

タイムシェアリング・システムの設計哲学と使い勝手

高橋 延匡

(東京農工大学 工学部)

1. はじめに

システム設計の基本は「対象」を明確化することにはじまる。タイムシェアリング・システム(TSS)も、その誕生には、それなりの設計哲学にもとづいて設計された。オペレーティング・システムは管理システムであり、管理対象や環境が変われば当然、その形態も変わる(変える)のが当然である。ところが、IBMのOS/360以来、大型汎用のオペレーティング・システムの開発にドライブがかかり、必ずしも「使い勝手の良いTSS」という面での研究開発が推進されたとは思えない。

近年のオペレーティング・システム、特に汎用のオペレーティング・システムは益々大型化し、管理対象をふやしている。主記憶容量も半導体技術の進歩で低価格化が進み、8メガバイト位が手軽に実装可能になりつつある。ある意味で、仮想記憶方式のオペレーティング・システムとして、やっと性能も伴って実用になって来たとも言えよう。しかし、その根底の設計思想には「絶対的処理能力」に対処するバッチ処理が依然として存在する。しかし、市場拡大のため、最近ではエンド・ユーザ指向や分散処理を打ち出している。

MITのMULTICS¹⁾の開発において、Saltzer³⁾はオペレーティング・システムの問題を「技術的な問題」と「本質的な問題」とに分けて検討している。技術的問題は時と共に解決されて行くような問題であり、本質的問題とは、技術の進歩だけでは解決できない問題を指している。この中で一番大切な問題は、「人間と計算機システムとのコミュニケーションの問題」である。

「使い勝手の良いTSS」というのは、まさに、「人間と計算機システムとのコミュニケーションの問題」なのである。したがって、この問題の解決には、ユーザのニーズを的確に掴むことから始める必要がある。また、率いなことに技術の進歩はユーザニーズを満たし得る環境や手段をメーカーの設計者に与えつつある。一例をあげると、計算機の端末機の低価格化やモデムの低価格化などは、急激に、かつ容易にユーザ数の拡大につながっている。このことは、従来のように計算センタを介して計算機システムを使用するのと異なり、多種多様な種類の業務を、一部の情報処理の専門家だけが扱うにとどまらず、情報処理の素人が多数扱うようになる。したがって、「使い勝手」の研究は益々重要になる。

2. TSSとパーソナル・コンピュータ

「使い勝手の良いTSS」をユーザの立場から考えれば、自分自身にどれだけ奉仕してくれるか、ということが原点になる。しかし、そこには経済原則が厳然として存在する。たとえば、病院に入院している時には、医者の回診(タイムスライス)と、何かの時に看護婦が直ちに抱えてきてくれること(レスポンス・タイム)が評価規準の中では特に重要である。VIPで、それ等が独占できれば、それは最良にはちがいないが、一般には経済原則から言って不可能である。まったく同様に、超大型計算機システムを購入し独占的に使用できるユーザは対象にしない。TSSのユーザ層は、一人で大型計算機システムを独占できる程のユーザではない。

しかし、計算機を使う多種多様なニーズを持っている。そのようなユーザをどれだけ満足させられるかがTSSの研究開発課題となる。

たとえば、かつてTSSのサービスの主力が簡単な計算であった時に、電卓が開発された。電卓とプッシュホンでの計算とでは勝負は明らかである。現在、プッシュホンでは電話計算サービスと新幹線座席予約サービスと短縮ダイヤルサービスができるが、後者の2つは有効にちがいない。利用頻度が多ければ多いほど利益を生む。新幹線の座席予約は情報のサービス（コンピュータ・ユーティリティ）に对应しているし、短縮ダイヤルはアクセス手数をへらすという点で、いずれも基本に沿っている。

最近のパーソナル・コンピュータは、その低価格なことで、独占的に使用できるという側面から、汎用大型計算機システムのTSSと、ある面で競合関係になる。特に、計算機の使用目的が数値計算から非数値処理へ大幅に移行しており、その面でも発展が期待される。現在のパーソナル・コンピュータでは、使用言語に制限があり、また、ファイル・システムも弱体であるが、いずれ、16ビット・マイクロプロセッサが実装され、強力なシステム記述言語が提供され、ファイルシステムが充実する段階では、大型汎用計算機システムのTSSはデータベースのアクセスが主体にならざるを得ないだろう。

3. TSSとオフィスコンピュータ

定型業務を中心とするユーザで、従来の大型計算機を使用していないユーザに対し、オフィスコンピュータが大幅に売れている。数年前の小型事務用計算機から脱皮し、専用プログラム・パッケージを用い、その操作性の良さを売り物にしている。特に、伝票発行で代表される帳票業務には、ほとんど日本語（漢字）の入出力ができるようになってきている。これ等のオフィスコンピュータは必ずしも、従来のメインフレームが中心でなかったため、従来システムとの互換性の制限にこだわらなくても良いので、ユーザ専用のシステムを供給し易い立場にある。したがって、日本語の情報処理機能はメインフレームよりは数年早く採用している。しかし、業務の拡大に伴い、いずれシステムの拡張が不可欠となろう。その時、大型計算機が歩んだ道を再び歩くかどうか、重要な判断の分水嶺となろう。

このような定型業務の入力に関しては、慣れと共に、入力速度（操作速度）が容易になるように設計するのが望ましい。HITAC 5020 モニタのシステム・オペレータ・コマンドの設計では、各コマンドに機械語に相当するコード番号を設けて置いたが、ほとんどの操作は、単純で手数の少ない入力方法で行なわれていた。これは汎用のTSSを考える際に非常に重要なことである。

4. TSSの設計と汎用オペレーティング・システムとの比較

汎用の大型計算機システムのオペレーティング・システムを設計する場合に、基本的には2つの立場が存在する。いずれも高価なシステム・リソースを多数持ったため、マルチプログラミングという方式を採用することは当然である。その場合に、TSS的な使い方を主と考えるか、バッチ処理的なことを主と考えるかによって、オペレーティング・システムの設計方針が大きく変わらざるを得ない。前者に重点を置くならば、全体としての性能は犠牲にしても、ある単位の問題に対しては直ちにユーザに奉仕するように設計されねばならない。一方、後者の立場に立つ

ならば、当然スループットに重点を置くべきである。この点に関してはIBMをはじめとするIBMと互換性を持つメインフレームの大型機では、明らかに後者に重点を置くオペレーティング・システムから出発している。

表1 汎用TSSと汎用オペレーティング・システムの設計目標の比較²⁾

リモート・アクセス中心の汎用TSS	バッチ中心の汎用オペレーティング・システム
レスポンス・タイムが目標	スループットが目標
オンライン・アクセスが中心	バッチ中心
バックグラウンド・バッチ	タイムシェアリング・オプション
オンライン端末	リモート・ジョブ・エントリー
便利なコマンド言語が必要	バッチのジョブ制御言語と共通
ファイル処理はメモリとしてアクセスする	ファイル処理は入出力装置としてアクセスする
プログラムを任意時点で中断(Quit)再開できる	チェック・ポイントとして計画的に行なう
プログラムを少しづつ作る	プログラムはまとめて作る
ダイナミック・リンクが中心	スタティック・リンクが中心
プロテクションの強化が必要(特にプライバシー)	プロテクション, 特に安全性が重要, RASが重点
汎用言語(FORTRAN, COBOL)も必要だが, 多種類の言語(APL, PASCAL, LISP, ...)が重要	汎用言語が重点。特に FORTRAN や COBOL の性能が高いものを必要とする
テキスト・エディタの使い勝手が重要	FORTRAN のエディタや COBOL のエディタが重要
多種多様な端末に即した入出力形態を受け入れられること	ラインプリンタの性能が重要
コンピュータ・ユーティリティの提供	コンピューティング・パワーの提供

一方、前者の典型的な例は、MULTICS にみることが出来る。MULTICS 自身は、その理想を実現するには機が熟していなかったが、仮想メモリ方式は、現世代の中心的な方式として定着してしまった。

表1に、その設計目標の比較を示す。表1から明らかなように、スループットに重点を置いて設計されたシステムに対し、マン・マシン性の良さが不足している点を強調するとか、TSSを指向したシステムに対し、バッチの性能がバッチ処理中心の計算機システムに劣るなどの指摘は余り的を射たものとは言いがたい。もっとも最近の汎用オペレーティング・システムは分散処理指向、エンドユーザ指向を強調しており、表1の両方の性格を主張している以上、評価は両側から行なう必要がある。

5. オペレーティング・システムの設計哲学の見直しの必要性

計算機システムがあると、大型計算機に限らずミニコンからマイクロコンピュータまでオペレーティング・システムが存在する。勿論、その規模や機能は千差

万別である。しかし主たる仕事は、計算機システムの各種のリソースの管理をすることには変りあるまい。問題は管理を精密にしたがるかどうかである。管理の基本は、金額が高い物（設備）を有効に働かすことにある。TSSの発生の由来も一番金額が高いCPUを有効に管理運営しようと考えたところにあった。CPUやメモリの低価格化が進んでいるなかで、必要以上に複雑なメモリ管理を果してやる必要があるのかがどうか、その原点に帰って考え直す必要がある。幸い端末機に、相当量の主メモリや16ビット・マイクロプロセッサを持たせることは容易になった。TSSの本質であるレスポンス・タイムを良くするには、なるべく多くの管理的なことは端末側に持たせ、中央の汎用大型計算機システムは、むしろ計算機のエンジンの役割に徹すべきであろう。たとえば、絶対的処理能力を要求するならば、専用のバックエンド・プロセッサを開発し用いれば良い。同様に良く設計されたデータ・ベース・マシンがあればそれを利用すれば良い。

各端末機のオペレーティング・システムでは、汎用計算機のオペレーティング・システムと異なり、非常に単純なことができれば良い。ちなわち、その機能は、
(1) 割り込み処理などの基本的な制御を実行する、コアの部分
(2) 良く考えられたシステム記述言語と能率の良いコンパイラ
(3) ファイル・システム、特に2次記憶装置の物理的な性質も良く知ったうえで、将来を推察した使い勝手の良いファイル・システム
(4) 使い勝手の良いテキスト・エディタ
(5) システムの状態の把握が容易にできるコマンド言語プロセッサ

などであろう。さらに、他の計算機システムと接続するために最低限度の手続きのための処理プログラムがあれば十分であろう。

いずれにせよ、大型計算機システムの汎用オペレーティング・システムの設計は基本的に見直しを段階に来ていって考えられる。

6. 大学の情報工学科の教育とTSS

大学の情報工学系では学科の設備として計算機システムを設置している。教育にはFORTRANのバッチ中心のシステムは不向きである。その理由は、計算機を自由に使うことなし、種々の新しい問題に取り組むことを考えると、バッチ式のターンアラウンド・タイムを必要とする方式は教育には適していない。演習も当然何人かが並行して使用できることが望ましくTSSはその一つの解である。このような使用法においては、一般企業における利用形態とは異なり、強力な処理能力までは要求しない。むしろ、使い勝手の良いことと、ファイル・システムが安定し、アクセスし易いことが大切となる。そのような目で計算機システムを見る時、メガ・ミニ級の使い易いTSS中心の設計思想を持つシステムが望ましい。残念ながら、国産機では、この条件を満たすシステムが極端に少ない。

それでは、パーソナル・コンピュータで良いかと言うと、それだけでは、データベースや各種汎用言語やシミュレーションなどの面で制約があり、ある程度の規模のソフトウェア財産のあることは不可欠の条件なのである。

東京農工大学の数理情報工学科では、学科用の計算機として以下のことを念頭に置いて評価し、機種⁵⁾の選定を行った。

- (1) 技術の推移からして、ハードウェア・コストは度々下がる。主メモリや端末が増設可能なこと。当然、仮想メモリ方式のオペレーティング・システムで

あること。

- (2) FORTRAN の運転手を省てる必要はない。
 - (3) ペーパーレス・カードレスを指向したい。省資源に徹し、ランニング・コストを削減したい。
 - (4) プロジェクト MAC で提唱した、Machine Aided Cognition の実現をしたい。そのような使い方をこそ、教育すべきである。
 - (5) 教育が容易であると同時に、学生が自由に使える環境を容易に提供できること。
 - (6) 計算機の操作をさるような特別のスタッフを必要としないこと。無人運転装置などが望ましい。
 - (7) ソフトウェアが充実していること。
- 上記を参考にして、ニーズに合った TSS の開発を期待したい。

7. TSS の評価に関する視点⁶⁾

「使い勝手の良い TSS」と言うのは前にも述べたように、「使う人自身にどれだけ奉仕してくれるか」に依る。したがって、あくまでユーザのレベルとの相対的な関係である。ここでは、情報工学科の学生を対象とした場合について考察する。当然、使用者のレベルは、(1) まったくの入門者と (2) 数百時間使用し、すっかり慣れて使いこなしている者が同時に存在する。筆者は評価の規準として以下の諸点を考えている。

(1) ユーザの手数が少ないこと

何手で端末を操作できるか、ということが第一の基本である。パーソナル・コンピュータと異なり、何人がで共同で使用する以上、ある程度のオーバーヘッドは避けられないとしても、できるだけ単純なことが好ましい。すべての面で手数を如何にへらすかについて考えて設計することが大切である。

- a. ログインやログアウトの手数が少ないこと。
- b. コマンド名など、基本単語は短かいこと。また、自然な用語を用い、頭にひっかがるような用語を用いないこと。
- c. コマンドやパラメータのデフォルトが適切であること。
- d. タイプミスの処理の手数が単純なこと。特に、出力形式の違い (ローラペーパーへの出力、ページ単位の出力、ディスプレイへの出力) を十分考慮すること。
- e. テキスト・エディティングの使用頻度は高い。手数が少なく強力であること。

上記の諸点は設計者は当然十分考えていると思うが、実際の使用にあたっては、TSS の基本的な設計面が重要になる。ちなわち、以下の諸点である。

- f. 中断や再開の手数が少ないこと。一寸の操作ミスで、一行入力し直しや、ファイルの入力し直しが起ったのではたまらない。TSS では、入力の途中や計算の途中で、割り込んで気のついたことを直したくなる。そのようなことが容易にできることと、もとへ容易に戻れること。また、不意の中断に対しても、再開時に状態が把握できること。
- g. 実行時の手数が少ないこと。ダイナミック・リンクができること、特に、ダイナミック・リンクができるかできないかは、プログラムの部分的テスト

ができるかどうかにかかわっており、実行時の手数に大幅な影響を与える。
h. ファイル・システムが明快であること。ユーザはできる限りファイルを特別な入出力装置と関係づけたくない。1レベル・ストレージの概念が望ましい。

(2) 学習の手数が少ないこと

TSSの使用法の学習が容易なこと。特に、マニュアルがユーザの身になって書いてあること。ポテッタブルで立ちようになること。また、マニュアルを探さず時間が短かいこと。これは、学生に対しては特に重要である。

(3) TSS使用中の知識の取り入れ方の手数が少ないこと

端末使用中に、自分の状態がわからなくなった時、容易に自己の状態がわかること。

a. 特に、エラーメッセージが読んですぐ理解できること。ユーザが対応できない意味不明のメッセージや、指示のないメッセージを出さないこと。

b. プログラムの再開、中断が容易なこと。

(4) 管理者の手数が少ないこと

センタの運営は無人運転が望ましい。

a. 学期の開始時の学生の一括登録が容易なこと。消去も簡単なこと。

b. 毎日のシステム・スタートやシステム・ストップの手数が少ないこと。

(5) TSSのパフォーマンス、特にレスポンス・タイムが速いこと

(6) TSSの機能として是非考えてもらいたいこと

a. システム記述言語をユーザにも開放すること。

b. CAIのような会話型処理サブシステムが容易に設計できること。また、容易にテストできること。

c. 多人数でサブシステムを作成できること。勿論、ライブラリの登録などが容易にできること。

8. おわりに

TSSの使い勝手の良否の判断は、ある一面を取りあげて論ずることも大切であるが、現実の計算機システムでは設計哲学や設計目標の差から生じている問題点もかなり多く存在する。いずれにしても、日本の計算機メーカーのTSSでは、日本の工業製品に多く見られる「日本的な気働き」の良さが見当たらないことは悲しむべきことである。人間工学的側面の重視などと言う前に、ユーザの身になって気働きのしてほしいものである。そうやってこそ、国際商品へと成長すると確信している。そのためには、設計者自身が徹底的に、種々のTSSを使って実際の仕事をしてみる必要がある。

参考文献

- 1) F.J. Corbato, V.A. Vyssotsky: Introduction and Overview of the MULTICS System, AFIPS, Vol. 27 (1965 FJCC)
- 2) 高橋達匡: オペレーティングシステムの評価, 情報処理 Vol.16, pp453
- 3) J.H. Saltzer: Traffic Control in a Multiplexed Computer System, MAC-TR-30 (1966)
- 4) Eititi Wada and N. Takahashi: Some Features of Interaction on the HITAC 5020 TSS, HITACHI REVIEW, Vol.17, No.11, 1968
- 5) 中森真理雄 他2: 情報工学科の計算機導入に関する一評価式の提案, 昭和53年度電子通信学会総合全国大会(1986)
- 6) 小谷善行 他2: タイムシェアリングシステムの評価に関する考察(同307)