

一般利用者の立場からみたT S S

木村 泉
(東京工業大学理学部)

・はじめに

はじめての計算機システムに対面したとき、われわれはとかく欲求不満を味わう。計算機システム一般に関して十分な常識をもっている利用者の場合であってさえ、そのシステム固有の約束ごとを習いおぼえるまでは何となくがたがたする。システムが利用者に一般常識を要求することは理の当然であるが、システム固有のこまかいくせに関する精通を要求することは好ましくない。しかるにそういう不当な負担を利用者に課しているシステムは多い。

例をT S Sにとろう。どんなT S Sにもハッカー (hacker) と呼ばれる人種がまつわりついている。システムのこまかいところをよく知っていて、常人には思いもよらないようなことを小細工 (ハッキング—hacking) によってやってのけてしまう。彼らは、その神秘的な能力ゆえに端末室などでは至って大きな顔をしている。一般利用者は、何となく厭然としないながら、圧倒的な実力差を認識してだまっている。

これは実によく見かける風景であるが、あるべき姿とは思われない。ハッカーの跳りようを許すシステムは悪いシステムである。理想をいえば、システムは計算機技術に関する一般常識をしっかりと身につけた紳士淑女ならばだれでもさっと使える、というものであるべきだ。システム固有の知識がまったくいらないというわけには行かないかも知れないが、必要な知識の量は少なければ少ないほどよく、また適当なマニュアルないしオンラインドキュメンテーションによって容易に入手できるようになっている必要がある。

本文では、こういう意味でのT S Sの「とりつきやすさ」について、二、三のT S Sにとりついてみた経験に基づいて論じたい。具体的には筆者 (どちらかというと事務計算向きのパッチシステムと、本来制御用として設計されたミニコン プュータに慣れた) が、通算約/年8か月にわたって 米国に滞在していた間に、一般利用者として (つまりシステムそのものについて知りたいよりはそれを使って何かほかのことをしたいという動機で) 接触した3種の代表的T S Sについて、どんな経験をしたかを述べ、それをもととして上の問題を考えてみたい。

1. 対象システムの概要

使ってみたシステムの具体的な名称をここにかけることは 必要とは思われないので、それらを仮にシステムA、B、およびCと呼んでおくことにしよう。いずれも第一流のシステムといってよく、それぞれに人気の高いものである。システムA は計算機科学関係の研究・教育機関で広く使われている標準的なシステムであって、かなり大規模なハードウェアを前提としている。ただし実際に使った版は商用の標準品ではなく、「地元」のハッカーたちが手を入れて拡張したものであった。システムAは、そのコミュニティーの表通りにあたるシステムであって、日常的な計算、思考、意志疎通の道具として使われていた。

システムB は、「流線型」の、使いやすいシステムとしてかねて定評があり、大型機よりは中小型機、大量計算よりは文書処理に向けたシステムである。たまたま使ったシステムは、特殊プロジェクト用として導入されたもので、一般公開はされておらず、筆者としてはやや遠慮がちに使ったかたむきがある。

最後にシステムC は、パーソナルコンピューティングの 将来をさぐるために、ある企業の内部で開発されたものであって非売品であり、最近米国内の二、三の有力計算機科学科に無償配布された。強力な入出力機器と高度に洗練された利用者界面をもち、開発者の一流の頭脳が一見して明らかであるようなシステムである。どうぞ使ってみて下さい、というスタイルでころがしてあり、部外者がしのび込んでゲームのプログラムに熱中している光景もしばしば

見られた。(このシステムは厳密にいうとTSSではないが、同様の外観を呈するものという意味でここに含めた。)

なお以上3システムは相互にネットワークでつながっていた。

2. 解説文書の整備状況

ここで、A、B、Cの各システムについてどのような解説文書(紙に印刷された)が用意されていたかを記しておくことにしよう。以下は、あくまで特定の計算機室の状況がどうであったかを述べたものであって、システムそのものの一般的文書整備状況はまた別の問題であることを明記しておきたい。

システムAについては、その計算機室の利用者用入門マニュアル(約200ページ)にファシリテイ全般に関する概説の一部として簡単な説明があり、おもなコマンドの名前とそのおよその意味、ファイルの名づけかたに関する約束、主要言語処理系の操作例などが出ていて、一応の手がかりはある仕組みになっていた。それ以上くわしいことは計算機メーカーが発行したマニュアルを見ればわかる手筈であり、共同利用の端末室にはそれらのマニュアルが種々ころがしてあったが、

1. さまざまな版のマニュアルが混在していて、どこにもっとも適切な情報があるか知りにくい。

2. 地元での手なおしがあるためマニュアルの記述が必ずしもそのまま通用しない。

3. しばしば表紙がとれていたり、前回見たときはあったものが次回には散逸していたりする。

などの事情があって、あまりたよりになる状態ではなかった。のちに体制が若干整備され、おもなものについてはメーカーから一括購入したマニュアルを/冊5ドルの保証金をおさめて借り出すことができるようになったが、それもなく一般的なものに限られ、どんな情報でも得られるというにはほど遠い状況であった。

ポイントは、そういう乱雑な状況にもかかわらず、実際にはあまり不自由はなかったという点にあるが、後述する。

システムBは、資源の関係で利用が限定されていたほか、導入に関するライセンス条件によってマニュアル類の配布方式に制限があった。システム利用について担当者と話がつくと、標準マニュアルに地元での追加をふくめたものを借りることができる仕組みであり、状況は要するに標準システムを標準マニュアルで使う、というに等しいものであった。一般利用者は見ることを許されなかったものの、システムの内部構造に関する情報も地元システム管理者はにぎっており、疑義が生じた場合はそこに話を持ち込めばすむ体制になっていた。

システムCは元来商用のシステムではなく部内用のシステムを公開した、というだけのものであったが、それなりに整った、読みやすい一般利用者向けハンドブックを伴っていた。その記述は厳密さよりはとりつきやすさ、才気、楽しさをねらいとしているものといえた。ただしちょっとつまんだこと(文書ファイルの形式など)についてはもはや記述がなく、変ったことをやろうとすると途方に暮れる面があった。また、たとえば地元のプリンタがどのような名称でシステムに登録されているか知らなければ文書の印刷はできないのであったが、その種の、計算機室固有の事項に関する情報は文書化はされておらず、地元担当者にきくしかなかった。

以下これらのシステムについて筆者がどんなことを経験したかを、いくつかの事例を例にとって述べたい。

3. コマンド名とファイルの命名

ごく簡単な話からはじめよう。新しいシステムに対面したとき、まず問題になるのは適切なコマンド名がさっと意識にのぼるか、ということである。

たとえばシステムAでは、ファイルを削除するには

del ファイル名
と打つ。(del は delete と打つのが正式であるが、コマンド名はほかのコマンドと区別がつく字数だけ打てばよい、という一般的約束があるので del で十分である。)
システムC でも、ファイルの削除は del でよい。次にスペースを打つとシステムが ete をおぎなうて、スクリーン上には
delete _
という形のものができる。(_ はカーソル)あとはファイル名を打てばよい。

これらはまずは常識的といえる。実際、ファイルの削除に関するコマンドを delete という語を短縮したものを選んでるシステムは多い。

ところがシステムB はその点奇妙に個性的である。はじめて対面した利用者がとっさに del と打つと
「そんなプログラムはない。」とっておこられる。筆者自身、わかっていたのに、ついやってしまった。正しくは rm でなければならない。rm は remove の略である。

これは本質的にはどちらが先に流行したかの問題にすぎない。rm も立派なコマンド名であるが、残念ながら del がすでに流行しすぎている。新入りは最初に / 回とまどうだけでなく、実はあとまで何かの拍子にとまどうことが多い。そしてうっかり del と打ったときの反応が、たとえば「コマンドの綴りちがい」というような、まだしも利用者の心理的実態に近いメッセージではないので、初心者はやけいとまどうのである。システムB はきわめて一般的にすっきりとできており、すべてのコマンドがプログラムの呼び出しと解せられることになっている。rm も実はプログラムの名前なのである。したがって必然的に以上のようなことになる。これは一度慣れてしまえば実に快適だが、その状態に達するまでの障壁は高い。痛しかゆしである。

なおシステムC もこの、コマンドとはプログラム名のことなり、という約束を使っているが、より伝統的なプログラム名を使っているほか、システムからの確認の打ち返しがあるので、この特定の場合に限っていえばずっと問題が少ない。

4. 事例2 — ダンプと dump

次も多少似たような話である。

システムA で、ファイルのダンプを取りたいと思った。文書ファイルをごく常識的な Fortran プログラムで加工しようとしたのであるが、いろいろ奇怪なことが起ってどうもうまく行かないので、原因をつきとめたいと思ったのである。(その原因は本質的には、/ 語36ビットの中に / 字7ビットの文字を5個つめ込み、行の終りでは語の残部に無効文字を置き、しかも36ビット中 / ビット余ったものを、行番号の表示に関連して特殊な使いかたで使っている、というところにあると判明したが、それはともかく) さっそく入門マニュアル(前述)を開いてみたが、dump というコマンドはないらしい。メーカーのユーティリティプログラムのことを記したマニュアルを端末室のすみから発掘してめくってみたが、やはりびったりしたものが見当たらない。(大部のものを立ち読みしたので

見落したのかも知れない。) 地元ハッカーの一人をつかまえて、「ダンプってどうやるの。」ときいてみた。するとさっそく教えてくれたのは、まずタイプライターをディスクであると宣言して、次に某々のファイル領域の中の何々というプログラムを起動せよ、という奇怪なものであった。よくわからないままにやってみたらうまく行った。そのダンプのプログラムはなかなかよくできていて、以後重宝した。

のちにシステムC で、やはりダンプがやりやすくなった。システムC ではいろいろな字体の文字が端末スクリーン上に出せる。ひらがなやカタカナを出したいと思ったが、字母ファイルの形式がわからない。提供されたドキュメント

類の中にはどうもびったりした情報がない。そこで字母ファイルをダンプしてみたいと思ったのである。

システムC ではファイル名をみればそれがコマンドとして起動できるプログラムかどうかはわかる仕掛けになっている。そこでファイルの一覧表を打ち出してながめてみると `dump` というコマンドがたしかにある。さっそく「字母ファイルX X を `dump` せよ。」とやってみた。あっという間に実行は完了し、起こったのは、その字母ファイルの中味が空になる、というだけのことであった。それを復元するのに苦労した。

システムB ではダンプを取りたいと思ったことはない。動機がなかった。システムB のファイルは基本的には8ビット単位の文字がびっしり並んだ、という形をしており、本来奇怪なことが起こりにくいのである。システムB のファイルシステムは明白性（*understandability*）の点で大変すぐれている、といえる。

5. 事例3——一般原則のかくれた例外

先に述べたように、システムA ではコマンド名を他のコマンドと区別のつくところまで打てばよいことになっている。ところがこの規則にはかくれた例外があり、ちょっと混乱した。

システムA では

- `save abc`

などと打つと、その時点でできているコアイメージをファイル `abc` に保存することができる。

（実際には `abc.XXX` という形の名前をもったファイルが二つできる。）それを呼び出して走らせるコマンドは

- `run abc`

である。

ところでシステムプログラムを呼び出すコマンドが

- `r プログラム名`

という形のものであることは、入門マニュアルの一番はじめに書いてある。てっきり `run` は `r` と同じもので、そう略してよいのだと思い込んでしまった。

実際には、`run` は `r` と区別するために `ru` と打たなければならないのであった。入門マニュアルをにらんでいる間はついにわからず、端末室のすみにすわり込んで、メーカーのマニュアルを熟読して、やっとわかった。（端末室には端末の数しかイスがないので、これはなかなか骨が折れた。）これはドキュメンテーションがわるかったともいえるが、もとの設計にも問題がなかったとはいえないであろう。

6. 事例4——`help` と `man`

システムA には `help` および `doc` なる2種のドキュメンテーション用コマンドがあり、

- `help プログラム名`

とするとシステムプログラムについての手短かな説明が、また

- `doc プログラム名`

とするとくわしい説明が、端末に打ち出される仕掛けになっていた。`doc` の方は情報の入っているファイルをエディタを使って読んだり、高速プリンタに打ち出したりすることも可能であった。`help` および `doc` の標準的な使いかたは、まず `doc` のファイルをエディタでちょっとのぞき、様子をたしかめてからプリンタで打ち出し、それを持って端末の前にすわり、`help` 情報をたよりに使ってみる、というものであった。これらはまったく常識的なことではあるが、ありがた味は大きかった。特に `help` 用、および `doc` 用のファイルが頻繁に更新され、ほとんどつねに

最新の情報を与えている点、ありがたかった。どちらかというと help の方が余計役立った。

(doc の情報は一般に長すぎ、腰を落して読まねばならず、特に端末上では扱いにくい感があった。)

たとえばあるとき、数万件の文字データを整列するという問題が発生した。そこでともかく

・ help sort

とやってみた。出てきたのはコマンドの表で、端末の画面一枚におさまりきらないのはちょっと閉口であったが、ともかくマニュアルをいっさい見ることなく、その表を見るだけのことで目的を達することができた。入力が可変長の行から成るときでも出力は固定長になってしまうなど、きちんと書いてないこともたくさんあって何回か驚かされたが、それは別に致命的なことではなかった。

そのほかネットワーク経由で他のシステムから情報を取り出すプログラムなど、プログラムの名前だけ知っていれば、あとは help の情報と常識だけで何とか使いこなせた。プログラムの名前は どうやって知ったかという、ハッカーからの口コミのほかユーザが思い思いに登録した汎用プログラムのデータベースを見る、知っていそうな人に電子郵便を送ってきく、それでもわからなければ電子掲示板を使ってみんなにきく、などの方法によった。総じてオンラインドキュメンテーションの威力は絶大である。

システム B では、doc にあたるものとして man という名のコマンドがあり、これと呼び出すと標準マニュアルの、指定されたプログラムに関するページが端末に出てくるようになっていたが、ディスプレイ端末を使っていた関係もあり、システム A の help ほど具合よくなかった。

システム C でも、マニュアルにはオンラインドキュメンテーションをネットワーク経由で取り出す機構についての記述があったが、これについては使ってみるチャンスがなかったので論評をさしひかえたい。

7. 結論

以上を振り返ってみると、次の諸点が目立つ。

1. オンラインドキュメンテーションはきわめて有力で、使いやすい。特に一役常識をもった利用者に向く。くわしいもの、粗略なものとも、それぞれに有用であるが、特に / 行サマリー的なものが有効である。
2. ただし / 行サマリーが使いものになるためには、システムが全体としてすっきりできていて、一部分から全体が類推できることを必要とする。またコマンド名などのような、どのようにでもきめられることについては、平凡であることをもってよしとする。
3. 口コミは非常に重要である。特に、ハッカーと呼ばれる人たちは多くの場合善意に満ちていて、自分の知識が人の役に立つことを大変よろこぶ。端末室の物理的な居心地をよくし、電子郵便、電子掲示板のようなコミュニケーションの手段を整備することは、システムの運用をスムーズにする上で思いのほか大きな効果をもつように見受けられる。
4. 紙に印刷されたマニュアル類は、紙媒体という点ではよいがアクセスがわるく、いざというときに役に立たないことが多い。また内容が時とともに実情からずれて行きやすく、その際の更新がきわめてやっかいである。さらにマニュアルに書いたことは知られたくないあいでも知られてしまう、というアクセス制御上の問題もある。

これらの考察から導き出される一つの結論は、とりつきやすいシステムにはファイル容量がたいせつ、ということである。それと同時に、よいシステムはでき上がるまでに時間がかかる、ということもいえる。オンラインドキュメンテーションは、だれかが書くのであるが、この性質上急いでがんばってまとめて書いてもろくなものはできな

い。ファイル容量に十分な余裕をもうけ、ハッカーたちを上質の素人と接触させ、ハッカー、素人双方の経験に基づいて少しずつ気楽に書きためて行くことが唯一の道であるように思われる。

8・補——紙がダメである理由

本文を終わるにあたり、前節のような結論が導かれるについての心理的背景について一言述べておきたい。

T S S の利用者の注意の中心は、通常ディスプレイの画面にある。(指ノ本でタイプライタを打つ習慣の利用者は別として、欧米の) 通常の利用者は、画面を見ながら鍵盤をたたく。鍵盤は見えない。ときには画面から目を離してメモや原稿を見ることもあるが、一般にはそういうことをするたびに作業、思考のテンポががったりとくずれる。例外は計算機からの反応が問題でない場合であって、文書エディタを入力モードに切り換えて原稿をどんどん打ち込むときがこれにあたるが、これはあくまで例外である。

さてそうすると、一般的常識はすでにもっていて、こまかいところのいいましその他だけがまだよくわからないような利用者にとって、マニュアルは有難いものではない。目を画面からはずしてじっくりと読まなければならないのでテンポがくずれる。これに対し、オンラインメッセージによるなら、目をあちこち動かす必要はない。必要なメッセージの量が画面におさまり切らないときは問題だが、高級素人利用者は簡明なメッセージから多くのことを察する能力をもっているものであるから、この点もさほどの大問題ではない。help が紙にまさるゆえんである。

なおオンラインメッセージを簡明にするための一つのコツは、要点 (たとえばコマンドの形式を示す B N F) を / 個所にコンパクトにまとめることである。そしてそれが、画面になるべく長い時間残るようにする。たとえばスクリーン上に窓を定義する機能 (システム C はそれをもっていた) があって必要な情報をそこにとっておけるなら理想的であるし、情報がどんどん流れて行く (スクロールする) 形式のディスプレイの場合であっても、要点をメッセージの最後に出す、というようなくふうは、いくらでもできるものである。