0	0	0	0	0	0
17:00	161	4	165	148	1	149	83	64	0	0	5	3	0	1	0	7	6	0	16	0	0	0	0	0	0	0
18:00	161	4	165	156	2	158	73	71	0	0	4	1	0	1	0	20	10	0	6	0	0	0	0	0	0	0
19:00	160	4	164	160	1	161	81	69	0	0	8	1	0	1	0	21	11	0	6	0	0	0	0	0	0	0
20:00	160	4	164	160	4	164	73	63	0	3	2	2	0	1	0	12	13	0	6	0	0	0	0	0	0	0
21:00	160	4	164	159	4	163	79	58	0	0	4	1	0	1	0	6	11	0	18	4	0	0	0	0	0	0
22:00	160	4	164	154	2	156	83	53	0	0	5	1	0	1	0	5	8	0	20	0	0	0	0	0	0	0
23:00	160	4	164	160	1	161	92	43	0	0	0	0	0	0	0	7	7	0	4	0	0	0	0	0	0	0
	160.8	4.0	164.8	157.7 98.1	2.0 49.0	159.7 96.9	65.2	57.7	0.6	0.1	1.4	2.1	0.0	13.2	0.0	12	6.3	0.0	10.7	0.7	0.0	0.0	0.0	0.0	0.0	0.0

② リミット値による実行制限

リミット値による実行制限は，システムの高効率を図る，特定ユーザの PE 専有を抑止する，昼間の時間帯はジョブのターンアラウンドタイムをできるだけ小さくするという目的から行われているが，制限がきつすぎることは好ましくない。しかし，PE割り当て不可［空き PE］に比べてもリクエスト数は少なく，リクエスト数が変動しているため，制限がきついとは考え難い。図 5.1 は，平成９年の４月から５月の間に処理されたジョブの要求 PE 台数の分布を示したものである。この図からも分かるように，1PEの投入リクエスト数が圧倒的に多いにもかかわらず最小 PE 台数が１の場合は少なく，1PE ジョブが実行制限された場合でもあまり待たされず処理されていると考えられる。

③ PE 割り当て不可［空き PE］

PE 割り当て不可［空き PE］は，システムの処理能力以上にジョブが投入されれば当然多くなる。したがってこの項のリクエスト数が多いことはスケジューラの良否には関連がないが，混雑度に対しパラメータの設定が不適切であると最優先リクエスト数が極端に増え，優先度の設定自体が意味をもたなくなる。しかし，最優先リクエスト数は一時的に多くなることはあっても連続して多いことはなく，現時点ではパラメータは適切であると考えられる。当然のことながら，“ min ”の項に示されている最小PE台数は，空き PE 台数より大きい。

④ PE 割り当て不可［PE リザーブ］

PE リザーブは，最優先リクエストキューの先頭のリクエストが PE の割り当てを受けられず，実行できないとき，すなわち，PE 割り当て不可［空き PE］の“ ge2 ”の項が１以上の時に発生する。15 日のデータでは８回，７日間では35回発生している。

表 5. 8 は，PE リザーブが発生したときの詳細情報を発生順に示したものである。ここでは，PE リザーブリクエストの情報（リクエスト id，要求 PE 台数，要求 CPU 時間，シークエンス番号）と表 5. 1 ～表 5. 7 の PE 割り当て不可［PE リザーブ］の項の情報を示している。ここで，リクエスト id は，PE リザーブリクエストを識別するための番号であり，シークエンス番号はジョブ内の総リクエスト数に対する当該リクエスト順位である。

この表から以下の事柄が明確になった。

① 35回の発生において，同じリクエスト id が連続しているのは，20 日の 10:00 と 11:00 だけであることから，PE リザーブジョブは大半の場合１時間以内で開始されている。

② PE リザーブの発生回数 35 回に対し，PE の割り当て不可が発生したのはわずか７回であり，PE 利用率の低下は見受けられない。

③ PEリザーブによるPE割り当て不可は必ずしも PE 台数が多い場合とは限らない。新スケジューラの場合，PE稼働率が高いため，1PEや4PEでも割り当てできない場合がある。

④ 先行リクエストの実行待ち中に最優先リクエストになる場合，後続のリクエストが他のリクエストより優先されることになる。しかし，シークエンス番号の項の情報では総リクエスト数が２以上の場合は５回と少なく，影響がでているとは考えられない。

(1) JQ02 クラスの場合

JQ02クラスの場合，使用可能なＰＥ台数が４と少ないが，大半が1PEのジョブであるため，ジョブが十分にあれば100%の稼働率達成も難しくはない。しかし，JQ02 クラスでは 1GB のメモリ量を使用できるという利点があるものの，4PE という設置台数のせいか，比較的ジョブが少ないのが現状である。そのため，週末や夜間にジョブがなくなることがあり，それが使用率の低下となっており，それ以外には，運用上の問題は見受けられない。

5. 1. 3 その他

新スケジューラで確認すべき項目の中にスワップアウトの頻発がある。スワップアウトは，リクエストを緊急に処理する場合などに有効な機能であるため，新スケジューラにも組み込み使用できる状態になっている。しかし，PE 利用率の低下要因ともなるため，緊急度の極めて高いジョブに対してのみ使用する方針であり，現在はほとんど使用しておらず，今回のデータ採取期間でも使用

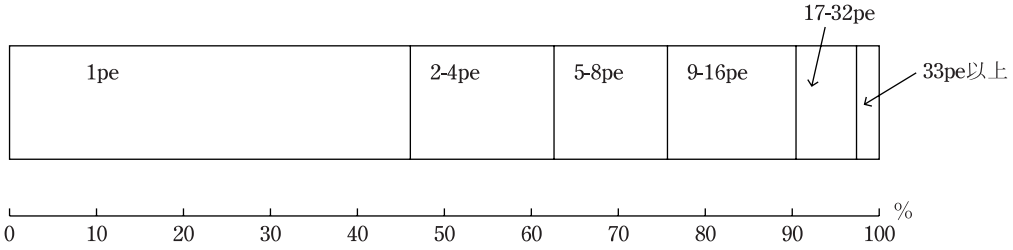

図 5.1 ジョブの要求 PE 台数の分布

表5.8 PEリザーブ詳細情報

no	日時		PEリザーブリクエスト情報				PE割り当て不可情報				
			リクエストid	PE台数	CPU時間秒	シークエンス番号	ge1	ge2	ge3	min	max
1	15日	00:00	22917	49	500	1/1	20	2	0	6	24
2		11:00	23422	1	480	1/1	0	0	0	0	0
3		12:00	23276	25	400	1/1	13	0	0	10	16
4		13:00	1165	1	180	?	0	0	0	0	0
5		16:00	23764	10	580	1/1	0	0	0	0	0
6		17:00	23818	10	600	1/3	0	0	0	0	0
7		18:00	23716	25	400	1/1	0	0	0	0	0
8		22:00	24004	8	500	1/1	0	0	0	0	0
9	16日	03:00	24094	17	900	1/1	0	0	0	0	0
10		10:00	24195	1	10	1/1	0	0	0	0	0
11		14:00	24420	4	150	1/1	0	0	0	0	0
12	17日	01:00	24872	4	150	1/1	0	0	0	0	0
13		12:00	24978	8	500	1/1	0	0	0	0	0
14		19:00	25312	4	150	1/1	0	0	0	0	0
15		21:00	25336	4	150	1/1	0	0	0	0	0
16		23:00	25454	4	150	1/1	0	0	0	0	0
17	18日	02:00	25501	4	150	1/1	0	0	0	0	0
18		10:00	25658	8	500	1/1	0	0	0	0	0
19		11:00	25292	36	2800	1/1	17	3	0	6	32
20		18:00	25723	4	150	1/1	2	0	0	1	1
21		21:00	25473	55	3000	1/8	2	2	0	14	24
22	19日	10:00	26028	14	600	1/1	0	0	0	0	0
23		11:00	25089	14	600	1/2	0	0	0	0	0
24		13:00	26216	16	150	1/1	2	0	0	1	1
25		14:00	26279	16	100	1/1	0	0	0	0	0
26		16:00	26317	25	100	1/1	0	0	0	0	0
27		18:00	26383	8	500	1/1	0	0	0	0	0
28		21:00	26488	8	500	1/1	0	0	0	0	0
29	20日	10:00	22105	64	1000	1/10	0	0	0	0	0
30		11:00	22105	64	1000	1/10	0	0	0	0	0
31		12:00	22693	4	150	1/1	8	0	0	1	50
32		13:00	26232	55	3000	1/1	0	0	0	0	0
33		14:00	1405	1	180	?	0	0	0	0	0
34		17:00	26936	4	150	1/1	0	0	0	0	0
35	21日	04:00	26883	64	1800	1/1	0	0	0	0	0

されていない。

5.2 考察結果

数値風洞の稼働データから以下の事柄が得られた。

① システムのPE利用率は極めて高く，新スケジューラの第一目標であるPEの有効利用は達成されている。しかし，メモリ容量が1GBのクラスの投入ジョブ数が少なく4台のPEが空くことがあるため，PE利用率を更に向上させようとするならば，PEの割り当て方法を検討し，JQ02用PEの有効利用を図る必要がある。

② 優先度別の実行待ちリクエスト数から，キュー遷移パラメータの設定が適切であったことを確認した。しかし，このパラメータはシステムの混雑度とも関係があるため，最優先リクエストキューのリクエスト数が極端に増大しないように常に監視

が必要である。

③ PE リザーブは長時間待ちのリクエストを優先的に処理するために有効な機能であるが，他のリクエストの実行を大幅に阻害する場合には，PE 利用率の大幅低下の要因となる。しかし，現状では，パラメータ設定も適切であり，他リクエストへの影響も少なく，PE 利用率の低下を引き起こしてはいない。

④ 大規模リクエストが起動優先リクエストとなった場合，要求台数の PE が確保されるまでに長時間かかる可能性もあるが，7 日間の稼働状況データからはその様な様子は見受けられなかった。

⑤ 航技研の運用方式を実現するために，1PE リクエストの実行数，同一ユーザの使用 PE 台数，リクエストの最大使用 PE 台数の制限を行っている。これらは，PE 利用率の向上，リクエストの適切なターンアラウンドタイムの保証等に有効な機能であるが，そのために特定のリクエストの実行が滞るような事態は見受けられなかった。

⑥ スワップアウトは発生しなかったが，仮に発生したとしても，適切なパラメータを設定すればスワップアウトの頻発は防止できる。

⑦ 従来のシステムと同様に数値風洞でも CPU 制限値では終了しないプログラムを実行するためにジョブステップの逐次実行機能を組み込んでいる。しかし，この目的で投入される一連のリクエストが実行待ち中に最優先リクエストになると，後続のリクエストは待つことなしに順次処理される可能性が高い。このようなジョブの割合が極端に増加した場合には，同一ユーザの長時間 PE 独占を抑止するための CPU 打ち切り値の効果が期待できなくなる。そのため，ジョブステップの逐次実行機能使用リクエストの優先度の設定方法を工夫し，必要以上に優先処理されないようにする必要がある。

5. 3 スケジューラの問題点と対処方法

前項の考察の結果，スケジューラは所期の目的を達成しており，かつ運用時の致命的な問題点もなかった。しかし，2 つの問題が提起された。ここで，それらの対処方法を提案する。

(1) JQ02 用 PE の有効利用

まず，JQ02 クラスに実行待ちのリクエストがなかった場合の 1GBPE 有効利用である。現在は，1GB の PE のメモリを有効に使用するために，1GB の PE，256MB の PE を使用するジョブクラスを設けて個々に割り当てを行っている。ここで，1GB の PE を要求するリクエストを 1GB リクエスト，256MB の PE を要求するリクエストを 256MB リクエストと呼ぶことにする。

256MB リクエストは 1GB の PE でも実行可能であるため，PE の有効利用を図るためには，256MB リクエストに 256MB の PE と 1GB の PE の割り当てを可能にしなければならない。このような処理は，166PE を要求するジョブを実行する際に現在でも使用している。ただし，256MB の PE から割り当てるというように割り当て順序を指定することができないため，割り当て手順は以下のようになる。

i) 1GB リクエストに 1GB の PE を割り当てる。

ii) 256MB リクエストに 256MB の PE または 1GB の PE を割り当てる。

仮に，1GB の実行待ちリクエストがなく 1GB の PE が空いている場合には，256MB の PE に空きがあっても 1GB の PE を 256MB リクエストに割りつけることがある。その結果，直後に投入された 1GB リクエストがあっても実行できなくなり，1GB リクエストの実行が多少遅れることがあるが，256MB リクエストの処理は進み PE の利用率も向上する。

通常の運用時間帯にこの処理方式を使用することはユーザの混乱を招く恐れがあるが，実行待ちの JQ02 リクエストがなくなることのある休日や夜間に限り，自動あるいは手動で使用することは好ましいと考える。

(2) ジョブステップの逐次実行機能使用リクエストの優先度の設定方法

この問題に対処するためには，先行リクエストがある場合には待ち時間が変化しても優先度を上げないという方法が一番良い。これを実現する方法は，ユーザタイムリクエストと同様に特権化情報ファイルを使用する方法が考えられる。一般リクエストは投入時間，ユーザタイムリクエストはユーザタイムが開始された時間により優先度が計算されるが，先行リクエストがある場合には，先行リクエストが終了した時間から優先度を決定すればよい。

6．おわりに

数値風洞は世界でもトップクラスの処理能力を誇るシステムであるため，運用に際しては PE の有効利用を図ることのできるジョブスケジューラが必須であった。しかし，このような規模の並列計算機システムは世界でも一般運用の例がなく，汎用計算機のアルゴリズムをそのまま適用できないことから，以下の 4 ステップから成る開発過程を経て，数値風洞用のジョブスケジューラの開発・組み込みを行った。第一ステップでは，現実のシステムの動きを模擬するジョブシミュレータを作成し，並列計算機用に考案したスケジューリングアルゴリズムを

組み込んだ。第二ステップでは，ジョブシミュレータに，数値風洞で想定されるジョブミックスを投入したシミュレーション実験を行い，有効性の検証を行った。第三ステップでは，ジョブスケジューラのインタフェース部分の変更と，故障PE台数の読み込み等数値風洞での情報収集機能の組み込み手続きの付加を行った。第四ステップでは，スケジューラを数値風洞へ組み込み，動作確認を行った。第一，第二，第三ステップの処理は主にC言語によるプログラム構築であるが，著者ひとりで行ったこと，初めてC言語を使用したことから約１年かかった。第四ステップは運用関係者や富士通㈱のSEの協力の元で行った実システムを使った動作確認処理であるが，一般ジョブを全て止めて行うため，休日や夜間を使用せざるを得なかったこと，ちょっとしたプログラムミスを直す際も，システムの停止，プログラム修正，システム立ち上げ等の一連の処理が必要となるため，一日かかっても数回のテストしかできなかったことから，テスト運用期間も含め半年程度かかった。開発したプログラムは約6000ステップになる。なお，このジョブスケジューラは，1994年10月から本運用に供している。

新スケジューラの開発以前は，機能の不足を人の手で補っていたために，処理する人によっては適切な対処ができないことがある，対処までに時間を要する，処理が煩雑である，常時注意深い監視が必要である，等の問題があった。しかし，新スケジューラの開発によりこれらの問題も解決し，第一の目標であったPEの高効率利用も可能となった。稼働状況データを使用した有効性検証では，1GBのメモリを持つPEの有効利用やジョブステップの逐次実行機能を使用しているジョブの優先度の設定方法の変更等，改善すべき点も明確になったが，所期の目的をすべて果たすことができ，その上大きな問題も発生していないことが確認できた。

謝　辞

本スケジューラの機能開発においては，現在地球シミュレータ研究開発センターの三好甫センタ長にいろいろなアドバイスを頂いた。また，計算科学研究部計算課の土屋雅子主任研究員には，航技研独自機能を考案する際には適切な助言をいただき，シミュレータの数値風洞への組み込み，運用テストなどの際にはいろいろお骨折りを頂いた。ここに，感謝の意を表する。

参考文献

1) 末松和代；並列計算機システム用ジョブスケジューラ，航技研報告 TR-1277 (1995)
2) 計算研究室；電子計算機使用法 (1961)
3) 中村，石塚，吉田；航技研FACOM230-75アレイプロセッサシステムセンタルーチンの作成，航技研資料TM-334(1978)
4) 計算課；航空宇宙技術研究所計算センタ利用手引，航技研技術資料N-24（1979)
5) 三好；航空宇宙技術研究所におけるアレイプロセッサシステム，FACOMジャーナル，Vol.5，No.3，pp37-43(1979)
6) 末松，中村，吉田，原田，三好；FACOM230-75のアレイプロセッサシステムに対するジョブプロセッシングシミュレータ，航技研資料TM-384(1979)
7) 末松，吉田，中村，三好；FACOM230-APシステムのシステムシミュレーション，航技研報告TR-590(1979)
8) 末松，吉田，土屋，畑山；主記憶有効利用のための諸方策および航技研システムへの適用検討，航技研資料TM419(1980)
9) 土屋，末松，吉田，畑山；計算機システムにおけるジョブ処理用新スケジューラの提案，航技研報告TR-659(1981)
10) 畑山，吉田，末松，土屋，小松；次期航技研計算機システムの運用計画，航技研資料TM-430(1981)
11) 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(1981)
12) 土屋，畑山；航技研計算機システム用トランザクション・プロセッシングシミュレータ，航技研資料TM-543(1985)
13) 数理解析部；NSシステム利用の手引き，航技研技術資料N-41(1988)
14) 土屋；NSシステム用ジョブ・ジョブステップ・スケジューラの開発，航技研報告TR-977(1988)
15) UXP／M　NQS（1990年11月)
16) NQSユーザライブラリ（1992年6月)