資料 5

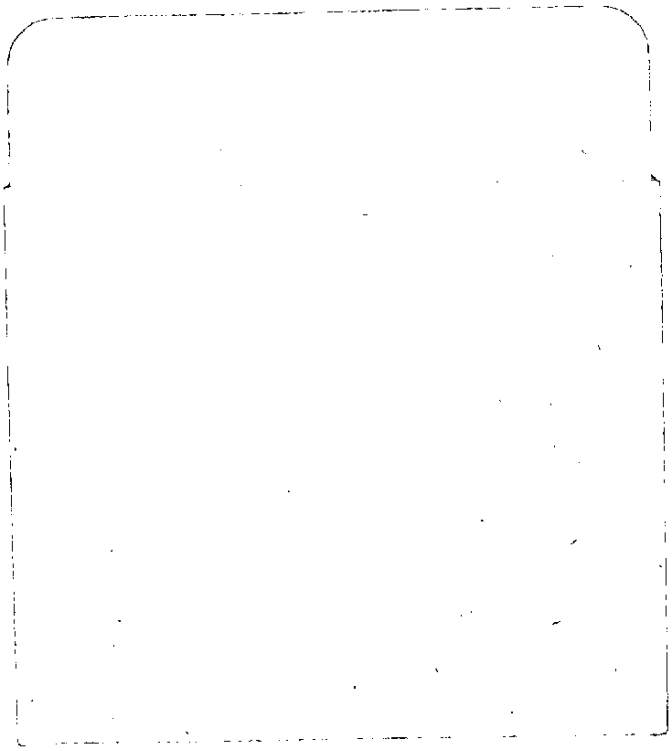
第５世代の電子計算機に関する調査研究報告書

―― ユーザ援助情報システム ――

昭 和 55 年 3 月

財団法人 日本情報処理開発協会

この報告書は，日本自転車振興会から競輪収益の一部である機械工業振興資金の補助を受けて昭和54年度に実施した「第5世代の電子計算機に関する調査研究」の成果をとりまとめたものであります。



目　　次

1．総論―使い易いシステムの探究

1. 総論 － 使い易いシステムの探究

1.1 調査研究の目的と方法

1990年代の情報システムに要求される重要な特性に"使い易さ"があげられる。情報システムが今後広範に普及し，人間社会に円滑に浸透してゆくためには，人間社会に広く愛され歓迎され，積極的に利用されるシステムでなくてはならない。このためには，使い易さの問題をいろいろな側面から研究・調査し，その実現に必要な研究項目を明らかにすることが重要である。

最終ユーザから情報システムをみた場合に，対象としている分野の情報に明るい専門家に個人的に相談し，必要に応じて仕事を依頼する場合と同じように対応してくれることが，"使い易さ"を一つの側面からみた場合の理想像ということができる。

「ユーザ援助情報システム」という調査研究題目は，このような側面から命名した題であるが，これでは飽くまでも限られた情報システムを対象とし，限られた立場からみた使い易さに限定される恐れがある。

本研究では従ってこの題目に余りとらわれずに，使い易さの問題を幅広く，いろいろな立場から検討するように心がけることにした。

本研究には下記四研究室の教官・職員，博士コース大学院学生の参加を求め，全員の参加する研究会を毎月開催して"使い易い計算機システム"について総合的に討論を行なうと共に，各自が分担調査した調査項目について報告し，その討論を通じて"使い易さ"の全体像を明らかにするよう努めた。

元岡研究室－元岡達・斉藤禎・上森明・鈴木達郎，和田研究室－和田英一・戸村哲・近山隆，井上研究室－井上博允，藤田裕二，田中研究室－田中英彦・和賀井フミ子

本報告は中間報告であり，これまでに行なった討論結果の概要と，調査事項に

ついて報告する。来年度以降の研究を通して，一つのまとまったシステム像を提示し，その実現に必要な研究項目を列挙してみたいと考えている。

1.2 使い易さ

使い易さの問題は対象とするシステムの最終ユーザにとっての使い易さに終局的には帰結できるにしても，思考の段階ではいろいろな立場からの使い易さを考える必要がある。例えば"拡張性を持ったシステム"にすることは，定形業務にそのシステムを用いているユーザのその日の業務といった立場からみた時は使い易さに関連のないシステムの特性であるが，環境の変化を伴うシステムのライフサイクルでみると最終ユーザにとっても，使い易さに関係する重要な視点になる。

柔軟性のように最終ユーザにとっては，使いにくさの原因となるおそれのある特性であってシステム管理者の立場からみる時には，使い易さを考慮する上で最も重視しなければいけない特性もある。適当な妥協点を見い出すことも総合的な"使い易さ"の観点からは重要である。

以下の議論の中で，念頭にあるシステムは情報工学の研究者達が研究を続けてゆく上で欲しいシステムが主となるので，"使い易さ"の評価にこのような点が強く反映されている恐れのあることを指摘しておく。研究・開発を目的に使うシステムであるから，非定形業務であり，汎用性が重視されることになる。

さて使い易さについては，機械を人間に合わせることのできるシステム，使い慣れたシステムといった総括的なとらえ方があり，重要な観点ではあるが，ここでは以下に述べる四つの立場から"使い易さ"について考察してみることにする。

(1) システム構成の立場

(2) マン・マシンインタフェースの立場

(3) 知識ベースと言語の立場

(4) システムの性能の立場

第一のシステム構成の立場からいえることとしては（ａ）使いたい時に使える

端末が手元にあること，（ｂ）信頼性が高く何時でも使えること，（ｃ）計算機システムの構成とユーザ組織の構成との間の対応がよくついていること，（ｄ）拡張性・柔軟性があり，ユーザ側の要求の変化に追従できる適合性を持つこと，（ｅ）保守・管理が容易なこと，（ｆ）システムの状態をいつでも容易に把握できること，などがあげられる。これらの多くは今日の分散データ処理システムの目指している目標であり，今日行なわれている開発努力が順調に進展すれば，この面からの大きな隘路は生まれないように思われる。ＬＳＩ化はシステムの分散を容易にし，（ｃ）の要求を満たすに必要な構成の自由度を大きくする方向にある。今後の経緯を見守る必要のある事項としては，通信のコストがあげられる。光通信・衛星通信など技術的な break through はあるが，これらが長距離通信にどのように生かされるかによって，システムの構成法が大きな影響を受けることになろう。

第二のマン・マシンインタフェースの問題は，第２章で詳述されるように，使い易さを向上させる上で今後に大幅な改善が期待されている分野の一つである。人間工学的な側面の研究も必要で，地道な努力の集積が要求されよう。

人間が意志ないし情報を伝達する相手として主として他の人間を考えてきたことからいえることは，人間と同じ感覚器官を持ち，また人間の持つ感覚器官に有効に働きかけることのできるシステムが使い易いシステムの一つの理想形であるということである。

この観点から先づいえることは今日の計算機の入力メディア，出力メディアが単純すぎ人間の限られた感覚しか使っていないことである。出力はラインプリンタにしろ，表示装置にしろ，視覚に訴える文字の羅列であり，音による聴覚も殆んど使われていない。臭覚や触覚は特殊な場合に限るとしても，視覚と聴覚に同時に訴える程度の努力は早急に実施すべきであろう。会話において話し方の抑揚や顔の表情による意志の伝達の重要さを考えると自然な情報伝達の難しさがよくわかる。同じ視覚にしても文字だけでなく図表，画像などによる情報伝達が多くの場合に大きな効果を発揮することはよく知られているが，今日の情報システム

で十分生かされているとはいえない。

入力についても鍵盤入力，タブレットによる指示では人間の自然な情報伝達手段でないことは明らかである。人間にとって今日一番自然な情報伝達手段と考えられるのは音声による話し言葉であり，次が文字による文章であろう。出力側ではプリンタが用いられ，音声出力も実用化が進められているのに較べると，入力側の立ち遅れが目立ち，そのことが今日の情報処理における最大の隘路のあり場所を示しているといえよう。最近のＬＳＩ技術の進歩によって音声入力の実用化も時間の問題となっており，印刷文字や手書き文字の認識システムも実用化が進んでいる。音声認識・文字認識共に人間の認識能力レベルとは尚格段の差異があり，この差を狭める研究努力を地道に続けることが使い易さ追求のために必要である。画像入力にはＴＶ技術・ファクシミリ技術が利用でき，その後の処理能力に問題がある。図形入力は一見簡単そうであるが，図形が本来抽象化された概念の表現形式であることもあって自動化の難しい領域である。

入力については音声・文章・図形・画像・物体といった各種の媒体を取扱えることと共に，多種の媒体による入力方式を有機的に結合して，必要に応じ最も使い易い入力媒体に自由に切り換えられることが望ましい。この場合，入力内容の即時出力と，それを用いた自然な修正・編集機能を備えていることの重要性はいうまでもない。

計算機内部におけるデータ処理の効率を重視する立場から，極端に抽象化された本質的な情報のみが入出力の対象として用いられる傾向にあったが，この抽象化とその逆の具象化のプロセスを人間の側から計算機の側へ移すことが使い易いシステムを実現する上に必要になってくる。非数値データの処理についての研究はこの目的を達成するのに重要である。

コンピュータと人間の能力差を考える時，感覚器官や口・手・足といった出力器官の差と共に大きな差と考えられるものに知識の差がある。コンピュータには記憶装置がありデータの蓄積は行なわれているが，これは蓄積であり，頭脳による記憶とは多くの点で差がある。記憶装置ではデータの内容とは無関係な格納場所

を表わすアドレスによって読み書きが行なわれるのに対して，頭脳の記憶は記憶内容の関連を用いた連想機能によって読み書きを行なっている。近年のデータベース管理機能の研究や知識データベース，知識工学の目指すところは，頭脳の持つこのような連想機能を夫々の立場から部分的にでも実装することにあるということができよう。

頭脳に近づけることは"使い易さ"実現の上で最も重要なことの一つであることはいうまでもない。第三の知識データベースと言語の立場からの"使い易さ"の議論としては，"知識をコンピュータが備えた時にどのような使い易さが生まれるか"といった立場や，"知識を引き出すための言語の役割"といった立場からの議論が考えられる。

一口でいえば"常識を持ったシステム"にすることが使い易いシステムに不可欠な要因であるが，常識をシステムに実装する方法にはいろいろな方法が考えられ，またある人の常識が常に通用する訳でもない。しかし常識のないことがソフトウェア作成の基本的な困難となっていることも事実であろう。ソフトウェア作成の際，正常時のアルゴリズムだけを指定し，例外時処理を指定する必要がなければ，ソフトウェア作成上の困難の大きな部分が解決するのではなかろうか。システムの常識に期待するのは，このような例外処理である。

データ入力の量なども，一度入力したものが保存され，自由に引き出して使えれば，可成り減少するし，常識乃至知識があれば入力しないで済ませることのできる入力データも多い筈である。データかプログラムかのいずれを問わず，情報伝達の量を必要最少限に済ませるためにはコンピュータの持つ常識及至知識を活用することが考えられ，入力データ量の減ることは使い易いシステム実現に大きく貢献すると考える。

使い易さの問題として自然言語によるコンピュータとの意志の疎通の問題がある。自然言語の持つあいまいさは逆にシステムを使いにくいものにする恐れも多い。あいまいさをカバーするものは常識であって，従って自然言語の問題と知識データベースの問題が密接に関連してくることになる。

Q－Aシステムのように，本来あいまいさを前提とする場所で自然言語ないしそれに近いものの使用を考えるのは当然であるが，逆の極端としてデータ処理のアルゴリズムの記述に自然言語を使うことはシステムを逆に使いにくいものにしてしまう恐れが多い。コンピュータシステムへの自然言語の導入は後章で論じるように対象を考えて進める必要があろう。

自然言語，特に日本語の場合，構文規則の自由度が多く，不必要にあいまいさを導入する危険がある。ある程度構文規則を制限することによって正確な表現法を規定し，コンピュータにも理解し易い標準日本語を作ることも"使い易い"システム実現には必要ではなかろうか。

自然言語までゆかなくとも，書き易く，理解し易い人工言語を使うことは"使い易さ"を向上させる上に大きな役割を果す。非手続き言語の導入なども一貫であり，ＱＢＥのように通常の言語でなく図表のようなものを用いて意志の疎通をはかることも目的によっては使い易さに大きく貢献するものと考える。プログラム作成用ツールの開発も同様に大切である。

最後にシステムの性能・機能について述べる。余りに当然のこととして"使い易さ"の観点から余り論じられないが，性能／価格化が優れ，経済性を余り気にしないで使えることは"使い易い"システムにとって必須の条件である。同様の観点から応答時間が短いこと，信頼性だけでなく，データ保護を始めとするSecurityやIntegrityの問題も重要である。

仮想記憶，仮想端末などのようにシステムの細部を考えずにソフトウエアを作成するのに役立つ各種の仮想化された機能も使い易さに役立つことはいうまでもない。

データベース管理技術，網を意識させないプロセス管理技術などの考え方も一種の仮想化技術と見ることができ，"使い易さ"に対する計算機技術としてのアプローチといえよう。浮動小数点表示で指数部を可変長にする提案なども抽象データ構造などと共にプログラムの書き易さを狙った動きとしてとらえることができよう。

以上使い易さの問題についていくつかの側面から概観したが，以下の各章では，夫々の側面からみた使い易さの問題を詳述し，関連技術に関する調査結果を報告する。

2．マン・マシン・インタラクションの立場からの検討

2. マン・マシン・インタラクションの立場からの検討

ある分野のエキスパートに相談して仕事を依頼するとか，自分のアシスタントにおおまかな指示を与えて仕事をしてもらうというような場合に，エキスパートやアシスタントを情報システムにおきかえて考えてみる。人間は人間同志のコミュニケーションに慣れているから，情報システムと対話する場合も，あたかも人間と対話するような形で意志や情報を伝え合うことができれば，それは自然であり，使い易いシステムと言えるであろうし，人間と情報システムの間のインタラクションのひとつの理想形であると考えられよう。

人間は，音声による会話だけでなく，必要に応じて，文書，図面，写真，実物を示しながら対話する。顔の表情，目の動き，身ぶり手ぶりをおりまぜて，意志や情報の伝達に努める。人間同志は常識という共通の基盤をもつから大筋だけを伝えれば，通常正しい理解が得られる。このような人間同志の円滑なインタラクションに比べ，人間と情報システムの間のインタラクションは，まだまだ一面的であり，形式的であり，変化に乏しい。技術的観点から見れば，使い易さは，情報システムの持つ常識と，入出力メディアに深くかゝわっており，個々のメディアの高度化と共にそれを有機的に統合していくことが大きな課題であると考えられる。以下本章では，人間工学の立場からの人間および情報システムに対する考察と，入出力メディアの現状と将来について述べ，マン・マシン・インタラクションの立場から情報システムの使い易さについて検討する。

2.1 人間と情報システムの比較

本節では，人間工学の立場から，人間と情報システムの特性をいくつかの観点から比較してみる。

(1) 情報の入力機能

人間は五感を同時に使って多様な情報を解釈し，選択しながら入力できる。人間の感覚機能の特徴は適応性及びパターン認識と結びついた知覚である。適応性の点では，例えば視覚では，検出感度の調節と感度の対数的性質があいまって，最小検出エネルギの10^9倍の光にも応じ得るほどダイナミックレンジは広い。パターン認識と結びついた知覚は人間の感覚の最大の特徴であり，視覚における複雑な形状の解釈，聴覚における非常に大きなノイズの中から必要な音だけをききわける能力はその好例である。また不完全な情報であっても豊富な記憶と連想によってそれを補い，完全なものとして認識することができる。文字の脱落があっても，一部まちがっていても，言葉が一部聞こえなくても，図形が汚れたり一部消えていても，不完全部分を補って正しく認識できることが多い。

情報システムでは，定形化されたものを高速に入力しうるが，入力情報がある一定の範囲から逸脱すると入力不可能となってしまう。これに対し人間の感覚は高度の情報処理に支えられた解釈と選択を行ないながら入力しているから，入力情報の質の許容範囲は広い。条件が悪くなれば入力速度は低下するが全く入力不可能となることは少ない。また，各種の感覚が同時に相補的又は選択的に機能することも人間の感覚の特徴である。

(2) 情報の出力機能

情報システムにおける出力は入力よりもバラエティーに富むが，広く普及しているものはプリンターやディスプレイによる文字や図形である。音声出力の普及が期待されるが現状ではほとんど人間の視覚に頼るメディアで出力していると言えよう。一方，人間の情報出力のうち最も自然なものが口で言葉をしゃべることであり，次が手で文字や図・絵を書くことであり，手足でマシンを操作することである。また人間の場合，補助的な情報出力として身ぶり，手ぶり，顔の表情などがあり，形態の模擬や感情表現に使われる。入力の場合と同様に，出力チャンネルは複合して，相補的あるいは強調的に使用される。更に，出力

機能は入力機能と独立しておらず，互に協調して働き，適切なフィードバックがかけられる。人間の手は多くの自由度をもち，視覚・聴覚・触覚等の高度のフィードバックがかけられて功妙に協調制御され，三次元的な多様な運動を行なって情報の入出力にも関与する。

(3) 情報処理機能

数値計算その他のルーチン的な繰返しの情報処理の場合はコンピュータの方がはるかに正確で速い。しかし，特徴抽出，帰納的能力，パターン認識，学習能力や，高度の経験を豊富に記憶しそれを適切に連想する能力など，高度の認識，思考，判断等は人間の方がはるかに勝っている。

(4) 持続性

情報システムは連続した単調な繰返し作業にも耐え，その間の性能は一定である。これに対し人間は一定の緊張レベルを長時間維持することが困難であり，適切な休養やレクレーションを必要とする。大脳の生理的特性によれば，外界からの刺激がなければ自己の活動レベルを維持し得ないので，刺激の少ない単調な作業は不得手である。大脳の活動レベルを常時高めておく為には，適切な刺激を常に与えなければならず，また，その刺激も内容が重要であって人間の作業意欲を高めるものでなければならない。良好なマン・マシン・インタラクションを実現するためには，このあたりの人間の生理的・心理的特性についての研究も大切であろう。

(5) 信頼性

情報システムの場合，設計時に考慮された環境と事態においては人間よりはるかに信頼性は高い。しかし，予想外の事態に対しては多くの場合全く無力となる。また，今のところ誤りを修復する能力もない。

人間から誤りをなくすことはできないが，適正な作業条件のもとでは間違いも少くなる。信頼性の面で人間が情報システムにはるかに勝る点は，時間的，精神的余裕があれば，予想外の事態が起きても問題を解決して適切に対処できる場合が少なくないという点である。しかし，新しい事態に対応するための適

応にはある程度の時間が必要であり，また突発的な緊急時には正常な対応ができないこともある。

(6) その他

柔軟性・適応能力の点では，人間は教育・訓練によって様々に対応できるし，学習によって能力を発展させていくこともできる。人間は全ての機能が五感五体の1つのユニットとして有機的に結合されており，冗長性が高く万能性にすぐれている。

2.2 人間と情報システムの役割分担

マン・マシンシステムにおいて，人間の特性の方は今後も変らないと思われるが，情報システム側の能力は技術の進展に伴い着実に進歩していき，人間だけが可能と考えられていた機能が次第に情報システム側へ移されていくものと思われる。以下には，現在の技術をもとに，人間と情報システムの大まかな役割分担について考えてみる。

情報システム側に分担させるのが適当と考えられるものとしては，(イ)きまりきった処理や計算の反復，(ロ)大量の情報資料の蓄積，(ハ)大量のデータの迅速な処理，(ニ)図面や文書を正確に速く編集・管理・印刷すること，などがある。

一方，人間でなければ完全には分担できないと考えられるものは次の様なものがある。(イ)夾雑物にさまたげられた情報の判別。(ロ)刻々と変化するパターンを判別すること。(ハ)種々雑多な入力を総合的に判断すること。(ニ)帰納的推理力を必要とする問題の解決。(ホ)発生頻度の低い事態に対する判断。(ヘ)不測の事態の発生を探知しそれに適切に対処すること。(ト)その他やり方がよくわかっていないもの。

なお，機能の配分を考えるときには，単に情報システムで可能か否かということだけでなく，それを組込んだとき人間と情報システムの間の円滑なコミュニケーションを分断しないような配慮が必要であろう。

2.3 人間と情報システムの情報のチャンネル

人間が情報システムから情報を受けとる場合には，一般に視覚が最もよく使われ，聴覚がこれに次ぐ，触覚等の利用は特殊な用途に限られている。このほか人間には，味覚，嗅覚，平衡感覚などの情報の入力チャンネルがあるが，これらは情報システムでは当面利用されることはなさそうである。感覚器官の性質にはそれぞれ特徴があり，その特徴を生かした使い方が必要である。以下に主要な感覚である視覚及び聴覚に適合する情報の性質について述べる。

視覚による伝達が適する情報としては，(イ)情報内容が空間的定位や図形・写真のとき，(ロ)伝達される情報の内容が複雑なものや抽象的なもの，(ハ)いくつかの情報を組合わせて表示したいとき，(ニ)情報内容が長いとき，(ホ)特に急を要しないとき，(ヘ)あとで参照する必要があるとき，(ト)人間がほぼ定位置で作業できるとき，などがあげられる。

聴覚による伝達が適当な場合としては，(イ)注意を喚起する場合，(ロ)情報内容が簡潔なとき，(ハ)情報内容の伝達速度が重要なとき，(ニ)タイミング等特定の時点を扱うとき，(ホ)情報内容をあとで参照する必要がないとき，(ヘ)定まった位置で作業できないとき，(ト)視覚への負荷が過重であるとき，などがある。

人間に伝達する情報の速度は人間の受入れ体制に適した速度に制御されなければならない。もし，伝達される情報の量や速度が人間にとってオーバーロードになれば，次のような人間独得の適応作用が起る。(イ)省略（情報入力処理の一時停止），(ロ)エラー（情報入力処理の確実性の低下），(ハ)待合せ（情報錯綜に対し処理を遅らす），(ニ)フィルタリング（特定のカテゴリの情報だけを処理して他を無視する），(ホ)弁別の精密さをゆるめる。

次に人間から情報システムへ情報を与える場合を考える。情報のチャンネルとしては声と手の運動が主なものである。使い易さという観点からは音声が自然である。限定話者・限定単語の簡単な音声認識は実用段階に入ったと考えられるが，一般的な音声認識技術の完成にはまだ多くの問題が残っている。現在主とし

て使われているのは，手を使った入力であり，その代表が鍵盤である。習熟すれば入力の速度は音声よりはるかに早い。なお，人間と人間とのコミュニケーションでは，身ぶり手ぶり，声の抑揚，顔の表情や動きなども重要な情報の伝達手段である。うなずくとか，視線の動きなどは何とか情報システムへの入力情報として利用できそうに思われる。

2.4 入出力機器

人間と人間の情報の交換に比べ，現在の情報システムと人間の情報の交換では極めて制限が大きい。その最大の原因は情報システムの入出力機器の未発達にあると考えられる。情報のメディアとしては，光と音と運動に大別される。そして，ここで伝達される情報はテキスト，画像，文字，図形の形態をとっている。現在の情報システムと人間とのインタラクションを考えてみると，人間が情報システムに情報を伝達する手段は，つきつめれば手の動きに依存している。また，情報システムから人間への情報の伝達は，つきつめれば人間の視覚に訴えるものに限られている。音声による入出力や画像の入力なども盛んな研究の結果，その技術は進展し，限定された範囲内でのこれらのメディアの実用化も大いに期待される。

音に関する入出力装置としては音声合成装置，音声入力装置，シンセサイザなどがあり，出力装置の方は実用と普及の段階に入りつつある。これに対し，一般の自由会話の音声認識の為には自然言語理解の問題が解決されなければならず，その実現は当分望めない。しかし，話者を限定し，また認識対象語彙を比較的少数に制限した簡易音声入力は実用できる段階に達している。マン・マシン・インタラクションの観点からは簡易音声入力や音声合成だけでなくシンセサイザや音の認識もうまく利用することで使い易さの改善に役立つものと思われる。

イメージ及び図形の入力手段としては，テレビカメラ，タブレット，ファクシミリ，フライングスポットスキャナ等があり，出力の方は，グラフィックディスプレイ，コピーマシン，ビデオプロジェクターなどがあげられる。イメージの出

力に於ては，グラフィックディスプレイのカラー化，高分解能化が進み，あわせてコンピュータ・グラフィクスの技術も進歩してきれいな出力が得られるようになった。メモリーやプロセッサーの高性能化，低価格化が進展すると共に，グラフィクスは今後急速に普及が進むものと思われる。これに対し図面や図形の入力にはまだ解決すべき問題が残っている。

テキストの入力には主としてキーボードが使われている。漢字の入力にはまだ決定版はない。また，日本人はタイプライタに慣れていない人も多いので，使い易さを改善するうえで音声入力の実用化が望まれている。手書きの英数字やマークシートはOCR，OMRによって入力することが可能である。文字の出力の筆頭はCRTであり,最近は一画面に普通の書類と同等の字数を表示するようになりつつある。この他，プラズマディスプレイや液晶ディスプレイなどもその寸法上の利点から将来が期待される。

ＣＲＴ画面をみながら情報システムと対話する際には，画面上の位置の情報を伝える必要がある。この為の装置として，ジョイスティック，トラックボール，タブレット，マウス，ライトペンなどがある。これらの装置は他の入出力機器と組合せて用いることにより使い易さを向上させる。

以上のほか，バーコードリーダ，写植機などがあり，また，やゝ特殊なものとして，多様な運動を実行するマニピュレータ，人間の眼球の動きを入力するアイモーションなども情報システムの入出力機器のレパートリーに加えればおもしろい。

以上述べた様に入出力機器としては多様なものがあるが，それをどんなぐあいに組合せて使用すればよいかという点については，まだ実験例が少なく，結論はでていない。

2.5 入出力関連技術の動向

本節では，今後の開発や普及が期待されている入出力手段について，その動向

を調べる。

2.5.1. 高機能端末

キャラクタ・デイスプレイにマイクロコンピュータを組込んで，ターミナルとしての機能の向上を図ったいわゆるインテリジェント・ターミナルが増加してきた。使い易くするためにファンクションキーの機能が充実され，プログラマブルになった。キャラクタの点滅・反転・下線附加等はプログラマブルとなり，また，ＣＲＴ画面上に表示される文字数の増大と，画面をいくつかに分割して各領域別に制御可能となり，種々の使い分けができるように工夫されつつある。編集機能が附加されターミナルだけでテキストの編集が可能となった。同時に，画面の延長としてのバッファメモリとその延長としてのフロッピー・ディスク等が付加され，それらをプログラムで自動的に連結し，大容量の仮想画面を提供するものもある。メモリー及び編集機能の強化により，プログラム作成時のターミナルの使いやすさは格段に向上する。インテリジェント・ターミナルの機能が強化されると次第に独立のパーソナル・コンピータへ近づいていく。ゼロックス社のAltoやＭＩＴのLisp マシンはこれからの高機能端末のあり方を示す好例である。

Altoはゼロックス社パロ・アルト研究センターで開発された16ビット64K語（ 256 K語まで拡張可 ）のパーソナル・コンピュータである。まず外見からみると，我々が普通に使う紙と同じように，ディスプレイ画面も縦型になっている。走査線 875 本のインターレース・ラスタ・スキャンのＣＲＴを用い，視野は 808 行，各行 608 点からなっており，テキストや図・絵を表示する。画面はいくつかのウィンドウに分割可能であり，各ウィンドウに異った種類のテキストや図を書き込む。Alto の入力には通常の鍵盤と若干のファンクションキーのほかマウスを用いる。マウスはトラックボールを逆さまにした様なもので，机の上でころがすと，ボールの動きが検出されて画面上のカーソルの位置が変り，画面上の位置情報の入力に便利である。鍵盤はメモリー内の 4 語が対応し，

各ビットは対応するキーのシフトの状態を示しているので，キーの解釈がフレキシブルになる。

Altoには2.5メガバイトのカートリッジ・ディスクが装備されている。Altoのファイルは論理的には固定長のページの列として構成されている。ファイルシステムの信頼性と保全性を高めるために，ファイルの構造に関する情報は各ページに分散されている。ディスクコントローラはAltoプロセッサーの一部であり，ファイル処理の多くをこのコントローラで行なっている。Altoにはラスタ・スキャンのページプリンタも用意されている。Altoのディスプレイとほぼ同じ考えで設計されているが，分解能はそれより高く8.5×11インチの画面に各行3000点，4000行のプリントを行なえる。

Altoで使えるソフトウエアには，コンパイラー・ベースのものと，インタラクティブなものがある。前者としてBCPLとMesaがインプリメントされている。インタラクティブな方式のものとしてSmall talkがあり，テキスト，グラフィクス，アニメーション，音楽を含むドキュメントを扱うのに便利である。Interlispも十分な速さで実行できるが，Altoの主メモリーが小さいので，能力が自ら制限されるのは仕方がない。なお，パロ・アルト研究センターでは，100台以上のAltoがEthernetで接続され，使い易い分散システムを構成しているようである。

もうひとつのパーソナル・コンピュータの例としてはMITのLispマシンがあげられる。このマシンも縦形の高分解能ディスプレイを用い，Altoと同様に画面をいくつかに区切って種々の用途に使い編集等を行なう。画面の制御にはやはりマウスを用いる。使用できる言語はLispだけである。カートリッジディスクにサポートされた3M語（32ビット）のユーザ領域をもつ本格的なLispシステムである。

Altoにしても，Lispマシンにしても Personal Computingと分散処理を前提とした使いやすいシステムを構成しており，これからの高機能端末の行くえを示している様に思われる。

2.5.2 日本語の入出力

日本語を計算機で処理する要求が高まって来た。開発する時には目的を明確にして研究開発する必要がある。例えば，単にアルファベットやカタカナでは読みにくい，或いは日本人にはなじまないというだけで各種の計算機メッセージを漢字かなまじり文で出力することは漢字ディスプレイ，漢字プリンタの置換えのみでコード変換すれば達成されることである。又，更新の少ない文献，書籍等の文書などは写真用フィルムに写しておけばすむことであり，索引，コピーが出来るための検索システムには計算機を利用するとしても日本語を処理することから遠く離れている。日本語（漢字かなまじり）を計算機で処理する必要があるものは文書であること，頻繁にアップデートする，少量のプリント物であることあるいは速やかに印刷物が必要なものであると考えられる。以上の点を考慮して文章を作る人，原稿があって単に入力する，校正用など目的に応じてどのような漢字入力装置が適しているか検討した。

漢字入力装置としてはタブレット式，２ストロークタッチ式，カナ漢字変換，音声入力式が重点的に開発されている。タブレット式は素人や少量物，校正用に使用され，２ストロークタッチ式は高速大量のために使用される。問題点としては，タブレットの場合，タブレットの大きさにより漢字の字種と配列にある。タッチ式ではどのキーとどのキーの組合せ順序でどの漢字を当てるか，字種をどの程度にするか解決する必要がある。タッチ式の場合，高速性に特徴があるとすれば，字種を制限せざるを得ず統計的にみて1000字前後の字種を選らべばよい。

話し言葉と書き言葉は一般的に異なるが，入力するにはカナ，印字されたものは漢字かなまじり文が読み易いことからカナ漢字変換方式が研究されている。将来音声認識が可能になればこれら２つが結合されて音声タイプライタへ発展の期待もある。問題点としては，ＪＩＳかなキーボードの配列をもっと人間工学的考慮がなされる必要がある。少なくともカナの部分は横に広くなったとしても三列にした方がよい。

漢字の入力方式としては，サイト法，タッチ法，パターン認識による方法がある。

(1) サイト法

漢字使用文字によって異なるが通常2000～4000文字をすべて盤面に配列されている。盤面上の所望の文字を捜す必要があるので文字配列に工夫が必要である。現在，音読み，訓読みの五十音順と部首順の二大別されるが各メーカで少しずつ配列に差がある。これを統一するためＪＩＳ化の動きもでている。

(a) 和文タイプ式：ドラム式，小型邦文タイプ式どちらも機構は同じである。ユニバーサルキーを移動して活字を選択し打鍵するときのユニバーサルキーの位置がコード化され漢字入力としている。この特徴はモニタ印字が容易に出来るところにある。

(b) タブレット式：ペンタッチするとその点のＸ・Ｙ座標が出力できるディジタイザを利用する方式で漢字が配列されている中から選択してペンタッチしていく方式ですべて電子機器で構成されているための保守など楽であり素人や初心者に向いている。

(c) マルチシフト式：盤面に200前後の文字用キーを配置し，その1個のキーには，12～16文字が割当てられている。右手でグループを選び左手でそのグループの1文字を選択するシフトキーがある。このシフトキーと文字用キーを打鍵することで漢字コードにされる。

(2) タッチ法

総数５０前後のキー（英文タイプ並）で漢字入力する方法である。

(a) マルチストローク式：２ストロークで約2000の字種の漢字を入力するラインプットから４ストロークの打鍵で約8000字入力できるSC8000など最近の電子技術の発展に共ない興味あるものがある。いずれも，打鍵のキー及び順序は記憶する必要がある。左右の手の交互打ちと各指の特性とキーの割り当てを考慮してあり，熟練すれば100字／分の速度が出る。

(b) 数字コード式：漢字1字を5けたの数字で表現し打鍵する方式である。熟練しコードを記憶しても全漢字のコードは記憶できず辞書（漢字数字対応表）を引く必要があるので実用化は困難である。

(c) ソクタイプ：速記用の鍵盤装置で独持の鍵盤配列をもち両手で必要なキーを同時に打鍵する。入力速度はプロで180～230字／分である。

(d) カナ漢字変換による方法：発音文字をカナ鍵盤によって入力し，計算機処理されて漢字かなまじり文とする方式である。これはＣＰＵ内の辞書によってカナを漢字に変換するが同音異字があるので選択する機能が必要である。

(e) 音訓・部首・字形による方法：カナキーボードで音読みを入力し，同音異字のとき続いて訓読みで入力し漢字を当てる方式，漢字の部首をタイプライターのキーに割り当てておき，〝へん〟〝つくり〟を打鍵するとＣＲＴディスプレイにそのグループが表示されるからその中から選択する。部首にこだわらず字形の組合せで表現する。

(3) パターン認識を用いる方法

(a) 手書きオンライン：直接漢字かなまじり文を手書き入力しオンライン処理されコード化される。認識系に大きな負担がかかることと漢字を手書きする速度が速くないので得策でないがカナだけで手書き入力すれば認識系，速度，記録ものこるなど将来性がある。

(b) 印刷漢字入力：漢字ＯＣＲの期待は商品管理，図書館，出版，文献検索など幅広い用途がある。但し，漢字認識の困難性はアルファニュメリックのそれより格段の差がある。今後は漢字ＯＣＲフォントの開発が重要である。

(c) 音声入力：音声タイプライタに相当するものは将来必ず開発され有用な計算機入力機器となる。認識系の研究開発が多方面で行なわれているのでカナコードに変換されれば先きにも述べたカナ漢字変換の研究と共に最も期待したい入力方式である。

(d) バーコード式：和文タイプライタの活字の部分に文字の他に長短のバーで"0"と"1"を示すなどバーコード（各種ある）が一体となって鋳込まれている活字を和文タイプに使用する。印字されたものをバーコードリーダで電気信号に変換し，さらに漢字コードに変換される。

2.5.3 音による入出力

マンマシンインタフェースとして各種の工夫がなされているが，その性質として第一に人間にとって使い易いものでなくてはならない。ところが人間の持つインタフェースの内の重要な器官である耳と口の二つを用いて，音及び音声をマンマシンインタフェースとして扱っているものは少ない。警告などの場合に端末でベルやブザーなどを鳴らすことは可能で，計算機の内部コードとしても用意されているが，これさえも限られた場合にしか使われていない。この様な音，特に音声をそのまま機械とのインタフェースとして用いることができれば，使い易いシステムになるということができる。また音声によるインタフェースは，計算機の操作に慣れていない人々にとっても極めて有効である。

以下，音（特に音声）をどのような形で計算機システムのインタフェースとして用いるか，また可能性はどうかなどについて報告する。

現在音声を計算機とのインタフェースとして用いる為の条件が整いつつある。それは以下の理由による。

(イ) 音声信号のディジタル処理分野での理論的研究が進んだこと。

これにより音声をいろいろな形で計算機の内部のディジタル値として扱うことが可能になった。

(ロ) ＬＳＩの発展により，計算機の高性能化，低価格化が進んだこと。

現在音声処理用の集積回路もいくつか開発されており，また高級な処理を行なう研究もマルチプロセッサなどの専用計算機を指向しているものが多いが，それが価格，性能共に現実的になってきている。

(ハ) 社会的な機運

音声をインタフェースとして用いた電子機器の商用化が始まっており，さらに高級な機能に対する要求が高まってきた。

音声を計算機システムが取扱うのにはいくつかのレベルが存在し，大きく分類して次の三つになる。

・音声応答システム

・話者認識システム

・音声認識システム

以下それぞれについて説明する。

(1) 音声応答システム

これは最も実用化の進んでいる方法で，音声が用いられるのは，システムから人間への方向に限られている。一般に音声情報は大量で，その蓄積方法によってさらに次の三つに分類することができる。

(イ) 人の会話の音声信号をそのまま蓄積する方法：これはディジタル録音とも言うべきで，音声波形をそのまま数値化して記憶するため，大量のメモリが必要になる。

(ロ) パラメータ合成法：上記の方法の欠点を補うため，音声信号そのものではなくて，解析することによってパラメータ化して記憶する。これにはフォルマントを用いるものや，線形予測の手法を用いるものなどがある。

(ハ) テキスト合成：音声を言語の文字列（テキスト）として蓄積する。合成する時は，発音記号の書いてある辞書を用い，さらに不自然さを除くため，文章中の役割などに応じて変化させるなどの処理を通常行なっている。

(イ)から(ロ)，(ハ)と進むにつれて，記憶量は減少し，計算量は増大している。

(2) 話者認識システム

音声を入力として用いるものの一つとして古くから研究されているのに話者認識がある。計算機による話者の認識はシステムに対する資格検査に用いることができる。これには検証と同定の二種類があり，検証は入力音声と，

対応する登録音声とが一定値以上の類似度があるかどうかを調べるもので，同定は多数の登録者すべてと比較し，最も類似しているものを求める手続きである。

現在までに作成されているシステムでは，認識率自体にはまだ問題があるようだが，人間による認識よりは正答率が高いという報告もある。

(3) 音声認識システム

音声認識は最も困難なものであり，単に音声を取り扱うだけではなく，場合によっては知識ベースを用いた人工知能的処理が必要になってくる。従って現在実用化されているのは，語い数，話し手の数，話し方などにかなりの制限が加えられているものが多い。

例えば，数字や新幹線の駅名を音声認識させるなど，会話の形式が固定で，内容も限られた範囲のものになっている。いわゆる音声タイプライタなどは，今後の課題であろう。

(4) 計算機システムへの導入

この様な音声システムを計算機システムに導入していくにはいくつかの段階が考えられる。計算機の出力あるいはオペレータへの指示などで重要なものは音声出力もあわせて用いれば見落しなどの事故も避けられ，効果は大きい。現在音声出力装置はＬＳＩの発展のおかげで小型化し，英語教育機器などの商品として実用化されており，計算機システムに導入することには問題ないと思われる。

話者認識を計算機システムのパスワードの代用として用いることは可能で，バンキングシステムなどでも大きく期待されているが，認識率などの点で問題があり，まだ実用化には到っていない。

計算機との音声による会話は，人間と人間のインタフェースをマンマシンインタフェースとして用いるために，最終的には不可欠であるが，現在の所人間の側が機械にわかり易いように，かなり努力する必要であり，かえって使いにくいシステムになる恐れがある。これは人工知能的処理の進歩を

待たなければならない。

また，これまでのタイプライタなどの入力をすべて音声に置き換えるのが果して使い易いかどうかも慎重に考慮しなければならないが，多元的入出力システムの一部として補助的に用いることの効果は大きいと考えられる。

2.5.4 画像の入出力と処理

人間が外界から受け取る情報の中でも，視覚を通じて受け取る画像（図形）の情報量が最も多い。画像処理に対する工学の貢献は，主として撮像・表示・記録の面に限られているのが現状である。特定の分野においては，画像の認識や理解について機械化や自動化がある程度進んでいるが，人間の持つ優れたパターン認識の能力に匹敵するほどには到っていない。

画像処理の目標を列挙すると，画質改善によって見易くきれいな画像を得ること，画像の持つ特徴や構造・性質を抽出し，人間が認識や判断を行うための援助となること，さらには，人間の代わりに処理を自動化すること等であろう。現時点では完全な自動化は望めず，むしろ人間が判断するための補助となることが主要目的である。したがって，対話的な画像処理端末の発展が期待されるわけで，以下に最近の動向と問題点についてまとめる。

画像処理の方式には，光学・写真・ビデオ技術などに基づくアナログ方式と，計算機によるディジタル方式がある。後者は前者に比べ融通性・精度・再現性などの点で有利であるが，画像の情報量の膨大さによって，処理時間や記憶容量が大きくなり，また専用の入出力装置が必要となる。しかし，最近のＩＣ・ＬＳＩ技術の進歩によって，記憶や演算装置の価格は低下してきており，ディジタル画像処理の普及は目ざましい。もう一つの利点は汎用性であり，ハードウェアもソフトウェアも種々の応用に対し共通に利用可能である。

画像処理の種類には，画質改善，画像伝送，画像強調，復元，画像の前処理と認識，等がある。各種類毎に，具体的な例をまとめると表2.1のようになる。

表 2.1　画像処理の種類と具体例

①画質改善…濃度歪・幾何歪の補正，雑音除去
②画像伝送…帯域圧縮・符号化
③画像強調・復元…フーリエ変換，アダマール変換，K－L変換
④画像前処理・特徴抽出・認識…空間フィルタリング（重みづけによる畳み込み），論理フィルタリング，回転，拡大，縮小，微分，細線化，パターン照合，しきい値処理，統計的分類
⑤画像生成…投影からの再生（CT），3次元表示

これらの画像処理は本質的に並列性や反復を多数含むので，並列処理やパイプライン処理が非常に有効となり，多くの画像処理専用装置はこれらの方式を採用している。また，各画素の濃度を表わすビット数もそれほど精度を必要としない場合が多い。したがって，アレイプロセッサや，固定小数点の四則演算を高速に実行するパイプラインプロセッサが画像処理専用装置として必要である。

画像情報処理システムの中で，画像入出力装置の果たす役割も大きい。特に，対話型の画像処理システムでは，入出力装置の速度や機能等が使い易さの要因となる。画像入力は，画像データが可視光や不可視光（赤外や紫外等），X線・超音波等種々の範囲に渡るので，センサも異なる。しかし，一般的な画像入力装置を考えると，オフライン入力とオンライン入力に大別される。前者は，一度写真に撮影してから入力するもので，FSS，ドラムスキャナ，OCR等がある。FSSもドラムスキャナも，現在25μm程度の解像度が得られるが，入力時間が前者は数秒，後者は30分程度を要する。OCRは，印刷英数字や手書きの英数カナ文字の読み取り等が実用化されている。オンライン入力では，TVカメラが圧倒的に多く，1画面の入力が30 ms で済み，カラー画像の入力も容易である。光導電型の撮像管以外に，CCDやフォトダイオードなどの固体撮像デバイスが進歩してきているので，小型低価格の画像入力装置の実用化が期待できそうである。

画像の出力・表示装置は，ソフトコピーとハードコピーの2つのタイプに分

類できる。ソフトコピーで一般的なのは，高解像のカラーＣＲＴディスプレイであり，解像度も大きなものが得られやすい。画像用メモリの価格も低下してきており，表示装置として普及するであろう。ハードコピー用の装置で，白黒用はかなり普及しているが，カラー用装置の開発は模索中の段階で，レーザ・プリンタ，フィルム・プリンタ，インク・ジェット・プリンタ等がある。前の２つはネガフィルムの作成用であるが，インク・ジェット・プリンタ等では，吸収の早くて横ににじまない記録紙の開発が重要課題である。マイクロフィルム出力（ＣＯＭ）は画像の記録保存のために，ファクシミリは画像の伝送のために利用される。しかし，カラーのゼロックスのように多色刷りのハードコピーは，あまり需要がないのでその応用を検討する必要がある。白黒や多色刷りのハードコピーの例としては，Ｘ－Ｙプロッタやドットプリンタ等がある。

画像の処理や入出力の他に重要な装置として，画像データ保存のための記憶装置がある。膨大な画像情報を蓄えるために，低価格大容量かつ検索速度の早いメモリが必要である。マイクロフィッシュやＶＴＲの活用，及びそのディジタル化とともに，光ディスクやホログラムメモリの発展が期待される。画像データベースの利用法やフォーマットの標準化，データの圧縮法なども今後の課題である。

画像処理の応用分野は，急激に広まりつつある。具体的な例として，環境資源におけるリモート・センシング，医療におけるＸ線・細胞診・ＣＴ・モアレ像・超音波断層像，生産における探傷検査・組立て自動化，科学における結晶・スペクトル解析，警察における指紋・顔写真解析，マスコミにおけるＴＶ画面・新聞紙面の編集等があり，多岐に渡っている。各分野によって，必要な入出力装置（特にセンサの種類），リアルタイム性，処理の種類が異なる。大切なことは，ディジタル画像処理の汎用性を生かすことであり，将来も画像処理の応用分野を普及させるためには，ハードウェアの高速化と低価格化，ソフトウェアの蓄積と高水準化が必要であろう。

まとめとして，対話的な画像情報処理システムでは，高解像度の表示装置，

並列処理による高速演算装置，アクセスタイムが早くて大容量の画像ファイル，便利で高機能な対話型コンソール，強力なシステム管理プログラムが必要である。専用画像処理ハードウェアの開発に伴い，そのソフトウェアや言語など検討すべき課題は多い。入力，処理，記録，出力の各構成要素はそれぞれ独立した高度の処理機能を有し，機能分散された形態になるであろう。各画像情報処理システム間の有機的結合も重要な課題であり，光ファイバ・ケーブルによる高速の画像転送などが有用である。

2.5.5 グラフィクス

コンピュータ・グラフィクスで代表される視覚情報は，人間にとって受け入れやすい情報なので，マン・マシン・インタラクションの有力な手段として，ハードウェア，ソフトウェア，アプリケーションの面で研究開発の努力が重ねられてきた。

表示管については，ベクトル形ディスプレイ，ストレージ形ディスプレイ，ラスタスキャンディスプレイ，プラズマディスプレイ等々種々のタイプがあるが，一般用としては，半導体メモリのコスト低下と共にラスタスキャンのリフレッシュ形のものの普及が進んでおり，その分解能も1000 × 1000程度のものへと移行しつつある。

曲面体に自然な陰影を表示することなど，図や絵をきれいに自然に表示するためのソフトウェアの開発も進歩した。また，最近の特徴は，カラー表示の普及であり，それに動きも加わって，意匠設計やアニメーションなど大変美しいみやすいものになっている。

一方，ＣＡＤ／ＣＡＭのように，グラフィクスを使ってインタラクティブに機械部品等を設計する場合には，単にコンピュータで作成したモデルを表示するだけでは不十分であり，人間とコンピュータが対象物に関する同等の幾何モデルを取り扱える様にしなければ，使いやすいシステムとはならない。この様な観点から，特にＣＡＤ／ＣＡＭの分野では幾何モデル構成法が重要な課題と

してとりあげられてきている。

グラフィクスの応用の場は広汎である。例えば，上に述べたＣＡＤ／ＣＡＭ，アート，アニメーション，地図作製，管制センター，ＣＡＩ，ＭＩＳ，写真の修正等があげられる。これらいずれの応用においても，使い易さという面から考えると表示技術よりも入力方法が遅れており，多くの問題が残されている。

グラフィクスには人間の視覚が直接向き合う。表示された内容は人間の視覚の生理的・心理的特性に従って認識されるから，ディスプレイ本体及びその設置環境についても人間工学的に検討する必要がある。ディスプレイ本体については，操作性（鍵盤の配置，押下特性，つまみ等の形・大きさ・配置）および視認性（表示の大きさ，提示時間，解像度，コントラスト，表示色）等が検討課題である。

一般に画像表示は音声による情報提供に比べて直観的把握が容易，一覧性が高い，情報選択の自由度が大きい，などの特徴を有している。画像表示の有効性はその表示内容に大きく依存するが，画像表示に対する利用者の心理面に及ぼす影響を明らかにすることが必要である。一般には，画像表示は情報内容を理解するうえで非常に役立つが，多くの場合，音声を伴うことにより一層その有効性が発揮される。表 2.2は，画像通信方式に関する評価の例であるが，情報システムのインタラクションの場合にも同様のことが言えると考えられる。

2.5.6　手書きの文字と図形

活字印刷された英数字のＯＣＲは，印字機構のガタのため隣接文字同志が接触した状態で印字された場合や，ラインプリンタのゴーストなどの雑音が加わった場合に問題点をかかえているが，きれいな活字のＯＣＲはほぼ実用段階に達し，端末ＯＣＲやハンドＯＣＲなども実用化されている。また，印刷漢字ＯＣＲの研究も進み，認識性能は人間が邦文タイプライタを打鍵する場合よりずっと高く，速度は 100 倍以上早い。なお，漢字とバーコードを同時に印刷しておき，バーコードを読取る方法もある。

表 2.2　画像通信方式の評価

通信内容	比較方式	主な心理要因（寄与率，%）	方式の優劣
1対1の自由会話	①面談 ②音声 ③音声＋動画	審美感（21.7）	①≃③＞②
		価値感（19.0）	①＞③＞②
		プライバシー（5.3）	②＞①＞③
	①音声，②音声＋静止画，③音声＋1秒コマ落し，④音声＋4秒コマ落し，⑤音声＋動画	心の平静さ（43.1）	①＞④≃⑤＞②≃③
		刺激感（24.1）	③≃④≃⑤＞②＞①
		親しみ，快さ（9.9）	①＞⑤≃④＞②≃③
1対1の書画を用いた説明，打合せ	①音声，②音声＋静止画，③音声＋2秒コマ落し，④音声＋動画	効率性（22.7）	④＞②≃③＞①
		近接感（17.9）	④＞②≃③＞①
		親近感（16.2）	④＞①＞②≃③
5人対5人の会議	①面談 ②音声 ③音声＋動画	心理的連帯性（33.0）	①＞③＞②
		刺激感（6.9）	①＞②≃③
		興味性（6.1）	②≃③＞①
センタツーエンドの情報案内	①音声 ②静止画のみ ③音声＋静止画	刺激感（15.8）	③＞②≃①
		興味性（15.5）	③＞②≃①
		分かり良さ（11.1）	③＞①＞②

手書き文書を機械で読みとる為には，書き方に適切な制限を加える必要がある。文字は離して書かなければならない。標準字体を用いて書く文字を常用手書き文字，書き方の制限を強めて特定のルールに従って書く文字を制限手書き文字と呼んでいる。英数字については，常用及び制限手書き文字ともＯＣＲが実用化されている。漢字の手書き文字ＯＣＲは今後の課題である。

手書きの図形の入力は書き上げられたものをオフラインでファクシミリやテレビカメラを用いて入力し認識する場合と，タブレット等を用いてオンライン

で入力する方法が研究されている。グラフィクスにおける幾何モデル作成のための入力方法として試みられた研究では後者の方式で手書きの文字と併用して図面の入力を行なっている。このような方法が確立されれば，マン・マシン・インタラクションの改善に効果があることは疑いがない。

3．ソフトウェアの使い勝手

3. ソフトウェアの使い勝手

ソフトウェアの使い易さについて一般論をのべる前に，使い易いといわれたことのあるソフトウェアについて，どこが使い易かったかの原因を調べてみることにする。

(1) ＰＣ－１の初期入力ルーチンＲ０，Ｒ１

ＰＣ－１は1958年３月に東大理学部高橋研究室で完成したパラメトロンの計算機である。この計算機そのものについては，高橋秀俊編パラメトロン計算機，岩波書店にくわしい。このＰＣ－１は一応ケンブリッヂのＥＤＳＡＣを手本にして作ったから，計算機が完成するとすぐＥＤＳＡＣのイニシャルオーダーのようなプログラムを作ることになった。この辺の顛末については，例えば高橋秀俊，コンピューターへの道，文芸春秋社にあるが，ＰＣ－１ではＥＤＳＡＣのような５単位のテレタイプではなくて，６単位のものを使ったので，いろいろ工夫のし甲斐があった。ＥＤＳＡＣの方では大文字の一部と数字が共通であり，機能鍵にもギリシャ文字をつけて入力コードにしていたが，これでは入出力のコードがちがってモニターすることもできなくなる。ＰＣ－１ではプログラムはグラフィックキャラクターのみとした。その際，大文字を使うか小文字を使うかまたは両方を使うかが問題になるが，ＰＣ－１では上下段シフトは，わずらわしいということで，数字と同じ段にある小文字だけを使うことにした。そうすると，小文字と同じ下段の特殊記号だけを使うという結果になるが，ＰＣ－１ではプログラムの書き方に工夫をして，下段だけでプログラムを表わすことにしたが，非常に使い易いものになった。ＥＤＳＡＣのイニシャルオーダーよりはるかに便利なものができたと思っている。

10進２進変換で面倒なのは小数部分のそれで，システムをさぼれば一定の桁数だけ入力させるという使い難い入力ルーチンができるけれども，初期入力ルーチンのＲ１は任意桁数で入力を中止してもよいようにできていた。これはしかし入力した桁

数を数えるプログラムを別にもっているわけではなく，10進２進の変換を行うと同時に桁数が数えられるという工夫をしたからで，プログラムを大きくすることなく，任意の桁数でとめられる便利な方式になっていた。近年の計算機は昔のＰＣ－１などに比べて記憶容量がふんだんにあるから，ＰＣ－１のＲ１で採用したようなプログラムはみられないようになった。

ＰＣ－１のころは今の計算機にくらべて計算機自身が弱体であったにも拘らず，意外と便利にできていたように思う。つまりシステムの設計者が，既存の不便なシステムに馴らされてしまったりせずに，独自に考察して便利なものを作ろうと考えたからである。ＰＣ－１もEDSACも，さらにはもうひとつの同時代の計算機イリノイ大学のＩＬＬＩＡＣも初期入力ルーチンを入れるには，殆んどワンタッチであったのだが，その１０年後くらいのミニコンでは１０数ステップの１ＰＬ（Initial Program Loader）を手で入れなくてはならないように退化してしまった。これはＩＰＬ暗黒時代ともいうべきもので，やがてマイコンによるルネッサンスが到来するとともに，ＩＰＬはＲＯＭに書き込まれるようになり，手動IPLという，操作に最悪の時代は過ぎ去った。

(2) 最初のモニター

計算機の操作システムは次々と大きくなって，いわゆるソフトウェアの危機を招き，悪評の元凶になっているが，1958 年頃にＩＢＭ704 に作られた最初のモニターは機能がきわめて限られていたために非常に便利と思われた。このモニターは一名Fortranモニターという名前で，704 の４Ｋ語の記憶装置の最後の 100 語（オクタルで144語といわれていた）に納められていた。この記憶容量は少いことも重要だが，その容量が覚え易いという点でも意味があった。

モニターに対する制御カードは一枚で，それがユーザーの番号，使用する磁気テープの番号の程度を示していた。このカードにはパンチの穴のまわりにミシン目がついていて，鉛筆の先でつつくと穴があき，モニターカードになるものもあった。モニターの行う仕事は，ジョブ開始，終了の時刻の出力と，課金の情報の

作成，つぎつぎのジョブの連続処理の程度であった。もちろん当時の計算機には記憶保護などないから，モニタープログラムもときどきは破壊されたが，この程度のモニターでは再ロードも簡単であった。

このモニタープログラムの使い易さの教訓は，壊されないような，また何でもやる気になればできるというような完全性をねらわず，9割以上のジョブが満足して走るような一般性のあるしかも簡単なプログラムを用意したことにあった。特に殆どがFortranによる標準的な数値計算ジョブであるセンターなら，いまでもこの種のモニターは充分実用になると考えられる。モニタープログラム，あるいは操作システムは，ユーザー集団がきわめてユニフォームなら，決して大変なものにならないということがブリンクハンセンのオペレーティングシステムの原理に書いてあるが，これは同時に使い易い操作システムを実現するための必要条件かもしれない。

(3) 7040のMAP

これはアセンブラの一種である。アセンブラとしては実用になったもののうちで，極めて高級な，複雑な機能をいろいろそなえたものであった。ちよっと考えてみると，沢山の機能をもっているものは，使い難くなるのが普通だが，このMAPに限っては非常によくできていたということができる。これは書くのに便利であり，またできあがったプログラムのリストを読むのに便利であった。特にIBM 7040の操作システムのIBSYS，IBJOB等はMAPで書かれていて，その核の部分は読んでいて実に楽しかった記憶がある。どこが使い易かったかといえば，いろいろ思いだせるけれども，次の諸点にあげるにとどめよう。

外部名が自動的に宣言できること，MAPでアセンブルされたデックは，外部名（EXTERNAL）と内部名（ENTRY）がリンク情報としてBCDのまゝ残され，リンケージエディタによって実行用のモジュールに作られる。そこでアセンブリの際，外部名にはいちいちEXTERNAL をつけた宣言をしなくてはならないのだが，モニターの持っているサブルーチンや変数の場所にはSYSまたは

ＳＳで始まる名前がついており，この規則の名前を使うとそれは自動的に外部名になって，ＭＡＰで書いたデックから参照することができた。これは実際に使ってみると大層便利なもので，モニターのもっているサブルーチンがどんどん利用したくなるものであった。その後数年してHitac５０２０が登場した。５０２０でもシステムのサブルーチン等にはＳＹＳ…の構文の名前を使うけれども，これは名前の構文だけを借用しているものであって704のMAPのような便利な機能は全く用意されず，普通の英字名にすぎなかった。ということは，ＭＡＰのつもりでＳＹＳ…を使ってみたところ，まず未定義ということで文句をいわれ，次にそれならというのでEXTERNAL をつけてみたところ，そのようなENTRYはないといってリンカーから叱られ，結局どうしたかというと，モニタープログラムでＳＹＳ…が何番地にわりあてられているかを前もって調べ，ＥＱＵでＳＹＳ…をその番地に結びつけておかなければならなかった。ということは，モニタープログラムをアセンブリし直して，対応する番地がかわるとＭＡＰではバイナリーデックからのリンクのやり直しだけですむのに，５０２０の方ではＥＱＵのカードの挿しかえとアセンブリのやり直しをしなければならないという事情が待ちうけていた。

クロスレファレンスの行番号のこと，それまでのアセンブラではクロスレファレンスをとると，この記号は何番地で使われているという表示になっていたけれど，ＭＡＰからはその表示が二次元の行番号になった。つまり，アセンブリのリストをみると，ソースプログラムと，それに対応するオブジェクトプログラムと，オブジェクトの入るべき記憶場所が出力されているけれども，ＭＡＰではそのほかに全行にわたる通し番号がついていて，クロスレファレンスの参照はその通し番号で行なわれる。これはマクロ定義など，どこにも格納されない部分も参照しなければならないため，また必ずしも格納番地の順にプログラムが書いてあるとは限らないという事実による。一方，マクロ展開があるとソースプログラムの一行が複数行になるので，その場合には小数点以下の行番号をつけて表示した。マクロ展開部分はオプションによりリストをとることも押えることもできるので，

そのいずれでも行番号がずれないためである。マクロ展開は入れ子になりうるが，行番号づけは二次元までででとめられていた。

強力なマクロのこと，ここでＭＡＰのマクロのいろいろな機能を説明しだすと，それだけで膨大なものになる。変ったところだけ紹介すると，ｉ）サブフィールドの一部をおきかえるもの。

```
Location   Operation    Address
NAME4      MACRO       A,B,C,D,E,F,G
A          B'T'C'D     E
           SD'F        G
           END
```

というマクロ定義に対して

```
           NAME4       AX,W,D,J,,L, **
```

というマクロコールをすると

```
AX         WTDJ
           SDL         **
```

がえられ，

```
           NAME4       AY;R,B,, 5,N,K
```

というマクロコールをすると

```
AY         RTB         5
           SDN         K
```

がえられる。

ii）かっこつきの実パラメータは，中にカンマがあってもひとつのパラメータとみられること。

```
Location   Operation    Address
CALLIO     MACRO       IOCOM, T1, OP, LABEL, T2, …
           TSX         (TAPE), 4
           PZE         IOCOM, T1, OP
```

```
        ……
        END
```

というマクロ定義に対して

```
        CALLIO   CITIO,2,((RBEP)),CITLB,……
```

というマクロコールをすると

```
        TSX      (TAPE), 4
        PZE      CITIO, 2, (RBEP)
        ……
```

がえられる。

iii) 実パラメータのかっこもＩＲＰ（無限回くりかえし）の疑命令があると，パラメータの回数だけくりかえされること。

```
Location   Operation   Address
SUMSQ      MACRO      T, B
           STZ        T
           IRP        B
           LDQ        B
           FMP        B
           FAD        T
           STO        T
           IRP
           END
```

というマクロ定義に対して

```
           SUMSQ      A, (X, Y)
```

というマクロコールをすると

```
           STZ        A
           LDQ        X
           FMP        X
```

```
FAD     A
STO     A
LDQ     Y
FMP     Y
FAD     A
STO     A
```

がえられる。このほかにもいろいろ面白い機能はあったがあとは省略する。論理的にきれいなマクロには例えばＧＰＭのようにごく限られたマクロ機能を縦横にくみあわせて何でもできますというのがあるが，これはパズルを解いているようなもので，解くのに結構時間がかかり，あまり実用的とはいい難い。704のMAPのマクロくらいが，あのアセンブラには非常にバランスのとれた機能のものであったような気がする。

(4) 構文エディタ

紙テープベースで仕事をしていて，プログラムを修正するときは，テープをコピーで対処していた。ＰＣ－１ではそれに電々公社の局内中継機の送受信機を流用して，オフラインでコピーしていた。この送信機にはスタートストップ用のスイッチがついていたが，90度回転するつまみの方式であったので，紙テープを１字分送るのは至難のわざであった。このスイッチは要するにクラッチの電流オンオフのためのものであったから，スイッチと並列にモールス信号用の電鍵を設け，それをトントンと軽くたたくことで，１字送りを実施し，大変重宝であった。計算機を使ってテープをエディットするようになったのはFacom 270 のTMDが最初の経験であったが，これは行番号のエディタで，あまり使い易いものではなかった。その理由としては ⅰ）アセンブラの打ち出すリストの行番号とファイルの行番号の間にくい違いがあった。ⅱ）ファイルの挿入削除はその後方の行番号を即座に変えてしまう，などがあったし，ファイルがアセンブラ用のものかFortran用のものかで，どういうわけか指定の仕方が違ったということもあった

ように記憶する。その後すぐにいわゆるＴＳＳの時代になり，ETSSが使えるようになったが，これはTSSだからファイルは計算機側に保存されており，好むと好まざるによらずエディタを利用しなければならない破目となった。ところがETSSのエディタはＭＩＴのＣＴＳＳにあった構文エディタを多少改修したもので，従ってこのときから構文エディタの味をしめることができるようになった。

構文エディタが実用になったのは，やはりＴＳＳによる計算機の会話型利用が可能になったせいである。これに対してバッチ方式は，いわば計算機に下駄をあづける形であったので，なんでも正確に規定してからはじめなければならなかった。ところが会話型の構文エディタは，ファイル中のこのような文字列のところ，というような指定をするわけだが，もしかしてそのような文字列の場所が複数ヶ所あるかもしれず，それでは下駄をあづけるようなことはとてもこわくてできないが，会話型だとその前後を印字してみて確認をとることができるし，また修正したあとも，すぐに確認できるという利点があり，実用になったのである。

構文エディタにもいろいろなものが提案されてきたが，ＱＥＤとTeco が多少古くなったもののその双璧だったのではないかと思う。この両者についていえるのは，長い間使われ改訂を重ねられるたびにいろいろな機能が順々に追加されて，全体の機能のバランスが今では相当悪くなっていることで，新たに加わった利用者の目には，全体は恐ろしく複雑なコマンドの集積とうつるに違いない。だがソフトウェアの使い易さという面からすると，少くともi)はじめのコマンド体系は整理されていて使い易いものであった，ii)その時点からずっと使いつづけている利用者には，現在のアンバランスなシステムも，結構使い易いものに違いない，ということができよう。これは丁度大都会に昔から住んでいる人が順応しているようなものであろう。もしソフトウェアの使い易さを，その時点での利用者の分布を考えて，大勢の人が使い易いと感じるものとするならば，恐竜のようになったソフトウェアは使い難く，また新規に設計しなおすのがよい筈である。そうでなければ，なるべく新しい機能の追加を押える必要がある。

エディタの例でいうならば，ＱＥＤやTecoの時代が１０年程たった頃，画期

的なエディタともいうべきEmacsが登場したが，これは前のエディタがテレタイプ向けであったのに対して，ディスプレイ向けのものである。

Emacsはひとまずおくとして，QEDやTecoの形のエディタの使い易さの議論で最初に問題になるのは，挿入はポインタの前へか後へかということである。また削除についてもポインタのどちら側をとり除くかということで議論になる。このことに関しては恐らく次のように考えるのが一番わかり易く使い易いのではないかという気がする。まずポインタは対象の文字を次々と指すか，また文字と文字の間を指すかということ。これは文字が1字もない最初のファイルのことを考えると，当然文字と文字の間を指すと考えるのがよいようである。そうするとポインタの位置に直接新しい文字が挿入できる。挿入というのはファイルを構成している文字列全体の先頭や最後にも挿入しなければならないので，ポインタは先頭の文字の前から最後の文字の後までを動きうるとする，ポインタのところに新しい文字を挿入するのはよいのだが，挿入後のポインタの位置をその文字を指していると考えるわけにはいかないので，新しい文字のどちらの隣にいるかをきめなければならない。つまり今の文字はポインタのところに挿入したわけだが，ポインタの右へ挿入したか左へ挿入したかということで，これは次々と挿入した文字が左から右へ並ぶように考えれば，ポインタの左へ挿入するというのが正解と考えられる。次は削除で，これはポインタの右隣の文字を削除するか左隣の文字を削除するかということになるが，これはどうも両方必要のように思われる。ある区間を削除するのに最初にポインタがそのいずれの端にあるかでポインタのいずれの隣を削除するかがきまる。QEDやTeco型では大体右を削除する。つまりポインタはまず区間の左端へおくというのが標準のようであるが，Teco では左から右にある文字列を探索にゆくと，見つかった文字列の右端でポインタが停止するので，その場合は左を削除というのが便利である。

Emacsでは，一般のグラフィック文字はすべて挿入文字として扱われ，ポインタの左へ挿入ということになる。つまり最後に挿入された文字の右隣にポインタがいるわけで，さてタイプミスを消去しようとラブアウトを押すと，それが削除

のコマンドとなり，最後の文字から左へ消す。Emacsでは左を削除というのが標準となっている。しかし右も消したくなることもあるので，削除はポインタに対して右も左も消せるようにしておくのがよいと思っている。右へ挿入という機能はまず不要と思う。

ポインタは文字と文字の間をさすという場合，ポインタをどう表示するかが問題で，ポインタのあるべき場所を，適当なグラフィック文字で示すのが一番便利，しかしそれではポインタを動かすたびにその行内のポインタ後方の文字がぞろぞろ動くので，アンダーラインで示すシステムもあるが，これは文字の下にしか現れないから本当のポインタの位置とは多少ずれる。アンダーラインの文字の左にポインタがあるとするのが普通である。ＱＥＤのような行単位にポインタが動くエディタでは殆ど同様の理由により，行と行の間にポインタがあると思って構わない。この場合は削除はポインタの下である。しかし行内の修正ということがあるので，その行はポインタのどちらの隣と考えるのがよいかがやはり問題になる。これはどうもアンダーラインの文字に対応させて，ポインタの次の行とするのがよいようである。

以上，いくつか使い易いソフトウェアについて，実例をあげてみた。このほか評判のよかったものにはＩＢＭ３６０用の操作システムのＭＴＳ，ＰＤＰ－１１用のUnix がある。前者は評判のわりには知られていない。後者は小まわりがきき，利用者がやりたいと思うようなコマンドが揃っている。次々と作業をつづけるとき，パイプラインといって途中のファイルを指定しないで作業ができるといった特徴があり，大変好評でいろいろの規模のUnix が登場している。

どういうソフトウェアが使い易いかということを一概にいうことはむずかしいが，次のようなことは結論としていえるのではないだろうか。

1） 常識からあまりかけはなれていないこと。

2） 機能相互のバランスがよいこと。ある機能は単純，他の機能は超大型命令というのでないこと。

3） 環境とのバランスがよいこと。マイコンのシステムはマイコンらしく，大型

計算機用のそれはそれらしく。

4） 利用者の立場から設計されていること。これには手間がかからない，勘ちがいしにくい，覚え易く，しばらく使っていなくても思い出し易い，操作はそうむずかしくなく何でもできる。

5） マニュアルが正確，平易にかけていること。しかしマニュアルにあまり頼らず使えることが望ましい。

その他いろいろ考えられるが，一方使い易いソフトウェアを設計するにはどうすればよいかというと，これは難問である。これは別にソフトウェアに限らず，使い易さ一般について設計はむずかしいといわざるをえない。使い易いシステムの設計者は，非常によいセンスを持ち合わせているように思える。このセンスが先天的なものか，あとから学ぶことができるのかわからない。学ぶことができなければ，設計の工学などありえないが，学ぶことができるなら，それは使い易いシステム（と使い難いシステムと）を使ってみ，反省をくりかえすことによって得られるより他はないように思う。いろいろ使いくらべられるためには，ソフトウェアが移植され易くできていなければならず，これは優れているというソフトウェアを作った設計者はその移植のし易さも考慮にいれること，またそのソフトウェアを宣伝することが義務であろう。反対に利用者は誇大宣伝にまどわされず，真に使い易いものを探し求める努力をおこたってはならないと思われる。

4．システムの立場からの検討

4. システムの立場からの検討

4.1 使い易いとは

計算機システムが使い易いという場合，それはその人の使い方によってかなり異なるであろう。一般利用者，システムプログラマ，システムオペレータ等，それぞれの人々に取っての使い易さというものは従って区別して考えてみた方が良さそうである。勿論，使い易さ一般から言えば，その人その人に合ったインタフェースでシステムが使用できること，必要なものを，必要な時に，必要な形で得ることが出来ること*等を指すものと言えよう。ここでは利用者を以下のように分けて検討する。但し，一般利用者（カジュアルユーザ）と言うのは，計算機に関する知識を殆んど持ち合わさず，又それを期待できない利用者をいい，専用利用者というのは，計算機に関するかなりの知識を持ち，玄人はだしのマニアのような利用者を指す。

(1) 一般利用者

まず，利用者の場所，使用時刻によらず自由に安価にシステムが使用できること，端末の物理的（サイズ，カラー，文字の形etc），論理的（使用法）インタフェースがその人に合っていることがあげられる。使用法に関しては，余り予備的な知識を多く必要としないことがあげられ，使用言語が非常にハイレベルでシンタクスが単純なものが望ましい。例えばコマンド／応答形式の場合，システム側からは解り易い形式でコマンド／応答を利用者に要求したり，ヘルプ機能を充実して利用者の便をはかること（ヘルプ機能の概要を知る為には Help Helpと入力する）等が必要であろう。

またシステムの信頼性が十分高く障害を気にせずに利用できるとか，誤った操作に対してフェイルソフトにシステムを作る，システムの処理結果には十分な信頼が置けること等も必要であろう。機密保護がしっかりしており，情報の

＊ 富士ゼロックスのコマーシャル

洩れ，外からの混入がないことを十分管理することも必要であろり。要するに，安心してシステムが利用出来，又その結果も信頼して使えるということが重要である。

勿論，システムの応答性が十分高く利用者に待たせないことは必要であるし，そのシステムが利用者の目的に合わせて十分良くカストマイズされていることが必要であろう。

(2) 専門利用者

同じ利用者でも計算機システムを様々な形で利用することを好むマニア的利用者の場合は少し重点の置き所が異なる。端末の親和性，システムの信頼性に関しては同様であるが，コマンドや言語に望ましい性質は異なる。即ち，利用者好みのシンタクスに変え得ること，慣れればかなりの簡略化が可能であること，幾つかの機能をまとめたマクロの作成を十分サポートすることが必要である。

また，端末から様々なリソースが利用できることが必要で，そのリソースが存在する場所，システムに依らずオンライン利用できることがあげられる。例えば，各地のシステム個々が持っているデータ，特殊装置（高速配列演算，大容量オンラインファイル），応用プログラム（グラヒックパッケージ，統計解析パッケージetc）等を，利用者の位置に依らず利用できるようにすることである。

システムへの機能追加・変更が行なえるようになっていることも要求されるであろう。利用者の目的に合わせて利用者自身がシステムを変更したり，新機能を開発してシステム内に他利用者に公開する等の作業を十分サポートし，システムの発展に資することで，利用者側の意見を十分取り入れるという意味から重要な項目の一つである。

(3) 応用プログラマ

様々な応用プログラムを開発する人々に取って使い易いシステムとは，要するに様々な応用プログラムが作り易いことであって，次のような項目があげら

れる。

① システムの物理的特性からの独立

② 様々な制約が少ないこと

③ 能率の良い言語

④ 諸開発サポートシステムの充実

①は，応用プログラマに見えるシステムが，出来る限り論理的に見えて，各種デバイスの物理特性を意識する必要がないということで，デバイスが変わっても応用プログラムには影響がないことを意味する。②の制約というのは，通常計算機内で課せられる様々な制約，例えば，語長による計算精度の上限，メモリ容量によるプログラムサイズ，変数の個数等に対する制限（配列要素数等），ファイル容量，偏微分方程式を解く場合のメッシュ精度の上限（計算速度限界）等のことで，これらの制限を出来るだけ広げる努力が望まれる。

また，地理的，距離的な制約ということでは，他計算機内のファイル利用とか，特殊他計算機の利用ということがある。システム相互のハードウェア差，ＯＳの差異に依らずに互にリソースを共用することが出来れば，システム応用の範囲はかなり広がるであろうと思われる。この制約にはともすれば，データ伝送回線の速度・コストから来る時間的な制約になり易い。巨大なファイルを他所で利用しようと思ってもその転送コストの為実行不可能であり，代りにそのファイルが存在する所で動くプログラムを作ってそこで実行する等はその例である。応用プログラマには，必要なものを，必要な時に，必要な形で提供するあらゆる手だてを用意したい。

③，④については当然のことで，応用に合った能率の良い（プログラマが慣れている，実行能率が高い，作成手間が最少の等）プログラミング言語を用い，プログラム作成上の諸ユーティリティを準備し，便利なエディタ，マニュアルの文書作成システム，プログラム仕様の記述用言語・様式の統一，プログラムパッケージ管理システムの充実（版，新仕様，新旧対照機能，日時，作成者等）等様々な援助システム（名前だけではなく，実際に使えるもの）が必要であり，

これにはローカルなシステム内資源の管理情報システムばかりに止まらず，ネットワークを構成している場合は，ネットワーク内資源の情報システム（機能，利用可能システム名，利用法等の管理）が充実していることが不可欠である。又逆に，ある所で開発した応用プログラムを，網資源として共用する場合の系統立った登録機構，様式も定めねばならない。

(4) システムプログラマ

システムプログラマに取って使い易いシステムというのは，ここではシステムプログラマが，システムプログラムを容易に開発できるようなシステムという意味でとらえる。この場合，マイクロプラグラムの作成ということもこの人々に含めることとする。ここでは次のようなことが考えられよう。

① モジュール構造サブシステム

② モジュール型システム記述言語の利用

③ 小規模ＯＳの利用

④ ハードウェアの仮想化

⑤ 障害対策のハードウェアサポート

①は，ハードウェア全体が明確な幾つかのサブシステムによって構成され，各サブシステム間インタフェースが明確に規定されていることを言う。外からサブシステムを見た時，それは規定されたコマンド体系（言語）だけに着目すればよく，内部でそれが如何に実現されているかについては知る必要がない。このような構造によってシステムプログラマは各自，自分の着目するべき範囲が狭ばまり且つ明確になる他，デバッグ時のエラー範囲が限定されよう。ここでいうサブシステムの実現法には，ハードウェア自身の物理的な分離から，マイクロプログラムによる実現迄あるが，これについては次節で述べる。また①には，非同期プロセスのテストを容易に行なう為の諸サポートシステムの充実も含まれる。

②は，システムプログラムを記述する為の言語として，抽象データタイプのようなモジュール構造を持った言語を用いることを意味する。大規模なソフト

ウェアを能率良く分解し複数のプログラマによって作成してゆくことが容易でなくてはならない。一般にシステムプログラム記述用言語としては，ハードウェアの物理的特性を出来るだけ隠してなるだけ論理的にすっきりと記述したいという要求と，逆にハードウェアの特性をうまく利用して能率良いプラグラムを組む（主記憶アドレスを直接扱う等）という要求とを兼ね備えることが必要になるが，今後の方向としては前者の要求に重点が移り，後者についてはマイクロプログラムに頼る方向が考えられる。

従来，大型機メーカの提供するＯＳは複雑化，大規模化の一途をたどり，あらゆる機能がもり込まれて来ている。集中化は確かに能率の良い手法であるが，ある程度以上は避けるべきではなかろうか，特に，今後，パーソナルコンピュータが普及して行けば，むしろ各個人に合った小規模なＯＳがより必要となる可能性がある。それが③である。

④は，ハードウェアの物理的特性，制約をできるだけ無くしたアーキテクチャをシステムプログラマに見せるということで，主記憶に対する仮想メモリ，ファイルアクセス法に於けるＶＳＡＭ，通信応用におけるアクセス法ＶＴＡＭ，等様々なレベルでの仮想化をおこない，システムプログラミングに階層構造を持ち込むことがある。但しこれは，下位レベルのシステムプログラマに問題を移すことになるおそれがあるので注意を要するが，階層構造は問題を整理する上で非常に有益である。勿論，マイクロプログラム作成の容易化も重要で，マイクロコードの各フィールドを余り重複して使わない単純な構成が望ましいし，マイクロプログラム作成上の諸サポートシステムの充実が必要である。

⑤は，障害が発生した場合，その診断を自己で行なう機能や，通信回線経由で診断を受ける機能等，これらができるだけハードウェアによってサポートされていることが望ましい。障害の確率を減らすと同時に，障害時の検査機構が組込まれ，障害管理ソフトウェアの負担を減らし，システム再構成容易（システム部分切離し・接続変更）なアーキテクチャである必要があろう。

(5) システムオペレータ

システムを運転する場合，出来るだけ簡単な操作で済むようにするとともに，コンソールからシステム内のあらゆる状態が容易に把握できることがあげられる。それには，ボタン・スイッチ類をできるだけ少くして運転の自動化をはかり，一般利用者でも容易に操作可能にせねばならない。特に今後，分散処理プロセッサが普及して多くの一般利用者がシステムを操作することを考えると，これはかなり重要な項目にあげられるであろう。

操作コマンドを少くし，システムがオペレータをガイドする工夫，エラー番号ではなく，より解り易い表示，それに対する対策のメニュー等，多くの検討事項がある。

更に，磁気テープ，プリンタ紙交換，磁気ディスクの定期清掃，カードリーダの掃除等，これらの作業を省いてゆくことも考えられる。可動部分のないデバイスの開発，騒音・匂いの無いデバイスの利用等，今後の研究項目であろう。又，計算機システムの周辺装置間接続を単純化し，細い１本のケーブルによる接続にしてゆくこともシステムを運用・設置する上で大切であろう。

(6) システム管理者

まず，システム動作の統計収集・分析が自動化しており，それを用いてシステムのボトルネックの検出，老化部分の切り出し等が容易におこなえること。更に，それらのデータを用いて部分的なシステムの改善が，少しの増分コストによって容易に可能なシステムが望ましい。

増大するデータ，ファイルに対してはその系統立った管理をおこなうためのデータベース管理システムの充実は今後，増々必要になってゆくであろう。特に，コンピュータネットワークが大幅に建設されるような環境下では，ネットワークへ提供するリソースの一元的な管理が必要で，公開データ，サポート機能，アカウンティング等に考慮が必要である。

また，データの機密保護に関しては十分な配慮が必要で，ローカルシステムの管理者の設定したポリシーに基づいてそれをサポートするシステム構造が望

まれる。

4.2 今後の研究課題

前節でみた「使い易さ」の分析より，今後の研究課題となる項目を考察すると以下のようなことが考えられる。

(1) 言語の設計

これについては前章で述べられているので余り取り上げないが，あらゆるレベルの利用者それぞれに対して適合した言語／コマンド体系の設定，プログラム開発サポート道具の充実，パッケージプログラムの外部インタフェースの統一的記述方式の設定などがある。これには，ジョブ制御文の記述にPascalのような言語構造を持ち込むことの検討，モジュール構造言語の普及なども含まれよう。

(2) システムプログラム

今後のＯＳとしては，その機能を使用状況の自動モニタリングに基づいて自動的に最適化してゆく機能とか，必要な機能を自由に追加・変更することが容易であること，仮想マシンのようにＯＳ自身を幾つか独立に持つような構造（ＯＳの変更・テストが容易），パーソナルコンピュータ用の小規模専用化ＯＳの実現等が考えられる。小規模のＯＳに関しては，個人用であることから，ＴＳＳ機能が不用となり，システムリソース管理が容易，カスタムメイドで不要な部分が無い等の利点が得られ，小規模ではあるが十分その個人の要求を満たすことができる可能性がある。その能力以上の仕事に対しては，ネットワーキングによって対処するので，リモートアクセスの機能は不可欠であろう。更に，分散処理の発達によって各地に広くコンピュータが普及することを考える時，システムの自動運転又は専用オペレータの不要化が必要であり，中央システムからのダウンローディング，リモート電源オン／オフ等の機構が必要になろう。

(3) モジュール構造

ソフトウェアをモジュール構造で作るに止らず，ハードウェア自身もモジュール構成とすることが考えられる。これについては以前より少しずつモジュール化が行なわれつつある。例えば，CPUからチャネルとＩＯ制御装置を独立させ，更に最近では通信制御機能を通信制御プロセッサとして独立させ，従来ＣＰＵ内のソフトウェアで実行していた処理がＣＰＵからＣＣＰに移ったことがそうであるし，従来のファイル管理機能をデータベースマシンとして独立させる動きもそうである。更に，コンソール制御，モニタリング機能をＣＰＵから取除き，Service Proeessor として独立させる動きもそうである。明確な機能を分割し，きちんとしたインタフェース（コマンド，言語，回線などを意味する）を定めることは，システム全体の見透しをよくする他，保守が容易になる可能性を持っている。

現在，サブシステムとして明快なインタフェースを持っているものにファイルアクセス法のＶＳＡＭ，通信回線アクセス法のＶＴＡＭがある。今後，サブシステム化がファイルに対しておこなわれるのは確実であろう。既に開発が進んでいるData Language／Ｉ（ＩＢＭ）はその方向を示すものである。

また，ＯＳの機能の内かなり基本的なものをプリミティブとしてマイクロプログラム等によって実装し，その上により複雑な機能を組上げてゆくという手法もこの１つである。例えば，プロセス間通信機能，プロセスの同期機能，プロセスの生成・消去などがそのプリミティブとして考えられ，分散処理環境では特にそれが役立つとも言われている。

(4) 制約の除去

以前の計算機では主記憶容量の制約によって大きなプログラムが作れないということがあった。しかし，仮想メモリの導入により事実上この制約は無くなったと言える。とは言え，要求には限りがなく，16 MB 程度の仮想アドレス空間ではもはや不足になって来ている。主記憶自体が数１０MBになって来ている今日，今後の仮想アドレス空間は１GB～１００GB程度となろう。一方，ランダ

ムアクセスファイルの容量もかなり増大しており，磁気ディスクの設置台数が100台を越すシステムも稀ではない。設置床面積も無視できず，IBM 3850のようなMass Storage Systemが用いられる。これは200MBの磁気ディスク3330を仮想化したもので，472GBの最大容量を持つ磁気カードリッジシステムである。しかし，これとてどんな応用にも十分という訳ではなく，地震解析のような仕事（広大地域の）では既に不足だと言われている。

数値計算では，かなりの精度迄の値が必要になることがよくあるが，現代の計算機では浮動小数点数を32/64ビット程度で表現している関係上，精度がそれで抑えられることが多い。その為，数値を可変長ビットで表現し，演算を，マイクロプログラムによって必要な精度が上がる迄実行することも考えられている。これも，計算機をプログラマに取って使い易いものにする1つの手段である。

計算速度の限界からの制約に対しては，プロセッサを多数並列に並べた並列プロセッサ，パイプライン制御によるアレイプロセッサ等があり，商用になっているものも多いが，速度の上限は現在250 MFLOPS*程度であり，様々な応用を考える時，尚一層の発達が望まれる。この候補の1つにデータフロー計算機があるが，未だその潜在能力については全くの未知数であると言えよう。GaAsを用いた素子，ジョセフソン素子の利用等のゲートレベルでの検討とともに，プロセッサ台数が10^3～10^4程度になるような構造の研究にも期待したい。

(5) 仮想化

現在，仮想化の技術は様々なレベルで広く使われるようになっている。これはサブシステムに論理モデルを設定し，外から見たインタフェースをそれに合わせるもので，次のようなものがある。

① 仮想記憶，仮想ディスク，仮想磁気テープ

② 仮想マシン

③ VSAM，VTAM

---

* Million Floating Operations Per Second

④　仮想端末，仮想ファイル

①，③については既に述べた。②は，VM／370のように，物理的には1台の計算機を，何台かの計算機のように論理的に見せる技術をいい，各々の論理的計算機を仮想マシンと呼ぶ。又一般に，階層構造でシステムを構成する場合，各階層から下位層をながめた場合のインタフェースを仮想マシンと呼ぶことがある。前者は複数のＯＳをのせたり，複数のサブシステム（ソフトウェア）によって1つのシステムを作り上げる場合に有用な構造であり，後者はＯＳ自身の構造やネットワークアーキテクチャ等を論じる時，問題の整理に有用な概念である。

一方，④はネットワークアーキテクチャで出て来る概念で，システム毎に少しずつ異なった端末やファイルシステムを相互に共用する場合の標準システムを規定する。こういう概念はシステムの種類の数が増すと不可欠であり，分散処理では特に重要な概念である。プログラムの種類を減らし，網上のデータやコマンドを標準化する上で有用である。

今後のシステム作りにおいて，システムの発展性，統合化を推進する為に，仮想化の概念は多くの分野で使われるであろう。例えば，分散データベースに於ける仮想データベース，コンピュータネットワークにおける仮想網などが考えられる。

(6)　ＲＡＳＩＳ

システムの信頼性は，それが使い易い為の基本条件である。一般にＲＡＳＩＳと呼ばれるが，この関係する項目は次の通りである。

①　信頼性（Reliabilily）

②　可用性（Availability）

③　サービス性（Serviceability）

④　完全性（Integrity）

⑤　機密性（Security）

①に関しては冗長構成やＬＳＩ化が手法として用いられ，②に関しては自動再

試行や自己診断，遠隔診断がおこなわれ，③は分散処理によって危険分散（部分障害は全体障害にならない）をはかったり，フェイルソフト機能を充実することによって実現されてきた。今後のシステムでこれらが重要なことは言うまでもないが，更に④と⑤を強化する必要が出て来る。例えば，データベースに於ける内容の正しさを保証することとか細かくアクセス権の制御を行なうことがそれである。

機密性に関しては，ネットワークシステムが多用され，公衆データ網が利用されるようになる今後，単一システム（計算機）内の機密性をはかるだけに止らず，通信網自体の機密性も考慮する必要がある。通信プロセッサ，交換機，伝送路，それぞれに対して保護対策を実施し全体としての機密保護をおこなわねばならない。保護に対する外乱には，あるサービスに対し本人と偽ってそれを利用する詐欺，秘密の盗聴，伝送データを変更したり贋のデータを送る偽造，等がありこれらに対して保護が必要である。一般には暗号が用いられ，変換関数とそれに1対1で対応するキーでそれを実装する。キーを秘密裡に相手に届ける為の2重キー（公衆キーと個人キー）方式，逆関数を求めることが事実上不可能に近い変換関数の作り方，メッセージにサインを付け確認を取る認証法等の研究が為されているが，より現実的に実装する場合の問題も検討されなければならない。これには，プロセッサハードウェア上の考慮も必要になる可能性がある。

(7) 分散処理／ネットワーキング

分散処理の目的は，集中処理に比らべて各所のローカルな要求に適応し易い，応答性がよくなる，回線コストが下がる，危険分散ができる，パーソナルコンピュータ志向に合っている等であり，現在は多くの企業内網がその企業組織に合わせた形態で実現されている。未だ分散処理の程度は一般に余り高くなく，ローカルなサマリーを中央に送るという形が多いが，今後，分散処理プロセッサの普及とともに分散レベルが高くなり，各分散プロセッサでかなりの処理を分担する形になってゆくであろう。これには次項で述べるデータベース技術

の発達がかなり影響するであろうと思われる。課題としてはその他，仕事やデータを中央とローカルに分割する手法の確立，分散リソースの管理技術，障害対策，分散プロセッサ用応用プログラムの開発サポート，カジュアルユーザでも容易にコンピュータを扱える言語／コマンド体系，及び自動運転／保守技術等である。

上述の分散処理が，従来集中して行なって来た処理を分割し分散させて処理するという方式の色相が濃いのに対し，各所で自然発生的に設置されたシステムを通信回線等で結合し大規模システムとして統合してゆく方向がある。ここではそれをネットワーキングと呼んでいる。一般にこのようなシステムの目的は，各所のデータ，プログラム，特殊プロセッサ，特殊機器等の資源を共有すること，代替による信頼性を高めること，電子郵便など高級な情報通信システムとして利用すること，システムの発展形態として自然であること等があげられる。最後の項目は，元々離れていたシステムを結合し，オンラインサービスを提供するという意味で少し意味あいが異なるが，今後の網間接続，国際網等を考えるとき重要な項目である。

ネットワーキングに於ける課題としては次のようなことがある。

① 高品質・高速・安価なデータ通信路
② ネットワークアーキテクチャの標準化
③ 通信プロトコルの記述・作成・検証方式
④ 分散処理用言語
⑤ 分散リソース管理技術
⑥ 機密保護・暗号化
⑦ 分散データベース

①は，光ケーブル，通信衛星，パケット交換，等の技術を用いて，公衆データ網やローカル網を如何に作るかという問題で，②は今後の網相互結合に対処する為の方向で，CCITT SG VII, ISO TC 97／SC6, SC16等において現在活発に勧告化が進められている。7層の階層モデルに基づく勧告がこ

こ１～２年の内におこなわれ，更に詳細が固まってゆくものと思われる。③は，具体的にシステムを実装する場合の問題であり，④は網処理に向いたＯＳや応用を記述する為の言語で未だ研究は余り多くないが，網の利用を推進したり，網上ジョブ管理を行なう上の道具として望まれるものである。

⑤は，網内にある多くの資源を利用者に使ってもらう為の網内リソース情報システム（どこにどんな資源があるか），各資源を利用する場合のアクセス援助システム，複数資源を単一ジョブが同時利用する場合の資源割当・管理方式等である。リソース情報システムに関しては，そのリストをパンフレットにしたり，オンラインで見る等のサービスが必要であるし，それに載せる内容も単なる機能記述に止らず，利用法，制限などを含める必要があろう。この場合，インタフェース等の記述は出来る限り統一した言語で記述するのが望ましい。アクセス援助システムは，各システム毎に異なるログイン手続き，コマンドの差等に対処する為に，統一された利用者コマンドを各システム特有なものに変換するとか，利用者の要求した種類の資源を捜し出して，利用者からの要求によりそのシステムと接続をする等の援助が考えられる。前者の例としてはNBSのNetwork Access Machine,Rand社のRand Intelligent Terminal Agent, Harvard Univ.のNetwork Interface System等があり，後者としてはMITRE社のREX（Network Resource Manager）がある。前者のものについては相手を２～４システムに限れば可能になっているが，後者については全く単純なサービス（Help）しかサポートしていない。いずれにせよ，この種の研究は広くリソース共有網を利用する場合に重要であり，今後，より根本的な形での解決が必要であるにせよ，実用的な方式として更に研究が必要である。

資源割当て・管理方式については，ジョブの実行前にそれが必要とする資源が定まっている場合のデッドロックフリーな資源割当て法，実行時動的に資源要求・割当てをおこない，デッドロックが生じた場合にその処理を無効にする方式などがある。網上での資源管理は一般に単一システム内管理の延長ではあるが，回線によって結合していること，非常に多くのシステムが存在し得る

こと等によって実行上の問題が大きく，少ないオーバーヘッドでこれを行なうことは難かしい。デッドロック検出と後戻りが実用的であるが，デッドロックの判定には網内のループを検出するという操作が入り，それを能率良く実現する方式が検討されている。この問題についても，より実用的な観点からの研究が必要である。

⑥については前述した。⑦については次項で取扱う。

(8) データベース管理システム

分散処理の初期段階では中央にファイルを集中し，各ローカルシステムではデータを収集しまとめて中央へ送るという形が多く用いられる。これがもう少し進むと，中央のファイルシステムをより論理化しデータベース管理システム（DBMS）を構成してゆき，各所からそれを共用してアクセスが行なわれる形になってゆくことが考えられる。DBMS自身の研究は古く多くのシステムが商用になっているが，今後のデータ量の巨大化，各所からのアクセストラヒックの増加を考える時，汎用の計算機上にのせたDBMSでは対処しきれなくなる恐れがあり，データベースマシンとして専用システムを開発してゆこうとする研究が行なわれている。汎用コンピュータを用いてデータベース管理に専用させるシステム（ベル研XDMS，CCA社のDatacomputer,Cullinane社のBackend DBMS），ディスク制御装置を知能化して高レベルの命令で利用する知能ディスクコントローラ方式（東大のVSAMサブシステム，Tang等によるDASDプロセッサ），特殊ハードウェアや連想プロセッサを用いる方式（I.CL社のCAFS，STARANを用いたAPCS），磁気ディスク等の補助記憶のトラック対応にプロセッサを設置した論理セル方式（トロント大のRAP,フロリダ大のCASSM，ユタ大のRARES），より本格的に大記憶を取扱ったデータベースコンピュータ方式（オハイオ大のDBM，電総研のEDC,ウィスコンシン大のDIRECT）等がある。しかし，これはいずれもDBMS機能全体を取込み且つ大スループット，大容量を満たすようなものではない。DBMSの仕事の内，ファイルアクセス部分のみを工夫したものか，通常のマシンで実現しそれを既存

システムに接続することにより簡便にDBMS機能をサポートすることを狙ったものか，どちらかである。その意味から，本格的なDBMSマシンの研究は今後に残されていると言える。大記憶の取扱い，同時実行制御，障害回復，データ保護等に対する配慮も必要である。

一方，データベースが発達して各地に設置されるようになると，当然それらを総括してより大規模なデータベースに統合する動きが出て来るであろう。これを統合データベースと呼ぶことにする。単一のDBMSでも，それが管理するファイルが地理的に分散している場合はそれを分散データベースと呼んで区別すると，データベースの発達は次のようなステップに分けて考えることができよう。

① 集中ファイルシステム

② データベースシステム

③ 分散データベースシステム

④ 統合データベースシステム

現状は①か②であるが，③の形態としては信頼性を考慮して複数のコピーを持つ方式が検討されており，コピー相互の内容の一貫性を維持するためにタイムスタンプを用いる方法，論理時計を用いる方法が提案されている。また，③の場合，1つの問合せに対してそれに答える方法は複数（複数リレーションを用いて，どこでどのような演算を施こすか）考えられ，通信量の少ない経済的な処理法を求めることが問題となる。④に関しては，DBMSそれだれのデータモデルの差異，問合せ言語の差異，コードの差異などの問題がある。これに関しては，網モデル，階層モデル，関係モデル相互の変換を扱うシステムの実験（仏のPOLYPHEMEプロジェクト），Entity－Relationshipモデルを標準モデルとしたSIRIUS（仏）などがある。又，同一モデルを用いての研究では，IBMで複数のIMSを用いた実験（VM／370上），関係モデルを使ってのプロジェクトがあり，後者は現在進行中である。

しかし，③，④の問題は実に多くの問題を内含しており，実用的なものが出

てくるにはまだまだ多くの研究に待たねばならない。

(9) そ の 他

以上の他，保守の容易な端末，デバイスとして，消耗品を使わないもの，可動部のない装置の研究があげられよう。カード利用からオンライン端末とファイル利用に移って来たこと，プリンタ端末からＣＲＴ端末への移行等はその方向であるし，今後は磁気ディスクの電子ディスク（磁気バブル等による）への移行も徐々に行なわれるかも知れない。

更に，パターン認識技術の取込みはそれによってシステムのかなりの部分に大きな影響（入力部は勿論，プログラミング言語にも）を及ぼす可能性がある。９０年代はそれに期待したい。

5．アプリケーションの立場からの検討

5. アプリケーションの立場からの検討

5.1 概　　要

システムの最終ユーザの立場からみた使い易さの問題を考えてみる。全ての使い易さの追求は終局的には最終ユーザの便宜を考えたものであるから，前章までの議論の多くが深いかかわりをもってくる。ここでは，これまでに論じられなかった点の幾つかをトピックス的にとりあげて論じてみる。

情報システムをなじみ易いものにするためには，使うための予備学習をできるだけ少なくする必要があるという議論がある。一方，自動車の運転やタイプライタの使用にもある程度の学習が必要であり，情報システムについても予備学習を前提とする方が最終的には使い易いシステムとなるという議論がある。

日本文入力方式について学習を殆ど前提としないカナ一漢字変換方式と学習を前提とする二打式タッチメソッドとの優劣が論じられているが，前述の予備学習論議と本質的には類似の争点といえる。後者については，専門家用と素人用ということで当面両方式の併用が考えられているが，人間工学的な研究を含め今後研究を必要とする課題である。

情報システムに自然言語を使うことの利害も殆ど同じような観点から論じることができる。意志の疎通を正確かつ簡潔に行なうためにはその分野専用の人工言語を使う方が一般に優れているが，その言語を事前に学習する必要がある。自然言語を使うとすると，その表現力のあいまいさからいい直す必要が生じたり，間違って理解されることを覚悟しなくてはならない。しかし，稀に使う素人ユーザにとっては自然言語による表現が可能なことは大きな利点となろう。但し，システム側でユーザの言語が十分理解できない時に〃聞き直す〃，〃システム側の言語で復唱する〃など，確認をとる十分な手段を用意する必要がある。この辺にどの程度の intelligence を導入できるかが自然言語の利用の成否を決定する

point となろう。

Word Processing, Text Processingなどで自然言語をデータとして使用し処理することは既に始まっており，この分野の本格的な発展について議論する必要はなかろう。

このレベルの処理で用いられる自然言語は完全なデータとしてであり，かつその意味をシステムが理解する必要は殆んどない段階である。但し，文章の誤りなどをシステムが指摘できるようにするためにはシンタクスのみならず，セマンティクスについての理解能力があるに越したことはない。

機械翻訳を考えると自然言語をデータとして使うにすぎないとしても可なりの程度の意味理解が必要になってくる。完全な自動化の前段階としてComputer Aided Translationが考えられ，この範疇にはText Processingと辞書を組み合わせたレベルのものから，相当程度の意味理解を用いるレベルまで考えられる。しかし機械翻訳で必要な意味理解は言語間の変換という枠内であるから，完全な知識としてシステムが理解することとの間には，尚大きなギャップがあるといえよう。

Q－Aシステムなどでの自然言語の利用は通常専門分野が限定されることからくる容易さがある一面，言語理解という立場からみると，機械翻訳よりは一般に一段と高度の理解能力が必要と思われる。

言語理解の立場から機械翻訳を一段低いレベルと考えたのは，夫々の言語の持つ単語の概念の間に1対1の対応がある時にはそれ以上の分析（理解）を必要としないと考えたからであり，たとえ1対1の対応がなくとも比較的簡単な対応関係を仮定しない限り，翻訳という業務自体が成り立たなくなると考えたからである。

Q－Aシステムにおける自然言語利用の難しさと，プログラム言語として自然言語を使うことの難しさとの間の差については今後の研究課題としたい。後者は適用分野が広くなるとすれば，それに伴うあいまいさの増加が考えられる。しかし適用分野を限定したときにも差がありそうである。Q－Aシステムとプログラ

ムシステムの本質的な差異を研究すれば解は自らでてくるものと思う。

学習能力は知能にとって可なり本質的な能力といえる。経験の蓄積が知識の根元であるにしても，これを系統化し，推論に役立てることができなければ知能の向上にはならない。〃使い易さ〃を実現するにしても，自己学習の能力があるか否かは大きな差となって現われる筈である。入力された知識を自動的に系統化する能力を当面の情報システムに期待することは困難としても，系統化の手法を示し，これを用いて入力された知識を整理し，その後のシステムの応答に役立たせることは可能である。各種パターン認識の辞書作りなどに実用されている手法をより広い応用分野に適用する努力ともいえる。

適応能力なども学習を前提とした能力ということができるが，入力を系統化し，定量化する機構と，変化させるべきパラメータのようなものが予め規定できれば適応能力を情報システムに持たせることは可能である。しかし予め予測できないような環境の変化に適応する能力を持たせることは非常に困難であり，当面あきらめるしかない。簡単なレベルでの適応能力を持たせることもシステムの性能向上に役立ち，〃使い易さ〃に貢献するものと考える。

連想能力は蓄積されているデータに効率よくアクセスするために欠かせない機能であり，頭脳の記憶能力をシミュレートできない最大の要因と考えられている。データ間の結合がなくとも全データをアクセスすることにより，必要なデータを読み出すことは原理的には可能である。しかし結合子そのものも情報を持っており，データ量の増加と共にアクセス時間が比例して増えることを考えると，是非とも実装する必要のある機能である。頭脳の脳細胞の一部しか実際には使われていないといったことも，今後の記憶装置の構成を考える上で考慮する必要のある事項であろう。アクセス時間の増加は単なる性能の劣化のように考えられるが，連想の問題では量的な差に留まらず，質的な差を作り出している要因となっている。今日の人工知能やパターン認識関係の研究をみていると，単なる量的な差と考えていたものが実際には質的な差を生み出しているように思える。記憶容量の増加とは無関係に一定のアクセス時間で急速に連想内容をとり出すことのできる

機能を実現することが理想である。

ユーザにとって使い易いシステムとしてそのユーザが行なう業務との対応の良さの問題がある。ユーザが個人の場合，すなわちPersonal Computerを考えると，そのユーザがそのコンピュータを使って行ないたい様々な業務に柔軟に対応してくれることが大切であろう。ユーザが組織であり，情報システムを中心に多数の人が共同作業をするような場合を想定すると，個々のタスクの入力と出力とが他のタスクの出力や入力と円滑に結合され，全体として一貫性のあるシステムとなっていることが大切である。また管理者にとっては各自の仕事の進み具合が情報システムを介して把握できることが望ましい。夫々の業務には特有の要求があり，これらを容易に情報システムに組み込めることが〝使い易い〟システム実現にとって大切な事柄である。

夫々の応用分野ごとに詳細な検討を進めれば更に多くの重要な示唆を得ることができると思うが，それらは今後の検討課題とし，いくつかのトピックスを述べるにとどめる。以下の各節では人工知能や自然言語処理などの諸問題を〝使い易さ〟の立場から調査した結果について報告する。

5.2 自然言語の概論

自然言語による意志の伝達は大まかに言って3つのレベルに分けることができる。

① 動機レベル：伝え手側から見れば，伝達を行なう目的，即ち相手に伝達したい何かの発生であり，受け手側から見れば，これを理解するのが最終目的となる。これは必ずしも文の表面的意味とは一致しない。例えば「時計をお持ちですか。」という質問は「はい」とか「いいえ」という答を期待しているわけではなく，「今何時ですか」という質問に等しい。

意志の伝達を効率的に行なうための戦略の決定もこのレベルに属する。

② 意味レベル：文の表現している意味が問題となるレベル。一般に言われて

いる自然言語認識ではこの意味の理解が問題となる。文の意味の抽出には普通知識による裏づけが必要となる。この知識には，一般常識や専門知識の他にその会話の行なわれている状況に関する知識も含まれる。

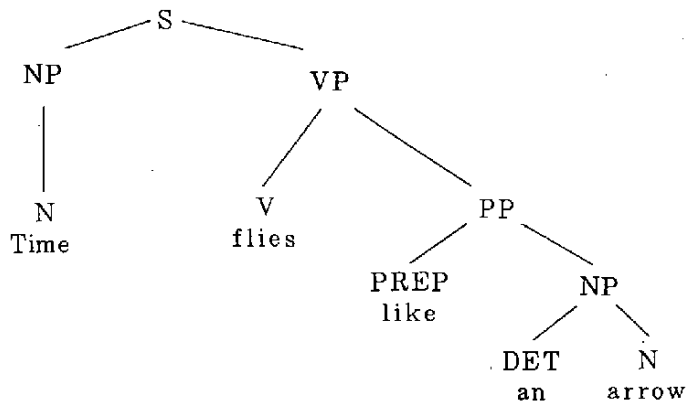
③ 文法レベル：実際に文が生成あるいは認識されるレベル。文は単なるシンボルの列としてとらえることができるから，このレベルなら機械的処理が可能であるが，普通多くの解釈が生まれ，文法のみではこのあいまいさが取り除けない。

以上一応3つのレベルに分けたが，これらは決して独立なものではなく互いに密接な関係を保ちつつ実際の会話が行なわれる。ここでいう会話とは，口頭（音声）によるものという意味ではなく，文書の交換あるいは端末からの入出力といった広い意味での自然言語のやりとりを指している。

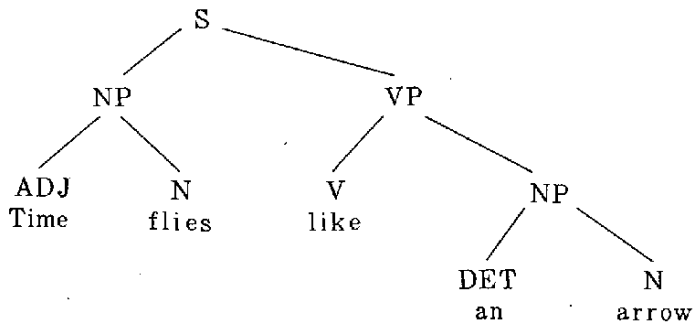
1960年代前半のいわゆる機械翻訳が研究された時代においては文法レベルのみによる言語の理解（翻訳）が試みられたのであるが，これは完全なる失敗に終わった。意味の裏づけなしには文法構造さえがあいまいであるということがわかってきたためである（一意に決定できなければ翻訳はできない）。有名な例として，

Time flies like an arrow.

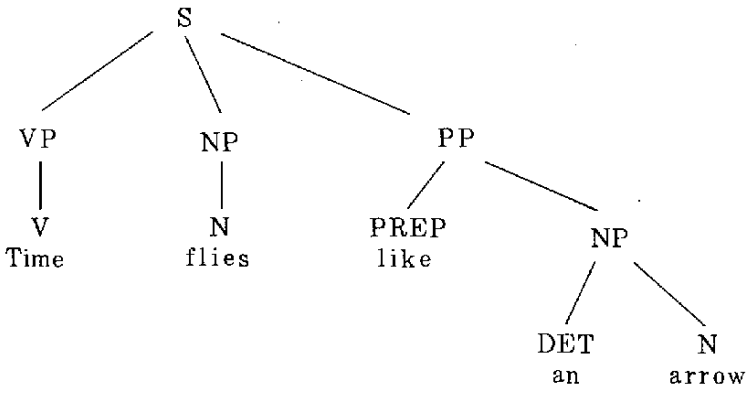
という文があるが，これは次の様に3通りに解釈できる。



光陰矢の如し



時縄は矢が好きだ



縄を矢と同様に計時せよ

これらのあいまいさは前後の文との意味のつながりを考えてはじめて解決できるのである。

通常の計算機との会話を想定するなら意味レベルまでの解析で充分であると考えられるが，実際の人間同志の会話では，強迫，哀願，いやみといったような，動機レベルまでの理解なしでは認識できないものも多い。人間の発する言葉は必ず何かの目的を持っているのであって，意味を解析するだけでは不充分なのであ

る。計算機の場合でも，本当に使い心地の良いシステムを目指すのならこのレベルまでの解析が必要であろう。例えばＱＡシステムにおいて，「……というデータを出力せよ」と言うよりは「……というデータは有るか」と聞くだけでそれを出力してくれる方がよほど使い心地が良いではないか。

以下，各レベルごとに現在行なわれている研究をまとめてみたい。ただし，この区分は筆者によるもので，研究者の意図とは無関係である。

(1) 動機レベルにおける研究

この分野の研究はかなり心理学や哲学の分野に入り込んでくるのだが，比較的そうでない部分を抽出して述べてみたい。

文には本来の意味の他にそれを発することにより他に及ぼす影響力（illocutionary force）を持っている。例えば「地球は丸い」という文は単なる事実の宣言以上に，相手にそれを信じさせようとする力を持っていると考えられる。相手が信じてくれれば成功であるし，そうでなければ失敗というわけである。相手がその事をすでに知っていることがわかっていれば，発話の必要すら生じない。窓が開いていて寒い場合に，状況によっては「寒い」といえば充分な場合と，「窓が開いている」と相手に教えなければならない場合，さらには「寒いから窓を閉めてくれ」と言わなければならない場合など色々考えられる。どのような表現をとれば（最少限の努力で）目的が達成されるかを知るためには自分の内部で相手をモデル化する必要がある〔COHEN and PERRAULT 79〕。又，相手が自分をどうモデル化しているか，及び自分が相手をどうモデル化しているかを知る必要がある。これはどこまでも続きそうだが，人間にはこの先は認知できないので3段階のモデル化で充分である。

計算機がユーザのモデルを内部に持っていて，それを利用してこちらの入力を解析してくれれば，入力に要する単語数等は大幅に減少することが考えられる。これは使い心地の向上に大いに役立つであろう。

(2) 意味レベルにおける研究

このレベルでは意味の表現方法，推論機構等が問題となる。又，文の意味を

理解するためにはその前提となっている世界に対する知識が必要である。常識として相手も知っていると考えられる情報は会話中には含まれない（前節参照）からである。

推論機構に関しては，従来の論理学は人間の思考形態とは必ずしも一致していないという考え方から様々の新しい方法が考えられているが，ここでは述べない。

知識表現に要求される機能としては以下のものが挙げられる。

① プロトタイプの表現：人間は典型的なものを基準として様々の表現，推論等を行なっていると考えられる。

② 標準値（default）の概念：得られる情報が不充分な場合に使用される標準の値。

③ 概念の階層化：概念どうしの間には半順序関係が成立していると考えられる。これらは木構造を成すとは限らず，分枝合流のあるものとなる。

④ 属性の遺伝：下位の概念は特に情報が与えられない限り上位の概念の属性を引き継ぐと考えられる。

⑤ 手続き的知識の表現：「百科辞典を引けば良い」といった類の，知識を得るための手続きに関する知識の表現が必要である。

以上の要請を満たすものとして，KRL（Knowledge Representation Language）〔BOBROW and WINOGRAD 76〕,frameの理論〔MINSKY 74〕, script〔SCHANK and ABELSON 77〕等が挙げられる。

KRLは心理学的成果をふまえたデータ，制御構造を採用する一方，人工知能の分野での汎用プログラミング言語となることを目指している。KRL－0の実現，評価〔BOBROW and WINOGRAD 77〕が終わり，現在KRL－1として同様の作業が続けられている。

Minskyのframe理論を大幅に取り入れたものとしてはFRL（Frame－Representing Language）〔ROBERTS and GOLDSTEIN 77〕がある。これは，標準値，階層化及び情報の遺伝を中心とする意味ネットワーク的な言

語である。

Schank の script は，会話に現われる場面（例えばレストランでの食事）を台本のようにしてあらかじめ知識として格納しておき，それを使って会話の理解を行なおうというものである。

以上のようなまとまったシステムを構成する道具としては，意味ネットワーク〔QUILLIAN 66〕や context dependency diagram〔SCHANK 72〕等が挙げられる。又，述語論理も広く使用されている。

人間の知識の総量は膨大なものであるが，分野を狭い範囲に限定すれば，その分野で必要とされる知識を計算機内に持つことは可能である。それを利用し，ユーザと自然言語で対話するシステムは数多く作られてきた。有名なものとしては，MYCIN(医療診断)，LUNAR（月面の岩石に関するＱＡシステム），SHRDLU（積木の世界），PLANE（米空軍の航空機の記録のＱＡシステム）等が挙げられる。

以上知識表現について述べて来たが，文の意味解析に関する興味深いものとしては，Hobbs による整合性（coherence）の研究〔HOBBS 78〕がある。会話の各部分は，目的に応じてある規則（elaboration，parallel又はcontrast）によって結合されているとし，これを解析することによって意味の理解ができることを示した。文の意味や一般知識を述語を使って表わし，述語論理に似た推論規則を使うことによって文の解析が機械的に行なえる。又，従来難かしいとされていた指示代名詞の解決が，この手法によって容易になることが示されているが，これなどは言語認識は文法レベルだけでは行なえないことのよい例であろう。

毛色の変わったものとしては，文法レベルを無視して，意味レベルだけで文の解析を行なおうという研究もある。これは人間が文法的に少々おかしな文であっても問題なく認識できるという事実に基づいている。

(3) 文法レベルにおける研究

自然言語の文法及び構文解析が主となるのが文法レベルである。広い意味では音声認識等もここに含まれる。前述のように，このレベルだけで文を解析するのは不可能なので，いかにして文法規則と意味との結合を計るかが主眼となる。従って，このレベルでは入力された文が意味構造に解析されるまでが問題となる。この過程で前節で述べた知識の利用や推論等が必要となるのは言うまでもない。

構文解析の手段として現在主流になっているのはATN（Augumented Transition Network）〔BOBROW and FRASER 69〕，格文法（case grammar）〔FILLMORE 68〕やcontext dependency〔SCHANK 72〕等である。

ATNは構文規則を状態遷移図として表現し，各遷移において色々な処理（意味づけ等）を可能にしたものである。

格文法では動詞を中心とした文の解析が行なわれる。動詞には満足されるべき条件（例えば主語や目的語の存在）がいくつか存在しそれらを探し出し，動詞の持つ意味によって結合するのが文の解析となる。frameの理論と結びつくことが多く，動詞が1つのframeを構成し，そのスロットに主語や目的語が入る形になる。

conceptual dependency diagramは各単語の意味をいくつかのprimitiveの組合わせによって表現し，それらの全体で文を表現するもので表面的に別の文であっても，同じ意味の文は同じ形に表現されるという特徴を持つ。

Montague は各単語の意味を述語で表わし，それによって文の意味を構成する研究を行なった。

(4) 結　論

以上見てきたように，現在の自然言語認識はかなりの分野に細分されて研究が行なわれている。数多くの成果があがっており，計算機との対話が自然言語で行なえる日も遠くはなさそうである。

人間の知識の全分野をカバーするのは無理としても分野を限定すれば実用になるシステムがいくつも生まれている。ＱＡシステムにおいては，入力の文の解析の結果，実行可能なプログラムが生成され，あとはそのプログラムを起動するだけであるので，文の解析が終われば仕事のかなりの部分が終了したと考えられる。したがってＱＡシステムのフロントエンドとして自然言語が使用されている例は多いようである。

計算機の入出力用として自然言語を用いる場合には，省略形を最大限利用して，できるだけ少ない語数での入力を可能にすることが使い易さの点であろう。

自然言語の持つあいまいさについては，システムの持つ知識で充分解決できるから，後は容量とスピードの問題であろう。

参考文献（順不同）

[COHEN and PERRAULT 79]
Philip R. Cohen, C. Raymond Perrault:
"Elements of a Plan-Based Theory of Speech Acts," Cognitive Science 3, pp.177-212 (1979)

[BOBROW and WINOGRAD 76]
Daniel G. Bobrow, Terry Winograd:
"An Overview of KRL, a Knowledge Representation Language," XEROX, Palo Alto Research Center CSL-76-4 (1976)

[BOBROW and WINOGRAD 77]
Daniel G. Bobrow, Terry Winograd, et al.:
"Experience with KRL-0, one cycle of a knowledge representation language," 5th IJCAI (1977)

[Minsky 74]
Marvin Minsky:
"A Framework for Representing Knowledge," MIT AIM #306 (1974)

[ROBERTS and GOLDSTEIN 77]
B. Bruce Roberts and Ira P. Goldstein:
"The FRL Manual," MIT AIM #409 (1977)

[SCHANK and ABELSON 77]
Roger C. Schank, R. P. Abelson:
"Scripts, Plans, Goals and Understanding: An Inquiry into Human Knowledge Structures," Laurence Erlbaum Associates, Hillsdale, New Jersey (1977)

[QUILLIAN 66]
M. Ross Quillian:
"Semantic Memory," Report AFCRL-66-189,
Cambridge MA: Bolt Beranek and Newman Inc. (1966)

[SCHANK 72]
Roger C. Schank:
"Conceptual Dependency: A Theory of Natural Language Understanding,"
Cognitive Psychology 3, pp.552-631 (1972)

[HOBBS 78]
Jerry R. Hobbs:
"Coherence and Coreference," SRI Technical Note 168 (1978)

[BOBROW and FRASER 69]
D.G. Bobrow, B. Fraser:
"An Augumented State Transition Network Analysis Procedure,"
IJCAI-69 (1969)

[FILLMORE 68]
C. Fillmore:
"The Case for Case," Universals in Linguistic Theory, New York:
Rinchart & Winston (1968)

昭和55年3月発行

発行所　財団法人　日本情報処理開発協会
東京都港区芝公園3－5－8
機械振興会館内
TEL　(434) 8211(大代表)

印刷所　山陽株式会社
東京都港区虎ノ門1－9－5
TEL　(591) 0248