■7 群（コンピュータ －ソフトウェア）－3 編（オペレーティングシステム）

1 章 序　論

（執筆者：吉澤康文）［2013 年 2 月 受領］

■概要■

オペレーティングシステム（以降 OS と略す）はコンピュータに不可欠なソフトウェアである．コンピュータのハードウェアの機能を補完し，多様なアプリケーションソフトウェアの開発を容易にする機能をプログラマに提供し，更にコンピュータシステムを運用管理する補助的な役割を果たしている．これらの機能により，コンピュータの基本的な知識のないユーザ（エンドユーザともいう）が容易にコンピュータを使いこなせる原理を理解できるよう説明する．

【本章の構成】

本章では，最初に OS の果たす役割（1-1 節）を説明する．工業製品の共通的な使命は使いやすさ，信頼性そして性能にあるが OS もその例外ではないのでその各々の概要を示す．使いやすさの例として，コンピュータの電源を投入すると自動的に利用可能な状態となり，組み込まれたプログラムが動作可能な状態となるが，その原理を説明する（1-2 節）．次に，プログラム開発者とコンピュータ管理者に提供される機能である機能マシンならびに資源管理機能について説明する（1-3 節）．最後に OS の発展を代表的な OS を例として示し，各々の OS の果たしてきた役割を説明する（1-4 節）．

■7 群 － 3 編 － 1 章

1-1 オペレーティングシステムの役割

（執筆者：吉澤康文）［2013 年 2 月 受領］

オペレーティングシステム（以下 OS と略す場合もある）の役割はコンピュータを使いやすくすることである.「使いやすくする」ということは難しい問題であり永遠の研究テーマであろうが，ここでは，OS が存在することでプログラムの生産性が向上すること，信頼性と性能保証をどのような考えで行っているかについてだけ短く説明する．**図 1･1** はこの三つの OS の役割を概念的に示した.

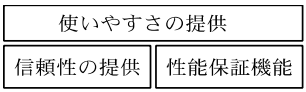

図 1･1　オペレーティングシステムの三つの役割

1-1-1 使いやすさの提供

コンピュータに電源を入れた直後に，最初に起動される一連のソフトウェアが OS である．電源を入れてから OS を主記憶に読み込む動作をブートストラップ（Bootstrap）と呼んでおり，OS の一部の機能である.

ブートストラップが完了すると，利用者（ユーザと呼ぶ）はキーボードから文字列を入れて OS に指示を与える．この文字列をコマンド（Command）という．OS への指示を与えることによりアプリケーションプログラムを実行し，ファイルを作るなどの作業を進めることができる．その結果は OS からの応答としてディスプレイ（Display）上に表示される.

個人利用のパーソナルコンピュータも，企業で使用する大規模なデータ処理用の大形コンピュータも基本的な動作は同じである．OS はユーザが容易にコンピュータを操作できる機能を備えることが第 1 の役割である．上記に示したようなキーボードからの文字列入力や，ディスプレイへの文字出力だけでなく，もっと簡単にコンピュータを利用できるように工夫されたインタフェースがある. 例えば, マウス（Mouse）のようなポインティング装置（Pointing Device）を使って，ディスプレイ上のアイコン（Icon）をクリック（Click）するようなインタフェースをグラフィカルユーザインタフェース（GUI : Graphical User Interface）という．クリック操作は決まった文字列のコマンドを入れる代わりであり，操作を簡単にしている.

ソフトウェア開発を容易にするのも OS の重要な役割りである. 初期の OS 開発の目的は，プログラマの生産性向上にあった．OS のないコンピュータの下でプログラムを開発する状況を想像すると，誰でもまず，コンピュータにプログラムやデータを読み込ませ，実行結果をプリンタなどに出力するという入出力のためのプログラムを作らなければならない．このような，どのプログラマにも必要となる機能はコンピュータと共に提供された方が使いやすい.

そこで OS は，入出力を容易に実行する機能を提供するものとしてハードウェアと共に提供されていた．しかし，コンピュータをより多様に，かつ高度に利用するには，単に入出力機能だけでは不十分であり，後の章で述べるプロセス管理，ファイル管理，メモリ管理などの機能を必要としてきたのである.

1-1-2 信頼性

コンピュータは電気，ガス，水道のように現代生活では常時利用できることが要求されている．このように，社会のインフラストラクチャ（Infrastructure）として組み込まれたコンピュータシステムは，ほかのインフラストラクチャと同様にシステムの全面停止を避けなければならない．

OS はハードウェアとアプリケーションプログラムとの中間に位置するソフトウェアである．このため，両者の障害を監視できる位置にあり，障害を検知したなら，部分的に切り離しを行い，その記録をとり，オペレータなどに知らせなくてはならない．機能の一部を切り離し，システムの全面停止を避けることをフォールトトレランス（Fault Tolerance）と呼ぶが，OS はフォールトトレランスを実現する機能をシステムプログラマに提供する必要がある．このために，アプリケーションプログラムに障害が発生しても，障害の発生を処置するプログラムを実行させ，場合によるとプログラムの実行を中断し閉塞する処置もする．

OS自身もソフトウェアであるので，バグ [*1] が潜在している可能性がある．つまり，OSといえどもバグは付き物でありバグを避けてとおれないのである．潜在していたバグが露呈した場合，OSの一般的な処置としては，バグの原因を追及できるようにメモリダンプ [*2] などを取得してシステムダウンを避け，処理を続行する．つまり，フォールトトレランスな構造としている場合が多い．

上記に述べたように，OS の第 2 の役割は信頼性の確保である．OS は，ハードウェアやソフトウェアの障害が起きてもシステムダウンにならないように故障や障害の診断，再試行，機能回復などのフェールソフト（Fail Soft）を自ら実現し，なおかつ，フェールソフトなシステム設計を可能にする機能の提供が必要である．

1-1-3 性能保証

コンピュータのハードウェアは年々高性能化，大容量化が進んでいる．これらのハードウェアの進歩を直接エンドユーザに受け入れてもらう必要がある．また，使用中のコンピュータに新しいハードウェアを増強した場合には，投資に見合っただけの性能強化がなされねばならない．

OS はハードウェアの進歩や装置の増強の効果をそのままエンドユーザに受け取ってもらうための機能を提供する必要がある．この場合大切なことは，アプリケーションプログラムを全く変更しないで済むようにすることである．

また，大形コンピュータやオンラインバンキングシステムのようなOLTP（On Line Transaction Processing）やWWW（World Wide Web）サーバには，同時に複数のクライアント [*3] が処理を要求してくる．この種のサーバマシンでは，多くの処理要求に対し，限られた資源をスケジューリングし，各々の処理性能を保証する必要がある．この機能は，一般的にエンドユーザには見えないところで働いているOSの資源管理機能であり，OS設計者にとって腕の見せどころでもある．

---

*1 バグ（Bug）：プログラムの誤りを習慣的に呼ぶ．
*2 メモリダンプ（Memory Dump）プログラム実行時のメモリ内容を 2 次記憶に退避し，後でデバッグ（Debug：バグを解決する）に利用する情報．
*3 クライアント（Client）：処理を要求する側．処理を実行する側をサーバと呼ぶ．

コンピュータは高速化ならびに大容量化している．しかし，使いやすさや信頼性の向上に伴う機能強化から，ソフトウェアが肥大化し，従来以上にコンピュータ資源を多用するようになった．このため，ハードウェアの性能向上に比例した効果を期待できないという問題が出てきている．

性能保障はソフトウェア開発過程の一つの重要な問題でもある．新しい機能の開発だけでなく,既存機能への改良などは特に性能を意識した設計が重要になる.このような状況から，性能評価という課題が生まれたのである．OS には性能評価を行う基本的な機能が組み込まれていることが多いが，公開されているか否かは OS に依存する．

OS はコンピュータ内で起こるすべての事象（Event）を把握している．このため，なにがどのような順序で実行されたかを把握し，装置がどのぐらい利用されているかを測定することが可能である．これらの情報を適切に収集しておけば，上記に述べた性能分析の要求に応えることができる．仮に，この種の機能を内蔵し収集情報などが公開されているならば，新規システムや拡張システムなどの性能改善や性能限界を見極めることが可能になり，コンピュータシステムのライフサイクルを予測することも可能になる．このような作業をキャパシティプランニング（Capacity Planning）と呼び,情報システム導入企画には重要な案件である．

以上のように，OS はハードウェアの性能を引き出し，エンドユーザに対しては性能保証を行う義務を有している．不特定多数の要求を処理するサーバなどでは，限られたコンピュータ資源を性能要求に応じて割り振り，性能を監視しなくてはならない．また，コンピュータシステム開発者や運用者に対して性能情報を提供する機能を必要とする.これらの情報は,性能改善やハードウェアの増強などに有効な情報となり，経営資源としてのコンピュータの投資計画を立てることができるのである．

■7群 － 3編 － 1章

1-2 プログラムはどのようにして動くのか

（執筆者：吉澤康文）［2013年2月 受領］

1-2-1 コンピュータ利用の手順

初期のコンピュータは科学技術計算を目的に開発されたが，今日では主にデータ処理に利用されている．つまり，コンピュータを利用することにより情報の収集を行い，既に蓄積された情報へのアクセスをすることになる．コンピュータの利用に際しては，個人で利用するときも1台のサーバを不特定多数の人間同志で共有利用するときも情報へのアクセス権の確認を常に行わなければならない．現在，個人利用のコンピュータでこの種の機能が使用されていなくても，徐々にセキュリティ（Security）の問題は必須機能となるであろう．

そこで，コンピュータを利用するときは，まず，コンピュータに利用者が正当に利用できる権利を有しているかを認識させる必要がある．コンピュータによっては，利用に際して課金制度を導入している場合もある．したがって，利用者の名前，グループ名，課金番号，場合によるとパスワードを入力する必要がある．1970年代に確立したタイムシェアリングシステム（TSS：Time Sharing System）では，端末からこれらの情報を対話的に入力するが，これをログイン手続き（Login Procedure）と呼ぶ．

1-2-2 プログラムの実行手順

ログイン手続きにより利用者がOSに認識されると，コンピュータを利用できる状態になる．現代の多くのコンピュータは対話形式でコンピュータを利用するのが一般的である．あたかもコンピュータと会話する形態であるが，このとき，コンピュータからの出力の多くはディスプレイであり，入力はキーボードやマウスなどを用いる．

ディスプレイ画面による対話形式では，コンピュータへの指示を促す（プロンプト：Prompt）メッセージが出力されるので，コマンド（Command）入力によりプログラムの起動を指示することで仕事を進める．処理目的に応じて各種のコマンドが用意されている．例えばメールソフト（Mailer），ワードプロセッサ（テキスト編集），WWWブラウザなどのプログラムを実行させ，通信，書類やプログラムの作成，情報収集や発信などを行うことができる．

きまりきった作業でなく非定型な仕事をこなすには，対話形式でコンピュータを利用するのが便利である．はじめは非定型であっても定型的な仕事として確立してしまうと，決まったコマンドを毎回入力するのは面倒になる．この場合，実行手順が定まった一連のコマンド列をあらかじめファイルに書き込み，いっぺんに実行させる方法がある．このようなコマンド列を記述したファイルをコマンドプロセジャ（Command Procedure）とか，シェルスクリプト（Shell Script）と呼ぶことがある．

上記はUNIXやDOSなどのコンソール画面での実行手順であるが，GUIによるアイコン操作では，プログラムの起動がマウスの操作で容易になっている．プログラムの起動はプログラムを示すアイコンをクリックするだけで済むのである．また，書類の編集なども書類をクリックするだけでプログラム起動が自動的に行われるようになっている．

このようなユーザインタフェースはウインドウ管理プログラムが指定されたアイコンを判定し，コンソールへのコマンド入力と同じ操作を行うことで実現している．

1-2-3 セッション管理

OS はユーザを認識し，指定されたコマンドを実行するために，ファイルへのアクセス権のチェックを行ったりして処理を進める．また，ユーザが一連の処理過程で引き起こす操作の誤りやプログラム実行中に起こす論理的な不良に対して，システムをダウンさせることなく継続して運転できる能力も提供する．

OS はディスプレイにプロンプトを出力し，ユーザからの入力を読み取り，コマンドを解釈実行する．指定されたコマンド実行のプログラムは，OS により主記憶に読み込まれ，制御が渡る．プログラムの読込みをプログラムローディング（Program Loading）と呼ぶ．

ユーザは一連の仕事を終了するとログアウト（Logout）コマンドを入力し，仕事の終了を OS に指示し，コンピュータから離れることができる．このようなログインからログアウトまでの一連の仕事をジョブ（Job）と呼んだり，セッション（Session）と呼んだりする．一つのジョブが完了すると，OS は課金の処理をしたり，セッションの記録をとったりすることがある．複数のユーザがコンピュータを共有して利用するケースでは特にこれらの機能が必要となる．

■7群 － 3編 － 1章

1-3 機能マシンとしての資源管理機能

（執筆者：吉澤康文）［2013年2月 受領］

図 1･2 に示すように，OS は二つの大きな機能をもっている．その一つは機能マシンである．この部分はプログラマとのインタフェースをもつ部分である．もう一つの機能は資源管理機能であり，企業などで使用するサーバコンピュータの運用者にとっては重要な要素となる．以下，この二つの概要を説明する．

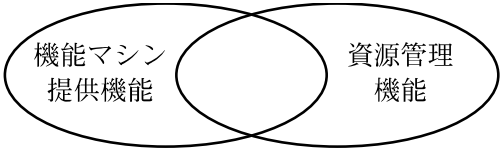

図 1・2　オペレーティングシステムの二つの主機能

1-3-1 機能マシンの考え方

(1)　入出力処理の汎用化

OS のない状態で（つまり裸のコンピュータとして）コンピュータを使用する状況を想像すると，プログラマは大変な労力が必要である．コンピュータはプログラムやデータを主記憶に読み出し，命令を実行する機能しかない単純な機械である．これらの基本的な機能を利用するにはアセンブラ言語をマスタしたプログラマにとってはそれほど苦痛ではないかも知れない．そのアセンブラユーザにとっても大変な労力を必要とするのは，キーボードやディスプレイそして磁気ディスクへの入出力の実行であり，これらは本来ユーザの抱える問題解決とは直接関係ないプログラミング作業である．

このような理由で，ユーザ間に共通の入出力プログラムとして OS は開発され，プログラマの生産性が高くなった．入出力装置の物理的な特性をなるべくプログラマが知らなくても済むように，プログラム間の約束（インタフェースという）を導入し，プログラマはハードウェアの仕様を調査してからでないとプログラムが作れないという状況から開放され，また，今まで開発したプログラムは新しいハードウェアに何の変更もなく適用できることになる．ハードウェアの変更点はすべて OS が吸収してくれるのである．

(2)　ファイルの論理化

入出力と密接な関係があるファイル格納機能は，多くのプログラマが汎用的に利用できるように考えられた．ファイルシステムをハードウェアに非依存なインタフェースにしておけば，プログラマはファイルを論理的な対象物として操作可能となる．言い換えると，OS は，プログラマに対して「ハードウェアを物理的な対象ではなく，論理的な対象物にする」努力をしてきたのである．ファイルシステムがその第 1 の対象物であった．

論理的な対象物として出現したファイルシステムによって，プログラマは安心して自分の問題解決のプログラム開発に専念することができるようになる．少し具体的に述べると，プログラマがファイルを読み込むには，まず，プログラムの中でファイルを開くために OPEN

なる命令を記述する．そして，次に，READ なる命令によってファイル内のレコードを読み込むことができる．この場合，約束として OPEN，READ のパラメータに読み込むレコードの領域，レコードの長さなどを指定する必要がある．そして，読込みがすべて完了したならば CLOSE なる命令でファイルを閉じる．このようにして，プログラマは論理的にファイルを扱うことができるようになった．

(3) 機能マシンとしての OS

このように，ファイルを読み込む場合には，OPEN，READ，CLOSEなどの命令を使用することで操作が容易になる．つまり，OSはマクロな機能 [*4] をプログラマに提供し，裸のコンピュータが提供しているミクロな操作はすべてOSが吸収している．この意味で「OSは機能マシンをプログラマに提供する」といえる．

機能マシンは OS がユーザに提供するインタフェースであり，そのすべてを公開することになる．ソフトウェアを開発する人は，この意味で機能マシンの内容を理解する必要がある．特に，特別なハードウェアに関連したプログラム開発や，オンラインシステム，データベースやコンパイラのようなシステムプログラムの開発者はこれらの機能マシンインタフェースをよく理解する必要がある．

1-3-2 資源管理機能の概要

「人」「物」「金」がなければ製品開発は不可能であるのと同じようにコンピュータに仕事をさせるには，以下の 3 資源が必要である．

(a) 演算装置（CPU：Central Processing Unit ということもある）
(b) 主記憶装置
(c) ファイル（外部記憶装置）

演算装置はプログラムを実行する装置であり，主記憶装置はプログラムやデータが格納される部分である．そして，プログラムやデータは一般的にファイルシステムに格納されている．このため，三つのコンピュータ資源は一連の仕事の単位であるジョブを実行するために不可欠である．

OS はこれらのコンピュータ資源を管理する使命がある．「管理」という言葉が OS ではよく出てくるが，この意味は，「資源の利用状況を常に把握し，場合によっては，将来の使用を予測し，複数のジョブからの資源要求に対して割り付ける順序を決める（スケジュール：Schedule）ことによってコンピュータの生産性を最大にしようとする）ことである．

上記の 3 資源のうち，演算装置の管理はプロセス管理（Process Management）あるいはタスク管理（Task Management）と呼ばれる．プロセスやタスクについては章を改めて説明するが，同一の意味 [*5] で使うかも知れない．主記憶の管理はメモリ管理の一部である．メモリ管理では，仮想記憶方式が現在一般的であり，その中で説明することになる．ファイルシステムはOSごとに異なる設計思想で作られている．本編では，共通する基本概念と実用化されているOSの例をファイル管理の章で説明する．これら三つの管理機能はそれぞれが独立してい

[*4] マクロな機能:ひとかたまりのマシン命令群からなる機能をあたかも命令のような形式で利用させること．
[*5] OS の用語：用語は OS によって使い方が異なるので注意されたい．

るのではなく，有機的に結びあって，3 資源を有効に利用し性能信頼性向上を目指している．

OS の資源管理機能は多くの場合，プログラマとのインタフェースが少ない．しかし，コンピュータの運用・管理者や，コンピュータの導入を計画する人にとっては重要な意味がある．このため，プログラマに対して「機能マシン」が「陽の機能」ならば「資源管理」は「陰の機能」である．「陰の機能」において OS 設計者の技量が評価されることが多い．

■7 群 － 3 編 － 1 章

1-4 オペレーティングシステムの発展

（執筆者：吉澤康文）［2013 年 2 月 受領］

技術の発展の歴史を紐解くと，技術の基本的な部分が見えてくることがあり，理解を速めることになる．ここでは，OS の発展の歴史と背景などを短く眺めることにする．定説があるわけではないが，ここでは，OS の発展の経緯を**図 1･3** に示すように八つの世代に分けてみた．

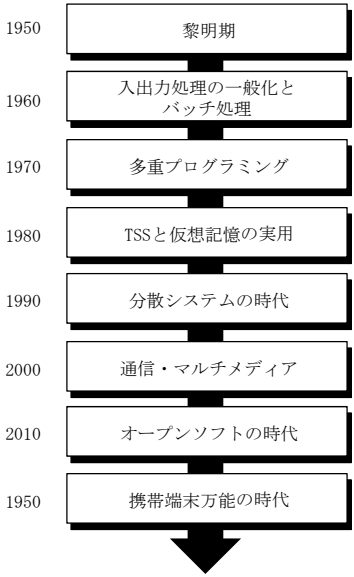

図 1･3　OS の歴史的発展と背景

1-4-1 黎明期

コンピュータが開発された初期の頃はソフトウェアというはっきりとしたものがなく，機械語を効率良く書くためのアセンブラ（Assembler）がその最初のもののように思える．フォンノイマン（Von Neuman）によるプログラム格納型コンピュータ（Stored Program Computer）が提案（1945 年：EDVAC）されてからでも，プログラムはバイナリーコードで入力されていたようである．

自分の問題解決のためにプログラムを書くのはそれほど苦にならないが，それ以外の部分に多くの時間をとられるのは苦痛である．その代表的なプログラムは先にも述べた入出力のプログラムである．データの読込みは紙テープやカード読取り機，計算結果の出力はラインプリンタやコンソールタイプライタを使っていたが，それらの入出力装置の制御コマンドや動作誤りに対する処置をきちんとプログラムしておかなければならなかった．これらのプログラムはユーザ本来の業務ではなく，必要悪として行っていたのである．そして，これらの

部分の作業に要する時間の方が長いこともあった．

これら入出力のプログラムは，多くの人が共通に利用するため比較的早い時期に開発され，後にファイル管理システムに発展することになる．

1-4-2　OS 第 1 世代：入出力処理の一般化とバッチ処理

OS はその名のとおり操作（Operation）を自動化する機能である．コンピュータ初期のころの操作は，現在のようにキーボードとディスプレイを用いるようなものではなかった．その当時は，アセンブラを使う場合には，紙テープにアセンブラのバイナリーコードを入れておき，これをまず紙テープ読取り機から読み込み，主記憶に格納する．その後に，カードに穿孔（穴明け）されたプログラムやデータをカード読取り機から読み込み，アセンブラを動かし，出来上がったオブジェクトプログラムを紙テープに出力し，再度これを主記憶に読み込ませ，読み込ませたプログラムの番地から計算を開始する操作を行う，といった手順を踏む必要があった．

このような面倒な操作をしていたのではコンピュータの生産性が向上しないのは明らかである．コンピュータの性能がいくら上がっても，人手の部分に生産性のネックができてしまう．この問題を解決するために，まず考えられたのが古典的なバッチ処理（Batch Processing）である．バッチ処理では，カードに穿孔されたジョブ（カードデックと呼んでいた）を連続して読み込み，ジョブを実行する．このための OS は，カードリーダからジョブを読み込み，Fortran やアセンブラなどの言語処理系を主記憶に読込むローダ（Loader）を備え，オブジェクトプログラムに自動的に制御を渡し，そしてプリンタに出力がなされてジョブが完了したら再び次のジョブを読み込むことを繰り返せる機能をもったものであった．

1-4-3　OS 第 2 世代：多重プログラミング

1960 年代の後半になると，コンピュータの性能が向上し，主記憶や入出力装置の能力が高くなってきた．コンピュータの第 2 世代である．上記の，連続バッチ処理ではジョブが一つずつ逐次的に実行されるために，入出力装置を多用するジョブでは CPU が遊んでしまいコンピュータの生産性は上がらない．

このような背景から，入出力と CPU の実行を独立に並行して行わせるハードウェアが出現したのである．CPU の演算と入出力処理が同時に実行できるようにするには，CPU が入出力の指令を行い，データチャネル（Data Channel）がその指令を受けて入出力装置を動かすという仕組みになる．**図 1･4** に CPU とデータチャネルの並列処理の動きを示した．

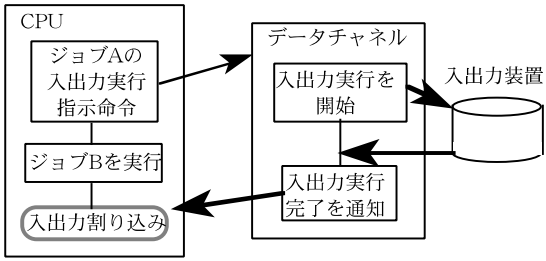

図 1･4　CPU とデータチャネルの並列実行の原理

ジョブ A の要求を受けて入出力装置が動作している間，CPU はほかのジョブ（B）を実行することができるため，ジョブ A と B を同時に実行できることになる．入出力装置はチャネルに接続され，その制御を受ける．チャネルは立派なコンピュータであり，命令体系（これをコマンドと呼ぶことが多い）をもち動作するのである．CPU とチャネルは独立に動くので，相互に交信を行う必要がある．この交信手段がチャネルの入出力完了割込みである．

このような割込み機能を使って CPU とデータチャネルが同期をとり，複数のジョブを同時に処理する方法を多重プログラミング（Multiprogramming）と呼ぶ．この技術は飛躍的にコンピュータの生産性を向上させ得ることを示唆したのである．そこで，多重プログラミングでの関心事は，いかにして多重プログラミングの多重度（Degree of Multi-programming）を向上させるかということであり，メモリ管理技術に期待が寄せられたのである．つまり，高価で限られた主記憶の中に，いかに多くのジョブを詰め込むかという問題に直面し，これがメモリ管理技術の発展の源になったのである．

1-4-4　OS 第 3 世代：TSS と仮想記憶の実用

多重プログラミングを実現する OS が出現し，コンピュータの処理能力が向上してくると，プログラマがジョブを依頼してその実行結果を得るまでの時間であるターンアラウンドタイム（Turn Around Time）の短縮に対する要求が強くなった．つまり，ターンアラウンドタイムに占める実効的なジョブ実行時間の割合が小さく待ち時間の割合が大きくなったのである．そこで，ユーザはカードに穿孔したカードを計算センタまで運んで依頼するのではなく，自室から直接プログラムをコンピュータに入力し，計算結果を直ちに得たい，という要求をもつようになった．

上記の要求を満たすには，1 台のコンピュータに複数のユーザが同時にアクセスできる通信機能と端末の開発が必要となる．そのため OS に通信ならびに端末制御機能が加わるのである．更に OS のプロセス管理には画期的な進展が起こる．

従来のバッチ処理での性能基準は単位時間当たりのジョブ処理件数，つまり，スループット（Throughput）が重要であった．しかし，プログラマが直接コンピュータにアクセスする環境では応答時間（Response Time）の短さが重要になり，新しい性能尺度が生まれた．この当時コンピュータは高価であり，対話型利用とバッチ処理が同一システムに共存する形態とならざるを得なかった．そこではバッチ処理にはスループットを保障し，対話型処理には応答時間を保障するという課題が生まれた．そこで，プロセスに CPU 時間を割当てる際に，時間をタイムスライス（Time Slice）という小さな単位に区切り，1 回の CPU 割当時間の上限とするスケジュール方式が考案され効果を上げた．

端末からプログラムを作成するには，テキスト編集が不可欠である．多くの場合，テキスト編集では人間のタイピング時間に比べてコンピュータの処理時間は極端に短い．したがって，上記のタイムスライス以内で処理が完了してしまう．このため，複数の端末で同時にテキスト編集作業を行っても，各々の処理が快適に実行されることになるので，各ユーザはあたかも自分一人がコンピュータを占有して使っているような気分になれるのであるタイムシェアリングシステム（TSS：Time Sharing System）はこの原理を利用した対話型処理機能である．TSS が普及すると，便利な機能は極限まで性能が追及されるようになる．

1970 年代に入るとより大きな主記憶に対するニーズが高まってきた．また，技術的にも，

仮想記憶が研究・実験段階から実用化の段階に入ったのである．つまり，コンピュータシステム全体の記憶領域の拡大が可能となったため，多重プログラミングの多重度を飛躍的に高めることが可能になった．例えば，24 ビットアドレッシングであった IBM 社の System/360 では実際の主記憶が 2 MB（メガバイト）しかない状況でも 16 MB までの仮想記憶領域が理論的には可能になったのである．

また，多くの端末を同時に利用可能とする TSS 環境に対する要求が生まれた．これは，多重プログラミングの多重度向上要求であり，上記の仮想記憶により理論的には可能になったのである．しかし，物理的に少ない主記憶に対する過剰なメモリ要求はスラッシイングという現象（5 章 5-4-5 項参照）を引き起こし性能低下となって現れた．この新しい課題が仮想記憶における多大な技術的な進歩をもたらしたのである．ページングやスワッピング（Swapping）に関する各種の研究は 1970 年代に数多く行われ成果をあげてきた．

1-4-5　OS 第 4 世代：分散システムの時代

上記の多重プログラミングの実用化や TSS の発展の主役は，ハードウェアとソフトウェアのメモリ管理技術であった．しかし，1980 年代の後半になると半導体技術の進歩によりメモリが安価になり，コンピュータはダウンサイジングの時代に入る．つまり，IBM を中心にするメインフレーム（Main Frame）コンピュータ万能の時代ではなくなり，適材適所のコンピュータ利用が進むのである．

ダウンサイジングの原動力は RISC（Reduced Instruction Set Computer）の発想による高性能 CMOS プロセッサの開発にある．また，UNIX がその OS として登場し，ベンチャ企業が次々とワークステーション（Work Station）の市場を形成しはじめるのである．

UNIX はアメリカの大学で飛躍的な発展を遂げる．特に，カリフォルニア大学バークレイ校（University of California, Berkeley）での開発は BSD（Berkeley Software Distribution）として現在でも広く利用されている．ここでの成果はたくさんあるが，なかでも通信とコンピュータを結び付けた分散処理技術は画期的である．分散処理の基本となるパケット通信は，1970 年代の ARPA（Advanced Research Projects Agency）ネットワーク（ARPANET）の成果であり，この発明も偉大な産物である．この技術が，現在の TCP/IP となりインターネットとして発展したのである．また，当時は高速であった，10 Mbps（bit per second）の性能をもつイーサネット（Ethernet）による LAN（Local Area Network）が 1970 年台初期に Xerox 社 PARC 研究所などにより開発されている．このような背景のもとに，ノベル社の Netware のようなネットワーク OS や各種の分散 OS が企業や大学によって開発され，商用化，実用化される段階に入った．

1-4-6　OS 第 5 世代：通信・マルチメディアの時代

RISC プロセッサの進歩によりコンピュータは急速な普及を遂げる．1990 年代に入ると，マイクロプロセッサの急速な進歩により，コンピュータの高性能化と低価格化が更に進み，コンピュータがオフィス業務で利用されはじめる．そして，パソコン（Personal Computer）によるオフィスワーカの情報管理が自動化され，OA（Office Automation）として定着する．

当初はスタンドアローン（Stand Alone）で利用されていたパソコンも，上記の分散処理技術などで蓄積された技術を基に電子メールなどにより，個人の情報管理や人間どうしのコミ

ュニケーションの道具として利用され始めた.イーサネット(Ethernet)による LAN(Local Area Network）によりパソコン同士が結合されてコンピュータネットワークを形成し，更に，コンピュータネットワーク同士が WAN（Wide Area Network）により結合されようになった．これが，インターネット（Internet）であり，世界中のコンピュータが情報交換できるようになったのである．

インターネットは電子メール利用により社会活動を大きく変化させたが，更に，WWW（World Wide Web）はより多くの情報を比較的容易に発進したり，収集したりすることを可能にしたのである．WWW では，音声，静止画，動画などを取り扱うことができ，マルチメディア通信を可能にしたのである．今後は，インターネットの帯域幅が大きくなると予想されるので，マルチメディアと通信の結合による新しい応用の世界が開け，またもや人間の行動が技術の進歩により変化していくものと予想される．

1-4-7 オープンソフトの時代

1990 年代の分散システムを推進した原動力は, RISC プロセッサの出現とその OS として脚光を浴びた UNIX ならびにその派生 OS である．その後，UNIX 系 OS は多くのベンチャビジネスが立ち上がり相互に勢力図を書き換えながら発展を続けてきたが, 90 年代の後半からソースコードを開示した無償ソフトウェアがいくつか出現しはじめた．これをきっかけに，ネットワーク上で相互にソフトウェアを開発し，相互に技術を競うコミュニティが形成され，OS を中心としてコンパイラ，データベースソフト，その他のアプリケーションなどが次々とネットワーク上で配布されるようになった．

この動きは企業の基幹業務にオープン（開示された）ソフトウェアを適用するというビジネス世界に大きな影響を与えはじめた．ハードウェアの価格低下に加え，基幹ソフトウェアの低価格化が進んだのである．

1-4-8 携帯端末万能の時代

2000 年代に入るとネットワーク上で商取引をする電子取引が実用的になっていた．携帯電話は人間のコミュニケーションの道具として既に実用化されていたが, 通信の自由化に伴い, 多くのキャリアが業界に参入し，機能の追加が行われ，それまではパソコンで使用されていたブラウザが携帯端末でも利用できるようになっていた．

このような発展から，パソコンとほぼ同等の機能を備えた携帯端末がスマートフォンというネーミングで世界的に急速に発展・普及した．つまりコンピュータを個人が携帯し，しかも大容量・高速な通信を可能とする技術発展がなされたのである．ここでは，オープンソフトで発展した UNIX 系 OS やパソコンの主力ベンダによるスマートフォン向け OS などが開発され，端末の能力を最大限に発揮するいくつかの主力 OS が出現した．このように発展した端末からは，情報のやり取りをどこにいても可能できるため，その利便性が万能端末といえる．

1-4-9 代表的なオペレーティングシステムの発展

(1) OS/360 とその発展

現代の OS の礎を築いたのは OS/360 である．上記の分類に従えば第 2 世代に属す．OS/360

は当初，IBM 社の System/360 シリーズ用に開発された OS であり，以下の機能を最初から備えていた．

(a) 多重プログラミングによる複数のジョブの同時実行機能

(b) SAM（Sequential Access Method），DAM（Direct Access Method）などのファイルアクセス法（Access Method）を提供し，統一したファイル管理を実現

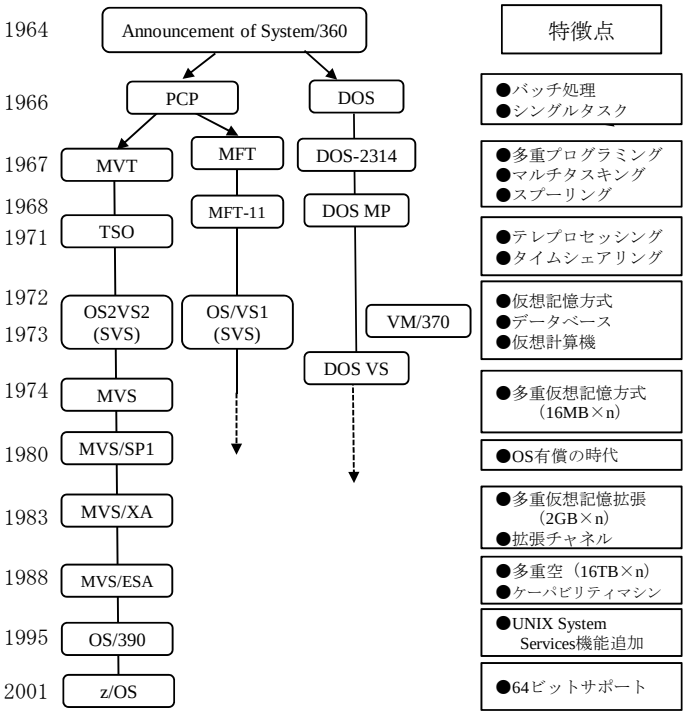

図 1・5　IBM 社 OS/360 の発展と OS の系譜

図 1･5 にその発展した OS の製品系列を示す．System/360 は半導体の進歩などからアーキテクチャとしての発展を遂げ，小型から大型までの製品ファミリーマシンを形成した初めての計算機であった．企業の成長に合わせた製品導入を可能とし，ソフトウェア財産の継承が行える移行性を備えていた．1970 年には System/370 と名称を改め，仮想記憶方式実現すると共に性能向上を果たした．OS/VS1，OS/VS2-Rel.1 はその OS である．これらの OS には以下の機能追加なされている．

(c) 仮想記憶による実記憶以上のメモリ容量をユーザに提供

(d) 大規模な OLTP（On Line Transaction Processing）を実現

(e) VTAM（Virtual Telecommunication Access Method）

1974 年に発表された OS/VS2 Rel.2.1 は OS/360 とは互換性を保持しているが，全く異なる設計思想に基づいていた．この OS は MVS（Multiple Virtual Storage）と名付けられている．

そこでは，以下の機能が更に追加され，先の分類では第 3 世代の OS になったのである．

(f) TSS の標準装備

(g) 各ジョブに独立な 16 MB の仮想記憶空間（アーキテクチャの限界）を提供

(h) TCMP（Tightly Coupled Multi-Processor）ならびに LCMP（Loosely Coupled Multi-processor）の 2 種類の異なる多重プロセッサ（Multi-processing）をサポートした．

MVS はなおも発展を続け，MVS/XA，MVS/ESA では以下の機能拡張を果たしている．

(i) 31 ビットアドレス空間のサポートによりユーザ空間を各々2 GB（Giga Byte）に拡張

(j) 拡張サブチャネルにより入出力スループットの大幅な向上を図る．

など，MVS は分散処理機能を備え，次世代の OS に発展している．

1990 年代は半導体技術の進歩により RISC（Reduced Instruction Set Computer）の出現により 1 チップのマイクロプロセッサが実用化された．この主たる OS が UNIX である．この両者の組合せはコンピュータ業界に大きな変革をもたらした．つまり，ダウンサイジングの波である．この流れを吸収するために，OS/390 が開発され，UNIX System Services 機能の追加がなされたのが OS/390 である．更に，64 ビット拡張のマシンをサポートする z/OS が開発される．z/OS は OS/360 時代から開発されたソフトウェアをすべて動作可能とし，エンタープライズマシンとしての役割を約半世紀の長きにわたり果たしてきた．

(2) UNIX の発展

もう一つの代表的な OS は UNIX である．1969 年に AT&T（American Telephone and Telegraph）社の Bell Laboratory において，Ken Thompson らによって開発された．その設計目標は以下のとおりである．

(a) TSS を前提にした対話型システムとして設計されている．

(b) 木構造のファイルシステムを提供．

(c) 柔軟なコマンドインタープリタ（シェルと呼ぶこともある）を備えている．

(d) プログラマ向きの機能豊富なテキスト編集．

(e) プログラム言語 C によるシステム記述とソースコードの公開．

UNIX は上記の特徴をもった OS として当初は商用ベースでなく提供されていた．ベル研究所では 1978 年の Version 7 まで開発された．また，ソースコードが公開されていたために大学・研究者などにより様々な拡張が行われることになった．その代表例は BSD である．1977 年に BSD はカリフォルニア大学バークレイ校にて Ken Thompson の指導のもとに PDP 11 にて開発されたのが始まりである．その後，Bill Joy などにより発展することになる．BSD は大学育ちの UNIX である．

UNIX はプログラマ向けの OS として歓迎された．その主たる要因はコマンドインタープリタがカーネルとユーザとの中間的な位置にあり，柔軟なコンピュータの利用法を提供しているからである．例えば，任意の文字列を * で表すようなワイルドカード（Wildcard）が利用できるので，ファイル名などの指定を短縮して操作が容易に行えること，端末への出力の代わりにリダイレクト（Redirect）と称して，ファイルに端末出力情報を蓄えることができる機能，そして，先に述べたコマンドプロセジャのようなユーザ専用のコマンドの定義をし，使用環境を定義できるなどの機能が備わっている．また，シェルはカーネルでないので，ユーザが自分の気に入ったシェルをプログラミングすることができる柔軟性も備えている．

BSD では仮想記憶，新しいシェルとして C-Shell，そして分散処理を大きく発展させるソケット（Socket）などが大幅な拡張であり代表的機能となった．BSD は仕様が公開され多くの人々が利用可能であり，また開発に参加するようになったため，各種のコンパイラの開発，MIT（Massachusetts Institute of Technology）による X-Window の開発などにより，プログラミング環境の進歩と応用範囲も格段に広がったのである．

1980 年代の後半になると UNIX は RISC によるワークステーションの OS として発展し，上記の第 4 世代の時代を形成する．なかでも分散処理は UNIX を核として発展しインターネットの草分けとなった．特に電子メールの普及に対する貢献は大きい．

UNIX 系 OS は高性能 RISC プロセッサと共に発展したもので，プログラム開発や CAD（Computer Aided Design）などのためにワークステーションに広く使用された．このように分散環境得で広く利用されるようになると，複数のワークステーションを統括管理し，プリンタや大容量ファイルのような資源を提供する各種のサーバが必要になる．また，電子メールのサービス，次に述べる WWW のサーバという新しいコンピュータビジネス分野が拓けたのである．

WWW（World Wide Web）のプロジェクトは 1989 年にスイスの欧州素粒子物理研究所（CERN）の Tim Berners-Lee らにより開始されたもので，本来は研究者グループの情報共有を目的としていた．WWW はハイパーテキストを使ったインターネット上の広域情報システムである．1994 年に米国イリノイ大学の NCSA（National Center for Supercomputing Applications）から提供された Mosaic は WWW の情報を容易に閲覧する機能（ブラウザ：Browser）であった．WWW は世界中のホストマシン上に公開された様々な情報の情報を収集することが可能であり，我々の情報活動，ビジネス活動，企業活動などを変化させている．

以上述べてきたように，UNIX はメインフレームとは異なった新しい価値をコンピュータにもたらし，メインフレーム万能の時代に終止符を打ったという意味で意義深い OS である．また，OS の機能を公開してオープンな環境を提供しており OS に依存しないアプリケーションの開発を可能とし，シンプルなインタフェースと共に，プログラム言語 C との組合せによりコンピュータの教育にも効果的であった．現在，UNIX 系あるいはそのクローン（Clone）がパソコンにほぼ無償でインストール可能であり，大学・研究所，ならびに個人により広く利用されている．BSD/OS などは安定したソフトウェアであり，ネットワーク関係のサーバとして広く利用されている状況にある．

(3) パソコン用 OS の発展

パーソナルコンピュータ（パソコン）が本格的に世にでたのは，1977 年に Steve Jobs ら設立した Apple 社の Apple II といわれている．BASIC が使えて 15 色カラーを家庭のテレビに映し出せ，たくさんのコンピュータゲームが動き，またビジネス用には表計算ソフト VisiCalc が使えた．その後，1981 年に IBM 社は Intel のマイクロプロセッサ 8088 にパソコンを開発し，世間を驚かせた．当時の OS はディジタルリサーチ社の CP/M-86 である．その後，マイクロソフト社の MS-DOS が普及することになる．IBM/PC には Lotus 1-2-3 という表計算ソフトが開発され爆発的に普及する．Apple II，IBM/PC で始まる第 1 世代のパソコンがなしえたビジネス革命は表計算ソフト（Spread Sheet）といわれている．

1984 年には Apple 社は Macintosh を発売開始する．このパソコンはモトローラ社のマイク

ロプロセッサ 68000 を使い，グラフィカルユーザインタフェースによる使いやすさを追求していた．この GUI を世界で初めて開発したのは，先に述べた Xerox 社の PARC であり Star という世界初のワークステーションである（1981 年）．そこでは，ビットマップディスプレイ，マウスが開発され，アイコン表示，プルダウンメニューなども GUI として開発された．またウインドウがあり，アプリケーションはその中を表示領域とするなど現在のパソコン使用環境と全く同じである．ディスプレイ画面をデスクトップ（机の上の仕事場）とみなす考えが生まれたのもこのときである．Apple の Macintosh が第 2 世代のパソコンといわれ，レーザプリンタと組み合わせた DTP（Desk Top Printing）が急速にパソコンの新しいビジネス分野での利用となったのである．

IBM は PC の仕様を公開したので多くのクローン（互換機）が出現することになる．このため，非常に多くの IBM/PC 互換のパソコンが出現し世界的に普及し，低価格化も進んだ．マイクロソフト社は GUI をもつ Windows 1.0 を 1995 年に出荷したが，コンピュータのメモリ容量と性能の問題もあり，タイリングウインドウという一つのウインドウしか利用できないものであった．その後，Intel 80486 が開発され，性能向上ならびにメモリ容量の増大から複数のウインドウが本格的に利用できるようになり，1992 年に Windows 3.1 出荷される．ここでは，Macintosh と同様のオーバラップウインドウが本格的に利用可能となった．GUI はその後のパソコン用 OS として一般的になり，IBM とマイクロソフトの OS 2，マイクロソフトの Windows 95, 98, NT, 2000 などに引き継がれている．

パソコン用OSはUNIXとは異なり各種のサーバ構築とかプログラム開発用というよりもエンドユーザコンピューティング [*6] を目指している．そのため，OSの使命である性能，信頼性向上よりも，利便性に対する要求が強い．したがって，コンピュータの内容を知らなくても簡単に使用できるインタフェースが必要とされる．

現在，ビジネス，教育，研究など様々な分野にパソコンとインターネットの利用がなされ，社会のインフラストラクチャとなりつつある．パソコンは高性能化，低価格化しており職場や学校だけでなく家庭にも普及が著しい．また，小型・軽量化が進み，外出時にも携帯できるようになってきた．携帯電話の普及により無線通信も容易に利用できる環境になり，パソコン通信を利用したモーバイルコンピューティングを仕事に生かしていく時代になっている．このように，コンピュータは人間どうしの究極のコミュニケーションツールになろうとしている．

[*6] エンドユーザコンピューティング：コンピュータの専門家でない一般ユーザ向けのコンピュータ応用領域をさす．例えば，ワープロや表計算ソフト，お絵描きプログラム，メールソフトなどである．

■7 群 － 3 編 － 1 章

1-5 演習問題

（執筆者：吉澤康文）［2013 年 2 月 受領］

(1) オペレーティングシステムの役割は何か.
(2) オペレーティングシステムの二つの機能とは何か.
(3) 多重プログラミングの利点を二つ以上あげて説明せよ.
(4) タイムシェアリングシステムの原理を説明せよ.
(5) パソコンとサーバマシンとの違いはどこにあるか説明せよ.
(6) パソコンでマルチプロセッシングが利用できる場合の応用を想像してみよ.
(7) コマンドはどのような役割をはたすのか説明せよ.
(8) OS/360 の果たした役割を説明せよ.
(9) システムコールとはどのような機能であるか説明せよ.
(10) キャパシティプランニングとはどのようなことか説明せよ.
(11) 多重プログラミング，マルチタスキング，マルチプロセッシングの各々の違いを説明せよ.
(12) 個人が主として使用するパソコンと不特定多数のユーザを相手とするサーバマシンとのオペレーティングシステムに求められる機能の違いを説明せよ.
(13) 分散処理の利点を述べよ.
(14) 大形メインフレームコンピュータのような集中処理の利点と欠点を述べよ.
(15) 個人が主として使用するパソコンにマルチタスキング機能を必要とする場合はどのような場合か考えてみよ.
(16) ブートストラップではどのような処理を行うべきか列挙せよ.