52－S　001

分散型リソース処理技術の研究開発

昭 和 53 年 3 月

JIPDEC

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５２年度に実施した「分散型リソース処理技術の研究開発」の成果をとりまとめたものであります。

序

当財団は情報処理技術の研究開発の一環として、昭和５２年度より「分散型リソース処理技術の研究開発」に着手いたしました。

コンピュータ利用の高度化および複雑化にともない、ハードウェア、ソフトウェア、データ・ベース等のリソースの集中処理はほぼ限界に達してまいりました。

分散処理システムの出現によって大量のデータおよび複雑な処理が可能になった反面、新たに発生した異機種に分散しているデータおよびプログラムをどのようにして効率よく、かつ容易に利用するかという問題の解決に迫られております。このような問題を解決するためには、分散処理システムのリソース全体を総合的にとらえる必要があります。

また、わが国の情報処理の特徴の一つである日本語情報処理について、日本語情報処理のためのすぐれたデータ・ベースならびに低コストで操作性にすぐれた日本語端末の開発が必要であります。

当財団は、コンピュータ・ネットワークＪＩＰＮＥＴを有し、かつ過去４年間にわたって日本語情報処理の研究開発をおこなっており、このような問題解決のための研究開発を実施しうる環境をそなえていると判断いたし、昭和５２年度より３ケ年計画で、分散処理におけるリソースの統合に関する研究開発をおこなうこととといたしました。

本年度は、第一年度目にあたり、分散処理に関する一般的な検討、分散処理におけるデータ共有方式の検討、ＮＣＣ（ネットワーク・クリアリング・センター）に関する検討、標準制御言語に関する検討、および日本語情報処理の一般的な検討、事例、問題点、日本語端末モデル構築についての検討を行いました。

本書は、研究開発の中間報告としてその成果をまとめたものであります。

本報告書がこの方面に興味ある方々に広く利用され、わが国情報処理技術向上の一助として寄与できることを念願いたす次第であります。

昭和５３年３月

財団法人　日本情報処理開発協会

会　長　植　村　甲午郎

分散型リソース処理技術の研究開発

目　　次

ま　え　が　き

当財団では、日本自転車振興会の機械工業振興資金の補助を受けて、昭和４８年度より４年間に渡ってコンピュータネットワークＪＩＰＮＥＴを開発した。この技術成果として、インプリメント側に立ったコンピュータネットワークに関する多くのノウハウを得、それを発表した。過去４年間に渡って４０人年以上の工数をかけて作り上げたＪＩＰＮＥＴは、ホストおよび端末装置からの資源の共有を可能ならしめたが、コンピュータになじみの薄い利用者の立場からの意見では、利用し易い形でサービスを受けようとした場合に、利用したいホストのシステムのマニュアルを読まなければならない等の少なからぬ困難さをともなうものであった。

確かにコンピュータネットワークによって資源の共有が可能になったことから、利用者は意志さえあれば非常に多くの情報あるいは処理サービスを利用できるようになった。しかしながら、利用者に対して、それを使用するために多くの情報あるいはスキル等を利用者が持つことを要求することにもなった。

このような背景で、昭和５２年度から３ケ年計画で「分散型リソース処理技術の研究開発」のプロジェクトが発足された。

今までのコンピュータネットワークを中心とした新しいアーキテクチャがインプリメント側あるいは通信側指導型で進められてきたのに対して、このプロジェクトは利用者の立場から容易な利用形態を開発することを第一義に考えた処理技術の研究開発を目標としている。このプロジェクトでは過去にコンピュータネットワークに関して蓄積したノウハウおよび実験台として利用できるコンピュータネットワークＪＩＰＮＥＴを十分に利用して、データ共有方式、日本語情報処理を主たるテーマにして研究開発を進めてゆくスケジュールにしている。

本報告書は、３ケ年計画の第一年度目にあたる中間報告であり、分散処理、データ共有方式の検討、日本語情報処理の３部より構成されている。

第Ⅰ部の分散処理は、分散処理の一般論を述べている。第１章では分散処理の出現の背景を述べ、分散処理の定義をおこなっている。第２章では、ネットワークユーザの分類をおこない、ユーザの視点の重要性を論じている。第３章においては、ネットワークで共有されるリソースを検討し、データが特に重要なリソースであることを指摘している。

第Ⅱ部では、分散処理におけるデータ共有方式について議論している。まず第１章では、ユーザの視点からみた分散処理技術に関して、第Ⅰ部で分類を試みた分散処理関係者の各層に対応したシステムをそれぞれ紹介している。第２章では、データを管理する強力な道具としてデータベース管理システム（ＤＢＭＳ）をとりあげ議論している。最初のＤＢＭＳと言われるＩＤＳ以降の歴史を振り返るとともに、既存ＤＢＭＳの代表的データモデルのうち、階層型モデル、網型モデル、関係モデルについて詳解する。また、ベンダが提供するＤＢＭＳとして、ＡＤＡＢＡＳ、ＳＹＳＴＥＭ２０００、ＴＯＴＡＬの３つを採りあげ、それぞれのＤＤＬやＤＭＬについて具体例を示すことにより解説を行なっている。

第3章では、ネットワーク技術とデータベース技術を有効に組み合わせて実現している分散型データベースについて議論している。分散型データベースの第1の形態として、既に独立して存在している複数の情報（文献）検索システムの統合化アプローチを試みたCONITシステムとEURONETについて紹介する。CONITシステムは既に稼動し、EURONETについても1978年12月を目標に稼動準備が進められており、3種の異なったシステムにおいて共通コマンドのインプリメンテーション中である。アプリケーションを限った場合には、この種のアプローチが他のシステムへの影響度が少いなど現実的であろう。第2の形態として、アプリケーションを限定しない関係モデルをベースにしたシステムを紹介している。ここで紹介するシステムは、INGRES、SDD－1、LADDER、POLYPHEMEである。INGRESについては既に実用システムとして用いられている例も見られ、関係モデルをベースにした分散型データベースがいよいよ現実的になってきたことを述べている。

第4章では、ネットワークに散在するリソースを統合していく上で重要な機能の一つであるネットワーククリアリングセンターについて議論している。特に、リソースの所在源情報管理の基本的技術としてDD／Dに焦点をあて、DBMSの結合形態やディレクトリの配置や管理方法について言及している。

第5章では、異機種のコンピュータから構成されたコンピュータネットワークを利用するユーザが、まず直面する問題であるジョブ制御言語の多様性を解決する技術について議論している。つまり、コンピュータの異機種性を克服するために設定した標準ジョブ制御言語の基本構想を示している。

第6章では、第5章までの検討に基づいて、異種コンピュータネットワークにおけるリソース統合問題をとりあげている。

最後に、第6章で検討した異種コンピュータネットワークにおけるリソース統合問題に関してまとめた英文資料を掲載している。

第Ⅲ部は、日本語情報処理技術に関する研究開発についてとりまとめたものであり、5つの章からなっており、それぞれ次のような内容について記述している。

第1章では、当研究開発の背景とその目的について述べている。

第2章の第1節では、日本語情報処理技術の現状として、漢字入出力装置の調査分析と適用分野の事例紹介を行なっている。第2節では、漢字入出力装置をとくにオンライン処理という観点からその現状分析を行なうとともに、オンライン漢字処理の事例を紹介している。第3節では、日本語情報処理に応用される関連技術について、その動向を述べている。

第3章では、第2章の現状分析結果から、日本語情報処理全般の問題点、入力に関する問題点考察、出力に関する問題点考察、についてそれぞれ述べている。

第4章では、第2章の調査分析と第3章の問題点考察の中から、とくにコンピュータ端末としての日本語情報処理に視点をあてた、日本語端末モデルの構築を目指して、その入出力構成と処理機能の最適化について検討した結果を述べている。

第5章では、日本語情報処理全般の問題点および日本語端末の検討・考察から、今後の研究課題と展望に対して述べている。

Ⅰ．分　散　処　理

本部は、分散型リソース処理技術の研究開発の全体的背景を論じる。詳細は、第Ⅱ、及びⅢ部において議論する。

ＡＲＰＡＮＥＴをはじめとするコンピュータネットワークの発展は、データと処理との地理的、又は組織的分散を可能とさせている。従来の大型コンピュータへデータと処理が集中している"集中処理システム"に対して、これらが分散している"分散処理システム"が有効となってきたことを示している。この分散処理の出現の背景には、ＬＳＩを始めとするハードウエア技術の進歩と、１９６０年代後半よりはじまったコンピュータネットワーク技術の進歩とをあげることが出来る。この様な技術の発展は、システムを構成するハードウエア、通信、人間とにおけるコストの相対的関係に変化をもたらしている。分散か集中かの議論は、これらのコストに関して議論されねばならない。上記の点は第1章で議論する。

基本的なリソース共有手法を確立した分散処理システムにとって、一般ユーザに対して容易なリソースアクセス手段を提供することは重要となっている。一般ユーザとは、コンピュータに関する専門知識をもたず、リソースのアクセス手続には関心を持たず、何のリソースをサービスされるかにのみ関心を持っている。この様な一般ユーザに対して、リソースの異種性の問題と、必要なリソースがどこに存在するかという分散の問題とを視せないことが重要である。この問題へのアプローチとして、ユーザと分散処理システムとの間に、容易なユーザインタフェースを設けるユーザ支援システムがある。このユーザ支援システムを中心に、コンピュータネットワーク技術について第2章において議論する。

次に、リソースのなかで何が最も重要なものかを、第3章において考える。リソースとして、ハードウエア、プログラム、データの3つが考えられる。データのプログラムへ対する特徴として、量の大きさと時間的変化とをあげることが出来る。この点は、データを物理的に分散させることを有利としている。ＥＵＲＯＮＥＴ、ＣＹＣＬＡＤＥＳ等においても、データが最重要なリソースとして上げられている。分散処理システムにおいて、データが最も重要なリソースとして指摘出来る。

1. 分散処理の出現

コンピュータを用いた情報処理は、統合データ処理システム（integrated data processing system）として実現されてきている。統合データ処理は高度な情報判断を下すために必要なデータを広範囲にタイミングよく収集し処理するためのものである。この統合データ処理の必要性は益々増加していくものと思われる。統合データ処理システム実現において、経済性と情報流通の高度化とが重要な要因である。

１９６０年代における統合データ処理実現における情報流通の高度化を達成するための集中処理は、データの蓄積と処理との物理的集中を意味していた。しかし、１９７０年代にはいると、これまでのデータと処理の物理的集中は、管理運用面での非経済性が問題となりだした。このことは物理的集中に対する分散処理を出現させている。ＬＳＩを始めとするコンピュータハードウエア技術の発展と、ＡＲＰＡネットワーク、ＣＹＣＬＡＤＥＳ等のコンピュータネットワーク技術の成果とを背景として、データと処理とを地理的に分散させることを可能としたわけである。

本章では、1.1で分散処理の出現、1.2で分散処理の定義、1.3で分散と集中の議論、1.4で分散処理とコンピュータネットワークとの関係を述べる。

1.1 分散処理出現の背景

分散処理出現の背景として、ハードウエア技術とコンピュータネットワーク技術の発展を上げることが出来る。

1.1.1 ハードウェア技術の進歩

近年の半導体集積回路技術、特にＬＳＩ（Large Scale Integration）技術の進歩は、コンピュータを数cm角のチップとして実現させるようになった。これらは、マイクロプロセッサ又はマイクロコンピュータと呼ばれている。前者は主として中央処理装置を意味し、後者は入出力装置等を備えたミニコンピュータを指向するものである。

マイクロプロセッサ／マイクロコンピュータの特徴として次の点を指摘出来る。

1）小型である。例えばＣＰＵは5mm角、記憶装置等を含めても１０～２５cm角のボード上におさまる。この様に小型であるために、端末等の種々の制御装置として容易に組込める。

2）安価である。一般的に、ＣＰＵだけの場合約２０ドル、メモリを含めた場合は約1,０００ドル、入出力装置まで備えた汎用コンピュータの場合は約1万ドルである。安価であるために、広範囲に利用出来る様になった。

この様に、小型で安価なプロセッサは、従来のグロッシュの法則 ― コンピュータの費用(C)と効果（Ｅ）の間には$E \propto C^2$ の関係がある ― には基づかないものである。大型コンピュータにおいては、図１－１に示すようにグロッシュの法則が成立している。しかし表１－１に示されているように、大型コンピュータとマイクロ／ミニコンピュータの単位費用当たりの処理能力の

間には１～２桁の差がある。このことは、マイクロ／ミニコンピュータで処理が可能ならば（複数台で処理することになったとしても）、大型コンピュータではなく、マイクロ／ミニコンピュータを用いた方が少ない費用で処理出来ることを示唆している。もちろん、マイクロ／ミニコンピュータでは処理能力は大型コンピュータと較べて限られたものである。この点は、マイクロ／ミニコンピュータの重大な制約となっている。通信技術を用いて、複数のマイクロ／ミニコンピュータを結合し、これらの相互の通信を可能とすることによって、システム全体として大型コンピュータに匹敵する処理能力の実現が可能となってきた。これはコンピュータネットワーク技術の発展によっている。

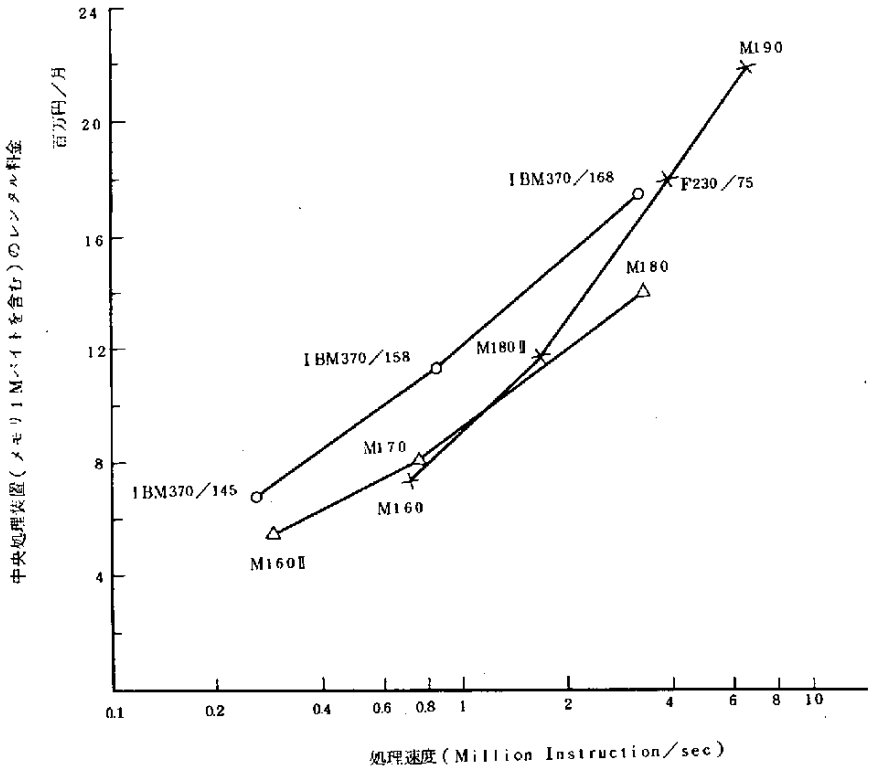


図１－１　処理速度　Ｖ．Ｓ処理装置コスト

表１－１　処理コストの比較

コンピュータ	買取り価格（百万円）	処理コスト（百万円／ＭＰＳ）
ＩＢＭ　３７０		
／１４５	３３２.５８	１２７９.
／１６８	５８６.４６	１８２.１
ＦＡＣＯＭ　Ｍ		
／１６０	３３３.	４８２.６
／１９０	９８５.５	１５２.８
ＨＩＴＡＣ　Ｍ		
／１６０Ⅱ	２４７.５	８５３.４
／１８０	６７.５.	２０９.６
ＰＤＰ　１１		
／０３	０.８	２.８
／０４	１.４２	４.５
／３４	２.５３	５.１

注　処理コスト＝買取り価格÷ＭＰＳ（百万命令／秒）
買取価格は、ＰＤＰは３２Ｋバイトのメモリと ＣＰＵ
他は１Ｍバイトのメモリと ＣＰＵ

1.1.2　コンピュータネットワーク技術の発展

ＡＲＰＡＮＥＴ、ＣＹＣＬＡＤＥＳ、ＴＲＡＮＳＰＡＣ等に代表されるコンピュータネットワーク技術の成果として、プロトコルとパケット交換網との２点を上げることが出来る。

プロトコル〔ＩＳＯ７７〕は、ネットワークの種々の要素間の通信を行なうための標準規則を定めている。要素とは、コンピュータ、端末のようなハードウエアのこともあるし、ソフトウエアのこともある。要素は他の要素へあるサービスを提供することもあるし、他の要素のサービスを受けることもある。このため、互いに通信し合うネットワーク要素を明らかにすること（即ちネットワークアーキテクチャ）が重要である。ネットワークの基本アーキテクチャは、図１－２と図１－３に示すような階層構造で表わされる。各々の階層は、上位の階層へ対してより専門的機能を提供することによって、ネットワークへ新たな価値を付加していく。

一般的にネットワーク階層は図１－２の様に表わされる。

1）ｎ層は、（ｎ－１）アクセスを用いて（ｎ－１）層から得られる（ｎ－１）サービスを利用する。

2）ｎ層は、（ｎ－１）層以下の階層を関知する必要なく、（ｎ－１）ボックスによるサービスだ

けに関している。

3）（ｎ）層は、（ｎ）プロトコルを用いて互いに通信する（ｎ）エンタティから構成される。

4）（ｎ）サービスを（ｎ＋１）層へ提供するために、（ｎ）エンタティは（ｎ－１）サービスを用いて（ｎ）機能を実行する。

5）アーキテクチャの層の仕様は、下位層によって与えられるサービスを何等かの形で参照していなければならない。この参照はアクセス機能を用いてなされるだろう。サービスへのアクセス機能の集合は、単にネットワークの論理構造を表わす手段としてみなされる。このことは必ずしもあるネットワークのインプリメンテーションにおける対応するインタフェースの存在を意味しない。

上記の一般形を、実際のコンピュータネットワークアーキテクチャへ適用した例を図１－３〔ＯＫＩ〕に示す。データ通信システムにおける階層化とプロトコルの概念は、従来のアプリケーションごとに特別なシステムを構築する方法から、一般的な融通性のあるシステムの構築を可能とさせた。

コンピュータネットワークのデータ交換網は、パケット交換方式を用いている。パケット交換技術は、従来の回線交換方式、メッセージ交換方式に対して、応答性、高信頼性、低費用が上げられている。この方式は、メッセージをパケットと呼ばれる固定長の情報単位へ分割（又は生成）し、各ノードのパケット交換機で高速に蓄積交換され目的地までパケットを送るものである。この背景には、先に述べたハードウエア技術の進歩による安価なマイクロ／ミニコンピュータを交換機として利用出来るようになった点をあげることが出来る。この様なパケット交換網の出現によって、分散処理のために必要なコンピュータ間通信が可能となった。

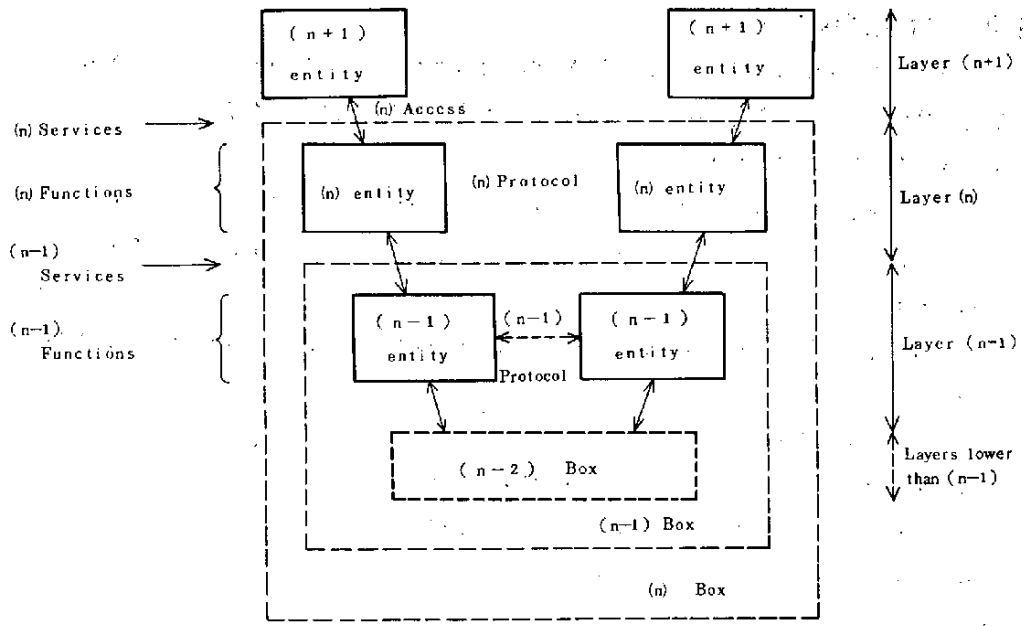


図１－２　プロトコル階層〔ＩＳＯ７７〕

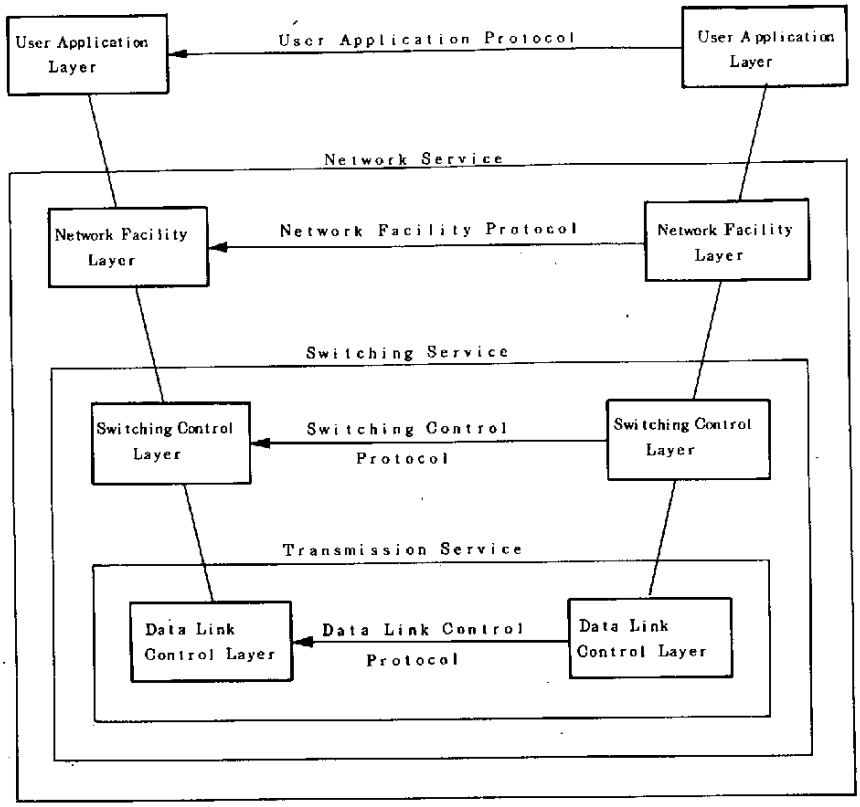


図1－3　DONAにおけるプロトコルの階層

1.2　分散処理の定義

本節では分散処理の定義を試みる。"分散処理とは何か"という問題に対して、種々の層の人々が種々の観点から定義を行なっている。まず、これらの定義を概観してみる。

A　谷野（東芝）〔YANO77〕

分散処理とは、プログラム内蔵方式のプロセッサが物理的に結合されており、各々のプロセッサ上のソフトウエアが与えられた機能を分担して、論理結合を通じて全体として所望の機能を処理するものである。

ANSA（Advanced Network System Archtecture）は、次の形で分散処理の適用をおこなう。

a．処理の分散（利用者プログラムレベルの分散処理）

b．データベースの分散（利用者データレベルの分散処理）

c．管理の分散（ネットワーク管理レベルの分散処理）

d．機能の分散（システム機能レベルの分散処理）

B　石原（IBM）〔ISHIHA76〕

分散処理の目的は、今日の集中化されたデータベースシステムを利用し、利用者のより良い

業務処理、生産性向上、コスト削減等を実現することにある。

分散システムの全般的な概念は、データベースおよびデータ通信機能を持った中央演算処理装置と、それより低いレベルのデータ情報と端末装置をもった分散制御装置を含んだ構成で示される。各レベルにおいては、それぞれのレベルで必要なデータを持つ。

分散処理というのは、広い意味では、一定の規則および約束の枠内で利用者の要求に合うように、各種のモードで、多くの場所で、データ処理機能を活用できる形態といえる。

このような分散処理は、ＩＢＭのＳＮＡ（ System Network Architecture ）と両立し、ハードウエア、ソフトウエアの両方でサポートされている。

C Canning〔ＣＡＮＮ７６〕

７０年代の初めごろ、それまで集中化の方向にそって発展してきた処理形態から、分散システムに向かう傾向が始まり、しだいにその傾向が強まってきた。

システムが分割される方法にも種々あり、次の分類ができる。

- 分散処理システム：データファイルは集中あるいは存在せず、アプリケーションロジックが分散
  - データ・エントリ・システム
  - リモート・ファイル・アクセス・システム
  - 分散コンピューティングシステム
- 分散データシステム：アプリケーションデータが分散されたプロセッサ近くに置かれるシステム
  - 集中／分散データシステム
  - 完全分散データシステム

図 １ － ４　分 散 処 理

分散システムにかかわる大きな問題として、どのように分割するのかというものがある。

a．アプリケーション・システムの分割

実際のアクティビティをそれぞれのノードに割当てるようにし、ノード間にまたがる活動ができるだけ少なくなるように留意する。これは結局は、場所的（地理的、組織的）な問題になる。

b．機能分野による分割

企業の各部門が機能的あるいは業務部門を中心として分割されたシステムを持つことを示す。

c．データ処理機能の分割

リモートデータエントリ用にミニコンピュータを持ったキーツーディスクやインテリジェント端末などを利用する。

D Enslow（Georgia Institute of Technalogy）〔ＥＮＳＬ７６〕

分散システムは次のエンタティが異った場所において、物理的あるいは概念的にばらばらに存在するものをいう。

a．演算機能

b．処理操作

c．ハードウエア・システム制御

d．ソフトウエア・システム制御

e．データベース

分散システムの基本要素は、次のものである。

a．複数個の汎用プロセッサ

b．各々のプロセッサはオペレーティングシステムを持つとともに、全体としてのオペレーティングシステムがあるものと考えられる。

c．情報、制御のためのプロトコルが存在する。

d．サービスがどの要素で実行されるかは利用者は関知しない。

e．仕事をおこなうための要素の選択は、前もって決められてはいない。

以上のエンタティが分散していたとしても、次のものは分散システムとはいわない。

a．簡単な端末装置が散らばっている。

b．メインフレーム内で処理が分割されている。（Ｉ／Ｏプロセッサ、演算プロセッサなど）

c．主従関係のあるもの。（インタフェースとして主装置の指示の拒否ができない）

d．専用プロセッサ（ベクトル乗算器、浮動演算装置）

e．インテリジェント端末システム

分散の度合を特徴づけるものとしては、次の３つがあり、それぞれを軸とする３次元空間上においてそのシステムの分散度を認識できる。

a．ハードウエア構成

単一ＣＰＵ、多重実行装置、専用処理装置、多重プロセッサ、多重コンピュータ

b．制御の構成

単一、主従関係、自律的な制御点が複数個、操作に関する複数の制御点、実行に関する複数個の同種制御点、実行に関する複数個の異種制御点

c．データ・ベース構成

集中、データコピーを複数個、完全な分散

分散処理の定義にあたって、どのような人々が関係を持つかを考えてみる。次のような部類に属する人々がそれぞれの立場から関与していると考えられる。

(1) コンピュータメーカ

a．ハードウエア担当

b．ソフトウエア担当

(2) コンピュータユーザ

a．ＥＤＰシステム開発担当

b．エンドユーザ

(3) 通信業者

A　ハードウエア担当

分散処理を実現するためには、安価な装置を世の中に出す必要がある。これは前項で述べたように、ＬＳＩ等の技術進歩によって達成されている。もう一つは高いパフォーマンスを得るために、メインフレーム中で機能分担（Ｉ／Ｏプロセッサ、浮動小数点プロセッサ）マルチＣＰＵ、あるいは複数個のコンピュータを安価に結合することを考えるであろう。

このように、ハードウエア担当者としての立場から考えてみると、ハードウェアの要素の分散という観点があることがわかる。

B　ソフトウェア担当

コンピュータのソフトウェアは、ハードウェアに指示をあたえるプログラムの総称であることから、新しいハードウェアの出現によって、当然新しいソフトウェアがでてくる。しかし、メインフレーム内での機能分担では比較的大きな変化は現われなかった。ソフトウェアの担当者に大きな影響をあたえたのは、複数個のコンピュータが結合され、独立したコンピュータ間での機能分担が発生した時点であった。

これは２つの形態をとって現われている。まず第１は、フロント・エンド・プロセッサ、バック・エンド・プロセッサといわれるようなもので、ソフトウェアの観点からすれば、オペレーティングシステムが分割されたものである。このため、この形態では、コンピュータ・ユーザからみたときには、処理の分散は全くみえない状態となっている。

第２は、ユーザからみたときにも、機能が分割されているのがみえるもので、ユーザプログラムを処理できるインテリジェント端末である。この形態においては、コンピュータ・メーカとしては、分散処理を可能とする道具をユーザに提供するものであり、その道具をどのように使うかは、ユーザに依ることになる。

ここでは、ユーザによってシステムの構築方法を全く変えることができるという意味でもってアーキテクチャという言葉が使用されていると考えている。

簡単にいえば、メーカのソフトウェア担当という立場では、複数個のコンピュータを有効に使うために、分散して処理を管理するオペレーティングシステムを提供することが分散処理に対するかかわり合いである。

C　ＥＤＰシステム開発担当

分散処理の出現によって、コンピュータ・ユーザのＥＤＰシステム開発担当者が最も多くの選択をおこなわなければならなくなった。すなわち、どこにどのようなデータを持ち、どの処理をどこでおこなうかなどの重要な選択をおこなわなければならない。

この意味で、分散処理の主要な観点は、ここにあると考えられる。

D　エンドユーザ

分散処理の出現によって、今までどこか遠くにあったコンピュータが身近かな所に設置される

ことになり親密感がでてきたかも知れない。しかしながら、どこでどのように処理されるかは、本来問題ではなく、正しく迅速に処理され、自分の求める情報が得られればよいという立場からすれば、あまり関係がないと考えるべきであろう。

E　通信業者

分散処理システムは、地理的に離れた部門、地理的ばかりでなく異なった組織に属するコンピュータを結合するものである。このために、通信業者のサービスを利用することは必須条件となる。近年は、通信業者のサービスも、単に回線サービスばかりでなく、付加価値ネットワーク（ＶＡＮ）サービスもおこなわれるようになってきている。さらに、端末装置の制御、ホストによる情報処理サービスまでも含めた範囲でのサービスを意図する動きもでており、コンピュータ関係業者との間に競合関係が生まれてきている。

このように、サービスの種類、内容は豊富になってきている。通信の両端にある装置の種類、機能はサービスの向上、需要の増大のための対策として意識はするが、本質的なものではない。この意味で、分散処理は通信業者に対しては、１つの利用形態にすぎないであろう。

それぞれの立場の人々の観点をまとめると、ハードウェア要素間での機能分担に関するもの、複数個のコンピュータを結合したとき、どのようなオペレーティングシステムを提供するかという制御を中心とした立場、前二者を利用して処理およびデータの蓄積場所を考慮し、処理コストを低くおさえる立場にあたるものがある。

このような条件をふまえて、ここでは分散処理を次のように定義する。＂分散処理＂とは、広い意味では、複数のコンピュータを結合し、それらのコンピュータ間で処理を分担し、全体として目的に合った処理をおこなわせる方式をいう。狭い意味では、広い意味の分散処理をおこなうために結合されているコンピュータのおのおのに、利用者の応用プログラムが組み込まれており、利用者の処理が分散しているときに、分散処理という。このように、分散処理の定義としては、２つの定義があるが、分散処理においては、ユーザの処理を主体と考えるべきであるという立場から、狭義の定義を採用する。

分散処理には、分散の度合というものがあろうが、これは、アプリケーションを処理しているコンピュータの数、データベースの数および分散処理全体の制御点の数がその要因となる。

1.3　分散と集中について

ハードウェア技術とネットワークに関するソフトウェア技術の進歩によって、集中処理が管理、運用などの面から考えて経済的ではないという側面がでてきた。これは、それまでおこなわれてきた物理的な集中に対して、論理的に統合され、物理的には分散された処理方式もシステム構築する上で考慮の対象となってきたというにすぎない。この意味で、分散処理は、現在の時点においては集中処理を駆逐するものではなく、アプリケーションシステムごとにコストの低い方式が採用されるという、相対的な価値しか持っていないと考える。

それでは、どのような分野において分散処理が採用される可能性があるだろうか。図１－５は、

１９６０年における処理、ソフトウェア、通信のコストを１００とした時に、それ以降どのくらい安くなったかを示したものである。また表１－２は、１９６０～１９７５、１９７５～１９８５にコストがどのくらい下がるかを表にしたものである。これらのデータから明らかなように、１９７５～１９８５年の間では、処理および蓄積コストにくらべて、通信およびソフトウェアコストの低下は少ない。このことは蓄積および処理をおこなうコンピュータのリソースを余分に使ったとしても、通信あるいはソフトウェアコストの削減をおこなった方が有利であることを示している。これは、通信の側面からみれば、ローカルな処理はローカルなコンピュータで処理し、通信量を減らすことによる方向、すなわち分散処理を推進する力となるであろう。また、ソフトウェアの側面からみたとすれば、コンピュータ処理コストよりも人件費を少なくするという意味で、コンピュータに業務をのせること、ハイレベル言語の利用により開発コストを削減する方向に向かうであろう。

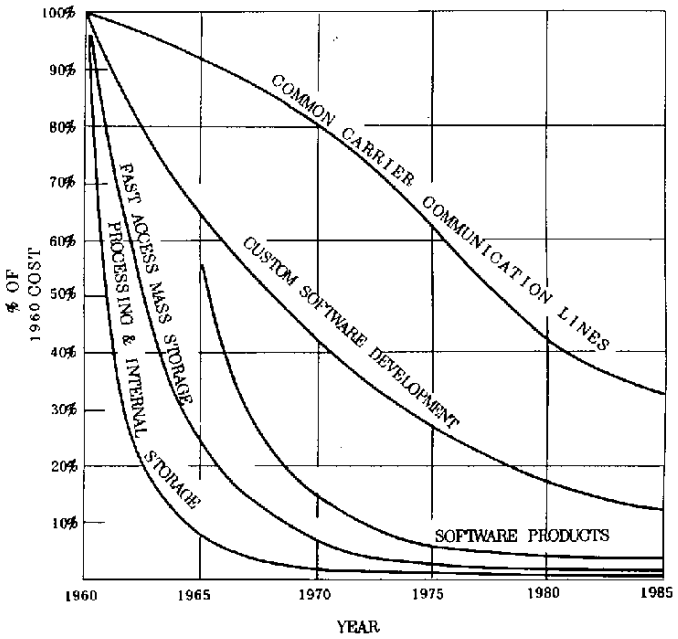


図１－５　COST　vs.　YEAR〔WAGN７６〕

表１－２　コスト　〔WAGN７６〕

	1975 Costs as a Fraction of 1960 Costs	1985 Costs as a Fraction of 1975 Costs
Hardware		
Processing and Internal Storage	0.005	0.20
Fast Access Mass Storage	0.02	0.10
Common Carrier Communication Lines	0.61	0.53
Software		
Custom Development	0.28	0.47
Software Products	0.06	0.33

コスト面以外にも分散化の要因となるものはある。例えば、コンピュータ処理量の増大化にともなって、１台のＣＰＵでは処理できなくなり、複数台のコンピュータを利用する、あるいはハードウェア要素の機能分担による広い意味での分散処理が出現した。また、データベースの増大は、保全性のために分散化の方が有利になることがあると考えられる。しかしながら、統合的なデータの利用は、必然的に制御の集中あるいは統一化を要求し、分散された資源の有効利用のために複雑な機構を要求することになる。また、各サイトにおける資源管理、安全対策に対するばらつき、分散設置による管理費の増大などを生む要因をも持っており、分散は集中にくらべて絶対的に有効なものではなく、アプリケーションシステムごとに選択をされる一つの方式である。この選択のポイントとなるのは、コストである。

1.4　分散処理とコンピュータ・ネットワーク

分散処理と同じような構成を持つものにコンピュータ・ネットワークがある。コンピュータ・ネットワークの定義として使われるのは、"おたがいの資源が利用できるように結合された自律的なコンピュータ"というものである〔ＲＯＢＥ７０〕。複数個のコンピュータの処理という点では、分散処理と全く同じである。分散処理がアプリケーションの処理を複数個のコンピュータでおこなうという、処理を中心に考えているのに対して、コンピュータ・ネットワークにおいては、それぞれのコンピュータが管理している資源を中心に考えている。

このように観点は異なっているが、コンピュータ・ネットワークは、分散処理に対して次の条件をつけたものである。

(1) 参加コンピュータは、独立で自律的でなければならない。

(2) 独立したコンピュータ間のサービスは、ギブ・アンド・テイクである。

このような、観点、範囲の差はあっても、基本的な技術、利用方法はほとんど同じである。このため、これ以後の検討にあたっては、コンピュータ・ネットワークの成果も分散処理を検討する上での重要な資料として参照する。

2. ユーザの視点

本章では、分散処理システムを検討する上で、ユーザの視点（User's View）が重要であることを述べる。更に、ユーザの分類を行ない、各々のユーザ分類に対応するシステム例を考える。これらのシステム例の詳細は、第Ⅱ部第1章で述べる。

2.1 ユーザの視点の重要性

ハードウェア技術とネットワーク技術の発展は、コンピュータ、通信、人間のコストにおける相対関係の変化をもたらしてきている。又、コンピュータネットワークの発展は、種々の分野のユーザの参加をもたらしている〔ＲＯＳＥ７６ａ〕。ユーザとは、数学、物理学、医学、社会学等の研究者達であり、学生、株の仲買人、座席予約システムの窓口オペレータでもある。この様な多種多様なユーザ層の多種多様な要求に対して、コンピュータサービス提供者は彼等の保有するリソースに基づいて最良と思われるサービスを設計し、市場へ出そうとしている。即ち、分散処理システムの市場価値は、上記した様な多種多様なユーザ要求を満たせるかどうかに依存している。1960年代後半より始まった分散処理システムは、基本的通信機能の構築を終わり、これらを用いたユーザアプリケーションレベルへとその視点を移してきていることでもある。分散処理システムの今後の検討において、一般のユーザの視点が重要である。

2.2 ユーザの分類

次にコンピュータネットワークに関わるユーザの分類を試みる。ここで述べるユーザとは分散処理に関係する人々という広義の意味である。分類は図２－１に示すように、第１にコンピュータネットワークの開発者と、彼等によって開発されたネットワークの利用者とに分類される。ユーザとは狭義にはネットワークユーザを意味する。

ネットワークユーザは、ネットワーク利用に関する熟練度によって、熟練ユーザ（expert user）と一般ユーザ（casual user）とに分類出来る。一般ユーザとはコンピュータとネットワーク利用に関してほとんど知識を持たない初心者ユーザである。一般ユーザにおいて重要な点は、自分の要求に対する答を非手続的に求めており、実際のシステム構成、プロトコル、インタフェース等には興味を持たないことである。これに対して、ある程度の知識を持っているものを熟練ユーザ（expert user）と呼ぶ。現在のネットワークユーザの大半は熟練ユーザであるといえる。しかし将来の分散処理システムのユーザは、一般ユーザと呼ばれるユーザ群が増加してゆき、ネットワークユーザの大半を占めるであろう。この様に、分散処理システムの市場価値は、この一般ユーザ群に依存することになろう。

一般ユーザを対象として、容易なリソースアクセス手段を提供しようとする試みは、ユーザ支援システム（user assistant）として知られている〔ＲＯＳＥ７６ａ〕。これはＲＥＸ〔BENO７４〕、ＲＩＴＡ〔ＡＮＤＥ７６、７７〕、ＮＡＭＣ〔ＲＯＳＥ７６ｂ〕等である。

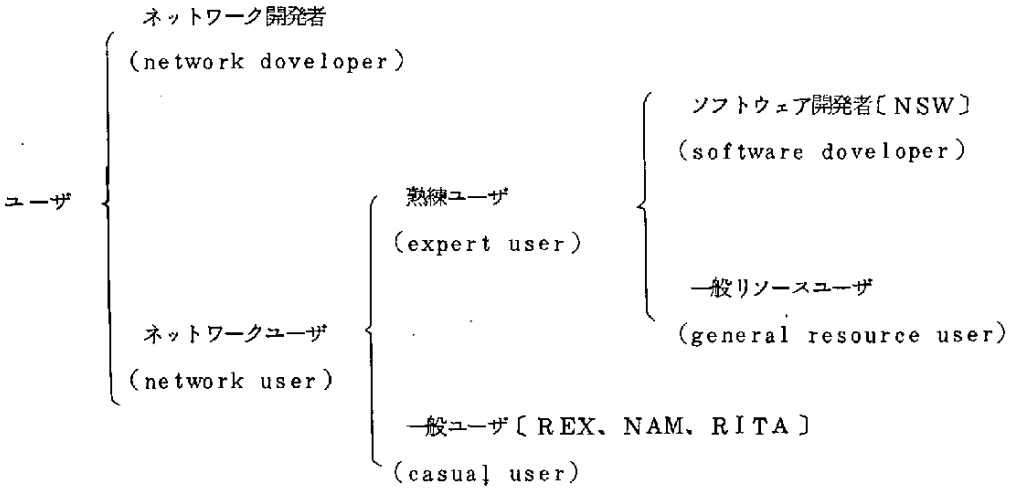


図 2 － 1　ユ ー ザ の 分 類

一般ユーザにとって、現在のコンピュータネットワーク利用の困難さは次の2点によると考えられる。

1）ネットワークに含まれるホストコンピュータの異種性。即ち、各ホストによってログオン、ログオフ、コネクションの初期設定、切断等の手順、コマンド言語、言語等は異なっている。このためユーザは各ホストのこれらの情報を詳細に知る必要がある。これは一般ユーザにとっては大きな負担である。

2）リソースがコンピュータネットワーク上に分散していることによってリソース所在を知る必要性。ユーザにとって必要となるリソースがどこに存在するかを明らかにする問題である。ARPAネットワークではリソースハンドブックとしてネットワーク内のリソース情報を配布しているが、一般ユーザにとって自分の仕事に必要なリソースを見つけ出すことは困難である。

ユーザ支援システムと呼ばれるものは、上記の様なユーザアクセスの困難さを解決し、容易なリソースアクセス手続を一般ユーザへ提供しようとしている。よって、ユーザ支援システムは、ネットワーク技術からみて重要である。

NAMは主に1）の問題を解決するものとしてあり、REXは2）に関している。RITAは、1）と2)へ合わせて容易なユーザインタフェースを目指している。これらの詳細は、第Ⅱ部第1章に述べられる。

熟練ユーザは、システム上のリソースをどの様に使うかによってソフトウェア開発者と一般リソースユーザ(general resource user)とへ分けられる。前者はネットワーク上に分散した幾つかのリソースを統合的に利用して新しいリソースを開発するものである。一方、後者は、サービス提供者によってサービスされる単一リソースの利用者である。前者の例としてはNSW〔WHIT75、SCHA76〕がある。NSWは、コンピュータネットワーク上に分散している

ソフトウェア開発用ツールを共有するためのユーザインタフェースを提供するものである。この目的は、ソフトウェア開発に要する費用、時間、労力を軽減することである。この詳細は第II部第1章を参照されたい。

3. 分散処理の主要なリソース

ＡＲＰＡネットワークの開発に際して、その提案者であるRoberts 等は、コンピュータネットワークにおいて共有されるリソースにはハードウェア、ソフトウェア及びデータがあると述べている〔ＲＯＢＥ７０〕。本章ではこの様なリソースのなかで、何が最も重要であるかを検討する。

3.1 共有されるリソース

Roberts が述べるように、分散処理システムにおけるリソースとは、ハードウェア、ソフトウェア、データの３つがある。まず、ＡＲＰＡネットワークの場合を考える。

ハードウェアとして、ＡＲＰＡネットワークでは、ＩＬＬＩＡＣ Ⅳをまず上げることが出来る。ＩＬＬＩＡＣ ⅣはＮＡＳＡ－ＡＭＥＳに設置され、ＰＤＰ－１０をＦＥＰとしてＡＲＰＡネットワークに結合されている。ＩＬＬＩＡＣ Ⅳは、超高速なマトリックス演算能力を備えた超高性能ハードウェアである。他に、ＣＣＡ(Computer Corporation of America)で開発されたデータコンピュータ〔ＭＡＲＩ７５〕がある。これは１兆ビット(10^{12} bit)のメモリを、ＡＲＰＡネットワークを通して容易に経済的に使用させるものである。異種コンピュータ間でのデータ共有、巨大メモリのサポート、集中型データベース管理機能とを特徴としている。

ソフトウェアとは、ＭＩＴのＭＡＴＨＬＡＢ、ＳＲＩのTheorem Prover、ＢＢＮのNatural Language Processor、リンカーン研究所のＬＥＡＰ、カーネギ・メロンのＬC^2 などを言う。

データとは、ＡＲＰＡネットワークの例でいえば、ＳＲＩのＰＤＰ－１０において実現されたネットワーク・インフォメーション・センタ(ＮＩＣ)の如きものを言う。ＮＩＣは、ネットワークに関する文献、マニュアル等のような情報を貯蔵しており、全ユーザに最新のハードコピーを配布する役目をしている。

以上述べたように、コンピュータ・ネットワークにおける主要なリソースは、ハードウェア、ソフトウェアおよびデータであることはすでに世の中で共通的に認識されている。

ＡＲＰＡネットワークの開発にあたって資源に関する考察は上記のとうりであったが、他のネットワークにおいてはどうであったかをみてみよう。CYCLADESもハードウェア、ソフトウェアとデータとの３つのリソースの共有を述べているが、その中心となるものとしてデータベースを考えていた。〔ＩＲＩＡ７４〕に次のように記されている。

「このネットワークはある特別な利用のために考えられるものではない。それは既存のアプリケーション、または開発中のアプリケーションに対して、より多くの顧客との接触、より大きな多様性をもったサービスの提供を可能にするものである。それはまた、単独のコンピュータではできない新しいアプリケーションの実現を可能にする。実験の主要な領域は、データ・ベースへのアクセスおよび官公庁内での情報の交換となるだろう。」

昭和４８年度より、３年間継続の特定研究として進められた「広域・大量情報の高次処理」〔ＳＨＩＭ７７〕のプロジェクトにおける主旨は次のとおりであった。

「研究・調査等における学術情報、すなわちデータならびに文献情報が急速に大量、広域になってきている。専門または地域において広域の多数の利用者によるこれら学術情報の効果的な利用のためには、効率的な高次情報処理を行なう巨大情報システムの研究開発が必要である。すなわち、学術情報のシステム化、情報資源共有ネットワーク（ resource sharing network ）の形成および高水準の構造解析にもとづく高度の情報処理システムの研究が要求されている。これらは、学界の緊急な課題であり、かつ、情報化社会の礎石となるべきものである。」

１９７５年からＥＣにおける科学技術情報分野の活動として、ＥＵＲＯＮＥＴの計画が開始された。ＥＵＲＯＮＥＴのサービスは、データベースにオンラインでアクセスしようとするユーザとオンラインサービスを目ざすデータベースサービスの提供機関に対するものである。１００以上のデータベースをサービスしている２０以上の機関がＥＵＲＯＮＥＴに参加する意向を示しており、この中にはＥＳＡ／ＳＤＳ（欧州宇宙開発機構／宇宙ドキュメンテーションサービス）、ＤＩＭＤＩ（西独国立医学情報センター）も含まれている。この様に、コンピュータ・ネットワークの分野におけるリソースの共有というものは、ソフトウェア、ハードウェアというよりはむしろデータが主体となりつつあると考えてよい。

3.2　主要な共有リソースとしてのデータベース

3.1節で述べたように、分散処理システムにおけるデータベース共有への大きなニーズがあることは明らかである。我々は、データベースを主要なリソースとみなし、今後の検討を進めていく。

先ずデータベースの特徴を考えよう。表３－１に、データベースに関する調査〔ＫＯＴＡ７７〕による結果を示す。表3－2にプログラムファイル の大きさを示す。両者の比較で明らかなように、量において10^2 ～10^4 程度の差があることが判明する。即ち、データベースの第１の特徴はその大量さにある。

第２に、処理とともに値が変化することである。変化の間隔としては、座席予約などのように秒単位のものと、文献検索のごとく１ケ月～３ケ月のようなものまである。

第３に、利用がきわめて地域あるいは組織的にかたよっていることである。例えば北海道地方の座席予約データの参照は、北海道地方におかれている端末からのアクセスが圧倒的に多いと想像される。

このような点を考えてみると、データは、量、時間的変化、地域性という特徴の故に、分散処理において分散して保持するリソースの主要なものと見なければならない。

データの有効な分散方法によって、処理ノード間のデータ交換の量を減少させ、コスト減が期待できよう。

表3－1　データベースの容量〔ＫＯＴＡ７７〕

蓄積されたデータの平均個数	3,６８１千　個
〃　　最大個数	６７,０００千　個
１個のデータの平均バイト数	1,２８１バイト
〃　　最大バイト数	５０,０００バイト
１個あたりのデータの属性の平均個数	１０８　　個
蓄積されたデータ・バンクの平均バイト数	4,０９７メガバイト

表3－2　プログラムファイルの容量

プログラムファイル	容　量（Kバイト）
実行形式システム（コンパイラなど）	１０,９００
ＦＯＲＴＲＡＮ関係	
コ ン パ イ ラ	７２０
実行時ルーチン	1,０００
ＣＯＢＯＬ関係	
コ ン パ イ ラ	６２０
実行時ルーチン	1,４００
ＰＬ／Ｉ関係	
コ ン パ イ ラ	８５０
実行時ルーチン	1,１００
リンケージエディタ	１６０

II．データ共有方式の検討

第Ⅰ部の検討では、分散処理において共有されるリソースの中で特にデータが重要であることを結論とした。さらに、分散処理においてはデータベースの共有というよりむしろデータベースをどのように分割して利用するのか、またそれを全体としてどのように統合化するかということが主要な問題であるという結論に至った。

本第Ⅱ部ではこの検討結果をもとに、分散処理におけるデータ共有方式について議論する。まず第1章では、ユーザの視点からみた分散処理技術に関して、第Ⅰ部で分類を試みた分散処理関係者の各層に対応したシステムをそれぞれ紹介する。ここでは、分散処理関係者の中で今後最も多くのユーザとなると考えられる一般ユーザに対する配慮が、市場価値を決定するという立場をとっている。第2章では、データを管理する強力な道具としてデータベース管理システム（ＤＢＭＳ）をとりあげ議論する。最初のＤＢＭＳと言われるＩＤＳ以降の歴史を振り返るとともに、既存ＤＢＭＳの代表的データベースモデルのうち、階層型モデル、網型モデル、関係モデルについて詳解する。また、ベンダが提供するＤＢＭＳとして、ＡＤＡＢＡＳ、ＳＹＳＴＥＭ２０００、ＴＯＴＡＬの３つを採りあげ、それぞれのＤＤＬやＤＭＬについて具体例を示すことにより解説を行う。さらに、ＤＢＭＳの今後の動向として、データモデルの三層化にあることを指摘する。第3章では、ネットワーク技術とデータベース技術を有効に組み合わせて実現している分散型データベースについて議論する。分散型データベースの第1の形態として、既に独立して存在している複数の情報（文献）検索システムの統合化アプローチを試みたＣＯＮＩＴシステムとＥＵＲＯＮＥＴについて紹介する。ＣＯＮＩＴシステムは既に稼働し、ＥＵＲＯＮＥＴについても１９７８年１２月を目標に稼働準備が進められており、３種の異なったシステムにおいて共通コマンドのインプリメンテーション中である。アプリケーションを限った場合には、この種のアプローチが個々のシステムへの影響度が少ないなど現実的であろう。第2の形態として、アプリケーションを限定しない関係モデル等の高位のデータモデルをベースにしたシステムを紹介する。ここで紹介するシステムは、ＩＮＧＲＥＳ、ＳＤＤ－１、ＬＡＤＤＥＲ、ＰＯＬＹＰＨＥＭＥである。ＩＮＧＲＥＳについては既に実用システムとして用いられている例も見られ、関係モデルをベースにした分散型データベースがいよいよ現実的になってきたことを述べる。

第4章では、ネットワーク上に散在するリソースを統合していく上で重要な機能の一つであるネットワーククリアリングセンターについて議論する。特に、リソースの所在源情報管理の基本的技術としてＤＤ／Ｄに焦点をあて、ＤＢＭＳとの結合形態やディレクトリの配置や管理方法について言及する。

第5章では、異機種のコンピュータから構成されたコンピュータネットワークを利用するユーザが、まず直面する問題であるジョブ制御言語の多様性を解決する技術について議論する。つまり、コンピュータの異機種性を克服するために設定した標準ジョブ制御言語の基本構想を示す。

第6章では、第5章までの検討に基づいて、異種コンピュータネットワークにおけるリソース統合問題を採り上げる。

最後に、第6章で検討した異種コンピュータネットワークにおけるリソース統合問題に関してまとめた英文資料を掲載する。

1. ユーザの視点からみた分散処理技術

第Ⅰ部第2章において、分散処理を考える上でユーザの視点が重要であることを論じた。更に、分散処理システム関係者の分類を示し、一般ユーザ（casual user）が最も重要であることを論じた。本章では、この議論を基にして各分類に対応するシステム例について議論し、今後の動向を求める。

1.1 概　　要

近年のコンピュータネットワークの発展は、広範なユーザの分散処理システムへの参加をもたらしている。ユーザの拡大と多様化に伴ないコンピュータサービス提供者（provider）は、サービスの拡大を余儀なくされてきている。特に、ユーザの大半を占める一般ユーザが分散処理システムの市場価値を決定してきていることに注目する必要がある。一方、サービス提供者は自らのリソースと市場に基づいて、提供するサービスを設計し、このための市場を開拓している。しかし、ユーザからこのサービスをみたとき、次に示す3点からその方向性と一様性の不足を指摘することができる。

- 通信機能からの制限
- メインフレームとOSのもつ特徴
- システム開発者の好み

更に、ユーザが受けようとするサービスが、論理的に同様なものであっても、システムごとに種々の異なったアクセス手続を持っている点は、ユーザにとって重要な問題である。つまり、ユーザはサービスの内容に対してよりも、サービスを受けるためのアクセス手続きに多くの労力を費さなければならなくなる。ユーザにとって必要なことは、いかに必要なリソース（サービス）へアクセスするかではなく、現在受けられるサービスは何なのかである。この問題をリソースの異種性問題と呼ぶ。他の問題は、ユーザの必要とするリソースの所在をユーザはどのように知るかである。数百～千のリソースを持ったコンピュータネットワークにおいて、ユーザの必要とするリソースがどのホストにあるかを見つけることは困難な問題である。同様な機能を持った幾つかのリソース（あるいはホスト）から最適なリソースの決定問題もこの問題に含まれる。これらの問題をリソースの分散問題と呼ぶ。

このために、ユーザと情報システム上のリソースとのインタフェースの改良が重要な問題となってきている〔PYKE74、ROSE76〕。この問題を解決するアプローチは、2つに大別することができる。

1）標準化

2）アクセス支援機能の提供

サービス提供者の技術は、現在のところ未成熟である。現時点における標準化は、各々の提供者の長所を失なわせてしまうので好ましくない〔PYKE74、ROSE76〕。本章では、2）のアクセス支援機能について議論する。

第Ⅰ部第2章において分散処理関係者の分類を試みた（図1－1）。

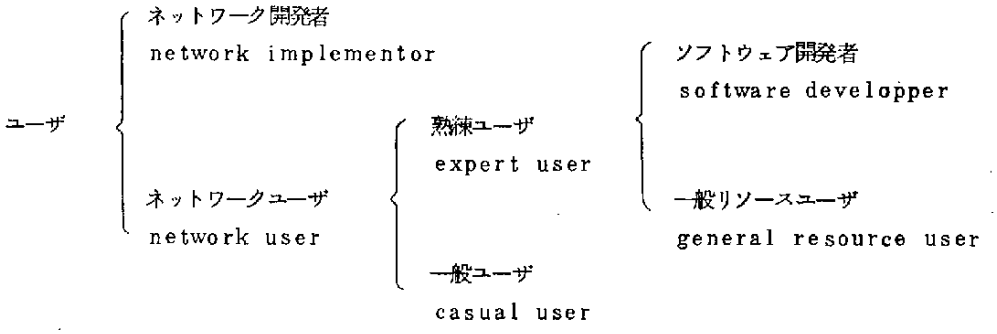


図 1 － 1　分散処理関係者の分類

ネットワークユーザは、その熟練度を基準にして熟練ユーザと一般ユーザとに分類した。さらに、熟練ユーザに関しては、ソフトウェア開発者と一般リソースユーザとに分類した。ここでの分類基準は、ネットワークに対するユーザのかかわり方、つまりリソースの利用目的によっている。ソフトウェア開発者とは、分散処理システムを利用して、ソフトウェアの開発、生産を行なおうとするユーザである。一般リソースユーザと一般ユーザは、熟練度の点で異なる。一般ユーザとは、種々の分野（物理、数学、心理学、株の仲買、座席予約業等）の人々であり、彼等の関心はサービスの内容だけである。この一般ユーザ群は、分散処理システム関係者の大半を占めている。この点において、分散処理システムの市場価値は、これらの一般ユーザによって決定されることになる。

本章では、アクセス支援機能を、上記に論じた分散処理関係者の分類に基づいて、実例を示しながら議論する。ここで対象とする関係者は、ネットワーク開発者、ネットワークユーザとしてのソフトウェア開発者と一般ユーザの三者である。

まず、1.2節では、ネットワーク開発者の立場から、ＡＲＰＡＮＥＴ、ＣＹＣＬＡＤＥＳ、ＪＩＰＮＥＴ等のネットワークに関して議論する。1.3節では、ソフトウェア開発者を対象としたＮＳＷ（National Software Works）に関して議論する。最後に、1.4節では、一般ユーザを対象にしたＲＥＸ、ＲＩＴＡ（Rand Intelligent Terminal Agency）、ＮＡＭ（Network Access Machine ）の3システムに関して議論する。

1.2　ネットワーク開発者からみた分散処理技術

1.2.1. ＡＲＰＡＮＥＴ

コンピュータネットワークの最も代表的なものがアメリカ国防省のＡＲＰＡ（Advanced Research Project Agent）がスポンサーとなって、１９６８年に開発を始めたARPANETである〔ＲＯＢＥ７０、ＭＣＱＵ７７〕。ＡＲＰＡＮＥＴは、最初の大規模、異機種分散型のパケット交換網である。

ＡＲＰＡＮＥＴの最終目標は、いかなるリソースもローカルな利用者が使うのと同じように、遠隔地から利用できるようにすることである。このようにして共用されるリソースは、ハードウェアはもちろんのこと、ソフトウェアやデータも含まれる。それまでにも、すでにタイム・シェアリング・システムによって局所的にはソフトウェアの共用が可能になっていたが、コンピュータ・ネットワークは、その局所的な制約をとりはずして、規模および距離の面での飛躍的な拡大をはたすことになった。

ＡＲＰＡＮＥＴは、１９６９年末に西海岸のＳＲＩ、ＵＣＳＢ、ＵＣＬＡ、ＵＴＡＨの４ケ所のノードおよびホストからはじまって、今では５０を超えるノードと、１００台を超えるホストからなるシステムに発展した（図１－２、１－３）。そのカバーする範囲も、アメリカ本土のみならずハワイ、イギリス、ノルウェー等にも拡張され、衛星通信まで利用している。

ＡＲＰＡＮＥＴに対する批判は数多くあるが、最初にして最大のシステムが出来あがったこと、そしてこれ以降に開発されたコンピュータネットワークに大きな影響を与えたという意味において、成功であったということができる。ＡＲＰＡＮＥＴが他のネットワークに与えた影響としては、次のものがあげられる。

(1) コンピュータ通信の手段としてのパケット交換の実現

(2) ミニコンをノードとした、高速、高信頼性の通信手段の実現

(3) プロセス間通信によるリソース共有の一般的アプローチ

(4) 階層構造を持ったプロトコルによるネットワーク作成者側からの標準化手法の実現

(5) メッセージのラウンド・トリップ時間などの一般的な標準値の設定

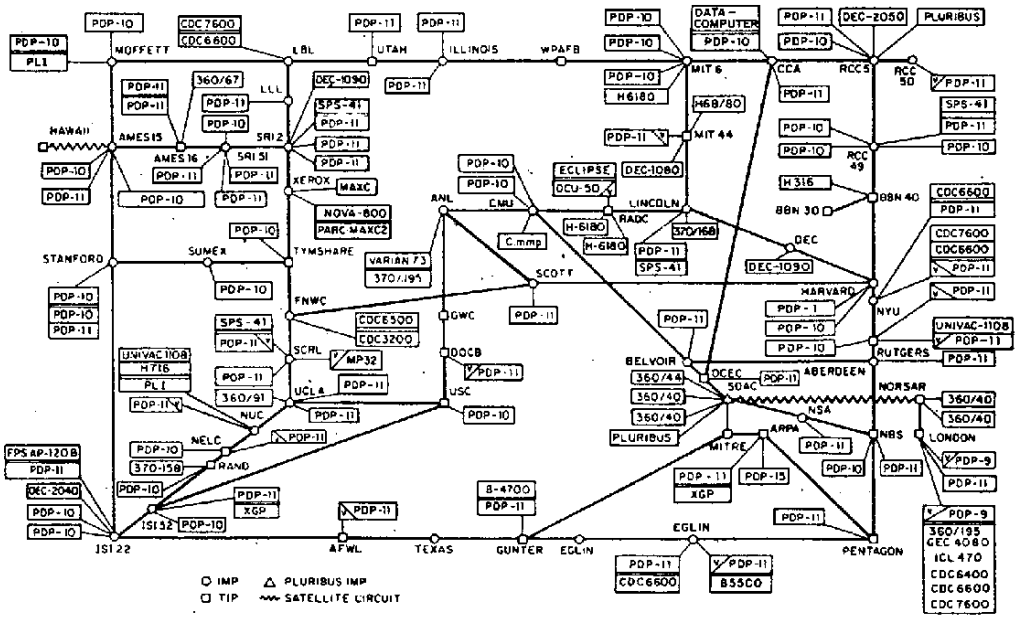


図１－２　ARPANET　logical map, March 1977〔ROBE77〕

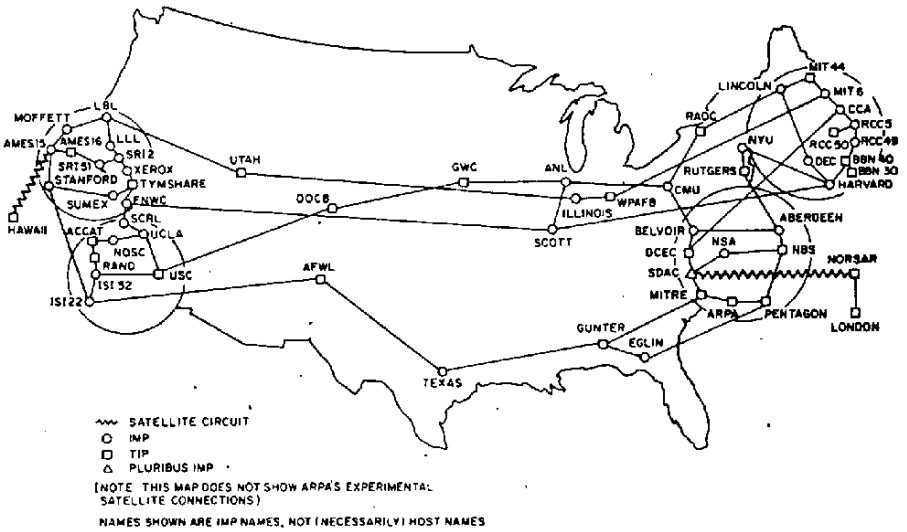


図1－3　ARPANET geographic map, April 1977〔ROBE77〕

1.2.2　CYCLADES

ARPANETの実験成果を生かして開発された、汎用異機種コンピュータネットワークの代表的なのが、フランスのIRIAを中心に進められているCYCLADESである〔POUZ75〕。CYCLADESは、政府機関である情報代表部のスポンサーかつ指導の下に、1972年に開始したもので、第一段階におけるすべてのネットワーク・センターは、研究機関や大学であった。CYCLADESの目的は、種々の新しい分野での実験道具となる原型のコンピュータネットワークになることであり、特に次の6つの分野が考えられた。

(1)　データ通信

(2)　コンピュータの相互会話

(3)　協同ジョブの研究

(4)　分散型データベース

(5)　インタフェースおよび共通言語

(6)　リソースの共有

CYCLADESの成果は、上記の6つのアプリケーションの実験が表面にあらわれたこと、パケット交換網とホストの役割を明確にしたこと。プロトコルの重要性を再認識し国際標準化への動きに対して大きな役割をはたしたこと、などがあげられる。

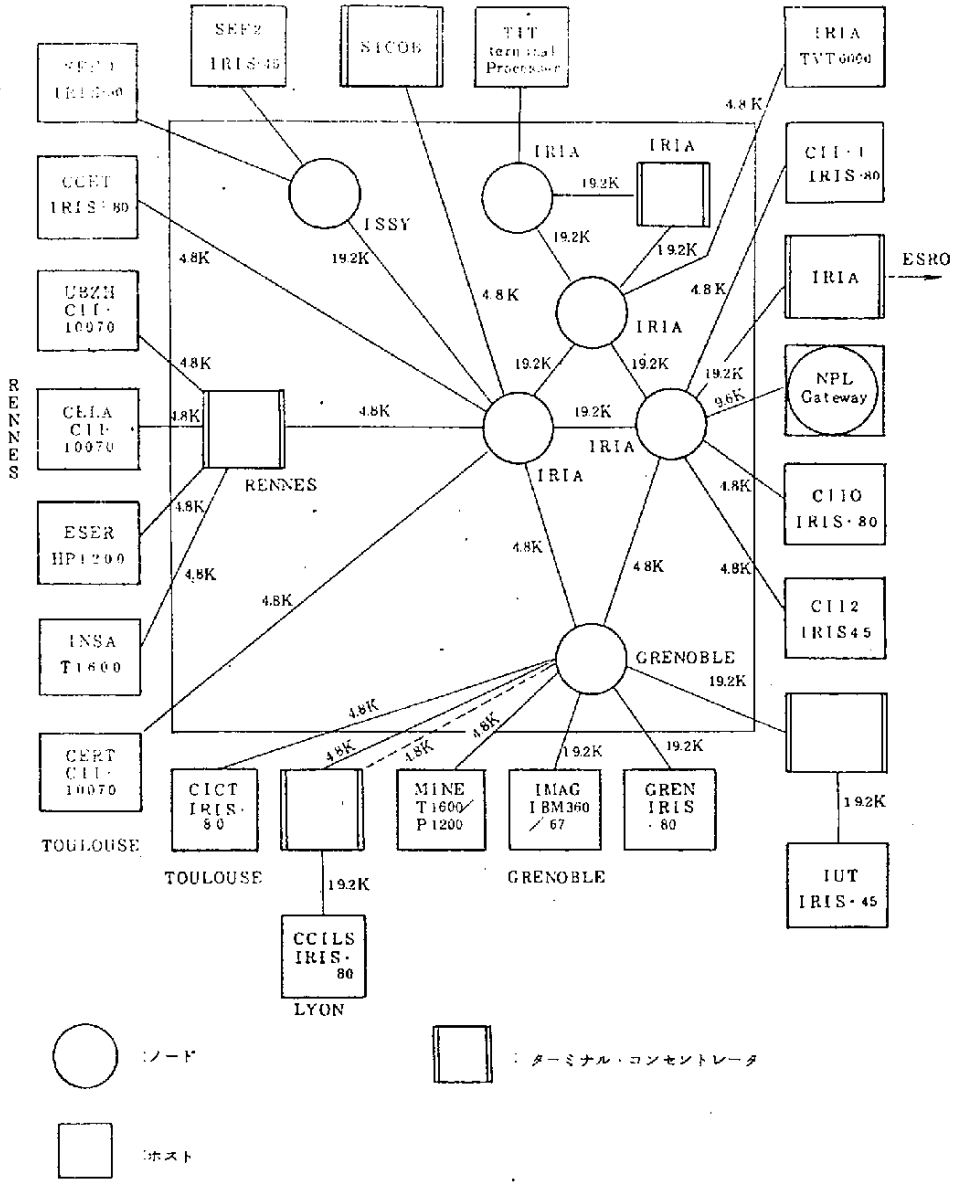


図1－4　CYCLADES構成図

1.2.3　JIPNET

JIPNET（JIPDEC　Integrated Project Network）は、（財）日本情報処理開発協会（JIPDEC）が、1973年度から4ケ年計画で開発したリソースシェアリングコンピュータネットワークである〔JIPD76〕。

JIPNETは下記に示すような実用と実験の両方の目的を有しており、ネットワークの規模は、協会内の国産3機種を結合したインハウスシステムではあるが、その構成は広域の分散型リソースシェアリングコンピュータネットワークのモデルを目指している。

先づ実用上の要求としては次の様なものをあげている。

(1) ＪＩＰＤＥＣに設置されている異なる３機種のハードウェア、ソフトウェア、データセット等の限界を除き、総合したリソースを利用したい。

(2) プログラムやデータファイルの互換性を標準化とは異なるアプローチにより確保したい。

(3) 漢字入出力、図形処理装置、マークシートリーダ等の特殊周辺装置をどのＣＰＵからも共用したい。

(4) 数年前から実施しているＴＳＳに於て、端末からどのシステムでもアクセス出来るようにしたい。

(5) 現在機種によるロードのアンバランスがかなりあるがなるべく平均化したい。

(6) ある機種のダウンによって不可能になっていた処理を他の機種でカバーする。

(7) コンピュータネットワークシステムの効率評価のデータを得る。

一方、実験的な目的としては次のようなものをあげている。

(1) 個々のシステムの、結合したが故の負荷の増大、機能や処理効率の低下などを実測する。

(2) 個々のシステムの結合前と結合後における運用上の互換性がどの程度保てるか。変更せねばならぬ点がどの程度であるかを体得する。

(3) 実用上、ユーザレベルのプロトコルとしてどの様なものが必要であるか、またそれらの処理が異なる機種間でどの程度効率よくおこなわれるか。

(4) ネットワーク利用のメリットをコスト効率という観点からとらえる。

(5) 異機種間の分散型データベースが常識的な効率下において管理出来る方式および限界を研究する。

(6) 異機種間のオートマチックなハードウェアおよびソフトウェア・シェアリングの方法および有効性などを研究する。

なお、ＪＩＰＮＥＴが利用している基本的技術および思想は、ＡＲＰＡＮＥＴのそれをそのまま利用している。

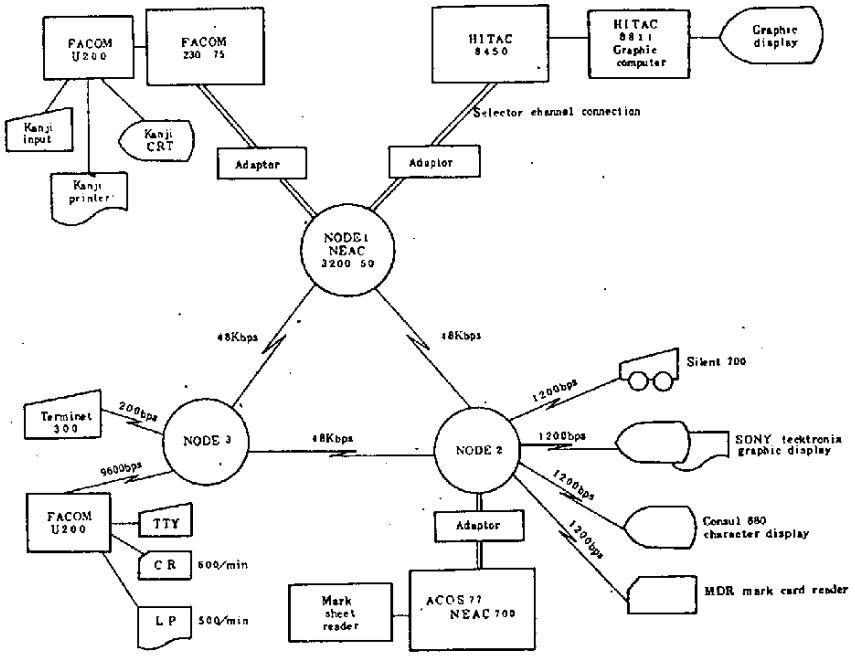


図１－５　ＪＩＰＮＥＴ構成図

1.2.4　ＮＰＬネットワーク

１９７３年７月に稼働を始めたＮＰＬネットワークは、イギリスの国立物理研究所（National Physical Laboratory ）の科学者や管理者に対して共用の通信手段を与えている〔ＳＣＡＮ７４〕。

ＮＰＬネットワークの目的は、1,５００人の人間が各サイトで利用している種々のコンピュータ間、あるいはコンピュータとテレタイプ、データ・ロガー、紙テープ穿孔機などの端末間にディジタルなデータ交換を提供することである。ＮＰＬネットワークは、手法として蓄積パケット交換を利用している。ノード（ＮＰＬネットワークではパケット・スィッチと呼ぶ）は１つであるが、コンピュータ（ユーザ・マシン）はパケット・スィッチを通じてパケットを渡たすことにより互いに交信する。端末装置はユーザ・マシンあるいはパケット・スィッチには論理的に他のユーザ・マシンと考えられるターミナル・プロセッサによってネットワークにアクセスできる。実際にはターミナル・プロセッサはパケットスィッチと同じコンピュータである。

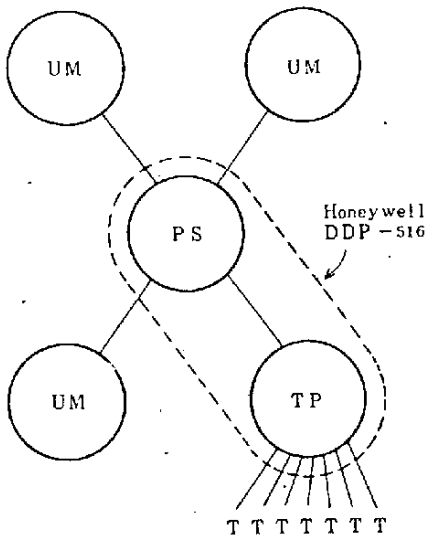


図１－６　ＮＰＬネットワークの論理形態

1.2.5 Datapac ネットワーク

Datapacは、１９７４年１０月にカナダのTrans － Canada Telephone System の Computer Communication Groupが発表したパケット交換網である〔CASH74〕。

このネットワークは、トロント、モントリオール、オタワ、カルガリーの４つの都市にノードを設置して１９７６年１１月から開始され、１９８０年までには少なくともカナダの１４都市にノードが拡張される計画になっている。

Datapac では、Datapac 1,500 およびDatapac 1,000 の２つのサービスが発表されている。Datapac 1,500 は、大量データの転送と、会話型の両者の利用を対象としたものであり、これの利用分野の主たるものとしてＰＯＳ端末や多くのキャシュ・レジスタを持つ小売店を考えている。一方、Datapac 1,000 はトランザクションのサービスをするもので、この主な利用分野は株式市況、座席予約などである。Datapac 1,000 は最初１つのノードがトロントに設置されるだけであり、このノードは信頼性を保障するためにバックアップシステムを持つスィッチング・コンピュータより構成される。

サービスされる基本的なものはデータ・グラムであり、パケット交換網としてユーザに提供する機能は次のものである。

(1) 単純なデータ・リンク制御手順によってDatapac にアクセスでき、データ・グラムの授受ができる。

(2) 授受されるデータ・グラムはシーケンシングされず、発信地で送った順序で目的に到着するとは限らない。

(3) データ・グラムが目的地に渡たされなかったならば、エラー・コードによって送信側に通知される。

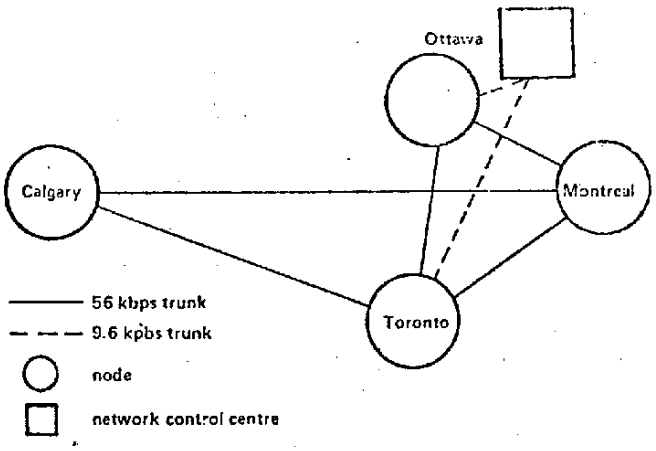


図1－7　DATAPAC　NETWORK　1976

1.3　ネットワークユーザとしてのソフトウェア開発者からみた分散処理技術

ここでは、ネットワークユーザとしてのソフトウェア開発者が使用するのに適した分散処理技術NSW（National Software Works）を紹介する〔WHIT75、SCHA76〕。

(1) NSW誕生の背景

ソフトウェアの開発は、依然としてほとんど自動化されていない労働集約的産業である。従って自動化によって、コスト効率のよいソフトウェア開発を行なう必要がある。現在、多くのソフトウェア開発のためのツール（tool）が作られている。しかし、これらのツールには、次に示すような問題点がある。

- いろいろなユーザで、重複してツールを持っている。
- ツールを設置している場所のユーザしか使えない。
- 開発に最適なコンピュータを使うことができない。
- 最新のソフトウェアの配布が困難

これらに対処する方法として、ネットワークを使うことが考えられる。ネットワークを使用すると、ツールを分散させて各サイトに置くことができる。そのため、いろいろなユーザで重複してツールを持つ必要がなくなる。

また、ネットワーク内の最適のコンピュータを開発のために使うことができる。たとえば、シンタクス・チェックはバロースのB6700、テスト・データ、ジェネレータはIBM370、最終的なソフトウェアのテストは、バロースのB4700で行うというように、それぞれの処理をその処理に適したコンピュータに行なわせることができ、コストを低減することができる。ま

た、最新のソフトウェアの配布も、ネットワークを用いれば、容易に行なうことができる。

しかし、ネットワークを用いようとした場合に、多くのユーザにとって障害となることがいくつかある。ある処理を行なうために何台かのコンピュータを使用しようとした場合には、それぞれのコンピュータが存在するサイトに、ユーザ自身を登録しなければならない。また、ユーザは使用するそれぞれのコンピュータのログインの方法も知らねばならないし、それぞれのコンピュータに対してのアカウントの問題もある。

これらネットワークの使用に際しての問題を解決して、ソフトウェアの設計、開発、テスト、ドキュメンテーションを促進するための方法がＮＳＷである。

ＮＳＷは、ネットワークをベースにしたソフトウェア開発のためのツールを提供している。ＮＳＷプロジェクトは、ＡＦＤＡＡとＡＲＰＡによって設立されたもので、ソフトウェアの開発コストを低減させるために、異機種間の問題に取り組んでいる。

(2) ＮＳＷの目的

ＮＳＷの目的を下記のように、まとめることができる。

1）ソフトウェアの設計、開発、テスト、ドキュメンテーションを促進するための開発用ツールを開発者に提供する。

2）ネットワークを使用することにより、ソフトウェア開発用ツールを各ホスト間に分散させて共有する。

3）ネットワーク上のソフトウェア開発用ツールに対する共通インタフェースを、開発者に提供する。

(3) ＮＳＷの構成

以上のような目的をもったＮＳＷは、３つの部分から構成されている。

そのうちの１つは、ＴＢＨ（"Tool Bearing Host" machine ）である。これによって、それぞれのソフトウェア開発用ツールを、最も適切なＴＢＨ上で利用することができる。ＴＢＨシステムは、ＮＳＷをサポートするために必要である特別なソフトウェアを持つ一般的なシステムである。このようなシステムは、現在、ＢＢＮ、カルフォルニア大学、ＭＩＴにおいて開発中である。

２番目の部分は、フロントエンド・システムをサポートしている端末である。これは、システムのユーザ側にある。このフロントエンドの開発は、ＳＲＩによって始められたものである。

最後の部分は、前に述べた２つの部分を結合する部分であり、ワークスマネージャ（Works Manager ）と呼ばれる中央制御のためのソフトウェアである。実際には、ワークスマネージャは信頼性向上の目的で、分散システムになっている。ワークスマネージャの仕事は、全てのリソースに対する共通インタフェースの提供に関する管理が主である。リソースがどこに存在するかが判明すると、適切なスケジューリング機能も提供する。

(4) ＮＳＷの課題

ここでは、ＮＳＷをネットワークユーザとしてのソフトウェア開発者が使用するのに適した分

散処理技術としてとらえている。

現状において、ＮＳＷはＰＤＰ－１０上に、ソフトウェア開発と、その保守をソフトウェア開発者に補助するための改良されたアクセスを提供している程度のものがインプリメントされている。また近い将来、ＰＤＰ－１１のようなミニ・コンピュータ上にも実現されるとも言われている〔ＲＯＳＥ７６〕。

ＮＳＷをインプリメントする上での問題点として、アカウンティング、アクセス、コントロール、ファイル管理などに関連した組織上解決すべきいくつかの問題があげられている〔ＫＩＭＢ７５〕。

1.4　一般ユーザからみた分散処理技術

本節では、一般ユーザ（casual user）からみたアクセス支援機能として、ＲＥＸ、ＲＩＴＡ、ＮＡＭについて議論する。ここで論じる各システムの特徴は、コンピュータの初心者（computer naive user）であるような一般ユーザを対象としている点である。各システムは、端末側のユーザと１つ以上のネットワーク上のリソースとの間のインタフェースとして存在する。

1.4.1　ＲＥＸ

ＭＩＴＲＥ社において開発されているリソースに関する所在情報や獲得方法などの情報を提供するシステムＲＥＸは、ＡＲＰＡＮＥＴ上のリソースの使用に関して、初心者ユーザを支援するためのシステムである〔ＢＥＮＯ７４、７３〕。

ＡＲＰＡＮＥＴのユーザにとって、ＦＯＲＴＲＡＮのような言語プロセッサを使用としたとき、いくつかの困難さに直面していた。つまり、どのホスト・コンピュータで、どのリソースを使用できるかを知ることの困難さである。又、必要なリソースを見つけられたとしても、そのリソースが存在しているホスト・システムとリソースの使用法を知ることは、さらに困難である。すでに、ＡＲＰＡＮＥＴでは、ＮＩＣ（Network Information Center）から、"ＡＲＰＡＮＥＴリソースハンドブック"〔ＥＬＩＺ７６〕が、オンラインと印刷物の両方で入手可能となっている。しかし、ユーザへとって満足なものではなく、本格的なリソース管理技術の発展が必要であった〔ＲＯＳＥ７６〕。

ＡＲＰＡＮＥＴ内でも、こうしたユーザの困難さに対する認識が高まってきた結果、ユーザ・サービスの開発に努力が向けられるようになった。ユーザの問題を解決する指針となった"自動リソース管理"は、リソースの共有を容易にしかつ自動化する技術である。なお、ＡＲＰＡＮＥＴにおけるリソースの意味は、ハードウェアを始めソフトウェア、データベース、システムソフトウェア、リソースの使用に関するエキスパートも含んでいる〔ＢＥＮＯ７４〕。

ここで、ユーザからみえるリソースを一般リソース（generic resource）と呼ぶことにする。例えば、ＦＯＲＴＲＡＮとは、一般リソースである。つまり個々のコンパイラが、Ｆ４、ＦＯＲＴＲＡＮ Ⅳ、ＦＯＲＴＲＡＮ－Ｈであるかは関係ない。一般リソースを共有するための機

能としてBenoitらは下記のものをあげている〔ＢＥＮＯ７４〕。

1）均一なシステムアクセス

2）ヘルプ機能の標準化

3）均一なリソースの呼び出し

4）一般リソース管理 …… リソースの管理機能としては、下記のものが必要である。

ⅰ）リソースが必要とするローカル・データとプログラムの決定

ⅱ）リソースを利用するジョブの実行の監視

ⅲ）エラー・リターンをかえし、可能な回復を行なう

ⅳ）会計情報を持ったジョブ統計情報の集積

ⅴ）結果をユーザへかえす

5）全ネットワーク的なアカウンティング

現在の異機種ネットワークでは、各ホストにおいて前記のような均一性をユーザに対して持つことは不可能である。つまり、各ホスト・コンピュータは、各々異なったコマンド言語、ファイル・システム、ログ・オン手続、アカウンティング等を持っている。従って、近い将来に上記の機能を達成する現実的方法は、各サイトのオペレーティング・システム、リソース、そして今後開発されるネットワーク標準との間の翻訳機能を、ネットワーク管理者内に開発することである。

次にＲＥＸシステムについて述べる。ＲＥＸシステムは、リソース管理者の基本機能の一部についての実験システムである。ＲＥＸは、リソース情報システムとしての即時的有用性を持っているが、これの主要な利用は、ネットワーク全体のリソース管理機能の構築のための種々の技術の実験である。

ＲＥＸが現在提供している機能を把握するために、コマンドの使用例を下記に示す。

1）ＦＩＮＤコマンド

このコマンドは、リソース又は、必要とするリソースの集合が存在する場所を見つける。

例えば、ＦＩＮＤ ＦＯＲＴＲＡＮ ＡＮＤ ＮＯＴ ＰＤＰ－１０は、ＦＯＲＴＲＡＮコンパイラを持つＰＤＰ－１０ではないホストの一覧をリストする。ＦＩＮＤコマンドは、ＦＩＮＤ ＜site － expression＞のフォーマットを持つ。＜site － expression＞は、リソース名又はホスト名のブール結合である。

2）ＤＥＳＣＲＩＢＥコマンド

このコマンドは、リソースのカテゴリ又は集合についての情報と、リソースについての集合に関する情報を得るために使用される。

例えば、ＤＥＳＣＲＩＢＥ ＦＯＲＴＲＡＮ ＣＯＢＯＬ ＡＮＤ ＳＮＯＢＯＬ は、ＦＯＲＴＲＡＮ、ＣＯＢＯＬ、ＳＮＯＢＯＬの各々に関する情報を生成する。このコマンドは、下記のようなフォーマットを持つ。

$$\mathrm{DESCRIBE}\left\{\begin{array}{l}\langle \mathrm{attribute\ FOR\ resource}\rangle \\ \langle \mathrm{resource\text{-}expression}\rangle\end{array}\right\}\left[\mathrm{AT}\ \langle \mathrm{site\text{-}expression}\rangle\right]$$

ここで＜resource－expression ＞は、リソース、カテゴリ名、ホスト名のリストである。下図に使用例を示す。

```
FIND IMS1
THE FOLLOWING HOST HAVE THE REQUESTED RESOURCE:
UCSB-MOD75 IS HOST NO. 3
DESCRIBE COST FOR IMS1 AT UCSB
AT SITE UCSB,
    COST: CONNECT-TIME: $5/HR, CPU-TIME: $8/MIN,
          FILE STORAGE: $5/VOLUME/MON
```

3）HELPコマンド

REXシステム自身に対する援助機能である。"？"の入力によって、コマンド一覧と各コマンドの使用法に関する簡単な説明が出力される。

4）QUITコマンド

このコマンドによってREXシステムから抜け出る。

5）ACQUIREコマンド

将来のREXの機能としてのリソース獲得機能である。

ACQUIRE機能は、ユーザが要求したリソースを自動的に獲得するためのものである。リソースの獲得に際しては、ユーザによる種々の手続きを必要としない。このための必要なステップとしては下記のものがある。

i）要求を満足する全てのホストの識別

ii）要求を満足するホストの1つを選択

iii）選択されたホストへのログオンと会計手続の決定

iv）必要なサブシステムの呼び出し

v）リソース使用の監視と終了

このコマンドの基本形式は、

ACQUIRE＜resource＞〔WITH＜attribute＞=value＞
〔AT＜site expression＞〕

である。

現在の初期バージョンでは、ACQUIREコマンドは指導用のヘルプ機能サービスに対してのみに制限されているが、ACQUIREコマンドの本格的なインプリメンテーションが検討されている。

1.4.2 RITA

RITA（Rand Intelligent Terminal Agent）〔ANDE75、76a、b〕は、PDP11／45－UNIXオペレーティングシステム上で稼働するコンピュータプログラムで、

RAND社により開発された。

ＲＩＴＡのシステム構成は、次のものから成り立つ〔ＡＮＤＥ７６ｂ〕。

1）核（Kernel） …… モニタ、データ・ベース、規則

2）フロントエンド …… テキスト処理、ＲＩＴＡユーザ・コマンド処理

ＲＩＴＡの開発に際して、下記のような点に関する考察があった。

1）多種多様なユーザの情報システムへの参加と、それに対するサービスの増加。

2）コンピュータ・ハードウェア技術の進歩によるコスト低下と、マイクロ・コンピュータ技術の進歩。このことによって、現在のミニコンと同程度の処理能力（プロセッサとメモリ）を有するインテリジェント端末が、数年内にコスト的に可能となるだろう。

3）端末が、ある程度のメモリと処理能力を持つことによって、テキスト編集、ヘルプ機能、入力コマンドとデータのチェック、ユーザの訓練等を行なうことができる。

4）端末が、こうした処理能力を持つことによって、外部情報システム（ex. ARPA ）とのインタフェースと成り得る。ユーザの仕事と直接関係のないプロトコル等の処理を端末は行なうことが出来る。

5）ユーザ・エージェント（ user agent ）を、ユーザ自身が定義し、動作させることが出来る。このユーザ・エージェントの機能としては下記のものがある。

ⅰ）事象のカレンダを見て、指定された時間にサービスを自動的に開始する。

ⅱ）種々のイベントの発生の監視をする。

ⅲ）端末間通信を行なう。

ⅳ）ネットワーク上に分散したサービス間のトランザクションの管理と終了（正常かどうか）の監視をする。

以上に論じた点は、現在のミニコンを用いて、インテリジェント端末を開発する強力な根拠となっている。ミニコンを用いる理由は、2）で論じたように、近い将来、現在のミニコンと同程度の能力を持った端末が出現すると思われるからである。

以上の議論に基づいて、ＲＩＴＡへ対する設計要求をまとめると下記のようになる〔ＡＮＤＥ７６ｂ〕。

1）端末ユーザの多くは、コンピュータの初心者つまり、一般ユーザである。一般ユーザが、容易に仕事が出来るように、各々のアプリケーション分野に適した基本システムを提供する必要がある。基本システムは、ユーザの要求によって、アクションを実行し、その動作をユーザへ説明出来ねばならない。

2）エージェントの機能の改良と拡張は、ユーザが言語を使用することによって行なわれる。しかし、既存のプログラミング言語は、下記の点で不適である。

ⅰ）初心者にとって、理解しにくい。

ⅱ）言語は、アルゴリズムに基づいている。端末ユーザは、アルゴリズムによってよりも、発見法（ heuristics ）によって端末を動作させようとする。

iii）プログラミング言語内の入れ子を持った制御構造は、ユーザプログラミングの主要なエラー原因となっている〔ＭＩＬＬ７５〕。

3）エージェントは、ユーザと外部情報システムとの両方に対して通信出来ること。

4）ミニコン上で動作出来ること。

5）仕事、スケジュール、期限等に関する履歴を持つことが出来ること。

6）エージェントは、検索、編集、テキスト格納の機能を持つこと。

7）エージェントは、約束（appointment）、スケジュール等の時間に関する仕事を処理出来ること。この時、エージェントは、カレンダと時間とを扱える。

ＲＩＴＡは、上記の設計要求を満足するものとして、プロダクションシステムを採用した。プロダクションシステムは、モニタの制御下でデータベースに基づいて機能あるプロダクション規則"条件（condition）－アクション（action）"（いわゆるＩＦ－ＴＨＥＮ－）の集合から成り立っている。個々の規則は英語形式によって書かれている。

規則とデータベースの形式（form）は、初心者にも容易に理解出来るようなものである。各々のプロダクション規則は、エージェントの動作を導くものとして発見法を用いている。下記に規則の例を示す。

Rule 1
IF: there is a message whose status is "awaiting action" and the identification-field of the message is not in the action-items of the user.
THEN: put the identification-field of the message into the action-items of the user;

Rule 2
IF: the latest-command of the user is "show action items" and the state of the system is "command unfulfilled"
THEN: send the action-items of the user to the user and set the state of the system to "command fulfilled".

モニタは、パターン導入（pattern－directed）と目標導入（goal－directed）の２つのモードで動作出来る。前者においては、プロダクション規則の条件部のテストを行ない、条件が満足されれば、そのアクション部を実行するような動作モードである。一方、後者は、プログラムの動作が、ある目標を達成しようとする試みによって制御されるような動作モードである。この時、目標を達成出来る一連の基本タスクへ到達するまで、必要な副目標を生成しながら仕事を進める。

ここで、プロダクションシステムの利点／欠点について整理してみる〔ＡＮＤＥ７６ａ、ｂ〕。

まず利点としては次の４項目があげられる。

1）ユーザに対する説明能力が高い。

2）入れ子を持たないために、言語の制御構造が簡単である。

3）ユーザが新たに定義することによって、知識と発見法を増殖することが可能である。

4）ユーザに対する教育性と学習性が高い。

また欠点として次の２項目が上げられる。

1）目標導入モードにおいては、プロダクション規則の形式で、オペレーションをコーディングすることは困難である。

2）プロダクション・システムは、無用なチェインの生成等を起こすために、有効に動作しなくなる場合がある。

現在ユーザエージェントは約５０の規則を持っており、この規則によって、RITAは、ARPANETのＦＴＰオペレーションと、ニューヨークタイムズのインフォメーションバンクとの交信が可能となっている。またＲＩＴＡは現在開発中のシステムであり、未解決な問題としては下記の項目がある。

1）一般ユーザが、規則の追加と改良をすることによって、ユーザ・エージェントを実際に動作出来るか。

2）動作の速度と有効性は、ユーザ・エージェントが増大し、複雑化した時、適当なものとなるか。

3）システムの機密保護の問題はどうか。パスワード、アクセス・キー、アカウント番号、データフォーマットのような知識を、ＲＩＴＡ内に持たせることによって、機密保護の問題への１つの解になるだろうか。

1.4.3 NAM

ＮＡＭ（Network Access Machine ）〔ＲＯＳＥ７４、７５、７６〕は、ミニコンピュータベースの装置で、ユーザの入力したコマンドを、あるネットワーク又はネットワーク上のホストコンピュータにおいて実行可能なコマンドへ展開するもので、ＮＢＳで開発されている。

多重異機種コンピュータネットワークにおいて現在まで行なわれた実験では、論理的に同様なリソースへ対しても種々の異なったアクセス手続きが存在している。例えば、ログオン、ログオフ、コネクション開設、コネクション接断等の手順は、各ホストにおいて異なっている。この問題の解答としては、アクセス手続きの標準化とアクセス支援機能の提供の２点が存在することを1.1項で述べた。現在のところネットワークに対する標準アクセス手続きは存在していない。

異機種ネットワーク上における標準化のアプローチとして、共通アクセス言語が考えられる。これは、ホスト間の通信プロトコル機能の拡張として考えられる。つまり、ネットワーク上のホストは、共通アクセス言語としての高級プロトコルをサポートする〔ＲＯＳＥ７４〕。ホスト内に、ユーザ・インタフェースを設けるアプローチである。一方、ＮＡＭはユーザにアクセス支援機能を提供している。つまり、ユーザとリソース間のインタフェースを改良するアプローチをとっている。

従って、ＮＡＭはユーザとネットワークとの間に設置され、リソースに対する適当なアクセス手続きを生成するミニ・コンピュータであると言える。ＮＡＭは、種々のリソースへのアクセス手続きが容易に参照出来るファイル・ディレクトリを持っている。またユーザは、アクセス手続きを自ら定義することができる。この時、ファイル・ディレクトリへの参照は、ユーザが定義したアクセス言語の基礎となる。一連のアクセス手続きはマクロ形式をとる。ＮＡＭは、このマクロを各リソース上で実行出来るダイアログへと展開する。ダイアログは、遠隔システムへ送信されるメッセージと、このメッセージに対する期待応答（ expected response ）とから成る。実応答と、この期待応答とは比較され、リソースへのアクセスが正常に処理されたことを確かめる。図１－８に、この関係を示す。

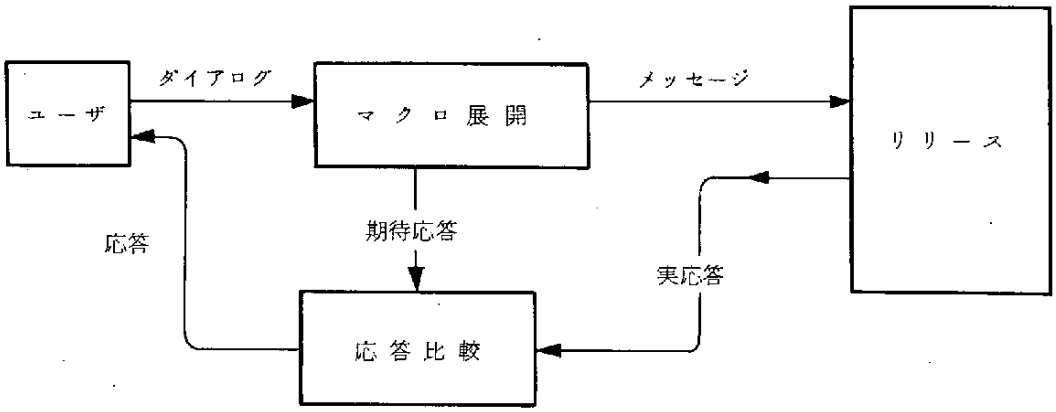


図 １ － ８　　ＮＡＭの構成

上記に述べた点は、ＮＡＭの基本機能である。現在は、ユーザ・コマンドの条件とパラメータによる拡張機能が追加されている。この機能によって、ユーザは、同一のコマンドを用いて異なったホストとネットワーク上のリソースへアクセス出来る〔ＲＯＳＥ７５〕。

現在拡張中の機能としては、あるアプリケーション分野での特別なコンピュータ利用に対して、期待応答時間を予想しようとしているものがある〔ＲＯＳＥ７６〕。

他の魅力ある機能は、ネットワーク全般の個別指導サービス（ tutorial service ）である。このためＮＡＭはユーザプロファイルを持っている。プロファイルには、ＮＡＭのユーザが使用しようとするコンピュータに関して、現在の利用法、内部インディケータの使用法、ヘルプ支援インディケータなどに関するデータが維持されている。ＮＡＭは、このデータベースを用いてホストに代行してユーザを支援する〔ＲＯＳＥ７６〕。

次に、ＮＡＭの実際の使用例を示す。図１－９は、ＢＢＮのＰＤＰ－１０ＴＥＮＥＸへのログ・オンを示している。下線は、ＮＡＭによって、遠隔システムへ送られるストリングを示している。

Macro Directives	NAM Sends/Receives
.MACRO LOGIN HOST	
.TERM 2 '[CR LF]'	
.MATCH 'HELLO'	
.SEND 'E'	E↓
	HELLO 321↓
.TERM '[CR LF]@ '	
.MATCH 'OPEN'	
.SEND ' @L HOST'	@ L 69↓
	LOGGER
	OPEN
	BBN - TENEX↓
	@
.MATCH 'PREVIOUS'	
.SEND 'LOG ROSENTHAL RMR 1[CR]'	LOG ROSENTHAL RMR 1↓
	JOB 24 ON TTY 16
	PREVIOUS LOGIN:
	1-AUG-74↓
	@
.ENDMACRO	

図 1 － 9　ＮＡＭの使用例

1.5 まとめ

本章では、ユーザの視点からみた分散処理技術 － 主にＡＲＰＡＮＥＴにおけるネットワーク技術 － を議論した。この章における第一の主張は、一般ユーザに対していかなるサービスを分散処理システムが行なうかに、その市場価値は依存するという点である。この背景には、近年のハードウェア技術と、ＡＲＰＡＮＥＴを中心とするネットワーク技術の発展によるユーザ、通信、リソース間のコストの相対的変化をみることが出来る〔第Ｉ部参照〕。このような技術的発展は、分散処理システムのユーザ層へも変化をもたらした。従来のコンピュータの専門家と称する層と、座席予約システムなどにおける単なるオペレータから、種々の分野、種々の階層のユーザの分散処理システムへの参加をもたらした。これらのユーザ群は、分散処理システムから、如何なるサービスを受けられるかに対してのみ感心を示す。つまり、自分の必要なサービスは、どこにあり、いかにして受けるかにはあまり関心を示さない。ユーザの大半を占めてゆくと思われるこれらのユーザを"一般ユーザ（ casual user ）"として定義した。分散処理システムのサービス提供者は、一般ユーザに対して、統一的で一様なサービスを提供しなければならない。なぜならサービス提供者としての市場価値は、一般ユーザに最も依存しているからである。

しかし、現在の分散処理システム、特にネットワーク技術を視た時、その方向性と一様性の不足を指摘せねばならない。これは、ユーザにとって大きな問題点である。この問題解決へのアプローチには、２つある。１つは、アクセス手続きの標準化であり、他はユーザとシステム上のリソースとの間のインタフェースの改良、つまりアクセス支援機能をユーザに提供する方法である。標準化によるアプローチは、分散処理技術の未成熟段階では望ましくないと思われる。したがって、本章では後者のアプローチを議論することにした。

ユーザ・インタフェースを考える時、その機能をどこにおくかが問題となる。ホスト内に設ける

か、あるいは専用のミニコンピュータ又はインテリジェント端末を設置するかの問題である。ここでは、前者のコスト的問題、またこれらの機能を設けることにより生じるパフォーマンスの低下等の問題から、後者を考えることにした。

1.1節で示した分散処理システム関係者の分類のなかで、ネットワーク開発者、ネットワークユーザとしてのソフトウェア開発者および一般ユーザを対象としたシステム例を議論した。ネットワーク開発者としては、ＡＲＰＡＮＥＴ、ＣＹＣＬＡＤＥＳ、ＪＩＰＮＥＴ等の既存コンピュータ・ネットワークを議論した。

ソフトウェア開発者としては、ＮＳＷ（National Software Wor　）をとりあげた。NSWは、ソフトウェア生産をコスト的に有効に行うために、ネットワーク上に分散した開発用ツールを、ソフトウェア生産者に使用させようとするものである。一般ユーザとしては、ＲＥＸ、ＲＩＴＡ、ＮＡＭの３つをとりあげた。ＲＥＸはネットワークリソース管理者を目指すものである。

ユーザは、リソースを一般リソース（gereric resource ）として把握している。これに対して、ＲＥＸは、一般リソースの概念とネットワーク上に存在する個々のリソースとのインターフェースとして存在する。現在のＲＥＸの機能は、必要なリソースの存在場所、使用方法等に関する情報を与えるにすぎない。リソースの獲得機能は、まだ備えていないが早期の開発に対する期待は大きい。

ＲＩＴＡは、インテリジェント端末であることが、第１の特徴である。このことの背景には、ハードウェア技術の発展により現在のミニコンと同程度のメモリと処理能力を持った端末が数年内に可能となるという点がある。人工知能におけるプロダクション規則を採用することによって、初心者のユーザ（一般ユーザ）への理解の容易さと、ユーザ自身に、自らの必要とするシステムの定義を可能としようとしている。

ＮＡＭは、各リソースへのアクセス手続きの相違に対して、共通のインタフェースを提供しようとするものである。ＰＤＰ－１１を用いたＮＡＭは、ユーザの入力したコマンド（共通コマンド）を、目的地のリソースが理解出来る形式へ展開する。このコマンドは、ユーザ自身によっても定義出来る。

この章における議論の結論は、下記のとうりである。

1）分散処理システムにおいて、一般ユーザは、今後重要な位置を占めてくる。

2）分散処理システムは、これらのユーザに対して一様な方向性を持ったサービスを提供しなければならない。

　このようなサービスを提供するためには、リソースの異種性問題と分散問題とのユーザに対する不可視性を提供する必要がある。

3）リソースへのアクセス手続きの標準化は、理想的であるが、近い将来の標準化は不可能である。現時点における標準化は、分散処理技術に対して、その技術の未成熟さの故に好ましくない。

4）ユーザと、分散処理システム上のリソースとの間のインタフェースの改良が、近い将来最も有効な方法となる。現在の多くの試みは、ＲＥＸ、ＲＩＴＡ、ＮＡＭを始めとして、この点に労力が集中されている。

2. データベース管理システム

第Ⅰ部第3章において、分散処理を構成する要素として特にデータが重要であることを認識するに至った。さらに、第Ⅱ部第1章においては、ユーザの視点からみた分散処理技術について、分散処理システム関係者の分類に従い、それぞれに対応した幾つかのシステムを引用し検討を行なった。

本章では、分散処理におけるキーがデータにあるという立場から、DBMS（Date Base Management System：データベース管理システム）について調査および分析を行なう。まず、2.1.では、ＤＢＭＳ出現の背景と最初のＤＢＭＳと言われているＩＤＳ（ＧＥ社１９６４年）以降の歴史を振り返る。2.2では、既存ＤＢＭＳのデータ構造を、階層型モデル、網型モデル、関係モデルに三分類し、それぞれについて解説を行なう。2.3では、ベンダが提供する代表的ＤＢＭＳとして、ＡＤＡＢＡＳ（西独Software AG社）、ＳＹＳＴＥＭ２０００（米国ＭＲＩ社）、ＴＯＴＡＬ（米国Cincom社）の三つを紹介する。最後の2.4.では、ＤＢＭＳのデータベースモデルの動向として、ＡＮＳＩ／Ｘ３／ＳＰＡＲＣスタディグループやＣＯＤＡＳＹＬの新提案にみられるような三層スキーマの概念が主流であることを述べる。

2.1 ＤＢＭＳ出現の背景と歴史

データベースとは、「ある企業体（enterprise）で用いられる種々の応用システム（業務）で利用できるように格納された運用データの収集」である〔ＤＡＴＥ７７〕。ここでいう企業体とは、製造会社や銀行、病院、大学、政府関係機関など広い意味をもっている。従って、運用データとはそれぞれの企業体により内容が異なり、製品や預貯金、患者、学生、計画などに関するデータを指している。そして、ＤＢＭＳとは、データの独立性やデータの保全性、データの機密保護、データの共有、ＤＤＬ（Data Description Language：データ記述言語）、ＤＭＬ（Data Manipulation Language：データ操作言語）などの諸条件を満たすために、データベースを管理運営するためのシステムといえる。

このようなＤＢＭＳを実現させた背景として、情報処理に係わる業務の多種多様化やデータ量の著しい増加などを指摘できる。個々の業務単位で処理するという、従来の伝統的なファイルシステムの概念では、このようなニーズに対応しきれなくなった。このため、データの共有化や統合化などの新しい概念が誕生した。これが、ＤＢＭＳの最も重要な概念の一つであるデータの独立性に継がって行くのである。

ＤＢＭＳとして初めて体系化されたシステムは、ＧＥ社（現ＨＩＳ社）が１９６４年に開発したＩＤＳ（Integrated Data Store）と言われている。このＩＤＳは、ＣＯＤＡＳＹＬ／ＤＢＴＧ（Conference on Data Systems Languages／Data Base Task Group：データシステム言語協議会／データベース作業班）が１９６９年と１９７１年に発表した二つの報告書に強い影響を与えた。特に、１９７１年の報告書〔ＣＯＤＡ７１〕はＤＢＴＧレポートとかＤＢＴＧ'７１などとよばれており、色々な意味で注目を浴びた。

ＤＢＭＳは過去多くの開発がなされた。世界的に一応名の通ったＤＢＭＳ数は、８０以上に達しているが、今日実際に使用されているＤＢＭＳは２０程度にすぎない〔ＨＯＴＡ　７７ｂ〕。ＤＢＭＳの分類方法には、視点により幾つかあるが、一般には次の二種類がとられている。

・親言語方式か独立言語方式かによる分類

・データ構造の表現方法による分類

今日迄のＤＢＭＳの発展を振り返るとき、三つの時期に大別して把えることができる。〔ＦＲＹ７６〕。

・初期の段階（１９６４年以前）

企業やコンピュータメーカより、政府、特に軍や情報関係（ＣＩＡなど）が原動力になっていた。

・ファミリの編成（１９６４～１９６８年）

孤立した発展は減少し、ＤＢＭＳの大きなファミリが、一つの企業あるいは政府省庁に限ることなく組織化が進んだ。

・ベンダ／ＣＯＤＡＳＹＬの発展（１９６８～現在）

占有のサービス会社による開発と、ＣＯＤＡＳＹＬのＤＢＴＧから出された勧告に基づいて発展してきている。

ＤＢＭＳという新しいデータ管理システムの誕生により、情報処理はプログラムをデータから分離することに成功し、処理中心の考え方からデータ中心の考え方に移行したのである。

図２－１に代表的なＤＢＭＳの発展過程を系列別に示す。また、表２－１は、商用ＤＢＭＳを中心に整理したもので、ＤＢＭＳ名、開発機関（メーカ、ベンダ名）、開発年、稼働機種を示す。なお、階層型、網型（ＣＯＤＡＳＹＬ型以外の網型を含む）、その他（関係モデル中心）の三分類を行なっている。

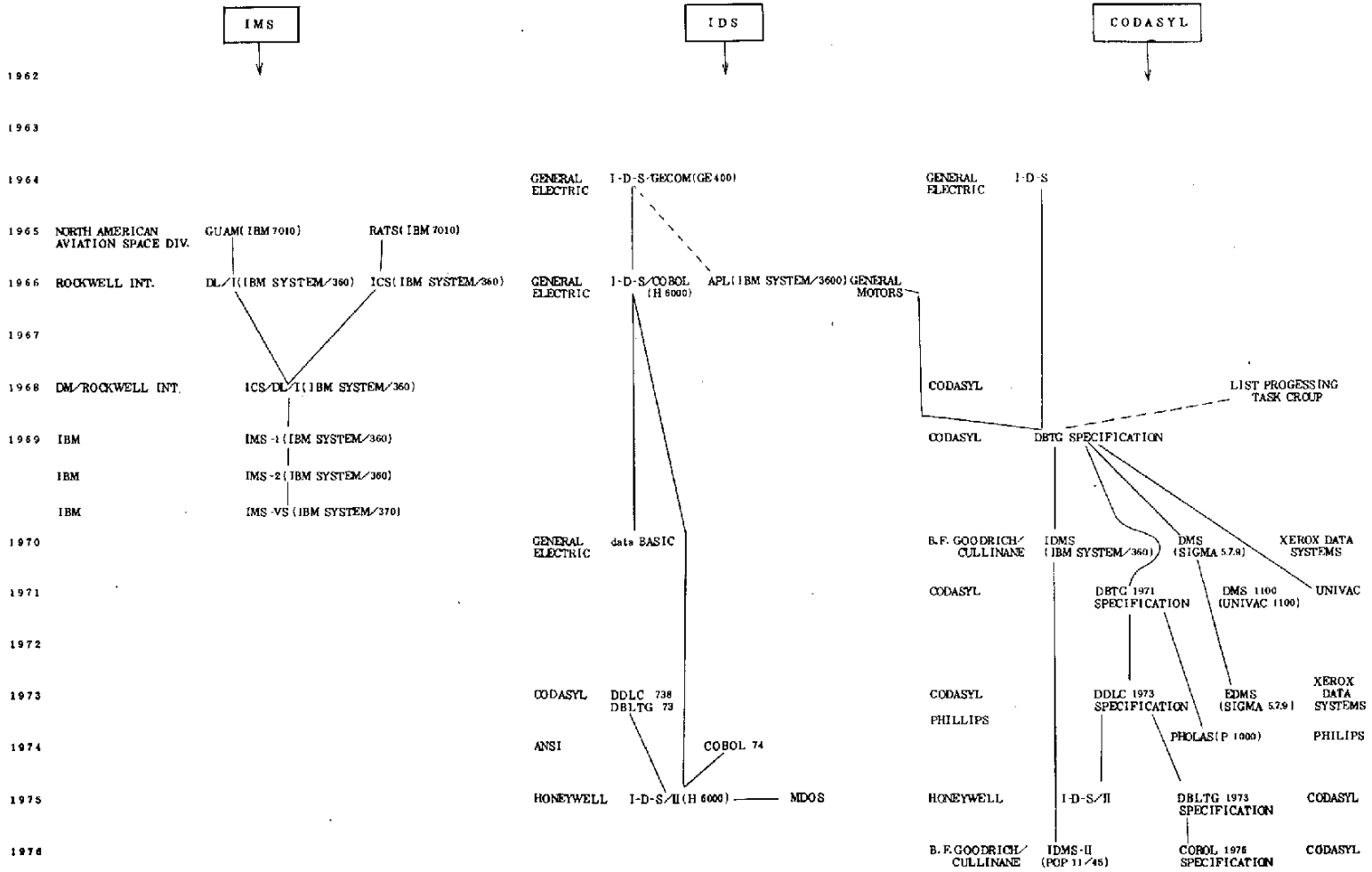


図２－１(a)　ＤＢＭＳの歴史　　IMS／IDS／CODASYL

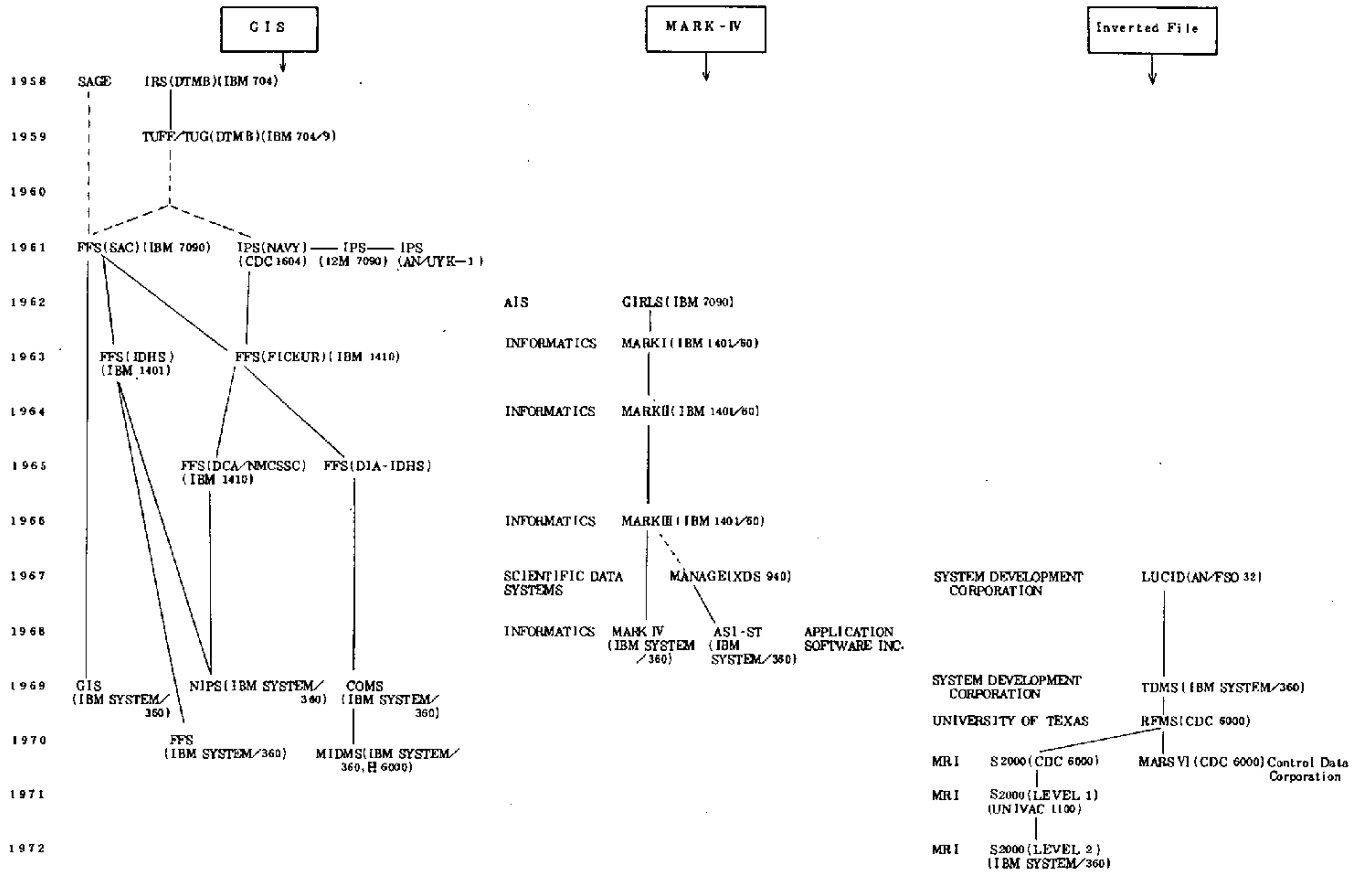


図2－1(b)　DBMSの歴史　　GIS／MARK－Ⅳ／Inverted File

表２－１　データベースモデルに基づいた商用ＤＢＭＳの分類

型	ＤＢＭＳ名	開発機関	開発年	稼働機種	備考
階層型	DL／1	Rockwell	1966	System 360	
	GIS	IBM	1969	System 360	
	IMS－1	IBM	1969	System 360	
	IMS－2	IBM	1969	System 360	
	IMS－VS	IBM	1969	System 370	
	CASSM	フロリダ大学			データベースマシン
	MUMPS	マサチューセッツ General Hospital			
	System 2000	MRI社			
網型（CODASYL型以外も含む）	Aberdeen 大学DBMS				
	TOTAL	Cincom Systems		数多し	
	IDS	GE	1964	GE 400	
	IDS－II	Honeywell	1975	H 6000	
	IDMS	B.F. Goodrich Cullinane	1970	System 360	
	IDMS－II	B.F. Goodrich Cullinane	1976	PDP 11／45	
	DMS 1100	UNIVAC	1971	UNIVAC 1100	
	DMS	Xerox Data Systems	1970	Sigma 5,7,9	
	EDMS	Xerox Data Syetems	1973	Sigma 5,7,9	
	XDMS	ベル研究所			データベースマシン
	PHOLAS	Phillips	1973	P 1100	
	DBMS 2200	NEC		NEAC 2200	
	AIM	FACOM		FACOM M	
	DMS－5			MELCOM－COSMO	
	DBMS－10,11,20	DEC			
	ADBS	NEC		ACOS 4	
	DMS 90			OUK 9000	
	SAAB DMS	SAAB			
	SIBAS	？			
その他	DPLS	千代田加工			
	FORIMS	日本ユニバック			
	RAP	トロント大学			データベースマシン
	SYSTEM－R	IBM			
	ADABAS	Software AG			
	RARES	ユタ大学			データベースマシン

2.2 既存のＤＢＭＳに見られるデータベースモデル

本節では既存のＤＢＭＳに見られるデータベースモデルに焦点をあて、どのような分類が行なわれているか調査する。この調査の結果、ＤＢＭＳの代表的なデータベースモデルは、階層型モデル、網型モデル、関係モデルの三つであるとし、それぞれについてモデルの解説を行なう。なお、各モデルの解説で引用している共通例題は〔ＡＣＭ Computing Surveys, Vol. 8, №1, March 1976 データベース特集号〕の大統領データベース（Presidential Data Base）である。

〔ＳＡＫＡ７８〕によれば、「データベースの種類は数１０を数える。データモデル分類学と称するものもある。しかし、どのモデルが最良かについて一致した見解はない。また技術的問題および使いやすさからみて、今後数年以内にどのアプローチが決定的な支持を得るかについても断定できない。」とある。さらに、「現在少くとも５種類のデータモデルが注目されている。階層モデル、ネットワークモデル、リレーショナルモデル、２項関係モデルおよび集合論モデル。」とある。

しかし、既存のＤＢＭＳを分類する場合、今日一般に行なわれる分類の中で代表的なものが、次に示す三分類方式である〔ＤＡＴＥ７７、ＴＳＩＣ７７、ＨＯＴＡ７７ｂ、ＮＩＳＩ７６〕。

- hierarchical model
- network model
- relational model

これらは、それぞれ階層型モデル（あるいは木構造）、網型モデル、関係モデルと訳されている場合が多い。

しかし、既存の全てのＤＢＭＳをこの分類に従って行なうことは、若干無理のようである。また、網型モデルのかわりにＣＯＤＡＳＹＬ　ＤＢＴＧモデルと表現される場合もある。さらに、関係モデルが平面的なファイルであるため、矩形モデルと呼ぶ場合もある。この矩形モデルのなかにインバーテッド型のＤＢＭＳを入れている場合もある。

さらにＡＤＡＢＡＳ型が矩形モデルである〔ＦＵＪＩ７７〕とか、独立に開発されたにもかかわらず、概念的に関係モデルに近いという分類もみられる〔ＩＳＨＩＩ７７〕。また、〔ＳＥＮＫ７７〕では、階層型、網型、単一階層型（single level）としている。前述の三分類に従って、既存のＤＢＭＳを次のように分類する場合が多い。

- 階層型モデル…… IMS（IBM），SYSTEM 2000（MRI）
- 網型モデル　CODASYL DBTG model …… IDS-II（HIS），DMS 1100（UNIVAC）
- 関係モデル…… SYSTEM-R（IBM）

〔ＴＳＩＣ７７〕によれば、データモデルとはデータの論理編成を理解するために用いられる知的な道具と定義した上で次のような分類をしている。

- 網型データモデル
  - 階層型データモデル
  - 網型データモデル
- 関係データモデル

また、既存のＤＢＭＳの分類を次のように行なっている。

- 階層型システム……ＩＭＳ，ＳＹＳＴＥＭ２０００
- 網型システム
  - ＤＢＴＧ網型システム……ＩＤＭＳ
  - 非ＤＢＴＧ網型システム…ＴＯＴＡＬ
- 関係システム

また現実にはＤＢＭＳの標準化は存在せず、ＣＯＤＡＳＹＬモデルにしても提案という形にとどまっている。そのため、ＣＯＤＡＳＹＬ型のＤＢＭＳにしても各インプリメンタによって実現形態が異なっているのが現状である。なお、ＴＯＴＡＬ（Cincom Systems）の場合、網型モデルとしたり、ＣＯＤＡＳＹＬモデルとしたりして混乱が見られる。これは、網型モデルつまりＣＯＤＡＳＹＬモデルと短絡的に結びつけた結果生じた問題であろう。

Michaels 等は関係モデルとＣＯＤＡＳＹＬモデルを比較した論文〔ＭＩＣＨ７６〕の中で次のように述べている。『多様な利用者によるデータの共有に対してどのアプローチがより良いかということに関する多くの議論が発生した。このような疑問に関して簡単な解答はない。それぞれのアプローチは、多様な利用者の一部のニーズに答えていることを確信している。

・ＣＯＤＡＳＹＬ　ＤＢＴＧのインプリメンテーションは、商用として売り出されている。

・関係モデルのインプリメンテーションは、大学や研究機関で開発されている。

現時点において、次の一点は明確にされるべきである。ＤＢＭＳは、広い範囲のエンドユーザに利用されるものであるから、近い将来において、ただ一つのデータベース管理のアプローチが望まれてもいないし、有力なものも出現してこないだろう。この事実は、この論文でおこなった比較を要約する場合に、心にとめておかなければならない。』

〔ＨＯＴＡ７７ｆ〕によれば、

「データベースの研究者たちの唱えるデータベース・モデルというのは、〔中略〕、その実現性に対しては、いささか無責任なものである。そこここの論文で盛んに議論されているデータベース・モデルのかなりの多くのものはこの種のものであって、実働可能なＤＢＭＳによって裏づけられているデータベース・モデルとは明確に区別して論議しなければならない。〔中略〕情報の具体的な表現方法というよりは、哲学的思考方法といったほうがふさわしい。」さらにＤＢＭＳが提供するデータベース・モデルを木型モデル、ネットワーク型モデル、矩形型モデルと分けながら、次のようなことわりをしている〔ＨＯＴＡ７７ｂ〕。「この分類方法は完全なものではなく、世の中のデータベース・モデルがこれだけであるという保障もなければ、分類が互いに排他的であるということでもない。あくまでも、ＤＢＭＳを手取り早く理解するための第１次近似として考えていただきたい。」としている。

ＣｕｒｔｉｃｅはＤＢＭＳの将来を展望した論文で次のように述べている〔ＣＵＲＴ７６〕。データベース・システムの性格は、記憶構造とサポートされているアクセス法の二つから議論されている。

- 記憶構造
  - 階層構造
  - 網型構造
  - 転置構造
- アクセス法
  - 乱アクセス
  - 順アクセス
  - 転置アクセス

Curticeによれば、5年後（１９８１年）には、このような分類は、余り問題視されなくなるであろうと言われている。つまり、個々のＤＢＭＳパッケージは、唯一の記憶構造やアクセス法についてのみサポートするのではなく、幾つかの組み合わせとしてインプリメントされていくとしている。

既存のデータベース管理システムは、１９８０年代初頭までは、機能強化を伴って存在していくであろう。

そのような機能強化は、多くの一般的なシステムにとって、パフォーマンス上の違いはあっても、機能的に相当類似して行く結果になろう。

全ての既存システムは、新しい記憶装置を利用することができよう。また、基本的なアクセス機構はハードウェアで提供される可能性がかなり強い。

関係モデルを用いたデータベースが既存のシステムに導入されるであろう。しかし、ごく限られたアプリケーションとなろう。大々的な利用は１９８０年代中頃に期待されるハードウェアの開発を待つことになろう。

分散型データベースは、非常に限られたアプリケーションに見られよう。しかし１９８５年まではそれ程普及しないであろう。

このようなことからも将来のＤＢＭＳが進むべき道は、現在データモデルと言われているモデルの一つだけをサポートするだけでは、ユーザの要求を満足することはできないであろう。また、多くのＤＢＭＳが単一のモデルに限らず、実質的には複数個のモデルをサポートしてきているのも現実である。最近発表されたＣＯＤＡＳＹＬ　ＤＢＴＧの変更では、ほとんどセット（親子集合）を意識しないでレコードをアクセスできるなど、良い意味で特徴がなくなってきている。

このように、将来はＤＢＭＳの分類は余り問題にされず、個々のＤＢＭＳも類似し統一化の方向へ進む可能性がある。ここでいう統一化とは、決して標準化といったものではなく、もう少こし弱い意味である。

以上のようなことから、次のような分類方法を提案したい。

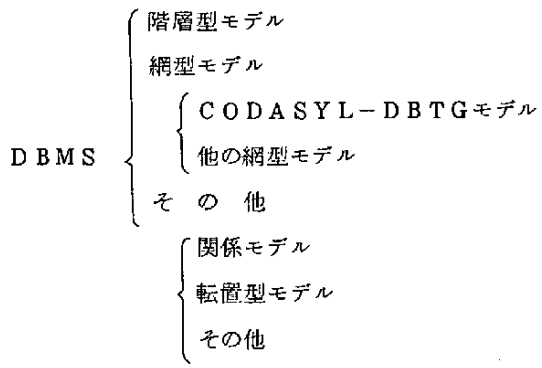


図２－２　ＤＢＭＳの分類

2.2.1　階層型モデル

本項では階層型モデルの特徴、言語、アクセス方式などについて解説する。階層型モデルは、二つのレコード間の関係を１：Ｎに限っていることが大きな特徴である。用いる言語は、データベースのデータ構造を表わすデータベーストリーに沿って検索するトリートラバーサル方式と、データベーストリーの構造を意識しないジェネラルセレクション方式とがある。また、レコードのアクセス方式においては、ポインタ方式とデータベーストリーの論理アドレスを用いるトレイス方式とがある。以下、主に〔ＴＳＩＣ７６，７７，ＷＡＴＡ７６〕を引用して階層型モデルについて解説を行なう。

A.　概要

現実の世界にはいろいろな構造が存在するが、最も一般的でしかも簡単な構造が階層構造である。階層構造は、自然界、人間界においてもありふれたものである。従って、ＤＢＭＳが階層構造を表わしたり、操作するのに必要な機能を提供するのは自然であると思われる。以下、階層型モデルの概要について紹介する。

一般に、データ間の関係には２種類あり、同一エンティティ内の属性間の関係とエンティティ間の関係がある。属性間の関係はレコード内で表わすことができるので、本項では主に異なったエンティティ間の関係について述べる。

エンティティ間の関係例を図２－３に示す。図２－３においてそれぞれのノードはエンティティを表わすレコード型を示し、それぞれの線はエンティティ間の関係を示している。どの２つのノード間においても多くの関係が可能であるが、他の関係と区別するために名前がつけられる。図２－３ではレコード型、ＰＲＥＳＩＤＥＮＴとＣＯＮＧＲＥＳＳ間の関係を始めいくつかが示されている。図２－３で描かれているそれぞれの関係は一般的なＮ：Ｍの関係である。ＰＲＥＳＩＤＥＮＴとＣＯＮＧＲＥＳＳはＮ：Ｍの関係になっているが、実際にはこれ以外のほとんどの関係が１：Ｎの関係になっている。

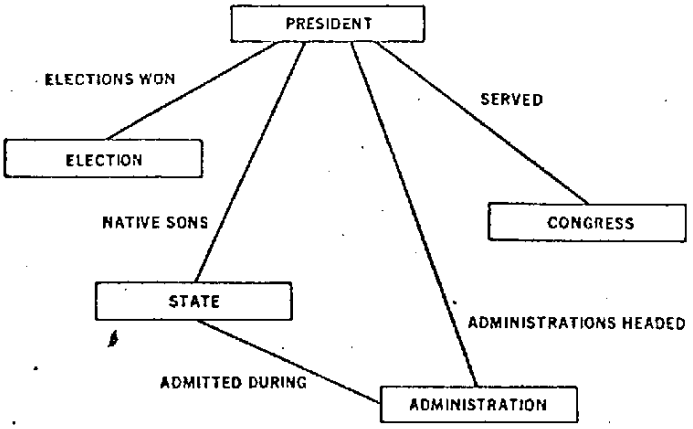


図２－３　一般関係図

Ｎ：Ｍの関係の中間に第三のレコード型を入れることで２つの１：Ｎの関係に変型する方法がある。この変型をくり返し適用することによって、図２－４に示すような、全ての関係が１：Ｎであるデータ構造図を得ることができる。

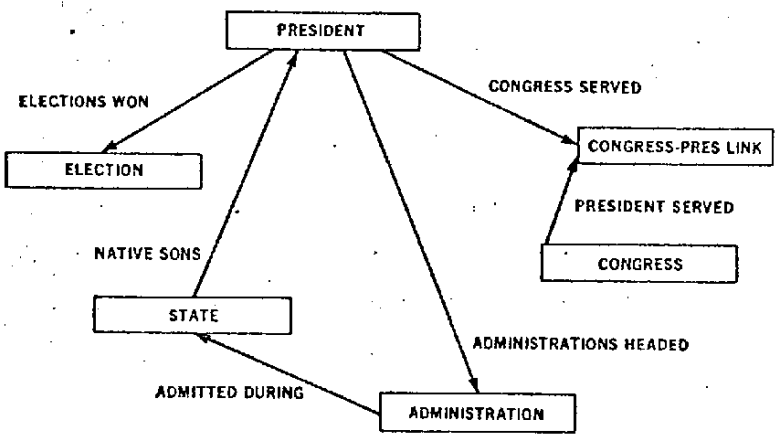


図２－４　データ構造図

エンティティ間の関係はレコード型間の関係として見ることができ、一般性を失うことなしに、全ての関係が１：Ｎであると仮定することができる。１：Ｎの関係はある方向性を持っていて、図２－４で示すように矢印を使うことで表わされる。

データ構造図が図２－５のように根から葉の方向に向かっているトリーを表わしている場合を考えてみると、任意の２つのレコード型間には、多くとも１本の矢印しかないので、その関係に名前をつける必要がない。このようなデータ構造図は階層型定義トリー（hierarchical definition tree）と呼ばれる。階層型定義トリーはデータベース内に含まれているレコード型とそのレコード型間の関係の両方を示している。図２－５は階層型定義トリーで図２－４で示したデー

タ構造図のサブ・セットになっている。

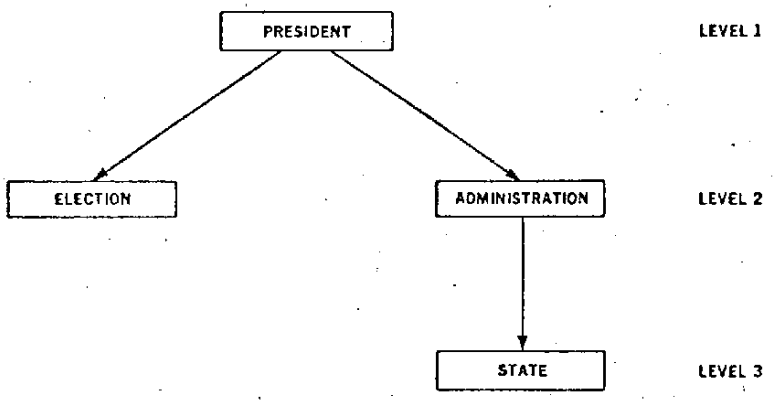


図２－５　階層型定義トリー

階層型モデルは階層型定義トリーの構造的な関係に従ってデータを論理的に編成したデータ・モデルである。階層型定義トリーにおけるレコード型のレベルはトリーの根からの隔たりによっている。図２－５の階層型定義トリーに対応するデータ・ベースを図２－６に示す。階層型データ・ベースはデータ・ベース・トリーと呼ばれているトリーの集合である。データ・ベース・トリーではノードとしてレコード実現値が使われる。従って図２－６においてはPRESIDENTのレコード実現値が３つあるので３つのデータ・ベース・トリーがあることになる。

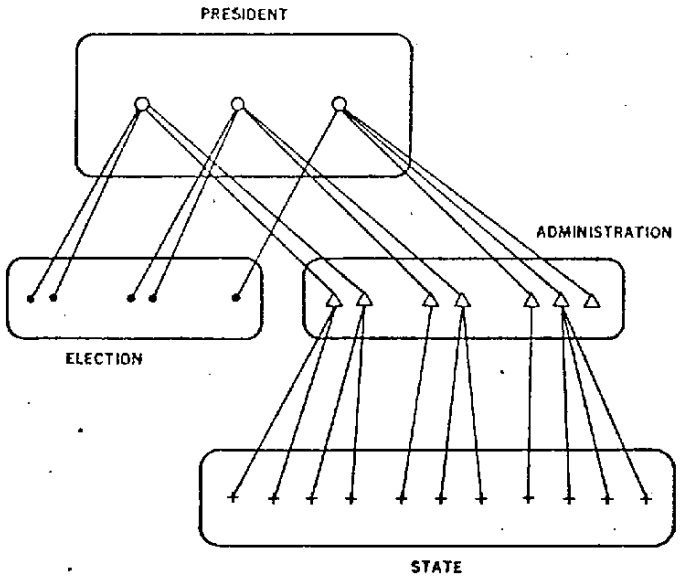


図２－６　データベーストリー

階層型データ・ベースにおいてレコード実現値間の親と子又はアンセスタ（ ancester （親、親の親…））とディセンダント（ descendant（子、子の子…））を自然な方法で指定することが

できる。

図２－６において注意すべきことは、各レベルのレコード型の実現値の数が一定でないことと、それぞれのレコード実現値（根レコード実現値を除く）はアンセスタレコードの実現値と結びついていなければならないことである。

図２－４で示された大統領データベースの例を１つの階層型定義トリーで示すことはできない。データ構造図はそれ自体ネットワークである。しかしながら図２－７に示すように三つの階層型定義トリーを使って同じ情報を表わすことができる。

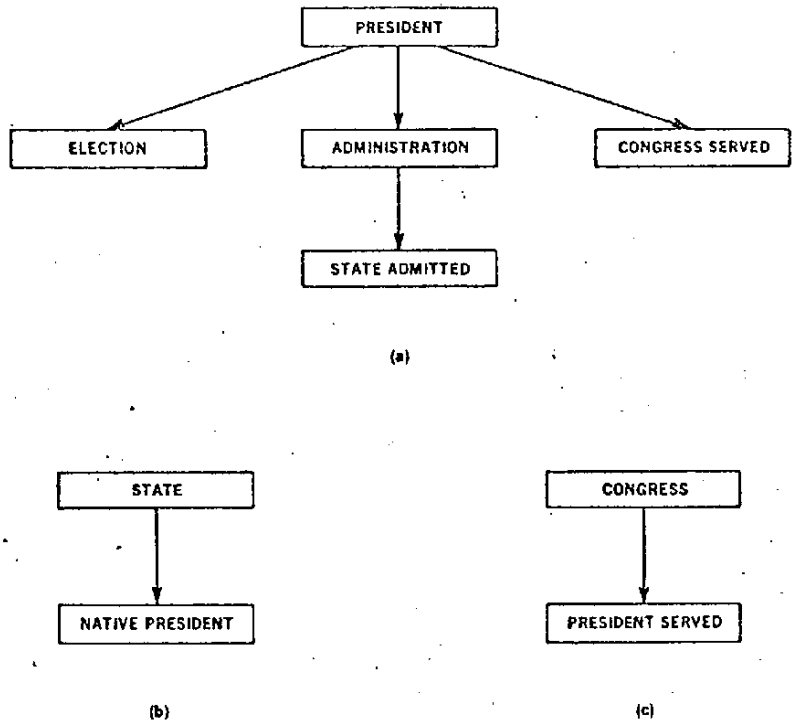


図２－７　大統領データベースの階層型定義トリー

なお、大統領データベースの各レコード型に含まれるデータ項目は次のとうりである。

PRESIDENT…… PRES NUMBER, PRES NAME, BIRTHDATE, DEATHDATE, PARTY, SPOUSE

ELECTION …… YEAR, PRES VOTES, LOSER VOTES, LOSER, LOSER PARTY

ADMINISTRATION…… ADMIN NUMBER, INAUG DATE, VICEPRESIDENT

CONGRESS SERVED …… CONGRESS NUMBER

STATE ADMITTED…… STATE NAME

STATE …… STATE NAME, POPULATION, STATE VOTES

NATIVE PRESIDENT … PRES NUMBER

CONGRESS ····· CONGRESS NUMBER, SENATE REP PERCENT,
SENATE DEM PERCENT,
HOUSE REP PERCENT, HOUSE DEM PERCENT

PRESIDENT SERVED ··· PRES NUMBER

B. 階層型モデルの言語

階層型モデルの性質によってデータベースに追加挿入される新しいレコードそれぞれ（根レコード実現値を除く）は親レコード型の実現値と結合されねばならない。通常、この実行は親レコードの選択と子の挿入によって行なわれる。

一方、レコード実現値が削除されるとそのディセンダントレコード実現値の全てが削除される。階層型モデルでは根無しレコードを許していない。図2－6においてAdministrationのあるレコード実現値が削除されると、そのレコード実現値に結合しているStateレコード型の全ての実現値が削除される。親レコードでなく、そのディセンダントレコードを残すこともしばしば必要であるので、システムによってはレコード自体ではなく、データ項目の値のみを削除する機能を提供している。言いかえれば、データ・ベース内に空（null）レコードが存在することを許している。

階層型データベースにおいてレコードの検索は木構造に従って選択され、条件付けられる。選択の基準を示している条件文は、データ項目と比較値を比較の関係演算子で結んだ条件部をさらにAND、OR、NOTのブール演算子を用いて複数の抽出条件を与えることができる。比較の関係演算子として<、≦、>、≧、=、≒又はそれらに相当するニモニックコードが使われる。

（PRES NAME=EISENHOWER）AND（YEAR=1956）という条件文ではPRESIDENTレコードとELECTIONレコードの両方が選択される。一般的には条件文は階層型定義トリーのどのレコード型におけるデータ項目も使うことができるが、ほとんどのシステムでは1つの階層パスにあるレコード型におけるデータ項目しか条件文に使うことを許していない。1つのレコードが選択された後に、他のレコードが検索のため条件付けされる。全てのレコードはデータベース内に1つのアンセスタを持っている。そのため選択されたレコードのアンセスタを検索のために条件付けすることができる。またディセンダントを持っているレコードは検索のためこのディセンダントで条件付けすることができる。選択されたレコードからアンセスタで条件付けするのを上向きの階層正規化（upward hierarchical normalization）、ディセンダントで条件付けるのを下向きの階層正規化（downward hierarchical normalization）という。

階層型データ・ベースにおける検索操作は、2つの水準の異なる方法のどちらかで実行される。

・トリートラバーサル方式（tree traversal）

・ジェネラルセレクション方式（general selection／hierarchical selection）

トリートラバーサル方式はデータベース・トリーを決められた順序にたどるためにデータベースのトリー構造を陽に使用する方法で、レコードの選択や条件付けはトラバースとは独立である。

ジェネラルセレクション方式は、レコード型のデータ項目の間の関係に基づいてレコードを選

択、条件付けするものでハイ・レベル・インタフェースを使ったものである。ユーザはデータ・ベースの構造を知っていなければならないが、レコードを検索するためにこの構造を陽には使用しない。そのかわりにシステムがレコードを決定するために階層構造を利用する。

以下述べる二つの方式を代表するそれぞれのＤＢＭＳであるＩＭＳやＳＹＳＴＥＭ２０００では、レコード型やレコード実現値に相当する用語として表２－２に示すものを用いている。しかし、以下の説明では固有の用語は用いないで、一般的な用語を用いるものとする。

表２－２　階層型ＤＢＭＳで用いる用語の比較

一　般　名　称	Tree Traversal 方式 （ＩＭＳ）	General Selection 方式 （SYSTEM 2000）
レ　コ　ー　ド　型 (record type)	segment type { root segment type / dependant segment type }	repeating group
レコード実現値 (record occurrence)	segment occurrence	repeating group occurrence
データベースツリー (data-base tree)	data-base record	logical entry
デ　ー　タ　項　目 (data item)	field	data element

(1)　トリートラバーサル方式

トリートラバーサル方式（tree traversal，以下ＴＴ方式という）を採用しているＤＢＭＳの１つにＩＭＳ（Information Management System，ＩＢＭ社）がある。ＩＭＳではデータベース・トリーをトラバースするために既定ＴＴ方式（preorder tree traversal ）が使われている。既定ＴＴ方式は次のように定義できる。

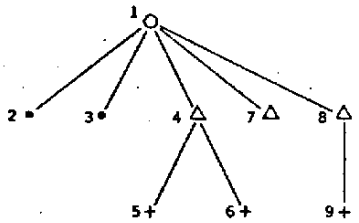
a）そのレコードがまだたどられていなければそのレコードをたどる。

b）さもなければ、まだたどられていない子レコードの最も左にあるものをたどる。

c）さもなければ、子レコード、孫レコード、…が残っていなくなったらその親レコードにもどる。

この方式は根レコードから始まって、上から下に、左から右の順に、本質的にはトリーの中の全てのレコードをたどっていく。

図２－８は、既定ＴＴ方式の例を示しており、番号はたどっていく順序である。



○ PRESIDENT

● ELECTION

△ ADMINISTRATION

\+ STATE

図２－８　既定ＴＴ方式

① ＩＭＳのＤＤＬ

ＩＭＳのＤＤＬではセパレータ方式の言語を採用している。データ定義はオペレーティングシステムの記憶領域割付機能を利用して単独のデータ定義ジョブとして実行される。その定義は木構造のレコードごとにサブセットとなるような命令の集合である。又レコードスキーマの定義はそれを構成している項目スキーマの定義より前に書く。

a) レコードスキーマ〔ＳＥＧＭ〕

レコードスキーマは次の型式で定義する。

ＳＥＧＭ　ＮＡＭＥ＝レコード名，ＰＡＲＥＮＴ＝物理的な親レコード名，

ＢＹＴＥＳ＝レコード実現値長〔，ＦＲＥＱ＝回数〕

この後にレコード順序キーの指定が続き、さらにレコードスキーマ内の他の項目スキーマを続けて記述する。ＦＲＥＱは予想される実現値数を指定する。

b) 項目スキーマ〔ＦＩＥＬＤ〕

ＦＩＥＬＤ　ＮＡＭＥ＝（項目名，〔ＳＥＱ（$\begin{matrix},U\\,M\end{matrix}$）〕），ＢＹＴＥＳ＝長さ，

ＴＹＰＥ＝型，ＳＴＡＲＴ＝文字位置

ＳＥＱは、この項目がレコード順序キーであることを示している。Ｕ，Ｍは値が１個（Unique）であるか、複数（Multiple）であるかそれぞれ示している。

ＴＹＰＥはＣ（英数字）、Ｐ（パック１０進数）、Ｘ（２進数）が指定可能である。

文字位置の値は、レコード内でのその項目の先頭のバイト位置である。

図２－９に示すような階層型定義トリーのＤＤＬを図２－１０に示す。

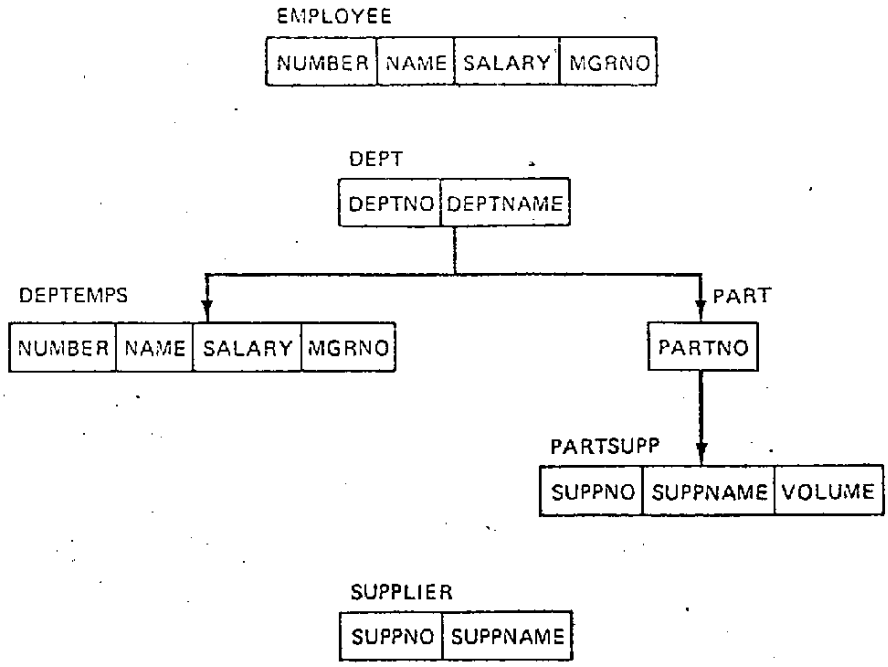


図２－９　階層型定義トリー

```
DBD       NAME=EMPDB,ACCESS=HDAM
DATASET   DD1=EMPHDAM,DEVICE=3330

SEGM      NAME=EMPLOYEE,BYTES=54,FREQ=100
LCHILD    NAME=(DEPTEMPS,DEPTHDAM)
FIELD     NAME=(NUMBER,SEQ,U),BYTES=9,START=1
FIELD     NAME=NAME,BYTES=30,START=10
FIELD     NAME=SALARY,BYTES=6,START=40
FIELD     NAME=MGRNO,BYTES=9,START=46
DBDGEN
FINISH
END

DBD       NAME=SUPPDB,ACCESS=HDAM
DATASET   DD1=SUPPHDAM,DEVICE=3330

SEGM      NAME=SUPPLIER,BYTES=35,FREQ=30
LCHILD    NAME=(PARTSUPP,DEPTHDAM)
FIELD     NAME=(SUPPNO,SEQ,U),BYTES=5,START=1
FIELD     NAME=SUPPNAME,BYTES=30,START=6

DBDGEN
FINISH
END
```

図２－１０　ＩＭＳのＤＤＬ例

② IMSのDML

IMSに対する要求は、PL／1、COBOL、アセンブラを親言語とする応用プログラムからDL／1（Data Language／One）をCall すれば良い。その際指定するCall パラメータには、パラメータ数、機能コード、PCB名、入出力領域名、SSAがある。指定できる機能コードは、大きくGET Call(読み込み)、INSERT Call（挿入）、DELETE／REPLACE Call（削除／更新）の三つに分けることができ、全てのCallにオプションとしてSSAを含むことができる。

- GET UNIQUE(GU) callは、データベース内の現在の位置とは独立にSSAによって選択し、レコードを検索する。このcallは非順次処理、又は順次処理の開始点を設定するために使われる。
- GET NEST(GN) callはデータ・ベース内の現在の位置から次々に検索するものである。データベース・トリーの既定TT方式の順に従って検索する。データ・ベース内のレコードを順次検索する場合はSSAを使用しない。callの中にSSAが含まれている場合は特定のレコードを検索するために使われる。
- GET NEXT WITHIN PARENT(GNP)callは、同一親レコード内の次のレコードを検索する。親レコードは最後のGN call , GU call 又はその前に使われたGN,GU callのSSAにおける機能コードオプションによって設定することができる。GN callとGNP callの相違は与えられた親の下にある最後の子を検索した後に得られる結果だけである。GN callはデータ・ベースの最後を示す状況コードをIMSが設定するまでレコードを検索し続けGNP callはその親のもとで子がなくなったことを示す状況コードを設定する。
- GET HOLD(GHU,GHN,GHNP) call は、レコードを検索し、またそのレコードをDelete又はReplace callのために保持するために使われる。Get Hold callは1つの応用プログラム単位での排他制御を行なう。
- INSERT(ISRT)call はデータベースをロードするため、又はデータベースに新しいレコードを追加するために使われる。

  SSAはレコードを挿入する位置を選択するために使われる。ISRT callは新しいレコードの格納と、そのレコードの親と結合させるという2つの操作を行なっている。この二つの操作は根レコードを除く、全てのレコードが親を持たねばならないので、必要である。
- DELETE(DLET) callはデータベースからレコードとそのレコードの全てのディセンダントを削除する。このDLET callは選択したレコードとその全てのディセンダントを削除するので" triggered delete " と呼ばれている。
- REPLACE(REPL) call は、データベース内のレコードを更新するためのもので、レコード内のキーとなっていないデータ項目を更新することができる。キーになっている

データ項目（例、PRESIDENTレコードのPRES−NUMBER）を変えようとするとエラーになる。

レコードの条件設定はＳＳＡ（Segment Search Argument，セグメント探索引数）によって指定することができる。ＳＳＡは、１つのレコード型に対して条件を指定するもので、レコード名、指令コード、条件部から成っている（図２－１１）。

項目	レコード名	指令コード（オプション）		条件部（任意）						
				修飾開始文字	修飾ステートメント＃1			ブール演算子	～～	修飾ステートメント＃n	修飾終了文字
内容		*	コード文字	(	フィールド名	関係演算子	比較値	＃ *（&） ＋（|）			

図２－１１　ＳＳＡの構成

＜レコード名＞は階層型定義トリー内のレコード型の名称である。

＜指令コード＞はオプションであり、callの際のいろいろなオプションを指定できる。重要なオプションとして次に示すものが許されている。

- １回のＤＬ／１call（path call）によって、根から特定のレコード型にいたるいくつか又は全てのレコードの検索又は挿入。
- 根レコードを除く、任意又は全てのレベルのレコードの下にある最初にある子のバックアップ。
- １つの親のもとで、特定の条件を全て満たすレコードの最後の実現値の検索。
- 特定のレコードを親に設定。

指令コードによって条件付けされている場合を除いて、ＳＳＡによって１つのレコードに決定することができない。又はＳＳＡがない場合には既定ＴＴ方式に従って次のレコードが選択される。

なおＤＬ／１callの結果は、７０数種の状況コード（Status Code）がコミュニケーション・バッファの状況コードパラメータに設定される。そのため応用プログラムの中で適切な処置をとることができる。

＜条件部＞はＳＳＡにおいてオプションであり、ＡＮＤ、ＯＲの演算子が使える。

次に、例題を用いてＩＭＳのＤＭＬを説明する。例題は、「民主党政権中に承認された州の名称を全て印刷する」応用プログラムで、ＰＬ／１で記述している（図２－１２）。

```
DLITPLI:PROCEDURE (QUERY_PCB) OPTIONS (MAIN),
    DECLARE QUERY_PCB POINTER;
    /* Communication Buffer */
    DECLARE 1 PCB BASED (QUERY_PCB),
      2 DATA_BASE_NAME CHAR (8),
      2 SEGMENT_LEVEL CHAR (2),
      2 STATUS_CODE CHAR (2),
      2 PROCESSING_OPTIONS CHAR (4),
      2 RESERVED_FOR_DLI FIXED BINARY (31,0),
      2 SEGMENT_NAME_FEEDBACK CHAR (8),
      2 LENGTH_OF_KEY_FEEDBACK_AREA FIXED BINARY (31,0),
      2 NUMBER_OF_SENSITIVE_SEGMENTS FIXED BINARY (31,0),
      2 KEY_FEEDBACK_AREA CHAR (128);
  /* I/O Buffers */
  DECLARE PRES_IO_AREA CHAR (65),
    1 PRESIDENT DEFINED PRES_IO_AREA,
    2 PRES_NUMBER CHAR (4),
    2 PRES_NAME CHAR (20),
    2 BIRTHDATE CHAR (8),
    2 DEATH_DATE CHAR (8),
    2 PARTY CHAR (10),
    2 SPOUSE CHAR (15);
  DECLARE SADMIT_IO_AREA CHAR (20),
    1 STATE_ADMITTED DEFINED SADMIT_IO_AREA,
    2 STATE_NAME CHAR (20);
  /* Segment Search Arguments */
  DECLARE 1 PRESIDENT_SSA STATIC UNALIGNED,
    2 SEGMENT_NAME CHAR (8) INIT ('PRES    '),
    2 LEFT_PARENTHESIS CHAR (1) INIT ('('),
    2 FIELD_NAME CHAR (8) INIT ('PARTY   '),
    2 CONDITIONAL_OPERATOR CHAR (2) INIT ('¬='),
    2 SEARCH_VALUE CHAR (10) INIT ('DEMOCRAT  '),
    2 RIGHT_PARENTHESIS CHAR (1) INIT (')'),
  DECLARE 1 STATE_ADMITTED_SSA STATIC UNALIGNED,
    2 SEGMENT_NAME CHAR (9) INIT ('SADMIT  ');
  /* Some necessary variables */
  DECLARE GU CHAR (4) INIT ('GU  '),
    GN CHAR (4) INIT ('GN  '),
    GNP CHAR (4) INIT ('GNP '),
    FOUR FIXED BINARY (31) INIT (4),
    SUCCESSFUL CHAR (2) INIT ('  '),
    RECORD_NOT_FOUND CHAR (2) INIT ('GE');
  /*This procedure handles IMS error conditions */
  ERROR:PROCEDURE (ERROR_CODE);
      .
      .
      .
  END ERROR;
  /* Main Procedure */
  CALL PLITDLI (FOUR,GU,QUERY_PCB,PRES_IO_AREA,PRESIDENT_SSA);
  DO WHILE (PCB.STATUS_CODE = SUCCESSFUL);
    CALL PLITDLI (FOUR,GNP,QUERY_PCB,SADMIT_IO_AREA, STATE_ADMITTED_SSA);
    DO WHILE (PCB.STATUS_CODE = SUCCESSFUL);
      PUT EDIT (STATE_NAME) (A);
  CALL PLITDLI (FOUR,GNP,QUERY_PCB,SADMIT_IO_AREA, STATE_ADMITTED_SSA);
    END;
    IF PCB.STATUS_CODE ¬= RECORD_NOT_FOUND
      THEN DO;
        CALL ERROR (PCB.STATUS_CODE);
        RETURN;
        END;
    CALL PLITDLI (FOUR,GN,QUERY_PCB,PRES_IO_AREA, PRESIDENT_SSA);
  END;
  IF PCB.STATUS_CODE¬= RECORD_NOT_FOUND
    THEN DO;
      CALL ERROR (PCB.STATUS_CODE);
      RETURN;
      END;
END OUTPU;
```

IMSは必要な初期設定を行なった後、DLITPL1と呼ばれる手続きに制御が渡される。ユーザのプログラム内で、ユーザはコミュニケーションバッファのマスクを定義し、入出力領域をアロケート、必要な種々のSSAの型式の設定をしなければならない。

PCB（Program Communication Block）はPL／1のプログラムの構造を定義している。PCBの中でセンシティブ・セグメントとして定義すると、そのプログラムごとの守備範囲（センシティブ・セグメント）が決定して、その論理データ構造があたかもセンシティブ・セグメントのみで構成されているかのように取り扱うことができる。

コミュニケーションバッファはユーザのプログラムとの間のコミュニケーションのために必要であり、IMSによってアロケートされる。それに対するポインタ（QUERY－PCB）はパラメータとして応用プログラムに渡される。アプリケーション・プログラムによってアクセスされるデータ・ベースそれぞれに対してコミュニケーション・バッファが1つ存在する。

入出力領域は階層型定義トリーにおけるそれぞれのレコード型に1つ存在するので、レコード型と対応していて、IMSによって検索されるレコードを、holdするために使われ、アプリケーション・プログラムでそのレコードを利用できるようにしている。

SSAは階層型定義トリーのレコード型それぞれに1つ存在している。応用プログラムではIMSデータベースをアクセスするのにはDL／1に対してサブルーチン・コールを行なう。PL／IにおいてDL／1コールはCALL PLITDL1によって行なうことができる。このcallにはいくつかのパラメータが使われる。このパラメータの数は可変で、例においては5つ含まれる。

- パラメータ数（この例では4）
- 機能コード（GET NEXT、INSERTなど）
- PCB名（コミュニケーション・バッファに対するポインタ）
- 入出力領域名（I／Oバッファのロケーション）
- SSA

例に示した要求は次のようないくつかのステップで実行される。

1) PARTYデータ項目がDEMOCRATと等しくなるようなPRESIDENTの最初の実現値を検索する。このアクションはPRESIDENT SSAといっしょにGU callを使って実行される。レコードが1つも選択されないならば処理は完了する。

2) そのPRESIDENTレコードの最初のSTATE ADMITTEDレコードをGETする。このアクションはSTATE ADMITTED SSAといっしょにGNP callを使い実行される。大統領の在任中に1つの州も承認されていない場合にはステップ4に行く。

3) この大統領に対するSTATE ADMITTEDレコードがもうなくなるまで、次々にSTATE ADMITTEDレコードを全てGETする。そしてそれぞれの州の名称を印刷する。

4) PARTYデータ項目がDEMOCRATと等しくなる次のPRESIDENTレコードをGETする。このアクションはステップ1でGUといっしょに使ったのと同じSSAといっしょにGN callによって実行される。もし1つのPRESIDENTレコードも選択することができないならば、この処理は完了する。他の場合はステップ2にいく。

図2－12　PL／1で記述したIMS／DMLの例

③ まとめ

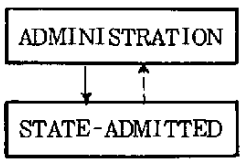
ＴＴ方式の言語では１度に１つのレコードしか操作できない。結果としてトリー上をたどっていくけれど、この処理は順次処理のように見える。ＴＴ方式の言語で使われるコマンドの性質はインプリメンテーションに影響を与えている。論理的に続いている階層上のレコードを物理的に隣接しておくことには効率上望ましい。この方法が実行されるならばデータベース・トリーの順次処理は非常に効率よくなると思われる。

(2) ジェネラルセレクション方式

ジェネラルセレクション方式（general selection／hierarchical selection，以下ＧＳ方式という）では１つのレコード型のレコード実現値を１組のデータ項目の集合として扱っている。

階層型定義トリーのレコード型のレコード項目間に見られる関係に従って、レコードは選択され、条件付けされる。ＧＳ方式の条件付けはＷＨＥＲＥ句で指定する。ＷＨＥＲＥ句は条件をブール演算子で結んだものとＷＨＥＲＥというキーワードから構成されている。ＷＨＥＲＥ句で指定されたレコードの選択が終了した後に、上向き又は下向きの階層正規化（upward／downward hierarchical normalization）が行なわれる。下向きの階層正規化は通常１つの階層パスに限って行なわれる。ＧＳ方式のＤＢＭＳ例としてSYSTEM２０００が挙げられる。この言語の特徴は英語に似ていて、会話型問合せ言語であることである。この言語のコマンドは２つの部分、実行部とＷＨＥＲＥ句から成っている。実行部は実行する操作を指定するものである。最も基本的な検索コマンドとしてＰＲＩＮＴコマンドがある。次の例で上向きの階層正規化を説明する。

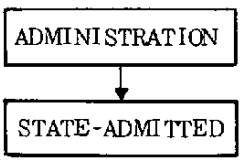
PRINT VICEPRESIDENT *WHERE* STATE NAME *EQ* ALASKA



この問合せではADMINISTRATIONレコードがＷＨＥＲＥ句を満足するＳＴＡＴＥ ADMITTEDを持っていなければならない。これを調べるために、ＷＨＥＲＥ句を満足するＳＴＡＴＥ ADMITTEDレコードを全て選択し、ADMINISTRATIONを条件付けするために上向きの階層正規化を実行する。

次に下向きの階層正規化の例を挙げる。

PRINT STATE NAME *WHERE* VICEPRESIDENT *EQ* NIXON



又上向き、下向きいずれの階層正規化も含まない例を次に示す。

PRINT ADMIN NUMBER *WHERE* VICEPRESIDENT *EQ* NIXON

この場合ADMINISTRATIONレコードだけを選択、条件付けすることで答えることができる。

次の例は、初心者が「副大統領がＡＧＮＥＷとＦＯＲＤのときの大統領」を問合わせたもので、その応答としてＮＩＸＯＮを期待している。

PRINT PRES NAME *WHERE* VICEPRESIDENT *EQ* AGNEW
AND VICEPRESIDENT *EQ* FORD

しかしこれに対する応答は、そのような大統領はいないというものとなる。これは、レコードを選択する時に同一レベルで全てのブール演算子を実行しなければいけないために起った現象である。レコードは一つのデータ項目に２つの異なった値を持つことができない。もしブール演算子がより高いレベルで実行されるならば、この問題は解決される。選択したレコードに上向きの階層正規化をすることによってブール演算子の実行レベルを上げるために次のようにＨＡＳ句が使われる。

PRINT PRES NAME *WHERE* PRES NAME *HAS* VICEPRESIDENT
EQ AGNEW *AND* PRES NAME *HAS* VICEPRESIDENT *EQ* FORD

この問合わせからは、期待通り「ＮＩＸＯＮ」が応答されると予想できる。ＧＳ方式では一般にレコードとそのデータベーストリー内の位置を区別して把握している。例えば、ＳＹＳＴＥＭ２０００にはＲＥＭＯＶＥコマンドがあり、レコード実現値からデータ項目の値だけを削除することができる。また、データベーストリー内を上下する必要のある問合わせも一つのコマンドで実行できる。これらの機能によってＧＳ方式はＴＴ方式より融通がきくものとなっている。なお、ＳＹＳＴＥＭ２０００に関しては、ＤＤＬを含めて2.3.2項に詳解してある。

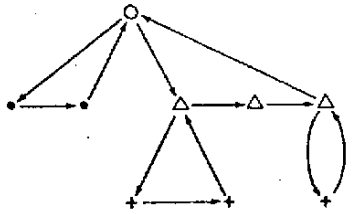
C. 階層型モデルのアクセス方式

階層型モデルのアクセス方式は、ポインタ方式と論理アドレスによるトレース方式とがある。以下、それぞれについて解説を行なう。

(1) ポインタ方式

階層型データ・ベースをアクセスする１つの方法としてポインタ方式がある。基本的には、階層型定義トリーの２つのレコード型間の関係は１組のポインタで表わすことができる。ポインタは親と子レコードの実現値同志の結合を表現している。ポインタを使って編成することは、具象化するのは簡単であるが、大量にスペースを必要としているという欠点がある。

必要なスペースを減らす方法として２つの方法が考えられる。１つの方法は親と子レコード間に前向き／逆向きポインタをそれぞれ設定することを避けて、兄弟ポインタを使うことでスペースを節約することができる。もう１つの方法としては論理的に関係のあるものを物理的に隣接させる方法がある。



○ PRESIDENT

● ELECTION

△ ADMINISTRATION

＋ STATE

図２－１３　階層型ポインタの実行順

ポインタを使ったアプローチにおいて、データ・ベースの物理的編成には種々のものがある。ＩＭＳを例にとれば階層順次（ＨＳＡＭ）、階層索引順次（ＨＩＳＡＭ）、階層直接（ＨＤＡＭ）、階層索引直接（ＨＩＤＡＭ）の４つの編成法がある。

以下の説明では図２－１４のデータ・ベース・レコードを使用する〔ＷＡＴＡ７６〕。

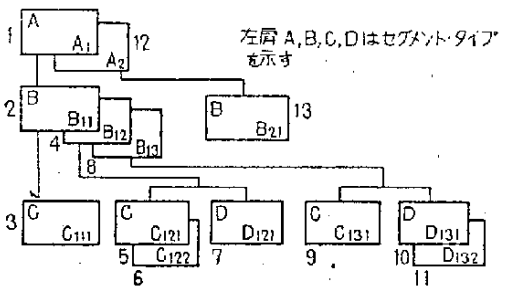


図２－１４　データ・ベース・レコードと階層順序

またレコード形式を図２－１５に示す。

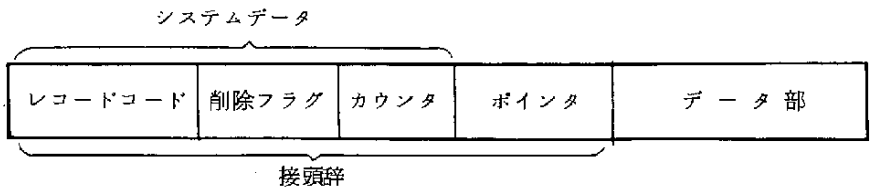


図２－１５　レコード形式

各レコードにはポインタがつけられていて、レコードは接頭辞とデータ部の２つの部分から成っている。データ部はユーザのデータを含んでいて、接頭辞はシステムが管理するものでシ

ステムデータとポインタが含まれ応用プログラムでは利用できない。システム・データの中にはレコード・コードと削除フラグとカウンタがある。レコード・コードはレコード型を指定するために、又カウンタはオプションで論理関係においてレコード型が付けられた場合に使われる。

a) 階層順次 (Hierarchical Sequential Access Method)

各レコードは階層順序通りにディスクやテープ上に配列されて、レコードは固定長でなければならない。追加・削除・置換は直接行なうことができず、データ・ベースを新たに作成せねばならない。これは順次処理にむいたものである。

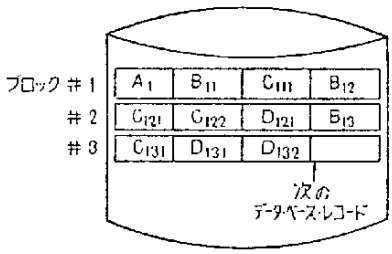


図２－１６　ＨＳＡＭ編成の例

b) 階層索引順次 (Hierarchical Indexed Sequential Access Method)

データ・ベース・トリーに対して索引を使ってアクセスする。それぞれのデータ・ベース・トリーは階層順序に、物理的に隣接した記憶場所に格納される。記憶場所は基本記憶域とあふれ域に分かれていて基本記憶域に収容しきれないレコードはあふれ域に入れられる。基本記憶域とあふれ域はポインタで結ばれている。根レコード型はキーデータ項目を含まなければならず、キーデータ項目はそれぞれのデータ・ベース・トリーをインデックスするために使われている。

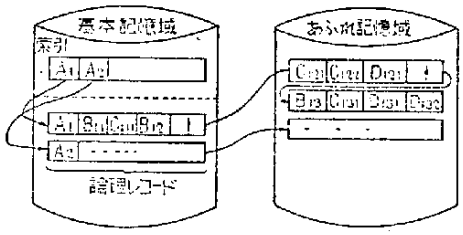


図２－１７　ＨＩＳＡＭ編成の例

順次処理に適すると同時に、索引による乱順処理も可能である。根レコードに近い従属レコードの乱順アクセスの効率はよいが、末端に近いレコードでは順次アクセスの方が編成に合っている。

c) 階層直接（Hierarchical Direct Access Method）

乱順処理の効率を主眼とした編成法である。レコードは根アドレス可能域と呼ばれる基本記憶域にハッシュされる。根レコード型はキー・データ項目を含んでいなければならない。根レコードから階層順に根アドレス可能域のブロックに入り、残りはあふれ域のブロックに入る。両ブロックはポインタで結ばれている。

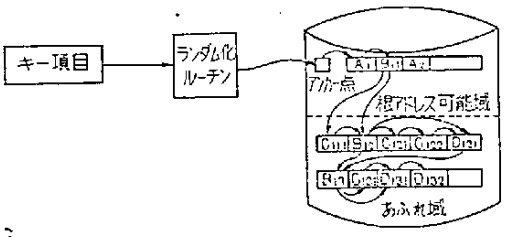


図２－１８　ＨＤＡＭ編成の例

d) 階層索引直接（Hierarchical Indexed Direct Access Method）

この編成の索引はＩＮＤＥＸという順次ファイルである。ＩＮＤＥＸファイルの中のそれぞれのレコードは根レコードのキーデータ項目の値とデータ・ベース内のその根レコードに対するポインタを持っている。

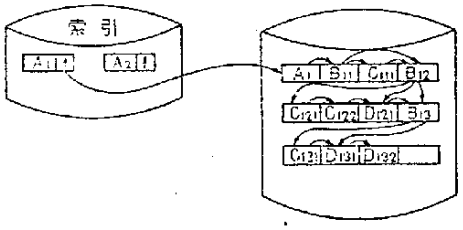


図２－１９　ＨＩＤＡＭ 編成の例

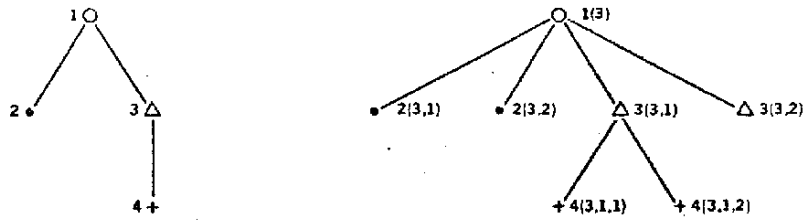
根レコードはＩＮＤＥＸにおけるキー・データ項目の値をサーチすることによってアクセスすることができる。そしてデータベースに対するポインタをたどっていく。データベース内のレコードは、ＨＤＡＭと同様にポインタで関係づけられている。他の編成法の長所を集大成した重装備の編成法で順次・乱順両用の処理に適している。

(2) トレース方式

ポインタを使わない別の方法として、データ・ベース・トリーのレコードに論理的なアドレスをつけるトレース方式がある〔ＴＳＩＣ７６〕。

データ・ベース・トリーのレコードにつけられた論理的なアドレスはトレースと呼ばれている。階層型定義トリーの中のそれぞれのレコードは図２－２０(a)で示されるようにタイプ番号をつけることができる。データ・ベース・トリーの中のどのレコードもデータ・ベース・ト

リー内のそのレコードの実現値にいたるパスを指しているジェネレーショントュプル（generation tuple）とタイプ番号をつけることができる。



(a)　　(b)

図２－２０　トレースの割り付け

(b)では階層データ・ベース内の３番目のデータ・ベース・トリーを示しているので根レコード型に１(3)のトレースがつけられている。トレースが与えられるとそのレコードのアンセスタ、ディセンダント、兄弟のトレースは次に示すように簡単に求めることができる。

- ○アンセスタトレース ── ジェネレーショントュプルの最後の数字をとり除き、タイプ番号を変える〔２(３，１)→１(3)〕。
- ○ディセンダントトレース ── ジェネレーショントュプルの後に数字をつけて、タイプ番号を変える〔３(３，１)→４(３，１，１)〕。
- ○次兄弟トレース ── ジェネレーショントュプルの最後の数字に１加える〔４(３，１，１)→４(３，１，２)〕。
- ○前兄弟トレース ── ジェネレーショントュプルの最後の数字が１でないならば１減じる〔４(３，１，２)→４(３，１，１)〕。

もしトレーステーブルを効率よくインプリメントすることができれば、階層データ構造をよりよく効率的にインプリメントすることに継がる。

D. 階層型モデルのまとめ

階層型モデルの特徴について整理すると次のようになる。

1) レコード型の集合｛R_1，R_2，…，Rn｝から成る。

2) １つのデータ構造図で全てのレコード型の関係を表わせる。

3) 任意の２つのレコード型Ri，Rj の間に１種類の関係だけしか存在しない。そのためその関係に名前をつける必要がない。

4) データ構造図の中で表わされている関係は全ての矢印がその葉に向かうようなトリーを形成する。

5) それぞれの関係は１：Ｎである。もしRi がRj の親であるとすれば、Rj のレコード実現値のそれぞれについて、それと結合している１つのRi レコード実現値が存在する。

階層型システムは長い間利用され、よく受け入れられてきた。特定の階層型システムが成功し

ているといっても、それをそのデータ・モデルが階層型であるからと簡単に結びつけることはできない。商用システムの質に影響を与えている他の多くの要素が存在している。しかしながら、いくつかの適用例たとえば企業の経営構造のようなものについては階層構造は大変自然なものである。場合によっては他の構造のシステムより作るのが難かしく冗長になることがあるかもしれないがほとんどのアプリケーションに階層型の編成でモデル化することができる。

階層構造における機密保護や保全性の機能として、ＩＭＳにセンシティビティ機能がある。これは、データベースに応用プログラムごとの守備範囲（センシティビティ）を定めるものである。ＰＣＢの中で、そのプログラムで使いたいものをセンシティブセグメント（レコード）として定義しておくと、データベースがあたかもセンシティブセグメントのみで構成されているように取り扱うことができ、その他の部分にはアクセスできなくなる。一方ＳＹＳＴＥＭ２０００では、データベース、コマンド、ＤＤＬの各コンポーネント番号などに、それぞれパスワードの設定機能を有している。

最後に階層構造の利点、欠点を挙げる。

階層構造の利点としては、

- ○使用するコマンドの数が比較的少ないので、利用方法をマスターすることが簡単なデータモデルである。
- ○許されている関係づけの型が限られているので、階層型モデル以外と比較して複雑な構造をより簡単にインプリメントすることができる。

階層構造の欠点としては、

- ○レコード間の関係を１：Ｎにするという制限によって、データの不自然な編成を強いることがある。例えばＭ：Ｎの関係においては、階層構造では直接表わすことができないので中間に第三のレコード型をおくことがある。
- ○きびしい階層の順序付けによって挿入、削除のような命令が非常に複雑になる。
- ○削除命令で空（null）レコードが許されていない場合には、削除の対象となるレコードの従属レコードに含まれている情報が失なわれるので、使用には十分な注意が必要となる。
- ○ＧＳ方式の例のように、対称的な問合せは答えられない場合がある。

階層構造のコマンドについてあまりにも手続的であるという批判があるが、この批判に対する答えとしてＧＳ方式のようなハイ・レベル・インタフェースをインプリメントすることが挙げられる。

2.2.2. 網型モデル

網型モデルのＤＢＭＳと一口に言っても、ＩＤＳ－ＣＯＤＡＳＹＬ系列のものや、ＴＯＴＡＬなどのように独立して開発されたものがある。本項では１９７１年にＣＯＤＡＳＹＬのＤＢＴＧが提案した、いわゆるＣＯＤＡＳＹＬ型と言われる網型モデルの解説を行なう。なお、このモデルの解説

にあたっては特に〔ＴＡＹＬ　７６〕を参考にした。

A．概要

１９７１年に発表されたＤＢＴＧ提案に対する議論は賛否両論が盛んに行なわれているが、既に多くのＣＯＤＡＳＹＬ型ＤＢＭＳが実用化されている。また、この提案がＤＢＭＳの標準化に与えた影響は、米国内に留まらず世界的と言われている。

本項では、まず１９７１年のＤＢＴＧ案について一通りの解説を行なう。その後で、最近（１９７７年末）のＣＯＤＡＳＹＬの進展状況を紹介する。

１９７１年のＤＢＴＧ提案の特徴は次の四点に集約できよう。

- Bachmanが開発したＤＢＭＳ　ＩＤＳ〔ＢＡＣＨ　６４〕の影響を強く受けていること
- データベース言語を記述用と操作用に分けて確立し、それが個々のＤＢＭＳに共通利用されることを目的としていること
- ＣＯＢＯＬ言語に数個の命令を付加した形でデータベース機能の実現を可能にしていること
- 階層構造を始め木構造、単純網構造、複合網構造、多重メンバ構造、環構造の一通りのデータ構造がデータベース上に実現可能なこと

このような特徴をもったＣＯＤＡＳＹＬ型について、データ構造の表現方法、ＤＤＬ、ＤＭＬの解説を以下に行なう。なお、インプリメント時に考慮されるべき事項については触れていない。

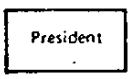
B．データ構造の表現方法

データ構造の設計は専門的技術を必要とする詳細で高度の技術的プロセスである。設計段階を通して、エンティティやエンティティ相互の関係を詳記するために利用できる表記法が必要となる。表記法の中で最も広く用いられているものの一つがデータ構造図（Data Structure Diagrams）である。データ構造図は、Bachmanにより導入された概念で次の二要素から成っている。


：長方形

：　矢

長方形の中にはデータベースで用いられるエンティティあるいはレコード型の名前が入る。以降、具体的な説明を行なう際には、本節の共通例題である「大統領データベース（Presidential Data Base）」を引用する。



——— PRESIDENTというレコード型の実現値を表わしている。

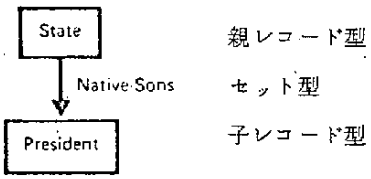
ここでレコード型、実現値等の概念を把握することにする。

- レコード型…１つ以上のデータ項目から成る
- データ項目…データベース全体としては定義されているデータ項目であっても、必要に応じて（参照時）省略が可能である。

・実現値（オカーレンス，occurrence）

…PRESIDENTがレコード型で、具体的なWASHINGTONやJEFFERSONが実現値に相当する。（今後は、レコード型のことを単にレコードともいう）

第二の要素である矢は、2つのレコード型を関係付けるもので、始点を親レコード型（owner-record type）、終点を子レコード型（member-record type）という。親から子へ関係付けた矢をセット（set type）という。



一組のセットの中で1個以上の子レコードをもつことも可能である。レコード型に対するレコード実現値に対応して、セット型の実現値をセット実現値と呼ぶ。セット実現値は、ある親レコードの1つの実現値と、それぞれの子レコード型の0以上の実現値から成る。つまり、親レコードの実現値があれば、必ずセット実現値が存在する。

セット実現値は、1つの親レコードとn（≧0）個の子レコード型との関係であると言える（1：n）。

全てのセット実現値においては、そのテナント（親／子レコード）間に次の3つの関係がある。

① 親レコードが与えられると、そのセット実現値の各子レコードの処理が可能である。

② 子レコードが与えられると、そのセット実現値の各親レコードの処理が可能である。

③ 子レコードが与えられると、同一セット実現値内の他の子レコードの処理が可能である。

次に具体的なセットの例をみてみる（図2－21）。

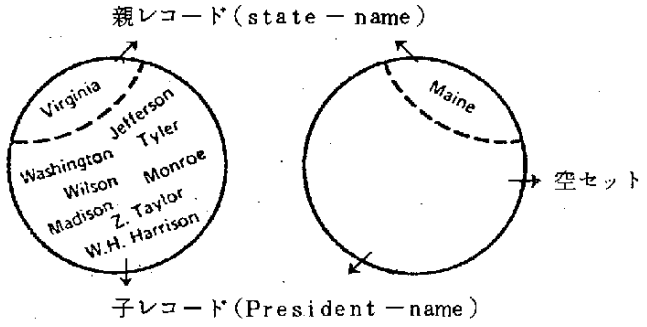


図2－21　セット実現値：NATIVE－SONS

この例は次のことを表わしている。

・セット名；NATIVE−SONS

・セット実現値；２個

・親レコード；STATE−NAME

・子レコード；PRESIDENT−NAME

・各セット実現値においては、１つのSTATE が複数のPRESIDENTに関係している。

・Maine州出身のPRESIDENTはいないので、このセット実現値を空セット（empty set）という。

なお、セットの概念に相当する用語として、Bachman は data structure set，Codd は owner-coupled sets と呼び、その他にも coset，fanset などの用語があるが、すべて同義である。

セット実現値についてもう１つの規則がある。１つの子レコードは、同一セット内に２つの親をもてないというものである。つまり、図２－２２に示すような大統領の出身州が二つとなるようなセットは認めていない。セットに関するこの規則は基本的でかつ重要である。図２－２２に示される制限に対処するための技術については後述する。

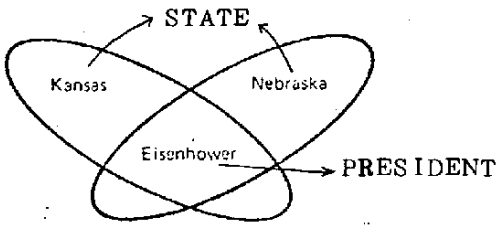


図２－２２　正しくないセットの例

次に三つのレコードから成る階層構造（１：多の関係）の解説を行なう。これを第一の共通構造という。

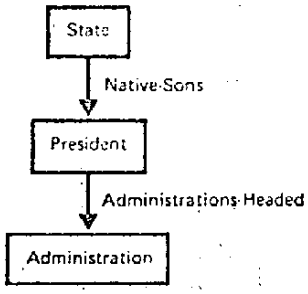


図２－２３　階層構造を示すデータ構造図

図２－２３は、STATE−PRESIDENT−ADMINISTRATION の三層から成る階層構造を示している。各レコードの実現値を考慮して図示すると図２－２４のようになる。

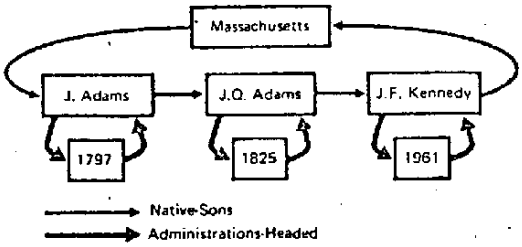


図２－２４　階層構造の実現値

次に、第二の共通構造である多：多のデータ構造の解説を行なう。

例えば、STUDENTレコードとCOURSEレコードの関係が多：多の関係にあると考えられる。

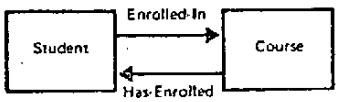


図２－２５　多：多の関係が存在する例

このような例では、図２－２６に示すように親が複数個発生してしまうので、同一セット内に２つの親はもてないという規則に反する。

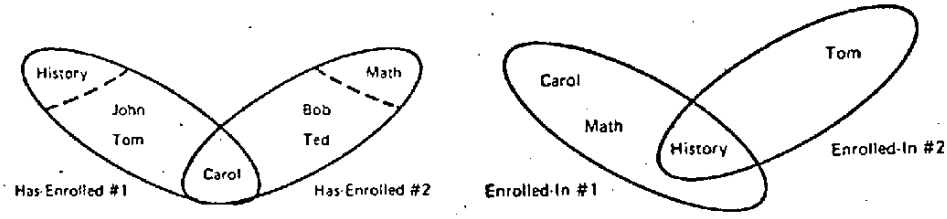


図２－２６　親が複数個ある例

多：多の関係で生ずる問題を解決するために、二つのレコード間に第三のレコードを設ける。このレコードをリンクレコードあるいはコネクタレコードと呼ぶ。

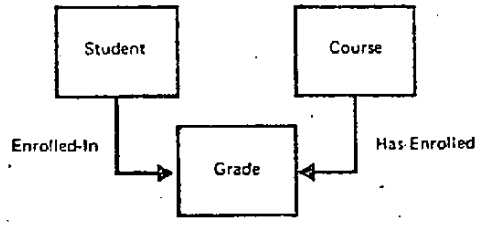


図２－２７　リンクレコードの設定

この新しいレコードＧＲＡＤＥは、二つのレコードを関係付ける為に用いられる。また、情報としては、ＳＴＵＤＥＮＴとＣＯＵＲＳＥの両者に関係しているものを保持する。

これまで示した例は、１つのレコードが１つ以上のセット型の子であった。

次に示す例は、１つのレコードが１つ以上のセット型の親である場合である。

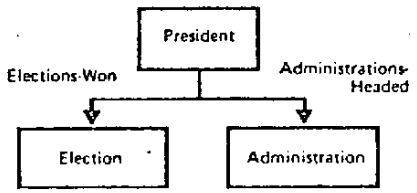


図２－２８　複数のセットタイプの親の例

この例は、Presidentは幾度かElectionに勝ち、多くのAdministration を率いているのがわかる。

また、部品展開のように部品は幾つかの部品からなり、その部品はまた幾つかの部品から成るという、いわゆる再帰的な構造（cycle）をもつデータ構造もある。しかし、ＣＯＤＡＳＹＬ型においては、このような構造は禁止されている。

図２－２９　部品に見られるcycleの例

これを解決するために、次のように二つのセット型を設ける。

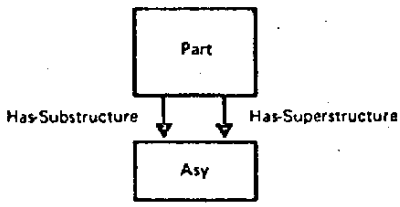
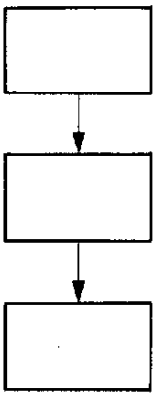


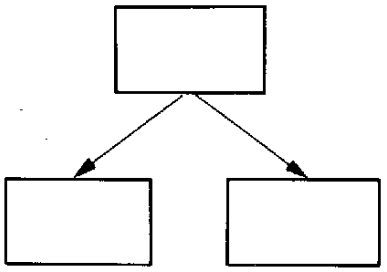
図２－３０　cycleを回避したデータ構造図

ここでAsy レコードは、補助部品の集合を表わしている。

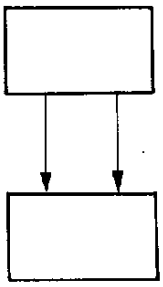
次にまとめとしてデータ構造の一般形を示す。



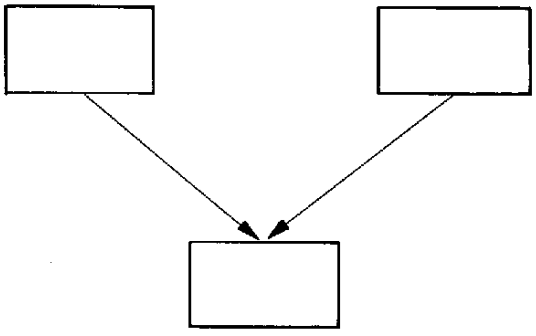
階層構造



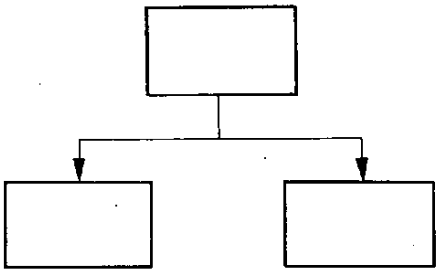
木構造



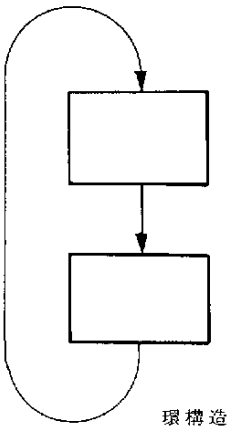
単純ネットワーク構造



複合ネットワーク構造



多重メンバー構造



環構造

図２－３１　データ構造の一般形

C. スキーマＤＤＬ（Schema Data Description Language）

スキーマＤＤＬ、サブスキーマＤＤＬ、ＤＭＬの各項で用いるデータ構造は、共通例題である「大統領データベース」でその全体図は次のとうりである。

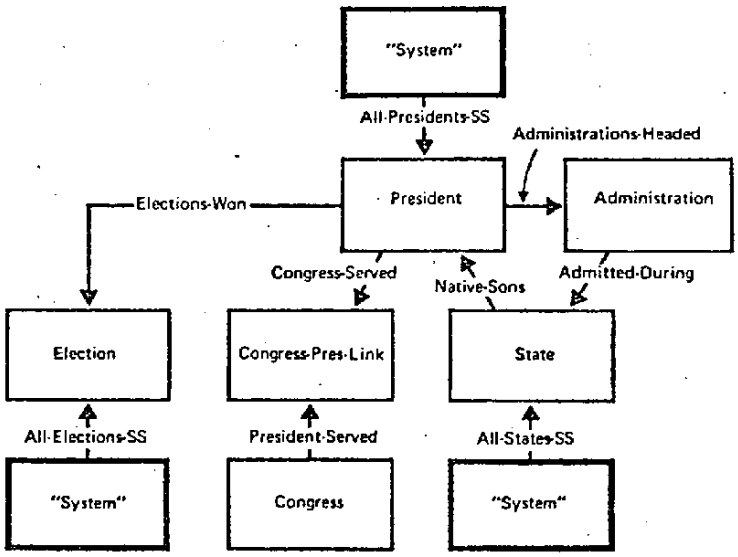


図２－３２　大統領データベースのデータ構造図

スキーマＤＤＬは、データベース全体を記述する言語で、データベース名、エリア名、レコード名、セット名などに関する記述を行なう。さらに、機密保護や保全性に関する決定なども行なう。ＣＯＤＡＳＹＬ型のデータベースは１つ以上のエリアから構成されている。エリアとはデータベースを論理的に分割した部分で、ファイルあるいはデータセットに相当するものである。このエリアを有効に用いれば、機密保護や保全性の効果は一層高まる。論理的に分割されたエリアは、さらに機密保護を高めるために物理的に分割される可能性がある。なお、このエリアをサブスキーマＤＤＬが定義する際には、レルムと呼ばれている。各エリアの物理的記憶装置への実際の割り付けは、装置媒体制御言語ＤＭＣＬ（Device Media Control Language）により行なわれる。ＤＢＴＧ提案に準拠して開発されたＤＢＭＳは、このＤＭＣＬ機能を各ＯＳのジョブ制御言語の中に組み入れている。

次に、ＣＯＤＡＳＹＬが１９７３年に発表したＤＤＬ　ＪＯＤ〔ＣＯＤＡ　７３〕の文法に基づいて、大統領データベースのスキーマＤＤＬを記述してみる。なお、スキーマＤＤＬは次のような構成をしている。

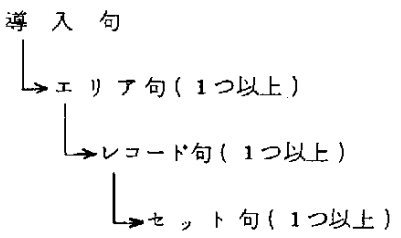


図２－３３　スキーマＤＤＬの構成

レコード句で指定するレコードの実現値をデータベースにどのように位置付けるかの具体的方法は、スキーマＤＤＬの例にあるＣＡＬＣ方式とＶＩＡ方式の他に、ＤＩＲＥＣＴ方式とＳＹＳＴＥＭ方式がある。これらについて説明する。

・ＤＩＲＥＣＴ…データベース中唯一に定まる物理的なキー（データベースキー）を用いた位置付け。

・ＣＡＬＣ…レコード実現値内の項目（≧１）をキーとして、ＤＢＭＳが提供するランダマイジング・ルーチンによる位置付け。

・ＶＩＡ…セット関係を保ちつつ、物理的に可能な限り隣接した位置付け。（近傍配置）

・ＳＹＳＴＥＭ…インプリメンタが定義したデータベース・キーの割り付け法による配置付け。

また、実際のＤＢＭＳにおける各レコード実現値の継がりは、次の三つのポインタを用いて実現できよう。

・next ポインタ（次の実現値を指す）

・prior ポインタ（前の実現値を指す）

・owner ポインタ（親レコードの実現値を指す）

これらの関係を図２－３５に示す。

```
SCHEMA NAME IS PRESIDENTIAL;
  PRIVACY LOCK FOR COPY IS "COPY PASSWORD"
  PRIVACY LOCK FOR ALTER IS PROCEDURE CHECK-AUTHORIZATION;.
```

SCHEMA

データベースを記述するスキーマ名を宣言。

PRIVACY（機密保護）に関する宣言。

```
AREA NAME IS PRESIDENTIAL-AREA;
ON OPEN FOR UPDATE CALL UPDATE-CHECK;
```

AREA

データベースとして使用するエリア名の宣言。

```
RECORD NAME IS PRESIDENT
  LOCATION MODE IS CALC
    USING LAST-NAME, FIRST-NAME, DUPLICATES ARE NOT ALLOWED
  WITHIN PRESIDENTIAL-AREA
  02  PRES-NAME
      03  LAST-NAME   PIC "A(10)"
      03  FIRST-NAME  PIC "A(10)"
  02  PRES-DATE-OF-BIRTH
      03  MONTH-B   PIC "A(9)"
      03  DAY-B   PIC "99"
      03  YEAR-B   PIC "9999"
  02  PRES-HEIGHT   PIC "X(10)"
  02  PRES-PARTY   PIC "A(10)"
  02  PRES-COLLEGE   PIC "A(10)"
  02  PRES-ANCESTRY   PIC "A(10)"
  02  PRES-RELIGION   PIC "A(10)"
  02  PRES-DATE-OF-DEATH
      03  MONTH-D   PIC "A(9)"
      03  DAY-D   PIC "99"
      03  YEAR-D   PIC "9999"
  02  PRES-CAUSE-DEATH   PIC "X(10)"
  02  PRES-FATHER   PIC "A(10)"
  02  PRES-MOTHER   PIC "A(10)"
```

RECORD

各エリアに存在するレコードの具体的記述。同時に位置付け方式を指定。

```
RECORD NAME IS ADMINISTRATION
  LOCATION MODE IS VIA ADMINISTRATIONS-HEADED SET
  WITHIN PRESIDENTIAL-AREA
  02  ADMIN-KEY   PIC "XXX"
  02  ADMIN-INAUGURATION-DATE
    03  MONTH   PIC "99"
    03  DAY   PIC "99"
    03  YEAR   PIC "9999"
```

```
RECORD NAME IS STATE
  LOCATION MODE IS CALC
      USING STATE-NAME DUPLICATES ARE NOT ALLOWED
  WITHIN PRESIDENTIAL-AREA
  02  STATE-NAME   PIC "X(10)"
  02  STATE-YEAR-ADMITTED   PIC "9999"
  02  STATE-CAPITAL   PIC "X(10)"
RECORD NAME IS ELECTION
  LOCATION MODE IS VIA ALL-ELECTIONS-SS
  WITHIN PRESIDENTIAL-AREA
  02  ELECTION-YEAR   PIC "9999"
  02  ELECTION-WON-ELECTORAL-VOTES   PIC "999"
RECORD NAME IS CONGRESS
  LOCATION MODE IS CALC
      USING CONGRESS-KEY DUPLICATES ARE NOT ALLOWED
  WITHIN PRESIDENTIAL-AREA
  02  CONGRESS-KEY   PIC "XXXX"
  02  CONGRESS-NUM-PARTY-SENATE   PIC "999"
  02  CONGRESS-NUM-PARTY-HOUSE   PIC "999"
RECORD NAME IS CONGRESS-PRES-LINK
  LOCATION MODE IS VIA CONGRESS-SERVED SET
  WITHIN PRESIDENTIAL-AREA
```

PRESIDENT
STATE
CONGRESS
} CALC方式

ADMINISTRATION
ELECTION
CONGRESS-PRES-LINK
} VIA方式

図２－３４　大統領データベースのスキーマＤＤＬ(1)

```
SET NAME IS ALL-PRESIDENTS-SS
  OWNER IS SYSTEM
  ORDER IS PERMANENT SORTED BY DEFINED KEYS
    DUPLICATES ARE LAST
  MEMBER IS PRESIDENT MANDATORY AUTOMATIC
    KEY IS ASCENDING LAST-NAME IN PRES-NAME
    SET SELECTION IS THRU ALL-PRESIDENTS-SS
      OWNER IDENTIFIED BY SYSTEM

SET NAME IS ALL-ELECTIONS-SS
  OWNER IS SYSTEM
  ORDER IS PERMANENT SORTED BY DEFINED KEYS
    DUPLICATES ARE NOT ALLOWED
  MEMBER IS ELECTION MANDATORY AUTOMATIC
    KEY IS ASCENDING ELECTION-YEAR
    SET SELECTION IS THRU ALL-ELECTIONS-SS
      OWNER IDENTIFIED BY SYSTEM

SET NAME IS ALL-STATES-SS
  OWNER IS SYSTEM
  ORDER IS PERMANENT SORTED BY DEFINED KEYS
    DUPLICATES ARE NOT ALLOWED
  MEMBER IS STATE MANDATORY AUTOMATIC
    KEY IS ASCENDING STATE-NAME
    SET SELECTION IS THRU ALL-STATES-SS
      OWNER IDENTIFIED BY SYSTEM


SET NAME IS ELECTIONS-WON
  OWNER IS PRESIDENT
  ORDER IS SORTED PERMANENT BY DEFINED KEYS
    DUPLICATES ARE NOT ALLOWED
  MEMBER IS ELECTION MANDATORY AUTOMATIC
    KEY IS ELECTION-YEAR
  SET SELECTION IS THRU ELECTIONS-WON
    OWNER IDENTIFIED BY CALC-KEY

SET NAME IS CONGRESS-SERVED
  OWNER IS PRESIDENT
  ORDER IS PERMANENT IMMATERIAL
  MEMBER IS CONGRESS PRES-LINK MANDATORY AUTOMATIC
    SET SELECTION IS THRU CONGRESS-SERVED
      OWNER IDENTIFIED BY CALC-KEY

SET NAME IS PRESIDENT-SERVED
  OWNER IS CONGRESS
  ORDER IS PERMANENT IMMATERIAL
  MEMBER IS CONGRESS-PRES-LINK MANDATORY AUTOMATIC

SET SELECTION IS THRU PRESIDENT-SERVED
  OWNER IDENTIFIED BY CALC-KEY

SET NAME IS ADMINISTRATIONS-HEADED
  OWNER IS PRESIDENT
  ORDER IS PERMANENT SORTED BY DEFINED KEYS
    DUPLICATES ARE NOT ALLOWED
  MEMBER IS ADMINISTRATION MANDATORY AUTOMATIC
    KEY IS ASCENDING ADMIN-KEY
    SET SELECTION IS THRU ADMINISTRATIONS-HEADED
      OWNER IDENTIFIED BY CALC-KEY

SET NAME IS ADMITTED-DURING
  OWNER IS ADMINISTRATION
  ORDER IS PERMANENT SORTED BY DEFINED KEYS
    DUPLICATES ARE NOT ALLOWED
  MEMBER IS STATE MANDATORY MANUAL
    KEY IS ASCENDING STATE-YEAR-ADMITTED
    SET SELECTION FOR ADMITTED-DURING
      IS THRU ADMINISTRATIONS-HEADED
        OWNER IDENTIFIED BY CALC-KEY
      THEN THRU ADMITTED-DURING WHERE
        OWNER IDENTIFIED BY ADMIN-KEY

SET NAME IS NATIVE-SON
  OWNER IS STATE
  ORDER IS PERMANENT SORTED BY DEFINED KEYS
    DUPLICATES ARE LAST
  MEMBER IS PRESIDENT MANDATORY AUTOMATIC
    KEY IS ASCENDING LAST-NAME IN PRES-NAME
    SET SELECTION IS THRU NATIVE-SON
      OWNER IDENTIFIED BY CALC-KEY
```

SET

親と子の宣言。子の並び順（ソート）の指定。また、そのキーの指定。子を追加するときの親との関係づけの指定。

この場合、全レコードに対してデータベースへ格納後の並び換えを禁止している。

（PERMANENT SORTED）

ソート・キーに重複があったときの処置。

（同一ソート・キーをもつレコードの最後に位置付ける…LAST）

例、ALL-PRESIDENTS-SS

（同一ソート・キーをもつレコードの追加を禁止する…NOT ALLOWED）

例、ALL-ELECTIONS-SS

子を追加するとき、親との関係付けをどうするか。

（自動的に親子関係を確立させる

…MANDATORY AUTOMATIC）

例、ALL-STATES-SS

（コマンドを用いて個々に関係づける

…MANDATORY MANUAL）

例、ADMITTED-DURING

図２－３４　大統領データベースのスキーマＤＤＬ(2)

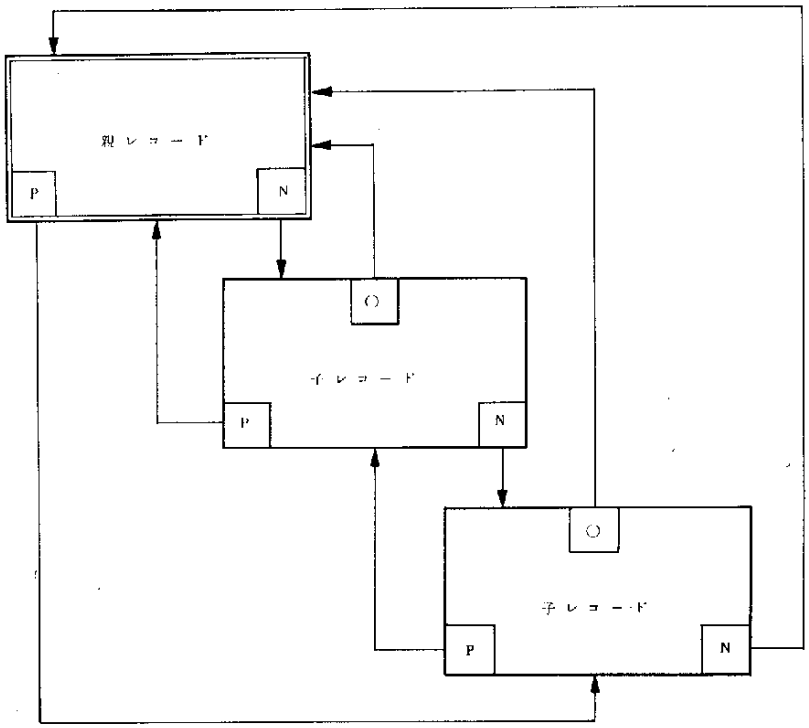


図２－３５　next, prior & owner ポインタ

D. サブスキーマＤＤＬ（Subschema Data Description Language）

スキーマＤＤＬにより構築されたデータベースを利用する個々の応用プログラムにとっては、ほとんどの場合データベース全体に関する詳細な事項は知る必要がない。従って、個々の応用プログラムにとって必要な（データベースの）部分だけを記述すれば良い。この記述に用いるのがサブスキーマＤＤＬである。このサブスキーマＤＤＬは、ＣＯＢＯＬのデータベース機能として設定されたものである。ここでは、大統領データベースのうち図２－３６に示す部分に相当するサブスキーマを記述する。従って記述に必要なレコードとセットはそれぞれ三つである。

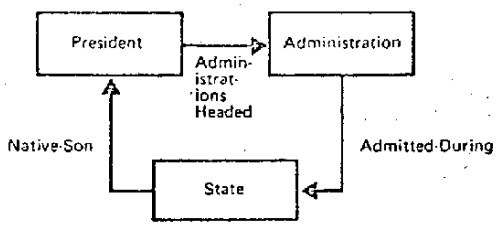


図２－３６　部分大統領データベース

図２－３７に、部分大統領データベースのサブスキーマＤＤＬを示す。

なお、サブスキーマＤＤＬのレコード部では、サブスキーマにとって必要なレコードだけを記述すれば良い。さらに、レコード内の不要な項目については一切記述する必要がない。

```
SS PRES-ADMIN-STATE-INFO WITHIN
  SCHEMA PRESIDENTIAL
  PRIVACY KEY IS 'COPY PASSWORD'.
```

⊙サブスキーマの名前付けと、プライバシ・キーの宣言を行なう。

サブスキーマ名：PRES-ADMIN-STATE-INFO

プライバシ・キー："COPY PASSWORD"

スキーマで定義したキーに対応している。

```
REALM DIVISION
RD PRESIDENTIAL-AREA.
```

⊙ここで定義するサブ・スキーマを通してデータベースにアクセスする応用プログラムが必要とするエリアを選択する。

COBOLにおいては、COBOLの予約語との関係で次のように言葉を使い分けている。

schema DDL ……AREA

sub sehema, COBOL DML ……REALM

```
SET DIVISION.
SD ALL-PRESIDENTS-SS.
SD ALL-STATES-SS.
SD ADMINISTRATIONS-HEADED.
SD ADMITTED-DURING.
SD NATIVE-SON.
```

⊙次にサブ・スキーマで必要とするセットの宣言を行なう。

```
RECORD DIVISION.
01  PRESIDENT.
    02  PRES-NAME.
        03  LAST-NAME   PIC A(10).
        03  FIRST-NAME  PIC A(10).

01  ADMINISTRATION.
    02  ADMIN-KEY   PIC XXX.
    02  ADMIN-INAUGURATION-DATE.
        03  MONTH   PIC 99.
        03  YEAR    PIC 9999.

01  STATE.
    02  STATE-NAME   PIC X(10).
    02  STATE-YEAR-ADMITTED   PIC 9999
```

⊙最後にサブ・スキーマの一部であるレコードと相当するデータ項目の名前を宣言する。

図２－３７　部分大統領データベースのサブスキーマＤＤＬ

E. DML（Data Manipulation Language）

大統領データベースに関して、スキーマとサブスキーマの定義が一応できたので、応用プログラムにとってデータベースに対するデータ操作（検索、更新など）の準備ができたことになる。データベースに実際にアクセス（操作）する際に用いる言語が以下に紹介するDMLである。COBOL言語をベースとするDMLは、既にCODASYLから提案がなされている。他の言語、例えばFORTRANにデータベース機能を付加させるための検討も行なわれており、これまでは内部資料の形でしか発表されていなかったが、1977年末には一応の完成をみたということである〔UEMU 78〕。しかし、ここではCOBOLを親言語としたDMLの記述例を図2－38に示す。

以下に示すプログラムは、次の問題（要求）に応えるために記述した例である。

『一人以上の大統領出身者のいる州を検索し、その州名と大統領名を印刷せよ。』

表2－3　DBTG提案とCOBOL JODの比較

報告書 / 機能	DBTG提案(1971)	COBOL JOD (1976)
制御コマンド	OPEN	READY
	CLOSE	FINISH
	IF	IF
検索コマンド	FIND	FIND
	GET	GET
	KEEP	KEEP
		REMONITOR
	FREE	FREE
	MOVE	ACCEPT
更新コマンド	STORE	STORE
	MODIFY	MODIFY
	DELETE	ERASE
	ORDER	ORDER
	INSERT	CONNECT
	REMOVE	DISCONNECT
特殊コマンド		USE

表2－3は、1971年にDBTGが提案したものと、1976年のCOBOL JOD〔CODA 76〕に盛り込まれているデータベース機能の対応を示したものである。

```
IDENTIFICATION DIVISION.
PROGRAM-NAME. SAMPLE-QUERY.
ENVIRONMENT DIVISION.
  identification of machine environment and
  declaration of non-data-base files
  (i.e., standard COBOL files)
```

IDENTIFICATION DIVISION
及び
ENVIRONMENT DIVISIONの宣言。

```
DATA DIVISION.
FILE SECTION.
DB   PRES-ADMIN-STATE-INFO WITHIN PRESIDENTIAL.
FD   REPORT-FILE.
     remainder of data item entries which make
     up a standard COBOL file.
WORKING-STORAGE SECTION.
77   PRESIDENT-COUNT USAGE COMPUTATIONAL   PIC 999.
77   DONE   PIC 9(5) VALUE "04021".
77   NO-MORE-SONS   PIC A(5).
77   NO-MORE-STATES   PIC A(5).
```

DATA DIVISION

サブスキーマ名の宣言。

その他必要なファイル、ワークエリアの宣言。

```
PROCEDURE DIVISION.
DECLARATIVES.
EXPECTED-ERROR SECTION.
  USE FOR DATABASE-EXCEPTION ON "04021".
EXPECTED-ERROR-HANDLING.
  EXIT.
UNEXPECTED-ERROR SECTION.
  USE FOR DATABASE-EXCEPTION ON OTHER.
UNEXPECTED-ERROR-HANDLING.
  here we would process unexpected error
  conditions.
END DECLARATIVES.
```

PROCEDURE DIVISION

データベースを参照中にエラーが発生したときの、処理ルーチンの宣言。

```
INITIALIZATION.
  READY PRESIDENTIAL-AREA.
  OPEN non-database COBOL files.
  MOVE "FALSE" TO NO-MORE-STATES.
  FIND FIRST STATE IN ALL-STATES-SS.
  PERFORM PROCESS-STATE THRU FINISH-STATE
    UNTIL NO-MORE-STATES = "TRUE".
  GO TO FINISH-UP.
PROCESS-STATE.
  MOVE 0 TO PRESIDENT-COUNT.
  IF NATIVE-SON IS EMPTY
    MOVE "TRUE" TO NO-MORE-SONS.
    ELSE MOVE "FALSE" TO NO-MORE-SONS.
  PERFORM COUNT-NATIVE-SONS
    UNTIL NO-MORE-SONS = "TRUE".
  GO TO FINISH-STATE.
COUNT-NATIVE-SONS.
  FIND NEXT PRESIDENT IN NATIVE-SON.
  IF DATABASE-STATUS = DONE
    MOVE "TRUE" TO NO-MORE-SONS
    ELSE ADD 1 TO PRESIDENT-COUNT.
FINISH-STATE.
  IF PRESIDENT-COUNT IS GREATER THAN 1
    FIND STATE CURRENT,
    GET STATE,
    write out state name and president count.
  FIND NEXT STATE IN ALL-STATES-SS
  IF DATABASE-STATUS = DONE
    MOVE "TRUE" TO NO-MORE-STATES.
FINISH-UP.
  FINISH PRESIDENTIAL-AREA.
  CLOSE non-database COBOL files.
  STOP RUN.
```

与えられた問題を解くための処理手続き。矢印はプログラムの流れを示している。

図２－３８　部分大統領データベースを用いたＤＭＬ

一般のプログラマは、ＤＢＭＳが現在データベースのどこにアクセスしているかについて特に配慮する必要がない。しかし、特に複雑な処理を必要とする場合には、現在位置を把握しておく必要がある。このために用いられるのが現在指示子（currency indicator）である。データベースには四種類の現在指示子があり、大統領データベースの場合には合計１７個の現在指示子があることになる。

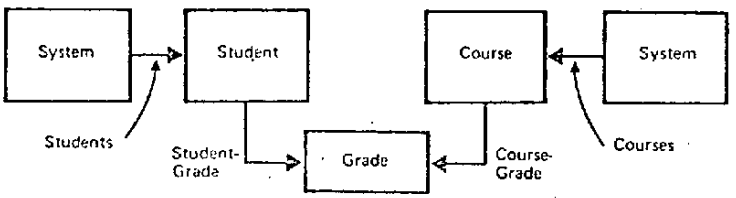
表２－４　大統領データベースの現在指示子

種　　類	個　数
実行単位	1
レコード型	6
セット型	9
エリア（レルム）	1

データベースに対する１回の操作は、四種類全ての現在指示子に影響を与える。現在指示子はＤＭＬ動詞の実行により変化するが、実行単位の現在指示子が対象となる命令は、ＧＥＴ，ＥＲＡＳＥ，ＣＯＮＮＥＣＴ，ＤＩＳＣＯＮＮＥＣＴである。

また、レコードの検索あるいは格納が行なわれると、それがエリアおよび実行単位の現在レコードとなる。また、親あるいは子が関係しているレコードあるいはセットの現在レコードとなる。

次に示す例は、セットの現在指示子が１回のアクセス毎に変わってしまうために、特別の配慮をしたものである。



問い合わせ：数学を専攻している学生の名前とその学生が専攻している他のコースを印刷せよ。

まず、数学を専攻している学生をさがして、その学生が専攻している数学以外のコースをさがして、印刷することになる。しかし、ＳＴＵＤＥＮＴ／ＣＯＵＲＳＥ のトラバースは往復するのでＣＯＵＲＳＥ-ＧＲＡＤＥセットの現在指示子は破壊される。従ってこの現在指示子を保存する必要があり、その手法としては次に示す二つがある。

①　ＡＣＣＥＰＴ文による

ACCEPT identifier FROM {realm-name / set-name / record-name} CURRENCY　　FIND record-name; DATABASE-KEY IS identifier

レルム、セット、レコード単位での現在指示子が、指示した＂identifier＂にセットされる。

② RETAINING CURRENCY句を用いる方法

この句が使えるのは、ＦＩＮＤとＳＴＯＲＥに限られるが、この句を指定するとＤＭＬを実行してもレコード、セット、レルムの各現在指示子は変わらない。ただし、実行単位の現在指示子は対象外である。

```
    MOVE 'MATH' TO COURSE-NAME.
    FIND COURSE IN COURSES USING COURSE-NAME.
**  We now have the Math record. We assume it exists.
    FIND FIRST GRADE IN COURSE-GRADE.
    PERFORM PRINT-STUDENTS-OTHER-COURSES UNTIL
      (DATABASE-STATUS = DONE).
    STOP RUN.
  PRINT-STUDENTS-OTHER-COURSES.
    PERFORM FETCH-AND-PRINT-ONE-STUDENT.
    PERFORM PRINT-THAT-STUDENTS-COURSES.
**  Now check for more students by trying to
**  FIND more grades associated with the given course
**  (Math in our example).
    FIND NEXT GRADE IN COURSE-GRADE.
  END-OTHER-COURSES.
  FETCH-AND-PRINT-ONE-STUDENT.
    FIND STUDENT OWNER.
    GET NAME OF STUDENT.
    Print it.
  END-ONE-STUDENT.
  PRINT-THAT-STUDENTS-COURSES.
**  Note the use of the RETAINING CLAUSE Below
    FIND NEXT GRADE IN STUDENT GRADE
      RETAINING CURRENCY FOR COURSE-GRADE.
    IF DATABASE-STATUS ≠ DONE
      GET GRADE
      IF COURSE IN GRADE ≠ COURSE-NAME
        Print the other course name.
      GO TO PRINT-THAT-STUDENTS-COURSES.
  END-STUDENTS-COURSES.
```

左の図２－３９は、先の問い合わせに対するプログラムである。

ここでは、コース名をＧＲＡＤＥレコードの一部としている。

図２－３９　現在指示子保存の例

F. データ保護とデータ独立

本項ではＤＢＴＧ提案におけるデータ保護とデータ独立について〔ＭＩＣＨ　７６〕を引用して解説する。

(1) データ保護

多くの利用者が大きなデータベースを共有するために蓄積されたデータの保護がＤＢＭＳの基本機能となっている。

ＤＢＭＳは二種類の保護をしなければならない。

・データの質の保護――不正な更新、破壊によってデータベースがこわれない。

・データの秘密保護――資格のないものの参照、変更、破壊からデータベースを守る。

データベースの完全性は、二つの方法によっておかされる。

・同一情報の同時更新

・不注意、不適当、悪意のある更新操作

a) 同時更新の制御

ＤＢＴＧの提案は、ロックと警告の二つのアプローチを同時更新の問題に提供している。ロックは、エリアレベルのものであり警告はレコードレベルである。エリアレベルのものは、

DMLのOPEN命令でおこなわれ、利用モード――検索、更新および排他、保護――を指定する。排他の場合には、当該エリアに関して他の実行単位の利用を禁止し、保護の場合には、他の実行単位が検索以外での参照を禁止する。エリアに対して保護、排他の指定をしなくて、更新の指定をしたときに、レコードは警告によって保護される。これは、レコードの更新操作が同時に起こったときは、これを実行単位に通知するものであって、その回復操作は利用者の責任となる。

b) 不正な更新の制御

データベースの不正な更新は、資格のない利用者が、不注意、不適当、悪意をもってデータを更新することによって発生する。DBTGのいくつかの機能は、保全性と関連したものである。例えば、on, duplicates, search key 句である。これ以外にDBTGのセットとして自動(automatic)が指定されると、親レコードが存在する場合に限ってレコードとして受け入れられる機能がある。また例に示すように、値の範囲の指定、DBAの準備した手続きが呼び出せるようになっている。

```
      .
      .
      .
POP   numeric(8)
      check is range of 0 through 20,000,000
      .
      .
      .
      .
      .
BIRTH-DATE numeric(8)
DEATH-DATE numeric(8)
  check after using CHECKED,
  BIRTH-DATE
      .
      .
      .
```

図2－40　不正更新制御の例

c) 機密保護(security)

データベースの機密保護の侵害は、権利のない利用者がデータをアクセスすることによって発生する。DBTGはプライバシを機密錠(privacy lock)、機密キー(privacy key)、データベース手続き(data base procedure)、サブスキーマで指定する。スキーマおよびサブスキーマで機密錠によって保護されたデータをアクセスしたり変更したりするためには、

機密キーを指定しなければならない。次に機密錠の例を示す。このレコードを変更するためには、応用プログラムにおいて同じキーを指定しなければならない。

```
record name is PRESIDENT
privacy lock for modify is "locked"
PRES    character 10
```

図２－４１　機密錠の指定例

d) データ保護への提言

ＤＢＴＧは、利用者のデータベースに対するアクセスは、サブスキーマを通しておこなわれ、サブスキーマで記述されていない部分のアクセスは自動的な保護がされるべきことが指摘されている。これ以外ＤＢＴＧでは、エリアアクセス、機密錠、保全性チェック（integrity check）の機能がある。このようなチェックを、データアクセスのたびに手続きの呼び出しやチェックをおこなうのは効率が悪いという議論がある。

(2) データ独立

ＤＢＭＳの変化に対する適応性は、それがおこなうデータの独立性に従属する。基本的には、データベースの利用者の見方を他のもの――データアクセス機構、蓄積されたデータベース、蓄積媒体、他の利用者のデータの見方――から切り離すことである。

ここでは二つのタイプのデータ独立を論ずる：

a) 物理的なデータ独立：蓄積構造の変化に対する免疫性を示し、効率化のため蓄積構造の変更、新しいハードウェアへの変更に耐え得る。

b) 論理的なデータ独立：データモデルの変更に対する免疫性を示し、データ属性の追加、削除、構造を他に移すことに対して耐え得る。

ＤＢＴＧの提案は、データ独立の不足に対して批判されてきた。利用者は、ＤＢＡのおこなった物理的なレコードの配置に対して種々の気をくばらなければならない。このようなことは性能を上げるうえでは良いけれども、データ独立を達成するための注意をもはらわなければならない。ＦＩＮＤ文には二種類ある。一つは関係によって探すものであり、もう一つは、最初にどのように格納されたかによって探すものである。このことがデータ独立を制限する。なお、アクセスパスの変化に対する考慮を利用者はしなければならない。

Taylor の提案したマクロプロセッサは〔ＴＡＹＬ　７４〕、性能の向上のためにＤＢＴＧ機能を保持しながら、データ独立をあたえる方法を示している。ある程度の論理的なデータ独立は、スキーマ／サブスキーマの機構で実現されている。これは、スキーマに新しいレコードや項目が追加されたとしても、サブスキーマに追加しなければ、利用者プログラムに影響はない。

G. 性能

ＤＢＴＧの利用に対する主たる基準の一つは、応答時間、実行時間、蓄積容量の面からの効率

である。この目標は、ＤＢＭＳが多くの利用者、異なったモードで使われることを考えあわせれば達成するのが困難である。また、性能の向上は、おうおうにして他のデータベースの目標と矛盾する。例えば、種々の利用者の見方を定義するには、それぞれに対してサブスキーマが指定される。サブスキーマ／スキーマの写像、データ操作機能、データ保護機能は、実行時間を増加させる。必然的にデータ独立は、コストの増加をなしに達成はできない。

ＤＢＴＧ　ＤＭＬのような手続型言語の利用者は、スキーマにもとづいた効率のよいアクセスができる。このため、効率が問題になる場合には、ハイレベルな非手続的なものよりも性能がすぐれているといえよう。

H. ＣＯＤＡＳＹＬ　ＤＢＴＧ提案の進展

ここでは、〔ＵＥＭＵ　７８〕を引用しＣＯＤＡＳＹＬ　ＤＢＴＧ提案の進展状況を把握する。

(1) 背景と概要

ＣＯＤＡＳＹＬにおける言語仕様の保守改訂作業は二つの委員会が担当していた。つまり、ＤＤＬはＤＤＬ委員会、ＣＯＢＯＬデータベース機能についてはＣＯＢＯＬ委員会がそれぞれ作業していた。ＡＮＳＩ（アメリカ規格）では、最近ＡＮＳＩ／ＣＯＢＯＬの再改訂やＡＮＳＩ／ＤＤＬを作る方向が固まるなど新しい動きを示している。このような影響をうけてか、ＣＯＤＡＳＹＬの二つの委員会は、１９７７年末を区切りとして積極的な言語仕様の改訂を行なった。また、１９７７年中にＦＯＲＴＲＡＮデータベース機能が完成した模様である。

(2) ＤＤＬの改訂

a　削除されたＤＤＬ機能

a.1.　サブスキーマ側の担当であるなどの理由で削除された句は次の二つである。

①　ＴＥＭＰＯＲＡＲＹ句、ＡＲＥＡ　ＩＳ　ＴＥＭＰＯＲＡＲＹ（一時句）

領域が一時的であることを指定する機能。

②　ＥＮＣＯＤＩＮＧ／ＤＥＣＯＤＩＮＧ句（符号化／復元句）

特別な変換を必要とするデータ項目を検索あるいは更新するたびに、つねに実行すべき手続きを指定する機能。

a.2.　ＤＤＬでは、ＤＢＭＳにデータの使われ方のみを指定し、データの表現方法を直接すべきでないという理由で削除された句。

①　ＬＯＣＡＴＩＯＮ句（配置句）

ＬＯＣＡＴＩＯＮ　ＭＯＤＥ　ＩＳ　{ ＤＩＲＥＣＴ | ＣＡＬＣ～ＵＳＩＮＧ～ＤＵＰＬＩＣＡＴＥＳ ～〔ＮＯＴ〕　ＡＬＬＯＷＥＤ | ＶＩＡ～ＳＥＴ | ＳＹＳＴＥＭ }

ＤＢＭＳがレコードに割り当てるデータベース・キー値を制御する機能。

② PRIOR句（逆方向句）

SET IS PRIOR PROCESSABLE

このセットに属する子レコードを、順方向と同等な効率で逆方向にも処理できるような手法を、ＤＢＭＳが優先的に選択する機能。

③ LINKED TO OWNER句

④ SEARCH句（探索句）

SEARCH KEY IS ～

[USING { CALC / INDEX 〔NAME ～ 〕 / PROCEDURE ～ }]

DUPLICATES 〔NOT〕 ALLOWED

a.3. 追加され、注目すべき句

STRUCTURAL句（構造制限区）

セットにおける親レコードと子レコードとの間で指定したデータ項目（群）の値が同一であることを条件づける句。

b. ＤＳＤＬの分離

ＡＮＳＩ／Ｘ３／ＳＰＡＲＣのデータベース・スタディ・グループによる三層スキーマの提唱の影響をうけてか、ＣＯＤＡＳＹＬのＤＤＬ委員会もスキーマＤＤＬを二分させて三層のスキーマを考えている。

・サブ・スキーマ（個別データ構造）

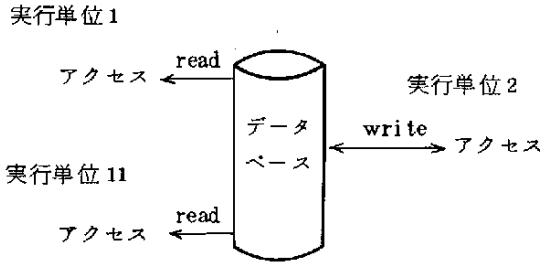
・スキーマ（固有データ構造）

・ストレージ・スキーマ（記憶構造）

新しい記憶構造記述のためにＤＳＤＬ（Data Storage Description Language）を設け、イギリス計算機学会のＤＢＡＷＧ（Data Base Administration Working Group）が開発にあたっており、１９７７年中に一つのＤＳＤＬ仕様をまとめている。

c. 一意な識別子　～データベース・キーの概念変更～

c.1. 従来のデータベース・キー



実行単位1と実行単位11でアクセスしたレコードの内容が同一である保証がない。

c.2 新しいデータベース・キー

① データベース・キーをＤＢＭＳと実行単位との間で一意に定める。

② 実行単位が生きている間だけ有効である。

③ データベース・キーの内部表現は作成者が定める。

④ データベース・キーを用いても能率がよいかどうかわからない（作成者による）。

⑤ TYPE IS DATA−BASE−KEYの削除。

c.3. レコード・キーの導入

① レコード中の任意のデータ項目（群）をレコード・キーと指定する機能。

② レコード・キーは、そのレコード型内で有効で複数個設けてもよい。

③ レコードをキーの論理的順番に並べるためのレコード順序キーも指定できる。

c.4. 新旧キーの対応

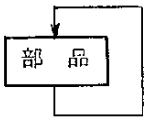
旧データベース・キー ⇒ 新レコード・キー

旧データベース・キーによる順序づけ ⇒ 新レコード・キーによる順序づけ

d. サイクルの禁止解除

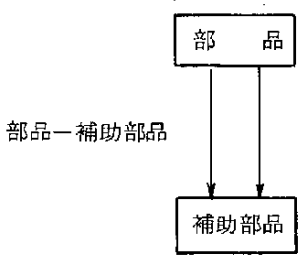
セットにおいて、親レコード型と子レコード型が同一である場合も認めた。

～ 再帰的親子集合（recursive set）



構成要素

部品展開のように再帰的データ構造が避けられないケースでは、これまで次のように考えて処理していた。



(3) ＣＯＢＯＬデータベース機能における改訂

a. 同時実行の制御と保管表

従来のＣＯＢＯＬデータベース機能のうち、同時更新制御は大きな問題の一つであった。今回の改訂では、レコード錠（record locks）の概念を導入し、レコードごとに更新錠

(update locks)と選択錠(selection locks)を設定している。この二つの錠と保管表の関係を図２－４２に示す。図２－４２では同時実行単位として"１"と"２"を考えている。実行単位１がＳＴＯＲＥ，ＭＯＤＩＦＹ，ＣＯＮＮＥＣＴ，ＤＩＳＣＯＮＮＥＣＴの何れかの命令を実行したレコードに対して、更新錠をセットすると、実行単位２は、ＦＩＮＤやＥＲＡＳＥ，ＭＯＤＩＦＹ，ＣＯＮＮＥＣＴ，ＤＩＳＣＯＮＮＥＣＴの命令の実行が不可能となる。さらに、現在指示子が指すレコードに対して選択錠をセットすると、別の同時実行単位は、ＥＲＡＳＥ，ＭＯＤＩＦＹ，ＣＯＮＮＥＣＴ，ＤＩＳＣＯＮＮＥＣＴの命令の実行が不可能となる。なお、ＫＥＥＰコマンドにより保管表に登録されたレコードにも同様の処置がとられる。

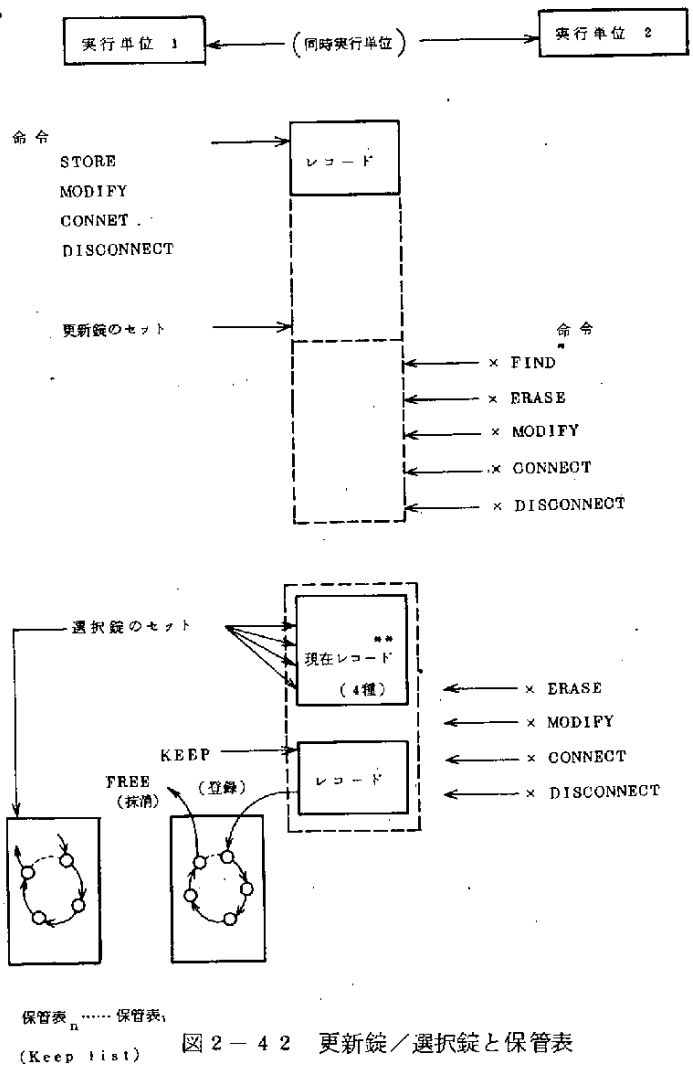


図２－４２　更新錠／選択錠と保管表

＊；×　ＦＩＮＤ…ＦＩＮＤ命令が実行不可能であることを示す。

＊＊；現在レコード…現在指示子が指しているレコードで４種類の指示子がある。

b．保管表による再呼出し

今回の改訂では、直接データベース・キーを用いた高速アクセスを行なうことはできないが、保管表内にスタックした値を利用することにより、データベース・キーの値を有効利用することができる。このとき、保管表内のＦＩＲＳＴ／ＬＡＳＴの指定が可能である。なお、保管表はＦＩＦＯ方式でスタックされ、実行単位別に何種類でもセットできる。

$$\left\{\begin{array}{l}\text{FIRST}\\\text{LAST}\end{array}\right\}\quad \text{WITHIN KEEP-LIST-NAME}$$

c．ロールバック機能と静止点

ＣＯＤＡＳＹＬでは三つの静止点（quiet point ）を設定している。

・実行開始時点

・ＣＯＭＭＩＴ命令実行直後

・ＲＯＬＬＢＡＣＫ命令実行直後

ＣＯＭＭＩＴ命令が実行されると次の処理が行なわれる。

・更新錠の解放

・選択錠の解放

・保管表の空状態

・現在指示子の空状態

ＲＯＬＬＢＡＣＫ命令を実行すると、直前のＣＯＭＭＩＴ命令を出した静止点の状態までデータベースの内容が戻される（図２－４３）。なお、途中のバックアップ情報の保管方法についてＣＯＤＡＳＹＬは一切触れておらず作成者にまかせている。

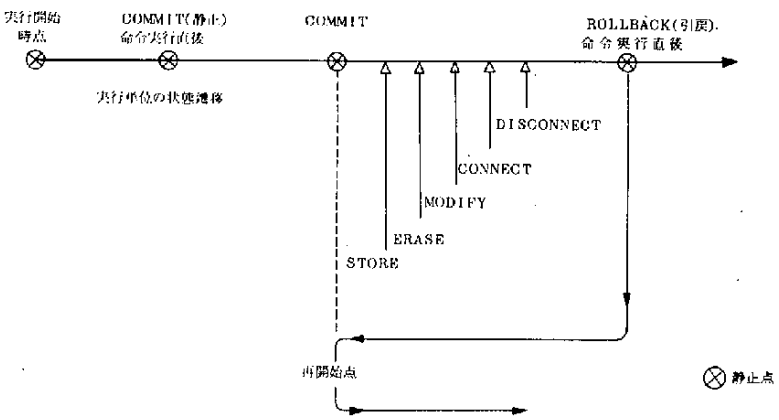


図２－４３　ロールバック機能と静止点

d．レコードキーによるアクセス

ＤＤＬにおいてデータベース・キーに代わる新しい概念としてレコード・キーが検討された。またＬＯＣＡＴＩＯＮ　ＭＯＤＥが削除されたので、ＦＩＮＤ命令のレコード選択式に

改訂があった。

① データベース・キーを用いる場合

```
FIND レコード名－１；DATABASE－KEY IS 一意名－１
                                  データベース・キーをセットする項目名
FIND {FIRST|NEXT} レコード名－１ 〔USING {一意名－１} …〕
```

② CALCキーを用いる場合

```
FIND {ANY|DUPLICATE} レコード名－２
FIND {ANY|DUPLICATE} レコード名－２ USING キー名－１
```

⊙キー名－１のデータ項目はレコード名－２のレコード型のレコード・キーである。

(4) まとめ

今回の改訂をまとめてみると次の三点がいえよう。

・データベース・キーの整理

・三層スキーマへの傾向

・関係モデルの影響

なお、セット（親子集合）の概念を全く使わないデータベースでさえも、この言語仕様で扱いうる感触がでてきている。

I. さいごに

現在CODASYL型と呼ばれているDBMSは多数存在する。1969年に、DBTGから最初の提案があって以来DBTG自身による継続的な検討は無論、外部からも賛否両論、数多くの意見やレポートが出されている。現存するCODASYL型のDBMSは、基本的には同じデータモデルを設定しているにもかかわらず、個々のDBMSにより若干異なる文法を採用していることも事実である。また、関係モデルという新しいデータベースの概念が出現し注目を集めており、それは特に分散型データベースに適したデータモデルとも言われている。しかし、最近になってようやく実用システムの例が見られるようになった程度である。一方、CODASYLにおいても、スキーマの三層化傾向を示したり、FORTRANデータベース機能を完成するなど、最近活発な動きがみられる。このような事からみても、今後ともCODASYL型のDBMSの動向を無視することは出来ないであろう。

2.2.3 関係モデル

Coddによって提案された関係モデル（relational model）〔CODD 70〕は、従来の階層型モデル（2.2.1項を参照）と網型モデル（2.2.2項を参照）に対して、高度のデータ独立性と、高

水準な非手続的データアクセス手段のユーザへの提供とを特徴としている。データの独立性とは、ユーザ（アプリケーション）が、データの物理的格納方法とアクセス方法（即ち、アクセス・パス）の詳細及びその変化へ独立であることを意味する。関係モデルにおいて、データは値としてのみ表現され、システムの内部表現とは独立である。値の間の論理的関係は、表表現によって表わせる。ユーザは、表表現をもとにして、非手続的言語によって、必要なデータをアクセス出来る。

関係モデルの他の大きな特徴は、抽象代数に基礎をおいていることである。関係モデルにおいて表として表わせられる関係とは、共通性質を持った値の集合（領域と呼ばれる）上で定義される数学的関係である。値の集合間の依存性を関数従属性の概念で表わすことによって、関係の種々の問題を、数学的に構文的に厳密に研究することを可能とした。関係の更新に対する異常性を除去するための関係の正規化（normalization）も、この様な関数従属概念に基づいている。関係モデルが、数学的概念に基づいていることは、一方では、データの意味の問題を生じさせている。

１９７０年の Coddによる関係モデルの提案以来、研究は、主として理論面についてなされてきた。関係モデルを持った幾つかのＤＢＭＳも開発されはしたが、実験的レベルのものであった。これらの開発における問題点は、十分な効率、機能とを備えたＤＢＭＳをいかに実現するかということである。実現の大きな要因として、連想メモリ等のハードウェア技術の進歩の必要性が上げられている。近年、ＲＡＰＥＳ（ＨＩＳ），ＭＡＧＮＵＭ（ＴＹＭＳＨＡＲＥ），ＮＯＭＡＤ（National ＣＳＳ）等の商用ＤＢＭＳが発表され〔SAKA ７８〕、今後５年ないし７年以内（１９８３－８５）には、本格的な関係ＤＢＭＳが実用化されるといわれている〔CURT ７６〕。

本項では、データモデルとしての関係モデルを議論する。関係モデルにおける関係、関数従属等に関する基本概念をＡで述べる。Ｂでは、関係の正規化を、第３及び第４正規形を中心に議論する。Ｃでは、関係データベースの論理設計について論じる。次に、問合せ言語をＤで述べる。Ｅでは、関係ＤＢＭＳの実際例について述べる。最後にＦでデータモデルとしての関係モデルの問題を、意味論を中心に議論する。

A. 基本概念

ここでは、関係モデルにおける基本概念の定義を行なう。

関係モデルにおける基本要素は、値である。＂Blue＂，＂Red＂，＂White＂，＂John＂，＂Deter＂，＂Tom＂等は値である。これらの値は、＂Colour＂，＂Person＂種々のクラスへ分類出来る。このクラス、即ち共通性質を持った値の集合を領域（domain）と呼ぶ。

関係（relation）とは、幾つかの領域上で定義される数学的関係である。形式的に関係Ｒは、下記の様に与えられる〔CODD ７１ａ，ROSE ６８〕。

$$R \triangleq \text{any subset of } \left\{ (d_1, d_2, \cdots, dn) \mid d_1 \in D_1, d_2 \in D_2, \cdots, dn \in Dn \right\}$$

ここで、D_1，D_2，…，Dnは領域である。ｎを関係の次数（degree）と呼び、次数ｎの関係をｎ項関係（n－ary relation）と呼ぶ。特に、ｎが２の場合には、２項関係と呼ぶ。（d_1，d_2，…，dn）を、組（tuple）と呼ぶ。関係は、図２－４４のような表表現によって表わすと便利である。関係を構成する組の集合は、組が挿入、消去、修正されるに従い、時間とともに変化

する。この点は、データベースの関係と、数学的関係の相違である。このため関係構造を関係スキーマと呼び、組の集合を関係と呼び、明確に分離している場合もある〔CADI 75, BERN 76, FAGI 77c〕。

図2－44において、YEAR, NAMEは、関係ELECTIONの領域である。この関係内で、同一の領域NAMEへ属する第2行と第3行は、異なった意味を持っている。前者は、大統領選挙における勝者を表わし、後者は敗者を表わしている。関係内の行と列の順番に依存させないために、関係内における領域の役目（role）の概念を設ける。役目は、関係内で同一領域からの値を持つ行の混乱を防ぎ、関係内の領域の役目を明らかにするためのものである。

NAMEは、役目WINNER、LOSERによって修飾されている。関係モデルにおける属性名（attribute name）とは、領域名と役目名とを結合したものである〔CODD70, CHEN76〕。属性名は、同一領域からの値を持つ行を区別するためのものであり、関係における行の意味を表わすものではない。例えば、2つの関係において同一の属性名（NAME）をもつ行があっても、同一の意味を表わさない。1方の関係では船のNAMEを表わし、他方では人のNAMEを表わしているかもしれない。

ELECTIONS

属性	YEAR	WINNER-NAME	LOSER-NAME
役目		WINNER	LOSER
領域	YEAR	NAME	NAME
組	1952	Eisenhower	Stevenson
	1956	Eisenhower	Stevenson
	1960	Kennedy	Nixon
	1964	Johnson	Goldwater
	1968	Nixon	{Humphrey Wallace}
	1972	Nixon	McGovern

図2－44　ELECTIONS関係

関係モデルにおけるデータベースとは、領域の有限集合$\{Di\}$上で定義され、かつ時間とともに変化する関係の有限集合である〔CODD 71a〕。

関係モデルにおいて重要な概念である関数従属性（functional dependency）について考える。関係Rの属性Aと、同一の関係Rの属性Bとの間の関数従属性は、次のように定義される。"Aの各々の値が、これと関連したBの値を1つ以上持たないなら、属性Bは、属性Aに関数従属する。" この関数従属関係は、A→Bと記述する。この場合属性Aを決定子（determinant）と呼ぶ。この関係は数学的な関数とは異なり、時間とともに変化するものである。このため、先に述

べた関係スキーマと関係との場合のように、数学的関数関係へ対して関数従属と呼ぶ〔ＢＥＲＮ７７〕。

次に、Ａを幾つかの属性の集合の場合を考える。属性Ｂは、属性Ａに関数従属するが、属性Ｂのどの副集合に対しても関数従属しない時、属性Ｂは属性Ａに全関数従属（full functional dependency）するという。以後、関数従属と書くことによって、全関数従属を意味させる。

推移従属について考える。Ａ，Ｂ，Ｃを、関数Ｒの相異なる属性とする。この時、Ａ→Ｂ，Ａ↛Ｂ，Ｂ→Ｃの関係が存在する時、Ａ→Ｃ，Ｃ↛Ａが成り立てば、属性Ｃは属性Ａに推移従属（transitive dependency）するという。

上記した、属性間の関数従属性は、後述する関係の正規化における主要な概念である。関数従属Ａ→Ｂは、属性（の集合）Ａの１つの値（値の集合）と、属性Ｂの１つの値との関係であることは重要である。これに対して、Fagin〔ＦＡＧＩ　７７ｂ〕による多値従属（multi-value dependency）は、属性の１つの値に対する他の属性内の複数個の値の従属関係である。多値従属については、Ｂの正規化において詳論する。

次にキー（key）について考える〔ＤＡＴＥ　７７〕。主キー（primary key）とは、関係の組を一意に識別出来る関係内の属性又は属性の集合である。関係の全ての属性の組は、この関係の組を識別出来るという意味で、あらゆる関係は常に主キーを持つと言える。主キーにおける重要な性質は、非冗長である点である。即ち、主キーを構成する属性のどの副集合も、関係の組を一意には識別出来ない。図２－４４の例で、（ＹＥＡＲ，ＷＩＮＮＥＲ－ＮＡＭＥ）も関係ＥＬＥＣＴＩＯＮＳの組を一意に識別出来るが、ＹＥＡＲだけでも同様に組を一意に識別出来る。このため（ＹＥＡＲ，ＷＩＮＮＥＲ－ＮＡＭＥ）は、主キーではなく、ＹＥＡＲが主キーである。関係は、関係内の組を一意に識別出来る属性の集合を複数持つ場合がある。即ち、主キーを含む集合をキー候補（candidate key）と呼ぶ。（ＹＥＡＲ，ＷＩＮＮＥＲ－ＮＡＭＥ）はキー候補である。全てのキーは、キー候補でもある。関係の主キーを求めることは、関係の属性間の関係従属をもとにして、Ｋ→Ω（Ωは、関係Ｒ内のＫ以外の全ての属性の集合）となる最小の属性数を持つＫをみつけることである。

次に、外来キー（foreign key）を考える。関係Ｒ１の属性Ａは、Ｒ１の主キーではないが、他の関係Ｒ２の主キーである時、属性Ａを外来キーと呼ぶ。図２－４７の例で、属性ＷＩＮＮＥＲ－ＮＡＭＥは、関係ＥＬＥＣＴＩＯＮＳ－ＷＯＮの主キーではないが、関係ＰＲＥＳＩＤＥＮＴＳでは主キーである。よってＷＩＮＮＥＲ－ＮＡＭＥは外来キーである。

主キーを構成する属性を主属性（primary attribute）、他の属性を非主属性（non-primary attribute）と呼ぶ。

B. 正規化

Ａで述べた基本概念をもとに、ここでは関係の正規化（normalization）を考える。正規化の主要な目的は、望ましくない挿入、更新、消去に対して、関係の集合をまもることである〔CODD　７１ａ〕。正規形（normal form）として、第１、第２、第３、第４の４つがある。以下に

各々について論じる。

① 第1正規形（1NF）

関係へ対する最初の正規化である。関係Rの全ての値が、不可分な原子的データである時、関係Rは1NFであるという。図2－44の関係は1NFではなく、非正規形である。何故なら、1968年のLOSER－NAMEとして、値の対｛Humphrey, Wallace｝をもっているからである。関係の各属性の値が不可分な原子的データでなければならない点は、関係モデルにおいて第一に重要な点である。

② 第2正規形（2NF）

2NFにおいては、キーの概念が導入される。関係Rが2NFであるとは、次の2つの条件が満足された場合に言われる。

i) Rは、1NFであること。

ii) Rの全ての非主属性は、Rの主キーへ関数従属していること。

2NFの例を、図2－45に示す。関係ELECTIONS－WONの関数従属関係を図2－46に図示する。この図から解かるように、主属性YEARへ対して非主属性WINNER－NAME，WINNER－VOTES，PARTY，HOME－STATEは関数従属している。

ELECTIONS-WON

YEAR	WINNER-NAME	WINNER-VOTES	PARTY	HOME-STATE
1952	Eisenhower	442	Republican	Texas
1956	Eisenhower	447	Republican	Texas
1960	Kennedy	303	Democrat	Mass.
1964	Johnson	486	Democrat	Texas
1968	Nixon	301	Republican	Calif.
1972	Nixon	520	Republican	Calif.

図2－45　ELECTIONS－WON関係

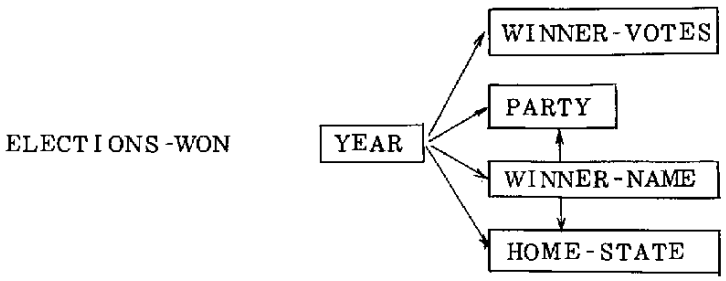


図2－46　ELECTIONS-WON関係の関数従属

２ＮＦにおいて、図２－４６でＷＩＮＮＥＲ－ＮＡＭＥ→ＰＡＲＴＹのように非主属性間に関数従属関係が存在し得る。即ち、主属性と非主属性との間に推移従属関係が存在し得る。このことは、次のような格納演算における問題をもたらしてしまう。例えば、１９６８年のＰＡＲＴＹをDemocratへ更新したとすると、１９６８年の大統領Nixonの出身政党はDemocratであり、１９７２年にはRepublicanとなってしまう。この更新の異常は、ＹＥＡＲ→ＰＡＲＴＹのＷＩＮＮＥＲ－ＮＡＭＥを介しての推移従属関係の存在があるからである。２ＮＦの更新異常を防ぐものとして、更に第３正規形（３ＮＦ）の導出を行なう。

③　第３正規形（３ＮＦ）

３ＮＦは、２ＮＦにおける様な更新異常を防止するためのものである。３ＮＦに対しては、幾つかの定義がある。Coldによる定義〔ＣＯＤＤ　７１ａ〕では、

1)　関係Rは、２ＮＦであり、

2)　あらゆる非主属性は主キーに対して推移従属しないならば、関係Ｒを３ＮＦと定義している。この定義は、２ＮＦにおいて述べた推移従属関係を、関係内から除去しようとするものである。Codd の３ＮＦは、主キー及び推移従属との概念に基づいている。

ＢＣＮＦ（Boyce／Codd Normal Form）〔ＣＯＤＤ　７４〕として知られるものは、次の定義である。"正規化された関係Ｒの全ての決定子（determinant）がキー候補であるならば、関係Ｒは第３正規形である。"

この両方の定義は、主キーが１つの属性から構成されているならば等価である〔図2—47〕。問題は、Dateによって指摘されているように〔ＤＡＴＥ　７７〕、２つの重複したキー候補を所有する関係において生ずる。例としてDateに示された〔ＤＡＴＥ　７７〕関係ＳＪＴ（Ｓ，

ELECTIONS-WON

YEAR	WINNER-NAME	WINNER-VOTES
1952	Eisenhower	442
1956	Eisenhower	447
1960	Kennedy	303
1964	Johnson	486
1968	Nixon	301
1972	Nixon	520

PRESIDENTS

NAME	PARTY	HOME-STATE
Eisenhower	Republican	Texas
Kennedy	Democrat	Mass.
Johnson	Democrat	Texas
Nixon	Republican	Calif.

図２－４７　第３正規形の例

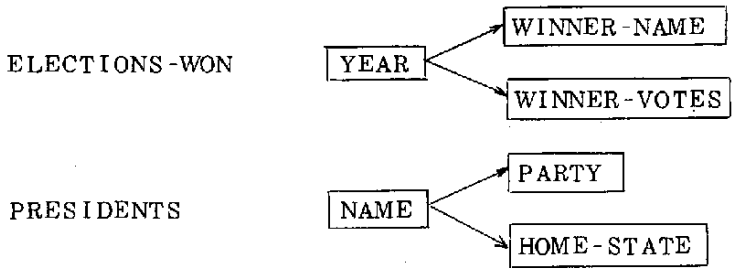


図２－４８　図２－４７の関係の関数従属

Ｊ，Ｔ）を考えよう〔図２－４９〕。ここでＳは学生名、Ｊは科目、Ｔは教授名を表わしてい

SJT

S	J	T
Smith	Math	Prof. White
Smith	Physics	Prof. Green
Jones	Math	Prof. White
Jones	Physics	Prof. Brown

図２－４９　ＳＪＴ関係

る。この関係の関数従属構造は、図２－５０のようになる。関係ＳＪＴは、２つの互いに重複

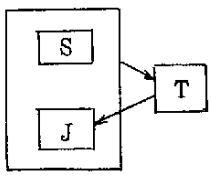


図２－５０　ＳＪＴの関数従属性

したキー候補（Ｓ，Ｊ）と（Ｓ，Ｔ）とを持つ。この関係は、Ｃｏｄｄの３ＮＦ定義に従がうと、推移従属性が存在しないので３ＮＦである。しかし、ＢＣＮＦに従がうと、３ＮＦではない。この点を議論するために、関係ＳＪＴの更新をみてみる。例えば、Ｊｏｎｅｓ がPhysics を学習している情報を消去すると、Ｂｒｏｗｎ 教授がPhysics を教えている情報も同時に失なってしまう。問題は、Ｔは決定子（Ｔ→Ｊ）ではあるが、キー候補ではないことによっている。この点は、ＢＣＮＦが関係ＳＪＴを３ＮＦではないとする理由である。

上述した例より、ＢＣＮＦは下記の点から、Coddの３ＮＦより優れているといえる〔ＤＡＴＥ７７〕。

1) 主キーの概念を持たない。
2) 推移従属性の概念を持たない。
3) 全関数従属性の概念を持たない。
4) より直観的である。

関係モデルにおける今までの議論は、属性間の依存性として、関数従属性に関してなされてきた。関数従属は、属性の１つの値又は幾つかの属性の値の集合が、他の属性の値の１つを決定することを意味している。よって関数従属によって、１つ又は複数の値の集合が、他の属性内の複数の値との間の従属関係を表わせない。例えば、関数従属ＥＭＰＬＯＹＥＥ→ＳＡＬＡＲＹは、１人の社員は、１つの給料を持つことを意味している。しかし、１人の社員が複数の子供を持つという事実を関数従属によっては表わせない〔ＳＣＨＭ７５〕。この様な関係を表わすためには、関数従属以外の従属性が必要となる。Fagin〔ＦＡＧＩ７７ａ，ｂ，ｃ〕とZaniolo〔ＺＡＮＩ７６〕は各々独立に多値従属（multivalue dependency）という新たな従属関係の概念を導入した。この概念に基づく新しい正規形を第４正規形と呼んでいる。

④ 第４正規形（４ＮＦ）〔ＦＡＧＩ７７ａ〕

関数従属について議論されてきた正規化に対して４ＮＦでは多値従属という新しい依存性の概念が導入されている。

関数従属とは、ある属性の１つの値と、他の属性の１つの値との従属性である。例えば、図２－４７に示されている関係ＥＬＥＣＴＩＯＮＳ－ＷＯＮ（ＹＥＡＲ，ＷＩＮＮＥＲ－ＮＡＭＥ，ＷＩＮＮＥＲ－ＶＯＴＥＳ）における関数従属ＹＥＡＲ→ＷＩＮＮＥＲ－ＮＡＭＥは、１つの年に、１人の選出された大統領がいることを意味している。

多値従属を説明するために、図２－５１の関係ＥＭＰ（ＥＭＰＬＯＹＥＥ，ＣＨＩＬＤ，ＳＡＬＡＲＹ，ＹＥＡＲ）を考えよう。この関係は、各々の社員の子供と給料の記録を表わしている。関係ＥＭＰは、関数従属構造を持たないが、ＣＨＩＬＤは、ＥＭＰＬＯＹＥＥに多値従属している。即ち、１人の社員は、ただ１人だけの子供を持つことはないが（即ち、ＥＭＰＬＯＹＥＥ↛ＣＨＩＬＤ）、各々の社員は、１人以上の子供を子供つことを意味している。この関係をＥＭＰＬＯＹＥＥ→→ＣＨＩＬＤと表わし、ＥＭＰＬＯＹＥＥがＣＨＩＬＤを多重決定（multi determine）すると読ませる。この多値従属性は、Faginとは独立にZaniolo〔ＺＡＮＩ７６〕によって見つけ出されている。

次に、多値従属の形式的定義を行なう。関係としてＲ（X_1，X_2 … ，Xm，Y_1，Y_2，… ，Yn，Z_1，… ，Yr）を考える。関係Ｒは、ｍ＋ｎ＋ｒ行の関係である。ここで、

$$X \triangleq \{X_1, X_2, \cdots, Xm\}$$
$$Y \triangleq \{Y_1, Y_2, \cdots, Yn\}$$
$$Z \triangleq \{Z_1, Z_2, \cdots, Zr\}$$
$$R(X,Y,Z) \triangleq R(X_1, X_2, \cdots, Xm, Y_1, \cdots, Yn, Z_1, \cdots, Zr)$$
$$x_i \in Xi, \quad y_i \in Yi, \quad z_i \in Zi$$
$$x \triangleq \{x_1, x_2, \cdots, x\}$$
$$y \triangleq \{y_1, y_2, \cdots, y\}$$
$$z \triangleq \{z_1, z_2, \cdots, z\}$$

及び $Yxz \triangleq \{y \mid (x,y,z) \in R\}$ とする。X,Y,Zは、互いに素な集合である。この時、多値従属X→→Yは、次の条件が成り立つならば、関係R（X，Y，Z）において存在する。

"Yxzは、xにのみ依存する。即ち、集合YxzとYxz'が空でないようなx，z，z'へ対してYxz＝Yxz'（$\{y \mid (x,y,z) \in R\} = \{y \mid (x,y,z') \in R\}$）である。"

図２－５１の関係において、

$$CHILD_{\underline{GAUS},\$40K,1975} = CHILD_{\underline{GAUS},\$50K,1976} = \{Gwendoly, Greta\}$$

であるので、EMPLOYEE→→CHILDは関係EMPにおいて成り立つ。

関数従属とは、多値従属の一種である。関数従属Xi → Yとは、Yxzがたった１つのxiに

EMPLOYEE	CHILD	SALARY	YEAR
Hilbert	Hubert	$35K	1976
Hilbert	Hubert	$40K	1976
Gauss	Gwendolyn	$40K	1975
Gauss	Gwendolyn	$50K	1976
Gauss	Greta	$40K	1975
Gauss	Greta	$50K	1976
Pythagoras	Peter	$15K	1975
Pythagoras	Peter	$20K	1976

図２－５１　EMP関係

依存していることを意味している。

X，Y，Zが互いに素であり、X→→Y、Y→→Zであるならば、X→→Zは成り立つ。即ち、X，Y，Zの排他性の条件下で、多値従属関は推移性を持つ。

X→→Yは、X，Y，Zが互いに素であることを仮定していたが、素でない場合への拡張も行なわれている〔BEER ７７〕。

次に、多値従属概念を用いて、４ＮＦを導入する。多値従属の重要な性質は、関係が何の情報の損失もなく、その関係の射影である２つの関係へ分割出来るための必要十分条件である〔ＦＡＧＩ　７７ａ，ｃ〕。ここでいう情報の無損失性とは、最初の関係は分割された２つの関係の結合によって再生され得ることである。この関係の分割は次の様である。Ｘ，Ｙ，Ｙを関係Ｒの属性名の互いに素な副集合とする。又、Ｘ，Ｙ，Ｚの和は、関係Ｒの属性名の全集合である。もし関数従属Ｘ→Ｙ又は多値従属Ｘ→→Ｙが関係Ｒ（Ｘ，Ｙ，Ｚ）へ対して成り立つならば、関係ＲをR_1（Ｘ，Ｙ）とR_2（Ｘ，Ｚ）へ情報を損なうことなく分割出来る。情報の無損失性は、関係Ｒ（Ｘ，Ｙ，Ｚ）は、R_1（Ｘ，Ｙ）とR_2（Ｘ，Ｚ）の結合であることを意味している。

４ＮＦを定義するために、自明多値従属（trivial multivalue dependency）を定義する。ＸとＺを互いに素な関係Ｒの属性名の集合とする。ＹがＸ又はＺの副集合である時、多値従属Ｘ→→Ｙは、関係Ｒ（Ｘ，Ｚ）へ対して常に成り立つ。この様な従属を、自明多値従属という。例えば、３つの属性Ａ，Ｂ，Ｃから成るすべての関係Ｒ（Ａ，Ｂ，Ｃ）に対して、自明多値従属$\{A, B\} \rightarrow\rightarrow C$が成り立つ。これを用いて、４ＮＦの定義を行なう。ＸとＹは、関係Ｒの属性の副集合であるとする。この時関係Ｒへ対して"自明でない多値従属"Ｘ→→Ｙが成り立つ場合常に、関数従属Ｘ→Ａが、関係Ｒの全ての属性Ａへ対して成り立つならば、関係Ｒは４ＮＦである〔ＦＡＧＩ　７７ｃ〕。これは、全ての従属関係はキーに基づいていることを意味している。又、４ＮＦの関係は、関係従属関係ではない"自明でない多値従属関係（non-trivial mulivalue dependency）"を持てないことを意味している。

図２－５１は関係ＥＭＰの全ての属性の集合がキーであるので、３ＮＦではあるが、関数従属性が存在しないためにこれ以上分割出来ない。関係ＥＭＰから、４ＮＦの導出を試みる。関係ＥＭＰにおける従属性は、ＥＭＰＬＯＹＥＥ→→ＣＨＩＬＤである。先に述べた関係の分割規則によって、ＥＭＰは、

ＥＭＰ１（ＥＭＰＬＯＹＥＥ，ＣＨＩＬＤ）

ＥＭＰ２（ＥＭＰＬＯＹＥＥ，ＳＡＬＡＲＹ，ＹＥＡＲ）の２つへ分割出来る〔図２－５２〕。この２つの関係は、ともに定義より４ＮＦである。

ＥＭＰ１関係

EMPLOYEE	CHILD
Hilbert	Hubert
Gausa	Gwendolyn
Gausa	Greta
Pythagoras	Peter

ＥＭＰ２関係

EMPLOYEE	SALARY	YEAR
Hilbert	S35K	1976
Hilbert	S40K	1976
Gausa	S40K	1975
Gausa	S50K	1976
Pythagoras	S15K	1975
Pythagoras	S20K	1976

図２－５２　４ＮＦである関係

⑤ 正規化のまとめ及び問題点

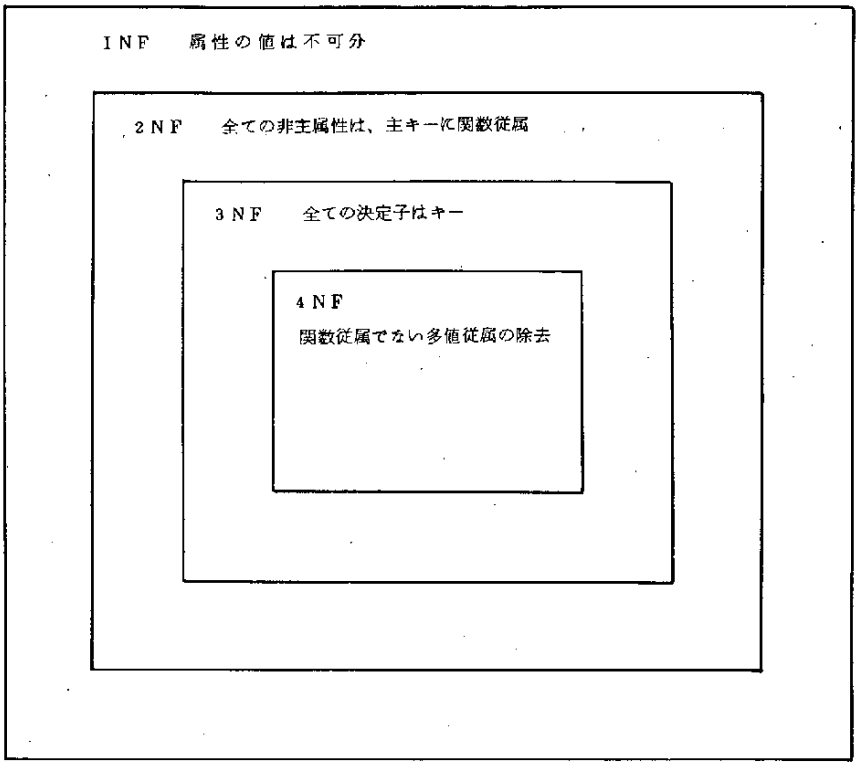


図２－５３　各正規形間の関係

図２－５３に、今まで述べてきた４つの正規形の関係をまとめる。２ＮＦは１ＮＦでもあり、３ＮＦは２ＮＦでもあり、４ＮＦは３ＮＦでもあり、４ＮＦが最も望ましい正規形である。３ＮＦまでは、属性間の関数従属関係へ基づいていたが、４ＮＦでは属性間の多値従属概念が導入されている。多値従属概念の導入は、関数従属では扱い得なかった属性関係の扱いを可能としている。関係モデルにおけるデータの意味は、属性間の従属性（関係従属及び多値従属）によって表わす。代数学的概念を関係モデルの基礎とすることによって、属性間従属に基づく正規化に対する構文的研究を進めさせてきた。この数学的基礎は、一方では、関係モデルにおける意味の取扱いに対する重大な問題を生じさせている〔SCHM ７５，BRAC ７６，CHEN ７６〕。次の例を考えてみよう。ある社員は１人の上司を持ち、上司はある給料を持つ。ある企業は１つの仕事を持ち、仕事は１つの仕事名を持つ。この事実は、ＥＭＰ→ＭＧＲ，ＭＧＲ→ＳＡＬ，そしてＥＭＰ→ＪＯＢ，ＪＯＢ→ＪＯＢ＃として表わせる。関数従属の推移性より、前者はＥＭＰ→ＳＡＬ，後者はＥＭＰ→ＪＯＢ＃となる。この両者は全く異なった事実を表わしている。即ち、前者のＳＡＬは上司の給料であり後者のＪＯＢ＃は社員の仕事名である。この様な意味を、数学的従属では表現出来ない。この関係モデルの意味論的問題は、後述するようにデータベースの意味を記述することを目指した種々の概念モデルを考え出さしている。

C. 関係モデルデータベースの論理設計

関係モデルのデータベースにおける問題は、関係スキーマ（即ち、テーブルの編成）の適当な集合を選択することである。論理設計のアプローチとしては、３ＮＦ分割アプローチ〔ＣＯＤＤ７１ａ〕、統合的（synthetic）アプローチ〔ＢＥＲＮ７７，ＢＥＲＮ７５〕、４ＮＦ分割アプローチ〔ＦＡＧＩ７７ｃ〕の３つがある。

Coddの３ＮＦ分割アプローチは、関数従属情報と関係（３ＮＦとは限らない）をもとにして、３ＮＦ関係への分割を行なうものである〔図２－５４〕。このアプローチではデータベースの意味は属性間の関数従属によってのみ表わせられ、関数従属は、関係内で閉じている。問題点としては、最終的に生成される３ＮＦ関係は、最初に入力される関係によって限定されてしまうことである。即ち、データベース設計者は、最終的３ＮＦとあまり異なっていない関係を入力せねばならない。

統合的アプローチは、属性の集合と、属性間の関数従属とを入力して、これらを統合して３ＮＦ関係を生成するものである〔図２－５５〕。３ＮＦ分割アプローチでは、関数従属は１つの関係内で閉じたものとして定義されていたが、ここでは関数従属は関係とは独立に存在する。このため、正規化の最後に述べたような問題（ＥＭＰ→ＭＧＲ，ＭＧＲ→ＳＡＬＡＲＹの時、ＥＭＰ→ＳＡＬＡＲＹは、社員のではなく上司の給料）が発生する。この問題を解決するために、関数従属の一意性を保障する（例えば、ＥＭＰ→ＳＡＬＡＲＹ＿ＯＦ＿ＭＧＲ）ことが必要になる。この一意性の仮定を確かめることは、複雑で困難である。又、４ＮＦの項で述べたように、従属関係として関数従属だけでは不十分である。

Faginによる４ＮＦ分割アプローチは、統合的アプローチに影響を受けながら、多値従属性を導入している。このアプローチは、統合的アプローチのように属性と従属性（関数従属及び多値従属）とを入力とし、４ＮＦ関係を生成する〔図２－５６〕。統合的アプローチにおいて指摘された従属の一意性の問題は、３ＮＦアプローチにおけるように一意性は１つの関係内においてのみ保たれるとすることによって解決している。即ち、ＡとＢが関係Ｒの属性ならば、関数従属Ａ→Ｂは関係Ｒ内においてのみ意味を持つことである。４ＮＦ分割アプローチでは、最初全属性から成る単一の関係を設け、このなかで従属の一意性を保つ。単一関係から４ＮＦ分割を始めることは、３ＮＦアプローチが最終的な３ＮＦ関係へ近い幾つかの関係を入力せねばならない点へ対して優れている。このことは、最終結果として４ＮＦ関係の幾つかの集合が考えられるが、どの集合が最適かの決定の必要性を生じさせる。Ｆａｇｉｎ は、４ＮＦ分割過程へ人間が介入し種々の決定（最適化etc.）を行なう余地があることを指摘している〔ＦＡＧＩ７７ｃ〕。統合的アプローチへ対しては、従属の一意性問題の自動的解決と、従属関係として多値従属を持つことが優れている。

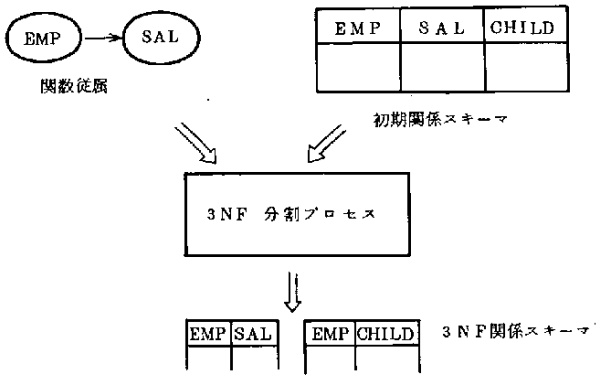


図2－54　3NF分割アプローチ

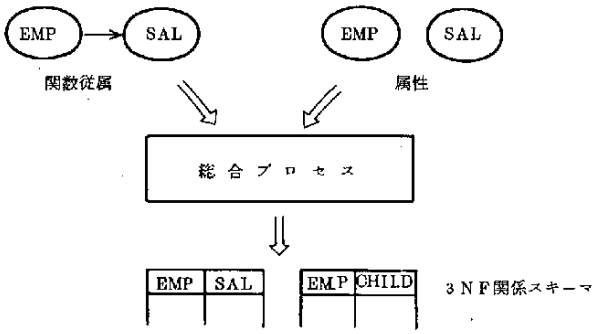


図2－55　統合的アプローチ

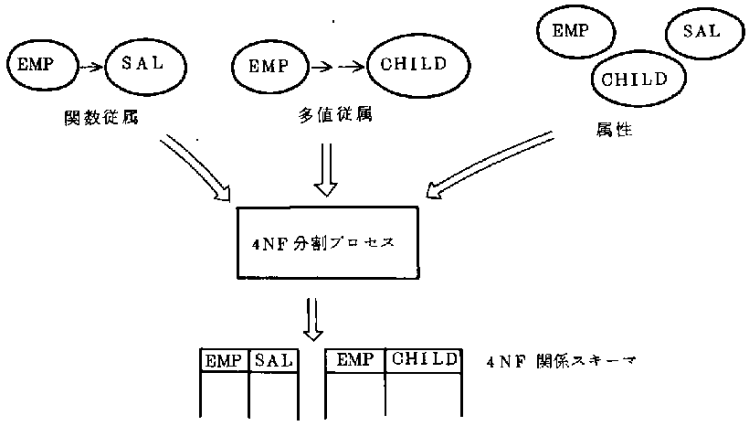


図2－56　4NF分割アプローチ

D. 関係言語

関係モデルデータベースの特徴の１つは、非手続的で高水準なデータ・アクセス言語を備えていることである。これらの言語を関係言語（relational language）と呼ぶことにする。関係モデルの数学的基礎に起因して、関係言語は高度の非手続性を持っている。

先ず、関係言語として、データ準言語（data sublanguage）と問合せ言語（guery language）の２種類を考える〔CHAM 76a〕。データ準言語は、親言語に組み込まれたものであり、データベースへの検索と格納の機能を持っている。問合せ言語は、ユーザがＤＢＭＳと直接会話するための独立した言語である。問合せ言語は、データ準言語に対して高水準非手続的である。

関係言語において必要となる機能としては、問合せ、データ操作、データ定義、データ制御の４点がある〔CHAM 76a〕。関係言語は、図２－５７のように分類出来る。

関係言語
- 関係計算言語…… ALPHA〔CODD71b〕,QUEL〔HELD75〕
- 関係代数言語…… 〔CODD71a〕
- 写像型言語…… SQUARE〔BOYC75〕,SEQUEL〔CHAM74,76a〕
- グラフィック言語…… Query By Exanple〔ZLOO77〕

図２－５７　関係言語

① 関係計算言語

この言語は、第１階述語計算理論にもとづいている。

Coddは、〔CODD 71b〕の中で、第１階述語理論に基づいた準言語ＡＬＰＨＡを詳細に示している。ＡＬＰＨＡの問合せは、目標（target）と条件部（gualification）とから成っている。目標は答えとして必要な特定な関係の特定の属性を指定する。条件部は目標の関係から特定の組を選び出すための条件を表わしている。条件部は複雑な形にも出来るし、全数作用素（∀）や存在作用素（∃）を使うことも出来る〔図２－５８ｂ〕。ＡＬＰＨＡの例として図２－５８ａを考えよう。ＲＡＮＧＥ文は、Ｐが関係ＰＲＥＳＩＤＥＮＴの組変数であることを示している。②は目標部、③は条件部である。ＮＡＭＥがＫＥＮＮＥＤＹである組ＰのＨＯＭＥ－ＳＴＡＴＥの値を作業領域Ｗに取り出すことを示している。

a) What was the home state of President Kennedy ?

```
RANGE  PRESIDENTS  P
GET  W  P. HOME-STATE : ( P. NAME=KENNEDY ).
        ①          ②              ③
```

b) List the election years in which a Republican from Illinois was elected.

```
RANGE  PRESIDENTS  P
RANGE  ELECTIONS_WON  E
GET  W  E. YEAR : ∃P ( P. NAME=E. WINNER-NAME  &
     P.PARTY='REPUBLICAN' & P.HOME-STATE='ILLINOIS' ).
```

図２－５８　ＡＬＰＨＡの例

関係計算に基づいた他の言語としてＱＵＥＬ〔ＨＥＬＤ　７５〕がある。ＱＵＥＬとＡＬＰＨＡへ類似した言語であるが、これと比較して次の様な特徴を持っている。

1) ＡＬＰＨＡは、算術演算に関して親言語に依存している。

2) ＱＵＥＬは、限定作用素を全く使用しない。

3) ＳＵＭ，ＭＡＸ等の集合演算（aggregate operation）を一般的に使える。

ＱＵＥＬ問合せの一般形式を図２－５９に示す。

問　合　せ　：：＝｛Range 文｝｛検索文｜更新文｝
Range　文　：：＝RANGE　OF　｛変数｝　IS｛関係｝
検　索　文　：：＝RETRIEVE　INTO　結果名〔目標リスト〕
　　WHERE　条件
更　新　文　：：＝コマンド　結果名〔目標リスト〕
　　WHERE　条件
コマンド　：：＝REPLACE｜DELETE｜APPEND　TO
結　果　名　：：＝検索文又は更新文によって新たに生成された関係名
目標リスト　：：＝｛結果領域＝関数｝
結果領域　：：＝生成された関係（結果名）内の領域名
関　　数　：：＝1.　属性についての算術演算
　　2.　集合値関数（set valued function）
　　3.　集合値関数を組みあわせた組合せ関数（aggregate function）
　　（例　COUNT，AVR，SUM，MAX，MIN）
変　　数　：：＝組変数（tuple variable）

図２－５９　ＱＵＥＬ問合せの一般形式

ＱＵＥＬの例用例を次に示す。検索を図２－６０ａ、更新を図２－６０ｂに示す。例として関係Ｃ（ＣＮＡＭＥ，ＳＴＡＴＥ，ＰＯＰＵＬＡＴＩＯＮ，ＡＲＥＡ）を用いる。

Example: CITY(CNAME,STATE,POPULATION,AREA)
"Find the population density of all cities in California with population greater than 50k"

```
RANGE OF C IS CITY
RETRIEVE INTO W(C.CNAME,
            DENSITY=C.POPULATION/C.AREA)
            WHERE C.STATE='California'
            AND C.POPULATION>50K
```

(note the default used for CNAME=C.CNAME and that the result of the query is a relation W(CNAME,DENSITY)).

図２－６０ａ　検索例

```
EMP(NAME,SAL,BDATE,MGR)
NEWEMP(NAME,AGE,SALARY)
RANGE OF E IS EMP
RANGE OF N IS NEWEMP
```

Example:

All new employees over 35 are to work for SMITH

```
APPEND TO EMP(N.NAME,SAL=N.SALARY,
  BDATE=1975-N.AGE,MGR='SMITH') WHERE
  N.AGE>35
```

Example:

Give all employees a ten percent raise who work for Jones

```
REPLACE E(SAL=1.1*E.SALARY) WHERE
  MGR='JONES'
```

Note: The keywords BY and IS may be used interchangeably with '=' anywhere in INGRES to improve readability.

Example:

Remove all employees who are in the EMP relation from the NEWEMP relation.

```
DELETE N WHERE N.NAME=E.NAME
```

DECOMPOSITION

図２－６０ｂ　更新例

ここでＡＰＰＥＮＤコマンドの"結果名"は、既存の関係名でなければならない。ＲＥＰＬＡＣＥとＤＥＬＥＴＥコマンドの"結果名"は組変数でなければならない。

ＱＵＥＬにおいて関数と条件のクラスが問題になる。関数グラフＣ０と条件クラスＱ０とを次の様に定める。

C０： 1) 定数はＣ０内にある。

2) 属性はＣ０内にある。

3) ｆとｇがＣ０内にある場合、ｆ＋ｇ，ｆ－ｇ，ｆ＊ｇ，ｆ／ｇ，log(f)ｇもC0にある。

Ｃ０内の関数は属性関数と呼ばれる。

Ｑ０： 1) Ｑ０内の基本論理式はｆ(comp)ｇである。compは、＜，≦，＝，≠の組合せであり、ｆとｇはＱ０内になければならない。

2) 基本論理式のブール結合（ＮＯＴ，ＡＮＤ，ＯＲ）もＱ０内にある。

Ｃ０とＱ０によって定まるＱＵＥＬをＱＵＥＬ０とする。ＱＵＥＬ０は問合せの基本機能を備えている。Ｃ０とＱ０とへ集合関数（ＭＡＸ，ＣＯＵＮＴ，etc）と集合値関数（set-valued funetion）とを導入することによって、より複雑なＱＵＥＬｉ（ｉ＞０）が定義されていく。図２－６０ａはＱＵＥＬ０問合せである。ＱＵＥＬ２の例を図２－６１に示す。ＣＯＵＮＴ，ＡＶＧは集合関数である。

```
SUPPLY(S#,P#,PRICE)
Query : Find those suppliers whose price for every part
    that he supplies is greater than the average price
    for that part.
RANGE OF S IS SUPPLY
RETRIEVE INTO W(S.S#)
WHERE COUNT(S.P# BY S.S# WHERE S.PRICE
                    >
                AVG'(S.PRICE BY S.P#)
=COUNT(S.P# BY S.S#)
```

図２－６１　ＱＵＥＬの例

ＱＵＥＬｎ（ｎ＞０）は高水準問合せである。この様な高水準問合せ問題として次の２点を上げることが出来る〔WONG 76〕。

1) 入れ子構造を持った集合演算（ＭＡＸ，ＣＯＵＮＴ）の存在

2) 多変数問合せの存在

集合演算を持った多変数ＱＵＥＬｎを、集合関数を持たず属性関数だけを持った一変数ＱＵＥＬ０へ分割することが必要になる。この分割プロセスは、次の２つの段階から成る。

1) ＱＵＥＬｎ→ＱＵＥＬ０

2) 多変数ＱＵＥＬ０→一変数ＱＵＥＬ０

2)の分割アルゴリズムは、Wong〔ＷＯＮＧ７６〕に論じられている。このアルゴリズムを用いて分散型データベースにおける問題、複数サイトへまたがった問合せを各サイトの問合せへの分割も同じくWong〔ＷＯＮＧ７７〕に例が示されてある〔詳細は、3.3.2項のＳＤＤ-1を参照のこと〕。

ＱＵＥＬは、分散型データベースシステムＩＮＧＲＥＳ〔ＳＴＯＮ７７〕、ＳＤＤ－１〔ＲＯＴＨ７７ｂ〕における問合せ言語として使用されている。

② 関係代数言語

この言語は、関係代数（relational algebra）に基づくものである。Coddは、〔ＣＯＤＤ７０，７１ａ〕の中でこの言語について論じている。関係代数は、関係を操作して結果として新しい関係を作り出す様な演算子の集合である。演算子としては、集合論的演算子（直積、積、和、差）、射影、制約、結合、除算が考えられる。

1) 射影（projection）

射影演算子は、与えられた関係の中から必要な列だけを取り出し、重複する組は除外する。例えば関係ＰＲＥＳＩＤＥＮＴＳから（ＰＡＲＴＹ，ＨＯＭＥ－ＳＴＡＴＥ）の一意な組み合せを全て見つけ出すには、ＰＲＥＳＩＤＥＮＴＳ〔ＰＡＲＴＹ，ＨＯＭＥ－ＳＴＡＴＥ〕という射影を行なえばよい。

2) 制約（restriction）

これは、ある関係から与えられた条件を満足する組だけを取り出す。選挙年が１９６４年以降の関係ＥＬＥＣＴＩＯＮＳ－ＷＯＭの組を取り出すには、ＥＬＥＣＴＩＯＮＳ－ＷＯＮ〔ＹＥＡＲ＞１９６４〕という制約演算を行なえばよい。

3) 結合（join）

２つの関係ＡとＢを考える。結合演算子は関係Ａの組と関係Ｂの組とを突き合わせて、与えられた条件が満足されるなら、２つの組を連結し結果として連結された関係をつくる。関係ＥＬＥＣＴＩＯＮＳ－ＷＯＮと関係ＰＲＥＳＩＤＥＮＴＳとを、ＷＩＮＮＥＲ－ＮＡＭＥ＝ＮＡＭＥである条件下での結合例を次に示す。

ＥＬＥＣＴＩＯＮＳ－ＷＯＮ〔ＷＩＮＮＥＲ－ＮＡＭＥ＝ＮＡＭＥ〕ＰＲＥＳＩＤＥＮＴＳ

4) 除算（division）

除算も２つの関係を入力として第３の関係をつくり出す。除算は被除算子は２項関係であ

り、除算子は１項関係であり、結果も１項関係である。この演算は除算子の組（１属性からなる）と、被除算子の関係の組と比較し、除算子と等しい値を持つ属性値をもった組の他の属性のみを集めて結果とするものである。例えば、

ELECTIONS−WON〔YEAR,WINNER−NAME〕／PRES〔WINNER−NAME〕

ここで　PRES

WINNER−NAME
NIXON

の時、結果は

YEAR
1968
1972

となる。

除算演算子は、他の演算子を組み合せて表現することも可能である。

5）集合論的演算子（set-theoretic operators）

通常の集合論的演算子、和、積、差等である。

6）入れ子（nesting）

関係代数では、上記の演算子を入れ子にして、任意の複雑な式を作り得る。

③ 写像型言語（mapping-oriented language）

この言語は、初心者を対象としたもので、全数作用素等の数学的概念を使用せずに、関係代数と関係計算と同等の機能を示している。これの代表的言語として、ＡＰＬ式の簡潔な記法によるＳＱＵＡＲＥ〔ＢＯＹＣ７５〕、英語のキーワードを用いた構造型言語〔ＣＨＡＭ７４，７６ｂ，ＡＳＴＲ７６〕がある。写像型言語の基本概念は写像である。即ち、ある既知の属性（又は属性群）を、何等かの関係によって希望する属性（又は属性群）への写像である。一般にある写像結果を別の写像の指定内に使用してもよい。このため写像の入れ子を用いて複雑な問合せが出来る。

ＳＥＱＵＥＬにおける例を次に示す。"Kennedyはどの州の出身か"という問合せは、

```
SELECT HOME-STATE
FROM   PRESIDENTS
WHERE  NAME＝'KENNEDY'
```

となる。又、入れ子を用いた例として、"Illinois州出身の共和党候補が当選した選挙年を全て挙げよ"という問合せは、

```
SELECT  YEAR
FROM    ELECTIONS-WON
WHERE   WINNER-NAME＝
        SELECT  NAME
        FROM    PRESIDENTS
        WHERE   PARTY＝'REPUBLICAN'
        AND     HOME-STATE＝'ILLINOIS'
```

となる。

次にＳＥＱＵＥＬにおける関係の更新例を示す。まず新たに関係ＥＭＰ（ＥＭＰＮＯ，NA-

ＭＥ，ＪＯＢ，ＳＡＬＡＲＹ）関係を考える。全てのプログラマの給料を１０％増加させるのは、

```
UPDATE  EMP
SET     SALARY＝SALARY＊1.1
WHERE   JOB＝'PROGRAMMER'
```

として行える。

ＳＥＱＵＥＬのデータ制御をみてみる。ＳＥＱＵＥＬにおいては、保全性の確認命題（assertion）を導入している。確認命題とは、データベースの内容に関する一つの文であり、システムはデータベースが常にこの確認命題を満足するようにチェックされる。満足しない更新は拒否される。例えば、全ての選挙において、当選者は落選者よりも多くの得票数をもたねばならない。このことを次のような確認命題として記述できる。

```
ASSERT ON ELECTIONS-WON X:
  WINNER-VOTES >
    (SELECT   MAX (LOSER-VOTES)
    FROM      ELECTIONS-LOST
    WHERE     YEAR=X.YEAR)
```

④ グラフィックス型言語

最近登場した有力な言語である。この言語では、利用者は問合せを左から右へ1行ずつ書いていくのではなく、グラフィック・ディスプレイ端末上で選択したり空白を埋めたりしながら問合せを構成していく。この言語としては、Query By Example〔ＺＬＯＯ７５，７７〕がある。

Query By Example 言語では、ユーザへディスプレイ上に空白な関係が表示される。ユーザは必要な答の例を、この関係の一つ又はそれ以上の列に記入する。既知の値を直接記入する。値が解らないものは差し当たり何でもよいから記入し、そこに下線を引いて例であることを示す。印字したい属性は、＂P.＂で指示する。例を図２－６２に示す。

Q1) What was the home state of President Kennedy?

PRESIDENTS

NAME	PARTY	HOME-STATE
KENNEDY		P. NEVADA

Q2) List the election years in which a Republican from Illinois was elected.

ELECTIONS-WON

YEAR	WINNER-NAME	WINNER-VOTES
P.1948	WILSON	

PRESIDENTS

NAME	PARTY	HOME-STATE
WILSON	REPUBLICAN	ILLINOIS

図２－６２　Query By Example の例

E. システム例

関係モデルを用いたＤＢＭＳ実現の最大の課題は、十分なパフォーマンスをいかに実現するかである。データモデルと言語とがより高水準なものになるにつれて、ＤＢＭＳの内部の多層化が進み、システムのオーバヘッドが増加する。十分なパフォーマンスの達成は、関係ＤＢＭＳの現在の主要な課題である。又、関係モデルをデータベースの管理システムとして用いるには、複数ユーザの同時実行制御、機密保護、保全性制御もちろんのこと導出視点〔ＣＨＡＭ７５〕の動的定義、自然言語に近い高水準非手続的関係言語等を十分な機能を備えることが必要である〔ＣＯＤＤ７４〕。

既に多くのシステムが開発されてきている〔表２－５〕。このなかで代表的な幾つかを検討してみる。

① ＰＲＴＶ（Peterlee Relational Test Vehicle）〔ＴＯＤＤ７６〕

英国Peterlee のＩＢＭサイエンティフィックセンタで開発されている関係代数を実働化させようとするシステムである。ＰＲＴＶでは通常の結合、射影といった関係代数演算子へ加えて、新しい関係演算子を追加してシステムを容易に拡張出来るようにしている。ＰＲＴＶは、1,０００万字規模の環境問題データベースとして、又、大ロンドン市議会で5,０００万字規模の都市問題データベースとして利用されている。

② System Ｒ〔ＡＳＴＲ７６〕

ＩＢＭのSan Jose 研究所において開発されている大規模な原型ＤＢＭＳである。関係モデルを、多数の同時実行と大量の要求との環境下へ適用した最初の例である。SystemRはデータ利用資格認定、システム使用登録、回復、複数の視点定義、データの一貫性、保全性の自動確保等のＤＢＭＳとしての十分な機能を備えようとしている。ユーザ言語はＳＥＱＵＥＬを使用している。

③ ＩＮＧＲＥＳ

ＵＣＢにおいて開発されているシステム〔ＨＥＬＤ７５〕で、現在は分散型データベースとなっている〔ＳＴＯＮ７７〕。これはＰＤＰ－１１のＵＮＩＸオペレーティングシステムのもとで稼動するものである。ユーザの問合せはＱＵＥＬ言語によってなされる。ＩＮＧＲＥＳはユーザの出すＱＵＥＬ命令を自動的に修飾する方法によって多様な機能を実現している。ＩＮＧＲＥＳの物理データ構造は、ハッシュ表、汎用ディレクトリ等がある〔ＨＥＬＤ７８〕。現在ＩＮＧＲＥＳはＵＣＢ外部４０カ所に設置され、ニューヨーク電話会社では１５０Ｍバイトのケーブル保守システム用のデータベースに使用している〔ＳＡＫＡ７７〕。

④ 関係データベースマシン

関係言語は、内部で原始オペレータへ分解される。この原始オペレータをハードウェアで実現しようとする試みである。これはデータベースを連想メモリに記憶させ、全てのデータ項目をアドレス指定によらず、値そのものを指定して検索させるものである。この例としてトロント大学のＲＡＰ，ユタ大学のＲＡＲＥＳ，フロリダ大学のＣＡＳＳＭがある。

⑤ この他、商用ＤＢＭＳとしてMDBM（ＨＩＳ），ＭＡＧＮＵＭ（ＴＹＭＳＨＡＲＥ），ＮＯＭＡＤ（National CSS)等が発表されている〔ＳＡＫＡ７８〕〔表２－５〕。

表２－５　関係ＤＢＭＳの例〔ＳＡＫＡ７８〕

システムまたは言語名	開発機関	特長
Mac AIMS	MIT	・MULTICSのもとで稼動
RDMS	ジェネラルモーターズ	・ディスプレイ向き検索システム
PRTV	IBM U. K.	・都市計画システムに適用
XRM	IBM ケンブリッジ	・アソシエイティブ・アクセス機構 ・SEQUEL の基本アクセス法に適用
SEQUEL	IBM サンノゼ研究所	・写像型言語
GMIS	MIT	・SEQUEL システムから拡張 ・エネルギー資源システムに適用
QUERY-BY-EXAMPLE	IBM ヨークタウン・ハイツ研究所	・グラフィック型言語
SYSTEM-R	IBM サンノゼ研究所	・複数ユーザによる共用のためのデータ管理機能が豊富
INGRES	カリフォルニア大学バークレイ	・複数ユーザによる共用のためのデータ管理機能が豊富
ZETA	トロント大学	・数種の言語をもつ
OMEGA	トロント大学	・ANSI/SPARC のデータ・ベース・システム・アーキテクチャに準拠
CASSM	フロリダ大学	・アソシエイティブ・プロセサ
RAP	トロント大学	・アソシエイティブ・プロセサ
RARES	ユタ大学	・アソシエイティブ・プロセサ
MDBM	HIS	・商用 DBMS
GAMMA-0	TYMSH-ARE	・商用 DBMS
NOMAD	National CSS	・商用 DBMS

F. 関係モデルＤＢＭＳの問題点

関係モデルの議論は、データモデルの問題とデータベース実現の問題との２つに分けて考えられる。データモデルに関する関係モデルの成果は、数学的概念を導入することによって関係の完全な取扱いとデータベースの厳密な設計を可能としたことである。しかし正規化の項で議論したように、データベースの意味表現の困難さを示している。

関係モデルは、数学的基礎を持った関係、これを構成する属性演算（結合、射影等）とを基礎としている。この点が先に述べた様に関係モデルの重要な利点の１つである。このことは、実世界が関係の集合によって表わされている場合に、関係モデルによって世界をよく理解できることを意味している。しかし、関係モデルは実世界を関係の集合として表現する方法を提供していない。即ち、属性の集合としての関係の実世界での意味は何なのだろうか。この問題は、関係モデ

ルの数学的概念からは答えることは出来ない。関係モデルは、関係に対する述語論理又は関係代数に基づいた数学的演算を形式的に行なわせる。関係へのこの様な演算結果はしばしば無意味な結果を生成し、無意味であることを警告することも出来ない〔ＣＨＥＮ７６〕。２つの関係ＥＭＰ（ＥＭＰ＃，ＥＮＡＭＥ，ＳＡＬＡＲＹ）とＭＧＲ（ＭＧＲ＃，ＭＮＡＭＥ，ＳＡＬＡＲＹ）を考えてみる。この２つの関係のＳＡＬＡＲＹについての結合は数学的に可能である。しかし社員の給料と管理者の給料との結合をとった関係の意味は何なのだろう。生成された関係は全く無意味なものである。関係における値は、それの属する領域を持っている。関係のいくつかの行の値が同一領域に属することによる混乱を防ぐために属性を用いた。属性名は同一領域に属する行を識別するためのものであり、関係における意味を表わすものではない。関係ＥＭＰにおけるＳＡＬＡＲＹと関係ＭＧＲにおけるＳＡＬＡＲＹは、同一属性名を持つが、その意味は異なっている。

同様のことは、正規化の過程においても生ずる。正規化は、しばしば実世界において意味のない関係を生成してしまう〔ＳＣＨＭ７５，ＭＯＵＬ７６〕。次の関係ＥＭＰＬＯＹＥＥ（ＥＭＰ＃，ＮＡＭＥ，ＤＥＰＴ＃，ＤＥＰＴＭＧＲ）を考えてみる。この関係における関数従属性は、ＥＭＰ＃→ＮＡＭＥ，ＤＥＰＴ＃，ＤＥＰＴＭＧＲとＤＥＰ＃→ＤＥＰＴＭＧＲである。よってこの関係は推移従属を含むために第２正規形である。第３正規化によって

ＥＭＰ１　（ＥＭＰ＃，ＮＡＭＥ，ＤＥＰＴ＃）

ＤＥＰＴ２（ＤＥＰＴ＃，ＤＥＰＴＭＧＲ）　　が生成される。しかし、社員の持つ情報に興味のある企業体にとって、関係ＤＥＰＴは意味のないものである。この様に第３正規化は意味のない関係を生成してしまう。〔ＣＨＡＭ７５〕による導出視点の問題も関係の意味に関している。

他の問題は、属性間の基本的関係性である関数／多値従属は実世界の概念を表現するのに適しているかである。Fagin〔ＦＡＧＩ７７ａ，ｂ〕とZaniolo〔ＺＡＮＩ７６〕とによる多値従属関係の導入により一般的な従属関係を扱かえるようになった。１人の社員が１人の上司をもつという関係は関数従属によって表わされるが、１人の社員が何人かの子供を持つという関係は関数従属では表わせず、多値従属によってのみ表わせる。従属関係を一般的に扱かえるようになったが、従属の推移性による意味の問題は解決されない〔ＳＣＨＭ７５〕。正規化において指摘した様に、ＥＭＰ＃→ＭＧＲ＃，ＭＧＲ＃→ＳＡＬＡＲＹから導かれる関数従属ＥＭＰ＃→ＳＡＬＡＲＹを考えてみる。ＥＭＰ＃→ＳＡＬＡＲＹにおけるＳＡＬＡＲＹの意味は社員の給料ではなく、社員の上司の給料である。従属関係の中にこの意味を表わせるだろうか。

関係モデルは、個々の実世界の対象をその属性とは独立には扱えない〔ＨＡＬＬ７６，ＭＯＵＬ７６，ＣＨＥＮ７６〕。関係モデルは、対象の属性間の関係によって、その対象を表わしている。関係は、それが結合している全ての対象が存在する場合にのみ存在するが、対象は関係の存在とは独立に存在している。この点は関係モデルへの主要な批判点となっている。

関係モデルを提供するデータベース管理システム実現上の問題の第１は有効なパフォーマンス

を達成することである。高度なデータ独立性と非手続的言語とは、データの物理的な記憶や検索方法とは独立であることを意味している。関係モデルのようなユーザ向けの高度なインタフェースの提供は、ＤＢＭＳの多層化をもたらし、これによるオーバヘッドが問題となってくる。

第２の問題は、ＤＢＭＳとしての十分な機能提供である。ＤＢＭＳとして必要な機能として、関係言語、多重ユーザの同時実行制御、機密保護と保全性確保、導出視点のユーザ定義等を上げることが出来る。

Coddは、〔ＣＯＤＤ７４〕の中で、次の４点を今後の課題として上げている。

1) 同時実行制御技術の開発
2) 有効なパフォーマンス
3) 多重導出視点のサポート
4) 自然言語による問合せ

2.3 ベンダが提供する代表的ＤＢＭＳ

本節では、ベンダが提供している数多いＤＢＭＳのなかから、ＡＤＡＢＡＳ，ＳＹＳＴＥＭ２０００，ＴＯＴＡＬの三システムについてその概要を紹介する。

ＡＤＡＢＡＳは、西独で開発されたことも興味を引く点であるが、データ蓄積の効率化や機密保護機能としての暗号化機能、検索機能としての音声化機能など他のＤＢＭＳにはない幾つかの特長ある機能をもっている。

ＳＹＳＴＥＭ２０００は、インバーテッドキーを有効に利用したＤＢＭＳの成功例の一つである。このＤＢＭＳは、プログラムの実行中にレコード間の関係を組み立てたり、解消したりする機能を持っており興味深い。

ＴＯＴＡＬは、単独のＤＢＭＳとしては、世界で最も多くのユーザを持っている。また、その裏付けとしてＴＯＴＡＬが稼働可能な機種も他のＤＢＭＳに比べて群を抜いている。機能的な特徴としては、データベースにマスタデータセットとバリアブルデータセットの二つを持ち、可変長データの取り扱いが可能なことと、項目だけを意識してアクセスすれば良いことも特徴と言えよう。

2.3.1 ADABAS

A. 概要

ＡＤＡＢＡＳ（Adaptable Data Base System）〔ＳＯＦＴ７７ａ～ｇ〕は西独のSoftware AG社が１９７１年に発表したＤＢＭＳである。開発はKreis, P.，とSchnell, P.の両名を中心にわずか５人で行なわれた。ＡＤＡＢＡＳは、Siemens４００４シリーズをはじめＩＢＭ３６０／３７０シリーズ、ＵＮＩＶＡＣ９０００シリーズで実働可能なシステムで、全世界約２００社以上で使用されている〔ＩＳＨＩ７７ａ〕。

主要な特徴としては、データベース構造とその諸関係を多種多様な形式に指定することができ、データ部を生データの集合として独立させて、関係情報のみで複雑なインバーテッド・リストを構成することである。データ定義とファイルの生成はユーティリティを使って容易に行なうことができる。またデータベースへのアクセスは、各種用意されているデータ操作言語を利用することができる。

B. ADABASのDDL

ADABASデータベースは、アソシエータデータセット，データストレジデータセット，ワークデータセットの三つのデータセットから構成される。

アソシエータには、データベースの管理・制御情報とデータ・ストレジの各レコードに対するインバーテッド・リストが作成される。このうち、アソシエーション・ネットワークと呼ばれるインバーテッド・バリュー・リストに、ファイル間、項目間の諸関係情報が作成される。

データ・ストレジには、各種業務用ファイルがシーケンシャルな編成で、最大２５５個まで蓄積することができ、しかもデータ圧縮機能によって、データ中の無意味な０や空白、値のない項目は格納されず、スペースの効率化が行なわれる。

ワークデータセットは、アソシエータ内のインデックスに対し集合演算などを行なったときの作業用領域としてや、回復情報の貯え場所として用いられる。データベースは、すべてADABASのユーティリティを使って作成・更新を行なう。ファイル生成は、LOADERによってデータ定義およびデータの蓄積を行ない、ファイル間、項目間の関係をCOUPLEによって行なう。

図２－６３に、ADABASデータベース作成過程の概要を示す。

LOADERは３つのプログラム（データ定義用としてADAWAN，アソシエータの再整理用としてADALD１，ADALD２）で構成され、ファイル（最大２５５個）ごとにデータ定義カードを与えてデータベースを創成する。

以下に例をあげて、データ定義およびカップリング機能を中心に説明する。

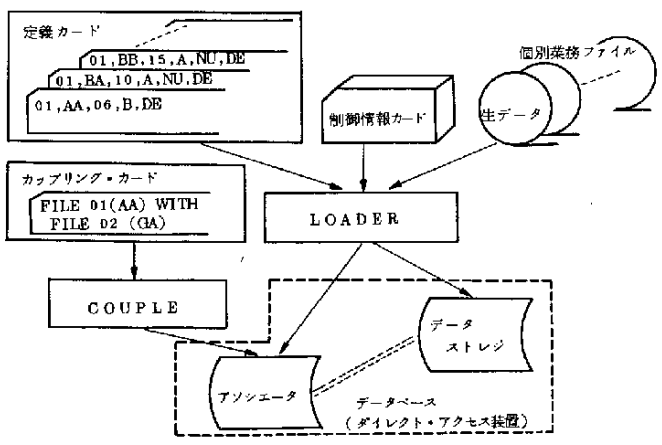


図２－６３　ADABASデータベース作成過程

① データ定義カード

各ファイルに対して、最大５００個までの個別項目（キー）を定義することができる。

・項目定義エントリの構成

level-number（レベル番号）：項目のグルーピングを行なって階層関係を設定する。
０１～０７のレベル番号を与える。

field-id （項目ＩＤ）：項目識別名を２文字で与える。第１文字は英字、第２文字は英数字のみ使用。
（low-level DMLでは、このfield-idを使って行なう。）

length （長　さ）：項目長をバイト数で与える。
formatによって最大長は異なる。

A：２５３バイト
B：１２６バイト
F：　４バイト
P：　１４バイト
U：　２７バイト

format （形　式）：項目形式を識別する。

A：英数字
B：２進数
F：固定小数点（４バイト／２進数）
P：パック型１０進数
U：アンパック型１０進数

options （項目特性）：項目特性

DE：descriptor status
FI：fixed storage
NU：null suppression
MU：multiple value field　（1～191）
PE：periodric group　（1～99）

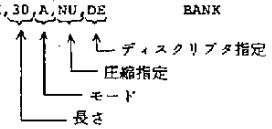
field-name（項目名）：独自な（Uniquely）項目名を与える。
（ＡＤＡＳＣＲＩＰＴでは、この名前を使って行なう）

ファイル1　Persannel-file （人事ファイル）

```
 1  01,AA,06,B,DE          PERSONNEL-NUMBER
 2  01,BA,10,A,NU,DE       LAST-NAME, NAME
 3  01,BB,15,A,NU,DE       FIRST-NAME
 4  01,BC,01,A,FI          INITIAL
 5  01,CA,01,A,NU,DE       SEX
 6  01,CB,02,U,NU,DE       AGE
 7  01,CC,10,A,NU,DE       FAMILY-STATUS
 8  01,CD,02,U,NU,DE       DSPENDENTS
 9  01,DA,05,U,NU          NUMBER
10  01,DB,20,A,NU,DE       STREET
11  01,DC,15,A,NU,DE       CITY
12  01,DD,02,A,NU,DE       STATE
13  01,DE,05,U,NU,DE       ZIP
14  01,DF,08,A,NU,DE       PHONE
15  01,FA,20,A,NU,DE       JOB
16  01,FB,06,U,NU,DE       SALARY
17  01,FC,06,U,NU          COMMISSION
18  01,GA,02,U,NU          YEARS-OF-EDUCATION
19  01,HA,02,U,NU          YEARS-WITH-COMPANY
20  01,IA,02,U,NU          VACATION-DAYS
21  01,KA,02,U,NU          SICK-DAYS
22  01,LA,30,A,NU,DE       HOBBY
23  PA = PHON(BA)
24  SB = DE(4,5)
```

ファイル2　Finance file
（財務ファイル）

```
 1  01,AA,08,B,DE          PERSONNEL-NUMBER
 2  01,MC,PE(2)            MAJOR-CREDIT
 3  02,CC,18,A,NU,DE       CREDIT-CARD
 4  02,CL,04,U,NU,DE       CREDIT-LIMIT
 5  02,CB,04,U,NU,DE       CURRENT-BALANCE
 6  01,OC,07,A,MU(2),DE    OIL-CREDIT
 7  01,NW,08,U,NU,DE       NET-WORTH
 8  01,CR,02,U,NU,DE       CREDIT-RATING
 9  01,IP,PE(1)            INSURANCE-POLICY-TYPES
10  02,IC,25,A,MU(2),NU,DE INSURANCE-COMPANY
11  02,PA,06,U,MU(2),NU,DE POLICY-AMOUNT
12  01,CG,16,A,NU,DE       COLLEGE
13  01,VC,PE(12)           VACATION
14  02,OV,01,A,NU,DE       ON-VACATION
15  01,IV,15,A,NU,DE       INVESTMENT
16  01,SV,04,P,NU,DE       SAVINGS
17  01,BK,30,A,NU,DE       BANK
```



ファイル3　Automobile-file
（自動車ファイル）

```
 1  01,AA,20,A,NU,DE       MAKE-MANUFACTURER, MAKE
 2  01,AB,20,A,NU,DE       MODEL
 3  01,AC,15,A,NU,DE       BODY-TYPE
 4  01,BA,02,U,NU,DE       NUMBER-OF-CYLINDERS
 5  01,BB,03,U,NU,DE       HORSE-POWER
 6  01,BC,05,U,NU          PISTON-DISPLACEMENT
 7  01,BD,05,U,NU          WEIGHT-IN-POUNDS
 8  01,CA,10,A,NU,DE       COLOR
 9  01,DA,02,U,NU,DE       YEAR
10  01,DB,16,A,NU          SERIAL-NUMBER
11  01,FA,06,U,NU,DE       DATE-OF-LAST-CHECK
12  01,FB,06,U,NU,DE       MILEAGE-AT-LAST-CHECK
13  01,GA,08,B,DE          OWNER-PERSONNEL-NBR
```

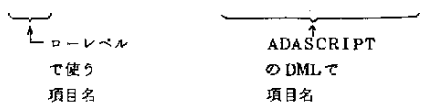


図２－６４　データ定義例

② カップリング

ファイル間の関係は、ＣＯＵＰＬＥによって任意の２つのファイルを関係づけすることができ、それぞれのファイルは他の８０個のファイルとカップリングできる。カップリングされるファイル間の構造は、複雑なネットワークを自由に構成できる。カップリングの方法は、１個のカップリングのために１回のＣＯＵＰＬＥユーティリティを実行し、必要なだけ繰返し行な

う方法をとっている。ＣＯＵＰＬＥに対しては、次のようなパラメータを与えなければならない。

FILE　×××$_1$（FN$_1$）　WITH　FILE　×××$_2$（FN$_2$）

×××$_1$, ×××$_2$ } 3桁以内のファイル番号

FN$_1$, FN$_2$ } 2文字からなる項目名（ディスクリプタ項目）

例

・ファイル１と２のPERSONNEL-NUMBERによるカップリング

FILE 01(AA) WITH FILE 02(AA)

・ファイル１のPERSONNEL-NUMBERとファイル３のOWNER-PERSONNEL-NBRによるカップリング

FILE 01(AA) WITH FILE 03(GA)

・ファイル２のPERSONNEL-NUMBERとファイル３のOWNER-PERSONNEL-NBRによるカップリング

FILE 02(AA) WITH FILE 03(GA)

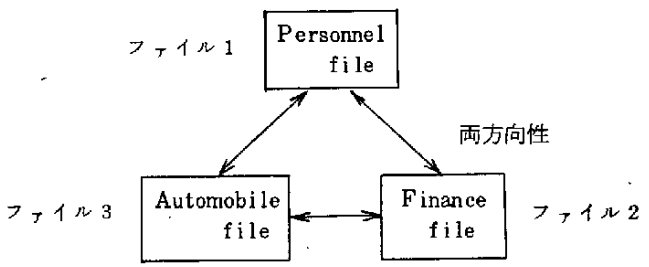


図２－６５　カップリング例

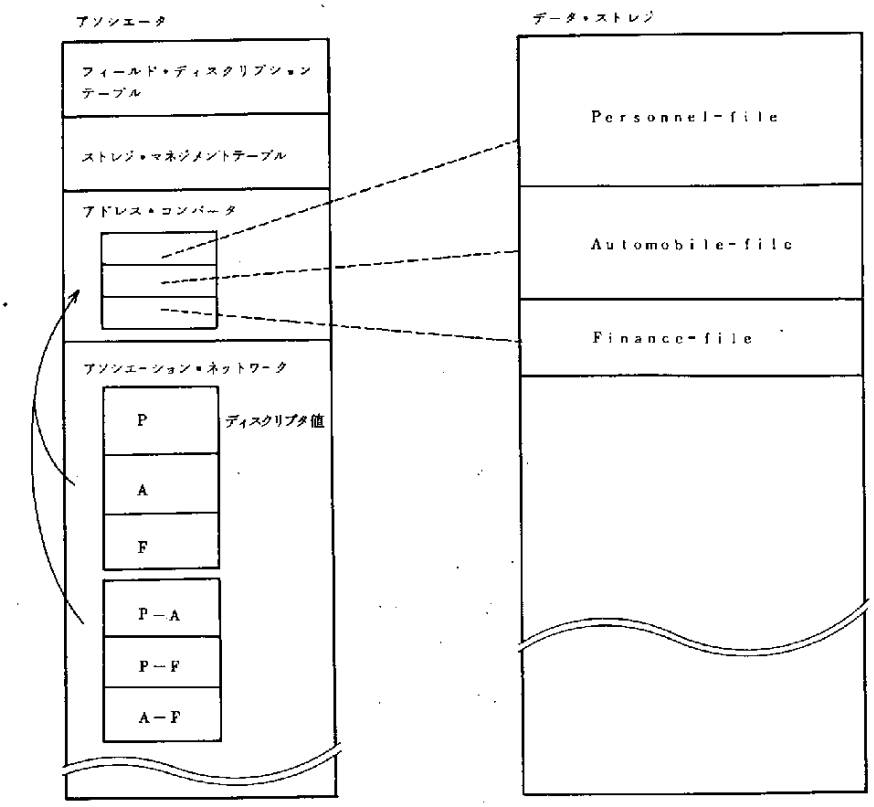


図２－６６　データベース構成例

③ ファイル／データ定義修正

ADDLOAD　ユーティリティ　……ファイルの追加
（ADAWAN，ADAZD１，ADAZD２）

FILEMOD　ユーティリティ　……アソシエータ・ネットワークの修正
- アンカップリング
- ディスクリプタの解除・登録
- 新項目の定義

DBMOD　ユーティリティ　……データベース修正
- ファイル削除
- フィジカル・スペースの増加
- その他

ADABAS　コマンド
- ADD
- UPDATE
- DELETE

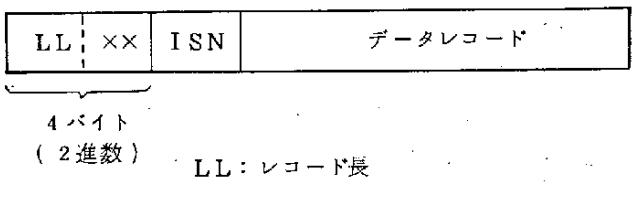
④ 特殊機能

a) 圧縮機能（compression）

個別ファイル（主データ）はデータ・ストレジにそのままの形態で記憶するが、この時圧縮機能によって、記憶領域の節約を行なうことができる。

データ定義方法は、定義カードの先頭に　ＶＡＲＩＡＢＬＥ　を指定することと、項目定義エントリで固定長（Ｆ，ＦＩ）以外のものに限って行なうことができる。

・データ・ストレジのレコード形式



・項目の圧縮

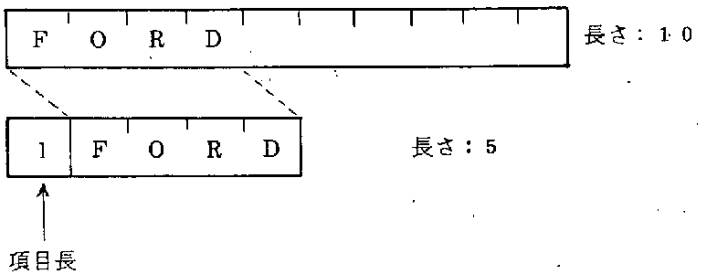


図2－67　圧縮の例

b) 音声化機能（phonetization）

音声化項目の定義

データ定義カードの後に次の形式で定義する。

pfn＝ＰＨＯＮ（fn）

pfn ：音声化項目名

fn ：項目名

指定できる項目は、英数字項目に限られる。（数字はスキップされ、全桁が数字ならばゼロになる。）

音声キーは、3バイトの2進数で内部に格納される。

o　ユーザは、特殊な業務に対して独特な音声化ルーチンを使って書き込むことができる。標準ＡＤＡＢＡＳ音声化ルーチンの代りに＂short description＂としてユーザ自身

の書き込みが可能である。

このルーチン名はＡＤＡＵＰＨであり、呼出し手続き例を次に示す。

```
          LA      13,SAVE
          LA       1,PARAM1
          L       15,=V(ADAUPH)
          BALR    14,15
          .
          .
          .
PARAM1    DC      F'0'
PARAM2    DC      F'0'
PARAM3    DC      F'0'
```

ＰＡＲＡＭ１に項目長、ＰＡＲＡＭ２に番地、ＰＡＲＡＭ３を２進数ゼロにして呼出すと、その結果、ＰＡＲＡＭ３に３バイトの音声キーの番地が戻される。

c) 暗号化機能（ciphering）

ファイルのローディングを行なうときに（ＬＯＡＤＥＲ）暗号化の指定をすると、暗号形式で格納される。

暗号コードは、ＬＯＡＤＥＲ（ＡＤＡＷＡＮ）の実行時に出力されず、データベースに格納されることもない。

８ケタの１０進数コードから暗号化が行なわれて、データを変換してデータ・ストレジに格納される。このジェネレーションにコード表が使われる。

なお、暗号化機能はファイル単位で実行できる。

C. ＡＤＡＢＡＳのＤＭＬ

ＡＤＡＢＡＳのＤＭＬには、次のようにデータベースを利用する層に応じて、数多く用意されている。

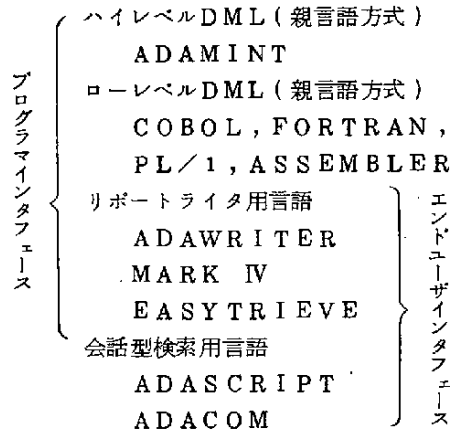


図２－６８　ＡＤＡＢＡＳのユーザインタフェース

ＡＤＡＢＡＳには、ＤＭＬとしてローレベルとハイレベルのＤＭＬ、および会話型検索用言語ＡＤＡＳＣＲＩＰＴ、さらにはレポート・ライタ用言語ＡＤＡＷＲＩＴＥＲが用意されている。

ローベルＤＭＬは、親言語方式（ＣＯＢＯＬ，ＦＯＲＴＲＡＮ，ＰＬ／１，ＡＳＳＥＭＢＬＥＲ）であり、親言語プログラムの指定されたバッファにパラメータやコマンド・コードを与えてＡＤＡＢＡＳにＣＡＬＬする方法である。ハイレベルＤＭＬは、ローレベルＤＭＬのマクロ形式であり、このアクセス・モジュールとしてＡＤＡＭＩＮＴ（ＡＤＡＢＡＳ Ｍａｃｒｏ ＩＮＴｅｒｆａｃｅ）がある。ＡＤＡＭＩＮＴアクセス・モジュールは、ＤＢＡが作成したマクロ命令をユーザが応用プログラムで呼出して使用する。

ローレベル／ハイレベルいずれのＤＭＬもコードや略記マクロ名を使わなければならない不便さがあるが、英語表現形式の使いやすい言語がＡＤＡＳＣＲＩＰＴである。オンライン端末のユーザは、ＡＤＡＢＡＳデータベースに対して会話型式で問い合わせを行なうことができ、システムからの「＊ＴＹＰＥ」メッセージに応答する形態がとられる。「＊ＴＹＰＥ」には、＊ＴＹＰＥ ＣＯＭＭＡＮＤ（コマンドと検索式条件の入力）、＊ＴＹＰＥ ＬＩＮＥ（行単位の修正）、＊ＴＹＰＥ 'ＳＨ','ＣＨ' ＯＲ 'ＥＸ'（入力情報の確認、修正、実行）という３種の形式があり、ユーザはそれに応じた情報をタイプインする。検索コマンドの代表的なものはＦＩＮＤコマンドである。ＦＩＮＤコマンドで検索条件を指定して、ＡＤＡＢＡＳデータベース（アソシエータ部を使って論理演算を行なう）をアクセスし、その結果をオンライン端末にＤＩＳＰＬＡＹ機能を使って出力することができる。その他、ＳＩＧＮＯＮ（あるいはＯＰＥＮ）コマンド（ファイルのオープン）、ＨＩＳＴＧＲＡＭコマンド（ヒストグラムの作成）や、ＴＩＭＥＲコマンド（処理時間の通知）、ＤＵＭＰコマンド（エラー時のダンプリスト作成）などがある。

表２－６　ＦＩＮＤコマンドに使用できる主なオプション

ＳＯＲＴ	検索情報を指定キーでソートする
ＵＰＤＡＴＥ	レコード更新
ＤＩＳＰＬＡＹ	検索結果の出力
ＰＡＧＥＳＩＺＥ	出力ページサイズの指定
ＴＩＴＬＥ	出力前のタイトル印字
ＡＣＣＵＭ	指定項目について総計
ＣＯＮＴＲＯＬ	指定キーごとに小計およびソート

```
ADASCRIPT - WAIT FOR '*TYPE' MESSAGE
*TYPE COMMAND
SIGNON PASSWORD ADACLASS WRITING PERSONNEL READING AUTOMOBILES.
ACCEPTED
OP, CID      , FNR 0, ISN 0, RSP 0
ISL 0, ISQ 0, OPT
SPO   , VAL
RB UPD=1,ACC=2.
*TYPE COMMAND
FIND PERSONNEL WITH STATE = DC, SORT BY NAME,
DISPLAY NAME, CITY, STATE, AGE,
ACCEPTED
11 RECORDS FOUND
REPORT READY FOR OUTPUT   (RW=48)

-
 7 DEC 76   4:17 AM                          PAGE 1

   LAST
   NAME                    CITY          STATE AGE

BREE                    WASHINGTON        DC    67
COLVILLE JR             WASHINGTON        DC    58
FLETCHER                WASHINGTON        DC    56
LYNN                    WASHINGTON        DC    46
PERRY                   WASHINGTON-       DC    38
```

「PERSONNELファイルからSTATEフィールドがDCであるレコードを捜しその結果をNAMEでソートし（NAME順に）、NAME，CITY，STATE，AGEの各項目（フィールド）を表示しなさい」

図２－６９　ＡＤＡＳＣＲＩＰＴ使用例

D. データ保護と機密保護

① データ保護機能

ハードウェア・エラーによるデータベースの物理的ブロックの破壊や、プログラム・エラーによるデータベースへの論理的エラーの発生から、正常な状態にもどす方法として次のものがある。

・データベース全体またはその一部を以前の論理的に正常な状態に回復する。

・データベース中のアクセス不能な物理的ブロックを修復する。

これらは、いずれもＡＤＡＢＡＳのＲＥＳＴＡＲＴユーティリティの機能である。ＲＥＳＴＡＲＴ機能を使用することによって、障害が起こる前の状態に自動的に再設定されて、ユーザ・プログラムは処理中であったトランザクションが再スタート可能である。ＡＤＡＢＡＳは、すべてのデータベースの変更を記録しておくためにデータ保護ファイルを使用する。この保護ファイルは、データベースの一部を変更するプログラムの実行中、ＡＤＡＢＡＳのニュークリアスが自動的に維持する。データ保護ファイルを維持するためのオーバヘッドを最小にする目的で、データ保護情報は、できる限り圧縮され、ブロックの全体の前と後の状態を書き出すのでなく、ブロックの修正される部分の前後の状態のみを書き出す。

もう一つの方法にＡＤＡＢＡＳのオートリスタート機能がある。この基本的機能は、ハードウェアのエラーによってデータベースが物理的に破壊されているか否かを確かめることである。データベースの更新コマンドを実行中にハードウェア・エラーが生じたが、要求された部分の物理的更新だけは正常に終了したという場合がある。

このような場合に、エラーが発生した正確な点を識別し、ＡＤＡＢＡＳ内部のコマンドのうちどのコマンドが正常に終了し、どのコマンドが正常に終了しなかったかを決定する。これらは、すべてユーザが介在しないで、オートリスタート機能が自動的に行なう。

修復後のプログラムの開始に当たっては、前の処理での未完了状態にある更新コマンドがないかをＡＤＡＢＡＳ作業用ファイルで調べる。もし、そのようなコマンドがあれば、データベースを、割込んだコマンドの前の処理に戻して、正常処理する。この自動処理によって、バックアップ用のコピーからのデータベースの再生をできるかぎり行なわないようにしている。

② 機密保護機能

ＡＤＡＢＡＳによって、２種類の機密保護が可能である。

・許されていないアクセスからの機密保護機能

・許されていない更新からの機密保護機能

両方に対して、各々２つのレベルの機密保護機能がある。

・ファイルレベル

・項目レベル

ＡＤＡＢＡＳの機密保護機能が働いているときには、各ユーザは、そのプログラム中にパスワードを与えなければならない。パスワードのチェックはＯＰＥＮコマンドを通して行なわれる。与えられたパスワードは、データ機密保護表内に存在していなければならず、もしあればパスワードはプロファイルに変換される。プロファイルは各ファイルに対して、アクセスと更新とに分離された形で存在し、パスワードが持つ使用権のレベルを示したものである。使用権のレベルは１５まで設定できる。パスワードのチェックがＯＫになると、ＡＤＡＢＡＳコマンドが許されるが、コマンドで使用するファイルがＯＰＥＮコマンドで指定されたものかどうか毎回チェックされる。

「アクセス」と「更新」の２つの機密保護は完全に独立である。そのため、あるファイルのレコードの項目すべてに対して読込みが許されるが、あるファイルのレコードでは特定の項目の更新のみが許される、というような設定も可能である。

もう１つの機密保護機能はＢで述べた暗号化機能である。これは、データベース作成時に暗号化コードを指定することによって、データが変換されて格納され、そのため他の暗号化コードを知らないユーザがデータの内容を見ようとしても、その内容がわからないようになっている。

E. その他の特徴

a) ＡＤＡＢＡＳのデータベースは、ＳＹＳＴＥＭ２０００と同様に、インバーテッド・ファイ

ルをベースとしているが、両者が根本的にちがうのは、ＡＤＡＢＡＳには陽なデータ構造の考え方が一切ないことにある。ＡＤＡＢＡＳでは、データベースはデータを生のままの集合であること以上の何物でもなく、ポインタやインデックスは分離してもっている。これは、集合理論に基づく明快な体系化によって高い評価をうけている関係モデルの思想に合致する方向であり、その点で注目されている〔ＩＳＨＩＩ７７ａ〕。

b) ＡＤＡＢＡＳの開発当初は、ローレベルのＤＭＬしかなかったために、ユーザにとって非常に使いにくいシステムであるといわれていた。ＳＹＳＴＥＭ２０００のような英文形式をとらずに、特定のバッファ領域に特殊な形式に従って書込まなければならず、さらには、検索式を作るときに、どの項目がインバーテッド・ファイルを有している（ディスクリプタ）かをあらかじめ知っておかなければならない不便さがあった。

c) ＳＹＳＴＥＭ２０００のようなプリ・プロセッサを必要としない点は良い評価を得ている。また、バッファ領域を介して、システムと直接コミュニケーションをする方式は、ＡＤＡＢＡＳパッケージそのものを１つのブラック・ボックスの感じにするもので、データベース・マシン実現の方向を示唆するものとして先進性がある。

d) ハイレベルＤＭＬ，さらにはオンライン検索用言語ＡＤＡＳＣＲＩＰＴの開発によって、ますます広範なユーザ層を得ようとしている。

e) 物理的に分散して保持されるデータベースを、論理的に１つに統合する、いわゆる分散型データベース機能をもっていることに注目されている〔ＩＳＨＩＩ７７ａ〕。

f) ＡＤＡＢＡＳとインタフェース可能なＤＤ／Ｄとしては、Datamanager と ADABAS Data Dictionary System がある。

2.3.2 ＳＹＳＴＥＭ２０００

A. 概要

ＳＹＳＴＥＭ２０００は、米国のＭＲＩ社が開発したＤＢＭＳであり、日本のユーザを含め全世界で約５２０社で使用されている〔ＯＨＩＳ７７〕。

ＳＹＳＴＥＭ２０００のデータ構造は階層型モデルであるが、任意のデータ項目に対してインバーテッドキーの指定が可能であるので、実質的には、網型モデルと同等の表現が可能である。ＳＹＳＴＥＭ２０００が稼働可能な機種は、ＩＢＭ３６０／３７０，ＵＮＩＶＡＣ１１００，ＣＤＣ６０００，ＣＹＢＥＲ７０の各シリーズである。また、データベースに対するアクセスは、自然言語を用いた（独立言語）方式とＣＯＢＯＬなどのコンパイラを用いた親言語方式の二方式がある。

B. ＳＹＳＴＥＭ２０００のＤＤＬ

ＳＹＳＴＥＭ２０００のＤＤＬは、ＣＯＢＯＬのファイル定義部に類似している。データベースの定義は、コンポーネント番号、名前、属性記述をもったコンポーネントの集まりである。コンポーネントは、データ項目、ＲＧ（Repeating Group：レコード）、文字列、機能定義の何れ

かである。ＳＹＳＴＥＭ２０００が提供するデータ型式は、name，text，integer，decimal，date，moneyの六つである。そして、各データ項目はＮＯＮ－ＫＥＹと宣言しないと自動的にインバーテッドファイルが生成される。ＳＹＳＴＥＭ２０００のデータベースは、データベース定義、インバーテッドファイル、実データを含むいくつかのファイルから構成されている。ＳＹＳＴＥＭ２０００のデータベースファイル構成を図２－７０に示す。

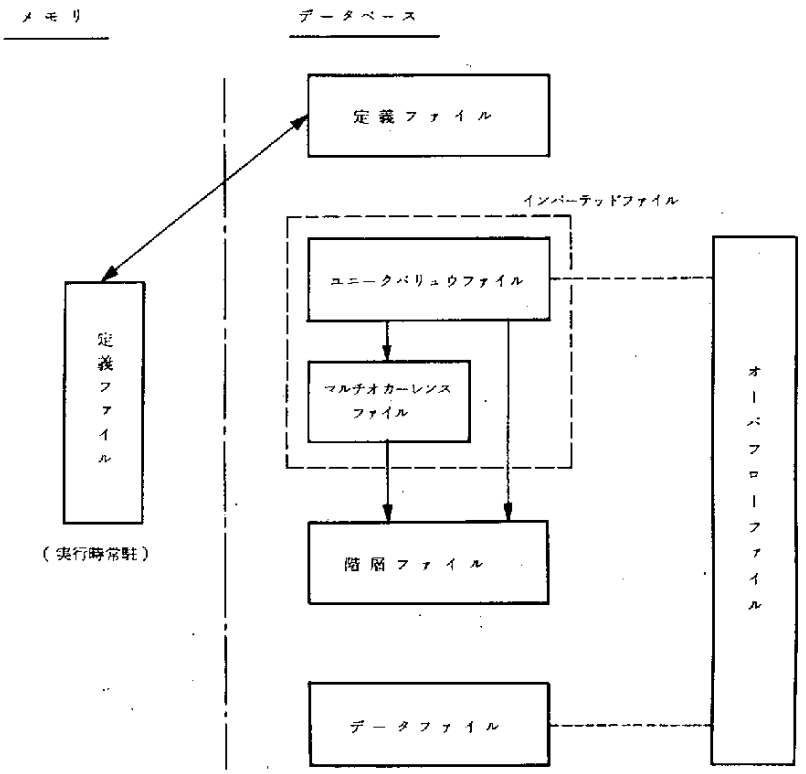


図２－７０　ＳＹＳＴＥＭ２０００のデータベースファイル

このように、１つのデータベースを分割している理由は、複数のユーザからの要求を同時実行する際のＩ／Ｏ待ち状態を極力避けるためである。

次に各ファイルの概要を述べる〔ＴＳＩＣ７７〕。

① 定義ファイル

データベース名、コンポーネント記述、機密保護情報、パスワード情報などを持ち、実行時にはメモリに常駐する。

② ユニークバリュウファイル

個々のキーデータ項目に対応するユニークバリュウ（論理アドレス）をもつ。

③ マルチオカーレンスファイル

複数のデータ項目が同一のユニークな値を持つ場合に、すべてのポインタを集めたものである。同一のユニークな値が複数発生する場合、ユニークバリュウファイルはマルチオカーレンス

ファイルのインバーテッドとなる。

④ 階層ファイル

ＲＧの親子関係、兄弟関係を管理しており、論理アドレスからディスク上の物理アドレスの対応を示している。

⑤ データファイル

ＲＧの発生順に格納されている。

⑥ オーバフローファイル

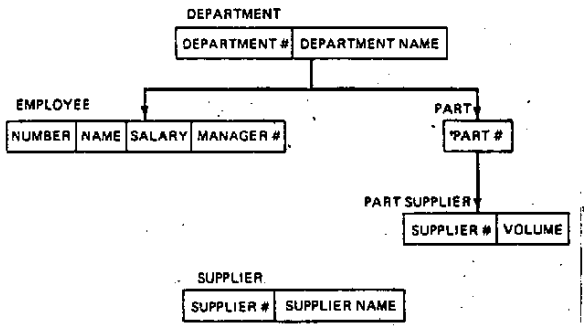
定義したデータ項目長より長いデータに対処するために用いる。このファイルは、ユニークバリュウファイルとデータファイルに用いられる。

⑦ その他

データベースの更新情報をもつロギングファイルがあるようである〔ＯＨＩＳ７７〕。

ＳＹＳＴＥＭ２０００は、インバーテッドによるアクセスを基本とし、階層ファイルを利用した階層型網型構造を可能としている。これは、用いられるデータを同一属性に基づいてグループ化を行なった場合に、グループ内の７０％が階層的構造を成す事に起因している。

図２－７１にＳＹＳＴＥＭ２０００のＤＤＬ例を示す。




```
DATA BASE NAME IS DEPARTMENT:
1* DEPARTMENT # (NAME X(3) ):
2* DEPARTMENT NAME (NON-KEY NAME X(20) ):
3* EMPLOYEE (REPEATING GROUP):
   4* NUMBER (NAME X(4) IN 3):
   5* NAME (NON-KEY NAME X(25) IN 3):
   6* SALARY (NON-KEY MONEY 9(5).9(2) IN 3):
   7* MANAGER # (NAME X(4) IN 3):
8* PART USED (RG):
   9* PART# (NAME X(4) IN 8):
  10* PART SUPPLIER (RG IN 8):
  11* SUPPLIER # (NAME X(4) IN 10):
  12* SUPPLIER VOLUME (INTEGER 9(7) IN 10):

DATA BASE NAME IS SUPPLIER:
1* SUPPLIER NUMBER (NAME X(4) ):
2* SUPPLIER NAME (NON-KEY NAME X(20) ):
```

図２－７１　ＳＹＳＴＥＭ２０００のＤＤＬ例

C. ＳＹＳＴＥＭ２０００のＤＭＬ

ＳＹＳＴＥＭ２０００のデータベースにアクセスする言語インタフェースは、次のように分類できる（図２－７２）。

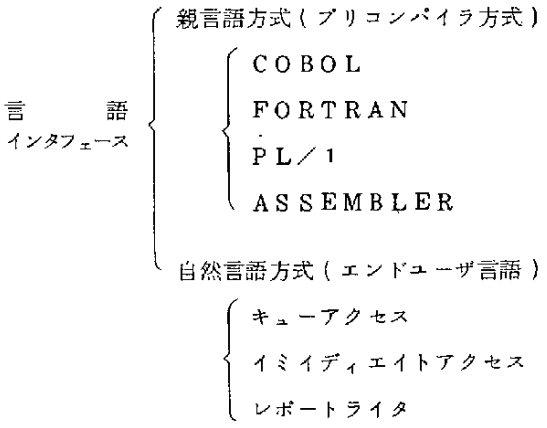


図2－７２　SYSTEM2000の言語インタフェース

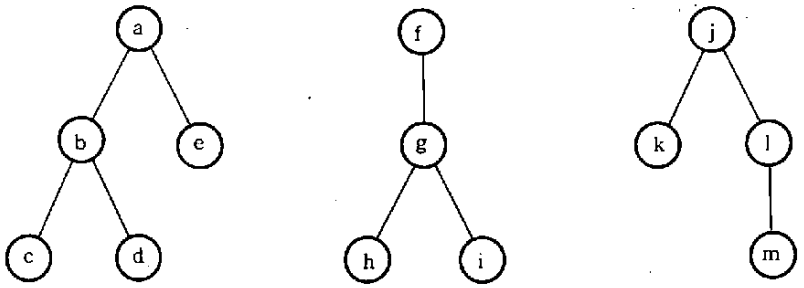
このように、ＳＹＳＴＥＭ２０００では親言語から出す方式と英語表現に準じた自然言語方式の二方式によるアクセスを可能としている。以下、それぞれについて解説を行なう。

① 親言語方式

この方式は、ＣＯＢＯＬなどのコンパイラを親言語とし、ＰＬＩ（Procedural Language Interface）を介してデータベースにアクセスする方式である。このために専用プリコンパイラが用意されており、データベースにアクセスするための手続への変換が行なわれる。

ＳＹＳＴＥＭ２０００では、プログラムの実行中にＲＧ間の関係についてセット／リセットする機能を有している。この機能は、ＬＩＮＫコマンドにより達成され、同時に１０個のＲＧを結ぶことができる。図２－７３に、ＬＩＮＫコマンドの実行前／後の状態変化を示す。

実行前　　データファイルA　　データファイルB　　データファイルC



LINKコマンド実行

LINK1 e TO f VIA 条件文（最大２６組）
LINK2 f TO k VIA 条件文
LINK3 k TO m VIA 条件文
LINK4 m TO g VIA 条件文
LINK5 g TO d VIA 条件文

実行後

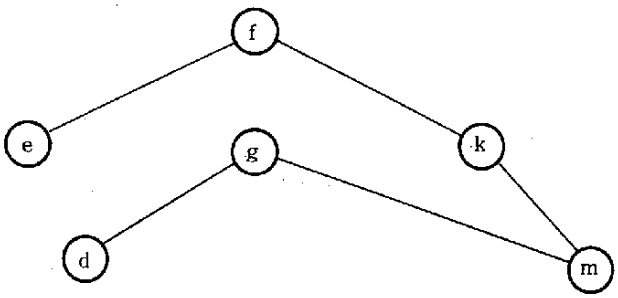


図２－７３　ＬＩＮＫコマンドの実行

このようにＬＩＮＫコマンドは、複数のデータファイル間においても実行可能である。この考え方を推し進めると、データベースのスキーマは事実上無いものと等しくなり、応用プログラムに与えられた自由度は相当高いものとなっている。なお、リセットは同じＬＩＮＫコマンド（ＬＩＮＫ０～９）を実行した時点で完了する。このような動的な関係付けが可能なのは、ＳＹＳＴＥＭ２０００がインバーテッドファイルをもっていることによる。

② 自然言語方式（エンドユーザ言語）

ＳＹＳＴＥＭ２０００がエンドユーザ言語を受け付ける状態は、control，define，access，report writer の４つである（図２－７４）。システム起動時はcontrol状態にあるが、システムの状態切換えは、図２－７４に示してあるコマンドを用いることにより簡単にできる。以下、各状態について概説する〔ＨＯＴＡ７７〕。

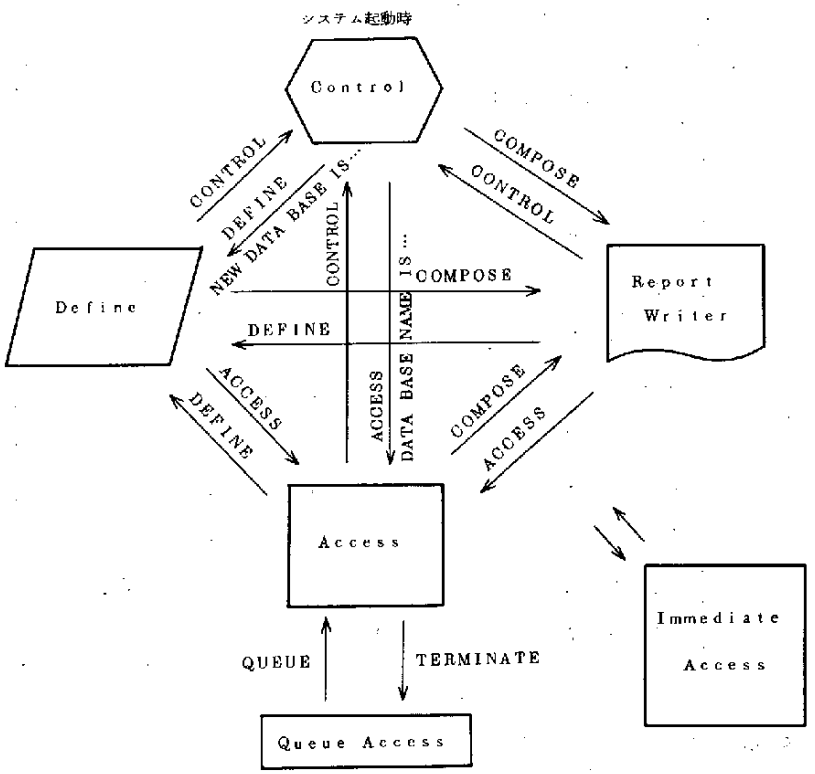


図２－７４　ＳＹＳＴＥＭ２０００におけるエンドユーザ言語の形態

a)　Control の状態

機密保護に関するチェック、パスワード制御、データベースのコピー作成、エラー回復用補助ファイルの準備ならびに操作など、データベースが動作できるための諸条件の整備を行う。

b)　Define の状態

データベースの定義あるいは定義の追加もしくは変更が可能である。

c)　Access の状態

i)　Queue Access

データベースへのアクセスをＱＵＥＵＥコマンドとＴＥＲＭＩＮＡＴＥコマンドで囲んで、リモートバッチ的に処理する。

ii)　Immediate Access

データベースへのアクセスは、会話型で処理される。

Queue Access の場合でも、Immediate Access の場合でも、データベースに対する検索や更新ができる。しかし、コマンド体系は独立した形になっている。両者のコマンドは次のとうりであるが詳しくは〔ＴＳＩＣ７７〕などに紹介されている。

問い合わせの例：LIST NAME WHERE DEPARTMENT HAS SUPPLIER＃EQ X:

表2−7 QUEUE ACCESSとIMMEDIATE ACCESSのコマンド比較

QUEUE ACCESS	IMMEDIATE ACCESS
LIST	LIST
PRINT	PRINT
UNLOAD	UNLOAD
LOAD	
APPEND(INSERT)	INSERT
ADD	ADD
ASSIGN	ASSIGN
CHANGE	CHANGE
REMOVE	REMOVE
REMOVE TREE	REMOVE TREE
REPEAT	

d) Report Writerの状態

形式指定に従ったリポート作成が可能である。一回の処理で、最大１００種まで作成が可能である。

D. その他の特徴

ＳＹＳＴＥＭ２０００は、ＩＭＳと同じように基本的には階層構造によるデータ構造表現を行なっている。しかし、ＩＭＳではレコードを検索するためには、ユーザが陽に階層構造をトラバースしなければならないのに対し、ＳＹＳＴＥＭ２０００では、ＤＢＭＳがその機能を遂行する。

ＳＹＳＴＥＭ２０００は、ＤＢＡにとって有益な多くのユーティリティ機能を提供している。例えば、更新情報を記録したり、機密保護機能としては、システム、データベース、コマンド、コンポーネントの各レベルでパスワードを設定している。さらに、端末からの使用資格チェック機能も備えている。また、ＳＹＳＴＥＭ２０００は多くの最適化機能とデバッギング道具を提供している。ＤＣ機能としてはＴＰ２０００、ＤＤ／Ｄ機能としてはＣＯＮＴＲＯＬ２０００がサポートされている。

2.3.3 TOTAL

A. 概要

ＴＯＴＡＬは、米国のCincom Systems 社が開発したＤＢＭＳであり、単独のＤＢＭＳとしては世界で最も多くのユーザをもち、日本の３０ユーザを含め、1,０００を超えている〔TOTT７７〕。ユーザ数が多い裏づけとしてＴＯＴＡＬが稼働可能な機種も他のＤＢＭＳに比べて圧到

的に多い。ＩＢＭ３６０／３７０シリーズをはじめ、System ３，ＰＤＰ１１／３５，４５，Honeywell ２００／２０００，６０／６６／６０００，ＣＤＣ　ＣＹＢＥＲシリーズ、ＮＣＲ，ＮＥＡＣ２２００シリーズ、ＡＣＯＳ７７など相当数になる。なお、日立のＤＢＭＳであるＰＤＭは、実質ＴＯＴＡＬに等しいといわれている〔ＨＯＴＡ７７ａ〕。

ＴＯＴＡＬは、網型のＤＢＭＳではあるがＣＯＤＡＳＹＬそのものではない。また、マスタデータセットとバリアブルデータセットの二種類のファイルを用意することにより可変データの処理を容易にしている。

B. ＴＯＴＡＬのＤＤＬ

ＴＯＴＡＬでは、データベースの対象とするデータを「物」に関するものと、「活動」に関するものに分けている。これに準じて、ＴＯＴＡＬは二種類のファイルを用意している。

・物に関するデータの格納……マスタデータセット

・活動に関するデータの格納……バリアブルデータセット

次に物と活動の関係例を示す。

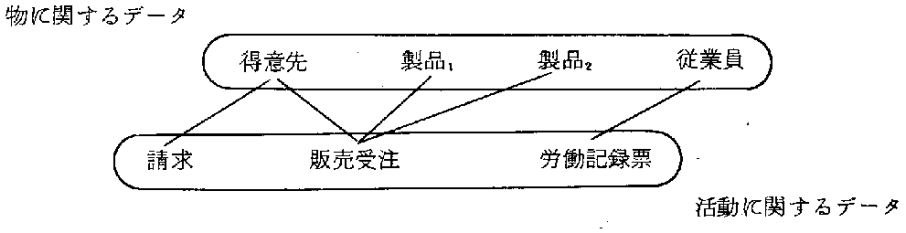


図２－７５　物と活動の関係例

ＴＯＴＡＬでは、マスタデータセットを１つ以上（何個でもよい）のバリアブルデータセットと結びつけることができる。また、バリアブルデータセットは１つ以上（何個でもよい）のマスタデータセットと結びつけることができる。しかし、マスタデータセット同志や、バリアブルデータセット同志を関係づけることはできない。しかし、１つのマスタデータセットに対して関連づけるバリアブルデータセットの数に制限がないから複雑な情報の表現も可能である。

ＴＯＴＡＬのＤＤＬは、ＤＢＤＬ（Data Base Definition Language）と呼ばれている。これは、独立した言語でデータベース生成プログラムＤＢＧＥＮの入力情報を提供する。ＤＢＧＥＮにより、アセンブラ言語のソースが出力される。次に、データベースにアクセスする応用プログラムがその定義を利用する。なお、データベースが用いるディスク領域は、ＦＯＲＭＡＴプログラムで前もって形式化が行われている〔ＴＳＩＣ７７〕。

ＴＯＴＡＬが構築するデータベースは、最高６５,０００のファイルを組み込むことが可能で、どのファイルのレコードも最高２５,０００のファイルのレコードと関係ずけることが可能である。なお、データベース内の各ファイルは、物理的に独立しており、データベースの変更や拡張が容

易である。図２－７６，２－７７にデータ構造例とデータ定義例を示す。ＴＯＴＡＬでは、ファイル名が４文字と限定され、項目名も４文字のファイル名を先頭にもつ８文字と制約されている。

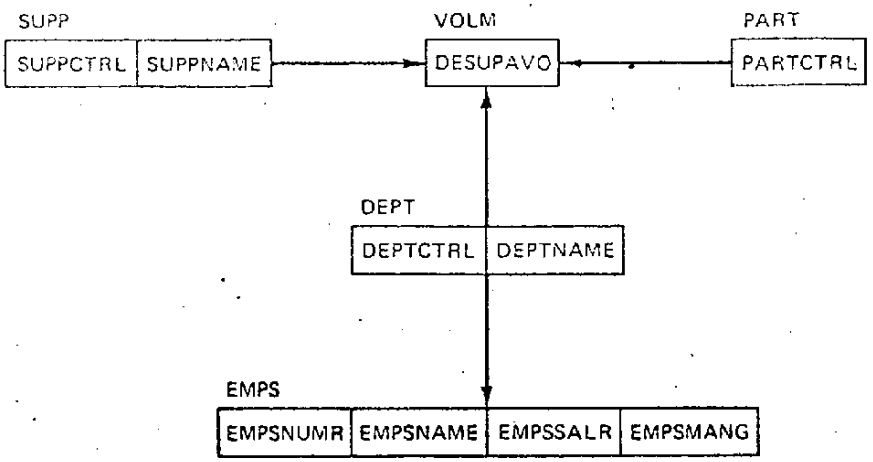


図２－７６　データ構造例

```
BEGIN_DATA_BASE_GENERATION:
DATA_BASE_NAME=COMPAN
SHARE_IO:
IOAREA=GENN

BEGIN_MASTER_DATA_SET:
DATA_SET_NAME=DEPT
IOAREA=GENN
MASTER_DATA:
DEPTROOT=8              REQUIRED BY TOTAL
DEPTCTRL=5              DEPARTMENT NUMBER
DEPTLKVO=8              LINK TO VOLM
DEPTLKEM=8              LINK TO EMPS
DEPTNAME=15             DEPARTMENT NAME
END_DATA:
DEVICE=2314             PHYSICAL ENVIRONMENT
TOTAL_LOGICAL_RECORDS=39920
LOGICAL_RECORD_LENGTH=80
LOGICAL_RECORDS_PER_BLOCK=20
LOGICAL_BLOCKS_PER_TRACK=4
TOTAL_TRACKS=20
END_MASTER_DATA_SET:
```

図２－７７(a)　マスタデータセットのＤＤＬ例

```
BEGIN_VARIABLE_ENTRY_DATA_SET:
DATA_SET_NAME=EMPS
BASE_DATA:
EMPSDEPT=5=DEPTCTRL      CONTROL FIELD FROM DEPT
DEPTLKEM=8               LINK FROM DEPT
EMPSNUMR=9               EMPLOYEE NUMBER
EMPSNAME=30              EMPLOYEE NAME
EMPSSALR=6               EMPLOYEE SALARY
EMPSMANG=9               EMPLOYEE MANAGER NUMBER
END_DATA:
    .
    .                    PHYSICAL ENVIRONMENT
    .
END_VARIABLE_ENTRY_DATA_SET:
```

図２－７７(b)　バリアブルデータセットのＤＤＬ例

C. ＴＯＴＡＬのＤＭＬ

ＴＯＴＡＬは、親言語方式を採用しており、ＣＯＢＯＬ，ＦＯＲＴＲＡＮ，ＰＬ／１，アセンブラの各言語からサブルーチンコールの方法によりインタフェースがとられている。データセットのアクセスは、三つの方式の何れかによって実行される。

・ハッシングによるアクセス

　マスタデータセットに対してはハッシング技術を用いて直接アクセスする方式

・ポインタによるアクセス

　バリアブルセットに対するアクセスは直接行なうことができず、マスタデータセットのレコードをアクセスした後で、ポインタ（リンケージパス）によりレコードをたどる方式

・シーケンシャルアクセス

　蓄積媒体上の物理的配置に従って順次アクセスするか、リンケージパスに従って順次アクセスする方式

ＴＯＴＡＬの優れている点の一つにハッシング技術がよくとりあげられる。例えば、割り当てたディスク容量の余裕率が８～１０％のときでも、データの１項目をアクセスするのに９５％は１回の物理的Ｉ／Ｏで十分と言われている。また、アクセスに際しては、レコード全体を意識しないで、項目のみを指定すれば良い〔ＴＯＴＴ７７〕。このことは、ＴＯＴＡＬにおけるデータの独立性をより高める役割を果たしている。次に、５つの項目から成るレコードに対し１と３の項目だけを、ユーザプログラムが要求したプロセスを示す。

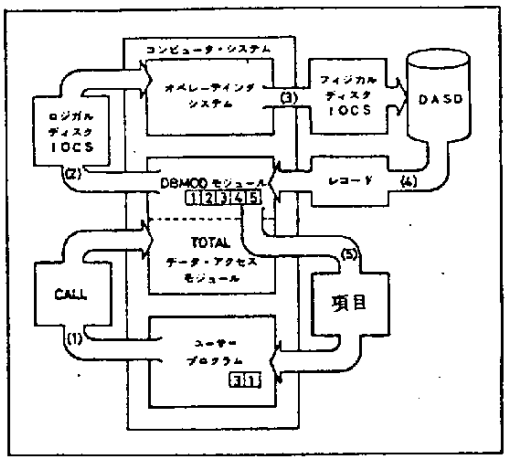


図２－７８　ユーザプログラムからのアクセスプロセス〔TOTT７７〕

① ＰＬ／１でＴＯＴＡＬのＤＢＭＳを用いる場合の一般形を以下に示す。

CALL DATBAS(operation, status, data set name, reference,
linkage path, control field, element list, record area, 'END.')

パラメータは必ずしも必要ではない。目的に応じたものだけを指定すればよい。

operation	実行する機能（例、ＲＥＡＤ）
status	ＣＡＬＬ実行完了状態をセットする変数
data set name	データベースで定義したデータセット名
reference	バリアブルデータセットの相対位置
linkage path	ＤＢＤＬで定義したチェイン名
control field	マスタデータセットに対してアクセスするとき使用するもので、ハッシュキーを指定
element list	データ項目名
'END'	ＥＮＤ で一連の記述が終了する。

② タスク管理関係

CALL DATBAS(operation, status, base, taskname, END)

operation

a) ＴＯＴＡＬ … ＴＯＴＡＬ ＤＭＬの全機能をローディングする

b) ＭＰＴＯＴ … ＲＥＡＤ／ＷＲＩＴＥ機能だけをローディングする
（３０％のメモリ節約になる）

c) ＱＵＥＳＴ … ＲＥＡＤ機能だけをローディングする
（６０％のメモリ節約になる）

d) ＤＥＱＵＥ … ＤＭＬあるいはＤＢＤＬからタスクを分離する

base	メモリーにローディングするＤＢＤＬのステップ名
taskname	ＴＯＴＡＬを用いるプログラム名

③ オープン／クローズ関係

CALL DATBAS (operation, status, data set name, 'END.')

operation

a) OPENM／CLOSM… マスタデータセットのオープン／クローズ

b) OPENV／CLOSV… バリアブルデータセットのオープン／クローズ

c) OPENX／CLOSX… データセットリストのオープン／クローズ

④ シーケンシャルアクセス関係

CALL DATBAS (operation, status, data set name, element list, record area, 'END.')

operation

a) SEQRM…マスタデータセットの全レコードを最初から最後までシーケンシャルに検索

b) SEQRV…バリアブルデータセットの最初のリンケージパスから順に、最初から最後までバリアブルデータセットをシーケンシャルに検索

c) SERLV…バリアブルデータセットの全レコードをシーケンシャルに検索

d) SEQWV…直前にSEQRV／SERLVコマンドで検索されたバリアブルデータセットのレコードのシーケンシャルな更新

e) RESTM／RESTV…レファレンスポインタのリセット

⑤ ダイレクトアクセス関係 (マスタデータセット用)

CALL DATBAS (operation, status, data set name, control field,
element list, record area, 'END.')

operation

a) READM…マスタデータセットのレコードで "control field" の内容が一致したものを検索

b) WRITM…更新

c) ADD－M… "control field" の重複チェックを行なった上でマスタデータセットにレコードを追加

d) DEL－M…マスタデータセットのレコードの削除 (関連するすべてのバリアブルデータセットは削除されなければならない)

e) LOADM…ADD－Mから "control field" のチェック機能を省いたもの

⑥ ダイレクトアクセス関係 (バリアブルデータセット用)

CALL DATBAS (operation, status, data set name, reference,
linkage path, control field, element list, record area, 'END.')

operation

a) READV…特定のリンケージパスに従ってバリアブルデータセットをシーケンシャル (前向き) に検索

b) ＲＥＡＤＲ…特定のリンケージパスに従ってバリアブルデータセットをシーケンシャル（後向き）に検索

c) ＲＥＡＤＤ

reference

i) 4文字の数値…特定のチェイン内の相対レコードを検索

ii) リンケージパスの直接指定（set switching）

d) ＷＲＩＴＶ…直前に検索されたバリアブルデータセットのレコードを更新

e) ＡＤＤＶＣ…バリアブルデータセットのレコードを追加する全リンケージパスの最後にセットする

f) ＡＤＤＶＢ…"control field"で決められたリンケージパス内の直前に検索したレコードの前にバリアブルデータをセットする

g) ＡＤＤＶＡ…直前に検索したレコードの後にバリアブルデータをセットする

h) ＡＤＤＶＲ…直前の検索コマンドで検索したバリアブルデータセットのレコードのリンケージパスを全て解放し、新レコードをセットし、リンケージパスの結合を計る

i) ＤＥＬＶＤ…直前の検索コマンドで検索したバリアブルデータセットのレコードを物理的に消去する

D. その他の特徴

ＴＯＴＡＬは、単独のＤＢＭＳとしては世界で最も多くのユーザを持ち、稼働可能なコンピュータの種類も一番多いと言われている。ここでは、その他幾つかの特徴について紹介する〔ＴＳＩＣ７７〕。

ＴＯＴＡＬにおける問い合わせの処理は、ＩＭＳに比べて非常に複雑である。ＴＯＴＡＬでは、1レコードを検索するたびに、ユーザ自身で通信領域（communication buffer）を用意しなければならない。さらに、レコードの条件設定は"control field"でのみ可能である。したがって、ほとんどの処理は、リンケージパスを経由してシーケンシャルに行われる。

ＴＯＴＡＬは網型モデルのシステムではあるが、ＤＢＴＧレポートに準拠していない。これは、ＣＯＤＡＳＹＬのＤＢＴＧレポートが発表される以前に、ＴＯＴＡＬが開発されていることからもわかる。例えば、ＤＢＴＧでいうセットの概念がない。

ＴＯＴＡＬのデータ構造表現機能は、秘密めいているが（cryptic）、マスタデータセットとバリアブルデータセットの関係付けの数に制限がないので、柔軟性に富んでいる。

ＴＯＴＡＬは、さらにＤＢＡにとって有益な幾つかのユーティリティ機能を提供している。例えば、データベースのローディング、ロギング処理、データベースの回復機能などがある。さらに、ＥＮＶＩＲＯＮ Ｉ，ＣＩＣＳ，ＴＡＳＫ－ＭＡＳＴＥＲなどのＤＣ機能を併用した会話モードによる処理も可能である。また、ＴＯＴＡＬとインタフェースが可能なＤＤ／Ｄとしては、Data Base Directory, Data Dictionary, Lexicon など数多い。

2.4　ＤＢＭＳの動向

本章では、ＤＢＭＳの背景と歴史、データベースモデル（階層型、網型、関係モデル）、代表的ＤＢＭＳについて議論してきた。これらの議論より、ＤＢＭＳの動向として多層化をあげることが出来る。ＡＮＳＩ／ＳＰＡＲＣによる三層スキーマは、多層化の主要な概念と考えられる。三層スキーマ概念に関連して、データベースの内部機構と独立なデータの意味を記述する概念スキーマとしてどのようなデータモデルを用いるかの多くの提案がなされている。

本節では、ＤＢＭＳ動向として、ＤＢＭＳの三層化を２.４.１項で、概念モデルを、２.４.２項で論じる。

2.4.1　ＤＢＭＳの三層化

既存のＤＢＭＳの分類において、これらのＤＢＭＳが種々の外部インタフェースを持っていることをみた。例えば、ＡＤＡＢＡＳは、関係的な問合せ言語を持っている。ＣＯＤＡＳＹＬ－ＤＢＴＧにおいても、同様な関係的な問合せ言語が提案されている〔ＣＯＤＡ７５〕。この様な外部インタフェースの多様化の背景として、ユーザが自分のアプリケーションへ適したデータモデル(構造)とデータアクセス言語とを使用することへの強いニーズをあげることが出来る。FryもＤＢＭＳが今後各種の（関係、網型、階層型）ユーザ・インタフェースを提供するようになることを指摘している〔ＦＲＹ７６〕。

もう１つの重要な点は、情報を実際に格納するコンピュータに依存したアクセス・パス、格納媒体上でのデータの物理的構成等の物理的問題と、論理的なデータモデルとの明確な分離である。１つのデータモデルを、各コンピュータは各々に適した最適の形態でインプリメンテーションをし、有効に動作させる。コンピュータの種々の物理的変化に対して、このデータモデルと言語とは何の影響も受けることはない。

ユーザが各々のアプリケーションへ適したデータモデルと言語とを用いようとする点と、データベースの物理的内部機構と論理的データモデルとの分離との２点を上記に指摘した。これらのことより、ＤＢＭＳ内部に種々のデータモデルと言語との共存を達成することが必要になってきている。これは、いわゆる共存アーキテクチャ（coexistence architecture）〔ＮＩＪＳ７６ｂ〕の達成である。共存アーキテクチャ達成のためには、データへ対する１つの視点から、これと意味論的に等価である他の視点への変換を可能とすることが必要である〔ＦＡＬＫ７７ｃ〕。まず、データベースのアプリケーションに依存した面と、実際のコンピュータでの効率に関連した面との分離が必要である。前者は個々のユーザアプリケーションへ適したデータモデルと言語であり、データベースの内部機構とは独立である。これはＡＮＳＩ／ＳＰＡＲＣの外部スキーマ（external schema）〔ＡＮＳＩ７５，ＪＥＦＦ７６，ＤＡＬＥ７６〕へ対応する。後者は、データベースを実現するコンピュータ固有の機能に基づいた有効な内部機構の記述である。これは、ＡＮＳＩ／ＳＰＡＲＣの内部スキーマ（internal schema）対応する。

先に述べた様に、種々の外部スキーマと内部スキーマとの間で、データベースの意味を保存しな

がら、変換を行なう必要がある。この変換は、変換コストを最少とするように行なわれねばならない。このため図２－７９に示すように、１つの共通スキーマを設定し、これと内部及び外部との変換を行なうことによって、変換コストを最少化できる。このスキーマは概念スキーマ（conceptual scheme）〔ＡＮＳＩ７５，ＴＳＩＣ７６，ＳＴＥＥ７６〕と呼ばれる。概念スキーマに要求されることは、データベースの意味が、種々の変換を通して完全に保存されることである。よって概念スキーマは、データベースの意味の完全で一意な形式的記述である。概念スキーマ層におけるデータモデルを概念モデルと呼ぶ。

これら３つのスキーマの関係を図２－７９に示す。同様のアーキテクチャは、SenkoのＤＩＡＭⅡ〔ＳＥＮＫ７３，７６ａ，ｂ〕においても示されている〔図２－８０〕。

最近ＣＯＤＡＳＹＬ－ＤＢＴＧにおいても、関係モデルとＡＮＳＩ／ＳＰＡＲＣの三層スキーマ概念の影響を受けて、次の様な３層化を考えている〔ＵＥＭＵ７８〕。

1) サブスキーマ

2) スキーマ

3) 格納スキーマ

新たに加えられた格納スキーマは、記憶構造格納のためのＤＳＤＬ（Data Storage Description Language）を設けている。

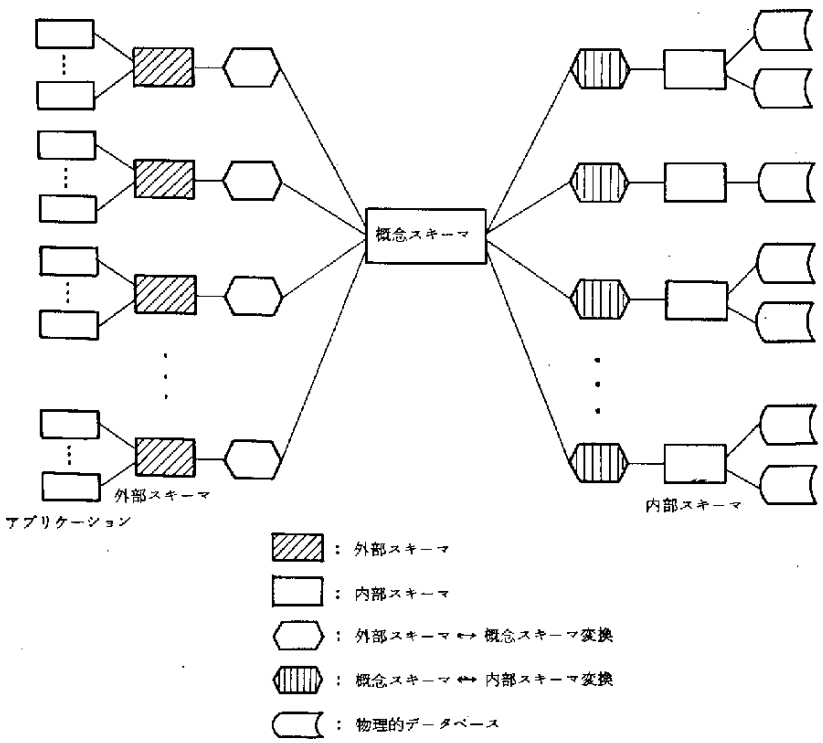


図２－７９　三層スキーマの関係

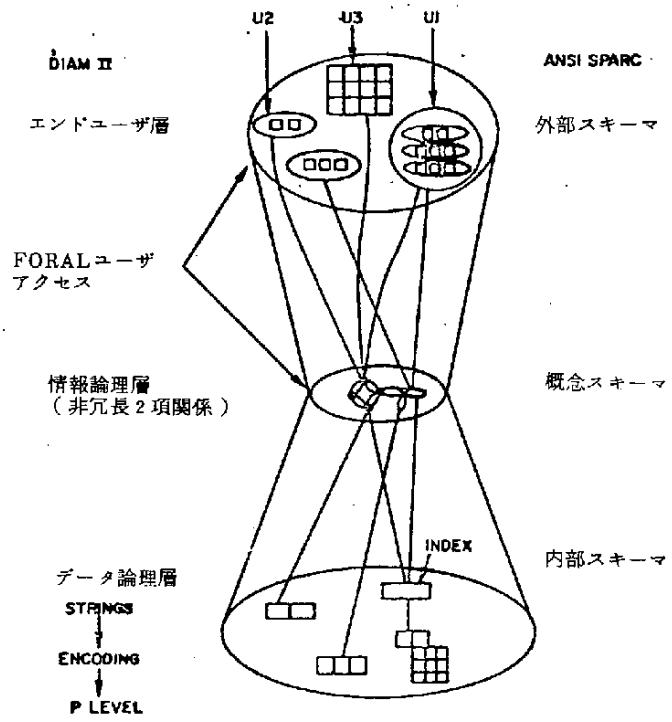


図２－８０　ＤＩＡＭ扇形図式

ＤＩＡＭⅡでは、ＡＮＳＩ／ＳＰＡＲＣの３つのスキーマへ対応して、エンドユーザ、情報論理（Info-logical)、データ論理（data logical）の３つの層を持っている〔ＳＥＮＫ７６ａ〕。エンドユーザ（end-user)層は、多くのユーザとプログラムにデータ構造を提供するとともに、既存のプログラム（ＦＯＲＴＲＡＮ，ＣＯＢＯＬ，ＩＮＳ，ＰＬ／Ｉ等）の持つデータ構造を有効に効果的に利用出来ることを目的としている。エンドユーザ層は、外部スキーマへ対応する。情報論理層（infological level)は、概念スキーマへ対応する。この層は、種々のエンドユーザ層とデータ論理層とへ写像されるもので、非冗長かつデータ独立な層である。又、システム保全性と複数ユーザによるデータ共有を制御も行なう。データ論理層（data logical level）〔ＳＥＮＫ７６ｃ〕は、内部スキーマへ対応している。この層は、コンピュータの有効利用のために一般的なファイル編成を備えている。図２－８０において、扇形図式の上側と下側が広がっているのは、１つの概念モデルへ対して種々のデータモデルが存在することを示している。ＤＩＡＭⅡでは、情報論理層として後述する事象集合モデル（entity-set model）を用いている。ユーザのＤＩＡＭⅡへの非手続的アクセス言語としてＦＯＲＡＬ〔ＳＥＮＫ７５，７６ｂ〕を、エンドユーザ層及び情報論理層へ備えている。

種々のスキーマが、１つ又は同一のＤＢＭＳ内に共存することを許す共存（co-existence）ＤＢ-ＭＳへのアプローチは、上述した様に、１つの概念スキーマを設定し、内部と外部との変換を行なわせるものである〔ＮＩＪＳ７６ｂ，７７ｂ，ＢＩＬＬ７７〕。概念スキーマにおけるデータモデル（これを概念モデルと呼ぶ）の提案は〔ＮＩＪＳ７６ｂ，７７ｂ〕のなかに多くの研究者によってなされている。先に述べたように、概念モデルは、データの意味の記述である。概念モデルは、

種々のスキーマの変換において、意味を保存させるために、データベースの意味の形式的で完全で一意な記述であることが何よりも要求される。次項では、現在提案されている幾つかの概念モデルについて検討をする。

2.4.2 概念モデル

概念モデルにおける必要な性質は、このモデルが他のスキーマに対して中心的なものであるために重要である。概念モデルに対する要求項目として下記の点が上げられている〔ＢＩＬＬ７７，ＮＩＪＳ７７b,c，ＦＡＬＫ７７ｃ，ＳＥＮＫ７７ｃ〕。

1) 意味の形式的かつ完全な記述
2) 定常性
3) 変換の容易性
4) 発展性
5) 理解の容易性（即ち、実世界へよく対応していること）

先に述べたように、概念モデルはデータモデルからアプリケーションの問題に依存した部分と、コンピュータ上での有効性に関連した部分とを除去し、ただデータの意味が保存されることを保障するとともに、種々の外部スキーマでの意味の一意な解釈を保障することが必要である。このため概念モデルが、先づ第１に意味の形式的で完全な記述であることが要求される。第２に重要な点は、外部スキーマは定常的であることである。内部スキーマ及び外部スキーマの変化と多様化とに対して、概念モデルが不変であることである。他に、各種スキーマへの変換が容易であること、概念モデル自身の発展性、人間へ対する理解の容易性が重要である。

現在はいくつかの概念モデルが提案されている。これらを論じる前に、既存のデータモデルの概念モデルとしての問題点を論じる。次に、代表的概念モデルとして、２項関係モデル、事象・関係性モデル、対象・役目モデルについて検討する。最後にKershberg, Biller等による種々の概念モデルの統合化へのアプローチを述べる。

A. 既存データモデルへの批判

ここでは、既存データモデルとして関係モデルと網型モデルとを考える。

① 関係モデル

ここで述べる関係モデルとは、Coddのｎ項関係モデル〔ＣＯＤＤ７０〕である。関係モデルは、網型モデルと階層型モデルへ対して、高度のデータ独立性と非手続性と強固な数学的基礎とを特徴とする点は、概念モデルへの候補となり得よう。又、Faginによる多値従属性概念の導入によって、より多くの従属関係を表現出来るようになっている。しかし、2.2.3項の関係モデルにおいて指摘したように、数学的概念に基づいた構文的研究の大きな成果の一方で、意味の問題が指摘されている〔ＳＣＨＭ７５，ＣＨＥＮ７６〕。

属性の集合としての関係の実世界における意味とは何なのだろうか。この問題に、数学的観点から答えることは出来ない。例えば、正規化の問題を考えてみよう。正規化は、しばしば、実世界において意味のない関係を生成してしまう〔ＳＣＨＭ７５，ＭＯＵＬ７６〕。関係ＥＭＰ（ＥＭＰ＃，ＮＡＭＥ，ＤＥＰＴ＃，ＤＥＰＴＭＧＲ，ＮＯ－ＯＦ－ＹＥＡＲ）を考える。関係ＥＭＰ内の関数従属は、ＥＭＰ＃→ＮＡＭＥ，ＤＥＰＴ＃，ＤＥＰＴＭＧＲ，ＮＯ－ＯＦ－ＹＥＡＲ；ＤＥＰＴ＃→ＤＥＰＴＭＧＲであり、２ＮＦである。第３正規化によって、ＥＭＰ１（ＥＭＰ＃，ＮＡＭＥ，ＤＥＰＴ＃，ＮＯ－ＯＦ－ＹＥＡＲ）とＤＥＰＴ（ＤＥＰＴ＃，ＤＥＰＴＭＧＲ）との２つの３ＮＦ関係を生成する。しかし、社員の属性に興味を持つ企業体（enterprise）では、ＤＥＰＴ関係は意味のないものである。

この様に属性の従属構造に基づく正規化は、４ＮＦ（又は３ＮＦ）の幾つかの集合を生成し得る。関係のある集合は、実世界において意味のない関係を持ち得る。関係モデルは、実世界における事実をモデル内に一意に格納するための何の標準形も用意していない〔ＳＥＮＫ７７〕。

これと関連して、関係演算は意味のあいまいさを導く。この問題を次の例で考えてみる。社員に関する情報を持った関係ＥＭＰ（ＥＭＰ＃，ＮＡＭＥ，ＤＥＰＴ＃，ＤＥＰＴＭＧＲ，ＮＯ－ＯＦ－ＹＥＡＲ）と船の情報を表わすＳＨＩＰ（ＳＨＩＰ－ＮＯ，ＮＡＭＥ，ＮＯ－ＯＦ－ＹＥＡＲ）とのＮＯ－ＯＦ－ＹＥＡＲについての結合を行なえる。しかしこの意味は何なのだろうか。関係ＥＭＰにおける属性ＮＯ－ＯＦ－ＹＥＡＲは社員の年齢であり、関係ＳＨＩＰでは船の年齢である。２つの関係の属性の領域は同一であるが、関係内での各々の属性の意味は異なっている。関係モデルデータベースシステムは、このことをユーザに対して警告することも出来ない。関係モデルにおける属性は、同一関係内で同一名を持つ領域を分離するために用いられているだけであり、意味を表わすものではない。関係モデルは、その基本概念として、関数従属及び多値従属との２つの従属関係を用いている。従属関係の問題として従属の推移性によって生成される属性間の従属関係の意味のあいまいさである。先に指摘したように、ＥＭＰ＃→ＭＧＲ＃，ＭＧＲ＃→ＳＡＬＡＲＹより関係従属ＥＭＰ＃→ＳＡＬＡＲＹを導き出せる。このＥＭＰ＃→ＳＡＬＡＲＹは、社員と彼の上司の給料との関係であって、社員と彼の給料との関係ではない。このことの意味を、関数従属によっては表現出来ない。

関数従属は、関係内の２つの属性間に１つ以上存在することを許されない。しかし属性間の関係として複数存在することは十分にあり得る。しかし、関係モデルでは表現出来ない〔ＳＥＮＫ７６ｂ〕。

他の重要な問題は、関係モデルが個々の対象を、その属性と独立に扱うことが出来ない点である〔ＨＡＬＬ７６，ＭＯＵＬ７６，ＣＨＥＮ７６〕。関係モデルは、対象の属性間の関係によって、その対象を表現している。関係は、これが結合している対象の全てが存在する場合にのみ存在出来る。対象は、関係の存在の有無に関わりなく個々独立して存在し得る。しかし、関係モデルでは、関係の存在なしに対象を扱うことは出来ない。このことは２項関係モデル及び事象・関係モデルの関係モデルへ対する重要な批判点である。

② 網型モデル

網型モデルは、図２－８１～８４に示すようなBachman〔ＢＡＣＨ６９〕によるデータ構造図（data-structure diagram）によって表わせる。この図式は、概念層における網構造を表わしたものである。又、この図式は、ＨＩＳのＩＤＳ（Integrated Data Store）データベース・システムにおける物理的なアクセスパス構造からの抽象化であった。

図において、四角はレコード型を表わし、矢印はデータ構造集合（data-structure set）を表わしている。図２－８１においてＤＥＰＡＲＴＭＥＮＴレコードは親レコードであり、１つの親レコードはn個（ｎ≧0）の子レコードを持つことを示している。網型モデルにおける

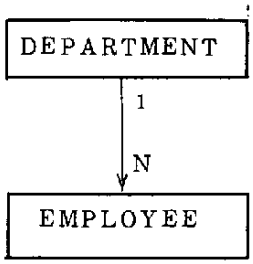


図２－８１　１：ｎ関係

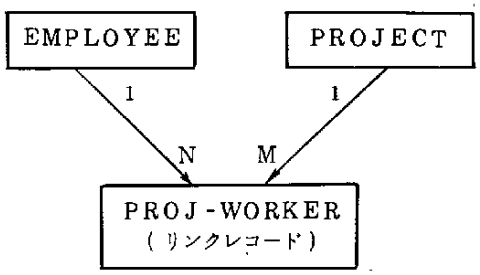


図２－８２　ｎ：ｍ関係

矢印の意味は、２つのレコード型間の関係であるとともに、親レコードから子レコードへのアクセスパスが存在していることでもある。この意味で、データ構造図は、概念モデルの記述ではなく、レコード編成の記述と考えられる。又、矢印は、一方向の関係のみを表わし、両方向関係（図２－８３）を表わしていない。又、図２－８４の様な再帰的関係も表わせない。

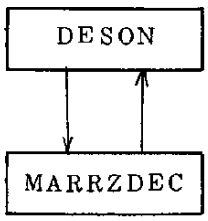


図２－８３　両方向関係

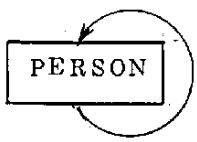


図２－８４　再帰的関係

B.　２項関係モデルと事象集合モデル

２項関係モデルにおいて、基本的事実は、２項関係の対として表わされる。２項関係（binary relation）をデータの意味の記述のために利用することは、Abrial〔ＡＢＲＩ７４〕において

導入された。Abrial はこの論文の中で、データベース・システムの論理層の概念と、２項関係の利用の第１歩として人工知能における意味網（semantic network)とを導入した。彼は意味網の節において、まだｎ項関係を用いていたが、２項関係が意味記述に重要であることを指摘した。Abrial のモデルの問題点として、次の点が指摘されている〔ＳＥＮＫ７７ａ〕。

1) 対象（object）の名前付けに対して、基本的すぎる名前付け構造を用いている。
2) アクセス言語が、全く手続的である。

Bracchi〔ＢＲＡＣ７６〕は、ｎ項関係モデルと２項関係モデルの比較を行ない、ｎ項関係モデルの欠点を指摘している。Senko は、ＡＮＳＩ／ＳＰＡＲＣにおける三層スキーマと同様な階層を持つＤＩＡＭ Ⅱ の情報論理層（概念スキーマに対応）のデータモデルとして２項関係性を用いた事象集合モデルを使用している。この様に、概念モデルとして２項関係性は多く用いられている。一方で、Falkenberg〔ＦＡＬＫ７７ｃ〕によって、モデルの再編性の困難さのための発展性の欠如が指摘されている。

① Abrial のモデル

Abrial のモデル（データ意味論モデル）における基本要素は、対象（object）と呼ばれる。例えば、"ＰＥＴＥＲ"，"ＪＯＨＮ"，"ＴＯＭ"や"ＢＬＡＣＫ"，"ＢＬＵＥ"は対象である。対象は、共通性質を持ったクラスへ分類出来る。この共通性質を持った対象の集合をカテゴリ（category）という。例えば、"ＪＯＨＮ"，"ＴＯＭ"は"ＰＥＲＳＯＮ"カテゴリへ属し、"ＢＬＵＥ"，"ＢＬＡＣＫ"は"ＣＯＬＯＲ"カテゴリへ属する。対象は、抽象的対象（abstract object）と具体的対象（concrete object）との２つがある。このモデルにおける２項関係とは、２つのカテゴリ間の連想（association)として定義される。この２項関係は２つのアクセス関数によって特徴化される。アクセス関数は、あるカテゴリから他のカテゴリへ達するためのものである。アクセス関数は、定義域のカテゴリと、値域のカテゴリとの対応濃度（cardinarity)を示す最小・最大指示パラメータを持っている。図２－８５は、カテゴリ著者（ＡＵＴＨＯＲ）と著書（ＢＯＯＫ）との２項関係を示している。

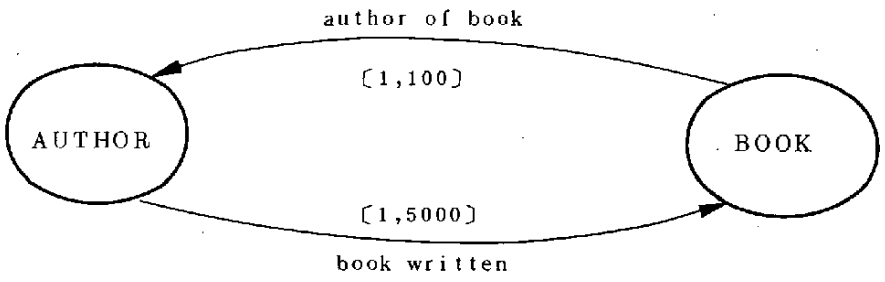


図２－８５　ＡＵＴＨＯＲとＢＯＯＫとの２項関係

図２－８５の円はカテゴリを表わしている。この図の意味は、１冊の本は最小１人最大１００人の著者を持ち、１人の著者は最小１冊最大5,000冊の著書を持つことである。この関係は形式的に次のように表記される。

rel（AUTHOR，BOOK，book written＝afn〔１，５０００〕，
author of book＝afn〔１，１００〕）

book writtenは、ＡＵＴＨＯＲカテゴリからＢＯＯＫカテゴリへのアクセス関数であり、author of bookはＢＯＯＫからＡＵＴＨＯＲへのアクセス関数である。このアクセス関数を用いて

author of book（雪　　国）＝　川端康成
book written　（川端康成）＝｛雪国、伊豆の踊子、…｝

のように必要な対象をアクセス出来る。アクセス関数における対応濃度は、カテゴリ間の保全性制限として用いれる。

このモデルにおいて、対象は、他の対象との関係性によって記述される。データの意味記述は、カテゴリ間の２項関係と、そのアクセス関数によって表現され、形式的自己記述性（self-descriptive）を目指している。データへのアクセスは、アクセス関数を用いたプログラムによって手続的に行なう。Senko〔ＳＥＮＫ７７ａ〕はこの点をデータアクセスの非手続性として批判している。しかし、この点は、それまでの伝統的な手続的言語ユーザへ対する非手続的データアクセスへの橋渡しとなったものである。

Abrial のモデルを用いてグルノーブル大学のAdibaは、既存のＩＭＳ，ＳＯＣＲＡＴＥ，ＣＯＤＡＳＹＬ－ＤＢＴＧ，Ｓ２０００，ＴＯＴＡＬ，ＡＤＡＢＡＳ，関係モデルＤＢＭＳ等のＤＢＭＳを一意に記述可能なことを示している〔ＡＤＩＢ７６，７７〕。彼は、Abrial のモデルを分散型ＤＢＭＳにおける統一的な概念モデルとして使用することによる異種ＤＢＭＳの統合を試みている〔後述する3.3.4の分散型データベースＰＯＬＹＰＨＥＭＥを参照のこと〕。

② Bracchi のモデル

Bracchi，Paolini とRelagtti〔ＢＲＡＣ７６〕は、概念モデルとして２項関係モデルの提案を行なった。このなかで、ｎ項関係モデルと２項関係モデルとの比較を行なった。彼等は、ｎ項関係モデルが

1) 論理的記述と物理的記述を混在させていること（ｎ項関係モデルはレコードの性質をまだ持っている。）と、
2) 最適化及び再構成における困難さを持っていることとを指摘している。

これに対して、２項関係モデルは次の特徴を持っている。

1) n項関係モデルにおいて意味論的に問題となった関係の属性間の従属概念を2項関係モデルは必要としない。

2) あらゆる関係性（1：n，n：m，1：1）を2項関係モデルは表現出来る。

3) n項関係モデルにおける関係内連想と関係間連想との区別が、2項関係モデルでは必要ない。

③ Senkoのモデル

Senko〔ＳＥＮＫ７３〕によるモデルは、2項関係性に基づいた事象集合モデル（entity set model）と呼ばれる。現実世界の基本要素は、事象（entity）と呼ばれる。事象とは、ある企業体（enterprise）にとって意味のある対象（object）、物、又は概念である。事象は、Abrial モデルの対象（object）の概念へ対応する。共通特徴を持つ事象の集合を、事象集合（entity set）という。この場合の、事象の特徴を属性（property）と呼ぶ。事象に関する事実（fact）とは、その事象の属性がある値を持つことである。

Senkoは、ＤＩＡＭ Ⅱ の情報論理層において、事象集合モデルを用いている。実世界の情報論理層における表現は、名前によって行なわれる。事象は、"Peter Jones","Bule","２７"といった名前を持つ。この名前を事象名（entity name）と呼ぶ。共通属性を持つ事象名の集合を、事象名集合（entity-name-set）と呼ぶ。事象名集合を参照する場合は、"ＮＡＭＥ","ＣＯＬＯＲ","ＱＵＡＮＴＩＴＹ"といった事象名集合名（entity-name-set-name）を用いる。

事象は、事象名集合名／事象名の対によって表わされる。例えば、ＮＡＭＥ／Peter Jones，ＥＭＰＬＯＹＥＥ－ＮＯ／２５８９，ＮＯ－ＯＦ－ＹＥＡＲＳ／２７である。事象は、他の事象との連想（association）によって表現される。この様な2つの事象の名前の間の連想を、事実表現（fact representation）という。モデルは、事象についての事実表現を格納することによって実世界の事象を表わす。図２－８６は、事象についての事実表現例である。事象"ＥＭＰ－ＮＯ／０１２３４５"のＤＥＰＴ－ＯＦ－ＥＭＰは"ＤＥＰＴ－ＮＯ／４２０"事象である。

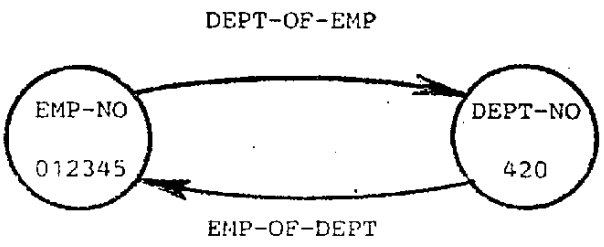


図２－８６　事実表現例

る。この場合のＤＥＰＴ－ＯＦ－ＥＭＰは、事象"ＤＥＰＴ－ＮＯ／４２０"が、事象"ＥＭ-

P－NO／０１２３４５"を表現するために演じる役目（role）である。EMP－OF－DEPTは、これと逆の役目をする。事実における重要な点は、２つの異なった文脈（context）のどちらかを用いて解釈されることである。もし、我々が文脈EMP－NO／０１２３４５を用いているならば、事実は社員の属性の１つへ値を与えること"DEPT－OF－EMP is ４２０"として解釈される。逆に、文脈"DEPT－NO／４２０"を用いているなら、"EMP－OF－DEPT is ０１２３４５"として解釈される。事象の事実は、役目名／事象名集合名／事象名の３つ組によってなされる。

このモデルにおける関係性（relationship）は、２項関係だけである。役目は、Abrialモデルのアクセス関数へ対応している。保全性は、定義域事象集合と値域事象集合との対応濃度の最大値によって保たれる。

２項関係モデルは、実世界の事象間の実際の意味論的連想（semantic association）を表現できる。関係モデルおいて問題となっている属性（attribute）間の従属構造を用いることなく、このモデルは事象間の意味を記述出来る。n項関係モデルでは、関係の属性の意味のあいまいさがある。例えば、関係R（MANAGER－NAME，SALARY，SECRETARY－NAME）のSALARYは、管理者のものか秘書のものかを決定することは困難である。何故なら、MANAGER－NAMEとSALARYの連想と、SECRETARY－NAMEとSALARYの連想は同一の形をしているからである。しかし、２項関係モデルでは、図２－８７に示すようにSALARYが管理者のものであることを容易に明らかに出来る。

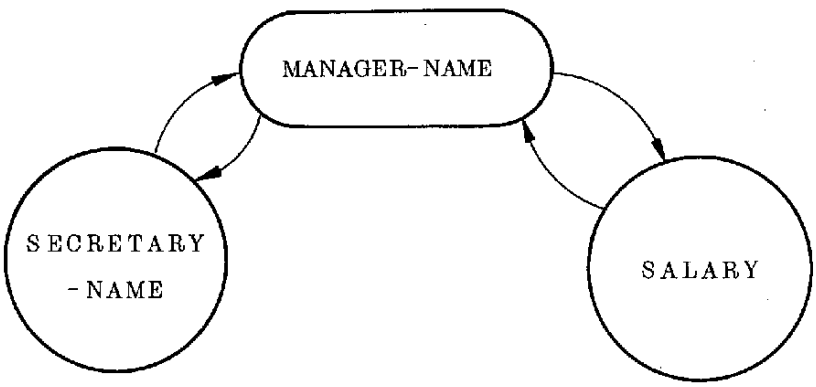


図２－８７　２項関係表現

DIAM Ⅱは、事象集合モデルに基づいてユーザへ非手続的高水準なデータアクセス言語－FORAL－を提供している〔SENK７６ｂ，７５ｂ〕。ユーザがFORALを使用するためには、システム内でどの名前と役目が有効であるかを知る必要がある。DIAM Ⅱでは、図２－８８に示すような図式をユーザは利用出来る。図において事象集合名は円内に示されている。矢印の対は、ある意味を持った事実の型を示している。DEPT－NOとEMP－MNOとの間の連想のように、事象間に複数の連想があることも示せる。２項関係モデルは、事象

間の種々の関係性（１：ｎ，ｎ：ｍ）を表すことが出来る。

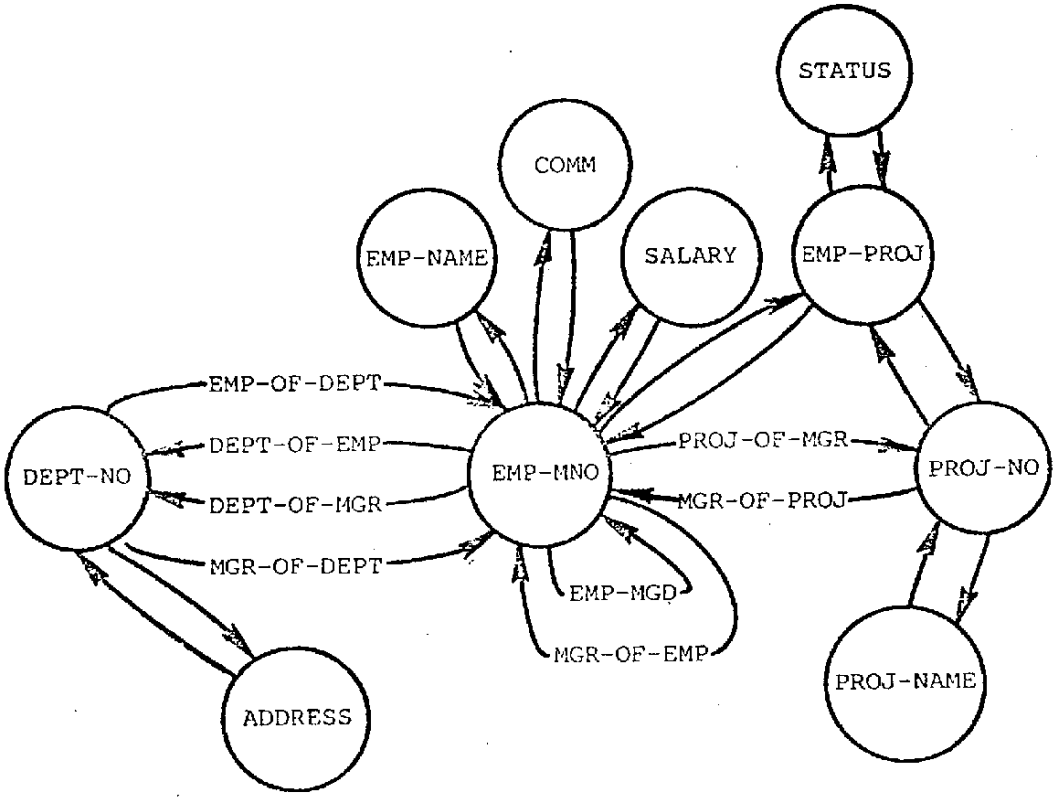


図２－８８　ＤＩＡＭ Ⅱ－ＦＯＲＡＬ型図式

ＦＯＲＡＬは、ユーザのための問合せ言語である。データベース・システム内のトランザクションの大部分は、事象の単一クラス上での情報の入力又は出力と見なすことが可能にみえる（例えば、ＤＥＰＡＲＴＭＥＮＴ内のレポート作成）〔ＳＥＮＫ７６ｂ〕。この情報には、クラス内の個々の事象名と事象の役目の値がはいっている。例えば、"上司よりも給料が多い社員のリストをつくれ"というＦＯＲＡＬ問合せは次のようになる。

```
OUTPUT⇐EMP−NO (WHERE  SALARY  GT
                ( SALARY  OF  MGR  OF  EMP−NO ) )
```

C. 事象・関係性モデル

Coddによるｎ項関係モデルにおける問題として、個々の事象（entity）を独立して扱えないことを指摘出来る〔ＨＡＬＬ７６，ＭＯＵＬ７６，ＳＥＮＫ７６ａ〕。Abrial，Senko等による２項関係に基づいた事象集合モデルでは、個々の事象の取扱いを強調している。他の試み〔ＨＡＬＬ７６，ＢＩＬＬ７６，ＦＡＬＫ７６，ＳＣＨＭＩＤ７６〕は、ｎ項関係モデルへ事象の概念を導入しようとするものである。これらのモデルにおける関係性は、簡約不能関係（irreducible relation）〔ＨＡＬＬ７６〕と呼ばれ、構文的にはFaginの４ＮＦ関係〔ＦＡＧＩ７７ａ〕へ等

価なものである。簡約不能関係は意味論的には各々の組（tuple）が実世界の基本事実を表わすようにより小さな次数の関係へ縮退しようとするものである。この関係は、言語学における重要な概念である深相意味にもとづいている。一方Coddの正規関係は、複雑なデータ処理概念（レコードと呼ばれる）へ対して適用された構文的な正規化手続きの結果である。このことによって、Coddの正規関係は、実世界で重要である意味を失なってしまっている。

Chen〔ＣＨＥＮ７６〕は、関係モデル、網型モデル、事象集合モデルといった既存のデータモデルを統一的に視るためのものとして事象・関係性モデルを提案（entity-relationship model）した。Chenは、先ず、データモデルにおけるデータの論理的視点の階層分けをした。この階層は、Senko〔ＳＥＮＫ７３〕等によるものの拡張である。Chenは、次の４層を考えた。

1) 第１層　我々が認識できる事象と関係性についての情報を表わす層
2) 第２層　情報構造を表わす層。情報構造とは事象と関係性とをデータによって表現するための情報の構造である。
3) 第３層　アクセス・パスに独立なデータ構造。検索スキーマ、インデックス・スキーマ等を含まないデータ構造である。
4) 第４層　アクセス・パスに依存したデータ構造

この階層分けにおいて、網型モデルは第４層へ対応する。ｎ項関係モデルは、第２と第３層へ対応する。事象集合モデルは、第１と第２層へ対応する。事象・関係性モデルは、第１と第２層へ対応するものである。

① 事象と関係性

事象・関係性モデル（entity-relationship model）は現実世界が事象（entity）と関係性（relationship）から成り立っているとみなす。事象とは実世界で明確に識別される事物である。例えば、ある人物、会社、出来事等は事象である。関係性とは、事象間の１つの連合（association）である。例えば、２つの"人物"事象間の関係性として、"父と子"がある。事象は、事象集合（entity set）へ分類される。事象集合は、他の事象集合と素である必要はない。

事象をｅ，事象集合をＥによって表現することにする。関係性集合（relationship）は、ｎ個の事象$e_1, e_2, \cdots, e_n$間の数学的関係である。即ち$R = \{(e_1, e_2, \cdots, e_n) \mid e_1 \in E_1, e_2 \in E_2, \cdots, e_n \in E\}$である。関係性集合Ｒの組$(e_1, e_2, \cdots, e_n)$を関係性という。例えば事象集合"人物"内の２つの事象の関係性として"結婚"がある。

ある関係性内の事象の役目（role）とは、この事象が関係性内で果たす機能である。例えば、関係性"結婚"における事象の役目は"夫"及び"妻"である。関係性内の事象の役目を次のように表示する：$(r_1/e_1, r_2/e_2, \cdots, r_n/e_n)$ここで$r_i$は関係性内の事象$e_i$の役目である。この様に役目を記述すると、関係性内の事象の順番は問題ではなくなる。

次に属性（attribute）、値（value）、値集合（value set）を考える。事象及び関係性に関する情報は、観測又は測定によって得られ、属性と値の対によって表わされる。例えば"３"，"赤"，"Peter"等は値である。値は、"ＦＥＥＴ"，"ＣＯＬＯＲ"，"ＮＡＭＥ"の様な値集合へクラス分けされる。

属性とは、事象集合又は関係性集合を、１つの値集合又は値集合の直積へ写像する関数である。即ち値集合をＶによって表わすと、属性は形式的に

$$f : Ei \text{ or } Ri \rightarrow Vi \text{ or } Vi_1 \times Vi_2 \times \cdots \times Vin$$ と表示できる。

図２－８９は、事象集合ＥＭＰＬＯＹＥＥ上で定義される属性を示している。属性ＡＧＥは、事象集合ＥＭＰＬＯＹＥＥを値集合ＮＯ－ＯＦ－ＹＥＡＲＳへ写像する。図２－９０は、関係性集合ＰＲＯＪＥＣＴ－ＷＯＲＫＥＲ上で定義される属性を示している。属性ＰＥＲＣＥＮＴ－ＡＧＥ－ＯＦ－ＴＩＭＥは、関係性集合ＰＲＯＪＥＣＴ－ＷＯＲＫＥＲを値集合ＰＥＲＣＥＮＴへ写像する。この属性は、ある社員があるプロジェクトへ従事している時間の割合であり、関係性集合ＰＲＯＪＥＣＴ－ＷＯＲＫＥＲ上で定義される。この属性は、事象集合ＥＭＰＬＯＹＥＥと事象集合ＰＲＯＪＥＣＴとのどちらの属性でもなく、ＥＭＰＬＯＹＥＥとＰＲＯＪＥＣＴとの両方に依存していることを意味している。関係性集合の属性概念は、データの意味の理解とデータ間の関数従属決定上重要である。事象・関係性モデルでは、他のモデル〔ＨＡＬＬ７６，ＢＩＬＬ７７〕と異なり、属性と値とを明確に区別している。事象及び関係性は、図２－９１，９２の様に表形式によっても表わせる。

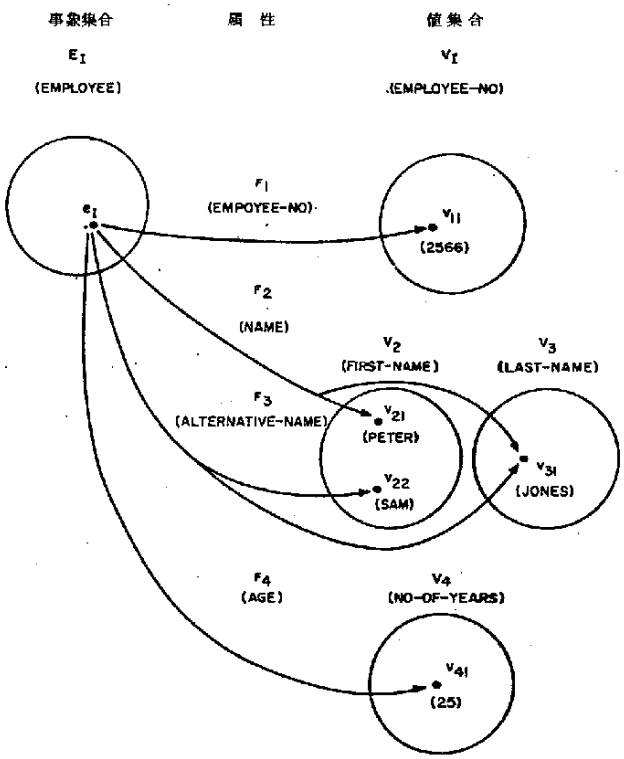


図２－８９　事象集合EMPLOYEE上で定義される属性

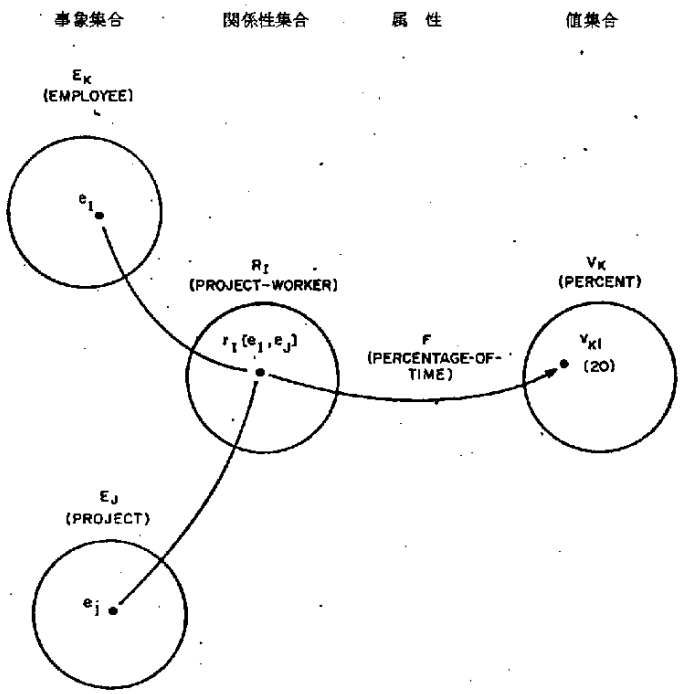


図２－９０　関係性集合PROJECT－WORKER上で定義される属性

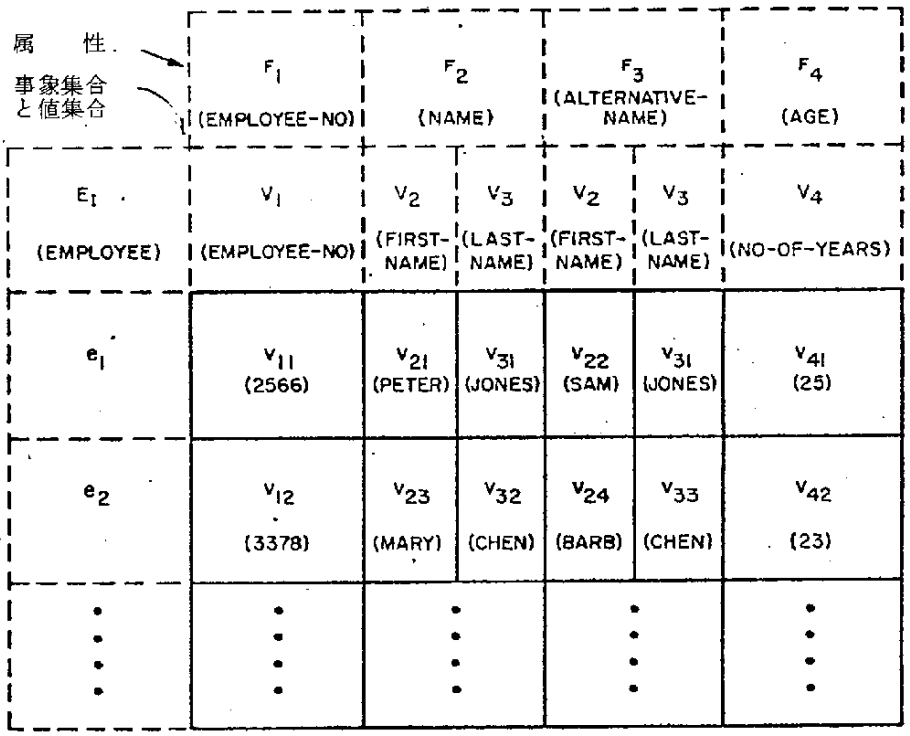
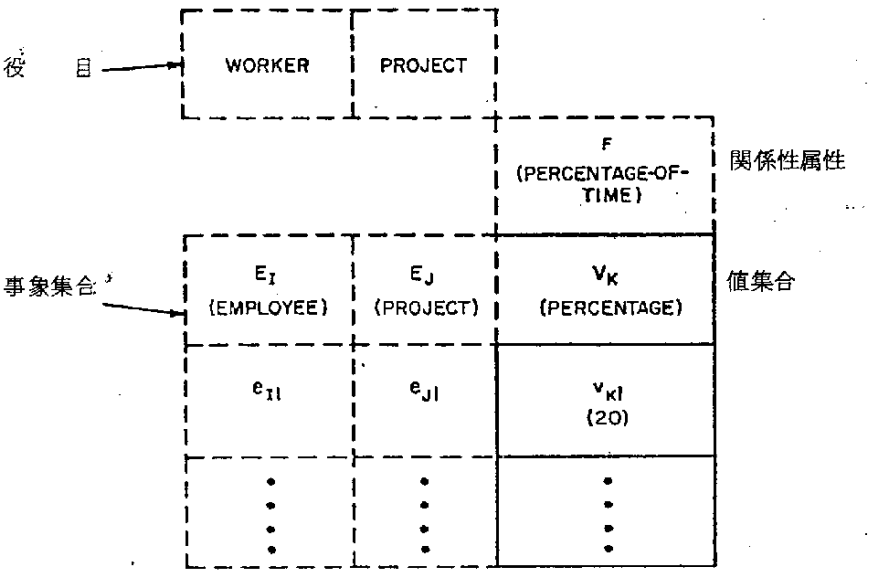


図２－９１　図２－８９へ対応した表表現



国２－９２　図２－９０へ対応した表表現

② 情報構造

今まで、事象、関係性、値、属性、役目とを議論してきたが、これらは第1層における概念的基本要素である。次にこれらの要素の記述、即ち情報構造（第2層）について考える。

事象は、属性と値との対によって表わされると述べた。よって、ある属性値は、事象を識別するために使いられる。例えば図2－89において、属性EMPLOYEE－NOの値は事象集合EMPLOYEE内の事象を識別出来る。事象キー（entity key）を、事象集合から対応する値集合の副集合への写像が1対1対応である様な属性の集合として定義する。主事象キー（primary entity key）とは、幾つかの事象キーが存在する場合、このなかで意味論的に意味ある事象キーである。関係性は、この中に含まれる事象によって識別される。よって、関係性の主キーは、関係性内の事象の主キーによって表わせる。

事象集合内の事象についての情報は図2－93に示すような表形式で表わせる。この表は、事象集合を構成する個々の事象を組（表の列）で表わし、行によって事象の属性を示している。関係性集合内の関係性情報は図2－94のように表わせる。このような表を関係性関係（relationship relation）と呼び、列を関係性組（relationship tuple）という。関係性を構成する事象は、事象関係（図2－93）の主事象キーを構成する属性によって示される。関係性関係は、関係性を構成する事象の主事象キーと、関係性属性の値から構成される。事象については、関係性における役目と、この事象の属する事象関係名とが示されている。

事象と関係性との関係には2種類ある。1つは親子の関係を表わす弱関係（weak relation）である。事象を識別するのに関係性を用いるものを弱事象関係、関係性がある関係性によって識別されるものを弱関係性関係である。例えば、社員の扶養家族は、事象関係EMPLOYEEによって識別される。この例を図2－95に示す。事象間の従属関係は、弱関係によって示すことが出来る。他は独立な関係をあらわす正規関係（regular relation）である。

主キー

属性

値集合（領域）

事象（組）

RELATIONAL VIEW

EMPLOYEE-NO	NAME		ALTERNATIVE-NAME		AGE
EMPLOYEE-NO	FIRST-NAME	LAST-NAME	FIRST-NAME	LAST-NAME	NO-OF-YEARS
2566	PETER	JONES	SAM	JONES	25
3378	MARY	CHEN	BARB	CHEN	23
⋮	⋮	⋮	⋮	⋮	⋮

事象

図2－93　正規事象関係EMPLOYEE（図1）

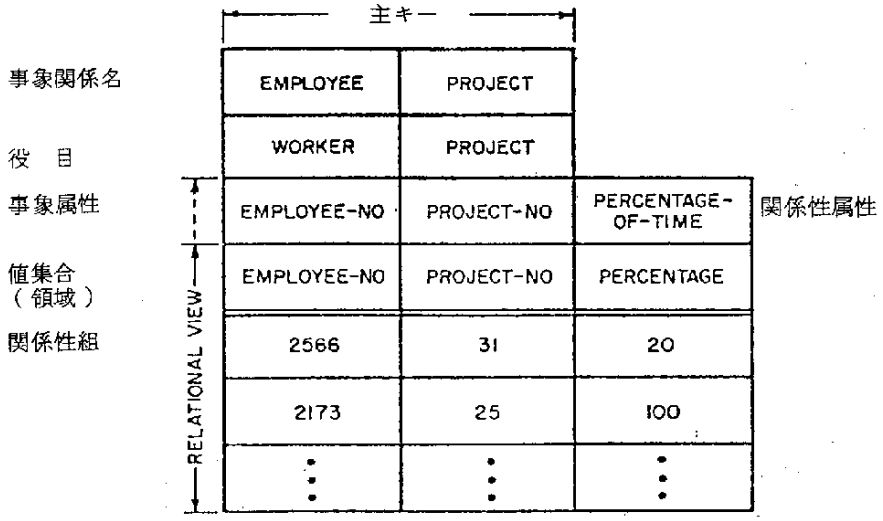


図２－９４　正規関係性集合PROJECT—WORKER（図２）

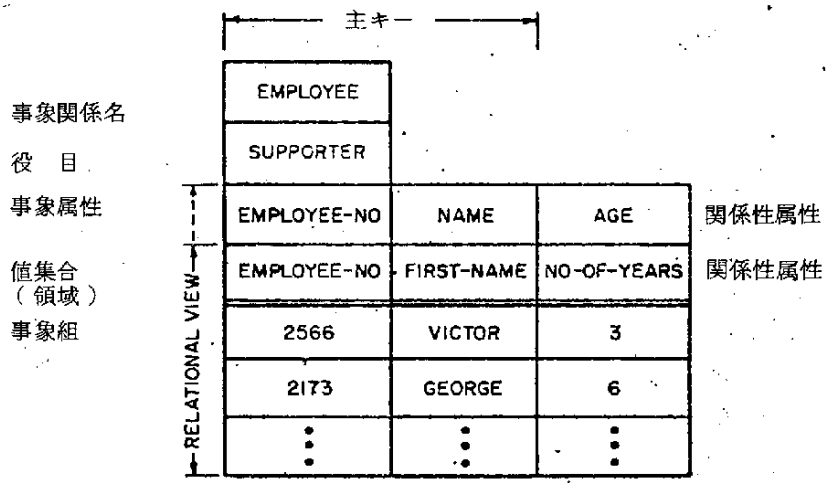


図２－９５　弱事象関係DEPENDENT

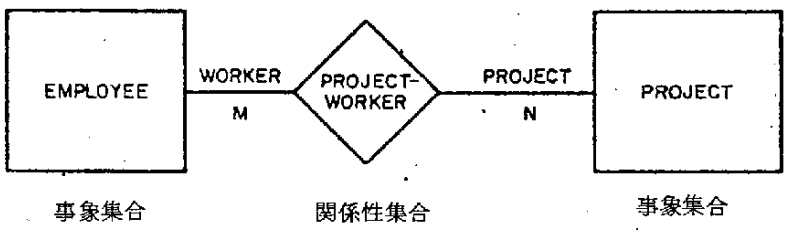


図２－９６　事象－関係性図式

事象と関係性を示すのに図２－９６の様な図式表現を使用出来る。

④　データベース設計

事象・関係性モデルを用いたデータベース設計には、次の４つの段階がある。

1)　関心のある事象集合と関係性集合を識別する。

2)　ある関係性集合が１：ｎ写像であるといった関係性内の意味情報を識別する。

3)　値集合と属性とを定義する。

4)　データを事象／関係性関係へ構成し、主キーを決定する。

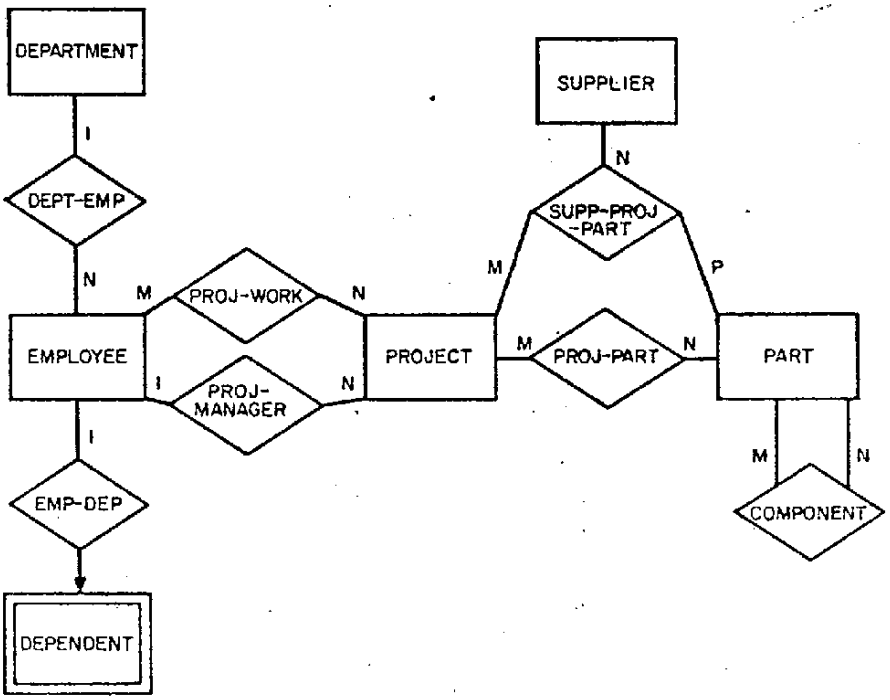


図２－９７　工場における情報の事象・関係性図式

⑤　言語

このモデルにおける問合せ言語は、集合論的記法と集合演算子とを用いる。問合せは、下記の基本演算の組み合せである。

1)　値集合から値の副集合をとり出す。

2)　事象集合から、事象の副集合を取り出す。属性値／他の事象との関係性を用いて、事象を取り出す。

3)　関係性集合から、関係性の副集合を取り出す。関係性は、属性値を述べること／関係性

内の事象を識別することによって取り出せる。

4) 属性の副集合を取り出す。

例えば、"体重が70Kg以上で、プロジェクト番号254のプロジェクトへ参加している社員の年齢をもとめよ"という問合せは、次のように表わせる。

$$\{AGE(e) \mid e \in EMPLOYEE, WEIGHT(e) > 70, \\ [e, e_j] \in PROJECT\text{-}WORKER, e_j \in PROJECT, \\ PROJECT\text{-}NO(e) = 254\}$$

第2層では、第1層における事象と関係性は事象キーと関係性キーによって表わされる。

$$\{AGE(EMPLOYEE) \mid WEIGT(EMPLOYEE) > 70, \\ (WORKER/EMPLOYEE.PK, PROJECT/PROJECT.PK) \\ \in \{PROJECT\text{-}WORKER.PK\}, \\ PROJECT\text{-}NO(PROJECT.PK) = 254\}$$

事象、関係性モデルは、事象集合モデルに対して、値と事象とを明確に分離している点である。事象集合モデルでは、全ては事象として扱かわれている。事象集合モデルでは、色を表わすのは事象COLOR/BLUEであるが、事象・関係性モデルでは値"blue"である。事象集合モデルにおいては2項関係性だけが許されるが、事象・関係性モデルでは、2項及びn項関係性が許される。

表2－8に、事象・関係性モデル、関係モデル、事象集合モデルの3つの概念の比較を行なう。

表2－8　事象・関係性、関係モデル、事象集合モデルの比較

事象・関係性モデル	関係モデル	事象集合モデル
事　象　e		事　象
事象集合　Ei		事象集合
関係性 $(e_1, e_2, \cdots, e_n)$ *)2項又はn－項	n項関係	2項関係
関係性集合　Ri		
属　性 f:Ri or Ri→Vi or $Vi_1 \times Vi_2 \times \cdots \times Vin$	役目又は属性	役目名
値	値	事象名
値　集　合	領　域	事象名集合
役　目 (事象が関係性内で行なう機能)		

D. 対象・役目モデル（object-role model）

対象・役目モデルは、Nijssen等〔ＮＩＪＳ７７ｂ，ｃ，ＦＡＬＫ７７ａ，ｂ〕によるデータモデルである。

① 情報概念

Nijssen は、データベースとは情報システムをサポートしようとし、情報とはこの様な情報システムの入力と出力であると考えている。この情報は次の３つのプロセスによって基本要素へ変換される（図２－９８）。

1) 情報を文への変換
2) 文を基本文への変換
3) 基本文を対象（object）と役目（role）への変換

これらの過程は概念化（conceptualization）〔ＳＣＨＡ７５，７６〕と呼ばれている。即ち情報とは１つ又はそれ以上の関連した文の集合である。基本文は、１つ又はそれ以上の対象－役目の対の集合である。各々の役目は、この集合内にどんなに多くても１度表われるだけである。役目は、対象タイプの基本文における役割である。概念スキーマとは、対象タイプ、役目、制約の記述である。制約とは、ある役目内のある対象型へ適用される。即ち、概念スキーマはどの情報（又は知識）が情報ベース内に入力されるか存在するかを表現している規則の集合である。又、これはデータベース内の全ての関係者の合意でもある。Falkenberg〔ＦＡＬＫ７７ａ，ｂ〕は、表相構造を深層構造への変換過程を概念化と呼び、逆の過程を非概念化（deconceptualization）と呼んでいる。

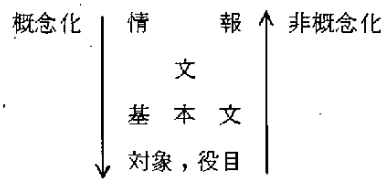


図２－９８　概念化と非概念化

② ＥＮＡＬＩＭの概念

Nijssenの提唱するモデルはＥＮＡＬＩＭ（Evolving Natural Language Information Model）と呼ばれる。このモデルは、自然言語に基づいた概念文構造（conceptual sentence structure）に基づいている。

ＥＮＡＬＩＭにおける基本概念は、次の４つである。

1) 原子対象(atomic object)

2) 役目(role)

3) 名前(name)

4) 分類(classification)

原子対象とは、ある概念世界において非可分な事象(entity)とみなされる対象である。役目とは、基本文内での原子対象の機能である。名前とは、ある文脈内の要素への一意な参照である。分類とは、要素の集合を素な副集合の集合へ分割する人為的過程である。この副集合内の全ての要素だけが、ある共通特徴を持っている。

この基本概念より導出される概念として次のものがある。

1) 原子文(atomic sentence)

2) 原子対象タイプ(atomic object type)

3) 原子対象母集団(atomic object population)

4) 原子文タイプ(atomic sentence type)

5) 原子文母集団(atomic sentence population)

6) 名前タイプ(name type)

7) 制限宣言(constraint declaration)

原子文とは、原子名と役目名とからなる対の1つ又はそれ以上の集合である。原子対象タイプはある共通特徴を持った全ての可能な対象の集合である。原子対象母集団は、ある瞬間に、同一の原子対象タイプへ属する全ての原子対象の集合である。原子文タイプは、同一の対象タイプに属し同一の役目を持つ対象ー役目対から成る全ての可能な文の集合である。原子文母集団は、ある瞬間に同一の文タイプへ属する全ての原子文の集合である。名前タイプは、ある共通特徴を持った全ての可能な名前である。

制限は、手続的又は宣言的な方法で記述される。宣言的記述方法で制限を述べることをEN-ＡＬＩＭは用いている。あるデータモデルから他のデータモデルへの変換可能性を議論する場合、識別子と関数副集合の2種の制限が必要である。

原子文タイプの識別子は、その原子文タイプのどの原子文母集団内でも一意である値を持つ対象・役目対の集合の宣言である。原子文の識別子ではないどの対象・役目対も、識別子によって定まる。識別子は、伝統的にキーと呼ばれる。

関数副集合は、ある対象・役目対の集合が対象・役目対の他の集合の副集合であることを述べている宣言である。他の制限は、データベース手続き(dbproc)である。データベース手続きは、1つ又はそれ以上の原子文がデータベースから消去又は追加される場合に必要な操作を表わすプログラムである。

次に、文が意味論的に簡約出来ない（irreducible）こと、即ち文が原子的（基本的）であることを決定する必要がある。この決定は表２－９に基づいて行なえる。４つの規則（１－４）のどれか１つに従うなら、その文は意味論的無簡約文である。

表２－９　文が意味論的に簡約可能かどうかを決定するテーブル

	規則１	規則２	規則３	規則４	他の規則
文内の役目数がn	n＝1	n＞1	n＞1	n＞1	
文内の識別子数	1	1	1	＞1	
各々の識別子内の役目数	1	n	n－1	n－1	
意味論的無簡約文（semantically irreducible sentence）	○	○	○	○	
意味論的簡約可能又は不正文					○

③　ＥＮＡＬＩＭ図式

ＥＮＡＬＩＭは、次に示すようなグラフ表現を用いて情報構造を表わす。ＥＮＡＬＩＭの図式には、タイプ図式と母集団図式（population diagram）との２種類がある。

まずタイプ図式について考える。原子対象タイプの図式を図２－９９に示す。円または楕円は原子対象タイプを表わし、この中の名前は原子対象タイプ名である。

図２－９９　原子対象タイプ（atomic object type）

図２－１００は、原子文タイプを表わしている。原子対象タイプと結合された各々の四角は、この文内で果たす対象の役目を示している。

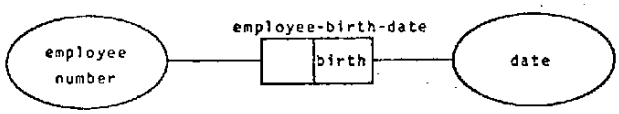


図２－１００　原子文タイプ

図２－１０１内の点線は、原子文タイプ内の識別子を構成する役目を示している。

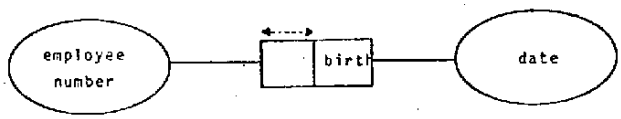


図２－１０１　識別子

２つの対象・役目対から成る原子文の識別子の構成として次の４つの場合がある（図２－１０２ａ～ｄ）。表２－９の決定規則のなかで、図２－１０２ａは規則４，図１０２ｂ,ｃは規則３，図２－１０２ｄは規則２を満足している。

図2－１０２ａ

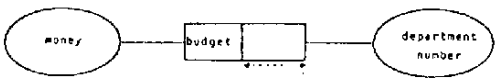


図２－１０２ｃ

図2－１０２ｂ

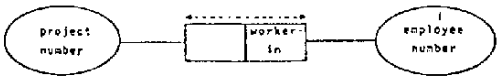


図２－１０２ｄ

図２－１０３は関数副集合を表わしている。２つの対象・役目間の点線は、副集合を表わしている。矢印のさしている対象・役目対は全体関数の値域であり、他の対象・役目対は全体関数の定義域であることを関数副集合は示している。図２－１０３の意味は次の様である：社員番号（employee number）によって参照されるある社員が、部門番号によって参照されるある部門で働らいていることを表わしている文がデータベース内へ入力されることを許されるのは、この部門の予算を表わしている文がデータベース内に存在する場合だけである。

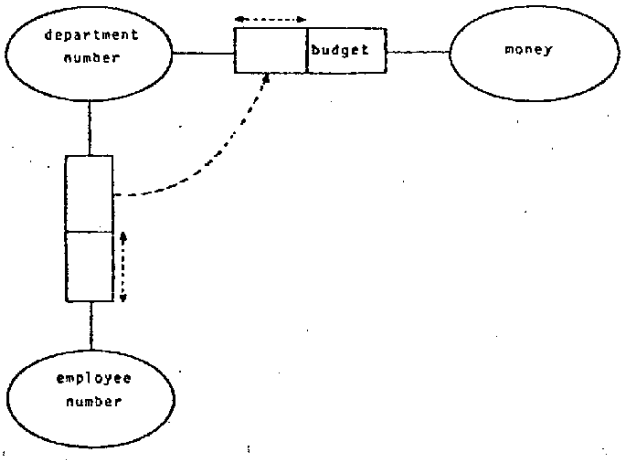


図２－１０３　関数副集合

今までタイプ図式をみてきたが、母集団図式を合わせることによってより理解しやすくなる。原子対象の母集団は、図２－１０４の図式によって示される。円又は楕円は原子対象母集団（atomic object population）を表わし、点（e_1, e_2, e_3）はこの母集団へ属する各々の原子対象を表わしている。

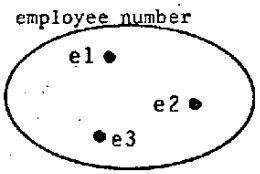


図２－１０４　同一の原子対象タイプに属する全ての原子対象の母集団

図２－１０５は、同一原子文タイプに属する全ての原子文の母集団を表わしている。各々の四角は役目を表わし直線によって結ばれたただ１つの原子対象を持つ。四角内の名前は、この

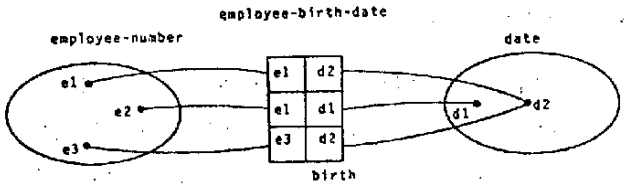


図２－１０５　原子文母集団

役目に対応する対象名が入る。共通サイトを持つこの様な役目は、１つの原子文実現値（atomic sentence instance）を表わしている。原子文母集団が複数の原子文実現値を持つならば図のように積み上げられて表わされる。

識別子と副集合は、タイプ図式と同様に表わせる。

図２－１０６に、１つの対象・役目対だけを持った文を示す。

図２－１０６

重複した対象タイプを図２－１０７に示す。

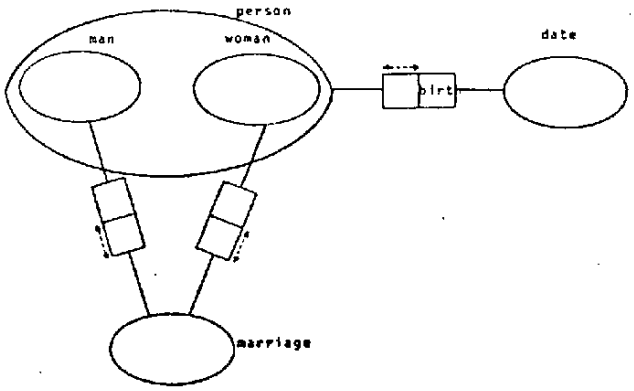


図２－１０７　重複対象タイプ

図２－１０８は、目的語文（objectified sentence）の例を示している。目的語文とは、あるユーザは原子文とみなすが、他のユーザは原子対象とみなせるような文である。ＤＢＭＳが共存する種々の視点を提供するのに必要なものである。

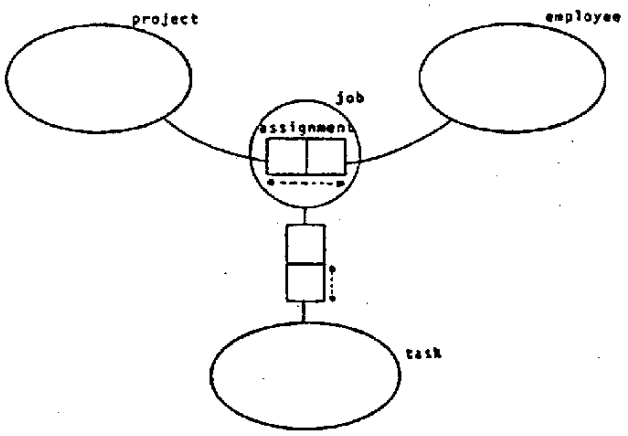


図２－１０８　目的語文

表２－１０　概念モデルの評価〔ＮＩＪＳ７７ｃ〕

概念モデルへの要求 ＼ モデル	完全性	公式化の容易性	理解の容易性	形式化の容易性	変化の容易性	変換の容易性	概念の直交性
ＥＮＡＬＩＭ	ＶＨ	ＶＨ	ＶＨ	ＶＨ	ＶＨ	ＶＨ	Ｈ
２項モデル	Ｈ	ＶＨ	Ｈ	ＶＨ	ＶＨ	Ｈ	ＶＨ
正規化関係	Ｈ	Ｍ	Ｍ	ＶＨ	Ｈ	Ｈ	Ｈ
ＣＯＤＡＳＹＬ (1973,1975)	Ｈ	ＶＬ	ＶＬ	Ｍ	Ｌ	ＶＬ	Ｌ
ＡＮＳＩ ＤＢＭＳ (1975)	Ｍ	ＶＬ	ＶＬ	ＶＬ	Ｌ	Ｌ	Ｌ

（ここで
ＶＨ＝very high，　Ｈ＝high，　Ｍ＝medium
Ｈ＝low　　ＶＬ＝very low　　である。）

Nijssen〔ＮＩＪＳ７７ｃ〕は、概念モデルとして要求される性質について、表２－１０のような評価を行なった。

E. モデルの分類へのアプローチ

トロント大学のＣＳＲＧ（Computer Systems Research Group）のKerschberg 等〔ＫＥＲＳ７６a,b〕は、データモデル相互の位置づけを明らかにするために、モデルの分類を試みた〔ＣＨＩＢ７７〕。分類は、1)構造的比較、2)論理アクセス方法、3)基本概念の比較、4)意味的比較の４点についてなされた。

構造的比較は、i)グラフ理論、ii)集合論、iii)数学的構造との３つの面について考察されている。グラフ理論的検討では、節の構造を分類し、各節や枝に各モデルのどの様な概念が対応するかを述べている（表２－１１）。集合論的検討では、集合を構成する組の次数、組の要素が集合であるかを述べている（表２－１２）。数学的構造検討では、各モデルにおける対象（object）間の関係性が、２項、ｎ項、２項＋ｎ項の３つのタイプのどれに属するかを分類している〔表２－１３〕。

論理アクセスでは、i)要素アクセス型（element-at-a-time）とii)集合アクセス型（set-at-a-time）の２つへ分類して考える。要素アクセスは通常の巡航タイプが含まれる。集合アクセス型には、1)代数型、2)写像型、3)計算型がある。

基本概念の比較では、事象（entity）、属性（property）、値（value）、関係性（relationship）

を基準概念として、各モデルのどの概念がこれに対応するかを示している（表２－１４）。

意味的比較では、データの意味をどのように定義し把握するかに関している。意味に対する言語学的アプローチに予報解析（prediction analysis）がある。あるモデルで、主文と関連叙述文をどのように表現しているかが解かれば、モデルが意味的叙述構造を情報構造にどの様に写像しているかが判る。各モデルが表現している情報構造は種々の意味的の抽象化レベルを体現している。これは表相意味から深層意味までを被う。よって、意味スペクトルを導入して各々のデータモデルを考察出来る。表相意味モデルとして、事象・関係性モデル、深層意味モデルとして、Abrial モデル、Senko モデル、Bracchi モデル及びＩＭＳ, System ２０００をあげている。

表２－１１　グラフ理論的分類

	MODEL	NODE	NODE ELEMENT	EDGE	FUNCTIONALITY
structured node elements	DBTG	Record Type	Record	(Co) Set Type	member → owner (against arrow)
	Hierarchical	Segment Type	Segment	Parent/Child Relationship	child → parent (against arrow)
	General Network	Record Type	Record	Link	none
	Information Algebra (Kobayashi)	Area	Point	Arc	none
unstructured node elements	DIAM-II	Identifier	Id. Value	Association Pair	none
	Binary Logical Associations	Set of Concepts	Concept	Logical Association	none
	Abrial's	Category	Object	Relation	none
	Functional	Set	Entity, Abstract Entity, Value	Function	domain → range (with arrow)
multiple node types	Entity-Relationship	Value Set Entity Set Relationship Set	Value Entity Relationship	Attribute Role	Attr: ES → VS Attr: RS → VS Role: RS → ES
	Semantic Network	Value Node Concept Node Event Node Characteristic Node	Value "Entity" Event Property	Characteristic Role Characteristic	ch: Char. node→Concept ch: Char. node→Event v: Char. node→Value

表２－１２　集合論的分類

MODEL	TUPLE SIZE	NESTING
Relational	arbitrary	flat
Infological	arbitrary	flat
Information Algebra	arbitrary	flat
I M E	arbitrary	nested
Extended Set Theory	arbitrary	nested
Hierarchical Relations	arbitrary	nested
DIAM-I	triplet	flat
Data Space	triplet	flat
Lindgreen's	triplet	flat

表２－１３　数学論的分類

BINARY	N-ARY	BINARY → N-ARY
Data Space	Relational	DBTG
Lindgreen's	Information Algebra (COOASYL)	Information Algebra (Kobayashi)
DIAM-I	Infological	General Network
DIAM-II		Semantic Network
Binary Logical Association		Hierarchical
Abrial's		Hierarchical Relations
Functional		Information Management Concepts
		Extended Set Theory

表２－１４　基本概念

MODEL	ENTITY	RELATIONSHIP	TYPE*	PROPERTY/APPLICABLE TO
Abrial-Data Semantics	Object	Relation.	b	————
DIAM I & II	Entity	Association	b	————
Lindgreen's	Entity	————	———	Property/Entity
Data Space	Entity	————	———	Attribute/Entity
Information Algebra (CODASYL)	Entity	————	———	Property/Entity
Information Algebra (Kobayashi)	Entity	Arc	b	Property/Entity
Benci et al.	Object	Association	n	Property/Object, Association
Entity-Relationship	Entity	Relationship	n	Attribute/ Entity, Relationship
Infological	Object	Relation	n	Property/Object
Schmid & Swenson	Indep. Object	Association	n	Characteristic/Indep. or Char. Object
Semantic Network	Concept	Event	n	Characteristic/Concept,Event, Characteristic.
Functional	Entity	Abstract Entity	n	Function/Entity or Ab. Entity

* b denotes a binary relationship
n denotes an n-ary relationship

Biller 等〔ＢＩＬＬ７７〕は、それまでに発表されている概念モデル〔ＮＩＪＳ７６ａ,etc〕の検討を行ない、現在のモデルの混乱が共通概念に対する種々の用語の存在と、モデルの水準（即ち、実世界、抽象モデル、記述言語が明確に分離されていない）であることを指摘した。この論文では、ＡＮＳＩ／ＳＰＡＲＣ提案〔ＡＮＳＩ７５〕における三層データベース・アーキテクチャに基づいたモデルのみを検討し、モデルの分類を試みてはいない。モデル間で同一の概念が、どの様に異なって使われているかに主眼を置いている。特に、種々のモデルの検討は、データモデルの意味解釈に基づいている。モデルは、実世界概念をデータベース内で表現可能な概念へ写像する抽象化過程について検討している〔図２－１０９〕。

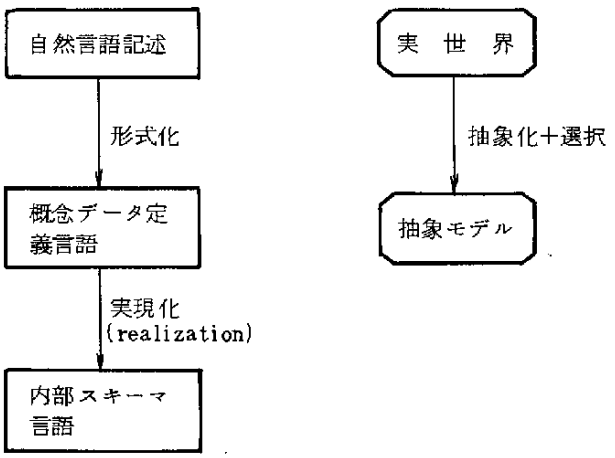


図２－１０９　実世界と対応する抽象モデル

Biller は、実世界の概念として、対象（object）、値（value）、性質（property）、関係性（relationship）、属性（attribute）、事実（fact）、時間（time）とを考えた。時間は、特殊な対象とすることにしている。これら６つの基本概念に対する各々のモデルの対応を検討した。表２－１５は、種々の論文によって導入された用語の相互対応表になっている。表中のｘは Biller における用語と同じであることを示している。表２－１６は、概念に関する分類である。

表２－１５　種々の論文内に導入された用語の相互参照

概念 / 著者	対象 (Object)	値 (Value)	性質 (Property)	関係性 (Relationship)	属性 (Attribute)	役目 (Role)	事実 (Fact)
Falkenberg	x	object	object type	association type	associatin type	x	association
Biller	x	object	x	x	relation-ship ship	x	existence of a relationship a relationship or property
Hall	entity	property value	entity set	relation	property	x	existence of a relationship
Senko	entity	property value	entity set	property	property		Triple: Entity／ Property／ Prop. Value
Adiba	x	object	category	relation	relation		
Grotenhuis	x	property	role	association	property		
Benci	entity	property value	type of entities	association	property type	x	
Moulin	entity	property value	class of entities	x	property		
Schmid	independent object	character-istic obj.	object class	association	character-istic		
Chen	entity	x	entity set	x	x	x	
Sundgren	x	x	property object group	object relation	x		elementary constellation

表２－１６　種々のモデルの分類

概念＼著者	Falken-berg	Biller	Hall	Senko	Adiba	Groten-huis	Benci	Moulin	Schmid	Chen	Sundgren
対　象 *) (Object)	a	x	x	x	x	x	x	x	c	x	x
値 (Value)			x			x	x	x		x	x
性　質 (Property)	x	x	x	x	x	x	x	x	x	x	x
関係性 **) (Relationship)	i／n	i／f	i／f	2／f	2／f	u／n	u／n	i／n	i／f	u／f	u／f
属　性 (Attribute)						x	x	x		x	x
項　目 (Role)	x	x	x				x			x	x
事　実 (Fact)	x	x	x								x
時　間 (Time)						x	x				x
基本概念数		2	2	2	2	3	3	3	2	3	3

*)　a＝原子対象だけ　　c＝複合対象

**)　i＝簡約化不能　　2＝2項　　u＝無制限n項　　n＝入れ子　　f＝平

Billerは、Hall等のモデル〔ＨＡＬＬ７６〕（簡約化不能モデル）が、いわゆる２－概念アプローチへ最も近いとしている。２－概念アプローチは、性質（対象集合）と関係性とを基本概念としている。一方、３－概念アプローチは、性質（対象集合）、属性、関係性の３つを基本概念としている。２－概念アプローチは、３－概念アプローチよりも基本概念数が少ない点でより基本的である。

更に重要な問題は、簡約不能関係とｎ－項関係との相違である。この相違は、対象指向とレコード指向アプローチとの相違である。ｎ－項関係モデルに見られるレコード指向アプローチは、実世界の不自然な意味解釈（第２章関係モデルの項を参照）をもたらし、概念モデルとしては適さない。

Billerは、次の点を結論として指摘している。

1)　これまでの種々の概念モデルが提案されているが、これらは同一の概念に対して異なった用語を使用していた。

2)　２－概念アプローチ（性質と関係性）と、３－概念アプローチ（性質、関係性、属性）との

2つのアプローチがある。

3) 概念スキーマ言語について論じられていない。

4) 概念スキーマの静的側面についての議論はやめるべきである。人間の実在物に対する概念を形式化出来ない限り、どの概念集合が最適であるかは決定出来ない。

5) 今後、概念スキーマの動的側面が研究されるべきである。動的側面とは、データ操作に関するものである。この面は、実際のデータベース構築上重要である。

3. 分散型データベースシステム

本章では、分散型データベースを実例について検討する。

3.1 概　　要

ARPAネットワーク、CYCLADES、EURONET等のコンピュータネットワークの発展は、種々のリソースの共有を可能としてきている。この様なリソースのなかで、データの重要性は、第1部において指摘した通りである。既存の各サイトの種々のデータベースをコンピュータネットワークによって結合し、ネットワーク全体で統一的利用を可能とさせる可能とさせる分散型データベースが重要な問題として登場してきている。分散型データベース構築のアプローチとしては、既存の各種データベースからネットワーク全体で統一的なスキーマを構成しようとする統合アプローチと、逆に統一的スキーマを設定し、このスキーマに基づくデータベースを各サイトへ分割する分割アプローチとの2つがある〔KUNI77〕。データベースのコンピュータネットワークへの最適配置問題は、〔CASE72、73、LEVI75、MORG77、CHU73〕によって扱かわれている。実用化を目指す多くのシステムは、前者のアプローチを用いている。

既に、いくつかの分散型データベースと呼ばれるシステムが開発又は実用化されてきている。実用化されたシステムの成功例は、MITのCONIT〔MARC77〕であろう。CONITよりも大規模なEURONETも1978年12月〔CEC77a〕にはサービス開始されようとしている。これらのシステムの特徴は、ある限定されたアプリケーション（ここでは情報検索）に対するデータベースの統合であることである。このため各データベースはファイルシステムに基づいているものが大半である。

これに対して、最近開発されてきたINGRES、SDD−1、LADDER、POLYPHEMEは汎用のアプリケーションを目指している。このため各データベースはDBMSである。ユーザに対して、関係モデル、2項関係モデルといった高度なデータモデルと非手続的言語とを提供している。

分散型データベースシステムの検討を、上記の2種に分けて行なう。先ず、CONIT、EURONETを3.2節で論じる。次に、INGRES、SDD−1、LADDER、POLYPHEMEを、3.3節において論じる。

3.2 実用システム

本節では、分散型データベースシステムを実用レベルで構築している例を紹介する。紹介する二システムは、MITのMULTICSシステム上にインプリメントしたCONITシステムと、EURONETの例である。この二システムには多くの共通点をみることができる。第一に、既存の文献検索システムの統合化を意図していること。第二に、統合化に際しては共通コマンド言語を設定し検索はローカルで処理させていること。第三に、データベースの数が全体で三ケタに達し大規模であること。

両システムとも現実には、実験あるいは研究の域を脱していないが、利用者にとっては、早期の開発が望まれるところである。MITのCONITシステムについては、既に四つの文献検索システムが共通コマンド言語により検索できるレベルまで達しており実験の段階においては成功したと言えよう。一方、EURONETは1978年12月を実施のめどとしており、EC諸国を中心として、国家レベルでのデータ共有や、参加が予定されている27機関のデータベースを総計すると約100に達するなど特筆すべき事項は多い。

以下、CONITシステムとEURONETにおけるデータベースの共有システムについて紹介する。

3.2.1 CONITシステム〔MEDI77, MARC75, 76, 77〕

A. 概　要

現在米国においては、Lockheed社のDIALOGシステムやSDC社のORBITシステムに代表されるように多くの情報検索システムが開発・運用されている。また、これらのシステムが提供するデータベースは、物理学、化学、電気・電子工学、航空宇宙工学、生物学、医療、経営・経済、歴史、時事ニュースなど極めて多岐にわたっており、ユーザの広範なニーズに応えている。各システムの検索機能に着目すると共通要素が多いことがわかる。しかし、コマンド言語に関しては名称やオプション機能などかなりの違いが見られる。コマンド言語の異なるシステムが存在すること自体が、ユーザのシステムに対する操作性向上の障害の一つになっているとも言えよう。CONIT（Connector for Networked Information Transfer）システムは、このようなシステムにより異なる手続上の繁雑さを低減することを目的として、MITのMULTICS上に開発されたものである。CONITシステムは、既存のコンピュータネットワークやホストコンピュータを有効に利用して、各システム間に共通インタフェースを設けることにより、上記の問題を解決する一つのアプローチを実現している。現在CONITシステムを通して検索が可能なシステムは、Lockheed社のDIALOG、SDC社のORBIT、NLMのMEDLINE、SUNYのMEDLINEの四システムである〔MARC77〕。CONITシステムでは、この四システムの統一的アクセスを実現するために共通インタフェースを設け、仮想システムの概念を確立している。

以下、四システムの概要、CONITシステムの基本概念、仮想システムの考え方、CONITシステムの構成、共通コマンド言語、検索例、今後の問題点などについて解説を行なう。

B. CONITシステムの実験インタフェースと参加システムの概要

現在、COINTシステムを通してアクセスできる文献検索システムは四つ〔MARC77〕であり、今後も増える傾向にあると思われる。しかし、これらのシステムがデータベースの数やユーザ数など規模的、あるいは知名度などの点からみても代表的なシステムであることから、初期の目的は充分達成していると言えよう。

現在CONITシステムは、共通インタフェースを、MITのMULTICS上に設定しており、ユーザは、ARPANETやTYMNET、Telenetを通して、四つの情報検索システムのどこへでもアクセスが可能となっている。

表３－１　CONITシステムに参加している四システム

システム名	運用開始年	データベース数	電子計算機	運用機関・所在地
DIALOG	1965	51	IBM 360／65	Lockheed Missiles and Space Company, Palo Alto, California
ORBIT	1965	36	IBM 370／158	SDC; System Development Corporation, Santa Monica, California
NLM MEDLINE	1964	11	IBM 370／158	NLM; National Library of Medicine, Bethesda, Maryland
SUNY MEDLINE	1968	3	IBM 370／158	SUNY; State University of New York, Albany, New York

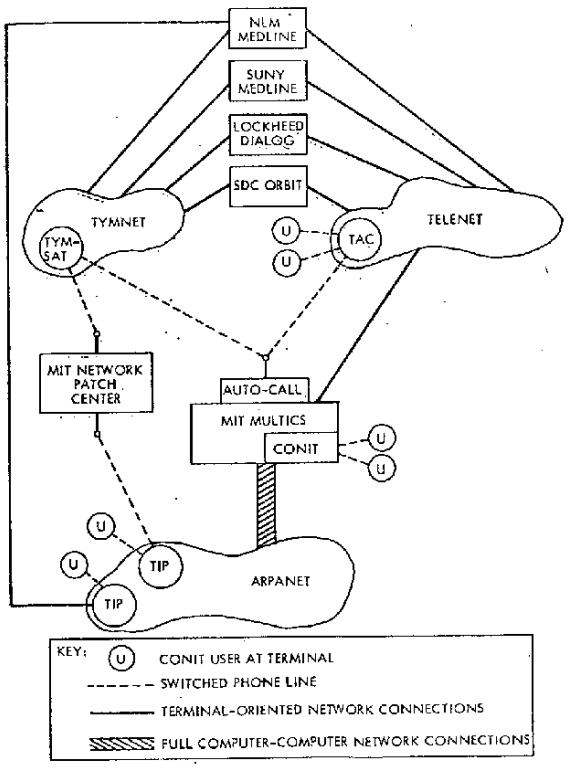


図３－１　Computer and Network Interconnections for CONIT Experimental Interfaces〔MARC77〕

CONITシステムのユーザは、“pick system”と指定するだけで、希望する情報検索システムとの初期手続が完了したことになる。つまりこのpickコマンドにより、システムとの接続、ログ・オンなどの手続きはCONITにより自動的に行なわれる。

接続が完了した時点では、検索用コマンド言語として自動的に共通コマンド言語がアサインされている。共通コマンド言語で可能な検索機能を上回る、きめの細かいそして深い検索を望むユーザにとっては、検索システム固有のコマンド言語（speak host）を指定することも可能である。次に参加四システムの概要を示す。

① DIALOGシステム〔SUMM75, LIS76〕

DIALOGシステムは、Lockheed社（Lockheed Missiles and Space Company）により開発・運用されており、SDC社のORBIT—Ⅲシステムと並ぶ世界的な二大商用オンライン文献検索システムの一つである。

1965年に開発されたこのDIALOGシステムは1972年には商用によるオンライン・サービスを開始している。現在DIALOGシステムが提供しているデータベースは51種、文献数は約1,200万以上、ユーザ数は1,800とほかの商用システムを圧倒している。また、TYMNETやTeleNET（米国、カナダのみ）などの通信網を利用することにより、米国本土は無論カナダやヨーロッパの各都市にもサービスを提供している。

米航空宇宙局（NASA：National Aeronautics and Space Administration）Informatics社、欧州のイタリアのFrascatiに本部をもつESA/SDS（欧州宇宙機構：European Space Agency／Space Documentation Service）でそれぞれ稼働している、通称RECON（REmote CONsole）と呼ばれているシステムは、Lockheed のDIALOGシステムをベースに開発されたものである。

② ORBITシステム〔HUMP74, CUAD75〕

ORBIT（On-line Retrieval of Bibliographic Information Time - Shared）システムはSDC社（System Development Corporation）により1965年に開発され、1973年には商用のオンラインサービスの運用を開始している。現在のORBIT—Ⅲが提供しているデータベースは36種、文献数は約600万件以上、ユーザ数も900を超え規模的にはDIALOGシステムに次いでいる。また、DIALOG システムと同様に通信網TYMNETを利用して、米国本土は無論カナダやヨーロッパの各都市へもサービスを提供している。

ORBIT—Ⅲシステムの特徴の一つとして、その運用形態をあげることができる。ORBIT—Ⅲシステムではいわゆる時間割方式（time window）を採用し、全データベースのうち数個を除いた残りのデータベースを四つの時間帯に分けてサービスを行なっている。DIALOGシステムがサービス時間帯には全データベースのサービスしているのと大きな違いを示している。次に紹介するNLMのMEDLARS／MEDINEシステムのソフトウェアELHILL—Ⅲは、ORBIT—Ⅲシステムをベースに開発されたものである。

③ NLM／MEDLINEシステム〔HUMP74，NIH76〕

MEDLARS（医学文献分析検索システム：MEDical Literature Analysis and Retrieval System）システムは、NLM（米国立医学図書館：National Library of Medicine）において1964年に稼働を開始している。現在のMEDLARS－Ⅱ システムは、1967年に実験を開始し1971年から本格的な運用を開始したものでMEDLINE（MEDLARS on－LINE）と呼ばれている。現在米国内ではバッチ方式による検索サービスは一切中止され、すべてオンラインシステムに移行している。このシステムが移行する際には種々のソフトウェアの検討がなされたが、最終的にSDC社のORBITを採用したという経緯がある。

MEDLINEシステムを通して利用できるデータベースは、NLM自身が提供するデータベースを中心に11種類あり、1975年度末の入力文献総数は約200万件になっていると言われている。またNLMが提供するMEDLARSデータベースのうち特に最近の1ケ月分だけを別ファイルとして管理することにより、オンラインによるSDI（Selective Dissemination of Information）サービス（SDILINEと呼ぶ）が実施されている点も特徴と言えよう。なおMEDLINEのユーザ数は約500である。

Lockheed／DIALOGやSDC／ORBITの各システムと同様に、MEDLINE システムも通信網を利用することにより、米国本土は無論カナダ、南米そしてヨーロッパの各都市へサービスを提供している。しかし、スウェーデンや西ドイツでは独自のMEDLINE ネットワークを築いて周辺の各国へのサービスを行っている。さらに、日本においてもJICST（日本科学技術情報センター）によるMEDLARSデータベースのオンラインサービスが1976 年7月から開始されている。

④ SUNY／MEDLINEシステム〔HUMP74，EGEL75〕

SUNY（ニューヨーク州立大学：State University of New York）では、IBM社のオンライン情報検索システムSTAIRS（STorage And Information Retrieval System）を用いて、1973年からSUNY／MEDLINEを運用している。なお、SUNYではSUNY／BCN（SUNY Biomedical Communication Network）という名称で、生医学文献に関する最初のオンライン検索サービスを1968年から運用している。SUNY／BCNネットワークは、医学および大学図書館を対象に当初は9図書館にすぎなかったが、1973年にSUNY／MEDLINEが稼働する時点では32図書館に拡張されている。

米国においてMEDLARSデータベースをオンラインで提供している機関は、NLMのほかにはこのSUNYとBRS社（Bibliographic Retrieval Scrvive）が知られている。

これらのシステムは、何れも NLMが提供する索引誌Index Medicusをベースとしているが、データベースの管理方法は若干異なっている。NLMでは最近3年間の文献をまとめたデータベース（MEDLINE）と最近1カ月の文献をまとめたデータベース（SDILINE）を提供しているのに対し、SUNYでは最近4年間の文献を年別にしたデータベース（im77，

im76，im75，im74）と最近1カ月の文献をまとめたデータベース（imca)を提供している。なお、SUNYではMEDLARSの他にERIC（教育）Psychological Abstracts（心理学）の計三つのデータベースが利用できる。

C．CONITシステムの基本概念（仮想システム）

異なる情報検索システムのインタフェースに係る問題の解決策の1つとして、種々の端末から多くのシステムにアクセスが可能な便利で経済的なコミュニケーション・ネットワークの充実が挙げられる。

最終的にはシステムやデータベース構造の標準化にあると考えられるが、標準化を達成するには既存システムの特徴や環境、ユーザの好み、異なるファイル構造をもつデータベースの再編成など余りにも多くの問題をかかえておりすぐに実現されるとは思えない。したがって、それに至る現実的解決策として考えられるのがコンピュータによるインタフェース（変換）技術である。仮に異なる情報検索システムがn個存在するとき、それぞれのシステム間（1対）に変換アルゴリズムを設けるとn（n－1）個必要となりnが大きい場合には効果的でない。

そこで、CONITシステムにおいては、共通インタフェースを導入することにより2n個の変換アルゴリズムを設けるだけでシステム間のインタフェースを達成している。つまり、共通言語でアクセスした内容はコマンドプロトコルに従って目的システムのホスト言語に変換され、目的システムからの応答はレスポンスプロトコルに従って共通言語に変換される（図3－2）。現在

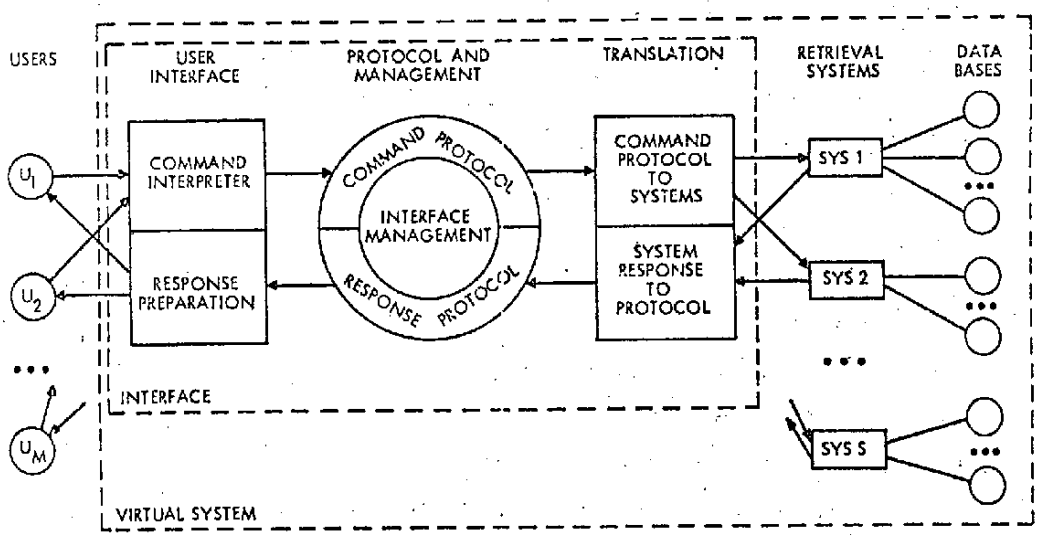


図3－2　LOGICAL DIAGRAM OF VIRTUAL-SYSTEM INTERFACE〔MARC77〕

のCNITシステムはn＝4であるから8個の変換アルゴリズムですむことになる。

CONITの仮想システムは、コマンドプロトコルとレスポンスプロトコルを中心にユーザインタフェース、トランスレーション、そしてそれらをとり巻く各種情報検索システムとそれぞれの

データベースから構成されていると考えることができる。

また、より効果的な仮想情報検索システムを構築するための構成要素としては、次の三要素が挙げられる。

・共通コマンド言語

・インデックス語い間の変換

・共通書誌データ構造

これらの構成要素について以下に記す。

① 共通コマンド言語

共通コマンド言語は、ユーザにとって利用しやすくしかも情報検索としてはすべての基本的機能をもったものでなければならない。そのような共通コマンド言語を開発するために、各システムが持っている言語の機能を最小単位に分解し、言語間の違いを見出し、共通機能の組み合わせとして、共通コマンド言語の開発を行なった。また、共通コマンド言語の開発に際しては、オープン・エンド性（open-endedness）、柔軟性、モジュラリティが、今日の情報検索システムの動的環境においては充分配慮される必要のある事項であると言われている。

② インデックス語い間の変換

CONITシステムでは、インデックス語いの仲介言語としては自然英語（natural English）が優れていると判断している。語いの変要にはMIT機構（Master Index and Thesaurus）が用いられており、そのMITにはそれぞれのデータベースのインデックスを始め全インデックスのＡＢＣ順リストおよび文献数、シソーラスの関係が含まれている。さらに、フレーズ分解（フレーズから単語への展開）と語幹抽出を行なうことにより、ほとんどの語い間の関係を自動的に識別できることになっている。

③ 共通書誌データ構造

共通コマンド言語が基本的な機能から構成されているのと同様に共通書誌データ構造も、基本的なデータ要素から成り立つと考えられる。いかなるシステムのデータ要素も共通書誌データ構造における基本的なデータ要素の組み合わせから成っている。したがって、これらの基本的なデータ要素を階層的に組み立てれば、ほとんどのデータベースとの対応は可能となる。

D．CONITシステムの変換テーブル

CONITシステムの変換テーブルには、コマンドプロトコルとレスポンスプロトコルに対応したコマンド変換表（Command Translation Table）とレスポンス変換表（Response Translation Table）の二つがある。この変換テーブルは、個々の情報検索システムとCONITシステムとのメッセージ変換テーブルと考えられよう。図３－３にその一部を示す。

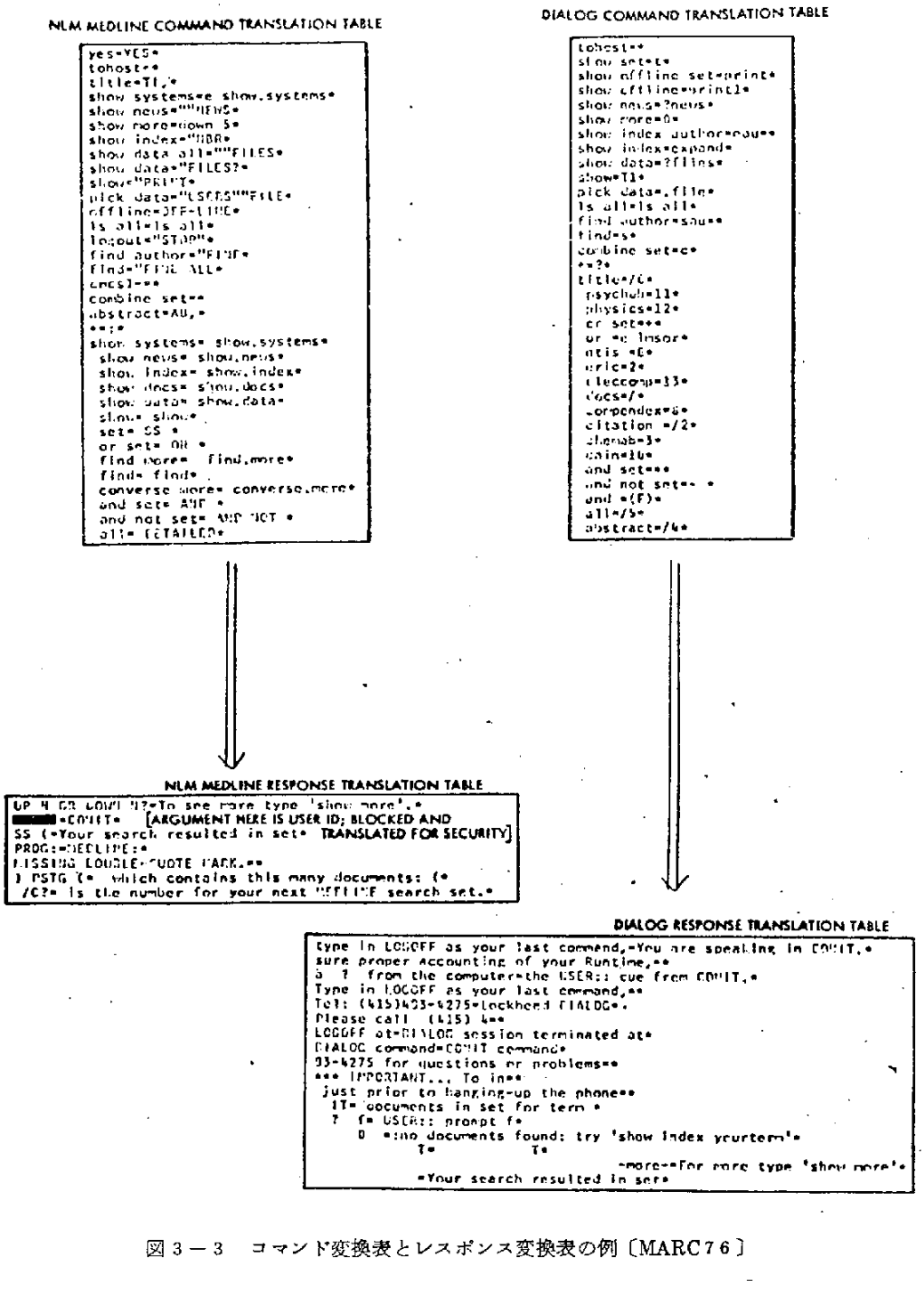


図３－３　コマンド変換表とレスポンス変換表の例〔MARC76〕

E．CONITシステムの検索コマンドと検索例

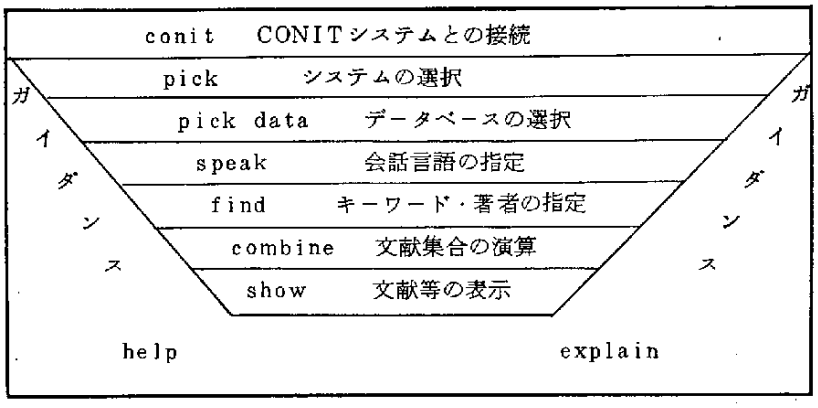


図３－４　CONITシステムの主な検索コマンドとその機能

CONITシステムの主な検索コマンドを機能別に示したのが図３－４である。

システムやデータベースの選択から、キーワードや著者による検索、検索結果の集合演算および表示、システムや個々のコマンドの機能についてのガイダンス機能まで網らされており、情報検索システムとして一応の機能を有していると判断できよう。

表３－２にCONITシステムの検索コマンドの概要を示す〔MEDI 77, MARC 77〕。また、将来の拡張計画の一部を表３－３に示す。CONITシステムでは、現在のコマンド体系に満足することなく種々の拡張計画をもっているが、ここではサーチ関係に焦点をあてその概要を示すことにより、将来のCONITシステムの一端を把握することにする。

ここに述べた機能を反映させた検索言語がCONITシステムにとって最終的なものではなく、また完全なものとは言い切れないので、今後、さらに多くの機能を必要とし充分なテストの積み重ねを必要とするであろうと言われている〔MARC 76〕。

CONITシステムの検索コマンドについて、現在の体系と拡張計画について概要を示したが、これらの機能のすべてが実際にインプリメントされたときのCONITシステムは、いわゆる情報検索システムとしても充分満足のいくものになるであろう。したがって、今後、検索コマンドに望まれる機能は、システム間の単なる変換機能ではなく複数のシステムへアクセスできる特長を最大限に利用した独特のものが要求される。そのような新しい機能としては、たとえば、次のような機能が考えられる。

① レファレルサービス的機能（ユーザにとって適切なシステムあるいはデータベースの教示）

guide system message
guide data message

message ：希望する検索テーマ指定

② ネットワークワイドな検索機能（複数データベース、複数システムに対する一斉検索）

表３－２　CONITシステムの検索コマンド〔MEDI 77〕

	コマンド名	省略形	人力形式	機能
①	conit	—	conit	CONITシステム起動用
②	help	—	help	CONITシステムの利用方法について、基本的な機能を把握するためにexplainコマンドの使用を指示する。
③	explain	e		コマンドやコンセプトを指定することにより簡単なガイダンスが得られる。
			explain concepts	CONITシステムにおけるconceptの種類と個々のconceptについて簡単な説明。(concept ; conit, start, commands, converse)
			explain conit	CONITシステムの概要説明
			explain start	実際に検索を開始するに当たって、必要な手続き（システムとデータベースの選択）について指示。
			explain commands	CONITシステムのコマンド・リストとその省略形、簡単な説明文を表示。 (commands;exlain, pick, find, show, combine, speak)
			explain converse〔more〕	CONITシステムにおいて、ユーザとCONITシステムとのコマンドのやりとりがどのように行なわれるかを説明。"more"を指定すれば詳細な説明が得られる。
			explain data x	DIALOGシステムの場合、指定したデータベースの説明が行なわれる。
			explain data	ORBITシステムの場合、接続中のデータベースの説明が行なわれる。
④	pick	p		CONITシステムを通して利用するシステムとデータベースの選択。
			pick ×〔Y〕	システムの選択（×；lms, sdc, medline, suny） ネットワークの選択（Y；tymnet, telenet）

	コマンド名	省略形	入力形式	機能
④	pick		pick data ×	データベースの選択(×; sdiline, ntis, …)
⑤	find	—	find ×	指定したキーワードをもつ文献を検索。(×; transportation, skin, …)
			find × +	指定した文字ストリングをもつキーワードを含む文献を検索。(×; radiation+, …)
			find author ×	指定した著者名による文献を検索。(×; marcus, …) findコマンドについてより詳細解説を要求するときは、"explain find more"。
⑥	show	—		CONITシステムに参加しているシステムやそれぞれのデータベース名、キーワードリスト、検索結果などを表示する。
			show systems	現在利用し得るシステム名を表示。
			show data [all]	現在利用し得るデータベース名を表示。"all"を指定すると、個々のデータベースの収録文献数とそ及範囲を表示。
			show index ×	指定したキーワード×に対して、アルファベット順リストを×の前後に渡って表示する。継続してキーワードリストを見たいときは"show more"。
			show news	接続中のシステムからニュースが得られる。
			show [offline] [setn] {all / title / abstract / citation} [docs j-k]	検索した文献の内容を表示する。offlineを指定するとセンタへオフライン出力。省略時にはユーザ端末へオンライン出力が行なわれる。指定した文献集合nについて、全項目、タイトルあるいは抄録のみを集合内の文献j番目からk番目まで表示。1件だけ表示する場合はj=k。省略時は5件表示。詳しい説明は、"explain show docs"。
⑦	comine	—	combine set j {and / or / not} set k	文献集合同志の論理2項演算を行なう。 set jとset kの論理積。 set jとset kの論理和。 set jとset kの論理否定。(差)
⑧	speak	sp		CONITシステムと接続した時点では共通言語が設定されている。

	コマンド名	省略形	入力形式	機能
⑧	speak	sp	speak conit	共通言語としてconit使用を宣言。
			speak terse speak verbose	簡潔なメッセージ・モード(通称short)に切り換え。 詳しいメッセージ・モード(通称long)に切り換え。 共通言語に対して有効。
			speak host	検索システムを提供している個々の検索語(host language)を指定。
			speak monitor speak user speak no screen speak screen	CONITシステムと各情報検索システムとの間でやりとりされる全てのメッセージを表示。 monitorモードからuserモードへの切り換え。 使用している検索システムの決まり切ったフォーマッティング文字の省略。 no screenモードからscreenモードへの切り換え。
⑨	name－file (open)	nf	name－file filename	検索結果を保有するためのコモンファイルの創成およびリオープン。 異なるデータベースからの検索結果の保存も可能。
⑩	file(save)	—	file	検索プロセスにおいて次に発生する検索結果を保有する。
⑪	view	—	view filename 〔view〕 lines j－k	指定したファイルに登録されているメッセージの行数を表示。 j行目からk行目まで表示。異なるデータベースをアクセスしている場合でも表示可能。
⑫	set table	st	set table 〔out〕 tablename	コマンド変換テーブルのセット。"OUT"を指定すると、レスポンス変換テーブルをセットする。
⑬	replace	rep	replace 〔out〕 $matchstring =replacement- string〔|〕	テーブルの内容を変更する。"OUT"を指定するレスポンス変換テーブルに対して変更を行なう。最後の〔|〕はスペースを付加するときに用いる。
⑭	list table	lt	list table 〔out〕	現在の変更テーブルの内容を表示する。"OUT"を指定するとレスポンステーブルの内容が表示される。
⑮	list status	ls	list status { all system language mode tip patch }	全項目(下記の5項目)について、CONITシステムの状況説明を行なう。 現在選択しているシステムと他に移動中のシステム名の表示。 現在用いられている言語。 現在の言語モード。verboseかterse, screenかno－sereen, monitorかuser。 現在用いられているTIPとportの表示。 現在patchに接続しているシステム名称。

表３－３　CONITシステムの検索コマンド（拡張計画）

	拡張機能	入力指定例	解説
①	データベースの選択とキーワードの指定の組み合わせ	find 〔pick〕 〔data〕 ntis radiation effects radiation effects → キーワード ntis → データベース名 〔pick〕 → データベースの選択 find → 検索コマンド	この形式を採用することにより、異なるデータベースを対象に検索を行なう場合のデータベース切り換え手続きが簡略化される。
②	複数の検索フィールドの指定可能	find title descriptors neutron scatting neutron scatting → キーワード descriptors → ディスクリプタ項目 title → タイトル項目 find → 検索コマンド	
③	ブール代数による二項目演算の多項式化	find A and B or C and not D find A and (B or (C and not D))	現在２つの文献集合に対してのみ可能な演算機能を、キーワード指定方式を採用し、しかも多項式による指定も可能とする。
④	キーワード中の文字の任意指定	find on ? line : system : l#ngu ＃　任意の文字数指定（二文字の場合は　＃＃） ：　任意の特殊文字数 ？　１つの特殊文字	
⑤	キーワード間の距離の指定	find A within〔exactly〕＋n units B n　AからBへのワード数等 ＋　右方向の指定（Aの後方） －　左方向の指定（Aの前方） exactly特殊文字を指定した場合のみ指定可能 units　character, words, lines, sentence ……	④と⑤より“a＃＃b”と“a within exactly＋3 characters b”が同義になる。
⑥	不要語（コモン・ワード、ストップ・ワード）や語尾変化の自動処理		

	拡張機能	入力指定例	解説
		find economics of computer communications "communicat" "comput" 削除 "econom" 検索コマンド	上記の例では、次の諸手続きが行なわれる。 ・語順の無視　・不要語の削除 ・語幹の抽出　・AND演算の実施
⑦	リスト類の通覧機能充実	show type vocab data 〔n line〕term term 表示するもので語、句、文字列の指定可能 type index アルファベット順リストの表示 phrase 指定したtermを含む句の表示 thesaurus シソーラスの表示 relations 上記の項目すべてについて表示 vocab 特別なシソーラスなどの指定（MeSH, NASA thesaurus, ……） data データベースの指定（MEDLINE, NTIS, ……） n 表示する行数を指定 次のような指定が可能となる。 show thesaurus synonyms term	
⑧	文献集合のネーミング	find computer networks name 〔set〕 cnet キーワード 検索コマンド cnetと名付ける find computer networks ; rename set k cnet 現在の名前　新しい名前	現在検索プロセスを通して作成された文献集合に対しては、システム側で一連番号をつけている。（set1, set2, …）しかし、内容の表現に不足な点があると思われるので、ユーザ側からも名前を付けることを可能とする。
⑨	検索履歴の照会と削除	show sets 〔set i〕〔set j〕…	指定したセットについて、文献集合名、検索式、文献数を表示。

	拡張機能	入力指定例	解説
		delete { set i set j / all sets }	指定した範囲に相当する文献集合あるいはすべての文献集合を削除する。
		delete number direction set i { number 番号かall direction beforeかafter	文献集合iからnumberで指定した文献集合まで（前、後）を削除する。
⑩	文献集合の演算式の多項化	combine(set 5 or set6)and set2 and not set7	③では、キーワードによる検索式の多項化について述べたが、本項では文献集合に対する演算式の多項化について述べる。
		find energy costs and not set4 ↳既存の文献集合 ↳論理否定演算 ↳キーワードの指定 ↳検索コマンド	またキーワードと文献集合の演算も検討している。
⑪	出力項目の複数項目指定可能	show title author abstract	現在文献の出力（表示）は、全項目あるいはタイトルや著者、抄録だけの出力が可能であるが、個々の項目について複数の指定を可能とする。
		show abstract documents 3 7 { to / _ } 10 15 (docs)	次のように既存の文献集合から一部の文献に対してのみの出力を指定することも可能である。
⑫	既存の文献集合から部分的文献集合の作成	find docs j-k	
⑬	文献集合の途中からの表示可能	show [title / abstract]	[from] doc j
⑭	文献集合のソート	show order field mode { field ソートフィールド mode { forward…昇順 / reverse…降順 }	

	拡張機能	入力指定例	解説
			文献集合に対してある種の順番（年代の降順）に並べるためのコマンド
⑮	システムとユーザとの対話	help human send mode to name address message message mode 対話モード（immediate, offline）の指定 name, address: メッセージの送り先 message メッセージ	この2つのコマンドを用いることにより、ユーザとシステムの自由な対話が可能となる。
⑯	検索結果の保存と再現（コモン・ファイル機能の充実）	save set j set k	現在の保存機能は、オンライン表示をしたものについてだけに限られているが、文献集合単位での保存指定を可能とする。
		view [file] page n view certain [page] certain=last, next, previous, first…	保存されている検索結果の再現を、現在の行単位からページ単位の再現も可能とする。

find all data × 接続中のシステムの全データベースを対象に同じキーワード× で検索する。

find all × CONITシステムが提供し得る全データベース（全システム）を対象に同じキーワード×で検索する。

③ 一次情報のサポート機能

検索を通して必要となった一次情報について、ユーザが入手可能な最も近い入手先と手続き方法の教示。

④ SDIサービス機能の充実

複数システムを対象としているので、当然重複するデータベースもあるが、全体的に見れば相当の分野をカバーすることになる。処理形態は、オンラインあるいはオフラインのいずれをとるにしても、広範な分野からのSDIサービスの提供を受けることが可能になろう。

CONITシステムのまとめとして、共通言語を用いて実際に検索した例を紹介する（図3－5）。

ここで紹介する検索例は、SUNY／MEDLINE，NLM／MEDLINE，Lockheed／DIALOGの各システムを利用しているもので、この検索例からCONITシステムの機能的概要を把握することができよう。特に着目すべき点は次の項目である。

・システムの選択・切り換え

・データベースの選択・切り換え

CONITシステムとの対話内容	解　　説	コマンド
conit Welcome to CONIT, for help on how to use CONIT you may type 'help' followed by a carriage return; otherwise, you may now type any CONIT command. USER:: pick suny Attachment successful.	CONITシステム起動	conit
SUNY/MEDLINE: Connection completed. SS 1 is number for your next search set.	SUNY SESSION	pick
suny is connected successfully You are now speaking to CONIT USER:: find radiation	キーワードradiatonの入力 2684件検索	find
SUNY/MEDLINE: Your search resulted in set1 which contains this many documents: (2684) SS 2 is number for your next search set.		
USER:: show title docs1-3	文献1～3のタイトル表示	show
SUNY/MEDLINE: 1 TI - ;STATISTICAL EVALUATEOH OF LIGHT PROTECTION FACTORS; 2 TI - PROGNOSIS AND POST-THERAPEUTIC FOLLOW-UP OF BREAST CANCERS BY THERMOGRAPHY. ...	MEDLINE SESSION	pick
pick medline Attachment successful. Logon to host started. LOGON IN PROGRESS AT 14:56:52 ON JANUARY 14, 1976 NO BROADCAST MESSAGES Response not yet recieved from medline Shall I continue listening? Type yes or y to continue, no or n to stop or dis to disconnect host. y TOTAL ACTIVE TSO USERS: 44 READY TSO LINE 6FA HELLO FROM ELHILL 3. YOU ARE NOW CONNECTED TO THE MEDLINE FILE.		
MEDLINE: SS 1 is the number for your next MEDLINE search set. You are now speaking to CONIT USER:: show data	データベースのリスト表示	show
MEDLINE: YOU MAY ACCESS THE MEDLINE, SDILINE, CATLINE, MESH VOCABULARY, JOURNAL AUTHORITY, NAME AUTHORITY, OLD MESH VOCABULARY, AVLINE, CANCERLINE, CHEMLINE, TOXLINE, CANCERPROJ AND EPILEPSY FILE SETS. YOU ARE NOW CONNECTED TO THE MEDLINE FILE. SS 1 is the number for your next MEDLINE search set.		
USER:: pick data sdiline	データベース，SDILINE指定	pick data
MEDLINE: 35 USERS LOGGED IN PRESENTLY. YOU ARE NOW CONNECTED TO THE SDILINE FILE. SS 2 is the number for your next MEDLINE search set.		
USER:: find radiation*	キーワード前方一致指定	find
MEDLINE: Your search resulted in set1 which contains this many documents: (369) SS 2 is the number for your next MEDLINE search set. USER:: ...		
find skin*		
MEDLINE: Your search resulted in set2 which contains this many documents: (700) SS 3 is the number for your next MEDLINE search set. USER:: ...		
find tissue*		

図３－５　CONITシステムの検索例　〔MEDI 77〕

・検索結果の保存・再現

F．コストと効果

複数の情報検索システムへアクセスする技術として共通インタフェースによるアプローチのコストと有効性について詳細な分析を行なうには時期的に早すぎると言えるが、ある程度の見積りは可能である。

共通インタフェースでは、検索システムにより通常行なわれる一部の機能（入力メッセージの解読や対話の操作）についてユーザと共通インタフェース、共通インタフェースと目的システム間に重複が行なわれる。また、通信設備も通常の検索システムと比べて二倍必要とする。つまり、共通インタフェースと情報検索システム、端末とコンピュータとの結合という２つの通信の確立が要求される。しかしながら、目的とするシステムの選択やそのシステムへの翻訳等幾つかの機能については機能上最低必要とするものである。単純なインタフェースを採用した単純なインタフェースの場合には約２０％のコスト増と見積ることができよう。大規模なMIT機構と教育能力のある精巧なインタフェースの場合には、５０～１００％あるいはそれ以上のコスト増になるものと思われるが、これによりもたらされる効果は、検索機能の充実とエンドユーザのデータに対する操作性の向上である。

G．さいごに

以上の点に留意してみてみると、CONITシステムが今日現在実験的なインターフェースの段階にあるといっても、採算面を除けば既に実用化に充分耐えうるシステムになっていることがわかり、複数の情報検索システムへアクセスすることに成功したCONITシステムに今後寄せられる期待はかなり大きなものであると思われる。

同じように共通コマンド言語を設定して、複数の検索システムへ統一的なアクセスを提供する例として次項に紹介するEURONETがある。

3.2.2　EURONETにおけるデータベースの共有

A．概　　要

EURONETは、科学や技術、社会、経済に関する情報を、企業や研究所、さらには個人的なユーザに、容易で平等にしかも安価で提供するために、現在開発中のＥＣを中心とするネットワークである〔CEC 77b〕。EURONETの歴史は、１９７１年６月、ＥＣ加盟各国*の閣僚級代表からなる閣僚理事会において、科学技術情報分野における加盟国家間の活動を調整する決議を、採択したことに始まる。この決議案の目的は、科学技術情報活動を刺激するとともに、この分野におけるEURONETの形成にあった。この決議案と並行し、閣僚理事会はＣＩＤＳＴ（科学技術情報委員会）を発足させ、決議案の実行に伴なう計画の作成とその執行にあたらせることになった。

＊ＥＣ加盟国：西ドイツ、フランス、イギリス、イタリー、ベルギー、ルクセンブルグ、オランダ、アイルランド、デンマーク

```
MEDLINE:
Your search resulted in set3 which contains this many documents: (1057)

SS 4 is the number for your next MEDLINE search set.

USER:>
combine set3 afor set2

MEDLINE:
Your search resulted in set4 which contains this many documents: (1885)

SS 5 is the number for your next MEDLINE search set.

USER:>
combine set1 and set4

MEDLINE:
Your search resulted in set5 which contains this many documents: (61)

SS 6 is the number for your next MEDLINE search set.

USER:>
show title docs1-3

MEDLINE:

1
TI  - ((Statistical evaluation of light protection factors))

2
TI  - Prognosis and post-therapeutic follow-up of breast cancers by
      thermography.

3
TI  - Microwave thermography: a method of detecting subsurface thermal
      patterns.

SS 6 is the number for your next MEDLINE search set.

USER:>

open skinrad
File has been opened.

USER:>
save

USER:>

st med

USER:>
show title docs1-2

MEDLINE:

1
TI  - ((Statistical evaluation of light protection factors))

2
TI  - Prognosis and post-therapeutic follow-up of breast cancers by
      thermography.

SS 6 is the number for your next MEDLINE search set.

USER:>
                    ...

pick data cancerline

MEDLINE:

 27 USERS LOGGED IN PRESENTLY.
YOU ARE NOW CONNECTED TO THE CANCERLINE FILE.

SS 1 is the number for your next MEDLINE search set.

USER:>
                    ...

USER:>
ss system

USER:>
show@save

USER:>
st med

USER:>
show title docs1-3

MEDLINE:

1
TI  - CLINICAL, HISTOLOGICAL AND HISTOCHEMICAL DIFFERENTIATION OF
      PRECANCEROUS CONDITIONS OF THE SKIN.

2
TI  - MINIREVIEW.  REPAIR OF DNA IN MAMMALIAN CELLS.

3
TI  - IMMUNOLOGIC ABNORMALITIES IN HEAD AND NECK CANCER.

SS 5 is the number for your next MEDLINE search set.

USER:>
ss mfsystem
```

2項演算（論理和）指定　combine
（#：誤入力の訂正）

文献1～3のタイトル表示　show

コモンファイル（保存ファイル）のオープン　open

次の検索結果，保存指定　save

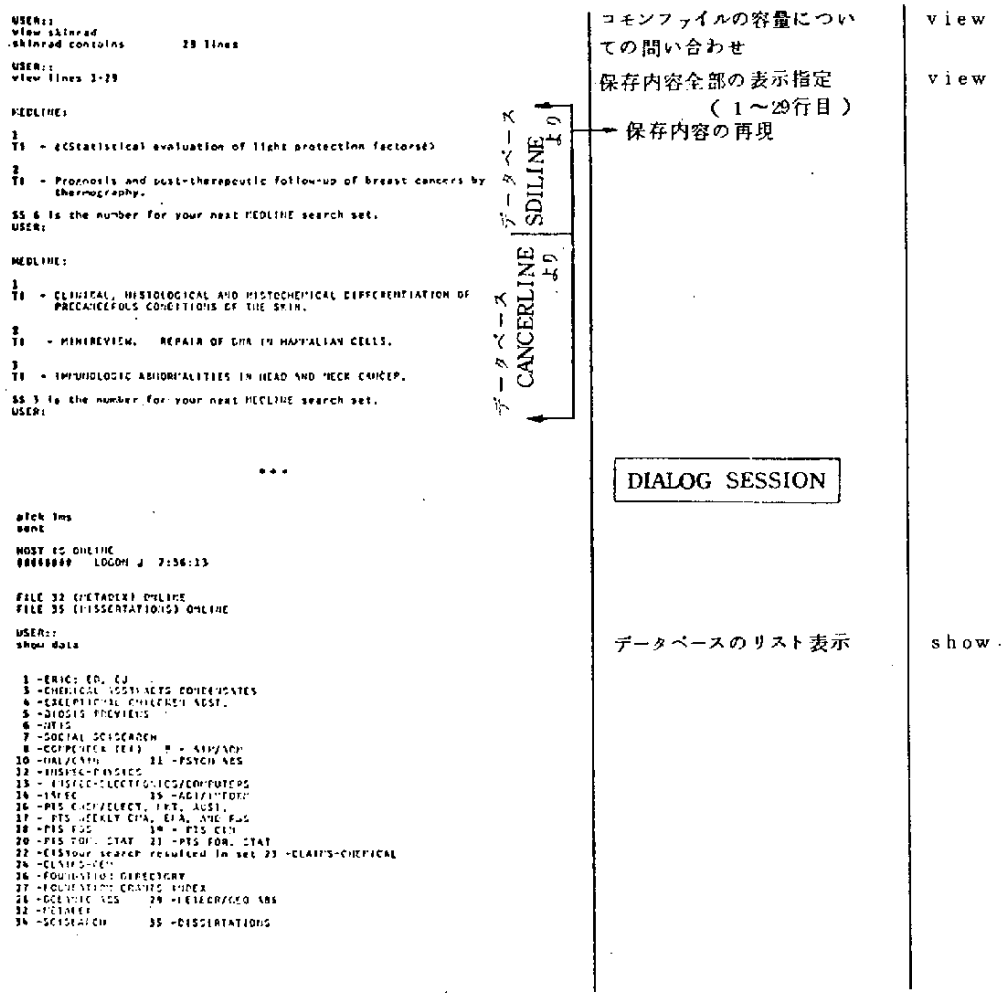
変換テーブルをmedに戻す

コモンファイル（skinrad）に保存（14行分）

データベースの変更　pick
（SDILINE→CANCERLINE）

@キャンセル・ライン指定

コモンファイルに追加保存（15行分）


USER::
view skinrad
skinrad contains 29 lines
USER::
view lines 1-29
MEDLINE:
1
T1 - @CStatistical evaluation of light protection factors@)
2
T1 - Prognosis and post-therapeutic follow-up of breast cancers by thermography.
SS 6 is the number for your next MEDLINE search set.
USER:
MEDLINE:
1
T1 - CLINICAL, HISTOLOGICAL AND HISTOCHEMICAL DIFFERENTIATION OF PRECANCEROUS CONDITIONS OF THE SKIN.
2
T1 - MINIREVIEW. REPAIR OF DNA IN MAMMALIAN CELLS.
3
T1 - IMMUNOLOGIC ABNORMALITIES IN HEAD AND NECK CANCER.
SS 3 is the number for your next MEDLINE search set.
USER:
データベース SDILINE より
データベース CANCERLINE より
コモンファイルの容量についての問い合わせ
view
保存内容全部の表示指定 （1～29行目）
view
保存内容の再現
DIALOG SESSION
データベースのリスト表示
show
atck ims
sent
HOST 65 ONLINE
00000000 LOGON J 7:56:13
FILE 32 (METADEX) ONLINE
FILE 35 (DISSERTATIONS) ONLINE
USER::
show data
1 -ERIC: ED, CJ
3 -CHEMICAL ABSTRACTS CONDENSATES
4 -EXCEPTIONAL CHILDREN ABST.
5 -BIOSIS PREVIEWS
6 -NTIS
7 -SOCIAL SCISEARCH
8 -COMPENDEX (EI) * - SIS/SCI
10 -HALCYON 11 -PSYCH ABS
12 -INSPEC-PHYSICS
13 - INSPEC-ELECTRONICS/COMPUTERS
14 -INSPEC 35 -AGRICULTURE
16 -PTS PROF/REPORT, TXT, AUST.
17 - PTS WEEKLY CPM, EPA, ANC F&S
18 -PTS F&S 19 - PTS CLN
20 -PTS FOR. STAT 21 -PTS FOR. STAT
22 -CIS(Your search resulted in set 23 -CLAIMS-CHEMICAL
24 -CLAIMS-CPI
26 -FOUNDATION DIRECTORY
27 -FOUNDATION GRANTS INDEX
28 -GENETIC ABS 29 -INTERCPP/CEO ABS
32 -METADEX
34 -SCISEARCH 35 -DISSERTATIONS

1974年9月、CIDSTは、1975～1977年の3ケ年計画を議会に提出し、翌1975年3月に採択されている。一方、CIDSTがもつ9つのワーキンググループの中心的存在となるTAG（Technical Aspects Group）により、科学技術情報のヨーロッパネットワークに関する通信施設について、1975年2月CIDSTに対して最終報告を行っている。以下、1975年3月に採択された3ケ年計画を中心に、加盟主要国の科学技術情報における動向、データベースの共有問題などについて議論する〔UTSU 76a，76b〕。

B．加盟主要国の状況

① 西ドイツ

西ドイツにおいては、既に幾つかのオンライン情報検索システムが運用されており、その一部は周辺諸国にも回線が伸びている。まずDIMDI*が1975年から提供しているオンライン検索システムを紹介する。DIMDIは、厚生省の付属機関であり医学情報管理部とデータ処理部から成っている。DIMDIで運用しているシステムは、DIRS－2といい、国内はもとより、MEDLARSのデータベースを中心に、オランダのアーヘン、ベルギーのブリュッセル、オーストリアのウィーンなどの各都市からもオンラインサービスの利用が可能となっている。化学工業協会の一部であるIDC（Internationale Dokumentationsgeselleschaft für Chemie mbH）も化学情報の提供を目的として、フランクフルトを基点にデュッセルドルフ（Dusseldorf）などへも専用回線が延びている。また、私企業ではあるが化学会社ヘキスト社（Hoechst）が、Excerpta Medica データベースを中心とするオンライン検索サービスを1971年から行なっている。

② フランス

フランスにおける科学技術情報活動は、文部省に属しているCNRSドキュメンテーションセンター〔CNRS 76〕が中心的役割を果たしている。同センターは、我が国のJICST（日本科学技術情報センター）と同様の性格を持つ公的機関である。同センターは、B.S.と呼ばれる抄録誌の刊行、PASCALテープの提供、検索サービス、シソーラスの作成など多くの業務を遂行している。同センターにおける情報処理サービスのほぼ全般を指してPASCALシステム（サービス）と呼んでいる。検索サービスのオンライン化については、センター独自の構想はもちながらも財政的問題が障害となり、実現はかなり遠いものと思われる。

一方、ネットワーク技術に関しては、第Ⅱ部1.2.2項において紹介したように、IRIAが中心となってCYCLADES〔POUZ 75〕を開発している。同ネットワークは、既に国外との接続も達成している。1974年8月にはイギリスのNPLネットワーク〔SCAN 74〕との接続を行ない、また、イタリアのフラスカティ（Frascati）に本部を持つESA／SDS（European Space Agency／Space Documentation Service）〔ESRO 73〕のネッ

---

* DIMDI：Deutsches Institut für Medizinische Dokumentation und Information

トワークとの接続も1975年9月に実現し、ESA／SDS へのオンライン文献検索が可能ととなっている。

③ イギリス

イギリスにおける科学技術情報は、1972年に国立4図書館が合併していらい、BL(British Library）が中心的役割を果している。イギリスでは1.2.4項で紹介したようにNPL（National Physical Laboratory：英国立物理研究所）ネットワークが、1973 年7月から稼働しているが、イギリスで利用可能なネットワークであるARPANETやTYMNET、ESA／SDSなどの試用から、技術的および経済的検討が行われ、今後ネットワークに対する構想がどのように展開されるか興味深い。

C．EURONETの3カ年行動計画

1975年3月、閣僚理事会が採択した内容は次のとおりである。

- 分野別情報システムの設立
- 情報ネットワークの整備
- 情報処理技術の開発、標準化、教育

① 分野別情報システムの設立

分野別情報システムには、ENDS（ヨーロッパ原子核情報システム）やSDIM（金属情報システム）、環境情報システム、EUDISED（ヨーロッパ教育情報システム）など多くのシステムが検討され、その一部については既に運用が開始されている。ここでは、SDIM について近況を報告する〔CEC 77b〕。1977年10月から3ケ月間オランダの企業や研究所にある異なる端末のユーザが、ルクセンブルグにあるSDIMデータベースへ実験的ではあるがオンラインアクセスを試みるとのことである。SDIMデータベースは、約105,000の文献を擁し、10,000のディスクリプタから成るシソーラスをもっている。なお、このシソーラスが複数の言語をもっていることも特筆すべき事項であろう。SDIMは、1977年末に一応の終止符がうたれ、その後は、1979年の本格的運用を目指した新SDIMになるといわれている。

② 情報ネットワークの整備

EURONETを構築するために、情報源の目録作成などの予備的調査やコンピュータネットワークに関する調査とパイロットプロジェクト、通信施設の創成、ネットワーク管理に関する検討、既存サービスのネットワークへの転換のための援助について充分な検討が必要とされている。1985年までは、ヨーロッパの郵政事業として行われるディジタル交換網が整備されないという前提で、ESA／SDSやCYCLADES、DIMDIなどの結合から、徐々にネットワークを広げていく計画である。EURONETでは、TYMNETやMARKⅢ などのアメリカ系のネットワークの利用は考えていない。しかし、将来の方向としては、必ずしも消極的ではないようである〔CEC 77b〕。1976年から段階的に通信施設の操業が開始され、同年中には限定された範囲で専用線を使った分散型データベースの結合実験が行われる予定である。表3－4にEURONETのデータ通信施設、図3－6にEURONETの全体イメージを示す。EURONET

は、ＥＣ加盟国だけの閉じたネットワークではなく、スイス（チューリッヒにEINのノードがある）などの参加を認めるなど、柔軟な姿勢を示している〔CEC 77b〕。

表３－４　EURONETのデータ通信施設〔UTSU76b〕

短期（1976～77年）	中期（1977～85年）	長期（1980年以降）
専用ネットワーク，既存のネットワーク，ハードウエア，ソフトウエアを含む。 ESA/SDS，DIMDI CYCLADESは最初に接続されるホストコンピュータの例となろう。	他ネットワークとの連結 1．EIN 2．ノルディック・ネットワーク（多分商用ネットワークとの連結が行われるであろう。）	ＰＴＴの国際ネットワーク，専用ネットワーク，ＰＴＴ，および他のネットワークの合体したものになるであろう。

注）ＥＩＮとはEurope Information Networkのことで，ＥＣをはじめヨーロッパ9か国が参加している最初の国際的汎用コンピュータ・ネットワークである。
ＰＴＴはPostoffice，Telephone，Telegramの略であり，国家的な郵政事業を指す。

③　情報処理技術の開発、標準化、教育

ＣＩＤＳＴの会議で確認された内容は、多国語間翻訳技術の研究と開発や、情報ネットワークに対する標準化、情報分析センター、情報専門家の教育と利用者の訓練、情報処理技術に関する事項である。

D．EURONETの需要予測

この需要予測のために設定された前提条件は次のとおりである。

- 提供するデータベースは、科学分野と工学分野を中心に２３分野とする。
- 検索サービスの利用形態は、RS（Retrospective Search）とSDI（Selective Disseminstion of Information）の二形態とする。
- 利用者の要求を満足するには、最低二つのデータベースを必要とする。
- 利用者一人当たりの年間平均ＲＳ利用回数とSDIの利用比率の割合を次のように仮定する。

ＥＵＲＯＮＥＴの需要予測の前提条件

予　測　年	年間平均ＲＳ利用回数	ＳＤＩ／ＲＳ の比率
1976 年	1.65 回／人	1
1980 年	2.83	0.5
1985 年	3.74	0.25

EURONET CONTACT POINTS:

It Per ulteriori informazione sulle rete di trasmissione dati di EURONET si prega di rivolgersi a:

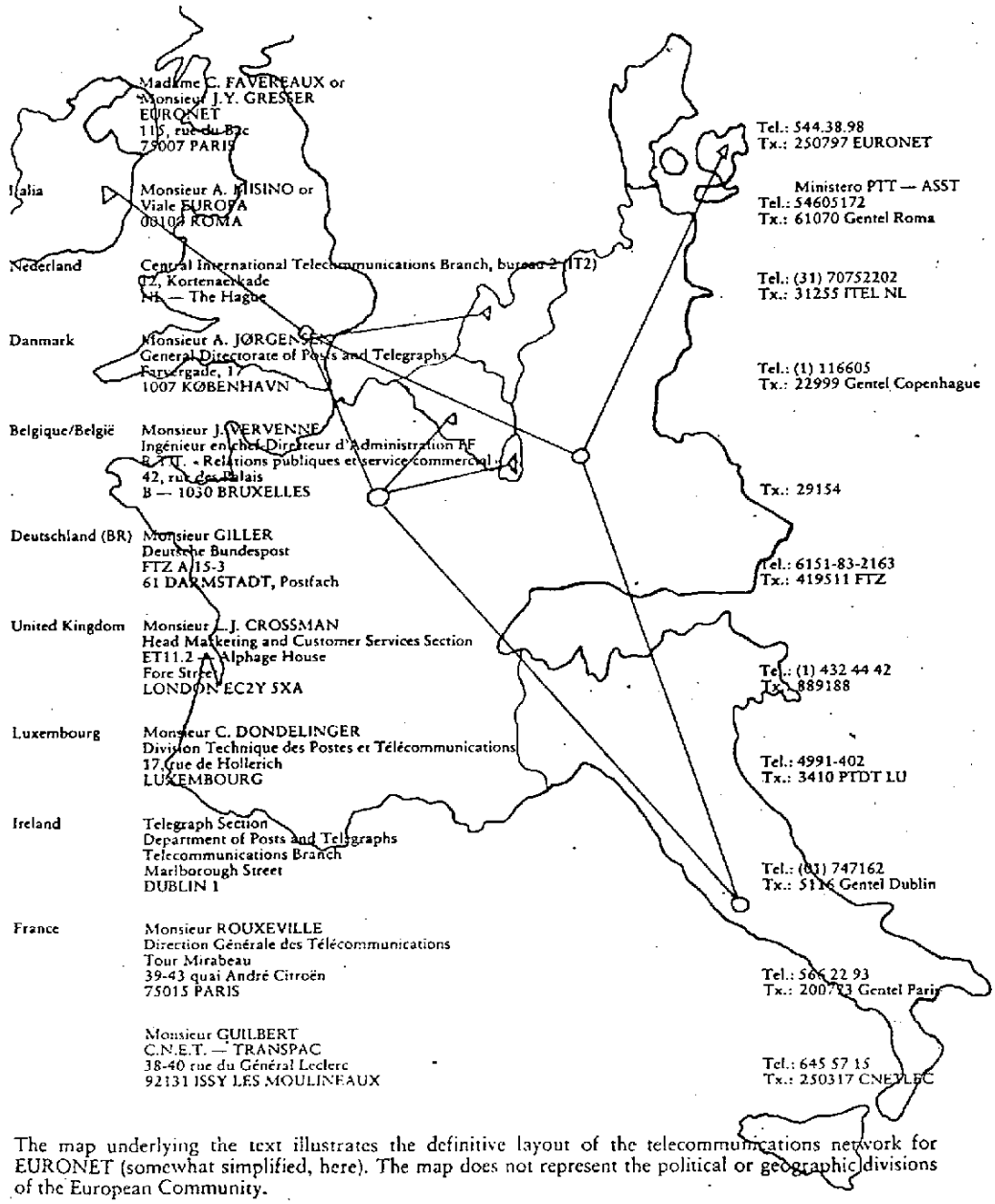
NL Als U meer wilt weten over de telekommunikatie-inrichtingen voor EURONET, kunt U zich in verbinding stellen met:

Dk Hvis De ønsker at vide mere om EURONET's telekommunikationssystem beder vi Dem henvende dem til:

F Si vous désirez davantage de renseignements sur le réseau de transmission de données pour EURONET, vous pouvez vous adresser à:

D Wenn Sie mehr über das EURONET-Datenübertragungsnetz wissen wollen, wenden Sie sich bitte an:

E If you want to know more about the telecommunications facilities for EURONET, you can now contact:



The map underlying the text illustrates the definitive layout of the telecommunications network for EURONET (somewhat simplified, here). The map does not represent the political or geographic divisions of the European Community.

図３－６　EURONETの全体イメージ〔CEC７７ｅ〕

・利用者の利用言語割合を英語７７%、フランス語５５%、ドイツ語５２%、イタリア語２５%とする。

これらの前提条件の結果得られた需要予測を表３－５に示す（UTSU76b）。また、EUSIDICによる、ヨーロッパにおけるオンライン検索の利用動向の調査によれば1971年の20万件以下から1976年には240万件に増加しているとある（図３－７）〔CEC 77a〕。

表３－５　　需　要　予　測

予　測　年	利用者（人）	利用度数（回）	トラフィック（bit）	1サーチ当りのコスト（$）
1976年	60,000	99,000	0.1×10^{12}	20
1980〃	960,000	2,720,000	3.3×10^{12}	10
1985〃	2,350,000	8,800,000	10.4×10^{12}	5

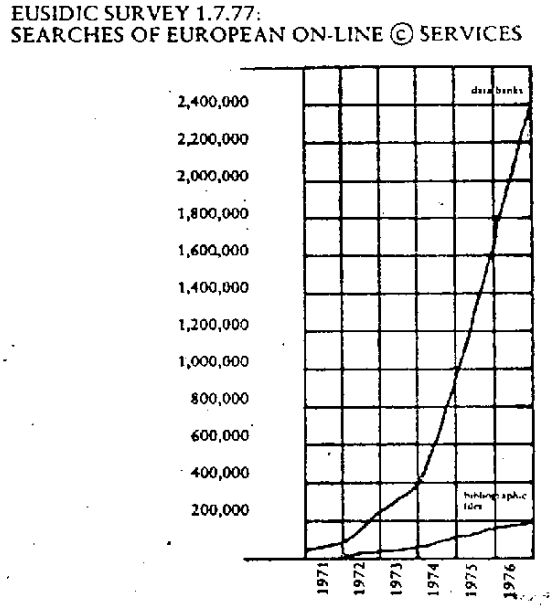


図３－７　欧州におけるオンラインサービスの利用状況

E．データベースの共有と実現形態

本項では、EURONETにおけるデータベースの共有について述べる。ARPANETをはじめとしてコンピュータネットワークは、リソースの共有を目指しているが、このEURONETは特にデータベースの共有に主目的をおいている。そしてデータベースの内容は、いわゆる科学技術情報であり、書誌的事項や抄録、数値データなどを含むことになっている。また、一次情報の提供まで含めたシステムの構築を目途していることは意義が深い。このように、多くの国々から賛同

を得られたEURONETは、西ドイツやフランスなどの優れた指導力もさることながら、加盟各国が同じような環境にあったのも見逃せない。つまり、各国がオンラインシステムを独自に開発するには、あまりにもコストがかさみ、採算ベースに乗せるには、相当困難な状況にある。また、データベースの多くは、英語表現がとられており、言語の問題もデータベースの共有に関しては比較的容易に解決できそうである。

なお、１９７７～１９７９年における多言語委員会（multilingual programme）が１９７７年の9月に設置された。この委員会はCETIL（Committee of Experts for the Transfer of Information between European Languages）と呼ばれ、ＥＣ加盟諸国のメンバから構成されている。この委員会の目的は、欧州系言語間の情報転送上の改良に関するCEC（Commission of the European Communities）の実行計画や遂行することである。まず、第一回の委員会では、機械翻訳、データバンクの用語、複数言語によるシソーラスに関して検討が行われている。

しかし、EURONETは既存の情報検索システムからのアクセスも提供しようとしているので、各システムがもつ固有のアクセス言語間のインタフェースの問題が、特に初心者ユーザにとって大きな問題となる。EURONETで提供される種々のデータベースに対して、ユーザが利用できるように、検索コマンドの標準セットに関する実現の可能性について研究が行なわれている。複数の国がデータベースを共有する問題に関連して生ずる問題として、コストについて議論する必要があろう。

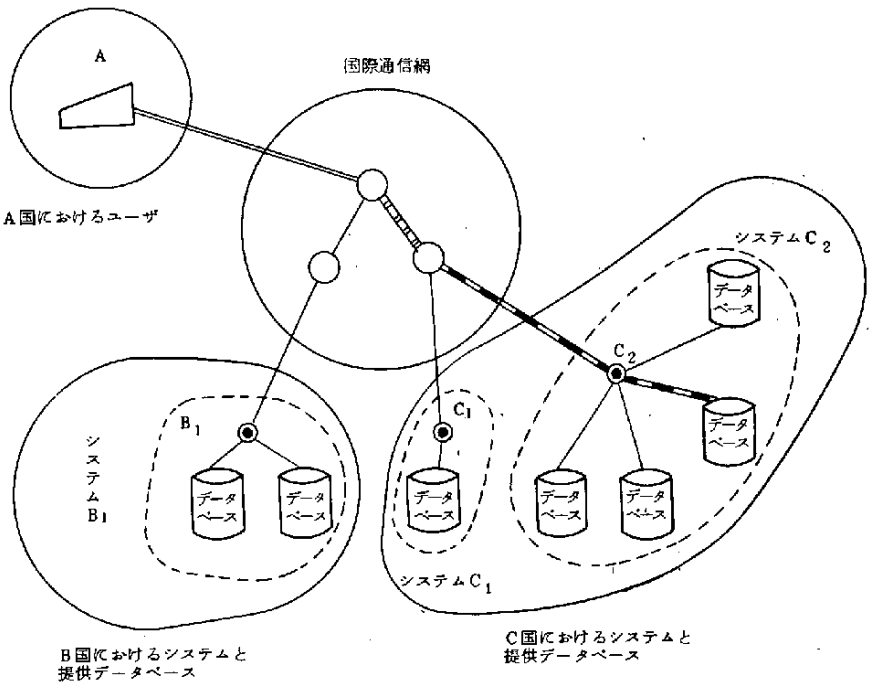


図３－８　EURONETにおける価格体系

図３－８は、A国のあるユーザがC国のC_2システムのデータベースを利用する形態を示したものである。この場合の価格は、三つに分けて考えることができる。

- ――― の部分…端末を最も近いネットワークアクセスポイントに接続するために用いる公衆電話網の料金（local PTT charge）。
- ┬┬┬┬┬ の部分…国際通信網の使用料金で、1978年初頭に設定される料金は、距離に独立した従量制を基本とする予定である。
- ■■■■ の部分…最終的に使用する個々のインフォメーションサービスシステムの使用料金である。

なお、初期の段階で参加予定の機関数は２７の多きを数えデータベースの数は約１００にものぼっている（表３－６）〔CEC 77a〕。

F．EURONETの検索コマンドと検索例

EURONETには２７の機関（システム）の参加が見込まれ、それによりアクセス可能なデータベースは約１００と言われている〔CEC 77a〕。既に個々運用されている検索システムとユーザ（特に初心者）とのインタフェースをどう解決するかは大きな問題である。個々の検索システムは、独立のコマンド体系をもっているのが普通である。したがって、あるシステムに熟知していても、他のシステムを利用するときには、そのシステムがもっている検索コマンドを一通り覚える必要がある。このような、システムが多く存在することに起因して生じる，ユーザへのわずらわしさを解決する手段としてはCONITに見られる共通言語方式がある。EURONETにおいても、CONITと同様のアプローチがとられており、標準コマンドセット（standard command set）として表３－７に示すような機能を設定している。各コマンドのオプション機能などに関する解説は〔NEGU 77〕が詳しい。また、西独のDIMDIシステムをEURONET経由で使用した検索例を図３－９に示す。

G．今後の期待

科学技術情報の交流を国家レベルで目指しているEURONETは、米国におけるコンピュータネットワークの技術的問題やインフォメーションサービスの実状とは異なった意味で興味をひく。複数のインフォメーションシステムの結合に関しては、前に述べたCONITが先駆者であり、既に一応の成果が得られている。しかし、規模的にはEURONETの比ではなく、EURONETに寄せられる期待は研究者や各企業に留まらず、国家的レベルから個人的利用者まで幅の広いものとなっている。今後は各国の情報処理政策も含めて、EURONET の動向に注目する必要があろう。

3.2.3 ま と め

分散型データベースを用いた二つの実用システム、CONETとEURONETを紹介したが、いずれも既に運用されている文献検索システムの統合化をねらったものである。このような構想が現実的となってきた背景には、文献検索システムに代表されるインフォメーションサービスが現在多くの問題点を抱えていることにもよろう。次にその一部を示す。

表３－６　Date Base Services Offered for Initial Connection to EURONET

	Host	Location	Data Bases
1.	British Library (BLAISE - British Library Automated Information Service)	London area	UK Marc, US Marc, Medline, Sdiline, Toxline, Chemline, Cancerline, other British Library data bases (including the B.L. Conference Index).
2.	Info-Line	London area	CAS, INSPEC data bases, Derwent data bases, and other, mainly UK, data bases.
3.	Computer Aided Design Centre	Cambridge	Engineering design.
4.	National Computing Centre	Manchester	Computing Hardware, Computing Software, Computing Services, Computing Education, Computing Literature, Computer Installations.
5.	Belgian Ministry of Economic Affairs	Brussels	INIS, EPIC.
6.	DIMDI/FIZ 1	Cologne	Biosis, Cancerline, Poisons data bank, Excerpta Medica, IDIS (Social Medicine) CAB Index Veterinarius, International Pharmaceutical Abstracts, Hospital Affairs, Medline, Psychological Abstracts, Science Citation Index (Asca IV), Toxline, Sport Science.
7.	IDC/FIZ 3	Frankfurt	Chemical Abstracts Condensates (CAC), Chemical Abstracts Subject Index Alert (CASIA), Chemical Industry Notes (CIN), IDC—Literature documentation, IDC—Patent documentation, DECHEMA—Literature documentation, DECHEMA—Materials data documentation, Literature documentation on plastics, rubber and fibres, Literature documentation on process engineering.
8.	ZAED/FIZ 4	Karlsruhe	Astronomy and Astrophysics Abstracts, Compendex, Energy data base, NTIS, Nuclear research and technology data base (IKK), INIS, INSPEC—Physics, Mathematics information system (MIS), NSA, Physics information system (PHIS), Physical data information system, High energy physics data base (HEP), Plasmaphysics Index, Plasmaphysics Technology Index, Surface and Vacuum Physics Index, Cambridge crystallographic data files, Evaluated nuclear structure data file (ENSDF), Conference calendar for energy, physics, mathematics, Conference reports on aerospace, Data base on institutions.
9.	DOMA, ZDE/FIZ 16	Frankfurt	DOMA mech. engineering literature data base, ZDE elec. engineering literature data base, DRE electrical engineering data base, Compendex, INSPEC — Electrotechnology/Computers and Control.
10.	ZMD	Frankfurt	AGRIS, Agricultural sciences literature data base, Food Science and Technology Abstracts, SDIM, Deutsche Bibliographie.
11.	Institut Textil	Paris	TITUS III.
12.	Centre Interuniversitaitaire de Calcul	Grenoble	Cancernet Sabir, Pascal medical data bases.
13.	Fédération Nationale du Bâtiment (CATED)	Paris	Ariane.
14.	Necker Hospital	Paris	Drugs data bank (BIAM).
15.	Thermodata	Grenoble	Thermodynamic data.
16.	PLURIDATA	Paris	Chemical data banks (including crystallographic and mass spectrometry data, nuclear magnetic resonance data, DARC system).
17.	INRA — Ministère de l'Agriculture	Paris	CAB, CAIN, Zoology, Bioclimatology, CDIUPA (food science technology).
18.	French Host	France	CBAC, DARC system.
19.	Institut de recherche des transport	Paris	IRRD.
20.	SDS	Frascati	CAS, SCISEARCH, INSPEC — Physics, Electrotechnology, Computers and Control, Compendex, NTIS, World Aluminium Abstracts, NASA, Electronic Components, Metadex, Environmental Science Index, Pascal Data Bases, Pollution Abstracts, Oceanic Abstracts, Biosis.
21.	Commission of the European Communities	Luxembourg	Community data bases.
22.	Datacentralen	Copenhagen	CAS, Food Science and Technology, Medline (through SCANNET), Compendex.
23.	CNUCE	Pisa	Data bases on: ecology, geothermal science, iconography, oceanography, legal documentation, fine arts, citation index (Italian authors).
24.	Interfaculty Computing Centre	Rome	US Marc, Marc Italy, SPIN.
25.	Interfaculty Computing Centre	Naples	MTS.
26.	CSATA	Bari	Meteorological data bank (concerning the South of Italy), agriculture.
27.	Joint Research Centre	Ispra	Nuclear sciences, material properties and other data bases.

表３－７　EURONETの標準コマンドセット

	コマンド名	省略形	入　力　形　式	機　　能
①	BASE	BAS	BASE　INSPEC BASE　MEDLARS　; ED=01.74 TO　07.77	使用するデータベースを指定する。
②	STOP	STOP	STOP	会話の終了を宣言する。
③	FIND	F	FIND　AU=MILLER	検索ステートメントを入力する。
④	DISPLAY	D	DISPLAY FT=SYNCHRON$	キーワードのＡＢＣ順リストあるいは、論理的に関係のあるキーワードの表示。
⑤	SAVE	SA	SAVE	検索ステートメントの保存指定
⑥	SHOW	S	SHOW SHOW R10 TO 20;15	検索結果のオンライン表示（タイプ／ディスプレイ）指定
⑦	PRINT	P	PRINT PRINT L=DIMDI,COLOGNE W=HAUSSTR. 27; SORT=AU;REMOTE	検索結果のセンタ出力指定
⑧	CONNECT	C	CONNET DIMDI VIA EURONET	システムとの接続指定
⑨	DEFINE	DE	DEFINE TL=ENGL DEFINE SORT=AU; PY	システムのデフォルト機能を変更するためのユーザマクロを生成する（シソーラスの言語やプリントサイズなどの指定）
⑩	DELETE	DEL	DELETE 17 DELETE ALL	検索ステートメントやPRINT要求の解除指定。
⑪	MORE	M	MORE	検索結果やキーワードリストの促進を指定。

	コマンド名	省略形	入　力　形　式	機　　能
⑫	BACK	B	BACK	検索結果やキーワードリストの遡及を指定
⑬	HELP	Hor?	? or HELP? ? COMMAND ?? or HELP?	オンラインのガイダンス機能を指定。 最後の検索状態の説明 コマンドの解説 システムの全般的な解説
⑭	NEWS INFO		NEWS INFO { EURONET COST SCHEDULE USERS STATUS }	システムの最新情報が得られる。 EURONETの最新情報が得られる。 検索などの料金情報が得られる。 検索サービスなどの時間に関する情報が得られる。 会話中のユーザ数が得られる。 その時点での検索に関する情報（検索NO. 検索者名、開始時刻）が得られる。
⑮	OWN		OWN	システム固有の検索コマンドによる実行が可能となる。

- 検索精度の向上（ユーザの満足度）
- 操作性の向上（初心者ユーザにも受け入れられるもの）
- 応答タイムの短縮
- 時間割方式の導入
- 通信網との関連
- 商用システムにおける採算性

米国の三大文献検索システム（Lockheed／DIALOG，SDC／ORBIT，NLM／MEDLINE）は、１１～５１のデータベース、２００～１２００万件以上の文献情報をユーザに提供している。ユーザにとって見れば、常時これらの情報が入手可能である状態にあるのが望まれる。一方、運用機関側からみると、コスト、記憶容量（二次記憶）の制約から、全情報を常に提供するのが難しくなってきている。現に、ORBITシステムは、３６のデータベースを四つの時間帯に分けて、サービスを行なっている。欧州のESA／SDSのRECONシステムも、増大する文献情報に対処する為、近い将来、この時間割方式を導入すると言われている。さらに、我が国のJICSTにおけるJOISシステムもこの方式を採用している。JICSTの場合は、曜日により四つのファイルから2～3のファイルを選択して提供している。この方式は、ユーザにとって決して好ましい状態とは言えず、

```
/?
CONNECT DIMDI VIA EURONET
user-id? UB-COLOGNE
password? EXAMPLE

you are now connected to DIMDI's siemens 7.755

/?
DO DIRS2
password? XYZ
customer-no? ZNØØ5
customer-name? UB-C
/?
DEFINE PROMPTS = SHORT; OS = 54,6
/?
?BASE
The base-command gives information about offered
databases (BASE) and/or assigns one database to the
user (e.g. BASE BIOSIS). Some base-parameters are
provided:
ED     =          .........
NOFIND=           .........
P      =          .........
SN     =          .........
?      =          .........     (explanation)

/?
BASE
1 medlars       01.75-08.77
2 biosis        01.75-09.77
3 cancerline    01.70-08.77
4 sci-search    01.77-07.77
5 psyabs        01.67-06.77
```

図3－9　EURONETの使用例

```
/?
1;ED=01.76 TO 08.77;NEW

new search-number: Ø138

/?
DISPLAY CT=HEART$

1 heart                      120
2 heart abnormalitites        40
3 heart artificial            10
4 heart atrium                30
5 heart ausculation           15
6 heart diseases              33

/?
MORE

7 heart injuries               5
8 heart rate                  10
9 heart sounds                13
10 heart surgery               8
11 heart-lung machine          7
12 heart, mechanical           7

/?
FIND (AND 2;9;CT=CHILD)

13 number hits is: 10

/?
FIND 13 AND FT=AORT$

14 aorta
15 aortae
16 aortal
17 aortic
```

```
/?
14;17

18 number hits is: 2

/?
SHOW TI;AU

ti: correlations between polygraphy, phonocardiographic
    data and interrarcadiac pressure values, in aortic
    stenosis
au: pezzano a; rizzi gm; bianchi em;

/?
MORE

ti: congenital valvular aortic stenosis: clinical
    detection of small pressure gradient
au: ellison rc; wagner hr; weidman wh; miettinen os;

/?
PRINT L=DIMDI,COLOGNE;ED=Ø1.66 TO 12.77; F=AU;TI;SO;AB;CT

print offline search 18, 2 hits
confirm? Y

/?
STOP QN=NEW

new search number : Ø139
```

```
/?
DISPLAY CT UP EAR CANAL; CR

1 sense organs                  A9                  50
2 ear                           A9,246              40
3 ear, external                 A9,246.272          30
4 ear, canal                    A9,246-272.396      12

/?
FIND (OR 1;2;3;4) AND FT=(EAR,EARS)...NOSE

5 number hits is: 7

/?
SHOW R1;TI

ti: rhabdomyosarcoma in the ear, nose and throat

/?
SAVE

/?
PRINT 15;R1 TO 7;F=TI;AB;AU;SORT=PY;REMOTE

print remote search 5, 7 hits
confirm? Y

/?
STOP

you're now leaving DIMDI and EURONET, good bye.
```

可能な限り避けなければならないと思われる。例えば、CONITシステムやEURONET のように複数システムが相互に情報（文献データ）を交換したり、米国カリフォルニア州の図書館ネットワークなどのように、今後はデータの交換という共有方式について一層の検討が必要となろう。また、検索コマンドは、個々のシステムがそれぞれ独自に開発を行っているのが現況で、実際に数多くの検索言語が存在する。ユーザにとってみれば、システムやデータベースの単なる違いだけで、一々別の検索言語を習得するわずらわしさがある。この問題を解決したのが、本節で紹介したCONITシステムやEURONETである。最終的な検索プロセスは、個々の検索システムにまかせてあるが、それに至る中間的なインタフェースを設置し、それに対する共通コマンドを設定している。これにより、ユーザは、一つの検索言語（コマンド）を習得することにより、各地に散在する複数のシステムに自由にアクセスすることが可能となった。通信網の発達なしには、この種の技術も実現はしなかったと思われる。

なお、共通アクセス言語の設定においては、CONITシステムとEURONETは共通であるが、インタフェース（変換）機構の設置方法に違いがある。CONITシステムがMITのMULTICS上に唯一のインタフェースを設置しているのに対し、EURONETでは、各検索システム側にそれぞれ独立に設置されている。しかし、今後はこの種のアプローチが、既存ユーザや他のシステムに与える影響も少なく、また比較的導入が容易であることからも盛んになると思われ効果も期待できる。

3.3 新しい分散型データベース — DBMS統合

前節で論じたCONIT、EURONETの例は、種々の会話型情報検索システムの会話インタフェースの変換を行なうことによって、データベースの統合を試みたものである。これらの特徴は、文献という同一の論理構造を持ったデータを扱かっており、ユーザはデータベースへ対して同一の視点を持っていることである。種々の情報検索システムの主要な相違点は、各々の文献データベース内でどのようなアクセスキーとキーワードが使用されるかに関している。

この節では、一般的なアプリケーションを想定している場合を考えるために、各データベースサイトはデータベース管理システム（DBMS）であるものを考える。即ち、各データベースサイトは、あるデータモデルを備え、あるローカルトランザクションを処理出来るオートマトンである。個々のデータベースサイトのDBMSは異種であり得る。

これらのDBMSは、コンピュータネットワークを通して互いに結合されている。DBMS 間がどのような通信レベルを用いて交信するかは重要である。Nahouraii 等〔NAHO 76〕は、DBMS 間の通信レベルとして次の4つがあるとしている。

1）問合せ言語レベル

2）親言語レベル

3）アクセス方法レベル

4）微視的レベル

親言語レベルでは、DBMSへのトランザクションは高級言語で書かれたプログラムによって作られる。微視的レベルでは、格納媒体から直接に完全なファイルをアクセスせねばならない。このレベルでの交信では、システムのデータ格納方法、システムの機密保護制御の大部分のバイパス方法、有効なデータ検索のためのアクセスパスの選択方法等の詳細な物理的情報をユーザは知らねばならない。このことはユーザへとって大きな負担であり、このレベルでの通信はあり得ない。どのレベルを選択するかは、高水準通信（問合せレベル）を用いることによるユーザのアクセス容易性と、このことによる高水準レベルから低レベルへの翻訳のコスト（有効性）とのかねあいである。即ち、問合せレベルのユーザの容易性と、アクセス方法レベルのシステムの有効性とのトレードオフでもある。

第2章のDBMSの議論において、DBMSの有効性に関するレベル（内部スキーマ）と、ユーザの問題に依存したレベル（外部スキーマ）とデータベースの一意で定常的な意味記述レベル（概念スキーマ）とへの分離が必要であることを述べた。DIAM Ⅱ〔SENK 76a〕におけるように、アクセスパスレベル以下は内部スキーマの問題である。DBMS間通信として必要なことは、互いの物理的特性の変化から独立であることである。この意味で通信レベルとして、アクセスパスレベルよりも上位である親言語レベルと問合せ言語レベルとを考える必要がある。後述する POLYPHEMEにおいても、通信レベルとしてこの2つのレベルが考えられている。

コンピュータネットワーク上に分散されたDBMSの統合において、ユーザが必要とするデータの所在と、データベースサイト固有のDBMSの異種性とはユーザへ視えないことが重要である。分散型データ管理におけるこの様な不可視性（invisibility）は、Rothnieによって強調されている〔ROTH 77c〕。ユーザに対する不可視性の実現のためには、ユーザにとって必要なデータの意味を記述するためのシステム全体で共通なデータモデル及びアクセス言語が必要となる。このためには、各DBMSの異種性によって生ずる問題と、コンピュータネットワークを通して DBMSを結合するために生ずるデータの分散の問題とを明確に分ける必要がある〔図 3－10〕。異種性の問題は、共通・ローカル、ローカル・ローカルの変換の問題である。分散の問題は、必要なデータが、どのサイトに存在するかを明らかにすることである。分散の問題は、異種性の問題よりも上位にあると考えられる。

以下に、各システムを、共通モデル及びアクセス言語、異種性の問題、分散の問題に関して論じる。

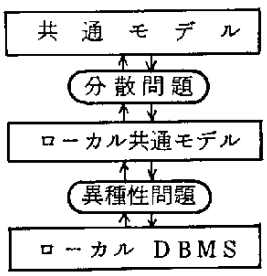


図3－10　分散型データベースシステムの基本概念構成

3.3.1 INGRES

INGRES(Interactive Graphics and Retrieval System)は、UC. Berkeleyにおいて開発された関係DBMSである。これは、PDP—11上にインプリメントされている関係データベースとグラフィックシステムとから成り、UNIXオペレーティングシステムのもとの通常のユーザジョブとして存在している〔HELD 75〕。INGRESは当初単一ホストシステムであった〔HELD 75〕が、最近コンピュータネットワークによって結合された複数のPDP—11上の関係データベースから構成される分散型データベースを管理するように拡張されている〔STON77〕。INGRESのホストコンピュータは、UNIXオペレーティングシステム〔RITC 74〕が稼働出来ることが前提とされている。即ち、INGRESを構成するDBMSは、同種の関係DBMSである。INGRESの実現は、UNIX自身が記述されている高級プログラミング言語"C"で行われている。解析(parsing)は、UNIX上で利用できるコンパイラコンパイラ"YACC"により遂行される。なお、UNIXは、ベル研究所でPDP11シリーズ用に開発されたコンパクトなTSSオペレーティングシステムである。またMITのMULTICSシステムの開発に、ベル研究所がかって参加していたこともあり、UNIXの設計思想には、MULTICSのアイデアが数多く導入されているのも特徴の一つと言えよう。

A. データモデルと言語

INGRESにおいては、個々のデータベースサイトは関係データベースであるとともに、ユーザへ対してはQUEL〔HELD 75, STON 74〕と呼ばれる関係計算言語を提供している。QUELは、CoddのALPHA〔CODD 71b〕に基づいたものである(第2章の関係モデルを参照のこと)。

B. アーキテクチャ

INGRESは、UNIXオペレーティングシステムの通信機能を用いて、1つのマシンのプロセスが他のマシン上の従属するプロセスに同期させ、そのプロセスとデータの交換を行なうことが出来る〔図3－11〕。

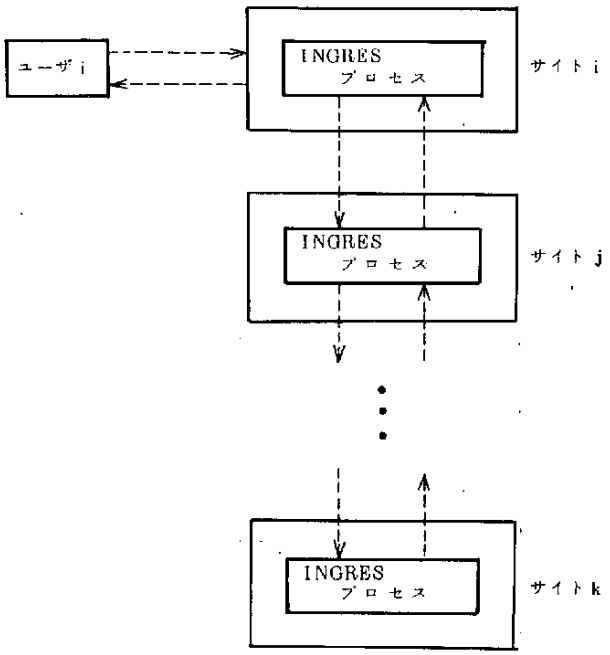


図３－１１　分散型 INGRES

C．分散問題

INGRESの各サイトのデータベースは、関係の集合とシステムカタログとから成る〔STON 76〕。第１の問題は、ユーザがどのように関係をアクセス出来るかである。即ち、必要となるデータ項目がどこに存在しているかを知る必要があるか（visible）、又はないか（invisible）の問題である。

第２の問題は、データ項目の分散形態である。関係が１つのサイト内で閉じているか、又は複数サイトへまたがっているかである。関係の複数サイトへの分割方法としては、組単位、属性単位、組と属性との組み合せたもの（関係の任意の一部）の方法が考えられる。この分散形態に関連して、関係の冗長コピーについても考えねばならない。冗長性は、信頼性と効率の点から望ましいが、一方では冗長コピー間の更新に対する同期の問題を生じさせる〔STON 77〕。又、関係が複数サイトへまたがった場合の冗長性は、分散形態をより複雑化させる。分散形態としてSDD—1における1つの具現化（materializtion）（SDD—1 を参照）を考える。即ち、不可視なユーザアクセスにとって、冗長性とはシステムによって、その信頼性とパフォーマンス向上のために、管理されるべきものである。この２点をまとめると図３－１２のようになる。図中括弧内の視点は、INGRESにおける分散型データベースの３つの視点（view）〔STON77，MARI 75〕である。

Aは、INGRESの視点１へ対応する。関係は１つのサイト内で閉じている。ユーザは、関係の存在するサイトを指定しなければならない。１つの会話では、１つのサイトへしかアクセス出来ない。

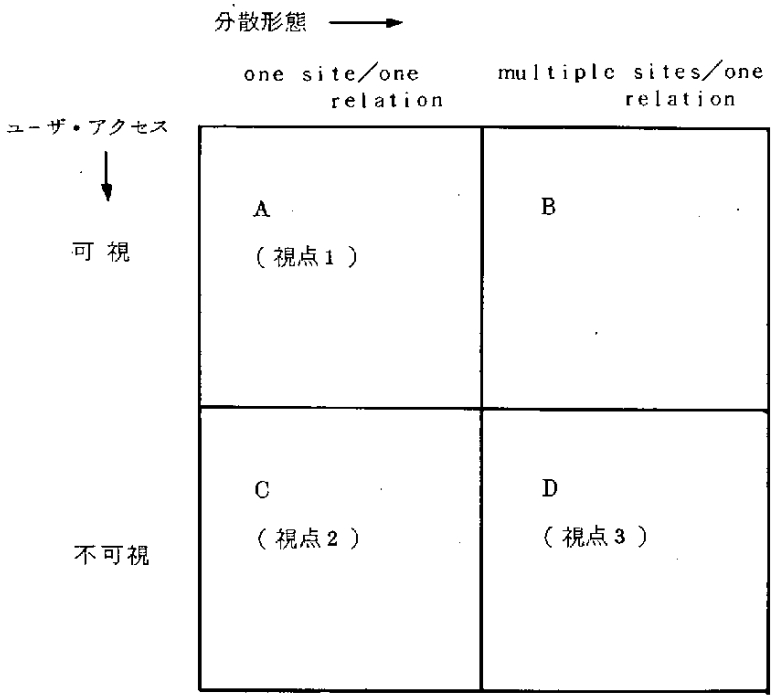


図 3 － 1 2　　分散の視点

Bは、関係が複数サイトにまたがっており、ユーザはこの分散形態を知っていなければならない。ユーザのアクセスは極めて複雑である。

Cは、視点2へ対応するユーザは、必要な関係がどこに存在するかを知る必要がない。関係は、1つのサイト内で閉じている。

Dは、視点3へ対応する。ユーザは、Cと同じ様に、関係の所在に係わりなくアクセス出来る。しかし、関係が複数サイトへまたがっているために、システムは関係の一致性（consistency）を保たねばならない。

INGRESにおいては、通常の関係に対して視点3（D）を適用している。システムカタログへ対しても視点3を用いている。

一致性条件は、次の通りである〔HELD 75〕。関係R_1，… ，R_nの全てがあるサイトS_0にある場合、これに対する交信Iによって生ずる結果をTとする。これらの関係R_1,…,R_nが各サイトS_1,…,S_nに各々分散している時、各関係への交信I_iとその結果T_iとする時、$I=\{(I_1, I_2, \cdots, I_n)\}$，$T=\{(T_1, T_2, \cdots T_n)\}$ならば一致性は保たれているとする。

D。データベースの分散に伴う必要機能

ここでは、分散問題を解決するために必要な機能として、ユーザコマンド、システムカタログ（ディレクトリ）、問合せ分割、同時実行制御、一致性制御について論じる。先に述べたようにINGRESは視点3を、関係及びシステムカタログへ適用している。又、INGRESは、分散に関

する機能に対する“GOD”を持たない。GODがパフォーマンスの隘路となることと、ほとんど全ての要求はローカルなものであることとが、この理由である。大部分の要求がローカルでないならば、GODを備えた巨大集中システムとなろう〔STON 77〕。

① ユーザコマンド

分散視点（1～3）をサポートするために下記の様なユーザコマンドが新たに必要となる。

a）LOCATION

関係の物理的所在情報を与える。

例） RANGE OF E IS EMP (LOCATION=1)

b）MOVE

関係を、あるサイトから他のサイトへ移動させる。

例） MOVE RELATION-NAME(LOCATION=X) TO RELATION-NAME
(LOATION=Y)

c）視点3における関係の分散方法として次の3つがある。

ⅰ）関係内の組を移動させる。

例） RANGE OF R IS RELATION
RETRIEVE INTO W (R.ALL)
WHERE QUALIFICATION
DELETE R WHERE QUALIFICATION
MOVE W TO RELATION (LOCATION=X)

ⅱ）組単位ではなく、関係ごと移動させる。

ⅰ）とⅱ）において、関係の分散の規定（distribution criteria）は、ユーザ自身が定めるとともに保障しなければいけない。

ⅲ）DIST_CREATE

ⅰ）とⅱ）へ対して、分散規定はシステムが自動的に管理する。これは一般的には次のように書ける。

DIST_CREATE RELATION_NAME({DOMAIN_NAME=FORMAT})
({QUALIFICATION=LOCATION})

例） DIST_CREATE EMPLOYEE(NAME=C10, AGE=I2, SALARY=I4)
RANGE OF E IS EMPLOYEE
(E.SALARY <10000=LOC1, E.SALARY>=10000=LOC2)

このコマンドは、分散された関係を生成するとともに、更新が分散規定を守ることを保障し、情報の検索範囲を限定するために分散規定を利用する。

d）BACKUP

バックアップコピーを生成する。

例） BACKUP OF RELATION_NAME(LOCATION=X) IS
RELATION_NAME(LOCATION=Y)

e）関係の生成と消去に関連して、一致性保障のために通信トラフィックの大きなオーバヘッドが生じる。このため、関係として、次の２種類を考える。

ⅰ）正規関係（regular relation）。これはネットワーク内で共有される関係である。

ⅱ）ローカル関係。これは１つのホスト内でのみ使用される関係である。

これらの２種の関係は、名前づけによって区別される。ローカル関係へのアクセスはオーバヘッドが少ない。

② システムカタログ

INGRESの各ホストは、自分のなかに存在する関係に関するシステムカタログを持っている。このシステム内カタログも関係である。システムカタログ内に含まれるべき情報としては次の４つがある。

a）関　係　名

b）解析情報（領域名、形式等）

c）効率化情報（関係の組数、格納構造等）

d）一致性情報（保護、保全性制御等）

INGRESにおいて、視点３がシステムカタログへ対して適用せねばならないとしている。

INGRESのユーザが、システム内の幾つかのサイトと交信するためには、他のサイトのカタログ情報が必要になる。あるサイトが、他のサイトのカタログ情報を持つことは、カタログの冗長コピーが存在することである。冗長さに対するオプションとして次の５つがある。

a）GODと呼ばれる１つのサイトに、システムカタログの全集合を格納する。他のサイトをアクセスしようとするサイトは、必ずGODから情報を得ねばならない。

b）全てのサイトが各ホスト上の関係名情報を持つ。

c）全てのサイトが、各ホスト上の関係名と解析情報とを持つ。

d）全てのサイトが、各サイト上のカタログ情報を全て持つ。

e）あるサイトが、他のサイトをアクセスしようとする場合そのサイトのカタログ情報の全てを格納する。他サイトのシステムカタログが更新された場合でも、このコピーされた情報への更新は行なわない。一定時間がたつと、このコピー情報は消去される。

一般的に、冗長コピーの存在は、更新情報の拡散のためのトラフィックの増加と、更新の同期の問題を生じさせる。一方、a）の場合のように1つのGODに全情報が集中されている時、更新の同期問題とトラフィックの増加問題は生じない。しかし、信頼性と有効性の問題を生じさせる。GODの障害は直ちに全システムの障害である。

INGRESでは、GODは存在させないことにしている。理由はGODがパフォーマンスの隘路となることと、更新アクセスへ対して検索アクセスが大半であることである。又、視点3をINGRESがサポートするためには、少なくともb）の程度（各サイトは、他の全てのサイトの保有する関係各情報を持つ）の冗長度が信頼性とパフォーマンスの点から必要である。

INGRESでは、e）の方法も有効と考えている。ある仕事に必要な情報だけをもってくるという作業集合（working set）〔SHAW 74〕概念に基づいたe）の方法は、参照が局所的であるならば、トラフィック効率からみて有効である。

INGRESにおけるシステムカタログの構成方法の議論は、まずシステムカタログ情報を4種（a－d）のクラスへ分類している。各クラスの情報は、ネットワーク全体の情報である。各サイトが（例えば、a）だけか、a）とb）だけかのように）どのクラスの情報を持つかが、システムカタログの分散問題である。クラス内の情報を、各サイトへ分割すること、即ち部分的冗長性についてはオプションe）の特殊な場合以外考えていないもINGRESにおけるシステムカタログの成果は、システムカタログの情報種類によって分散形態を考えた点である。

③ 問合せの分割（query decomposition）

複数サイトへまたがる問合せ、即ち多変数問合せ（maltivariable query）を、単一サイトへの問合せ、即ち一変数問合せ（single variable query）へ分割することが必要である。Wongは、多変数問合せから一変数問合せへの分割の最適化について論じている〔WONG 76〕。問合せ分割は、SDD－1において詳論する。SDD－1における問合せ言語はINGRESと同じQUELを用いており、問合せの分割については同様に議論できる。

④ 同時実行制御

同時実行制御（concurrency control）問題は、デッドロックの防止と検出とを行なうことである。

デッドロックは、前もって1つのINGRESコマンドにとって全て必要なリソースを要求することによって防止される。安全性（safety）は、リソースのロックの適当な集合を要求することによって保障される。必要なロックが全て得られないならば、全てのロックは解かれ、ある時間の後に再試行される。

デッドロックが検出されると、どれかのプロセスを犠牲としてとり除く。デッドロック検出は、全ネットワークの“ロックアウト図”（lock-out diagram）〔MACR 76〕をつくり、この図の中の循環部分を見つけることである。この操作を行なうには、ネットワーク全体の全ての同時実行情報を集めねばならない。このことは、GODの存在を有効なものとするだろう。GODがないならば、デッドロック検出は困難で高価なものとなろう。しかし、INGRESでは、GOD方式を用いていない。

犠牲の選出と、これの徹退（backout）は、トランザクションとそのスレーヴとを生成したINGRESプロセスへ回復（recovery）スキーマを適用することも含む。

⑤ 一致性制御

INGRESにおける一致性問題（consistency ploblem）として次のものがある。

i）INGRESは、データベースDがただ1人のデータベース管理者（DBA）を持つことを保障せねばならない。即ち、データベースDは、2つのサイトで異なったDBA を持てない。さらに、2人のDBAが同一名を持つデータベースを同時に生成しようとする同時実行問題も解決せねばならない。

この問題解決のために、CREATDB（Create Data Base）コマンドは、データベースを生成しようとする時刻を印したメッセージを各サイトへ送る。全てのサイトから ACK が返ってくれば、データベースを生成する。もし、他のサイトからよりはやい時間の同様のデータベース生成の意志を印したメッセージが届いたのなら、ユーザへエラーを返す。同時刻のものであったなら、どちらかのメッセージが任意にこわされる。

ii）視点1と2では、関係は、ただ1つのサイトにのみ存在することを保障せねばならない。視点3では、同一名の関係は、同一の構成であることを保障せねばならない。CREATE コマンドが実行されるたびごとに、他の全てのサイトへ新しい関係が生成されたことを通知せねばならない。

iii）破壊回復ソフトウェアは、一致性を持ったネットワークを再現せねばならない。特に、回復ソフトウェアは、システムカタログへの更新を含んでいる場合は困難である。

ネットワーク内での復旧は困難である。ある交信の処理中にホストのどれかが障害を起こすことがある。この障害は、ACKが返ってこないこと又はある時間内に処理されたことを示すDONEが返らないことによって検出出来る。

INGRESコマンドは、種々のホスト上での演算について完全な木構造を持っている。どれかの演算が障害を起こしたのなら、木の各節に当たる演算を徹退させながら、木の根までたどっていくことによって、コマンド全体を徹退出来る。

もし、システムカタログが冗長であるなら、他の問題が生じる。障害を起こしていたホストが復旧した時、冗長なシステムカタログは、ネットワーク内での一致性を保障しながら更新されねばならない。

iv）バックアップ・コピーも一致性が保障されねばならない。第一の解は、主コピーを持ったサイトを設けることである。これに対して、バックアップサイトに順序づけを行なう。更新は主コピーに対してなされる。バックアップサイトから主コピーサイトへは周期的に"Are you alive?"を送る。ACKが返らない時は、第1のバックアップサイトが主コピーサイトになっていく。この主サイトの切替時、いくらかの更新情報は失なわれ得る。しかし各バッ

クアップサイトは、主サイトが復旧ソフトウェアを実行しだした時間についての一致性は持っている。障害を起こしていたサイトが復旧した時、この時点でバックアップサイトと一致するデータベースを復旧する。これは障害を起こしていた間の全ての更新情報を必要とする。この後"I am alive"を全バックアップサイトへ発進し、ACKを全て受信した後、主サイトとなる。

第二の方法は、更新は先ず主コピーへ向けられ、その後この更新を全てのバックアップサイトへ送る。更新の正常終了のACKが全バックアップサイトから返ってくる時、更新は処理される。結果として、全てのコピーの同期をとることが出来る。この方法は、始めに述べたものに対して大きなオーバヘッドと遅延とをもつ。しかし、あるサイトからの検索要求は、そのサイトへ一番近いコピーに対して行なうことが出来る。これはアクセス要求の大半は検索要求であることから良いパフォーマンスをもたらす。冗長コピーを持たす理由の1つは高信頼性とともに高パフォーマンスの獲得である。この意味でINGRESは、この第二の方法がよいとしている。

3.3.2 SDD−1

SDD−1(System for Distributed Detabases)は、CCA(Computer Corporation of America)において設計開発中の分散型データベースシステムである〔ROTH77b〕。SDD−1は、データモデュール(datamodule)と呼ばれる要素から成り立っている。データモデュールは、種々(帯域、遅延)の通信回線によって互いに結合されており、これによって地理的に分散している。

SDD−1は、数百のデータベースを維持しようとしている。現在のところデータモデュールとしては、CCAにおいて開発されたデータコンピュータ〔MARI 75, FARR 76〕だけを考えているが、将来は種々のDBMSをサポートしようとしている。データコンピュータ(Datacomputer)は、3兆ビット以上へ拡張可能な大容量記憶と、コンピュータネットワークを通してアクセス可能な高水準なデータ管理機能とを持つ。又データコンピュータは、複数の異種コンピュータ上のプロセス間でのデータ共有を行なうための変換機能と柔軟なアクセス制御機能とを備えている。データコンピュータへの ットワークを通してのアクセスのためのインタフェースはデータ言語(Data language)と呼ばれる高水準言語である。

SDD−1は、INGRESと同様にデータベースの冗長性を強調している。冗長性とは、システムの信頼性と応答性(responsibility)とを高めるために、論理データ項目の全て又は一部分が複数サイトで格納されていることを意味している。データベースの冗長コピーの存在は、一方では多重コピーの更新の同期という効率における困難な問題を起こす。しかし、今後のデータベース利用は、更新よりも検索が圧倒的に多数であると考えられる〔ROTH77c〕ことより、利用度の高いサイトにデータ項目のコピーを備えることは有効である。

SDD−1の設計目的は、次の3点である〔ROTH 77b〕。

1）信頼性／長寿命性（survivability）

2）効率の調整可能性

3）モデュール拡張性

SDD－1は、ユーザへ対して関係モデルと、INGRESと同様なQUEL問合せ言語とを備えている。

他の大きな特徴は、不可視性（invisibility）〔ROTH 77c〕である。即ち、SDD－1内のデータの分散と冗長性をユーザは意識することなくデータをアクセス出来ることを大きな目標としている。不可視性は、汎用の分散型データベースにおいて第一に重要な概念であると考えられる。

A．アーキテクチャ

SDD－1は、データモデュールを要素として構成される。これらのデータモデュールは、互いに通信出来るように種々の通信回線によって結合されている。どのデータモデュールも、ある外部要求に対して同様に反応できるという意味で、関数的に同一である。しかし、データモデュールは種々のバラエティをもっている。例えば、あるデータモデュールは巨大記憶を持ち、コンピュータネットワーク内のデータ格納機能を司さどることが出来る。又、あるデータモデュールは、小規模記憶を持ち、近傍ユーザへの即応性を持ったデータキャッシュとして動作することも出来る。

データモデュールは、更にLDMとGDMとの2つの要素により構成される〔図3－13〕。

LDM（Local Data Manager）は、データモデュール内に存在するデータを管理するものである。LDMは、データの分散について関知せず、GDMからの要求へ対応して、検索、修正等の格納機能を行なう。LDMは、現在の典型的なDBMSの持つデータ操作機能に対応したレベルにある。現在のところLDMは、データコンピュータに対するデータ言語へGDMの要求を変換する。LDMが他の異種DBMSを扱かえるようにするための検討もすすめられている。

GDM（Global Data Manager）は、データ格納のための記憶を持たず、SDD－1内の分散に関する問題を処理する。GDMの機能としては次のものがある。

1）ユーザの要求を内部形式へ変換する。内部形式は、どの演算が実行され、どのデータ上で演算されるかを示している。

2）必要なデータモデュールの所在を明らかにする。ディレクトリ情報を用いて、要求されたデータを格納しているデータモデュールの決定を行なう。

3）コンピュータネットワーク内のLDMによって実行されるべき演算を決定する。この決定は、アクセス効率とデータベース一致性とを考慮してなされる。

データモデュール・アーキテクチャとしてLDMとGDMとの2つを考えた理由は、データ の分散に関る問題と、DBMSの変化、多様性等の個々のDBMS固有問題（異種性問題）との分離

である。LDMは、既存のDBMSの多様性へ対処し、GDMはこの様なDBMSの異種性とは独立に分散問題を処理させる。このことによって、種々のシステム要素（DBMS等）の変化へ対して、システム自体の定常性と長寿命性を獲得しようとするものである。

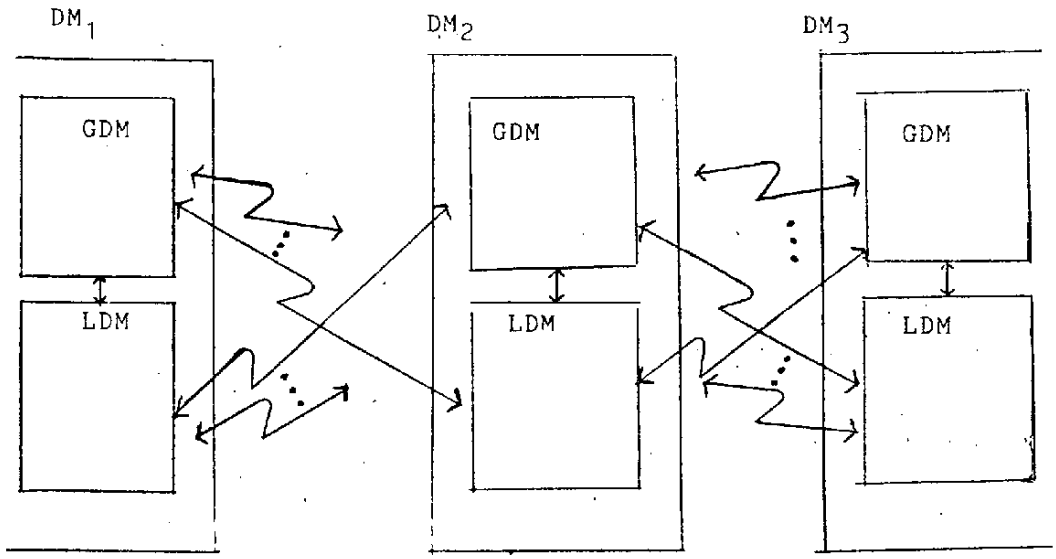


図3－13　データモデュール・アーキテクチャ

B．データコンピュータ〔MARI 75，FARR 76〕

データコンピュータは、ユーザへ対してコンピュータネットワークを通しての大規模データ管理と格納ユーティリティとを提供するものである〔MARI 75〕。データコンピュータ システムは、1兆ビット（10^{12} bits）の大容量記憶の経済的な利用とともに、複数の異種コンピュータ上のプロセス間でのデータ共有のために必要な変換機能と柔軟なアクセス制御とを備えている〔FARR 76b〕。データコンピュータは、ARPAネットワークでの過去4年間の使用によって12台のコンピュータ（1976年）上の数百のユーザによって使用されている。SDD－1，SRIのLADDERもデータコンピュータを利用している。ユーザへ対する記憶コストも512語（36ビット／語）当たり月額1セントと見積もられている。ちなみに、通常のTSS サービスでは同じデータ量へ対して月額25セントである。〔FARR 76〕

データコンピュータは、データ管理システムを備えている。これは、ARPAネットワークのホスト・プロトコルを用いてアクセス可能である。データは、ネットワーク上の全てのコンピュータ間で共有可能である。異種コンピュータが同一情報を利用出来る様に、異なった形式と機械表現の間の変換のための機能を持っている。柔軟なアクセス制御は、データベース管理者（DBA）が、正しい利用方法を記述し、不正アクセスからの防止を行なうことを可能としている。

データ管理機能は、データ言語と呼ばれる高水準言語で表わされたコマンドと問合せに対応し

て実行される。即ち、データコンピュータへのアクセスはこのデータ言語を介して行なわれる。変換機能とデータ管理機能とにおいて、データコンピュータ（場所、サイズ、メーカ、アーキテクチャが異なっているコンピュータ）上のユーザをサービス出来る。

次に、データ言語〔MARI 75，FARR 76〕について考える。データ言語の機能としては、データの記述、データベース生成と保守、データの選択検索、種々の予備機能とサービスへのアクセスがある。データ言語は、ユーザがデータの格納されている物理機器とは独立な観点でデータを表現出来る高水準言語である。ユーザは、物理機器に依存した検索とスケジューリング技術を何等知る必要はない。

ユーザは、自分のコンピュータにおけるデータ表現とデータ構造とを、データコンピュータのものへ写像出来ねばならない。即ち、各コンピュータはデータ言語を用いてデータコンピュータをアクセスする。

データ言語の基本特徴は、全てのデータを記述できる点である。この記述は、データコンピュータ・ディレクトリへ格納され、ユーザプログラムによって利用される。記述は、データの表現と構造とに関する情報を持っている。

I／Oトランザクションは次の2つの記述を必要とする。

1） ファイル記述

2） ポート記述

ファイル記述はデータコンピュータ内に格納されるデータに関係している。ファイル記述によって、データ管理者は、このデータがデータコンピュータ上でどのような形式で格納されるかの制御を行なえる。DBAは、最もひんぱんに使用されるアクセス方法へ対応したデータの表現を選択出来る。この様に、再形式化のために必要な計算を最少化し、高パフォーマンスが達成される。ポート記述は、ネットワークインタフェースを通って、データコンピュータを出入りするデータへ対するものである。ポート記述によって、ユーザは、自分のコンピュータ上で視えるデータがいかに形式化されるかを制御出来る。

ポートとファイル記述の間には必要な対応はない。即ち1つのポートは多くのファイルへ対して使用出来るし、同時にいくつかのファイルからデータを要求することも出来る。異なったポート記述をファイルの副集合又はファイルの集合を選択するために使用することも出来る。異種（語長や文字コード）のコンピュータは一意な方法で同一データをアクセスするためのポートを用いることもある。

データコンピュータ内のデータの論理構造は階層型である。データ型としては、LIST，STRUCTURE，STRING，INTEGER，BYTE，及びLISTとSTRUCTUREを組み合わせたものとの6つある。

図3－14は、データ記述の例である。

```
CREATE DATA.BOOKS FILE LIST (,500,10000)
    BOOK STRUCTURE
        TC INTEGER
        TITLE STRING (,50,500),C=TC,I=D
        SUBTITLES LIST (,1,10),C=1
            SUBTITLE STRING (,100),C=1
        AUTHORS LIST (,1,10),C=1
            AUTHOR STRING (,60),C=1
        PUBLISHER STRING (,30,100),C=1
        DATEPUBLISHED INTEGER(4)
        PAGES INTEGER (,3,5),C=1
        SUBJECTS LIST (,1,10),C=1
            SUBJECT STRING (,25),C=1,I=I
        KEYAUTHORS LIST (,1,5),C=1
            AUTHOR STRING (,?5),C=1,I=I
    END;

CREATE DATA.IMAGEPORT PORT LIKE BOOKS;

CREATE DATA.ASCIIPORT PORT LIST,P=EOF
    BOOK STRUCTURE
        TITLE STRING (,132),P=EOR
        SUBTITLES LIST (,10),P=EOB
            SUBTITLE STRING (,100),P=EOR
        AUTHORS LIST (,10),P=EOB
            AUTHOR STRING (,60),P=EOR
        PUBLISHER STRING (,50),D=' '
        DATEPUBLISHED INTEGER (4),D=','
        PAGES INTEGER (,5),D=' '
        FILL STRING (2),F='P',P=EOR
        SUBJECTS LIST (,10),P=EOB
            SUBJECT STRING (,25),P=EOR
    END;

CREATE DATA.TITLEPORT PORT LIST, P=EOF
    BOOK STRUCTURE
        TITLE STRING (,132),P=EOR
        SUBTITLES LIST (,10),P=EOB
            SUBTITLE STRING (,100),P=EOR
    END;

CREATE DATA.AUTHORPORT PORT LIST,P=EOF
    BOOK STRUCTURE
        AUTHOR STRING (,25),P=EOR
    END;
```

図３－１４　データ言語によるデータ記述

データ言語には、図３－１５に示すようなデータの管理にも利用できる。管理者としてディレクトリ構造を生成し、特権を定義し、リソースを割り当て、利用状況を管理することがある。主に、データコンピュータのディレクトリへ関連している。ディレクトリは木構造であり、アクセス制御と割り当て情報は木の全ての階層の節へ与えられる。

もう１つのデータ言語利用として、データ操作がある（図３－１６）。データ操作言語は、転置情報とデータ転送のためのネットワーク結合を明らかにし、データの検索手順を明らかにしている。

```
LOGIN CCA.JF; CREATE LIBRARY,M=25;
CREATEP LIBRARY,H=31,S=13303816,G=CLRWA;
CREATEP LIBRARY,D=RWA;

LOGIN LIBRARY; CREATE LOCAL; CREATE FOREIGN; CREATE DATA;
CREATEP LOCAL,H=31,G=L; CREATEP FOREIGN,P='READMORE',G=L;
CREATEP DATA,U=CCA.JF.LIBRARY,G=RWA;
CREATEP DATA,U=CCA.JF.LIBRARY.LOCAL,G=RA;
CREATEP DATA,U=CCA.JF.LIBRARY.FOREIGN,G=R;
CREATEP DATA,D=RWA;
```

図３－１５　データ言語の管理的利用

```
MODE BOOKS WRITE DEFER;

CONNECT IMAGEPORT TO 13303820;

BOOKS=IMAGEPORT;

FOR QUERY IN AUTHORPORT
    BEGIN DECLARE I INTEGER  I=0
        FOR TITLEPORT, BOOKS WITH ANY KEYAUTHORS
                                  WITH AUTHOR = QUERY.AUTHOR
            BEGIN
                BOOK=BOOK
                I=I+1
            END
        COMMENT
 QUERY.AUTHOR ! ' HAS ' ! I ! ' BOOK(S) IN THE LIBRARY.'
    END;
```

図３－１６　データ言語の操作的利用

C．データモデル及び言語

SDD－1における統一モデルは関係モデルである〔ROTH 77b〕。問合せ言語は、INGRES〔HELD 75，STON 77〕と同じくQUELである〔WONG 77〕。ユーザはQUEL 言語を用いて、データ項目の分散と冗長性と意識することなく、データベースをアクセス出来る。

D．分散問題

SDD－1における統一モデルは関係モデルである。SDD－1における分散問題は、INGRESにおける視点3へ対応する。論理レベルにおける関係の物理的なデータモデュールへの分割が、データの分散の基本概念である。この様な論理データ項目のデータモデュールへの割り当ての単位をフラグメントと呼ぶ。フラグメントとは、関係の一部である。即ち、フラグメントは関係の限定と射影として定義される。限定は、〔ESWA 76〕によって定義された次の様な単純述語形式である。

属性　＜関係演算子＞　：定数

ここで＜関係演算子＞：＝"＝"，"＜"，"＞"，"≦"，"≧"である。

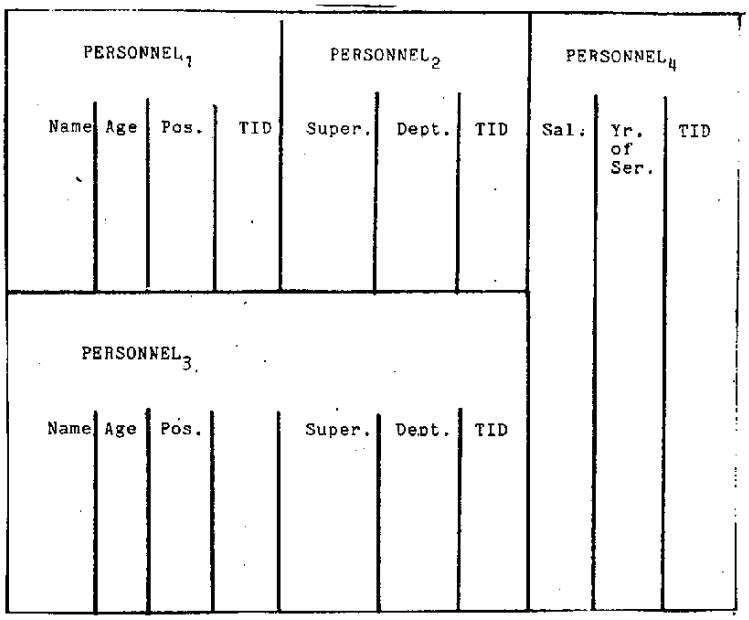
又、更新異常〔CHAM 75〕を回避するために、各々の射影は、主キーを持つことが必要とされている。これを実現する1つの方法は、システムで内部的に用いられるTID（tuple ID）〔STON 76，ASTR 76〕を全ての関係が持つことである。TIDは、各組が一意であることを保障している。即ち、主キーは、単にTIDであり得る。

図3－17に、関係PERSONNELのフラグメントへの分割を示す。フラグメントは、論理データ項目のデータモデュールへの割り当て単位である。又各々のフラグメントは、1つ以上のデータモデュールに冗長に格納され得る。あるデータモデュール内に格納されたフラグメントを格納フラグメント（stored－fragment）と呼ぶ。これを格納Fi，m（stored－Fi，m）と記す。これは、データモデュールm内のフラグメントFiの記述であることを意味している。図3－18は、論理データベースのフラグメントへの分割と、フラグメントのデータモデュールへの割り当てとを示している。四角は、フラグメントを表わし、丸はデータモデュールを表わす。

フラグメントは、冗長に格納され得ることは既に述べた。各データモデュールへ分散されたフラグメントから、完全で非冗長な関係を再構成するにはいくつかの組合せ方法がある。論理データベースの完全非冗長コピーを形成するフラグメントの1つの集合を具現（materialization）と呼ぶ。SDD－1は、ある時に、いくつかの具現を備えている（図3－19）。この場合、これらの具現を、設定具限（supported materialization）という。設定具現は、関係のフラグメントへの冗長分散形態を示している。SDD－1はログインしたユーザへある具現を割り当てる。ユーザの要求する検索は、このユーザへ割り当てられたこの具現内のフラグメントに対してのみなされる。具現は、ユーザがログインしている間は変わらず、静的にユーザへ割当てられている。具現の割り当ては、ユーザの検索がこれに基づいておこなわれるために、システムの効

率、一致性制御等に重要である。具現の割当て方法については、明らかにされていない。

設定具現概念は、SDD－1のデータ分散問題における重要な要素である。これは、データベース一致性に対して主要な役割を演じるとともに、冗長データの更新問題へ大きな影響を与えている。



各フラグメントは、次のように定義される。

- $PERSONNEL_1$:=
  PERSONNEL where Salary > $30,000,
  projected on Name, Age, Position, TID
- $PERSONNEL_2$:=
  PERSONNEL where Salary > $30,000,
  projected on Supervisor, Department, TID
- $PERSONNEL_3$:=
  PERSONNEL where Salary <= $30,000,
  projected on Name, Age, Position,
  Supervisor, Department, TID
- $PERSONNEL_4$:=
  PERSONNEL projected on Salary,
  Years-of-service, TID

図３－１７　関係PERSONNELの分割

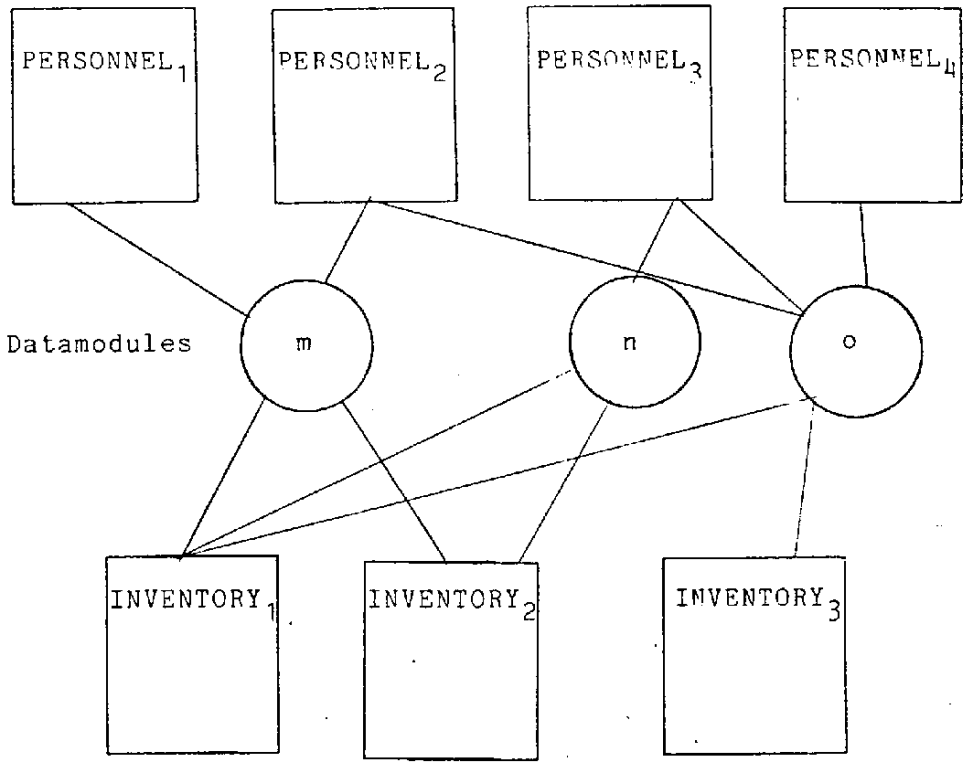


図３－１８　フラグメントのデータモデュールへの割当て

設定具現	PERSONNEL 関係 フラグメント				INVENTORY 関係 フラグメント		
	1	2	3	4	1	2	3
1	m	m	n	o	m	m	o
2	m	m	o	o	o	n	o
3	m	o	n	o	m	m	o
4	m	o	n	o	n	n	o
5	m	o	o	o	m	m	o
6	m	o	o	o	n	n	o

図３－１９　フラグメント割り当て

E．データベースの分散に伴う必要機能

分散問題を処理するために必要となる機能として、SDD－１では､ディレクトリ管理、冗長コピーの更新、問合せ分割を検討している。

① ディレクトリ管理

全体システムディレクトリ（global system directory）は、GDMが次のことを行なうために必要な情報を保っている。

ⅰ）ユーザ要求の解析をする。
ⅱ）興味あるデータ所在を明らかにする。
ⅲ）アクセスと更新戦略を選択する。

ディレクトリの構成方法として次の様なものが考えられる。

ⅰ）1つの中央データモジュールに、ディレクトリを集中する。
ⅱ）各々のデータモデュールにディレクトリの完全コピーを置く。
ⅲ）ディレクトリを、データモデュールの集合上へ非冗長に分散させる。
ⅳ）ⅲ）に対して、ある程度の冗長さを持たせてディレクトリを分散させる。

ディレクトリ構成方法としてどれを選択するかは、コンピュータネットワーク内のトラフィック形態に依存している。

SDD－1では、ディレクトリを通常のユーザデータとして扱かっている。このことによって、ディレクトリ管理へ対しても、機密保護、保全性、同時実行制御の様な通常のユーザデータへ対する機能を用いることが出来る。ディレクトリを構成するデータは、先に述べた通常の関係の分割と同様にフラグメントへ分割される。フラグメントは、ネットワーク内で冗長性を持って分散される。この様にフラグメントが冗長分散している場合、ディレクトリがどこに存在しているかを明らかにするディレクトリが必要になる。この様なディレクトリが存在しなければ、どこからも所在情報を得られなくなってしまう。このため、全てのデータモデュールは、ディレクトリの各々のフラグメントが格納されている場所についての情報が維持されているデータモデュールに関するディレクトリ情報を格納している。このディレクトリ情報は小さなものであり、もっぱら検索されるだけである。SDD－1では、INGRESとは異なり、ディレクトリの部分冗長性も検討している。

② 冗長コピーの更新

論理データ項目の冗長コピーの更新は、分散型データベース・システムにおいて重要である。冗長コピーの更新において、データベースの一致性が保たれることと、更新の同期のための余分なオーバヘッドが生じないこととが必要である。

SDD－1におけるデータベース一致性は、具現（materialization）を用いて定められる。具現は、ユーザから視えるデータベースの非冗長な論理的表現であるので、具現内において一致性は強力に保たれる。伝統的な集中的データベースシステムと同様に各々の具現の内部一致性を保つためには、次の形式が強力に維持されねばならない。

i）各々のトランザクションは、常に、ある一致性を持った状態から、他の一致性を持った状態へ写像する。これは、確かさ（validity）の条件である〔ROSENK 77〕。

ii）内部一致性は、直列性（serializability）〔ESWA 76，ROSENK 77〕によって定義される。直列性とは、データベース上のトランザクションを同時実行させることによって生ずる結果は、これらのトランザクションを直列に、重複することなく遂次的に実行させた結果と同じであることを意味する。

一致性のこの形式は強力なものではあるが、この形式を保つことは通信コストと処理遅延上高価なものとなる。

このためSDD－1では、具現内ではなく、具現間相互の一致性をより弱い形式で保つだけで十分と考えている。具現間の一致性にとって、データベースコピーが時間とともに発散しさえしなければよい。コピーが、あらゆる瞬間に同一である必要はなく、同一の最終状態へ行きつけばよい。

冗長コピーの更新を制御するための最も簡単な方法は、あるトランザクションによって読み書きされているデータベースの部分をロックすることである。しかし分散型データベースの場合には、ロック情報をネットワーク全体へ伝播するための遅延が問題となる。

Thomas〔THOM 76b〕による方法は、投標プロトコル（voting protocol）を用いることによって、ロックのために必要となるデータモデュール間の通信コストを縮少するものである。投標プロトコルは、ロックの集合に対して全てのデータベースコピーの同意を受けるのではなく、コピーの過半数からの同意だけを必要とするものである。

Alsberg等による方法は、更新へ対する主サイトを持たすものである。全ての更新は、主サイトを通してなされ、全てのロックも論理的には主サイト内だけで起こるように見える〔ROTH77b，c〕。INGRESも、この主サイトを設ける方式を用いている。

SDD－1による方法〔ROTH 77a，b〕は、前者の2つの方法の改良である。SDD－1の方法は、前者に対して定性的に異なっており、更新トランザクションを検索トランザクションと同様の速さで処理させようとするものである。又、全データベースのロックの効率のただ単なる改良ではなく、可能な限り全ロックを避けようとするものである。この背景として、多くの場合、トランザクションが発生したデータモデュール内でだけ、ロックされる必要がある点をあげることが出来る。この様な場合、全ての冗長コピーへの更新情報は、モデュールごとにロックを要求するだけのプロトコルによって伝播される。

あるトランザクションにとって、全ロックが必要かどうかは、他のトランザクションが全ロックを必要としているかどうかによって決まる。この決定は、データベース設計時にDBAによってなされる。

DBAは、データベース・アプリケーション内で共通に実行されるトランザクションのクラスを定義する。DBAは、各々のクラスに対して、どのデータモデュールがそのクラスのトランザ

クションを入力しようとしているかを定める。トランザクションのある配置に対して、各クラス内のトランザクションにとって必要となる同期化の情報量を決定出来る〔ROTH 77〕。この結果を表わしているテーブルは、トランザクションにとって必要な適当な同期レベルを決定するためにデータモデュールによって利用される。

③ 問合わせの分割

分散型データベース・システムにおいて、複数サイトへまたがった関係に対する複雑な問合せを処理する必要がある。問題はこの様な問合せを処理するために必要となる全てのデータを1つの処理装置に集めねばならないために生ずる遅延である。

SDD−1による方法は、Wongの論文〔WONG 77〕に述べられている。この方法は、非分散型データベースにおける多数問合せの一変数問合せの最適な分割方法〔WONG 76〕に基づいている。〔WONG 77〕では、最適化の目標関数として、通信コストを用いて、分散型データベースへ対する拡張例を示している。

問合せ分割における問題点は、問合せが複数の関係へまたがってなされるために生ずる関係の大きさ(組数)の組合せ的増加(combinatorial growth)をいかに避けるかである。n個の関係へまたがった問合せを処理する場合、これらの関係の直積を作らねばならない。直積によって作られる関係の大きさは、n個の関係の大きさの和ではなく、通常相当大きなものになる。Wongによる分割アルゴリズム〔WONG 76〕は、ⅰ)直積を作らずに、ⅱ)検索する組数を可能な限り少なくすることを目的としている。ⅱ)は、分散型データベースにおいて、サイト間で転送される関係又は組を極少とすること、即ち通信コストを最少とすることに対応させられる。このアルゴリズムに基づく、分散型データベースによる例題は同じくWong〔WONG 77〕に示されている。

問合せ処理は、ⅰ)サイト間でのデータの転送と、ⅱ)転送されたデータのローカル・サイトでの処理とから成る。SDD−1は、冗長コピーの存在を許すが、Wongは、1つの具現(materialization)を考えることにより、分散してはいるが非冗長なデータベースを検討している。先ず分散型データベースの視点として、次の3つを考える。

V1)ユーザの視点── データモデルと分散情報とから成る。

V2)分散視点(distribution view)── 関係はサイト当たり1つとした場合の関係の集合。各々の関係は、あるサイトにある全部の関係の直積である。

V3)ローカル視点(local view)── サイト当たりの1つのローカル視点が存在する。ローカル視点は、サイトに存在する全関係を含んでいる。

問合せ処理は、上記の3視点を用いて、表3−8のような5段階に分けることが出来る。この表内の分散(distribution)は、多変数問合せを関係の所在を意識しながら、一連の1変数問合せへの分割を意味している。分散処理によって生成される問合せ(一変数V_2 問合せ)

は、各サイト単位の問合せである。このレベルでの変数はサイトである。〔WONG 77〕は、問合せの分散処理を扱かっている。分散の処理は、次の3つの基本段階から成っている。

表 3 － 8　問 合 せ 処 理 過 程

処　理　名	概　　要
変換（conversion）	V_1 問合せ　→　一変数V_2 問合せ
分散（distribution）	多変数V_2 問合せ　→　一変数V_2 問合せ
再変換（reconversion）	一変数V_2 問合せ　→　多変数V_3 問合せ
分割（decomposition）	多変数V_3 問合せ　→　一変数V_3 問合せ
ローカル一変数問合せ処理	分散問題とは独立な問合せ処理

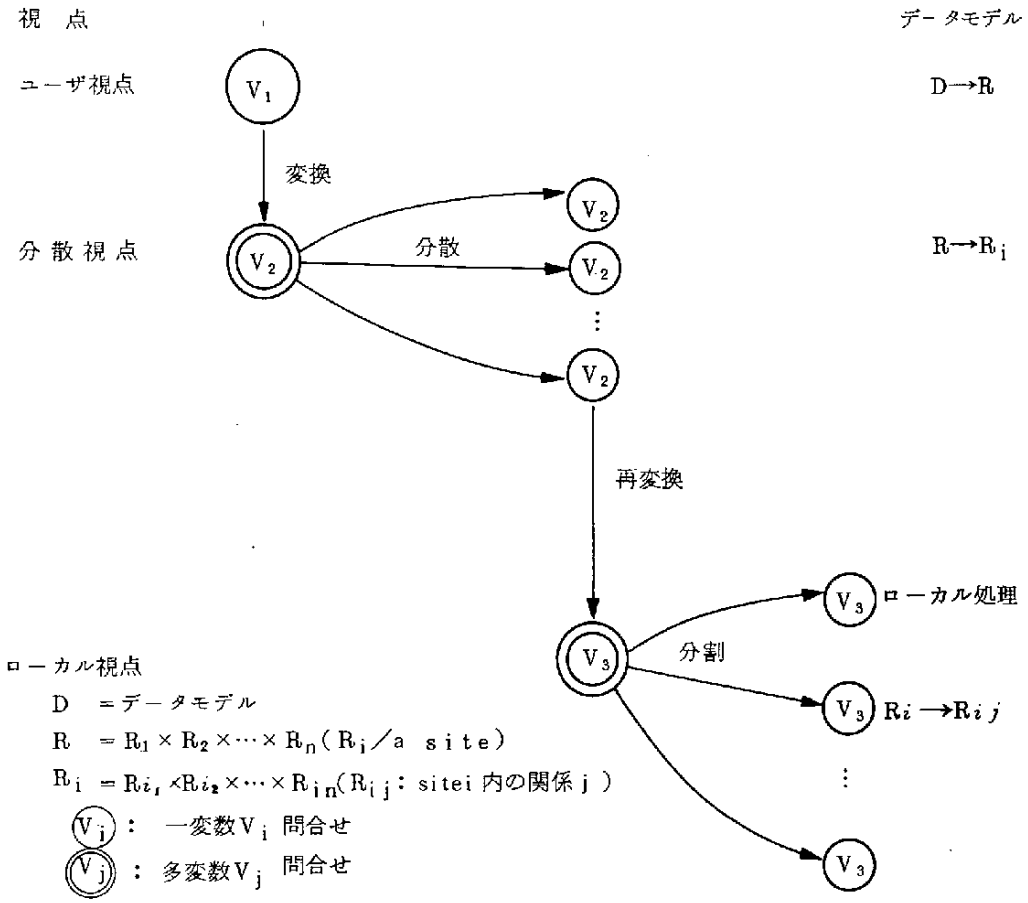


図 3 － 2 0　問合せ処理概要

ⅰ）1変数副問合せ（one-variable subquery）；1つのサイト内で行なえる演算をまず第一に実行する。

ⅱ）縮少（reduction）；ただ1つだけの重複変数を持った2つの部分へ問合せを分割出来るなら、分割する。

ⅲ）組置換（tuple substitution）；変数の1つを、変数の値域関係内の実際の組によって置き換える。これは、サイト間で、必要な都度V_2 組の転送を必要とする。組単位（tuple-at-a-time）の転送よりも、関係単位の様な大量データ転送の方が効率的であるために、WONGは組よりは関係を転送するようにしている。

通信コストは、ローカル処理よりも高価であるので、分散処理の最適化は、主にその転送方法に依存しており、ローカル処理の最適化とは独立である。よって問合せの分割においては、通信コストに基づく最適化を行なう。

次に、問合せ分割問題を例3.1のような例について考える。

例3.1

RELA（SS#, SCity, VS#, VP#, VQOH）（サイトA）
RELB（PP#, PPname）（サイトB）
RELC（JJ#, JCity, YJ#, YS#, YP#, YQty）（サイトC）
} V_2視点

```
RANGE OF (A, B, C) IS (RELA, RELB, RELC)
RETRIEVE (A, SS#)
WHERE (A.SS#=A.VS#) AND (A.SS#=C.YS#)
  AND (A.VS#=C.YS#) AND (A.VP#=B.PP#)
  AND (B.PP#=C.YP#) AND (A.VP#=C.YP#)
  AND (A.VQOH>C.YQty) AND (C.YQty>1000)
  AND (A.VQOH>1000) AND (A.SCity=C.JCity)
  AND (B.PPname='Bolts') AND (C.JJ#=C.YJ#)
```

QUEL問合せの条件項のなかで、C.YQty＞1000，B.PPname＝'Bolts'のような限定は1サイト内で閉じており1サイト内で処理できる。従って、このようなサイト内で閉じている限定を除くと、サイト間で関連のある属性は例3.2のようになる。

例3.2

サイト	属性
A	(A.SS#. A SCity. A.VS#. A.VQOH)
B	(B.PP#)
C	(C.YS#. C.YP#. C.YQty. C.JCity)

これより生成される問合せを例3.3に示す。この問合せは、サイト間で関連のあるものである。

例3.3

```
RETRIEVE(A. SS#)
WHERE(A. SS#=C. YS#) AND (A. VS#=C. YS#)
  AND(A. VP#=B. PP#) AND (B. PP#=C. YP#)
  AND(A. VP#=C. YP#) AND (A. VQOH>C. YQty)
  AND(A. SCity=C. JCity)
```

1サイト当りの1つの関係をフラグメントへ分けると例3.4のようになる。

例3.4

サイト	フラグメント	フラグメント間関係
A	ASupplier(A.SS#, A. SCity) AAvailability(A. VS#, A. VP#, A. VQOH)	WHERE (A. SS#=A. VS#) AND(A. VQOH>1000)
B	BParts(B. P#)	WHERE (B.PPname='Bolts')
C	CSupply(C. YS#, C. YP#, C. YJ#, C. YQty) CProject(C. JJ#, C. JCity)	WHERE (C. YQty>1000) AND(C. JJ#=C. YJ#)

次に、これらの分散された関係(フラグメント)を、どこかのサイトへ集めて処理する場合、最少の転送量がどれかを決定する。

まず、1つのサイトで全処理を行なうとした場合、どのフラグメントを移動させねばならないかを例3.5に示す。

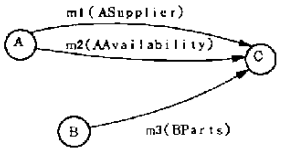
例3.5

サイト	転送されるべきフラグメント
A	BParts, CSupply, Cproject
B	ASupplier, AAvailability, CSupply, CProject
C	ASupplier, AAvailability, BParts

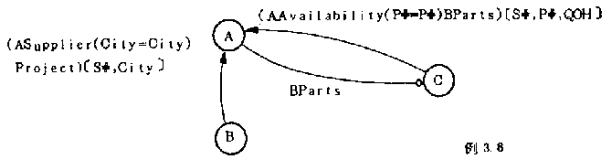
Bは転送フラグメント数が多いために考えられず、CSupplyは大容量であるために転送されないことが望ましい。このことより転送の初期設定M(Q)を例3.6の様に定める。

例3.6 M(Q)

$$M(Q)=\left\{\begin{array}{l} m1="\text{move ASupplier from A to C}" \\ m2="\text{move AAvailaility from A to C}" \\ m3="\text{move BParts from B to C}" \end{array}\right\}$$



例3.7



例3.8

M(Q)をもとに、発見法を用いて最適な転送パターンを求めていく。例えばM(Q)の転送の替わりに、P#について射影(AAvailability(p#=p#)BParts)[S#, p#, QOH]をCからAへ送り、Cityについての射影(ASupplier(City=City)CRroject)[s#, City]をBからAへ送ることも出来る〔例3.8〕。この様に、まずM(Q)として、フラグメント単位の転送を初期設定として考える。次にいくつかのノードで処理を分担することにより、より小さなフラグメントの一部を転送させるようにするものである。1つの転送パターンから、いくつかの新たな代替パターンが考えられてくるが、どちらが最適かは転送コストによって決定される。Wongによる最適化は、ネットワークのサイト間での転送コストを極少とすることによって発見法的に行なわれる。通信コストが問合せ処理コストへとって支配的である点は、Wongのアルゴリズムの主要な前提である。

3.3.3 LADDER

LADDER(Language Access to Distributed Data with Error Recovery)は、SRIにおいて開発された米国海軍のための分散型データベース・システムである。LADDERの目的は一般ユーザに対して、種々のDBMSの管理下で複数のコンピュータ上に格納された情報への容易なアクセスを提供することである。LADDERは現在PDP−10(KL−10)上にインプリメントされており、ARPAネットワーク上のデータコンピュータをデータベースとして持っている。LADDERは、海軍の情報(船等)に関する14個の関係からなる関係データベースをサポートしている。

A．アーキテクチャ

LADDERは、図3−2−1に示す様に、INLAND，IDA，FAMの独立した3つの要素から成る。INLANDは、制限された英語形式の問合せを入力として、高水準な形式的問合せを生成する。INLANDでは、情報構造については何の知識も持たず、データの項目名(field，関係モデルの属性名)の知識を持つだけである。IDAは、項目名だけの問合せへ、情報構造を付加して、実際のDBMSへの問合せのプログラムを生成する。重複ファイルに対して一般ファイル名によ

ってアクセスを行なう。また、問合せの結果を、ユーザの要求に合うように合成する。現在関係モデルだけを対象としているが、他のＤＢＭＳのデータモデルの構造も、ここで付加されると考えられる。FAMは、一般DBMS問合せ("generic"DBMS query)を、コンピュータネットワーク上のファイルの所在を明らかにし、ネットワークを通して問合せを送出する。FAMへの入力の"一般" DBMS

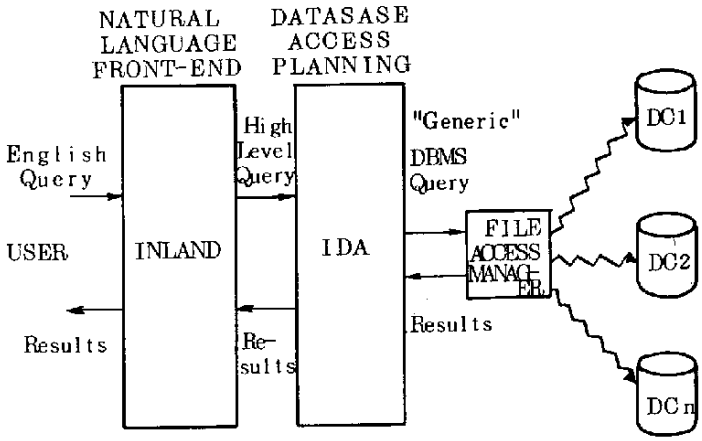


図３－２１　LADDERの全体構成

問合せと、出力のDBMS問合せは、データ言語を用いている。"一般" DBMS 問合せの"一般"の意味はデータ言語のファイル名として一般ファイル名を使用していることである。現在LADDERが扱かえるDBMSは、データコンピュータだけであるが、他のDBMSへ対しても検討がすすめられている。

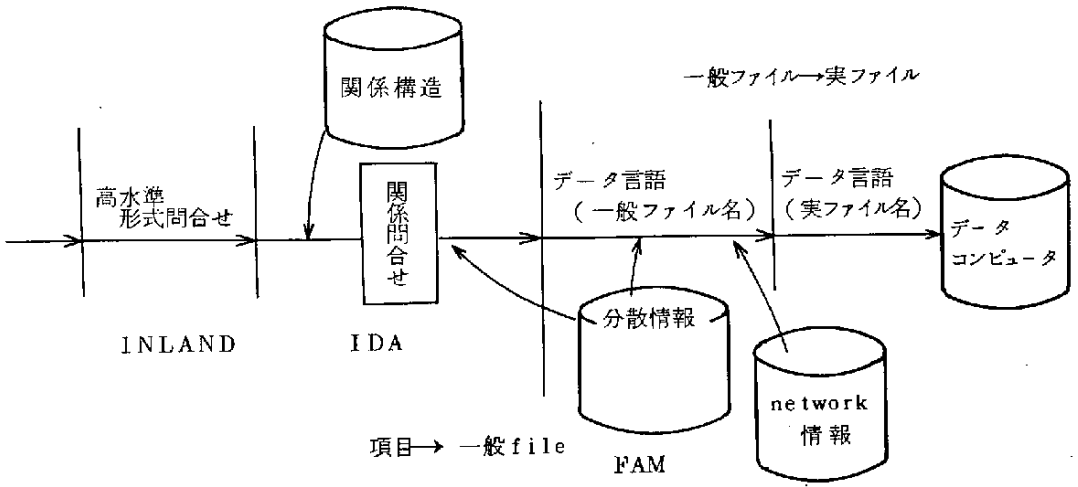


図３－２２　LADDER内の変換

B. INLAND

INLAND（Information Natural Language Access to Navy Data）〔HEND76b〕は、LADDERにおける第一番目の要素であり、コンピュータネットワーク上に分散したデータに対して自然言語に近い形でのアクセスを提供する知的インタフェースである。INLANDは、LIFFER（Language Interface Facility with Ellipsis and Recursion）〔HEND77a，c〕と呼ばれる自然言語インタフェースを用いている。LIFFERは、他のコンピュータソフトウェアへの自然言語インタフェース生成機能を持っている。INLANDは、この様な自然言語問合せを入力として、ネットワーク上に分散したデータベース全体に対するひとかたまりの高水準形式的問合せ群を生成する。生成される問合せは、必要なデータ項目名だけの情報を持ち、項目の構成方法、即ち情報構造に関する情報を何等持たない。例えば"What is the length of the Kennedy？"という自然言語問合せへ対して、INLANDは、形式的問合せ

((NAM EQ JOHN#F. KENNEDY)(? LENGTH))

を生成する。ここでLENGTHは、長さを示す項目名であり、NAMは船名項目名であり、JOHN#F. KENNEDYはKennedyと関連したレコードのNAM項目の値である。

INLANDは、ユーザへ容易なデータベース・アクセス手段を提供する外部インタフェースと考えられる。分散型データベースにおいて重要となるデータモデル、分散問題、異種性問題は、IDA，FAMのレベルで問題となる。

C. IDA

IDA（Intelligent Data Access）〔SAGA77〕は、データベース内のデータ項目名だけの情報を用いて、一般ユーザにデータベースから情報をデータベース構造とは独立に検索させるために開発され、LADDERにおける第2の要素となっている。IDAは、項目名だけの問合せからデータベースの構造を考慮して実際のDBMS問合せを生成する。生成される問合せは"一般"DBMS問合せ（"generic" DBMS query）と呼ばれ、CCAにおいて開発されたデータコンピュータ〔MARI75〕のアクセス言語，データ言語（Datalanguage）である。IDAにおいて、データの存在するファイルは一般ファイル名（generic file name）として参照される。1つの一般ファイル名は、ファイルの冗長コピー全体の総称であり実際のファイルの所在情報は付加されない。所在情報は、後のFAMによって明らかになる。IDAは、ユーザの問合せから一般DBMS問合せを生成するとともに、この一般DBMS問合せの結果からユーザの目的に合うような結果の生成も行なう。現在IDAとFAMは、データコンピュータ用に開発されているが、近いうちに他のDBMSへの拡張が計画されている。

先ず、IDAの入力となる高水準問合せについて考える。先に述べたように、この問合せには、データベースの構造（データモデル）に関する情報は含まれず、次の様な項目（field）に関する情報だけである。

ⅰ）項目についての制限リスト（例　NAM EQ TOHN#F. KENNEDY ）

ⅱ）項目の値についての要求（例　？ PRESIDENT）

ⅲ）MAX，MIN，COUNT等の演算機能（例　*MAX WINNER-VOTES ）

例えば，"１９００年と１９４８年の間に選出された全ての大統領を表にせよ"という要求は、

（？ PRESIDENT）（（ELECTION-YEAR LE 1900）

AND （ELECTION—YEAR LE 1948））

という形の問合せとしてＩＤＡへわたされる。又"最大標を獲得した大統領は誰か？"は

（？ PRESIDENT）（*MAX WINNER—VOTES）となる。

次に、構造と独立なＩＤＡへの問合せから、データベース構造を考慮した問合せプログラム生成過程を考える。ＩＤＡは、構造スキーマ（ structural schema ）と呼ばれるデータベース構造のモデルがある。構造スキーマは、データベースのレコードと項目の編成を述べている。構造スキーマは、ⅰ）関係枠（ relation frame ）とⅱ）項目枠（ field frame）との2つの枠から成っている。関係枠は、ある関係と関連ある他の全ての関係との間の関係性を示している。項目枠は、ある項目が属している全ての関係のリストである。枠（ frame ）とは Winograd〔ＷＩＮＯ７５〕が言うような属性と値の対（この対をスロットとも呼ぶ）のリストであり、枠がモデル化している事象についてのある情報を備えている。枠を説明するために、ＡＣＭ Computing Survey〔March 1976〕による次の様な大統領関係を用いる。

PRESIDENTS （PRESIDENT, HOME-STATE, PARTY）

ELECTIONS （ELECTION-YEAR, PRESIDENT,

WINNER-VOTES OPPONENT, LOSER-VOTES）

ELECTION-STATE （ELECTION-YEAR, STATE, CANDIDATE,

VOTENUM）

関係枠は、データベース内の実際の関係に対応し、関係間の関係性についての情報を備えている。即ち、２つの関係の可能な結合（ join ）を定めている。関係ＰＲＥＳＩＤＥＮＴＳの関係枠は、

〔①ELECTIONS （PRESIDENT）

②ELECTION-STATE （（ELECTIONS））〕である

各々のスロット（①と②）は、ⅰ）関係名、ⅱ）その関係とのリンク又は、関接リンクがある場合は第３者関係の名とから成る。第１スロット（①）は、関係ＰＲＥＳＩＤＥＮＴＳと関係ＥＬＥＣＴＩＯＮＳの結合は、ＰＲＥＳＩＤＥＮＴ項目（属性）についてなされねばならないことを示している。第２スロット（②）は、関係ＰＲＥＳＩＤＥＮＴＳと関係ＥＬＥＣＴＩＯＮ-ＳＴＡＴＥとは直接結合は出来ず、関係ＥＬＥＣＴＩＯＮＳを含めた３つの関係の結合を行なわねばならないことを示している。

項目枠は、ユーザ問合せ内の項目名に対応し、各々の項目が属する関係のリストである。例え

ば、ＰＲＥＳＩＤＥＮＴ項目へ対する項目枠は

〔RELATIONS （PRESIDENTS ELECTIONS ELECTION-STATE）①
ALIAS-IN-ELECTION-STATE CANDIDATE〕②となる。

第１スロット（①）は、PRESIDENT項目は、関係PRESIDENTS，ELECTIONS，ELECTIONS-STATE 内に表われることを示している。第２スロット（②）は、ＰＲＥＳＩＤＥＮＴ項目は、関係ＥＬＥＣＴＩＯＮ－ＳＴＡＴＥ内ではＣＡＮＤＩＤＡＴＥという項目名を持つことを示している。更に、〔ＷＩＮＯ７５〕にあるように、項目名へ、必要となる処理をも付けることが出来る。例えば、ＴＯＴＡＬ－ＶＯＴＥＳ項目に、各々の州内の得標数を見つけ、それを加算するためにＩＤＡを再帰的に呼出す処理を付加することが出来る。

上述した関係枠と項目枠とを用いて、項目名だけから成るユーザ問合せから、実際の関係への問合せを生成出来る。これは、問合せ内の全ての項目を被う関係の数が最少となるようにする最小被覆問題（minimum covering ）である。この時関係枠は、節を関係、節間の結合線は２つの関係のリンクとするグラフによって表わされる。最小被覆問題は、問合せ内の項目を全てをおう最少の結合副グラフ（即ち、節数が最少であるもの）を見つけることである。ＩＤＡでは、検出のために発見法を用いている。常に良い解を見つけられるが、問題は、項目と関係が多くなった場合の検索時間である。ＩＤＡにおける発見法は次のようである。まず、すでに選択された関係のリスト（初期状態ではこのリストは空である）と、まだ被覆されていない項目リストがあるとする。無作為に、１つのまだ被覆されていない項目を取り出し、

ⅰ）この項目を含み、
ⅱ）すでに選択された関係（もし存在するなら）と直接リンクを持ち、
ⅲ）可能な限り多くのまだ被覆されていない項目を被覆する様な関係を見つけることを試みる。

現在のＩＤＡでは、この手法は、アクセスされるべき関係数をコスト関数として極小化することを試みる。最適化は、関係が種々のコンピューターに存在する場合等で異なってくるが、ほぼ最適のものを得れる。ＩＤＡ問合せのＤＢＭＳ問合せへの変換例を考える。"カリフォルニアから選出された最後の大統領は、何年に選出されたか？"という問合せは、ＩＤＡ問合せとして次の様に表わせる。

（？ELECTION-YEAR）（*MAX ELECTION-YEAR）
（HOME-STATE EQ 'CALIFORNIA'） (1)

生成される問合せプログラムは関係ＰＲＥＳＩＤＥＮＴＳに対して

（HOME-STATE EQ 'CALIFORNIA'）（?PRESIDENT） (2)

関係ＥＬＥＣＴＩＯＮＳに対しては（*MAX ELECTION-YEAR）
（？ELECTION-YEAR）
（（PRESIDENT EQ---） OR---）①となる。 (3)

(3)内の①部には、(2)の問合せ結果がはいる。ＩＤＡは、ＩＤＡ問合せ(1)内の項目が属している関係を、項目枠を用いて決定する。ＰＲＥＳＩＤＥＮＴＳとＥＬＥＣＴＩＯＮＳとの２つの関係が必要であることがわかる。次に、関係ＥＬＥＣＴＩＯＮＳ内の検索を限定するために、関係PRESIＤＥＮＴＳからこのような情報が必要かを決定する。２つの関係のリンクは、ＰＲＥＳＩＤＥＮＴ項目であり、(2)の問合せ結果、即ちカリフォルニア選出の大統領名を(3)の問合せへわたし必要な結果を得る。

ＩＤＡについて、項目名だけの問合せから　関係モデルを考慮した関係ＤＢＭＳへの問合せ生成過程をみてきた。ＩＤＡにおける問題点を以下に指摘する。第１に、Carlson〔ＣＡＲＬ７６〕とRoussopoulos等〔ＲＯＵＳ７５〕によって指摘された多重パス問題がある。非手続的問合せ（ＩＡＤ問合せ）から、実際ＤＢＭＳへの問合せの生成は、各関係単位への問合せからなる手続的なプログラム（アクセスパス）の生成である。多重パス問題とは、ある問合せに対して異なった結果を生成してしまう複数のアクセスパスが存在することである〔ＣＡＲＬ７６〕。この例として２つの関係間に複数のリンク（直接又は間接）がある場合をあげることができる。この問題の解決には、データの意味の理解が必要となる。第２章で指摘したようにデータベースの意味記述の困難さは関係モデルの主要な問題点である。

次の問題は、異種ＤＢＭＳの取扱かいをどの様にするかである。ＩＤＡは、構造情報なしの項目名情報を持った問合せを、情報構造を付加して関係単位への問合せを生成する。次にこの問合せを、データコンピュータ用のデータ言語へ変換する。データコンピュータは関係ＤＢＭＳではない。"他の"ＤＢＭＳと言う場合、データコンピュータが他のＤＢＭＳ（例えばＣＯＤＡＳＡＬ－ＤＢＴＧ）へ変わるとともに、ＩＤＡもＣＯＤＡＳＹＬ－ＤＢＴＧ用に書き替える必要があることを述べている。この意味でＩＤＡはデータモデルを持たない標準問合せ言語をユーザへ提供するとともに、異種性の問題（即データモデルの付与）はＩＤＡで行なっていることになる。項目名だけの問合せから、関係モデルの問合せは生成され得ることは示されているが、他のデータモデル（網型、階層型）のものが生成されるかは問題が多い。関係問合せからデータ言語生成に関しては何も述べられていない。データコンピュータのＤＢＭＳは関係モデルを用いてはいない。各データベースの異種性をどこで処理し、分散性をどこで処理するかＩＤＡでは明らかになっていない。

ＩＤＡは、データ項目名だけの情報を持った問合せから、データコンピュータ用のデータ言語を生成する。この場合、項目又は関係の存在するファイルとの対応づけのためにディレクトリが必要になる。分散の問題である。この点については何も触れられていないが、ＩＤＡにおいて項目の所在情報を持ったディレクトリが必要となる。

D　FAM

ＦＡＭ（File Access Manager）〔MORR７７〕の目的は、データベースに関する非本質的事項からユーザとプログラムを解放するとともに、分散型データベースへの信頼性の高いアクセスを提供することである。ＦＡＭのユーザ（ＩＤＡも含む）は、一般ファイル名（generic

file name ）によってファイルを指定出来る。一般ファイル名は、主ファイルとコピーファイルに対する総称である。ＦＡＭの主要な役目は、ファイルの所在を明らかにし、実際のネットワークのデータベース・サイトへ問合せを発進することである。現在ＦＡＭはＣＣＡにおいて開発されたデータコンピュータのみをサポートしているだけであるが、他のＤＢＭＳへの拡張も検討されている。ＦＡＭは次の機能を持つ。

ⅰ）ＡＲＰＡネットワーク上の種々のホスト上の種々のＤＢＭＳと結合（ connection)を確立する。

ⅱ）種々のファイル内から最新のものを見つける。

ⅲ）ＩＤＡによる一般ファイル名を実ファイル名で置き換えた実ＤＢＭＳ問合せを生成し送出する。

ⅳ）あるタイプの障害からの回復を行なう。

Ｅ　まとめ

ＬＡＤＤＥＲにおける統一モデルは、ＩＤＡとユーザ（ＩＮＬＡＮＤ）とのインタフェースに存在しているとみれる。ユーザはデータはデータ項目名だけを用いてデータベースへアクセス出来る。

ＩＤＡは、この様な問合せに対して情報構造（項目がどの様にレコード又は関係へ編成されるか）を導入する。構造を備えた問合せをデータコンピュータ用のデータ言語へ変換する。この時、項目名又は関係名とこれらの所在するデータコンピュータ上のファイル名との対応が必要になる。これはデータモデルの変換問題である。よって、ＩＤＡは、統一モデルからローカルモデルへの変換（異種性問題）と、項目所在を明らかにする（分散の問題）との２つの機能を持っていなければならない。分散型データベースの問題、即ちデータモデル、分散問題と異種性問題はＩＤＡにあると考えられる。ＩＤＡについてのＳａｇａｌｏｗｉｃｚの論文〔ＳＡＧＡ７７〕には、分散と異種性問題は、明確に分離されて論じられてはいない。

ＦＡＭは、ファイルの冗長コピーの管理を行なうが、その役目は、主にコンピュータネットワークを通してのデータベースアクセス（ログロン／オフ等）の支援であると言える。

3.3.4　ＰＯＬＹＰＨＥＭＥ

ＰＯＬＹＰＨＥＭＥは、グルノーブル大学のＡｄｉｂａによって開発されている分散型データベースである〔ＡＤＩＢ７７〕。ＰＯＬＹＰＨＥＭＥは、種々の異種ＤＢＭＳ（ＣＯＤＡＳＹＬ－ＤＢＴＧ，ＩＭＳ，ＳＯＣＲＡＴＥ，ＳＹＳＴＥＭ２０００，ＴＯＴＡＬ，関係ＤＢＭＳ）をコンピュータネットワークを用いての共働を目指すものである。上記の様な種々の既存ＤＢＭＳを想定していることと、システム全体の統一モデルとしてＡｂｒｉａｌ のデータ意味論（ data semantics ）モデル（２項関係モデル）〔ＡＢＲＩ７４〕とを用いていることがＰＯＬＹＰＨＥＭＥの特徴である。ＰＯＬＹＰＨＥＭＥアプローチの背景には、ＡＲＰＡ〔ＲＯＢＥ７０〕，ＣＹＣＬＡＤＥＳ，ＴＲＡＮＳＰＡＣ，ＥＵＲＯＮＥＴ等のデータ通信機能を備えたコンピュータネットワークの発展

がある。このＰＯＬＹＰＨＥＭＥも、こうした既存ネットワークの基本通信手段に基づいている。

Ａｄｉｂａの論文〔ＡＤＩＢ７７〕は、データベースのデータ記述及びデータ操作問題に関している。彼の分散型データベースへのアプローチは、次の点を分散型データベースの問題として扱かっている。

1）ローカルデータベースの記述
2）ローカルデータベースの集合全体へ対する全体視点（ｇｌｏｂａｌ　ｖｉｅｗ）の記述
3）全体視点の利用。共通問合せ言語を設け、新しいアプリケーションはこれを用いる。
4）ローカルデータベース間の通信レベルの選択。通信レベルとしては次の4つが考えられる〔NAHO76〕。

ⅰ）問合せ言語レベル
ⅱ）親言語レベル
ⅲ）アクセスパスレベル
ⅳ）微視点

ＰＯＬＹＰＨＥＭＥでは、上記の問題1）と2）、即ちローカルデータベースの記述と全体視点の記述のために２項目関係モデルを用いている。Ｓｅｎｋｏも、ＤＩＡＭⅡ〔ＳＥＮＫ７６,７７〕の概念スキーマに対応する情報論理層のモデルとして２項目関係モデル〔事象集合モデルと呼ばれる〕を用いている。3）に関しては問合せの分割を議論している。通信レベルとしては個々のローカルＤＢＭＳの内部機構と独立とするために問合せ言語レベルとの２つについてのみ考える。

ＰＯＬＹＰＨＥＭＥにおける重要な特徴は、データベースの異種性の問題と、ネットワークを用いてデータベースを共働させるために生ずる分散の問題とを分離して扱かっている点である。まず個々のＤＢＭＳを共通データモデル（２項関係モデル）によって記述する。この段階にＤＢＭＳの異種性問題が表われる。次にこの記述されたローカルデータベースから全体的な視点を記述する。この段階に、データベースの分散の問題がある。このアプローチは、２つの問題を明確に分離したという意味で価値あるものと考える。

A．ローカルデータベースの記述

ローカルデータベースとはＤＢＭＳであり、あるローカルトランザクションを処理出来るオートマトンであるとする。ローカルデータベース間の情報交換はコンピュータネットワークを用いて行なわれる。ローカルデータベースは２項関係モデルを用いて記述される。

ローカルデータベースの記述要素として次の点を考えねばならない。

1）抽象カテゴリ名
2）有形カテゴリ名とキー
3）カテゴリ間のアクセス関係

4）逐次アクセスパス

各ＤＢＭＳにおける用語と、上記の概念への対応を表３－９に示す

表３－９　ＤＢＭＳ用語間の対応

COOPERATION SYSTEM TERMINOLOGY	TABLE 2.1 - CORRESPONDENCE BETWEEN DBMS TERMINOLOGY					
	I.M.S.	SOCRATE	CODASYL-DBTG	TOTAL	S2000	RELATIONAL MODEL
CONCRETE CATEGORY	ROOT SEGMENT DEPENDANT SEGMENT	ENTITY INCLUDED ENTITY	RECORD REPEATING GROUP	DATA RECORD	RECORD REPEATING GROUP	RELATION NAME
ABSTRACT CATEGORY	FIELD	CHARACTERISTIC	DATA-ITEM	DATA ELEMENT	ITEM	ATTRIBUTE
REL-ACCESS FROM CONCRETE CATEGORY TO ABSTRACT CATEGORY	SEGMENT TO EACH FIELD (1,1)	ENTITY TO EACH CHARACTERISTIC (1,1)	RECORD TO EACH DATA-ITEM (1,1)	DATA RECORD TO EACH DATA ELEMENT (1,1)	RECORD TO EACH ITEM (1,1)	PROJECTION
REL-ACCESS FROM A CONCRETE CATEGORY TO ANOTHER CONCRETE CATEGORY	SEGMENT HIERARCHY (0,N)	1) HIERARCHY 2) LINK BY REFERENCE 3) LINK BY INVERSE	SET BETWEEN ONE OWNER RECORD AND MEMBER RECORDS	LINKS BETWEEN RECORDS (NETWORK)	HIERARCHY AND KIND OF NETWORK POSSIBILITIES	SHARING ATTRIBUTES AND FOREIGN KEY
REL-ACCESS FROM AN ABSTRACT CATEGORY TO A CONCRETE CATEGORY	INDEXES	DICTIONARIES	DIRECT ACCESS BY DATA BASE KEYS	DIRECT ACCESS FILES	INVERTED FILES KEY FIELDS	SELECTION
LEVEL OF COMMUNICATION INTERACTIVE	YES WITH IQF	YES	YES (DMS 1100)	YES (ENVIRON/1)	YES	YES WITH relational Languages
HOST LANGUAGE	YES COBOL	YES COBOL	YES COBOL	YES	YES	YES (see INGRES for example)

```
entity RESORT
    |   RESORT-NAME
    |   ALTITUDE
    |   DEPARTMENT
    |   HOT inverse all HOTEL
entity HOTEL
    |   HOTEL-NAME
    |   RES reference one RESORT
    |   CLASS
    |   NB-OF-ROOMS
    |   entity RESERVATION-STATE
    |       |   DATE
    |       |   NB-OF-FREE-ROOMS
    |       |   entity RESERVATION
    |       |       |   ROOMS
    |       |       |   NAME
```

図３－２３　ＳＯＣＲＡＴＥスキーマ

図３－２３のＳＯＣＲＡＴＥスキーマにおいて各カテゴリを表３－１０にまとめる。ＨＯＴとＲＥＳはリンクを表わしているのでローカルカテゴリではない。有形ローカルカテゴリ識別子（ＩＤＯＣ）と、ローカルデータベース識別子（ＤＢＩＤ）との対（ＩＤＯＣ，ＤＢＩＤ）によって有形ローカルカテゴリは、ネットワーク内で識別される。

表３－１０　カテゴリの対応

カテゴリ	SOCRATE
有形ローカルカテゴリ	RESORT HOTEL RESERDATION RESERVATION－STATE
抽象ローカルカテゴリ	RESORT-NAME ALTITUDE DEPARTMENT HOTEL-NAME CLASS NB-OF-ROOMS DATE NB－OF－FREE－ROOMS ROOM# NAME

次に、カテゴリ間のアクセス関係を検討する。アクセス関係としては、有形カテゴリ・抽象カテゴリ、有形カテゴリ間の２つを考えねばならない。図３－２３における有形カテゴリRESORTと抽象カテゴリＲＥＳＯＲＴ－ＮＡＭＥとのアクセス関係は、relaccess（ＲESORT，ＲＥＳＯＲＴ－ＮＡＭＥ，ＲＡ１＝afn（1,1））と表わせる。これは、１つのＲＥＳＯＲＴ実現値に対してただ１つのＲＥＳＯＲＴ－ＮＡＭＥ実現値が対応していることを意味している。ｒをＲＥＳＯＲＴ対象のＩＤＯＣとすると、ＲＡ１(r)はＲＥＳＯＲＴ　ｒのＲＥＳＯＲＴ－ＮＡＭＥとなる。もしＲＥＳＯＲＴ－ＮＡＭＥがキーであるとするとアクセス関係は、

relaccess（RESORT，RESORT-NAME，RA1=afn(1,1)，inv-RA1=afn(0,1)）

となる（図３－２４）。

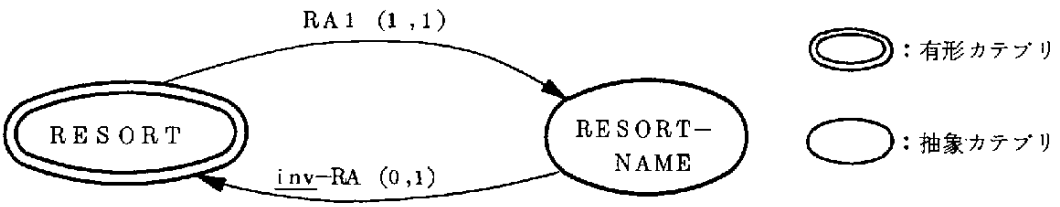


図３－２４　RESORT-NAMEがキーである時のアクセス関係

次に有形ローカルカテゴリ間の関係をみる。ＲＥＳＯＲＴとＨＯＴＥＬとのリンクは、ＲＥＳＯ－ＲＴ内のＨＯＴとＨＯＴＥＬ内のＲＥＳによってつくられている。これは、

relaccess(RESORT，HOTEL，RA3=afn(0，n)，inv-RA3=afn(1，1))と表わせる。

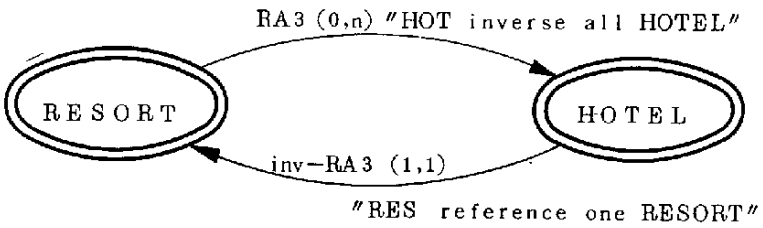


図３－２５　ＲＥＳＯＲＴとＨＯＴＥＬとの関係

アクセス関数ＲＡ３はＲＥＳＯＲＴのＨＯＴ内の全てのＨＯＴＥＬを意味している。inv-RA3は、ＨＯＴＥＬのＲＥＳを表わしている。

図３－２６に、３つのローカルデータベースを示す。３つのデータベース関の論理的関連は，Ｂ１内のカテゴリＣＯＤＥ、Ｂ２内のカテゴリＣＯＤＥと、Ｂ３内のカテゴリＮＵＭＢＥＲとは同一の値をとることである。これを基にして、ローカルデータベースを記述すると図３－２７の様になる。

```
B1 : SOCRATE
entity TEACHER
    TEACH-NUMBER   key
    NAME
    entity COURSE
        CODE
        CLASS-ROOM
        HOUR
        HNUMBER

B3 : SOCRATE
entity MODULE
    NUMBER  key
    CYCLE
    TITLE

B2 : CODASYL-DBTG
record TEACHER
    T#    key          (location mode is calculing T#
    NAME                in TEACHER duplicates are not allowed)
record COURSE
    T#
    CODE
    NB-OF-HOURS
set   TC
    owner is TEACHER
    member is COURSE mandatory automatic ;
    set occurrence selection is thru location mode of owner
```

図３－２６　データベースＢ１，Ｂ２，Ｂ３

```
Base B1 : SOCRATE

    concrete local category TEACHER key is TEACH-NUMBER
                            COURSE key is (TEACH-NUMBER, CODE)
    abstract local category TEACH-NUMBER, NAME, CODE, CLASS-ROOM, HOUR, HNUMBER
    C11 : sequencing order on TEACHER
    relaccess (TEACHER, TEACH-NUMBER, R11=afn(1,1), inv-R11=afn(0,1))
    relaccess (TEACHER, NAME, R12 = afn(1,1))
    relaccess (TEACHER, COURSE, R13 = afn(0,n))
    relaccess (TEACHER x COURSE, CODE, R14 = afn(1,1))
    relaccess (TEACHER x COURSE, CLASS-ROOM, R15 = afn(1,1))
    relaccess (TEACHER x COURSE, HOUR, R16 = afn(1,1))
    relaccess (TEACHER x COURSE, HNUMBER, R17 = afn(1,1))
```

```
Base B2 : CODASYL

    concrete local category TEACHER key is T#
                            COURSE key is (T#, CODE)
    abstract local category T#, NAME, CODE, NB-OF-HOURS
    C21 : sequencing order on TEACHER
    relaccess (TEACHER, T#, R21 = afn(1,1), inv-R21 = afn(0,1))
    relaccess (TEACHER, NAME, R22 = afn(1,1))
    relaccess (TEACHER, COURSE, R23 = afn(0,n)) (comes from set TC)
    relaccess (TEACHER x COURSE, CODE, R24 = afn(1,1))
    relaccess (TEACHER x COURSE, NB-OF-HOURS, R25 = afn(1,1))
```

```
Base B3 : SOCRATE

    concrete local category MODULE key is NUMBER
    abstract local category NUMBER, CYCLE, TITLE
    C31 : sequencing order on MODULE
    relaccess (MODULE, NUMBER, R31 = afn(1,1), inv-R31 = afn(0,1))
    relaccess (MODULE, CYCLE, R32 = afn(1,1))
    relaccess (MODULE, TITLE, R33 = afn(1,1))
```

図３－２７　Ｂ１，Ｂ２，Ｂ３の記述例

B．システム全体の記述－全体視点

全体視点（ｇｌｏｂａｌ　ｖｉｅｗ）は、新しいアプリケーションを構成するために定義される。全体有形及び抽象カテゴリとこれ等のカテゴリ間の関係との定義である。ディレクトリは、全体レベルで、各々の対象識別子に対して、この対象の実現値を持つローカルデータベースの識別子を与える。

全体レベルにおける対象は全て個々のデータベースの対象である。ローカル対象（ｌｏｃａｌ　ｏｂｊｅｃｔ）を用いて、全体レベルにおける新しいカテゴリを定義する必要がある。例えば、３つのローカルデータベースＤＢ１，ＤＢ２，ＤＢ３は、次の様なローカル対象を持つとする。

$STUDENT : \{S_1, S_2, \cdots, S_n\}$　　（ＤＢ１）

STUDENT : $\{S_1, S_2, \cdots, S_p\}$ （DB2）

TEACHER : $\{t_1, t_2, \cdots, t_g\}$ （DB3）

全体視点として、これらのローカル対象を集めて新たにＰＥＲＳＯＮカテゴリを生成するには次の様にする。

concrete global category PERSON composed with STUDENT of DB1,
STUDENT of DB2, TEACHER of DB3.

ＰＥＲＳＯＮは、集合$\{S_1, \cdots, S_n, S'_1, \cdots, S'_p, t_1 \cdots, t_g\}$となる。有形全体対象識別子（global object identifier）（ＧＩＤ）は、ＩＤＯＣとＤＢＩＤの対、即ちGID＝（ＩＤＯＣ，ＤＢＩＤ）である。又Ｎ（ＧＩＤ）＝IDOC of GID, B(GID)=DBID of GID である。

図3−28の例より、全体カテゴリPROFESSORは、Ｂ１とＢ２のＴＥＡＣＨＥＲから構成される。ＰＲＯＦＥＳＳＯＲのキーＰ#は、Ｂ１のＴＥＡＣＨ−ＮＵＭＢＥＲとＢ２のＴ#から構成される。これは次の様に記述される。

concrete global category PROFESSOR composed with TEACHER of B1,
TEACHER of B2,
key is P#
abstract global category P# composed with TEACH−NUMBER of B1,
T# of B2

次に、データベースサイトで対象が重複している場合を考えよう。同じ例で、B3のMODULEは、Ｂ１とＢ２におけるＣＯＵＲＳＥと同じである。このことは次の様に表わせる。

concrete global category COURSE composed with MODULE of B3
occurs as COURSE of B1,
COURSE of B2,

全体対象がローカル対象の副集合のこともある。

concrete global category PERSON composed with
STUDENT of B1 （AGE≧18）

全体カテゴリ間のアクセス関係は、有形ー抽象カテゴリ間と有形カテゴリ間について定められねばならない。全体レベルでのアクセス関数はローカルアクセス関数を用いたプログラムである。

図３－２７に対する全体視点を図３－２８に示す。

concrete global category COURSE composed with MODULE of B3
occurs as COURSE of B1,
COURSE of B2.

concrete global category PROFESSOR composed with TEACHER of B1,
TEACHER of B2.

abstract global category P# composed with TEACH-NUMBER of B1, T# of B2.
NAME composed with NAME of TEACHER in B1,
NAME of TEACHER in B2.
CYCLE composed with CYCLE of MODULE in B3,
TITLE composed with TITLE of MODULE in B3.
NUMBER composed with NUMBER of MODULE in B3,
occurs as CODE of COURSE of TEACHER in B1,
CODE of COURSE in B2.
TOTALH result of F3

図3－28　全　体　視　点

```
rel (PROFESSOR, P#, F1 = afn(1,1), inv-F1 = afn(0,1))
rel (PROFESSOR, NAME, F2 = afn(1,1), inv-F2 = afn(0,∞))
rel (PROFESSOR, TOTALH, F3 = afn(1,1), inv-F3 = afn(0,∞))
rel (PROFESSOR, COURSE, F4 = afn(0,n), inv-F4 = afn(0,p))
rel (COURSE, NUMBER, F5 = afn(1,1), inv-F5 = afn(0,1))
rel (COURSE, CYCLE, F6 = afn(1,1), inv-F6 = afn(0,∞))
rel (COURSE, TITLE, F7 = afn(1,1), inv-F7 = afn(0,∞))
```

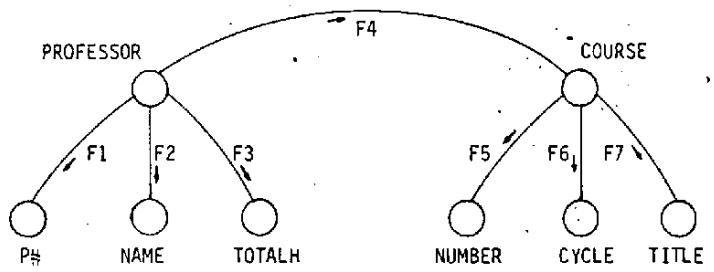


図3－29　全体カテゴリ間の関係

例えば全体アクセス関係数Ｆ２は、次の様に示される。全体有形対象識別子（ＧＩＤ）ｘに対して、Ｂ（ｘ）はローカルデータベース（ＤＢＩＤ）を明らかにし、Ｎ（ｘ）はローカル有形対象（ＩＤＯＣ）を明らかにする。このプログラムはまずある全体有形対象がどのデータベースサイトに存在しているかを調べる。次にこのローカルデータベース内の対応するローカル対象へ、ローカルアクセス関数（Ｒ１２又はＲ１３）を適用して必要な対象を明らかにする。

```
accessor F2 ← prog(x)
    if b(x) = B1 then resume (R12(N(x)))
    else resume (R22(N(x))) endif
end F2
```

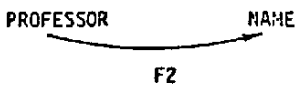


図３－３０　全体アクセス関数Ｆ２のプログラム

新しいアプリケーションは、上述した様な全体視点を用いることが出来る。例えば、図3－23の全体視点のもとに、"ある学期ｘにおいて少なくとも１つの授業を持っている教授名を全てあげよ"という問合せを考えよう。これは全体アクセス関数（図３－２９）を組み合わせて、次の様になる。

$$P \leftarrow F2(\underline{inv}-F4(\underline{inv}-F6(x))) \qquad (1)$$

ここでＰは教授名（ＰＲＯＦＥＳＳＯＲ's　ＮＡＭＥｓ）の集合である。この問合せは、ある学期（ＣＹＣＬＥ）ｘ内の全ての授業（ＣＯＵＲＳＥ）をアクセス（inv－Ｆ６）し、こうした各々の授業から担当している教授（ＰＲＯＦＥＳＳＯＲ）をアクセス（inv－Ｆ４）し、最後にこの教授の名（ＮＡＭＥ）をアクセス（Ｆ２）することを意味している。

ＰＯＬＹＰＨＥＭＥにおける問合せ言語は、述語論理における限定作用素（∃と∀）とを持った全体アクセス関数を用いた関数記述である。(1)の問合せＲは、

Ｒ：ｉ　name　∀(inv-professorcourse ∃(inv-cycle=x))？となる。

ここでcycleはＦ６，professorcourseはＦ４，nameはＦ２へ対応している。

C．分散問題とその必要機能

ここでは分散問題と、必要な分散機能として問合せ分割を論じる。

①　分散問題と多重グラフ

ＰＯＬＹＰＨＥＭＥにおける分散問題は、全体視点と、ローカルデータベース記述との対応である。全体視点における全体カテゴリとローカルカテゴリ及び全体アクセス関数とローカルアクセス関数との対応である。この対応は図３－３２の様な多重グラフによって示される。多重グラフは、個々のローカルデータベースの記述（図３－２７）と、あるアプリケーション向けの全体視点の記述（図３－２８と図３－２９）内の情報を用いて構成される。ＰＯＬＹＰＨＥ－ＭＥにおける分散問題は、多重グラフとこれに対応した記述とによって表現される。この２つをディレクトリと考えることが出来る〔図３－３１〕。これは次の様に情報を備えている。

ⅰ）抽象と有形ローカルカテゴリ

ⅱ）ローカルアクセス関数

ⅲ）抽象と有形全体カテゴリ

ⅳ）全体カテゴリとローカリカテゴリとの関係（分散、冗長さ、制限等）

ⅴ）全体カテゴリ間のアクセス関係（全体アクセス関数）

ⅵ）全体アクセス関数とローカルアクセス関数との関係

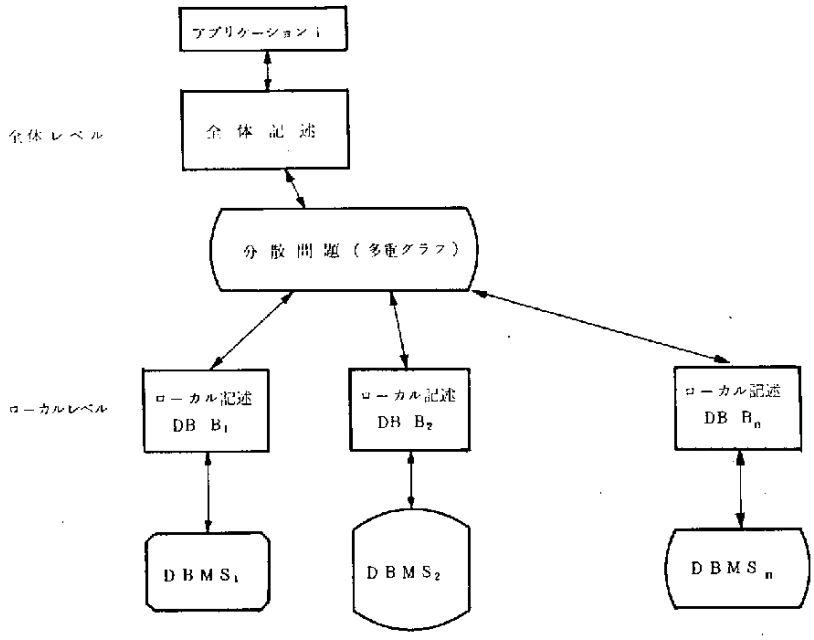


図３－３１　ＰＯＬＹＰＨＥＭＥにおける分散問題

図３－３２は、図３－２７と図３－２８とに基づいた多重グラフである。

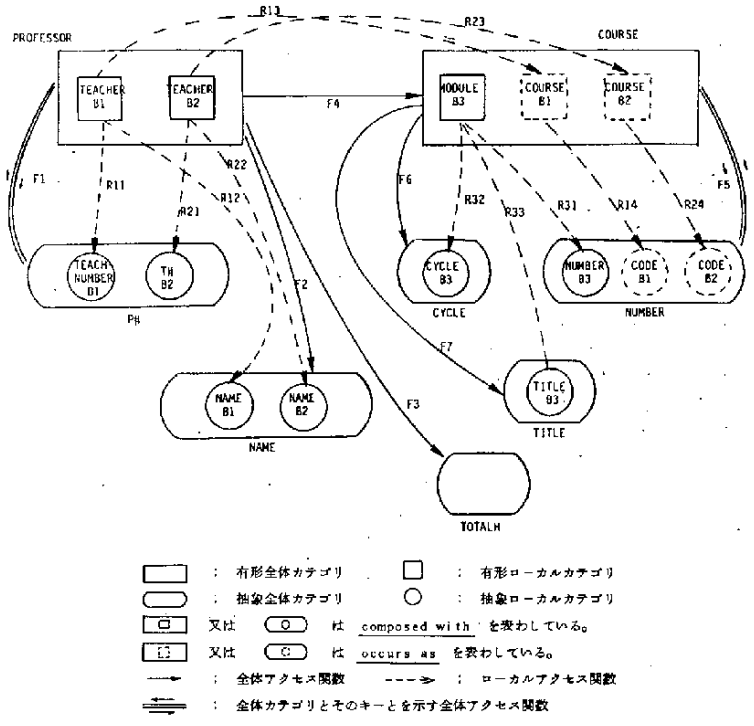


図３－３２　多重グラフによるローカル及び全体視点の記述

2）問合せの分割

全体レベルでの問合せ〔これを全体問合せ（global guery）と呼子〕は、個々のローカルデータベースサイトへの問合せに分割されねばならない。問合せ分割における留意点は次の通りである。

ⅰ）カテゴリが幾つかのデータベース上に分散している場合、各々のデータベースへ同一の問合せを生成する。更に、逐次パスの利用法を検出する。

ⅱ）２つの全体カテゴリ間に関係が存在するならば、この関係を、共通カテゴリ（しばしばキーとなる）を持つ幾つかのローカルカテゴリの組合せへ変換する。

ⅲ）障害対策や異なった問合せを順番にさせるために補助的な演算が必要となる。

全体問合せは、全体視点の多重グラフを用いてローカル問合せに分割される（図３－２８）。ローカル問合せとは、ローカル視点の記述に基づいたものであり、各ローカルDBMSにおける問合せではない。全体問合せからローカル問合せの生成は、まず全体問合せを、多重グラフを用いて分割が用易なように木構造へ翻訳される。ローカル問合せへの分割に伴って、ローカル問合せの各々の実行の同期を制御するために、これらの問合せの半順序づけ（partial ordering）を行なう。

Bで述べた問合せ

R：i　name ∀（inv-professorcourse ∃（inv-cycle＝x））？の分割を考える。

ⅰ）ローカルデータベースＢ３への問合せｒ１の生成

r_1　：　number ∀（inv-cycle＝x）

R３１　∀（inv-R３２＝x）　（ローカルアクセス関数）

Ｂ１，Ｂ２，Ｂ３を結びつけているカテゴリは、Ｂ２のＮＵＭＢＥＲと、Ｂ１とＢ２のＣＯＤＥであった。よって、r_1 はＢ３からＣＹＣＬＥ＝ｘであるＭＯＤＵＬＥの全てのＮＵＭＢＥＲを得るために生成される。この問合せｒ１は、ＳＯＣＲＡＴＥ問合せ言語へ変換されて、ＮＵＭＢＥＲの集合Ｎ＝$\{ n_1, n_2, \cdots, n_k \}$を与える。Ｎは、Ｂ２とＢ３への問合せに使われる。

ⅱ）ローカルデータベースＢ２への問合せｒ２の生成。

r_2　：　name ∀（inv-professor course ∃（inv-number＝N）

R２２ ∀（inv-R２３ ∃（inv-R２４＝N））

ⅲ）ローカルデータベースＢ１への問合せr_3 の生成。

r_3　：　name ∀（inv-professor course ∃（inv-number＝N）

R１２ ∀（inv-R１３ ∃（inv-R１４＝N））

カテゴリＣＯＵＲＳＥは、Ｂ２とＢ１へ分散されているために、ｒ２とｒ３の同一問合せが生成される。これによってＣＯＵＲＳＥ番号の集合Ｎが与えられると、ＣＯＵＲＳＥのＣＯＤＥがＮのどれかの要素と等しい全ての教授名が得られる。ローカル問合せから、実際のＤＢＭＳ（ＳＯＣＲＡＴＥ，ＤＢＴＧ）の問合せ言語の変換は、分散問題とは独立であり、異種性問題で論じる。

次に半順序化について考える。全体問合せＲがローカル問合せ r_1 ， r_2 ，…， r_n へ分割されるとする。この時、これらのローカル問合せの半順序化を定める。例えば“ r_1 は、r_2 と r_3 よりも先に処理し、r_2 と r_3 は独立である”と定義する。この定義は、Ｂ２への問合せ r_2 とＢ１への問合せ r_3 とが平行して処理された後、Ｂ３への問合せ r_1 が処理されることを示している。

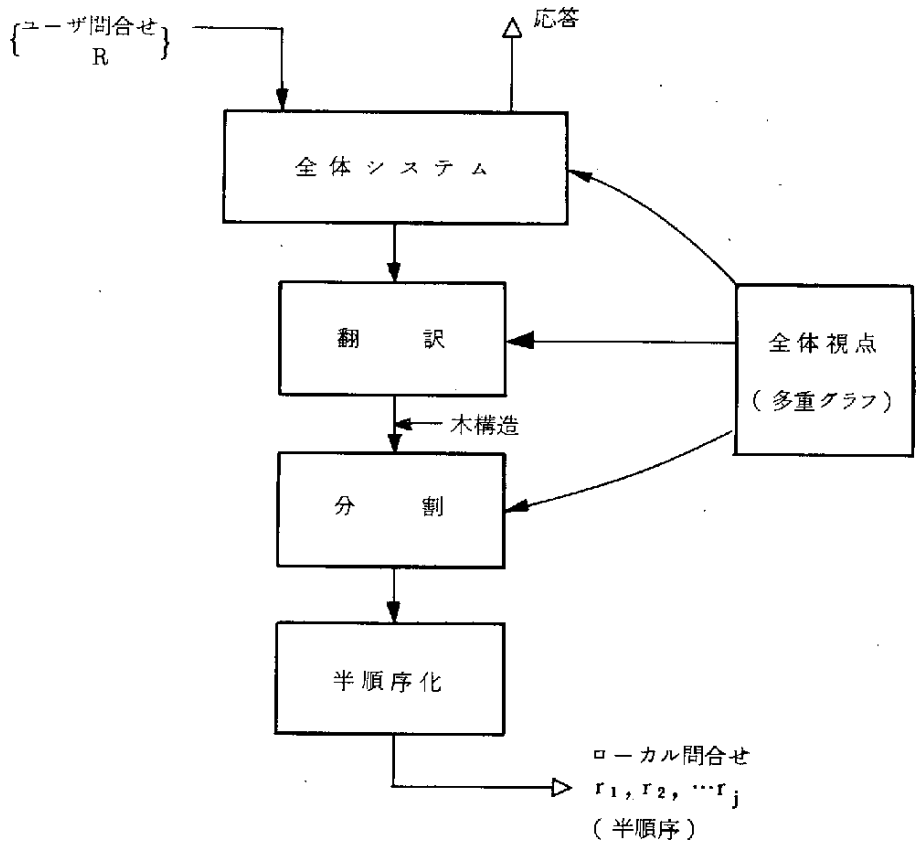


図３－３３　全体問合せＲの処理

D．異種性問題

ＰＯＬＹＰＨＥＭＥにおける異種性問題は、２項関係モデルによって記述されたローカルデータベースと、実際のＤＢＭＳ（ＣＯＤＡＳＹＬ－ＤＢＴＧ，Ｓ２０００、ＳＯＣＲＡＴＥ等）との対応関係である。異種性の問題は、共通↔ローカルの変換の問題である。ＰＯＬＹＰＨＥＭＥのアプローチは、各種ＤＢＭＳをAbrialによる２項関係モデルによって、サイト単位に記述する。各種（関係、網型、階層型）データモデルを２項関係モデルで記述することの 可能性は、〔ＡＤＩＢ７６，７７〕内に示されている〔表３－１０〕。

次の問題は、全体問合せから分割されたローカル問合せを、実際のＤＢＭＳのアクセス言語に変換出来るかである。

1）変　　換

ローカル問合せの実際のＤＢＭＳで処理される言語への変換を考える。この場合、ＤＢＭＳ間でどの通信レベルで交信するかが問題となる。先に述べたようにＰＯＬＹＰＨＥＭＥでは、問合せ言語と親言語レベルを通信レベルとして考えている。このことは、ＤＢＭＳの各型に対して変換器が必要であることを意味している。

この点に関して言えることは、多くのＤＢＭＳは、ＣＯＢＯＬの様な同種の言語を親言語として用いていることである。このことは、変換器作製コストを縮少させることが出来る。

第２の方法として、全体レベルでのトランザクションを、ローカルプログラムの呼出し形式へ変換するものがある。これ等のローカルプログラムは、全体システムを構成する時点で、ローカルデータベース内に格納されていねばならない。

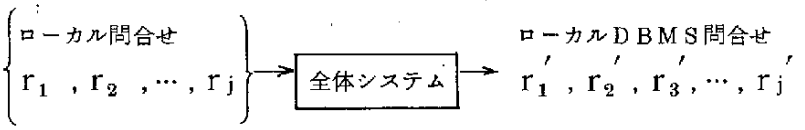


図３－３４　ローカル問合せ変換

2）ローカルＤＢＭＳ問合せの実行

ローカルＤＢＭＳ問合せr_k'は、ローカルＤＢＭＳによって処理される。ここにおける問題点として次の点がある。

ⅰ）全体システムから、個々のＤＢＭＳの起動。全体システムを、プログラム又は問合せを送信するユーザのように考えることが必要である。

ⅱ）ローカルＤＢＭＳ問合せの実行結果の全体システムへの転送。ここでの問題は、次の様な特別な状態が起きることである。

a）あるＤＢＭＳがソフト又はハード障害を起こしている。

b）ネットワーク障害によって、あるデータベースと結合を行なえない。

c）ＤＢＭＳが誤まったメッセージ等を送っている。

iii）Bの２）で考えたように、半順序化によったローカル要求の同期化。

iv）全体問合せへ対する全体応答をより便利な形式にすること。

3.3.5 ま と め

以上、ＩＮＧＲＥＳ，ＳＤＤ－１，ＬＡＤＤＥＲ，ＰＯＬＹＰＨＥＭＥの４つの分散型データベースシステムを検討してきた。４つの大きな特徴はコンピュータネットワーク上のデータベースサイトがＤＢＭＳであることである。

本節では、分散型データベースを次の点から検討した。

1）分散型データベースシステム全体における統一的なデータモデルと問合せ言語は何としているか。必要とされる特徴は何か。

2）ネットワークを介してデータベース間の通信を行なうことによって生ずる問題、即ち分散問題は何か。及びデータがネットワーク上に分散していることによって必要となる機能は何か。

3）種々のＤＢＭＳをシステム内に含むことによって生ずるデータベースの異種性の問題、と異種性によって必要となる機能は何か。

上記の点に関した各システムの検討を表3－11にまとめる。表３－１２は、１）データモデルと問合せ言語　２）分散問題　３）分散機能　４）異種性問題　５）異種性機能についてまとめた。１）は各システムがどの様なデータモデルと問合せ言語を用いているかを示している。２）は分散問題であり、ユーザに視えるモデル要素と、実際の所在サイトとの対応関係を示している。３）には分散問題のために必要となる機能をあげてある。４）は異種性問題であり、統一データモデルとローカルＤＢＭＳとの対応を示している。５）は必要となる異種性機能をあげている。

ＩＮＧＲＥＳとＳＤＤ－１は、１）に関して同様な視点に基づいている。この２つは、データベースサイトとして同種のＤＢＭＳを用いているために異種性問題は存在しない。ＳＤＤ－１は、データコンピュータ以外のＤＢＭＳについても検討を始めている。

ＬＡＤＤＥＲは、ＩＮＬＡＮＤ，ＩＤＡ，ＦＡＭの３つの要素から成っている。ＩＮＬＡＮＤは自然言語インタフェースであり分散型データベース問題とは別個に考えられる。よって、ＩＤＡとＦＡＭとを表３－１１にのせた。ＩＤＡ，ＦＡＭを分散及び異種性の観点から議論する場合、この２つの問題が未分離であることを指摘出来る。分散機能としては多くの点が不明である。

ＰＯＬＹＰＨＥＭＥは、他のシステムに対して、既存の種々のＤＢＭＳのネットワークを通しての統合を目指していることが重要な特徴である。このため異種性問題の検討が行なわれている。他の大きな特徴は、各種ＤＢＭＳのスキーマを、２項関係モデルによって記述されたローカルデータベースから統一モデルも同じく２項関係モデルによって記述している。分散及び異種性問題

はこの記述内に明らかとなっている。統一的なデータモデルの重要性を、我々は指摘出来る。第2章で述べたように統一モデルとしては、データの意味を記述する概念モデルであろう。

分散型データベースにおいて、ユーザは個々のデータベースの異種性と、必要なデータがどのデータベースサイトに存在するかを知る必要がないこと、即ち不可視性（invisibility）〔ROTH77c〕が重要である。不可視性の実現は、実際のローカルデータベースの集合から、異種性問題とデータの分散問題とを除去し、ネットワーク上へ分散したデータの意味を記述するデータモデルと高水準非手続的問合せ言語とを提供することである。この点より、既存の分散型データベースシステムを1）データモデル　2）分散問題　3）異種性問題の3点について検討したわけである。

これらの検討より、分散型データベースは次の3層から成ると考えられる〔図3－35〕。

1）ローカル内部スキーマ層（local internal schema level）
2）ローカル概念スキーマ層（local conceptual schema level）
3）全体概念スキーマ層（global conceptual schema level）

ローカル内部スキーマは、コンピュータネットワーク上の各データベースサイトに対応している。各データベースサイトは、データモデルと問合せ言語とを備えている。ローカル内部スキーマは、各サイト内に格納されている情報を、各々のデータモデルに基づいて記述したものである。コンピュータ等の各データベースサイトの固有情報も記述されいる。

ローカル概念スキーマは、ローカル内部スキーマにおける各データベース固有の問題（即ち、異種性問題）を除き、分散型データベースシステム全体で共通のデータモデルを用いて、各データベースサイトを記述したものである。ローカル概念スキーマとして記述されるものは、各サイト内のデータの意味である。

全体概念スキーマは、個々のデータベースサイトのローカル概念スキーマからデータの分散情報を除去し、ある企業体にとって関心あるデータの意味の記述である。全体概念スキーマは、ローカル概念スキーマと同じデータモデルを用いる。

この様に、異種性の問題はローカル概念スキーマ層への変換時に導入される。逆にローカル内部スキーマ層からローカル概念スキーマ層への変換時には除去される。ローカル概念モデルは、異種性に関する不可視性を実現している。一方、分散問題は、全体概念スキーマとローカル概念スキーマとの対応問題である。ローカルから全体概念スキーマへの変換時に、分散問題は除去される。逆の変換において、分散問題は導入される。即ち、全体概念スキーマ層は、分散と異種性の不可視性を備えている。図3－35におけるような三層の構造を分散型データベースの基本概念として、第6章において分散型データベースシステムの基本構想を論じる。論じる点は次の点である。

表 3 － 1 1　分散型データベースシステムのまとめ

検討点＼システム	INGRES	SDD－1	POLYPHEME	LADDER IDA	LADDER FAM
1）データモデル 問合せ言語	関係モデル QUEL	関係モデル QUEL	2項関係モデル アクセス言語	項目問合せ	データ言語 （他の言語？）
2）分散問題	関係 ローカル関係	関係 フラグメント（具現） 冗長コピー	全体視点 ローカル視点	関係問合せ データ言語	一般ファイル名 実ファイル名
3）分散機能	システムカタログ 問合せ分割 一致性 同時実行	ディレクトリ 問合せ分割 冗長コピーの更新	多重グラフ 問合せ分割 ローカル問合せの半順序化	？	？
4）異種性問題	（PDP－11　UNIX）	（データコンピュータ） （　他のDBMS？　）	ローカル視点 ローカル DBMS	項目問合せ 関係問合せ　他のモデル？	（データコンピュータ） （他のDBMS？）
5）異種性機能			問合せ変換 問合せ応答合成 ローカルDBMS問合せの同期 DBMSの起動	構造スキーマ	ネットワーク支援 （ログオン／オフ） 障害管理

1）データモデルと問合せ言語

2）分散問題

3）分散問題における必要機能

4）異種性問題

5）異種性問題における必要機能

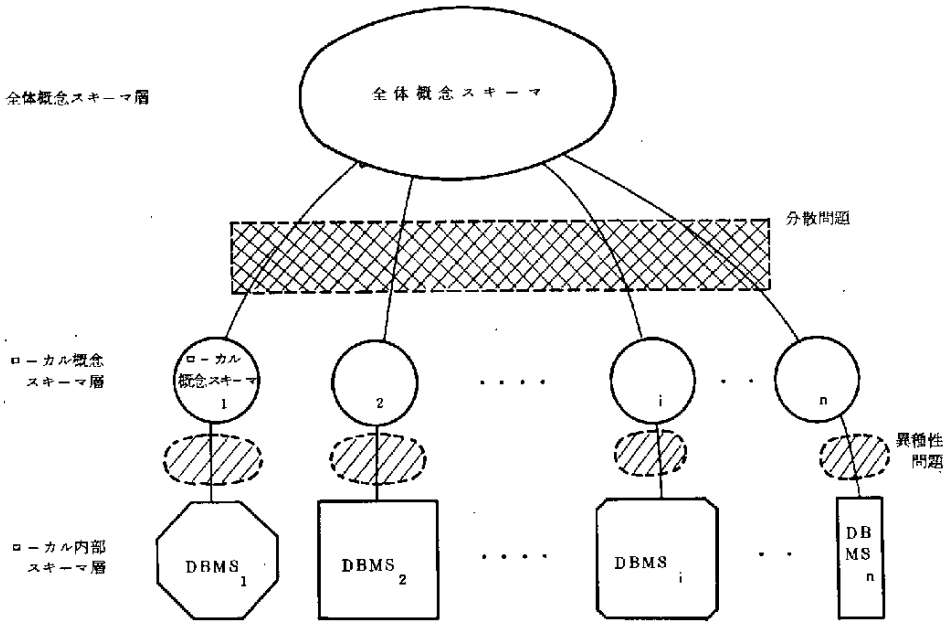


図３－３５　分散型データベース

3.4　ま　と　め

本章では分散型データベースのシステム例について検討してきた。

3.2節において検討したようにＣＯＮＩＴシステムの成功と、ヨーロッパにおける国家間規模のＥＵＲＯＮＥＴの開発をまず指摘できる。これらは、既存の文献データベースをコンピュータネットワークを介して結合しようとするものであり、文献検索という１つのアプリケーションにおける実用化例である。種々の情報検索システムの持つデータベースは"文献"に関する情報を持っており、これらのデータの論理構造及びユーザの視点は全て同一である。相違点は、これらの情報検索システムの会話型ユーザインタフェースである。インタフェースは、各情報検索システムにおおいて同様な検索機能を備えている。このためシステムはネットワーク全体における共通機能に対して共通インタフェースを設定し、このインタフェーズと各種ローカルインタフェースとの変換を行なうものである。

上記の２つのシステムの成功と発展は、"文献"という１つのアプリケーションにおけるもので

ある点は第一に指摘すべきである。文献データの意味記述は、どのデータベースにおいても同一であるために、インタフェースの変換において共通的なアクセスが可能となったわけである。よく定義されたアプリケーションのみを対象とするには、このアプローチが適していると考えられる。この様なアプリケーションでは、データベースは、同一で簡単な論理構造をもち、データベースの意味記述も同一である。1つのアプリケーションに対してではなく一般的な場合を考える上では、上記のアプローチは適さない。一般的な場合、即ち、データベースの意味記述が各々異なっている場合には、上記システム例の様なファイルシステムではなくＤＢＭＳを扱かう必要がある。この点より、我々は3.3節において、ＤＢＭＳをサポートする分散型データベースについて論じた。ＤＢＭＳの特徴は、あるデータモデルと言語とを備えていることであり、ユーザはこれらを用いてデータ独立にアクセス出来ることである。

この様な各種ＤＢＭＳ（ＣＯＤＡＳＹＬ－ＤＢＴＧ，ＳＹＳＴＥＭ2000，TOTAL，ADABAS，関係ＤＢＭＳ等）間の通信レベルとして、各ＤＢＭＳの個々の有効性に関連したレベル（内部機構レベル）ではなくより上位のレベルを用いることが望ましい。このため、問合せ言語と親言語レベルとを考える必要がある。

3.3節では、現在開発中のシステムとして、ＩＮＧＲＥＳ，ＳＤＤ－１，ＬＡＤＤＥＲ，ＰＯＬＹＰＨＥＭＥ，の4つのシステムについて検討した。これらのシステムにおいて、一般的なアプリケーションを想定しているために、各アプリケーションにおけるデータの意味を記述するための共通データモデルが必要になる。又、ユーザにデータアクセスの容易な高水準問合せ言語を提供することも必要になる。

分散型データベースにおける重要な概念として、不可視性を指摘できる。即ち、不可視性とは、ユーザがシステムを構成する各データベースサイトに固有な異種性と、必要なデータがどのサイトに存在するかのデータの分散性とを意識することなくデータをアクセス出来ることである。このために必要なことは、システム全体における統一データモデルの設定と、分散問題と異種性問題との分離とである。データモデルは、構成するローカルなＤＢＭＳの内部機構とは独立であり、データの意味を、一意に形式的に記述したものである。このデータモデルは、他に各サイトの内部機構の変化とユーザのアプリケーションの多様化とへ対して定常性があること、データの内容の変化へ対して発展性、ユーザの理解の容易性、他のデータモデルへの変換の容易性等も必要な要求としてもっている。ＩＮＧＲＥＳ，ＳＤＤ－１は、関係モデルを用いているが、第2章で指摘したように関係モデルは、データの意味記述に問題点をもっている。これに対してＰＯＬＹＰＨＥＭＥは、より意味記述に優れた2項関係モデルを用いている。意味記述を目指さすいわゆる概念モデルとして、種々のモデルが提案されている。どの様なモデルを採用するかの検討は今後必要である。

分散型データベースにおいて、分散と異種性の問題の分離と、各々の問題に対して必要となる機能とを検討した。分散の問題とは、システム全体の概念スキーマと、異種性情報を持たないローカル概念スキーマとの対応である。この対応を実現するために必要な機能として、ディレクトリ、問合せ分割、分割されたローカル問合せの半順序化と同期、同時実行制御、一致性制御、冗長コピーの

構成等を検討する必要がある。異種性問題は、ローカル概念スキーマと、各々の固有情報を持ったローカル内部スキーマ（既存ＤＢＭＳの問合せ言語又は親言語レベル）との対応問題である。異種性問題は、共通・ローカル間の変換の問題である。この問題は、Adiba等によるPOLYPHEMEにおいてのみ検討されている。既存の種々のＤＢＭＳ統合を行なうためには、必要となる問題と機能である。ＤＢＭＳ間のデータベース体全の変換問題は、〔ＳＨＯＳ７５〕等によってなされている。

以上より、汎用の分散型データベースシステムの検討は、次の点についてなされる必要がある。

1）データモデルと問合せ言語
2）分散問題
3）分散機能
4）異種性問題
5）異種性機能

これらの点について、第6章において、我々の基本構想を論じる。

4．ネットワーククリアリングセンターの検討

本章では、異機種のリソースシェアリングシステム上にあるリソースを統合していく上で、重要な機能の一つであるＮＣＣ（ネットワーククリアリングセンター）について議論する。このＮＣＣがもつ機能を一言でいうなら、どのリソースがどこにあるかをユーザに知らせることといえる。まず4.1で、世界的な視野でみたときの情報界の新しい動きを把える。さらにＮＣＣを広義に解釈して、いわゆる情報管理でいうクリアリングセンターについて定義を行なう。4.2では、科学技術情報の分野で検討されているクリアリングセンターの構想例を示す。最後に4.3で、リソースがどこにあるかの情報、つまり所在源情報管理の基本的技術に関する検討を行ない、ＤＤ／Ｄ的アプローチの有効性とデクショナリの管理形態に言及する。

4.1 クリアリング機能

クリアリング機能は、１９６３年米国においてワインバーグ報告（Weinberg Report）〔ＮＩＰＤ６６〕の中に発表されて以来、各国で開発されてきている。米国においては、スミソニアン協会のＳＳＩＥが最も代表的なものである。一方、我が国においても、１９６９年１０月に、科学技術会議が答申したＮＩＳＴ構想（科学技術情報の全国的流通システム構想：National Information System for Science and Technology）〔ＮＩＰＤ７０〕が、１９７０年代における日本の科学技術の発展をささえるものとして注目を集めた。現実的には、１９７６年7月よりＪＩＣＳＴが全国的なオンライン・サービスを目指して稼働したＪＯＩＳシステムの中に、１９７７年６月から、クリアリング・ファイルが追加されたこともあり、このシステムが最大規模のものと言えよう。今後、このシステムに寄せられる期待は益々高まるものとなろう。

世界的にみると情報の交換（共有）といった観点から幾つかの新しい動きをみることができる。以下、その主なものを列挙する。

① 世界的情報ネットワークの実質的発展

- ＵＮＩＳＩＳＴ（世界科学情報システム）
- ＩＮＩＳ（国際原子力情報システム）
- ＡＧＲＩＳ（ＦＡＯ、農業情報システム）
- ＩＲＳ（国連環境会議に基づく環境情報源案内サービスの国際レフェラルサービス）

② 世界の地域ネットワーク化・ブロック化

- ＥＵＲＯＮＥＴ（ＥＣ）
- ＮＯＲＤＦＯＲＳＫ情報ドクメンテーション・ネットワーク（スカンジナビア諸国）
- ＭＩＳＯＤ（国際出版物情報システム、１９７７～１９７８年までに稼働）
  ＶＩＮＩＴＩ（全ソ科学技術情報研究所）が開発し、９カ国（ソ連、東欧諸国）がメンバーとなっている。
- ＥＳＡ／ＳＤＳ（欧州宇宙局／宇宙ドキュメンテーション・システム）

- ＴＥＣＨＮＯＮＥＴ（東南アジア諸国の中小企業のための産業情報システム…カナダ援助）
- ＦＩＤ／ＣＬＡ（ラテンアメリカ諸国、国際ドクメンテーション連盟・ラテンアメリカ地域協議会）
- ＡＩＢＡ（農業情報バンクのアジア・ネットワーク）

③ 機関ファイルの世界市場への侵出・拡大

- ＮＴＩＳ（米国商務省下）
- ＮＬＭ（ＭＥＤＬＡＲＳファイル）
- ＣＡＳ（アメリカ化学協会）
- ＳＳＩＥ（スミソニアン協会）
- ＩＮＳＰＥＣ（イギリス、ＩＥＥ）
- ＣＮＲＳ（フランス、ドキュメンテーション・センター）

④ 各国情報政策推進方策の見直しと実効化

- イギリス … National Lending Library の British Library への統合
- 西ドイツ … Information und Dokumentation の樹立
- フランス … ＢＮＩＳＴ（仏国科学技術情報局）設立
- アメリカ … ＮＳＦ（国立科学財団）を中心とした情報政策の見直し
- 日　本 … ＮＩＳＴ計画の見直しと推進

⑤ 米国における新法人設立案の動向

- ＮＴＩＳ、ＳＳＩＥ、ＮＳＦ／ＯＳＩＳ（科学情報サービス局）の三機関を合併して Science and Technology Information and Utilization Corporation の設立を要望する新しい議案が提出されている。
- 最近開催された下院聴聞会の議案では、さらにScience Information Council を加えて一機関に併合させようとする動きがある。

4.1.1 クリアリング機能とは

ここでは、幾つかの文献からクリアリング機能（機構、情報）について、定義を行なう。

A．科学技術会議の意見によるクリアリング機構〔ＮＩＰＤ７０〕

目的：錯そうする情報流通過程のなかから、必要な情報を迅速に選択し利用者に提供する。情報の案内所として情報の入手方法についての情報を提供する。

業務：(1) 国内文献その他の資料、そのコピーの入手方法に関する情報の提供

(2) 研究者の所属機関、研究歴、現在の研究テーマ、発表論文等の情報提供

(3) あらゆる情報機関との連絡

(4) 海外のクリアリング機構などとの連絡

(5) 上記業務のための基礎資料の整備調査

B．利用研・制度研究班報告書によるクリアリング機能〔ＲＩＹＯ７６〕

クリアリング機能とはいくつかの組織体の有するデータやソフトウェア等のリソースを有効に利用していくための情報を提供する機能である。この機能をサービス或は情報提供の形態で分類すると、所在源情報提供、レファレルサービス、一次情報の提供、の三つに分類でき、以下それぞれについて概説する。

① 所在源情報提供

情報につけられた表題に対してその情報の所在に関する情報だけを提供する。

② レファレルサービス

利用者の必要とする情報の内容をも考慮し、利用者に的確な情報をもつ組織体或はデータベースを紹介するものである。

③ 一次情報の提供

クリアリング機構を介して利用者が一次情報を直接入手するものである。

C．情報管理、連載講座「情報の効果的な入手と利用法」によるクリアリング情報〔ＴＡＮＡ７６〕

ある主題についての研究はどこで行なっているかという進行中の研究課題の情報、どこの何という機関または人が最も信頼できる情報を提供できるかといった情報源に関する情報、そして政府の研究費の援助を受けて行なわれた研究成果の報告書（レポート）に関する情報、これらの三つの情報を総合してクリアリング情報と呼ぶ。

いいかえれば、どこで研究しているか、どこへ聞いたらよいか、レポートは出ているか、に関する情報をクリアリング情報とよぶ。

4.1.2 レファレンスサービス

本項では情報サービスの一つであるレファレンスサービスをとりあげる。4.1.1項で述べたクリアリング機能（サービス）も情報サービスの代表的な一つであるが、レファレンスサービスと明確に区別する必要がある。

定義A

レファレンスサービスとは、何らかの情報を求めて図書館を利用する人が、図書館を効果的に利用できるように、図書館員が直接その利用者にできるだけ援助する仕事のことである〔ＯＤＡ６６〕。

定義B

レファレンスサービスとは、利用者の要求とそれに適合する情報源とを結びつけるために、ライブラリーや情報センターなどの情報活動組織が、その組織力を用いて、計画的、組織的に利用者に対し、直接あるいは間接に行なう援助業務およびその管理をいう〔ＮＩＫＫ６５〕。

サービス内容：１９６１年に、米国図書館協会のレファレンスサービス部会内に設けられたレファレンス基準・統計委員会が、レファレンス関係の諸活動について、次のような案を提出している〔ＫＡＷＡ７７〕。

(1) レファレンス援助

(2) 調査援助 … 検索、翻訳等

(3) 利用指導 … 目録の利用指導、グループ指導等

(4) 諸活動の計画 … 集会、討論会等

(5) 書誌作成 … グループ・個人に対して等

(6) 資料の選択 … 図書、パンフレット等

(7) レファレンスの組織 … 図書の発注、図書・フィルムの廃棄等

4.1.3 クリアリングサービスの現況

本項ではクリアリングサービスを機能別に三分類して、国内外の代表的実例をそれぞれ紹介する。

A. クリアリングサービス（研究課題情報*案内業務）

現在計画中もしくは研究中の研究課題に関する情報を提供すること。広義にはレフェラルサービスも含める。

* ongoing research or current research information

B. レフェラルサービス（情報源案内業務）

利用者が知りたい事柄の内容そのものを教えるのではなく、最も信頼しうる情報を提供できる機関又は個人等を案内すること。

C. 一次情報の提供

政府機関の援助のもとに研究が行なわれた研究報告書を提供すること。

A. クリアリングサービス（研究課題情報源案内業務）

① ＳＳＩＥ（Smithonian Science Information Exchange、米国）

スミソニアン協会の下部機構の一つであるＳＳＩＥは、１９４９年にNational Institute of Healthの内部組織として、Medical Sciences Information Exchange が設立されたことに始まる。その後、１９５３年にBiosciences Information Exchange と改称され、同時にスミソニアン協会に移管された。

目的：１９４９年、政府委託により医学分野の研究プロジェクト、研究者、研究機関の調査を開始、どこで、だれが、今何を研究しているかということについての情報を提供すること。

対象分野：当初の医学分野を始めとして、生物学、心理学、社会科学、物理学、農学、理工学と多岐に渡っている。

情報源：研究基金を授与している約1,３００の機関、つまり連邦及び州政府機関、公益団体、大学、企業体などで、計画又は進行中のカレントな研究についてのみの情報を対象にしている。

利用状況：年間約１０万件の情報を蓄積し、年間約１万件の質問に応えている。

特記事項：ＳＳＩＥ独自の検索サービス（ＳＤＩもあり）もあるが、ＳＤＣ（System

Development Corporation ）のオンライン検索システムＯＲＢＩＴの一データベースとしても利用できる。さらに、ファイルそのものをinformation packageとして入手可能である。

表４－１　ＳＳＩＥの検索サービスの内容〔ＴＡＮＡ７６〕

名称		内容	料金
カスタム検索		注文によるサービスで，主題による検索と機関名による検索の２種類	１件50ドル
SDI	スタンダード	カレントな情報を月１回定期的に知らせる	年間 180ドル
	カスタム	カレントな情報を年４回定期的に知らせる	１回につき50ドル
スタンダード検索		ある選定基準に基づき，あらかじめ SSIE 側で作成したテーマの検索	１件35ドル
研究者検索		研究者名から研究課題を検索	１名につき２ドル
受入番号検索		SSIE の受入番号による検索	１件１ドル

※　その他主題・機関名・研究者名からのそ及検索も可能である。

入手可能項目：主研究者、協同研究者名、研究機関、所在地、研究開始年、終了年、出資機関、研究のタイトル、要約と進行状況、研究費等

② ＣＲＩ（Current Research Information System、米国）

目的：ＣＲＩＳシステムは、農務省、各州の農業研究機関及び同省と協力関係にある研究機関の研究課題を検索。

参考：ＣＲＩＳシステムのデータは、ＳＳＩＥにも収録され、１９６９年から運営されている。

③ ＪＯＩＳ（ＪＩＣＳＴ、日本）

ＪＯＩＳシステム：ＪＯＩＳ（ＪＩＣＳＴ On-line Information System ）は、１９７６年７月より、ＪＩＣＳＴ（日本科学技術情報センター）が一般に公開しているオンライン文献検索サービスである。現在４つのデータベースで総計４００万文献の検索サービスを行っている。

ＪＯＩＳのクリアリング情報サービス：１９７７年６月より、ＪＯＩＳシステムの一貫としてサービスを開始している。

内容：国内約４００の公共試験研究機関における、現在進行中の研究課題ファイルで、理工学分野の研究課題約１５,０００件が収録されている。ファイルは、ＪＩＣＳＴにより作成されている。

検索機能：

(1) キーワード方式

(2) 分類コードによる方式

① 所在地コード

② 機関番号コード

③ 研究種別コード

④ 課題索引番号

④ RECRAS（Retrieval System for Current Research in the Agricultural Sciences── 農業試験課題機械検索システム、農林省、日本）〔NIPD７７〕

目的：研究者および行政関係者が農林水産関係試験研究機関の各分野にわたる実施中の試験研究課題を網ら的に把握することにより、試験研究をより効率的に推進し、かつその成果を有効適切に利用すること。

原情報：現在発行中の農林水産試験研究年報（農林省、農林水産技術会議事務局発行）

開発：昭和４７年度より農林水産技術会議事務局の試験研究課題機械検索システム検討委員会で、システム開発の基本的検討を行ない、システム開発の実務は、㈱日本科学技術振興財団（ＪＳＦ）が行なっており、昭和５２年度で一応このシステムの基本が完成している。

⑤ REGISTER（Retrieval System for General Information of Scientific and Technical Research── 科学技術研究情報検索システム、日本科学技術振興財団（ＪＳＦ）、日本）

目的：我が国のおもな理学、工学、農学、医学に関する研究機関で国立研究所、国立大学附属研及び研究施設、公社、公団、特殊法人の研究機関など（但し、大学学部を除く）で行なわれている研究課題から、利用者に特定主題についての情報源としての機関を紹介すること。

サービスの種類：クリアリングサービス、これにもとづく調査サービス及び公害技術リストの作成。

検索依頼：年間６０件

B．レフェラルサービス（情報源案内業務）

① NRC（National Referral Center、米国）

ＮＲＣは、ＮＳＦ（米国立科学財団：National Science Foundation）の援助で、１９６３年３月にＬＣ（米国立議会図書館：Library of Congress）の中に設置された。１９６７年以来現在は、科学技術部（Science and Technology Division）の中に位置づけられている。

目的：情報そのものでなく、情報を提供してくれる機関、人などを案内するサービスを行なう。

情報源の数：機関や専門家について、約１万件のデータを保有している（ＮＲＣマスタファイル）。なお、ＮＲＣマスタファイルは、１９７５年からＬＣのＳＣＯＲＰＩＯシステムの一データベースとしてオンラインアクセスが可能となっている。

対象分野：社会科学を含めた科学技術全般

対象機関：政府関係、企業体、大学、学術団体

特記事項：①問い合わせに対する回答は、国内外を問わず無料。

②利用者は、情報源が役立ったかどうか知らせる義務がある（フィード・バック・システムの採用）。

表４－２にＮＲＣの利用状況（１９６３－１９７４年）を示す。

表４－２　ＮＲＣ（１９６３－１９７４年）の利用状況〔ＭＣＦＡ７５〕

A comprehensive breakdown of 31,200 referral requests received by the National Referral Center for the period February 1963–February 1974.

I. Organization	Number of Requests	Percentage
*Congressional	91	.3%
*Library of Congress	646	2.1%
*Other Government Agencies	2,636	8.4%
*Department of Defense	1,368	4.4%
State Governments	588	1.9%
Local Governments	477	1.5%
Foreign Governments	1,155	3.7%
Colleges and Universities	4,825	15.4%
Other Schools	598	1.9%
Societies	218	.7%
Associations	424	1.4%
Commercial Firms	10,936	35.1%
Nonprofit Organizations	1,245	4.0%
Individuals	5,891	18.9%
Others	102	.3%
	31,200	100.0%
* Total U.S. Government.	4,471	15.2%

II. Occupation	Number of Requests	Percentage
Grade School Student	95	.3%
High School Student	669	2.1%
Undergrad Student	864	2.8%
Graduate Student	933	3.0%
Other Student	61	.2%
Grade School Teacher	60	.2%
High School Teacher	266	.9%
College Teacher	1,293	4.2%
Other Teacher	82	.3%
Engineer	3,154	10.1%
Scientist	1,600	5.1%
Librarian	8,373	26.8%
Editor	336	1.1%
Lawyer	159	.5%
Physician	224	.7%
Other Professional	4,879	15.6%
Administrator	3,809	12.2%
Technician	123	.4%
Clerical Personnel	291	.9%
Nonprofessional	147	.5%
Unknowns	3,782	12.1%
	31,200	100.0%

III. Broad Subject Area	Number of Requests	Percentage
Physical Sciences	3,200	10.3%
Biological Sciences	4,210	13.5%
Engineering/Technology	10,956	35.1%
Social Sciences	9,590	30.7%
Miscellaneous	3,244	10.4%
	31,200	100.0%

IV. How Received	Number of Requests	Percentage
Letter	15,576	49.9%
Telephone	11,095	35.6%
Visit	1,309	4.2%
Form	3,220	10.3%
	31,200	100.0%

V. How Answered	Number of Requests	Percentage
Letter	18,681	59.9%
Telephone	11,927	38.2%
Visit	592	1.9%
	31,200	100.0%

VI. Time Taken in Answering Requests	Number of Requests	Percentage
Same Day	7,150	22.9%
One Day	4,556	14.6%
Two Days	3,839	12.3%
Three Days	3,327	10.7%
Four Days	2,920	9.4%
Five Days	2,354	7.5%
Six Days	1,708	5.5%
Seven Days	1,236	4.0%
Eight Days	942	3.0%
Nine Days	654	2.1%
Ten Days	512	1.6%
Over Ten Days	2,002	6.4%
	31,200	100.0%

VII. Multiple and Repeat Requesters

Multiple Requests	1,044	Percentage	3.3%
Repeat Requesters	10,578	Percentage	33.9%

VIII. Feedback Analysis

Requests	31,200		
Feedback Sent	8,556	Percentage	27.4%
Feedback Received	3,662	Percentage	42.8%

Feedback	YES**	Q†	NO‡
Number of Returns	2,893	486	283
Percentage	79.0%	13.3%	7.7%

** The term "Yes" means "the information resource to which you referred us satisfied our inquiry."

† The term "Q" means "the response was qualified, or part of the question remained unresolved."

‡ The term "No" indicates that "no, the referral point did not solve our problem."

(24,146 requests were answered within the required five days. This represents 77.4% of all requests.)

② アジア太平洋地域のためのクリアリング機構（米国）

The Registry of Scientific and Technical Services for the Asia-rand Pacific Region

１９６６年の第一回アジア太平洋会議閣僚会議において提案され、１９６８年オーストラリアのキャンベラに事務所が開設され１９６９年６月より運用されている。

目的：アジア太平洋地域における科学者・技術者グループに関する情報のクリアリング・ハウスないしはレフェラル・センターとして機能を果すこと。

加盟国：オーストラリア、中国、日本、韓国、マレーシア、ニュージーランド、フィリッピン、タイ等

現況：現在進行中の研究課題に関するディレクトリーの発行……畜産関係、土木関係、産業・技術関係を出版している。

③ 日本における現状

我が国において、この種のクリアリング情報を専門に提供している機関はなく、次に示す資料類が情報源を調べるための有効なトゥールとなろう。

- 専門情報機関総覧 … 専門図書館協議会

  情報提供のサービスをしている機関を２,００６機関リストしている。

- 全国学協会総覧 … 日本学術会議

  日本における学協会1,０８０機関を主題別にリストしている。

- 全国各種団体名鑑 … ミカミマーケッティング・インスティテュート

- Directory of Information Resources in the Unitad States … National Referral Center

  ６シリーズから構成している。

- Guide to European Sources of Technical Information … Francis Hodgson

  欧州における情報源を調べるための資料で、情報がどこで提供され、どこから得られるか８５０以上の情報源を紹介。

- World of Learning … Europa Publications

  全世界的なダイレクトリで、１１４か国の官公庁研究機関、学協会、図書館、博物館、大学など２4,０００機関を収録

- 職業別電話帳（いわゆるイエローページ）

④ 国内で利用されている主な情報機関〔ＩＮＯＵ７７〕

表４－３は専門図書館協議会が専門情報機関を所有していそうな親機関4,１２２ケ所を対象に、昭和５０年４月に行なったアンケート調査の結果（回収率５５.７％、２,２９４通）を整理したものである。これからわかるように、今日我が国において利用されている外部情報機関は、ＪＩＣＳＴ、国立国会図書館、発明協会の上位三機関で、全体の４１％を占めている。

表４－３　主な外部情報機関（「専門情報機関総覧」からの集計による）　〔ＩＮＯＵ７７〕

機関の名称	票数	機関の名称	票数
ＪＩＣＳＴ	431	神奈川県立川崎図書館	4
国立国会図書館	87	㈱富士経済	4
発明協会	45	ＪＡＰＡＴＩＣ	4
日本規格協会	16	野村総合研究所	4
		日本技術貿易㈱	4
日本医学図書館協会	15	日本船舶振興会	4
中小企業情報センター	15	大阪商工会議所図書館	4
日本能率協会ＭＤセンター	15		
東京大学図書館	11	日本化学会図書館	3
		中小企業振興事業団	3
ＪＥＴＲＯ	10	福岡県文化会館図書部	3
アジア経済研究所	10	経済企画協会	3
経団連図書館	10	日本土木学会	3
国際医学情報センター	10	日本開発銀行中央資料室	3
政府資料等普及調査会	10	日本生産性本部図書室	3
日本医薬情報センター	10	日本新聞協会	3
		日本鉄鋼連盟資料室	3
九州経済調査協会	9	全国教育研究所連盟	3
電力中央研究所	7		
日本建築センター	6	ほかに：	
富山県立図書館	6	２機関から指定を受けたもの　４３機関	
三菱総合研究所	6	１機関から指定を受けたもの　４５２機関	
		計　５３７機関	
国際連合広報センター	5		
慶応大学理工学情報センター	5		
日本経済研究センター資料室	5		
名古屋大学図書館	5		
日本建築学会	5		

C．一次情報の提供

① ＮＴＩＳ（National Technical Information Service、米国）〔ＨＡＹＡ７６、７７〕

歴史：現在のＮＴＩＳは商務省に属しているが、次のような歴史をたどってきている。

i) １９４５年６月～

ＰＢ（Publication Board、出版局）の創設

ii) １９４６～６３年

ＯＴＳ（Office of Technical Service、技術サービス局）の設立

iii) １９６４～７０年

ＣＦＳＴＩ（Clearinghouse of Federal Scientific & Technical Information）、ＮＢＳに所属

iv) １９７０年９月～

ＮＴＩＳの設立、ＮＢＳから離れ、商務省内の独立機関となる。

サービスの概要：ＮＴＩＳで行なっているサービスは、次の三つに大別できる。

- 政府研究リポートの提供（ＡＤ、ＰＢリポート等）
- 抄録誌の発行（ＧＲＡＩ、ＷＧＡ）とその磁気テープファイル（アナンスメント）
- 機械検索サービス（NTI Searches … en-tee-search 等）

i) 政府研究リポートの提供

ＳＤＭサービス（Selective Dissemination of Microfiche）、ＳＣＩＭ方式（Selected Categories in Microfiche）を経て、現在はＳＲＩＭ方式（Selected Research in Microfiche）が採用されている。

対象：ＰＢリポート、ＡＤリポート

ii) 抄録誌の発行

ＡＤ―、ＰＢ―、のような具体的な番号が不明の場合、例えば、研究機関やタイトルしか判らない場合に、次の抄録誌を用いると便利である。

抄録誌：ＧＲＡＩ（Government Report Announcements & Index）

iii) 機械検索サービス（ＮＴＩ Searches）

- Published Searches … プロファイル方式
- On-line Searches

なお、ＮＴＩＳのデータベースは、Lockheed 社のＤＩＡＬＯＧ、ＳＤＣ社のＯＲＢＩＴの各オンライン・システムのデータベースとしても利用されている。ＮＴＩＳの検索サービスは、これとは独立している。

② 米国におけるＮＴＩＳ以外のシステム

現在米国で稼働しているオンライン情報検索システムは数多い。しかし、一次情報までサポートしているシステムは下記に示すとうり一部に限られている。

（一次情報をサポートしている運用機関／システム名）

ＮＬＭ／ＭＥＤＬＩＮＥ

ＮＹＴ／ＩＮＦＯＲＭＡＴＩＯＮ　ＢＡＮＫ

Stanford Univ.／ＳＰＩＲＥＳ

Council for Exceptional Children

LC／SCORPIO

NTIS／RECON … NTI Search

（一次情報はサポートしていない運用機関／システム名）

Battelle／BASIS

Lockheed／DIALOG

NC／STRC

SDC／ORBIT

Belfour Stulen Inc.／MPDC

CSC／INFONET

CDC／TECHNOTEC

Lawrence Radiation／WRISC

Lehigh Univ.／LEADERMART

Purdue Univ.／CINDAS

Rice Univ.／R.I.C.E.

Univ. of Dayton／CUDOS

Univ. of Iowa／IDIS

③ VINITI（USSR All-Union Institute of Scientific and Technical Information ―全ソ科学技術情報研究所、ソ連）

VINITIは、1952年にソ連科学アカデミーの下部機関として設立され、1955年全ソ閣僚会議の科学技術委員会とソ連科学アカデミーに同時に管理されるように再編成されたもので、世界一の規模をもつ情報サービス機関である。VINITIの情報活動は次のとうりである。

i) 科学技術のほとんどの分野（自然科学、応用科学、情報理論等）の世界の文献を網らする抄録誌の編集・発行。

ii) レファレンスサービス

iii) 科学技術情報処理の基礎的理論と応用に関する研究と実験

iv) 国内における他のセンターや図書館との協力（1970年段階でソ連には37万の図書館がある。）

v) 科学技術情報の蓄積、複写、翻訳、SDIサービス

vi) 図書館の機械化等における訓練及び国際協力

④ JICST（Japan Information Center of Science and Technology ―日本科学技術情報センター、日本）

JICSTはいうまでもなく日本における情報機関の中枢で、科学技術庁で検討中のNIST計画の具体化から、

i) 情報源の紹介

ii) 研究情報案内サービス

iii) 政府関係機関や国からの受託の下に実施された研究の報告の提供

等の業務が必要となり、昭和４７年（設立は昭和３２年）から実施にとりかかっている。なお、昭和５１年度における資料の収集状況は次のとうりである。

- 外国雑誌 …… 5,５００種
- 国内雑誌 …… 2,７００種
- 技術レポート ……４２,０００件
- 会議資料 …… ４００件
- 特許明細書 ……５７,０００件

これらの中から有用な文献を年間約３７万件抄録処理している。

以上、クリアリング機能について、クリアリングサービス、レフェラルサービス、一次情報の提供、の三分類を行ない、それぞれについて解説を行なった。

利用研、制度研究班の報告書〔ＲＩＹＯ７７〕によれば、クリアリング機能の、今後の適用分野として、次のものを掲げている。

- 現在クリアリング機能としては行なわれていないが情報ネットワーク又はデータベースによる情報提供を行なっている分野又は問題。
- 科学技術庁の２,０００年予測より重要課題の分野又は問題。
- 社会科学及び人文科学におけるクリアリングについての開発。
- 判例、白書、国連情報等の分野のクリアリング機能。

4.2 諸例にみるクリアリングセンターの構想

本節ではクリアリングセンターのイメージをより明確にするために二つの構想を紹介する。一つは、統計データの収集・利用研究会が昭和４６年度から３ケ年に渡って調査検討した統計データバンクにおけるクリアリングセンターであり、他の一つは、行政情報システム研究所が昭和４７年度から３ケ年に渡って調査検討した行政情報案内センターである。

4.2.1 統計データバンクに関する調査研究〔ＳＯＲＩ７４〕

総理府統計局で行政情報処理調査研究の一環として昭和４６年度から３ケ年に渡って行なわれた「統計データバンクに関する研究」を〔ＳＯＲＩ７４〕の要約から引用する。特に、クリアリングセンターの構想については本論から引用する。

A．研究期間：昭和４６年４月～昭和４９年３月

B．研究機関：統計データの収集・利用研究会（事務局：総理府統計局）

C．研究経緯

昭和４６年度 … 統計データの収集・利用に関して検討対象とすべき問題点の指摘を行なった。

昭和４７年度 … 前年度に指摘された問題点のうち特に主要な基本的事項として「ネットワークシステム」と「クリアリングセンター」及びその周辺的事項として「統計法制」と「各省庁窓口体制」について調査研究を進めた。また、これらを逐次行政ベースで進めて行くための方法として、関係省庁係官による窓口体制協議会の設置を提案した。

昭和４８年度 … ３ケ年にわたる調査研究を集約・体系化し、国における統計データバンクのあり方及びその維持・発展の手順等を総括した。

D．統計データバンクの概要

① 統計データ・バンクの必要性

近年、統計データに対する需要の増大、多様化に伴い、国が作成した統計データを刊行物で提供する方法では、十分な対応ができがたくなっている。しかも、国内で利用されるほとんどの統計データは国が作成供給するものであることから、これに応えることは国の責務であるといえる。更に電子計算機の普及、利用技術の進歩は統計需要に応え得る道を開いた。これらの要請に応えるため統計データ・バンクという新しい機能体が必要されるにいたっている。

② 統計データ・バンクの目的

統計データ・バンクの目的は、統計データに関して多様化する利用要請に的確に対応して提供活動を行うことである。また、統計データについて大量に取り扱うこと及びその正確性の保持並びに秘密保護を要する点から社会的任務を負うものであり、統計データ・バンクの活動は公共的性格を有する。

③ 統計データ・バンクの構成

ｲ) 分散型データ・バンクへの指向

利用者からみると、１か所で所要統計データを入手できる集中型データ・バンクが便利である。しかし、複雑、大量にわたる統計データを集中的に管理するために、人員の確保、技術的対応等には困難な問題が多い。

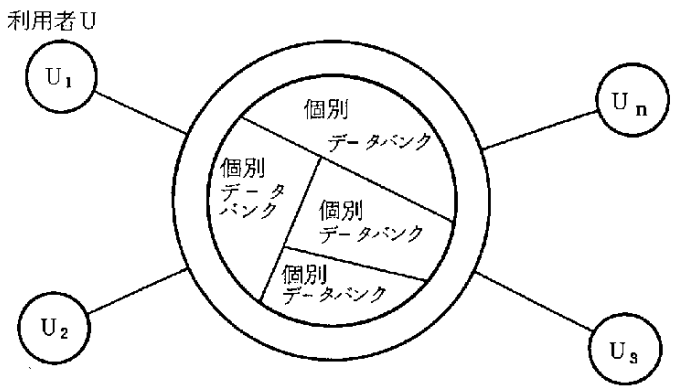


図４－１ 集中型データ・バンク

これに対し、分散型データ・バンクは、各省庁に統計データ・バンクを設け統計データの

正確性、利用者に対する責任の明確化等をはかり、利用者からみて不便な問題についてはクリアリング・センターという機能体を設け、ネットワークを形成して分散型からくる欠陥を補って、各省庁間の提供活動等の相互関係の調整をはかっていくのが実現性が高く、適当な方法であると考えられる。

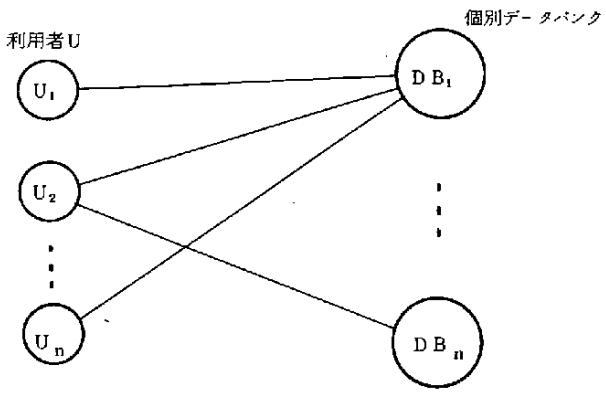


図4－2　分散型データ・バンク

ii）統計データ・バンクの構成

分散型データ・バンクでは、統計データ・バンクは「各省庁における統計データ・ベースを中心とした利用及び提供の機能体としての省庁別の個別データ・バンク」及び「個別データ・バンクの機能を向上させるための機能体であるクリアリング・センター」の2種類のもの、並びにそれらの間を連けいするネットワークで構成される。これを広い意味の統計データ・バンクとすると個別データ・バンクとクリアリング・センターは狭い意味の統計データ・バンクとなる。

④　ネットワークの形成

i）意義

ネットワーク体制は分散型の統計データ・バンクの欠陥を補完し、統計データの相互利用の促進等をはかって、統計データ・バンクの機能を総合的に発揮させ、統計データ・バンクにおける二重投資、競合の排除等につとめ、統計データの相互利用の効率化をはかろうとするものである。

ii）ネットワークのパターン

個別データ・バンクは担当する統計データについて収集、蓄積、提供のほか所在に関するデータの整備等を行い、一方、クリアリング・センターは、統計データの仲介・あっせんを行うほか、統計所在源情報の総合的提供等を行う。

また、狭義の統計データ・バンク間をネットワークで結び、統計データの相互利用の促進をはかる。

iii) ネットワークの形成

ネットワークが形成されているためには、狭義の統計データ・バンク間で統計データの経常的相互利用が行われていること、２つ以上の狭義の統計データ・バンクの連結があることが必要である。ネットワーク形成の充実段階では、ネットワークに参加を希望する個別データ・バンクがすべて参加し、統計データの利用者が所要データを入手できる状況になる。

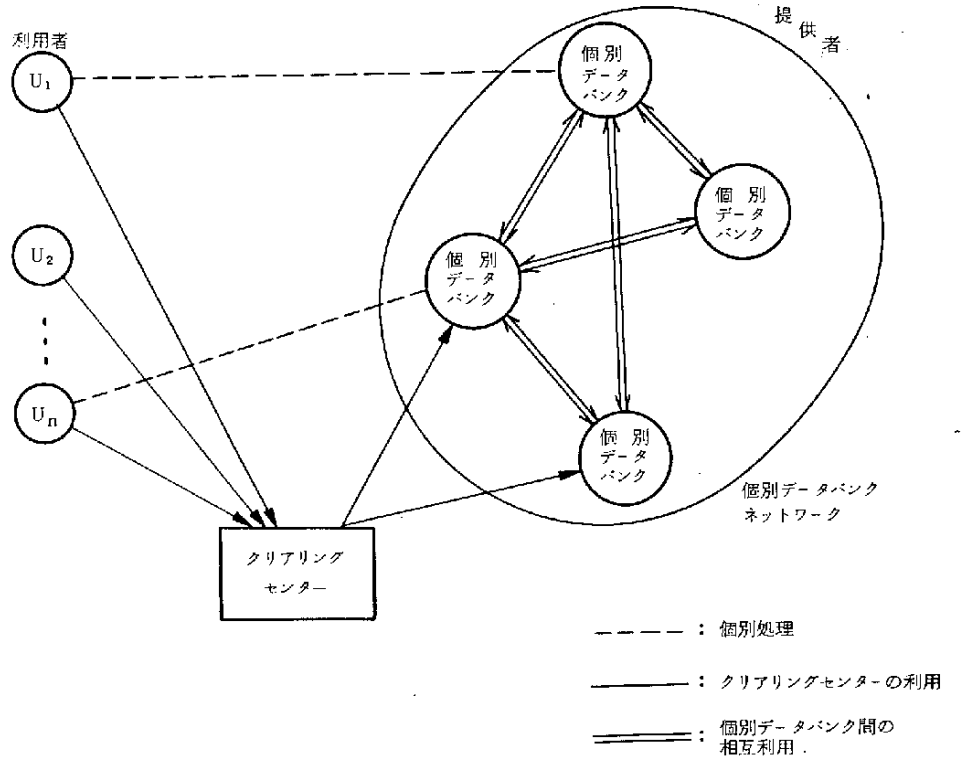


図４－３　統計データ・バンクネットワーク構成

⑤ 統計データ・バンクの機能

i) 機　　能

統計データ・バンク活動に必要な統計データの収集・整備等を行い、利用要請が多いが欠落している統計についてはその作成等を要請し、統計データとして、利用不十分なものの積極的活用をはかる。また、電子計算機による大量データの迅速・正確な処理機能を活用し、多様な記録媒体で統計データ提供活動の多様化をはかる。このために必要な各種の利用技術を充実する。

統計データの所在等に関する情報を収集整備し、照会等に対し提供する。

ii) 機能要件

統計データ・バンクは、統計データ・ベースを保有し、統計データの提供機能を備え利用者からの要請を受ける窓口機能を保有していることが必要である。狭義の統計データ・バン

クはこれに準じる。

⑥ 統計データ・バンクの業務

i) 利用者と統計データの種類

統計データ・バンクを利用する者は国の機関はもちろん、地方公共団体、民間等である。

統計データ・バンクが取り扱う統計データは、国が作成する結果データ、サマリー・データ、個別データ、参照用データなどのほか、国以外の機関が作成する公表データを主に対象とする。

ii) 統計データの収集・整備・蓄積・提供

個別データ・バンクは自省庁作成の統計データ及び自省庁業務に必要な外部機関作成の統計データを、クリアリング・センターは補完的機能を担当する立場で、個別データ・バンクが取り扱わない統計データを対象とする。

提供の内容は、磁気テープ・データのコピー、加工・分析・再編集結果等多様化をはかる。

iii) 統計の所在、内容情報等の整備・提供

利用者が指示した統計データの入手先、あるいは利用者がもつ課題に適合する統計データについての情報案内サービスを行う。

iv) その他の業務

統計データ等の仲介、あっせん、業務の代行、利用技術の開発等を行う。

⑦ 統計データ・バンクに関連するその他の事項

多様化する統計需要に対して、的確に対応していくため多方面からのニーズを把握しなければならない。

提供する統計データの正確性を維持し、利用者がそれを正しく理解して使用できるように、必要な資料の整備を行い、迅速な提供を行うようつとめるべきである。

また、統計データの時系列性、異なる統計間での比較整合性等、統計利用の拡大のために積極的に寄与していくことが必要である。

統計データ・バンクの活動を果たしていくうえで、被調査者の秘密の保護、責任の所在、権利の帰属等を明確にし、秩序ある活動を遂行しなければならない。

E. 個別データ・バンク

① 個別データ・バンクの意義と役割

個別データ・バンクは自省庁の業務上必要な統計データを収集し、一般的に省庁内利用に応える機能体として設けられるものである。更に、今後は国の役目として、統計データの利用の拡大をはかるため、自省庁所管統計データに対する他機関等からの利用要請に応えて迅速・的確に提供する機能も果たすものである。

② 業　務

i) 統計データの収集・整備・蓄積

個別データ・バンクが収集・整備・蓄積する統計データは自省庁所管のもの及び自省庁業

務に必要な外部機関作成のものである。ただし、外部機関作成の非公表データは、原則として蓄積しない。

外部機関作成の統計データの入手経路は、他の個別データ・バンク又は、クリアリング・センターから経常的にネットワークを経由する方法と直接入手する方法とがある。

ii) 統計データの提供

個別データ・バンクの提供サービスは蓄積してある統計データから必要なものを抽出して提供するレディ・メイド・サービスと利用者の注文仕様により加工等を行った結果を提供するオーダ・メイド・サービスがある。

提供の経路はネットワークを経由する方法と直接提供する方法とがある。

iii) 統計所在源データの整備・提供

自省庁所管統計データについて、その所在、内容等に関する情報を収集・整備してクリアリング・センターに提供し、また、他の利用者の照会に対して応える。

iv) その他の業務

個別データ・バンクは上記の基本的業務のほか、自己の組織を運営し、統計データ・バンクの機能の充実のため、統計需要の把握、その他必要な業務を行う。

F. クリアリング・センター

① クリアリング・センターの意義と役割

クリアリング・センターは、統計データ・バンクの機能を十分に発揮させるため、個別データ・バンクとともにネットワークを形成し、その構成単位となって個別データ・バンクの機能を補完し、いわゆる分散型体制からくる欠陥の一部を補うものである。

i) 意　義

個別データ・バンクが、外部機関作成統計データを収集し、あるいは外部機関等から個別データ・バンクが所管する統計データに対して利用要請があった場合、個別データ・バンクが個々にそれらの業務を処理することも可能ではある。しかしながら、統計データが増大しかつ複雑化する中で、この種の業務が増加し、複雑化してくるに従って、利用者と提供者との間に立って、仲介等を行ういわゆるクリアリング機能が必要となってくる。さらに、統計データの利用の円滑・迅速化等をはかるためには、仲介等のほかに統計データの名称、種類、所在等に関する情報を総合的に整備し、要請に応じて提供する機能が要求されてくる。これらの機能を有するものがクリアリング・センターであって、統計データの利用の拡大をはかるためには必要不可欠なものであり、統計データ・バンクの中において重要な役割を果たすものである。

ii) 役　割

クリアリング・センターは、個別データ・バンクの機能を補完しながら、全体としての統計データ・バンクの目的に奉仕するものである。したがって、個別データバンクと業務が重複し、個別データ・バンクの育成・発展を阻害するものであってはならないことはもちろん

であり、取り扱う統計データの種類についても、個別データのように個別データ・バンクが取り扱うことを適当とするものについては、クリアリング・センターでは取り扱わないものとすべきであろう。また、業務の競合等により、国の機関以外の者の利益をそこなうことのないような配慮が必要である。

② クリアリング・センターの業務

クリアリング・センターは、広義の統計データ・バンクの業務のうち個別データ・バンクが所掌しないものを遂行する任務を有する。その業務の性格が本質的に個別データ・バンクの機能を補完するものであることは、前述のとおりである。

i) 統計所在源情報の収集、提供

ア. 統計データを利用するためには、その前提として、データの名称、内容の概要記録媒体、所在、入手条件、入手方法等に関する正確な情報を得ることが必要である。また、行政施設等の資料として統計データを必要とする場合、それに適合したものを捜し出すことができる体制が必要である。ところが、統計データの増大、多様化に伴って、その内容等はますます複雑化し、これらの情報を迅速かつ正確に入手することが困難になりつつある。このため、クリアリング・センターにおいて、統計所在源情報を総合的に収集・整備し利用者の要請に応える体制を作る必要がある。

イ. 統計所在源情報の収集は、個別データ・バンクから所管統計データに関する情報の提供を受けて行う場合と個別データ・バンク以外の機関から情報を入手する場合がある。この収集した統計所在源情報の利用者としては、個別データ・バンクはもちろん、国の内外にわたる広範のものが規定される。

ii) 統計データ等の仲介・あっせん、業務の代行

a. 意　義

クリアリング・センターは、個別データ・バンクが他機関の統計データ又は電子計算機による統計データの利用技術を必要とする場合、あるいは利用者が個別データ・バンクの所管統計データ又は利用技術を必要とする場合において、自ら提供が可能なときは自ら行い、不可能なときは、仲介・あっせんを行って利用要請に応える必要がある。個別データ・バンクがこれらの業務を行うことは必ずしも不可能ではないが、かかる業務が増大し、複数の個別データ・バンクが関係するなど業務が複雑化してくると、個別データ・バンクにおいて処理することは著しく困難となり、また効率的なことでもない。

更にクリアリング・センターは、個別データ・バンク間における発展段階の不統一のため、広義の統計データ・バンクの機能が十分に発揮されない場合、発展のおくれている個別データ・バンクの業務の一部を必要に応じて代行する。

b. 内　容

クリアリング・センターが仲介・あっせん、代行の対象とするサービスとしては、次のものが考えられる。

ア．個別データ・バンクの統計データ又は利用技術の収集の仲介・あっせん。

イ．個別データ・バンクの保有する、統計データ又は利用技術の提供の仲介・あっせん。

ウ．個別データ・バンクにおける統計データの収集・整備・蓄積・加工提供の代行。

エ．その他広義の統計データ・バンクの業務のうち、個別データ・バンクの所業したいもので、クリアリング・センターで行うことが適当とみとめられるもの。

c．方　法

クリアリング・センターは、①仲介・あっせんのみを行う場合、②仲介・あっせんの対象となった統計データ等の受け渡しをする場合、③受け渡しに際して、加工等処理を行う場合などが考えられ、②及び③は一種の代行業務であるが、そのいずれを行うかは、当事者間の取り決めにしたがうべきである。

d．態　様

ア．統計データ等の仲介・あっせん、業務の代行は、臨時的なものと経常的なものとがある。仲介・あっせんには臨時的なものが多く、代行には経常的なものが多くなると思われるが、可能なかぎり業務を定形化し、その円滑化をはかるべきである。

イ．利用者から依頼を受けて業務を行う場合で、提供者の行為を必要とするときは、仲介・あっせんにたよるより方法がない。このため、クリアリング・センターは、常に仲介あっせんに必要な情報を的確に把握するなど利用要請に対し迅速・的確に応えられるよう準備しておくべきである。更に仲介・あっせん業務を経常化する場合、クリアリング・センターは、提供者が、あらかじめ統計データ等の提供を受けておくことも考慮すべきである。

ウ．個別データ・バンクから業務の一部の代行を依頼された場合、クリアリング・センターは当該個別データ・バンクの指示等を受けることとなるが、その際の責任関係等を明確にしておくことが大切である。（G項参照）

iii）その他の業務

クリアリング・センターは、上記の基本的業務のほかに、付随的に個別データ・バンクの機能の補完のため、また、自己の組織運営のため次の業務を行う。

a．統計データの収集・整備・蓄積・提供

ア．クリアリング・センターは、個別データ・バンクが整備されるまでの間、個別データ・バンクが取り扱わない統計データのうち利用頻度の高いものについて、収集・整備・蓄積・提供を行う。その際、収集先との間において、著作権等の問題が生ずる場合があるので、それらの処理を行っておく必要がある。この点については、G項において考察する。

イ．統計データの収集先としては、

- 個別データ・バンク以外の国の機関
- 地方公共団体、民間
- 外国、政府機関その他の外国の機関がある。

ウ．統計データの収集方法としては、

- 直接入手
- ネットワークを経由する入手（個別データ・バンクの収集依頼）

がある。

エ．統計データの提供先は、個別データ・バンクその他の国の機関が主たるものであるが、国の機関以外への提供も必要となろう。

b．利用技術の提供

クリアリング・センターが開発した統計データの利用技法及び電子計算機による利用技術で汎用性が高く、かつ、有益なものは他の利用者にとっても利用価値が高いので、この利用要請に対して、統計データの有効活用及び技術開発の重複投資を避けるため、応えるべきであろう。

c．そのほか、異種統計間における比較分析及び加工統計データの作成・提供、統計相談、クリアリング・センターが取り扱う統計データに対する需要の把握、クリアリング・センターの内部管理等の業務がある。

G．統計データ・バンク活動の維持・発展

① 統計データの詳細化等

統計データに対する需要の多様化は、統計データについて、統計分類の表章区分の詳細化、正確に理解するための情報の整備、提供の迅速化を要求している。このため、1)公表データの拡大、サマリーデータの活用等統計データの種類の多様化、2)磁気テープの活用等統計データ記録媒体の多様化、3)統計説明資料の整備等統計データ正確性の確保、4)統計データ提供の迅速化等をはかる必要がある。

② 秘密の保護

被調査者の秘密の保護をはかることは、回答の真実性確保等の見地から、統計データの利用面においても重要である。このため、秘密の範囲の画定等にあたっては、慎重な配慮を必要とすべきであり、また、統計データの公開は、被調査者の調査事項が個々に識別できないような形で行うべきであり、これらを担保するため、秘密保護の体制の充実をはからなければならない。

③ 統計データ・バンク活動に伴う権能、責務等

統計データ・バンクを構成する組織体が、公表データの収集・蓄積・加工することは当然自由とすべきであるが、非公表データの収集・蓄積・加工には一定の制限を設けるべきである。特に、個別データの蓄積、利用には秘密保護上の制限を設けるべきである。また、統計データの加工等処理に伴う責任と結果物の帰属及び統計データの再提供には問題があり、取り扱い上の原則、条件を明確にする必要がある。すなわち、加工等処理に創作性が認められる場合は、処理結果はその処理者に帰属し、また、統計データをそのままの形で再提供することを禁止するなどの原則を確立する必要がある。

④ 統計データ等提供に伴う料金問題

統計データに対する対価としての料金は、原則として徴収しないこととすべきであるが、加工処理等特別のサービス行為に対する対価としての料金は、実費弁償として、徴収すべきである。ただし、国の機関相互間においては、当面料金を徴収する必要はない。

H. 統計データ・バンク形成の手順

① 意　義

統計データ・バンクは他で作成された統計データを収集し、加工等して提供することから、統計データの作成機関の提供体制に制約される面が多い。

形成の手順を考察するには、更に個別データ・バンクの存在とその相互間の関係、国の財政支出能力等の問題も考慮することが重要である。

② 手　順

統計データ・バンクの機能はおおむね１０年以内に充実段階に到達することを想定し、各省庁の動向、現在の技術力等を勘案して、３期に大別した。

第１期では、個別データ・バンクの育成をはかり、クリアリング・センターは個別データ・バンクの機能を補完する。また、主要な統計データを国において利用できることとする。

第２期では、統計データ・バンクの機能水準の高度化をはかり、組織面での充実をはかるほか、取り扱う統計データの種類の拡充、利用技術サービスの多様化をはかり、国の機関はもちろん、地方公共団体からの要請にも応える。この間、民間の一部にも提供可能とする。

第３期では、統計データ・バンク機能の充実段階を迎える。利用者も民間等広範囲にわたり、複雑・多様な注文にも容易に対応できる。

③ 方　法

統計データ・バンクの形成を推進する１つの方法として、４７年度に研究会が提案した、各省庁窓口担当者による窓口体制協議会の活用がある。

4.2.2 行政情報案内センターに関する調査研究〔ＧＹＯＳ７３、７４、７５〕

A. 研究期間：昭和４７年４月～昭和５０年３月

B. 研究機関：（社）行政情報システム研究所

C. 研究協力：三井情報開発㈱

D. 研究目的及び背景：

① 行政機関の保有する情報の多角的かつ高度的な利用に対する要請が高まってきている。

② ①を阻害する要因

- データの所在が不明確である。
- 収集に際し手続きが複雑で時間がかかりすぎる。
- 事故防止・機密保護などの問題がある。

③ 「ＮＩＳＴ中間報告書」以来、官民を総括した情報流通ネットワークの必要性がとなえられ

ている。

④ 行政機関相互間のみならず民間の需要に対しても情報流通を促進するための方策の一つとして、行政機関及び関係団体等の保有する情報の所在を案内する機能をもつ「行政情報案内センター」を想定した。

E．研究項目：

昭和４７年度：「行政情報案内センター」の背景、役割、機能

昭和４８年度：情報源、関連諸機関との機能分担、所在情報の収集、提供システム

昭和４９年度：

① 省庁作成情報提供機関及び省庁外郭団体における調査研究情報の流通の実態把握

② 情報源の分類と情報源の開発計画の策定

③ 案内センター・サービス・システムの開発計画の策定

④ 事業化計画

F．行政情報案内センターの概要：

以下に、３年間に渡って検討された案内センターの構想の概要について述べる。（Dの研究目的及び背景と一部重複する。）

① 案内センター構想の背景と目的

一般不特定多数者が、省庁あるいは行政関係機関が作成する刊行物や統計などに対して、高いニーズを持ちながら、次の阻害要因が在るため、情報の流通が円滑に行われていないことが明らかになった。

- 一次情報の所在が不明確。
- 情報収集の際、入手手続きが煩雑であり、時間がかかる。
- 機密保護、データ保護についての制度上の整備がなされていない。

（統計データに関する問題点）

- 調査と結果の公表とのタイム・ラグが大きく適時性に欠ける。
- メッシュが不揃いで、一応性や完備性に欠ける。

以上の諸問題を解決し、特に行政（関係）機関が作成する情報を一般不特定多数に提供する場合の流通促進を目的として行政情報案内センターを構想した。

② 案内センターの取り扱い情報と媒体

取り扱い情報：統計データ、調査研究報告書

関係機関：省庁自身、政府関係機関ならびに調査研究を目的とした省庁認可公益法人

情報媒体：書籍文献類とコンピュータ入力媒体

対象分野：特に限定しない

（行政活動・産業活動を行うにあたって当案内センターが分野を限定することは、ニーズの多様化している現状から判断すると不適当と思われるため。）

③ 案内センターの収集ネットワーク

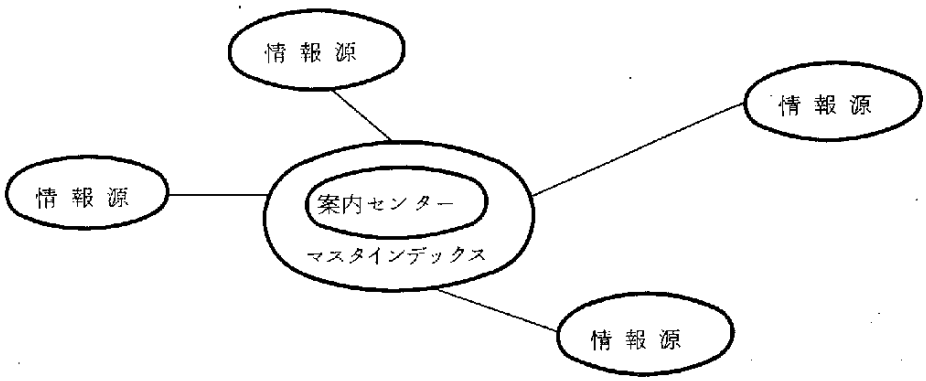


図４－４　案内センターと情報源

- 案内センターは一次情報を保有しない。（スペース、人員、資金的に困難、重複投資）
- 情報源との協力体制を確立して、案内サービスに必要なインデックス情報を提供する。

④ 案内センターのサービス機能

案内センターのサービス機能は、次の４機能より構成される。

- 所在情報提供サービス
- 情報斡旋サービス
- コンバージョン・サービス
- 加工サービス

i）所在情報提供サービス

案内センターの中枢的サービスで、情報源が保有する情報に関し、利用者の要請に応じて所在情報の提供を行う。

- 索引簿配布サービス
- 所在情報の検索サービス
  - ＳＤＩサービス
  - ＲＳサービス

ii）情報斡旋サービス

利用者に対する一次情報の斡旋。

- 案内センターが情報源から直接入手し、利用者に提供する方法
- 案内センターが、情報源に対して利用者の紹介状を出す

iii）コンバージョン・サービス

磁気テープ・データの利用に際して、利用者のコンバージョン作業を、案内センターがその作業を代行する。

Ⅳ) 加工サービス

磁気テープに対して、利用者側の要請によって、加工を行う。

- セレクト・サービス

  必要な情報を抽出する。

- サマライズ・サービス

  統計情報など、情報源のテープに対して、別の観点から集計する加工サービス。

- ハード・コピー・サービス

  磁気テープの内容をプリントする。

G．案内センター開発の効果と今後の課題

(1) 案内センター開発の効果

a) 利用者動向

行政機関の保有する情報の提供を行うため信頼性が高く、多様な分野にわたり正確かつ迅速に入手できるため利用者にとって、つぎのような効果が生ずると思われる。

① 情報の不足、不正確さが改善される結果、各機関（利用者）の研究開発・統計分析、経営計画・立案等の業務遂行に多大なメリットを与える。

② 利用者の要望に合致した情報が容易に入手できるため、従来の収集作業の煩雑さが解消される。

③ 本システムがトリガー的存在となることにより、業界、研究機関等の行政情報以外の流通が促進される。

b) 情 報 源

① 省庁情報提供窓口機関及び調査研究機関等（外部団体）は潜在ニーズの顕在化が計られ、より広い提供の機会や場所が与えられることにより、従来一般に提供できなかった問題が解決される。

② 特に調査研究機関においては機関相互の情報流通の促進が図られるため重複研究の防止や協同研究が促進される。

③ ①及び②より適切な事業計画の立案や業務の遂行を行うことができる。

④ 省庁図書館においては、従来の利用者が本センターへ移向するため、従来の業務である省庁職員のための利用、機密保護等の面で業務が簡便となる。

c) 情報流通に関する施策

① 情報流通促進機関（省庁を中心とした機関及び省庁外郭団体を中心とした組織）の役割りが拡大し相互のコミュニケーションが活発に行われることにより情報流通の諸施策が図られる。

② 情報流通の根源である施策として公開の基準、機密保護等の基盤の整備が促進される。

③ システム化が遅れがちな情報提供に関して先取り的あるいは整合性の高いものへと変化する可能性がある。

④ 磁気テープ提供へのトリガー的存在となり今後コンピュータリーダブルな流通が促進される。

(2) 具体化に際しての検討課題

a) 情報収集

① 行政情報を総合的に取り扱うことは本システムの存在価値を高めるが、具体的な取り扱いは情報源との間で現実的な仕様、作業分担及びそれに伴なう費用の負担などの観点から検討を要する。

② 本システムの最も望ましい情報源は省庁等の情報作成機関であるが収集の容易性、実現性の観点から提供機関をその対象とする。このため、情報の正確性、責任体制等を十分検討したうえで実施する必要がある。

③ 情報の収集は省庁をはじめ、提供機関等の協力が前提となる。したがって、これをいかにとりまとめるかが本システムの実現性を左右するといえる。そのため、開発主体は情報源のメリットが明確になるよう情報の活用方法を具体的に表示する必要がある。

④ 行政情報の分類化、体系化及びインデックスの内容等の標準化は本システムの基礎となるため実現性の乏しいものとならないよう開発主体は関連機関と十分協議し、調整指導を行いつつ具体化を計る必要がある。

b) 情報提供

① 情報の提供は経験と知識に富む係員との対話を通じて行う要素が特に検索サービスについては強いこと、利用者のニーズは専門かつ細分化されると同時に関連分野の情報までを要請するなどの点から提供に関してはそれらに対応しうる人材の確保が重要となる。

② 本センターの中心サービスは、2次情報の提供のため利用者の拡大、センターの意義を十分発揮するためには1次情報を効率よく提供しうる斡旋サービスを期をみて行う必要がある。したがって情報源との斡旋手続き、省庁での交換手続きなど関連機関との手続きに関する検討を行う必要がある。

③ 本研究では取り上げなかった行政に関する行事、催物、相談など個人を含めた情報提供に関しては、その流通実態を把握し、ニーズに合致した方法でサービスの1つとして組み入れ総合的な案内サービスを検討する必要がある。

(3) 国に対する要望

① 情報流通体制

国においては㋐各省庁内の情報化、㋑省庁間の情報流通、㋒データバンク構想などの情報交換に関する施策を進めてきている。しかしながら、これらのクローズなシステムにおいても情報流通、交換は十分に行われていない。このため、行政機関以外を対象としたオープンなシステムはまさに今後の課題といえる。

これら諸施策を実行するにあたっては、情報流通、交換に関する長期ビジョンの策定とともに情報公開の基準、交換手続きの統一化など情報流通における基礎的事項解決の体制づく

りを行う必要があろう。

② 財政面での援助

本システムは行政情報という公共的性格の極めて強い情報提供を行うため事業体の収支をある程度、度外視して考える必要も生じよう。したがって、資金的な制約からサービス内容が低下したり、システムが硬直化することがないよう国をはじめ各方面から長期的な視野に立った財政面での援助が期待される。

③ 基盤の整備

本システムは情報の網羅性、正確性、迅速性を確保することについては情報源となる外郭諸団体の意識に依存する面が強い。そのため外郭団体のあり方、特に情報提供のあり方について積極的に指導、育成を行うことが望まれる。

4.3 所在源情報管理の基本的技術（ＤＤ／Ｄ）

本節では、4.1および4.2で調査したクリアリングセンターの一機能である所在源情報に関する管理技術について検討する。つまり、コンピュータネットワーク上に分散しているリソース（ハードウェア、ソフトウェア）に関する情報の管理である。換言すれば、どのようなリソースがどこにあるかとか、あるリソースと他のリソースとの関係はどうなっているのか、とかいう問い合わせに応じられるために、どのような情報をどこに持つかという議論である。

ここでは、データに関するデータ（いわゆるメタデータ）の管理技術として代表的な概念ＤＤ／Ｄ（Data Dictionary ／ Directory ）について、その概要、商用ＤＤ／ＤシステムとＤＢＭＳとの関係、ＤＤ／Ｄ情報の管理形態について検討する。なお、今回のＤＤ／Ｄに関する一連の検討においては、ＤＤ／Ｄが誕生するまでの歴史的背景や、商用ＤＤ／Ｄシステムの詳細、ＤＤ／Ｄの構成要素、メタデータの名前付け規則、プライバシ、機密保護、保全性の検討は行なっていない。

4.3.1 ＤＤ／Ｄの定義

ＤＤ／Ｄとは、「データに関するデータ（メタデータ）を保有する貯蔵所（一種のファイル）である」と定義できよう。ＤＤ／ＤＳ（ＤＤ／Ｄ System ）と呼ばれる実体は、ＤＤ／Ｄとほとんど等しいものと思われ、管理機能を特に主張したいときには、このような表現が用いられる場合もしばしばみられる。しかし、ＤＤ／ＤとＤＤ／ＤＳは、ほとんど同義と解釈しても問題はないであろう。ＤＤ／Ｄの概要を把握するために、幾つかの論文からその定義を引用する。これらの定義の中には、ＤＢＭＳに直結したものに限らず、ある"物"の所在情報を管理するという意味では共通テーマと解釈して引用したものもある。また、今日では、ＤＤ／ＤがＤＢＭＳにとって欠かせないものとして把握されているが、ＤＢＭＳ以前から存在し、またＤＢＭＳとは独立した機能としても充分活用できる。さらに、標準化なし得ないものに対して一つの解答を提供する有効な道具とも考えられる。

A. Canning による定義〔ＣＡＮＮ７８〕

データディクショナリは、ある組織体のデータ定義事項を格納したり、操作するためのシステムおよび手続きをいう。このシステムや手続きはマニュアルであったり、自動化であったりする。さらにデータディクショナリの存在理由は、コンピュータに一切関係ない。

B. Schussel による定義〔ＳＣＨＵ７７〕

ＤＤ／Ｄは、データの定義や構造、利用に関する情報の貯蔵所であり、データそのものは含んでいない。ＤＤ／Ｄが保有する情報は、個々のデータ・タイプ（項目）名、定義（大きさと形式）、またデータの所在やデータの用法、他のデータとの相互関係などについてである。

C. Hotaka 等による定義〔ＨＯＴＡ７７ａ〕

ＤＤ／Ｄは、データベース自身の環境を記述する一種のデータベースである。つまり、ＤＤ／Ｄはメタ・データベースである。ＤＤ／Ｄを用いれば、データベースに含まれているファイルの物理構成や、ユーザあるいは応用プログラムとファイルの相互関係に関する情報が入手できる。

D. 酒井による定義〔ＳＡＫＡ７７、ＳＡＫＡ７６、ＪＥＩＤ７６〕

ＤＤ／Ｄは、メタ・データの集中貯蔵所である。メタ・データとは、データの属性に関するデータの意味で、データの名前、表現、構造、所在、使い方、さらに、データの意味、発生源、収集責任者、使用権、有効期限などを指す。そして、メタ・データを管理するシステムがＤＤ／Ｄである。

E. ＧＵＩＤＥ　ＤＤ／Ｄプロジェクトからの要望〔ＧＵＩＤ７５、ＪＥＩＤ７６〕

ＤＤ／Ｄは、ＤＢＭＳが知っている全てのデータ記述の組織化された収集である。ＤＤ／Ｄは、データベースに含まれるデータ単位に対するディスクリプタの完全な集合を含まなければならない。ＤＤ／Ｄは、ディクショナリとディレクトリとしての機能をもつ。つまり、データ単位を定義する全ての性質と、データ単位をアクセスするために必要な全部の情報を含むからである。さらに、データ単位間の関係と、データ単位とプログラム、処理、ＤＢＭＳ入力、出力との関係を記述する。

F. ＡＮＳＩ／Ｘ３／ＳＰＡＲＣ　Study Group による定義〔ＡＮＳＩ７５〕

ＤＤ／Ｄは、メタデータ・データベースである。ＤＤ／Ｄは、データベース記述（declaration）、プログラムにより参照されるデータベース・オブジェクト、データ・アクセスに関する利用統計、機密保護宣言、実行時における障害回復およびリスタートに必要な制御構造に関する情報の貯蔵所である。

G. Gradwell による定義〔ＧＲＡＤ７５〕

データディクショナリは、データベース設計集団やプログラミング集団に対するインターフェースをつかさどるデータベースチームが利用するためのインフォメーションバンクである。その情報は、常に最新である必要がある。最新でなければ、システムにとって単に歴史的な記録にすぎなくなってしまう。提供される報告書は、適時でしかも様式が整っていなければならない。さらに、データディクショナリは、個々のデータ項目に関する詳細な属性を維持する必要がある。

H. Thomas による定義〔THOMAS７５〕

データディクショナリという言葉は、過去３年（'７２～'７４）データ定義の集中化問題に興味がもたれ、このアイデアを記述するために導入された新語である。

I. Uhrowcz による定義〔UHRO７３〕

ＤＤ／Ｄは、他のデータとの関係、責任、発生源、用法、形式を定義するデータ記述に関する情報の集中貯蔵所である。

企業経営やデータベース管理者、システム分析者、アプリケーション・プログラマに対して、計画や制御を効果的にしたり、データ・リソースの利用に際して、補助的役割を果たすものでデータベース関係の中でも基本的な道具である。

J. Plagman等による定義〔PLAG７２〕

彼らは、統合化企業データベース（ＩＣＤＢ：Integrated Corporate Data Base ）の五大構成要素（データバンク、ＤＤ／ＤＳ、ＤＢＡ、ＤＢＭＳ、ユーザ／システムインタフェースの中で、ＤＤ／ＤＳを次のように定義している。ＤＤ／ＤＳは、データバンクに関する特徴、相互関係、権限などに関する全ての定義情報の貯蔵所である。データバンクとは、ある企業で用いる全データを論理的に集中化した貯蔵所である。

本項のまとめとしてＤＤ／Ｄの内容と構成について概説する〔UHRO７３〕

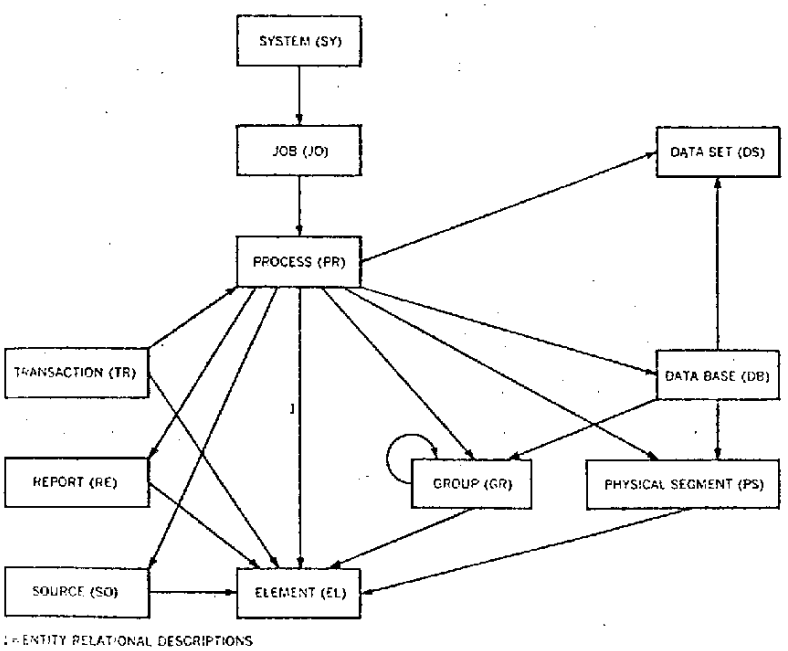


図４－５　ＤＤ／Ｄのエンティティ間の関係〔UHRO７３〕

アプリケーションシステムとサポートデータを定義するために必要な基本的エンティティについて、図４－５を用いて説明する。以下に述べる各エンティティは図４－６に示すような階層関係を

構成している。

Element（EL）…… プロセスが参照するデータの最小単位

Group（GP）…… 論理的に関連のあるエレメントおよび／もしくはグループの集合。（COBOLでいう論理レコード、IMSのロジカルセグメントに対応）

Physical Segment（PS）… 外部記憶装置に対するアクセス可能なデータの最小単位

Data Set（DS）…… 物理ファイルともいう。PSの集合。

Data Base（DB）… 1つ以上の関連したDSあるいはDBからなる。

Process（PR）…… 決められたデータ処理を行うための手続き。ジョブステップ、プログラム、ユニットプロシデュアとも言う。

Job（JO）…… 関連したPRのセット。

System（SY）…… JOの組み合わせ。アプリケーションシステムとも言う。

Transaction（TR）… 決められたPRあるいはJOを実行するための入力データのセット。

Report（RE）…… 人に提供される情報。例えば、印刷によるもの、CRT上に表示するものなどがその例である。

Source（SO）…… ソースドキュメント。

アプリケーションシステムにとって重要なエンティティである利用者など、必要に応じて定義する必要が生じよう。

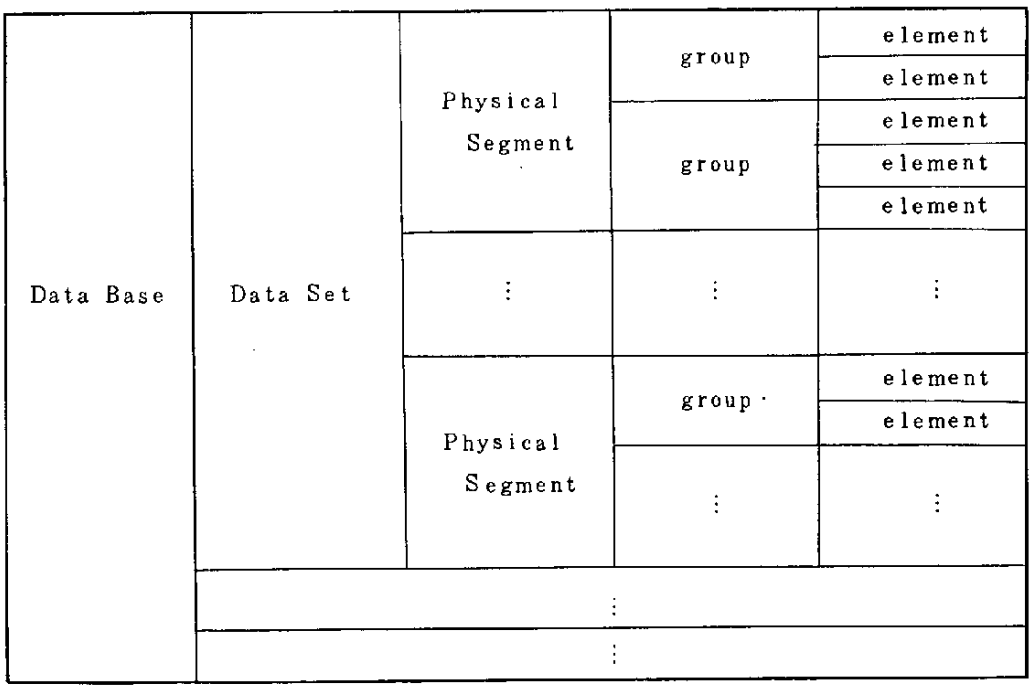


図4－6 データに関するエンティティ間の関係

4.3.2 商用DD／Dシステムの展望〔CANN78、SCHU77、SAKA77、JEID76〕

表4－5に代表的なDD／Dシステムの概要を示す。この表には、自社専用、自社用から商用へ発展したもの、最初から商用を目的に開発されたものと多種多様である。各システムの比較項目は次のとおりである。

表4－4　商用DD／Dシステムの比較項目

項　目　名	解　説
システム名（開発年）	DD／Dシステム名と開発年
開発機関	DD／Dシステムを開発した機関
DBMS結合	どのDBMSと結合が可能か
必要なDBMS	DD／Dシステムの運用に必要なDBMS
機密保護機能	不当なアクセスから守るための機能有無
オンラインアクセス	オンラインによる問い合わせ、更新機能有無
ディクショナリの入力は自動生成可能か	COBOLやDBMSのスキーマ、サブスキーマに含まれるデータ定義情報を自動的にコピーし、データディクショナリの入力情報が生成可能か
ディクショナリからデータ定義文が自動生成可能か	データディクショナリ内の定義情報から自動的にスキーマ、サブスキーマを生成したり、プログラム言語の定義文の生成が可能か（COPY機能）
単体利用の永久ライセンス料金	単体で利用した場合の永久ライセンス料金

表４－５　代表的な商用ＤＤ／Ｄシステム（システム名のＡＢＣ順）

システム名（開発年）	Control 2000（1977）	Data-Base Directory	Data Catalogue（1973）	Data Dictionary（1976）	Data Dictionary	Data Dictionary System（1970）	Data-manager（1975）	DB/DC Dictionary（IMS Dictionary System）（1976）	IDMS Dictionary	Lexicon	Pride Logik（1974）	"Three D" System（1973）	UCC 10 or UCC TEN or Data Dictionary／Manager（1970）
開発機関	MRI Systems	Easterm Air Lines	Synergetics Corp.	Cincom	British Rail	Caterpillar Tractor（自社専用）	Management Systems and Programming Inc.	IBM Corp.	Cullinane Corp.	Arthur Anderson and Co.	M. Bryce & Association Inc.	Anderson Clayton Foods	Univ. Computing Company
DBMS結合	System 2000	TOTAL	IMS TOTAL	TOTAL		IMS DL／1	IMS TOTAL IDMS ADABAS	IMS DOS DL／1	IDMS	IMS TOTAL IDMS	予定 TOTAL	IMS	IMS
必要なDBMS	System 2000（ライセンス必要）	TOTAL	無	TOTAL			無	IMS あるいは DOS DL／1	IDMS（ライセンス不要）	無		IMS	IMS
機密保護機能	有	有	有	有			有	無	無	無		有	無
オンラインアクセス	TP 2000	無	TSO（問い合わせのみ） CICS	無		予定	Intercomm Taskmaster ROSCOE IMS／DC CICS TSO CMS ETSS	IMS／DC（VS）	TSO IDMS Query	TSO	無	有	IMS／DC
ディクショナリの入力は自動生成可能か	予定		有	無	無		有	有	有	無			無
ディクショナリからデータ定義文が自動生成可能か	予定 COBOL PL／1 FORTRAN BAL		COBOL PL／1 BAL	無	COBOL		COBOL PL／1 BAL MARK Ⅳ	COBOL PL／1	COBOL PL／1	COBOL PL／1		FORTRAN	COBOL
単体利用の永久ライセンス料金	$15,000		$12,900～$18,900	$11,000			$9,900～$24,900	$290～$580（月）	$15,000	$10,000 Arthur Anderson社のサービスの一環として導入されている			$18,000

4.3.3 DD／DとDBMSとの結合形態〔HOTA77、JEID76〕

A．現　状

DBMSがデータの貯蔵所であるのに対して、DD／Dはメタデータの貯蔵所といえる。しかし、個々のDBMS内にもDD／Dに準じた機能が当然盛り込まれている。また、コンパイラやアセンブラで記述されたプログラムにはそれぞれのデータ定義部が保持されている。プログラム自身やDBMS、さらにはDD／Dというように同一情報を重複してもつことは種々の面で好ましくない。したがって、今後はDBMSを始めとしてこれら諸機能との関連において一元管理（共通の情報源）を前提とした、DD／Dの明確な位置づけを行う必要があろう。データの定義時から実行時までを統一した利用形態を次に示す〔UHRO73〕。

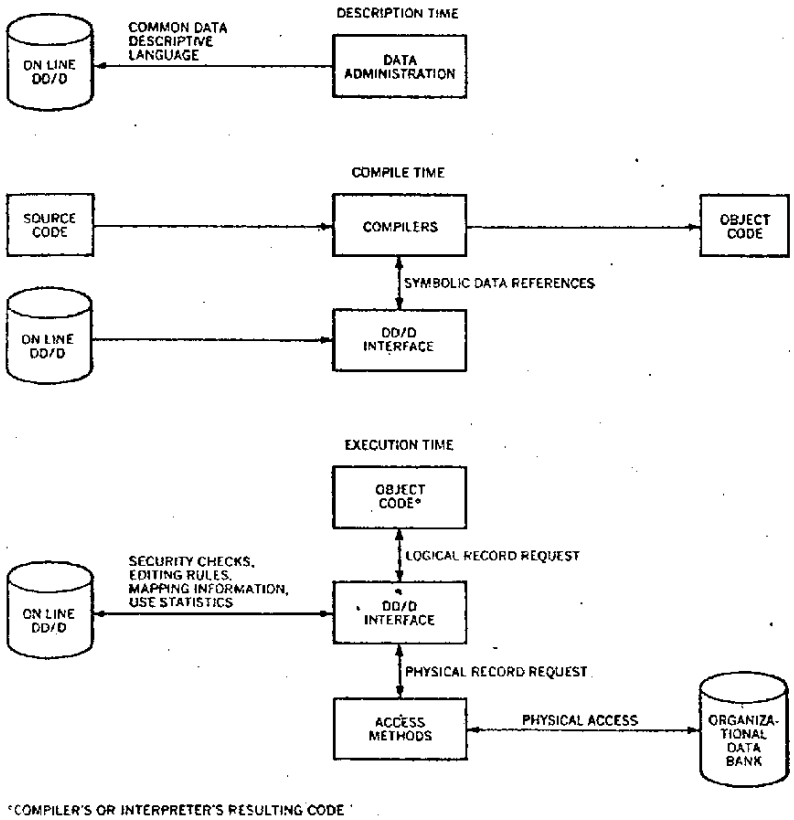
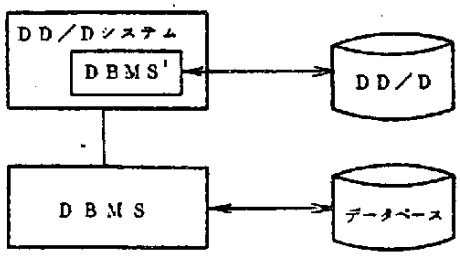
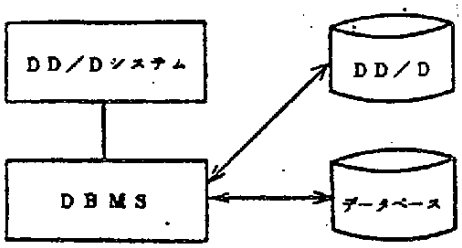
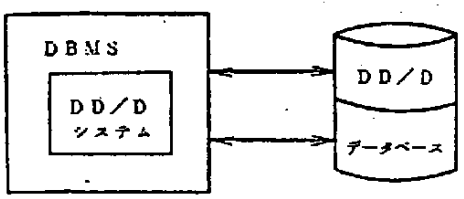


図4－7　DD／Dの仮定的利用〔UHRO73〕

次に現在提供されているDD／Dシステムが、既存のDBMSとどのような（従属）関係にあるかみてみる〔JEID76、HOTA77〕。

表４－６　ＤＤ／Ｄシステムとの関係

タイプ	形態とシステム例	概説および特徴
完全分離型	 • Lexicon • Datamanager	• ＤＤ／ＤシステムとＤＢＭＳは完全に独立しており、ＤＤ／Ｄへのアクセスも独自のアクセス法による。ＤＤ／ＤシステムとＤＢＭＳとのインタフェースが必要となる。 • ＯＳの下でアクセスするので、どのＤＢＭＳでも使え、ＤＢＭＳ導入以前にＤＤ／Ｄを利用できる。
部分的結合型	 • ＵＣＣ　ＴＥＮ • "Three D" System • ＩＭＳ Dictionary System	• ＤＤ／ＤシステムはＤＢＭＳから独立しているが、ＤＤ／Ｄへのアクセスは、共存するＤＢＭＳを用いる。つまり、ＤＤ／Ｄシステムは ＤＢＭＳを用いたアプリケーションの一つと言える。 ＤＤ／Ｄの定義、更新などはＤＤ／Ｄシステムによる。 • 特定のＤＢＭＳの存在を仮定しているためユーザの制限があるが、ＤＢＭＳの機能を充分に活用するようにしてある。
完全統合型	 • ＧＵＩＤのＤＤ／Ｄ要望 • ＡＮＳＩ／Ｘ３／ＳＰＡＲＣのデータベースシステムモデル • ＤＰＬＳ（千代田化工）	• ＤＤ／ＤシステムがＤＢＭＳの中に初めから統合されている。 • ＤＤ／Ｄへのデータ定義情報は、そのままＤＢＭＳに対する定義情報になるので両者間に矛盾は生じない

B．将　来

完全分離型や部分的結合型のＤＤ／Ｄシステムは、ディクショナリの管理が主体となり、ディレクトリ管理はＤＢＭＳに依存している。この結果、管理が二系統になり、メタ・データにおける冗長性（重複）などの問題を新たに発生させる。完全統合型においては、従来ＤＢＭＳに備わっていた機能を、ユーザ側に解放したものであるので、更新などに重複作業が発生しないなどの利点がある。ＧＵＩＤＥやＡＮＳＩ／Ｘ３／ＳＰＡＲＣの中間報告書もこの部類に入り、将来のＤＢＭＳはこの方向にあると言われている。

ディレクトリもデータの一部とみなしてデータベースを記述していく自己記述的ＤＢＭＳ（self-descriptive data base）が〔ＨＯＴＡ７７ａ〕で提案されているが、完全統合型の一例といえよう。

4.3.4　ディレクトリの管理形態

前項までは、ＤＤ／Ｄそのものに着目して、その定義や商用パッケージ、ＤＢＭＳとの関係について検討してきた。本項では、ネットワーク上に分散したデータベースを管理するディレクトリの管理形態について言及する。なお、ここではディレクトリに格納する情報やその情報のネーミング規則に関しては詳解しない。つまり、管理すべきディレクトリが集中しているのか分散しているのか、またディレクトリの内容に冗長性をもたすのかもたさないのかという、二点に着目したディレクトリの管理形態に関する議論である。

以下、幾つかの論文からディレクトリの管理形態について引用する。まず〔ＲＯＴＨ７７ｃ〕からディレクトリ管理の調査結果を示し、その後で関係モデルに基づいた分散型データベースシステム、ＩＮＧＲＥＳ〔ＳＴＯＮ７６、７７〕とＳＤＤ－１〔ＲＯＴＨ７７ｂ〕の管理形態について紹介する。そして、これら一連の研究の基礎となっているＣｈｕ のアイデアを紹介する〔ＣＨＵ７５、７６〕。最後にまとめとして、実現性のある管理形態について検討する。

A．Ｒｏｔｈｎｉｅ等による研究〔ＲＯＴＨ７７ｃ〕

彼らは分散型データベース管理の分野における研究開発の調査に関する論文で、ディレクトリの管理形態の分類を行っている。最近のＤＢＭＳにおける重要な機能として、データベース自身を管理するために定義するディレクトリ（しばしばスキーマとも呼ばれる）があると指摘している。そして、このディレクトリに含まれる情報としてまず、論理構造定義情報、物理構造定義情報、ファイル情報、会計情報の四つをあげている。さらに、分散型データベースについては、ディレクトリに情報を付加する必要がある。つまり、ネットワークのどこに、データベースの個々の部分があるかという情報である。これらの関係を図４－８に示す。

ＤＢＭＳは、ユーザの要求を解析したり、アクセス戦術の選択や実行したり、使用されたリソースの会計を行うためにこのディレクトリにアクセスする。従って、ディレクトリをどこに確保するのかという問題が新たに生じる。つまり、ディレクトリの管理形態に関する問題である。その際、論理構造定義情報や所在情報のように頻繁に用いられるものと、会計情報のように使用頻

度の低いものとは、区別して管理する必要があるとしている。

論理構造定義情報			物理構造定義情報			ファイル情報		会計情報			所在情報
関係名	領域名	その他	データ項目形式	インバーテッドキー項目	その他	ファイル容量	その他	ファイル使用者	ファイル保有者	その他	

図4－8　分散型データベースのためのディレクトリ構成

次に、ディレクトリ管理の分類について紹介する。彼らは、個々のディレクトリ項目が唯一の場所に格納するか幾つかの場所に格納するかで二別している。これを冗長／非冗長（redundant／non-redundant）による分類としている。さらに、それぞれについて、集中、分散、組み合わせ（centralization、distribution、combinations）の三形態について議論している。

ディレクトリの管理形態
- 冗長
  - 集中
  - 分散
  - 組み合わせ
- 非冗長
  - 集中
  - 分散
  - 組み合わせ

以下、六つの管理形態についてそれぞれ概要を示す。

表４－７　分散型データベースのためのディレクトリ管理(1)
（冗長によるアプローチ）

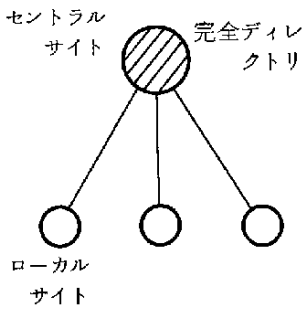
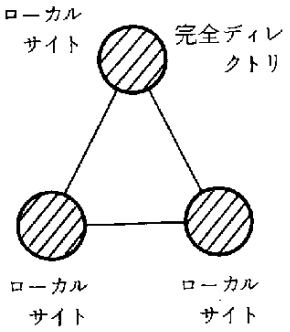
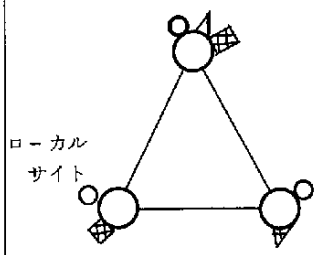
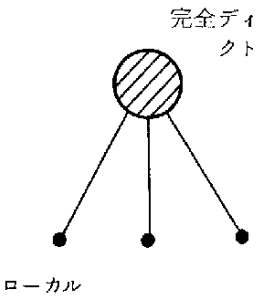
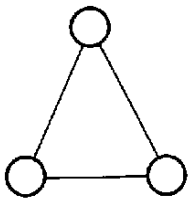
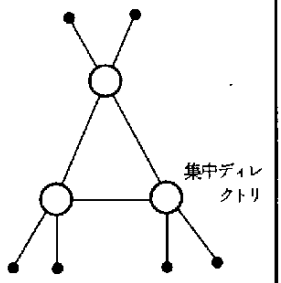
管理形態	集中型	分散型	組み合わせ型
形態			
解説	• 各ローカルサイトは自分自身のディレクトリをもっている。 • １つのセントラルサイトに完全ディレクトリをもつ。 • リモートデータの所在に関しては、セントラルサイトに全て問い合わせる。	• 各ローカルサイトにそれぞれ完全ディレクトリをもつ。 • 全ユーザはリモートに照会しないで実行の開始ができる。 • ディレクトリの更新情報を全サイトに知らせる必要が生じる。	• 多くの形態が考えられるが、上の例はその１つである。 • 上の例では、各ローカルサイトに自分自身のディレクトリを含めて、種々のサブセットをもたしている。 • この形態は柔軟性に富むがディレクトリのディレクトリが必要になる。つまり、どのディレクトリがどこにあるかということである。

表４－７　分散型データベースのためのディレクトリ管理(2)
（非冗長によるアプローチ）

管理形態	集　中　型	分　散　型	組み合わせ型
形態			
解説	• 完全ディレクトリが唯一のサイトにおかれる。 • ディレクトリに対する検索や更新は、全て完全ディレクトリのあるサイトへのアクセスが必要となる。	• 各ローカルサイトは自分自身のディレクトリをもつ。 • 完全にローカルなアクセスはローカルディレクトリのみで処理できる。 • リモートデータに対するアクセスが必要なときは、全サイトにブロードキャストする必要がある。	• ネットワークを幾つかの部分に分割する。 • 部分毎に集中ディレクトリを設定する。

※　この非冗長によるアプローチの大きな特徴は、ネットワークワイドでみても、ディレクトリの情報が決して重複しないことにある。

以上示した六つの管理技術からどれを選択するかは、システムに要求される要素により異なる（図４－９）。

① ディレクトリの検索が高頻度の場合
- 冗長方式
- 分散方式 … 多くの検索はローカルに対して行われるという前提

② ディレクトリの更新が高頻度の場合
- 非冗長方式
- 集中方式 … 分散しているとディレクトリの全コピーに変更を知らせなければならない。また、分散方式だと、情報の内容に関して同期をとる必要が生じる。

③ 高信頼性が要求される場合

- 冗長方式
- 分散方式

上記に示したのはあくまでも例にすぎないが、ユーザが要求する処理は、全て検索を伴なうので、冗長によるディレクトリの管理形態が望ましいと言えよう。

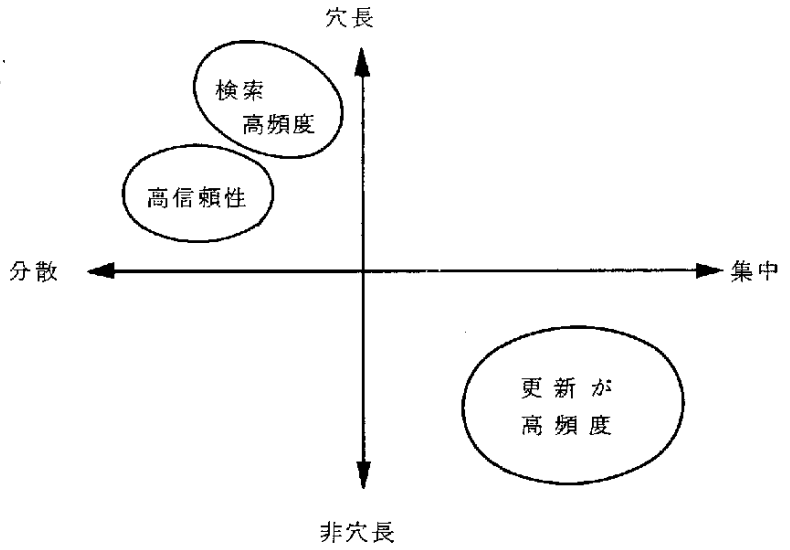


図４－９　ディレクトリの管理形態とシステムの要求

B. INGRESにおけるシステムカタログの研究〔STON76、77〕

INGRES（Interactive Graphics and Retrieval System）は、DECのコンピュータPDP11／40、11／45、11／70用にベル研究所で開発したUNIXオペレーティングシステム上に実現された関係DBMSである。

INGRESの詳細は、3.3節に紹介されているので、ここでは、コンピュータネットワーク上にある複数のコンピュータに分散したデータベースを管理するためのシステムカタログについてのみ紹介する。

INGRESのシステムカタログでは、個々のマシンが、物理的にそのマシンに存在する関係に関するシステムカタログを持たなければならないとしている。そして、各マシンが持つシステムカタログに登録される情報は図４－１０に示す四項目であり、このシステムカタログ自身も一つの関係として存在する。

関係名	解析情報			効率化情報			一致性情報		
	領域名	形式	その他	組数	蓄積構造	その他	保護	保全条件	その他

図４－１０　関係：システムカタログの概要

ＩＮＧＲＥＳは、自分自身のシステムカタログだけを持つことを原則としているが、他のマシン上にある関係に対するアクセス効率を向上させるために、オプション機能として、冗長によるアプローチを検討している。以下にその具体例を示す。表４－８に、システムカタログの基本型と五つの応用型を示す。応用型は、何らかの形で冗長情報をもっている。ここでいう冗長とは、複数のローカルマシンが関係名など四情報の一つ以上を共通にもつこととする。

なお、ＩＮＧＲＥＳでは、ＧＯＤを存在させない。理由は、ＧＯＤがパフォーマンスの隘路となることと、アクセスが更新に比べて検索が大半であることによる。また一つの関係が複数のローカルマシンに分割されていることをユーザに視せないため（ＩＮＧＲＥＳではこれを視点３という）には少なくとも応用型２程度の冗長度がパーフォマンス上必要である。

表４－８　ＩＮＧＲＥＳにおけるシステムカタログ(1)

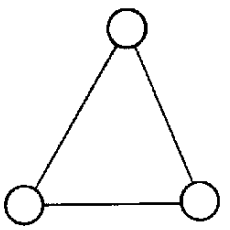
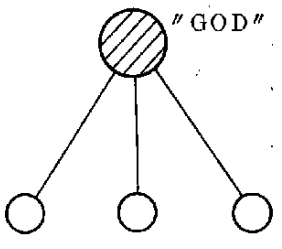
管理形態		基本型	応用型－１	
形態				
保有情報と特徴	関係名	・各ローカルマシンは、自分自身だけの情報を確保する。	○	・あるマシンを"ＧＯＤ"と設定する。そこに、ネットワーク上の全マシンに関する全情報を確保する。 ・各ローカルマシン自身の情報は各自で別にもっている。
	解析情報		○	
	効率化情報		○	
	一致性情報		○	
利点		・検索がローカルに集中しているときは、信頼性やスペース効率の面で有効である。	・他のマシンへの拡張コマンドが不要である。	
欠点		・他のローカルマシンへの問い合わせが多いときは、通信コストが大幅にかさむ。	・"ＧＯＤ"が障害を起こすとネットワークの機能が完全に停止してしまう。	

表4－8　INGRESにおけるシステムカタログ(2)

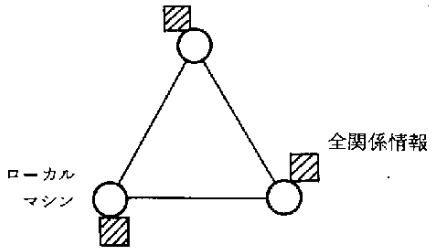
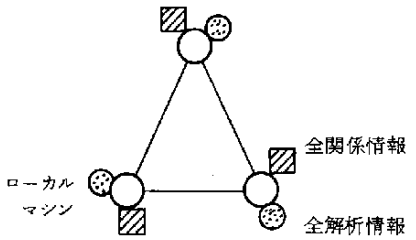
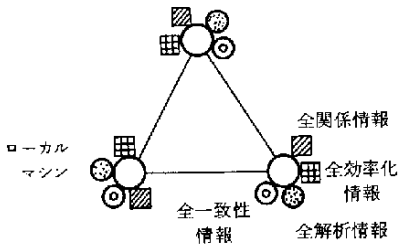
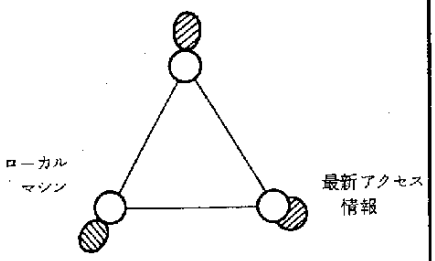
管理形態		応用型－2	応用型－3
形態			
保有情報と特徴	関係名	○	○
	解析情報		○
	効率化情報		
	一致性情報		
		• 全マシンに、ネットワーク上の全関係名を確保する。 • 各ローカル自身の情報は各自で別にもっている。	• 全マシンに、ネットワーク上の全関係名と全解析情報を確保する。 • 各ローカル自身の情報は各自で別にもっている。
利点		• 関係名に関する拡散が不要である。	• 解析情報を各ローカルサイトで保有しているので、リクエストが発生したサイトで処理できる。
欠点		• 各サイトのカタログ内容を最新情報に保つためのオーバ・ヘッドが大きい。	• 効率化情報と一致性情報に関しては、リモートサイトからの問い合わせがある。 • これらの情報の更新に関してはオーバ・ヘッドが大きい。

表4－8　INGRESにおけるシステムカタログ(3)

<table>
<tr><td colspan="2">管理形態</td><td colspan="2">応用型－4</td><td colspan="2">応用型－5</td></tr>
<tr><td colspan="2">形態</td><td colspan="2"> 
</td><td colspan="2"> 
</td></tr>
<tr><td rowspan="4">保有情報と特徴</td><td>関係名</td><td>○</td><td rowspan="4">・全マシンに全情報をもつと説明があるので、結局全マシンが"GOD"になると解釈できる。
・詳細な最新情報を維持するためのコストは、検索集中型のシステムであれば充分受け入れられるであろう。</td><td>○</td><td rowspan="4">・一度アクセスした他のマシンの関係に関する全情報を、ローカルシステムカタログに確保する。
・各ローカルマシンが保有している最新アクセス情報は、一定時間経過後棄却される。</td></tr>
<tr><td>解析情報</td><td>○</td><td>○</td></tr>
<tr><td>効率化情報</td><td>○</td><td>○</td></tr>
<tr><td>一致性情報</td><td>○</td><td>○</td></tr>
<tr><td colspan="2">利点</td><td colspan="2">・応用型－3の発展形で、実行に関する解析と決定のためのトラフックはなくなる。</td><td colspan="2">・システムカタログの管理にワーキングセットの技術を導入しているため、再使用の場合には、高速でアクセスが可能である。</td></tr>
<tr><td colspan="2">欠点</td><td colspan="2">・全てのローカルマシンがＧＯＤであるための全情報の一致性を保つためのオーバ・ヘッドが大きい。</td><td colspan="2">・各ローカルマシンが保有している最新アクセス情報の更新は行われないので、不正確な情報を使用する場合もある。</td></tr>
</table>

C．ＳＤＤ－１におけるディレクトリ管理の研究〔ＲＯＴＨ７７ｂ〕

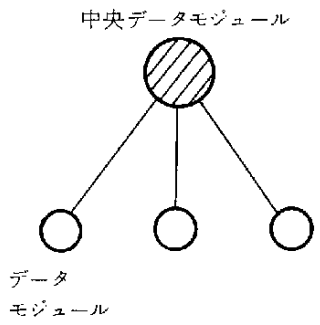
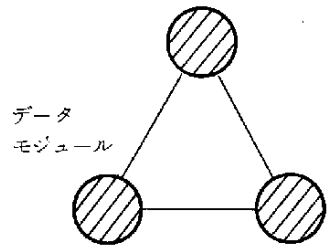
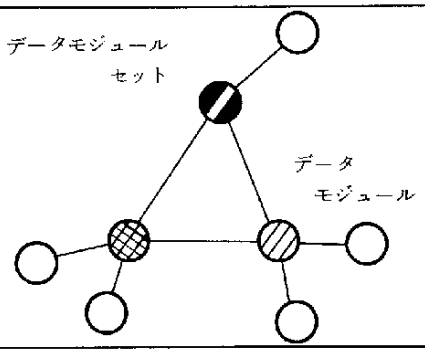
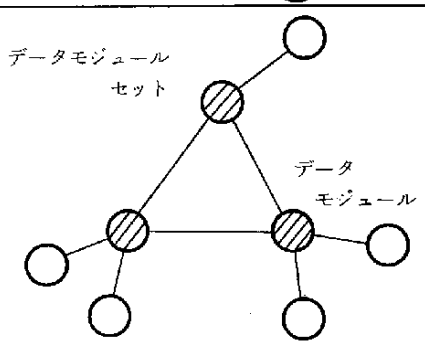
ＳＤＤ－１（Ａ System for Distributed Database ）は、ＣＣＡ（Computer Corporation of America）で設計・実現されている分散型データベースシステムである。ＳＤＤ－１は、データモジュール（datamodule ）と呼ばれるコンポーネントのセットで構成されている。このデータモジュールはＧＤＭ（global data manager ）とＬＤＭ（local data manager ）の二つの内部モジュールに分けられる。ＧＤＭは、ユーザの要求を解析したり、データの割り付けなどを行う。ＬＤＭは、ローカルに蓄積されているデータに対するアクセスを行うものである。

ＳＤＤ－１の詳細は、3.3節に紹介されているので、ここでは、ＳＤＤ－１におけるディレクトリ管理を中心に議論する。分散型データベースシステム設計上の重要な問題点の一つに全体システムディレクトリの管理がある。ＧＤＭがアクセスするディレクトリには、ユーザの要求を解析（分散）したり、適当なデータを割り付けたり、アクセスや更新戦術の選択に関する情報が含まれている。ディレクトリの管理形態としては、四つが用意されている（表４－９）。

どの管理形態を採用するかは、ネットワーク内で処理されるトラフィックの状況による。理想的には、汎用の分散型データベース・システムを構築する際、ディレクトリの管理形態は固定的にするのではなく、データベース設計の一部として選択の余地を残すべきであろうとしている。

ＳＤＤ－１におけるディレクトリ管理の大きな特徴の一つは、ディレクトリ・データを通常のユーザ・データと同レベルで取り扱っていることである。ディレクトリをユーザ・データと同一に扱えないという見方がある。つまり、ディレクトリ自身を管理するディレクトリ情報をどこに置くかという問題があるからである。言い換えれば、分割されたディレクトリの部分がどこのデータモジュールに在るかという所在情報を、システムがいかに管理するかということである。ＳＤＤ－１では、ディレクトリに関するディレクトリ情報を分割して、データモジュール内の個々の関係の中に組み込むことにより実現している。これは、この種のディレクトリ（関係の一つ）サイズが小さくて、検索集中型であるという前提にたっている。このように、ディレクトリをユーザ・データの一部として管理しようとするアプローチが他にも見られる〔ＨＯＴＡ７７ｂ〕。

表４－９　ＳＤＤ－１で検討されているディレクトリ管理の４形態

	形　　態	解　　説
①		中央のデータモジュールにだけディレクトリを置く。
②		各データモジュールに、それぞれ完全なコピーを置く。
③		データモジュールのセットに対して、非冗長にディレクトリを分割する。
④		データモジュールのセットに対して、若干の冗長性を伴なって、ディレクトリを分散させる。

※　ローカルのデータモジュールは、自分自身の情報（ディレクトリ）を、①～④のどの形態でも、常にもっているものとする。

D. Chu 等による研究〔ＣＨＵ７５、７６〕

彼等はまず、ファイルディレクトリを「コンピュータネットワーク上に分散されたデータベースの利用者にとって有効なファイル情報リスト」〔ＣＨＵ７５〕と定義した上で、分散型データベースのためのファイルディレクトリの設計技術に関して次のような提案を行っている。つまり、各ローカルサイト固有の情報だけをもつローカルディレクトリと、ネットワークの全体情報を管理するマスタディレクトリを設定し、五種類の管理形態について利害得失を論じている〔ＣＨＵ７６〕。

- ファイルディレクトリシステム
  - 集中型ファイルディレクトリシステム
    - 単一マスタ型
    - 拡張集中型
    - 多重マスタ型
  - 局所型ファイルディレクトリシステム
  - 分散型ファイルディレクトリシステム

図４－１１　ファイルディレクトリシステムの分類

なお、実際にどの管理形態を採用するかは、さらに次の各項目について充分な検討が必要とされている。

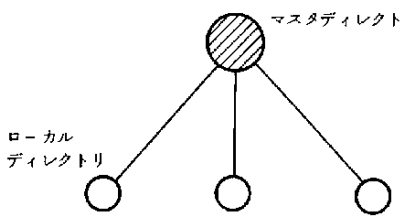
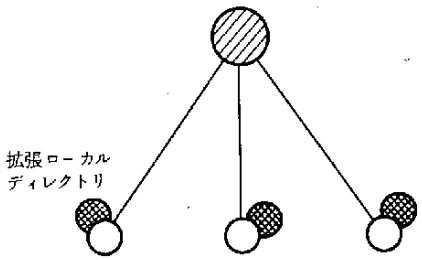
- 通信コスト
- 蓄積コスト
- ディレクトリサイズ
- コード変更コスト
- ディレクトリ問い合わせ率（頻度）
- ディレクトリ更新率（頻度）
- ディレクトリ応答時間

Chu は、ファイルディレクトリシステムに関する一連の研究の成果として、ディレクトリの更新率（問い合わせに対する）を基に、各種の管理形態について比較している。なお、前提条件として、転送コストが蓄積コストが大幅に上回る状況を設定している。

更新率が１０％以下の場合では、分散型が集中型より低い運用コスト（operating cost）ですむ。更新率が１５％を超える場合は、集中型が分散型や集中型より運用コストの面で優れている。一般に、更新率が50％以下のように低い場合は、拡張集中型が集中型（単一マスタ型）より運用コストの面で優れている。また、更新率が５～１０％と特に低いときには、拡張集中型にすべきであると、 Chuは主張している。さらに、分割されたネットワークで更新率が５～１０％以上の場合には、分割された個々のクラスタにマスタディレクトリを設定する多重マスタ型が、運用コストと応答時間の両面で、拡張集中型より優れている。局所型では、ローカルサイトで保有していないファイルをサーチするために大量の通信コストが必要となり、集中型に比べて運用コストが相当高い。更新率が５０％のように特に高くて、通信コストが蓄積コストより低い場合は、局所型が分散型より優れている。局所型は、検索時間や通信量の点に問題があり、運用コストの減少や応答時間の短縮を達成するようルーティング戦術に関しては充分配慮されなければな

らない。分散型や拡張集中型は、応答時間の点で、集中型や局所型より優れている。また、多重マスタ型のときには、ユーザからの問い合わせは分散されるから、単一マスタ型に比べて応答時間が短い。

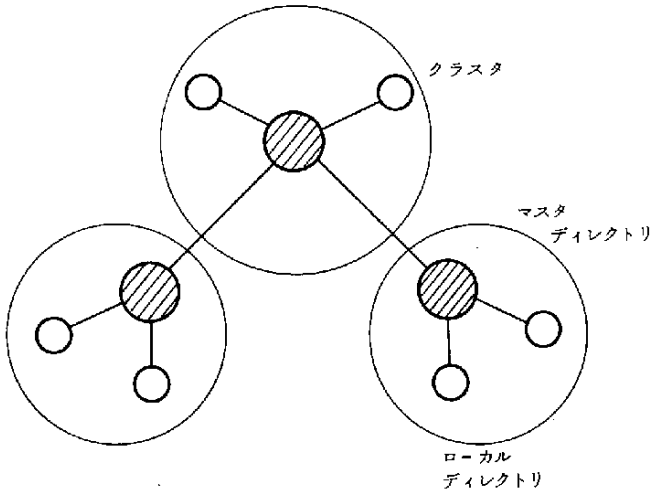
表４－１０　分散型データベースにおけるファイルディレクトリシステム(1)

管理形態	集中型ファイルディレクトリシステム	
	単一マスタ型*	拡張集中型**
形態		
概要	① マスタディレクトリはただ１つである。 ② 各ローカルサイトは固有の情報をもっている。 ③ ローカル以外のファイルにアクセスする場合は、マスタディレクトリへ問い合わせる。 ④ マスタディレクトリは、ファイルの所在や内容に変更が生じたときには更新する必要がある。	① 単一マスタ型の変形である。 ② ローカルサイトのユーザがマスタディレクトリを用いてファイルの所在を確認したときには、その都度その情報を、各ローカルディレクトリに格納する。 ③ マスタディレクトリに更新があると拡張ローカルディレクトリにも更新の必要が生じる。
利点	更新が容易	マスタファイルディレクトリへの問い合わせに関する通信コストが減る。
欠点	ローカルサイト以外へのアクセスが多いと通信コストがかさむ。	マスタディレクトリと拡張ローカルディレクトリの内容に関する同期が必要。

* single master directory

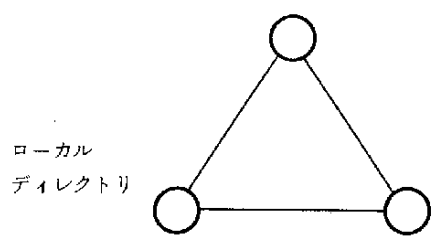
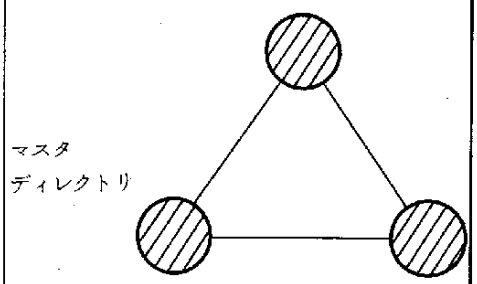
** extended centralized directory

表４－１０　分散型データベースにおけるファイルディレクトリシステム(2)

管理形態	集中型ファイルディレクトリシステム 多　重　マ　ス　タ　型*
形　態	
概　要	①　コンピュータネットワーク上にあるｎ個のコンピュータ（ローカルサイト）をｒ個（ｒ≦ｎ）のクラスタに分割する。 ②　分割方法には次の二通りが考えられる。 i）ネットワークトポロジに基づいて、通信コストが最小になるように分割する。 ii）ディレクトリに対する問い合わせや更新に対して、最小の応答時間で遂行できるように分割する。 ③　多重マスタ型システムにおける通信コストを節約することは、ファイルディレクトリの格納および更新コスト以上に重要視する必要がある。 ④　ここでのマスタディレクトリは、クラスタ内の情報だけでなく、ネットワーク全体のマスタである。
利　点	単一マスタ型に比べて応答時間が速い。
欠　点	ディレクトリの内容の変更が頻繁にある場合は、マスタディレクトリ間の情報の一致性と通信コストが問題になる。

* multiple master directory

表４－１０　分散型データベースにおけるファイルディレクトリシステム(3)

管理形態	局所型ファイルディレクトリシステム*	分散型ファイルディレクトリシステム**
形態		
概要	① システムにはマスタディレクトリがない。 ② ローカルディレクトリにないファイルにアクセスする場合には、他の全ローカルに問い合わせる必要がある。	① システム上の全ローカルサイトにマスタディレクトリをもつ。
利点	更新が容易	応答時間が速い
欠点	① 通信コストが高い。 ② 応答時間が遅い。	① マスタディレクトリを格納するためのコストが著しく高い。 ② ディレクトリ更新のための通信コストが高い。

* localized directory

** distributed directory

E．管理形態のまとめ

本項では、ディレクトリの管理形態について、幾つかの論文を引用して基本的なパターンの考察を行った。ディレクトリが持つ情報は、単なるリソースの所在情報から、スキーマ／サブスキーマに相当するデータベースの物理構造や論理構造に関する情報、会計情報、さらには他のディレクトリの所在情報、ディレクトリ間の一致性情報、機密保護に関する情報など広い範囲に渡っ

ている。したがってこれらのディレクトリ情報を分散してもつか集中してもつか、また冗長性を持たせるか否かについての議論を画一的に行おうとするのは若干無理が生じよう。つまり、情報の内容（種類）による管理も充分配慮すべき点である。ディレクトリに対するアクセス時の配慮も必要となる。例えば、使用頻度の高い情報や政策上防御すべき情報を考慮した管理も必要である。

このようなことから分散型データベースシステムにおけるディレクトリの管理形態は、構築しようとしているシステムが持っている主要な目的や規模、利用形態などについて十分な調査・分析を行った上で、最も適当と思われる形態を選ぶ必要があろう。

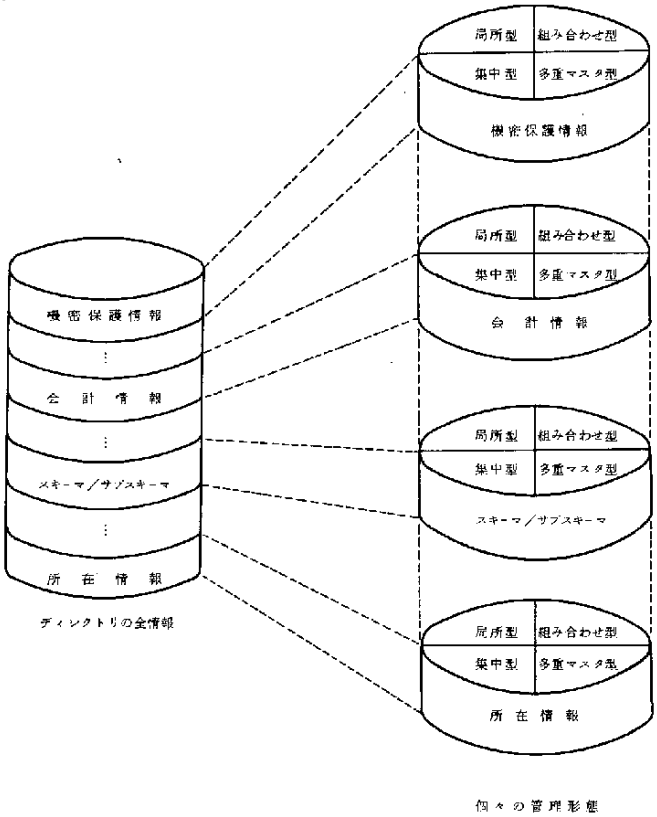


図４－１２　ディレクトリ情報とその管理形態

図４－１２は、ディレクトリが持つ情報の内容により、管理形態が異なりうることを表わしている。ここで考えているディレクトリの管理形態は次の四つである。

局　所　型
組み合わせ型
集　中　型
多重マスタ型

これら何れの管理形態も、自分自身の情報は自分で保持しかつ管理することを原則としている。四つの管理形態の概要は次のとおりである。

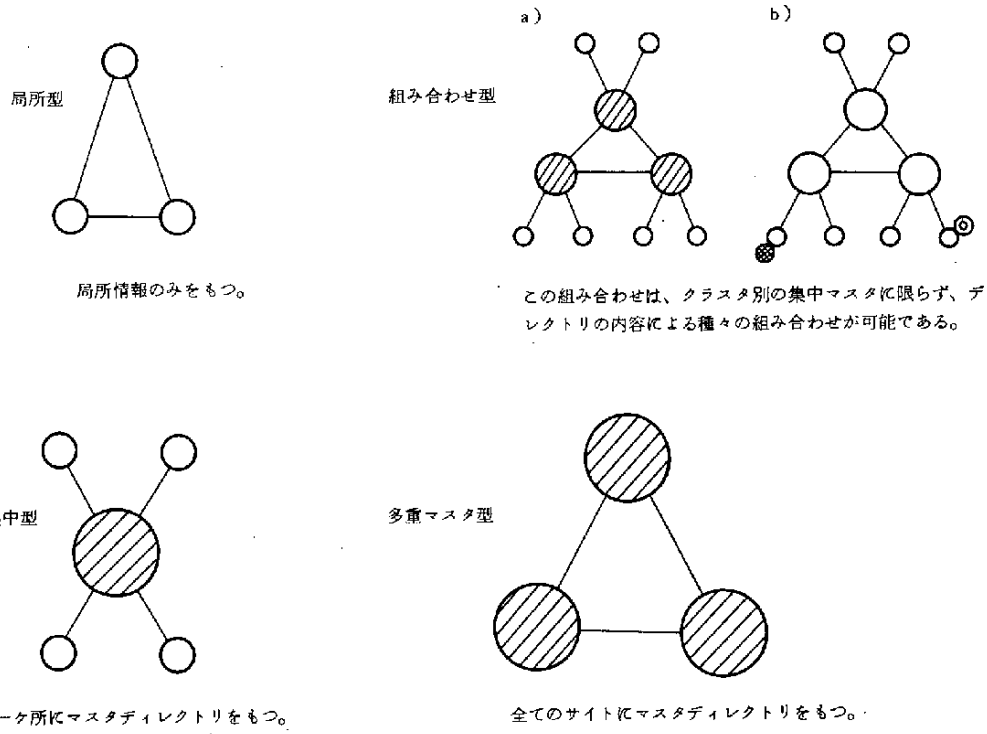


図４－１３　ディレクトリの代表的な管理形態

図４－１４は、四つの管理形態に関して、五項目について得失を示したものである。比較した項目は次のとおりである。

① 経済性 … ディレクトリサイズに基づいた蓄積コストについて

② 障害に対する信頼性 … あるサイト、特にマスタディレクトリのサイトに障害が起きたときの影響について

③ ディレクトリの更新頻度が高い場合の効率について

④ ディレクトリに対する検索の局所性が極めて高い場合の形態について

⑤ 他サイトにあるディレクトリへのアクセス頻度が高い場合の応答性について

これらは、図４－１２に示したように個々のディレクトリ情報やシステムの目的に十分照らし合わせて、最適なディレクトリの管理形態の設計が望まれる。

例えば、機密保護情報については一カ所集中型で、会計情報については利用者の階級別に分散し、ディレクトリのディレクトリに関する情報は全サイトにもつとかといった組み合わせも充分考えられよう。

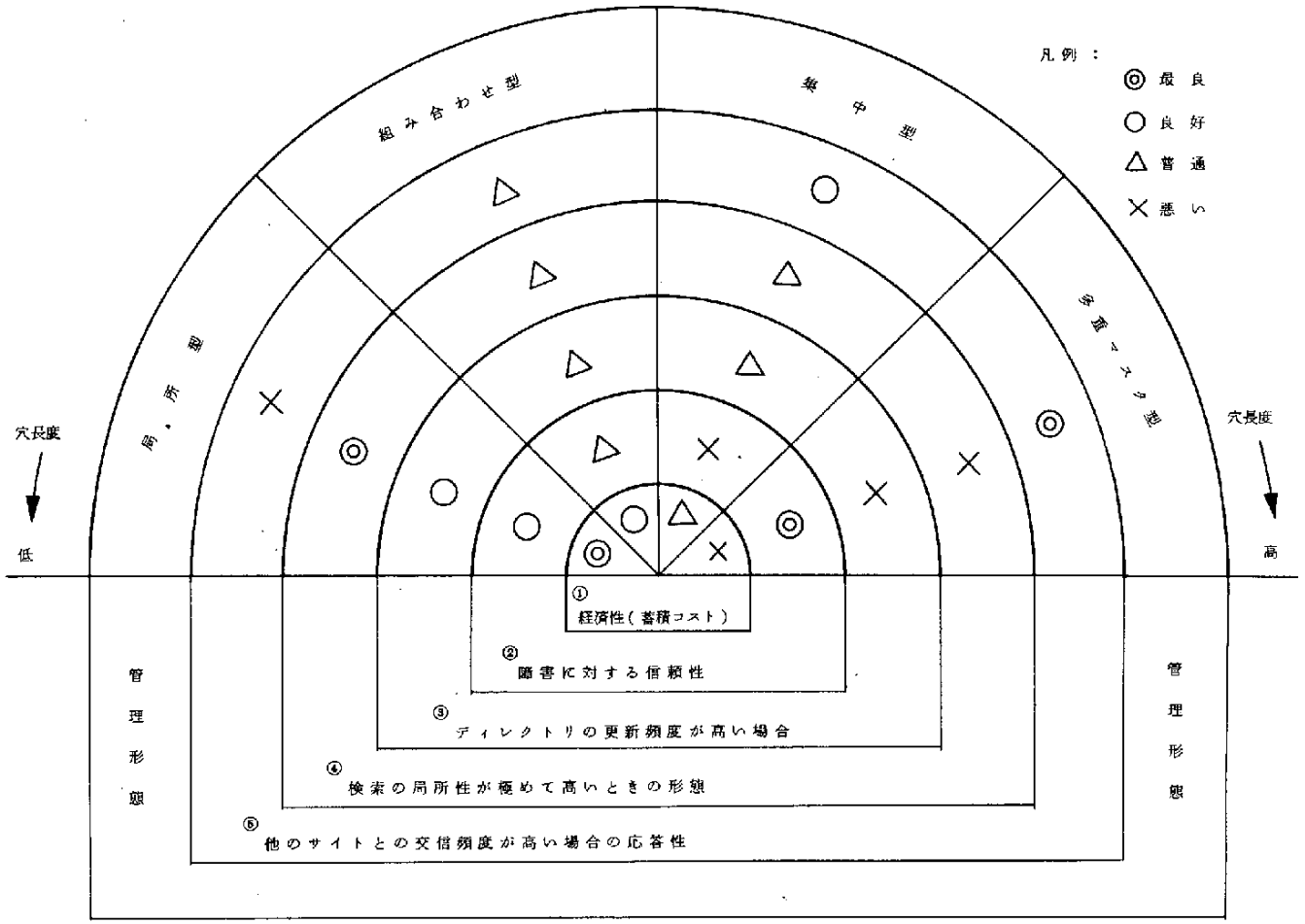


図４－１４　ディレクトリの管理形態と得失

4.4 ま と め

第3章までの議論で、コンピュータネットワーク上に散在するリソースの統合を目的とした場合、例えば分散型データベースシステムを構築するときには、分散と異種の両面について、ユーザへ不可視性を提供することが必要であると結論した。このような不可視性を実現するために必要な種々の情報を管理するためには、本章で議論したＮＣＣが必要である。案内センターの機能を有するＮＣＣは、ＤＢＭＳでいうＤＤ／Ｄ的位置付けが可能である。

本章ではこのような議論に基づいて、以下に述べる議論を行った。まず、4.1では情報管理でいうクリアリング機能に関する検討を行った。ここではクリアリング機能をクリアリングサービス（研究課題情報源案内業務）、レフェラルサービス（情報源案内業務）、一次情報の提供に三分しそれぞれ定義を行った。4.2では、クリアリングセンターの構想例を示した。一つは「統計データの収集・利用研究会」が検討した「統計データバンクにおけるクリアリングセンター」であり、他の一つは「行政情報システム研究所」が検討した「行政情報案内センター」である。この二つの例は、何れも構想に留まり、具体的には実現されていないが参考になる例である。4.3では、所在源情報管理の基本的技術としてＤＤ／Ｄ的アプローチが有効であることを述べた。ＤＤ／Ｄの定義を行い、ＤＢＭＳとの結合においては従来ＤＢＭＳ自身が持っていたメタデータとの一元管理が望ましい形態であると述べた。さらに、各氏が提案しているディクショナリの管理形態について利点／欠点を論じた。ディクショナリの管理形態は、冗長／非冗長、集中／分散の二面から論じられるが、ディクショナリが持つ情報の内容と構築するシステムの目的に充分あったディクショナリを形成する柔軟な姿勢が重要であると結論した。つまり、単純に集中か分散かという議論では解決しない。例えば、システムへのアクセス資格をチェックする機密保護情報は集中の形態を採用し、データベースのスキーマ／サブスキーマ情報はローカルに分散させる形態を採用することも充分考えられる。このようにディクショナリの管理形態は政策、効率、地理、組織等の種々の面からの検討が必要であろう。

5. 標準ジョブ制御言語システムの開発

5.1 はじめに

最近、コンピュータ・ネットワークが実用段階に達してきている。コンピュータ・ネットワークの研究段階では、コンピュータに精通したエキスパートしかネットワークを使用しなかったが、実用・普及段階に入ると一般のエンド・ユーザも使用するようになってくる。

数多くの異機種コンピュータから構成されたリソース・シェアリング・コンピュータ・ネットワークの利用においてエンド・ユーザがまず直面する問題は各機種用のジョブ制御言語の多様性である。

コンピュータ・ネットワークが存在する以前は、自分が利用している計算センターのコンピュータのジョブ制御言語のみを知っていれば充分であった。しかし、コンピュータ・ネットワークが実現され、ネットワーク内の複雑のコンピュータを使用するためにはそれらのすべてのジョブ制御言語に精通しなければならない。さらに、いくつかの異なったジョブ制御言語を同時に使用することは非常に繁雑である。こうしたことが、初めてネットワークを使用しようとするエンド・ユーザを遠ざけてしまう結果となり、少なからずネットワーク普及の障害となっている。

それ故、ユーザがネットワーク内のすべてのリソースに対して容易にアクセスできるように、すべてのコンピュータに共通のジョブ制御言語を必要とする。即ち、ユーザは一つの共通のジョブ制御言語のみを学ぶ事によってネットワーク・システム全体がアクセス可能となる。

このジョブ制御言語としては異機種間のリソース・シェアリング・コンピュータ・ネットワーク環境下に適した機種に依存せぬ標準ジョブ制御言語である事が必要となり、その言語仕様とシステムを作成した。

5.2 システムの概要

5.2.1 目　的

プログラミングを始めたばかりの初心者が、FORTRAN、COBOL、PL1などの文法を学んで、プログラムを作成し、これからいよいよコンピュータにかけてコンパイル・リンク・実行を行なおうとする場合、一番やっかいな問題となるのはジョブ制御用のカードデックを作成することであろう。一般には、ジョブ制御言語は各コンピュータ会社により、また同一の製造会社のものでも機種によって異なるのが当然のこととなっているし、機種によっては通常のコンパイル・リンク・実行を行なうのに際しても２０～３０枚のジョブ制御カードを必要とするのも多い。初心者にコンパイル言語を会得する労力に加えて、このような負担をかけるのは好ましくないように思われる。また、初心者でなくとも、特にその機種のシステムに関するプログラムを作成している場合などでなければジョブ制御言語は簡単な程、好ましいものであり、機種に依存しないものであることが望まれる。

特に、コンピュータネットワーク内で対象となるリソースごとに別のジョブ制御言語を用いなくてはならないのであれば、余りにも手続きが煩雑となり、コンピュータ・ネットワークの特性を著しく損なってしまう。

以上のような点を鑑みて、開発したのがこの標準ジョブ制御言語およびそのジェネレータシステムである。すなわち、ユーザーは自己のプログラムに対するジョブ制御カードを作成する場合にはとくに対象機種のジョブ制御言語を意識せずに、この共通の標準ジョブ制御言語を用いればよい。その標準ジョブ制御言語のステートメントを、標準ジョブ制御言語のジェネレータシステムが受け取って、対象となる機種用のジョブ制御言語に翻訳し、当該機種のシステムに送る形となる。つまり標準ジョブ制御言語のジェネレータシステムは汎用言語における言語プロセッサに対応する働らきをすればよいわけである。

この共用の標準ジョブ制御言語の言語仕様は、IBM370、UNIVAC1108、NEAC-ACOS6、FACOM OSⅣ/F4（Mシリーズ）、MELCOM BPM（COSMOシリーズ）のジョブ制御言語を検討した上でそのいずれの形にも容易に変換しうるように設計を行なった。この際、最も問題となったのは、ジョブステップ（アクティビティ）の区切り方が、必らずしも一様ではなく、機種によって異なる場合があるという点であった。機種が違い、ジョブ制御言語の言語仕様が異なる以上、プロセッサ等のシステムの名称は勿論、ファイルの書き方やその取扱いが多少異なるのは当然のことで、これらをジェネレータで翻訳するのはさして難しいことではない。しかし根本的なジョブ内の手順が異なるのは困った問題なのである。

例えば最も簡単な例について考える。今、FORTRANで書いたユーザプログラムの実行を行なうものとする。この場合には、コンパイル、リンク、実行の3つの段階に分かれるのが普通である。ただしUNIVAC1108ではリンクのジョブステップは実際には存在しているが、ジョブ制御言語上では陽に書き表わす必要はなく、NEAC ACOS6では、リンクと実行とのジョブステップの区切りは判然としていない。

次に同じようにFORTRANのソースプログラムを実行する場合でも、前以ってそのソースプログラムがファイルに入っており、そのファイルの中からいくつかのプログラムを選んでコンパイルする場合を考えると問題はかなり複雑になってくる。ACOS以外の場合は、前の場合とほぼ同じで入力媒体をファイルとし、そのファイルから現在コンパイルしようとしているプログラムを選び出すことが可能である。ところがACOSの場合は、これらと著しく異なり、ソースプログラムの入っているファイルを一旦入力用のファイルにコピーしてからコンパイルにとりかかる。このコピーの際にソースプログラムのファイルがシークェンシャルファイルであるため、その構成を熟知していないとコピーできない。すなわちA、B、Cの3つのソースプログラムが入っている場合にBのみコンパイルしたいとすると、Bの直前のプログラム――ここではA――がわかっていないとBのみコピーすることができず、従ってコンパイルもできないことになる。これは多数のソースプログラムを一度にコンパイルする場合には、不要なものを除いて指定できるので便利かもしれないが、標準ジョブ制御言語の仕様上、不必要なものをユーザに指定させるのは、どうかと思われる。さらにACOSでは、このようにファイルからプログラムを選び出してコンパイルする場合、直接FORTRANコンパイラを呼び出してコンパイルするのではなく、ソース・オブジェクト・ライブラリ・エディタというシステムの管理下で処理を行なうため、あらかじめFORTRAN コンパイル用

のジョブ制御カードをソースプログラムに付けて入力ファイルを作り出しておかなければならない。いわばジョブステップが入れ子になっているわけである。このようにACOSについては、他の機種との共通性を求めることが大変に難しく、そのため、標準ジョブ制御言語の仕様を決定する際にも、いろいろな問題が起ってしまった。これを解決するために標準ジョブ制御言語のジェネレータシステムの実現に当ってACOSに対しては、ファイルをコピーする別のルーチンを作成し、それを用いることによって他の機種との共通性をもたせるようにしている。そのため、本来、ACOSの特徴となっている性質を部分的には切り捨て、まわり道をしているような場合も起りうることを記しておく。ACOSシステムに習熟しているユーザならば、ACOS用ジョブ制御言語を標準ジョブ制御言語の一部として書くことも許しているのでその機能を用いればよいと考える。

標準ジョブ制御言語の概要については次の項で述べることにする。

5.2.2 言語の概要

前項でも述べたようにこの標準ジョブ制御言語は、対象となるコンピュータによらず、共通のジョブ制御言語として、初心者にも書きやすいことを目的として作られている。また、特に複雑な作業内容のジョブや各機種固有の性質を利用するようなジョブを記述することはできないが、一般的で使用頻度の高いようなジョブについては、できるだけ便利に使えるように考慮している。

この標準ジョブ制御言語は、便宜上3つのレベルに分けて構成されている。第1のレベルはコンパイラ言語で書いたプログラムをコンパイル、リンク、実行する場合に、最少限度のジョブ制御カードをつけるだけでジョブを定義できるようにしたものである。すなわち、いま、FORTRANによるソースプログラムの実行までを行ないたいのであれば、標準ジョブ制御言語レベル1では、次のように書けばよい。

```
$JOB                JOB1,ACT001,NAME1
$$FORTRAN
        ┌──────────────────┐
        │ FORTRANソースプログラム │
        └──────────────────┘
$$DATA
        ┌──────────────────┐
        │      デ ー タ      │
        └──────────────────┘
$END
```

このような形ならば、初心者もジョブ制御言語に煩らわされずに、気軽に自己のプログラムのコンパイルや実行が可能であろうし、プログラミングに慣れたユーザであっても、プログラムのデバ

ッグや実際の実行に当って、標準的な形のコンパイル・リンク・実行を行なう作業が楽になるものと思われる。

この反面、この第1のレベルでは入力はカードデックに限り、実行時の入出力も標準使用のカードリーダーおよびラインプリンタに制限されていて、ファイルの使用等は指定できないので、このような作業を行ないたい場合には、第2のレベルの標準ジョブ制御言語を用いればよいようになっている。第2のレベルは通常、使用頻度の高い作業内容を想定して設計したもので、コンパイル、リンク、実行、ソースプログラムの更新作業をより簡単にできるようになっている。標準ジョブ制御言語の第2レベルは大体次のようなステートメントから成っている。

A．ジョブの定義に関するステートメント

A－1	JOBステートメント …	ジョブの定義
A－2	END 〃 …	〃 の終り
A－3	PASSWORD 〃 …	カタログ、ファイルのパスワードの指定

B．コンパイルに関するステートメント

B－1	FORTRANステートメント…	FORTRANのコンパイル
B－2	COBOL 〃 …	COBOL の 〃
B－3	PL1 〃 …	PL1 の 〃
B－4	SYSIN 〃 …	コンパイラの入力ファイルの定義
B－5	OFILE 〃 …	オブジェクトプログラム・ファイルの定義

C．リンクに関するステートメント

C－1	LINKステートメント …	オブジェクト・プログラムの結合、編集
C－2	EFILE 〃 …	実行形式プログラムファイルの定義
C－3	USERLIB 〃 …	オブジェクト・プログラム・ライブラリの参照

D．ユーザープログラムの実行に関するステートメント

D－1	EXECステートメント …	ユーザプログラムの実行
D－2	FILE 〃 …	ユーザプログラムで使用する入出力ファイルの定義

E．プログラムの更新に関するステートメント

E－1	EDITSステートメント …	ソースプログラムをファイルへ移したり、ファイル上のプログラムの更新を行なう
E－2	EDITO 〃 …	オブジェクト・プログラム・ライブラリの更新を行なう
E－3	EDITE 〃 …	実行形式プログラム・ライブラリの更新を行なう
E－4	EDITD 〃 …	ファイルの抹消を行なう

標準ジョブ制御言語の第2のレベルを越えるような作業を行なおうとする場合には、（例えば、アセンブラを使用する時、各機種に独得のユーティリティを使用する時、そのほか、その機種の特長を生かして効率のよい作業を行ないたい時など）、第3レベルの取り扱いとして各機種に固有のジョブ制御言語を用いることもできる。第3レベルと区別してはいるが、$ENTER、$EXITの2つのジョブ制御ステートメントの間にその機種で指定された固有のジョブ制御言語を書くことで第2レベルの標準ジョブ制御言語とは並列に書くことができる。ふつうの第2レベルの標準ジョブ制御ステートメントは、ジェネレータシステムで特定の機種に合ったジョブ制御言語に翻訳を行なうのであるが、第3レベルのもの（即ち、$ENTERの次のカードから、$EXITの前のカードまで）は、ジェネレータシステムでは内容を調べずにそのまま当該機種のジョブ制御言語とみなして転送することになる。

5.2.3 システムの構造

標準ジョブ制御言語システムは、コンパイラに対する言語プロセッサのように、各機種（コンピュータ・ネットワークにおける各リソース）に共通なこの標準ジョブ制御言語で書かれた情報をその機種固有のジョブ制御言語に翻訳するシステムである。翻訳に際してはできるかぎり、標準ジョブ制御ステートメント1個から1個または数個の翻訳後のジョブ制御ステートメントを作り出すようにし、ソースの標準ジョブ制御ステートメント間の関係が多数のステートメントに渡る形、すなわちステートメントをずっと先読みしないと翻訳結果が得られないような翻訳の仕方は避けるように努力している。

このシステムは、実際にはコンピュータ・ネットワークシステムの一部として作られるもので、ユーザがターミナルからリモートバッチジョブを投入した時、そのジョブをホストコンピュータに転送する際に介在するシステムである。すなわちリモートバッチジョブの一部として与えられる標準ジョブ制御言語で書かれたジョブ制御ステートメントを、この標準ジョブ制御言語システムでホストコンピュータのジョブ制御言語に翻訳したのち、そのジョブ全体をホストコンピュータのリモートバッチ処理システムに送ることとなる。

次にこの標準ジョブ制御言語システムの大まかなフローチャートを示す。実際には、リモートバッチシステムの一部として他との関連が出てくるわけであるが、ここでは標準ジョブ制御システムだけをとり出して述べることにする。

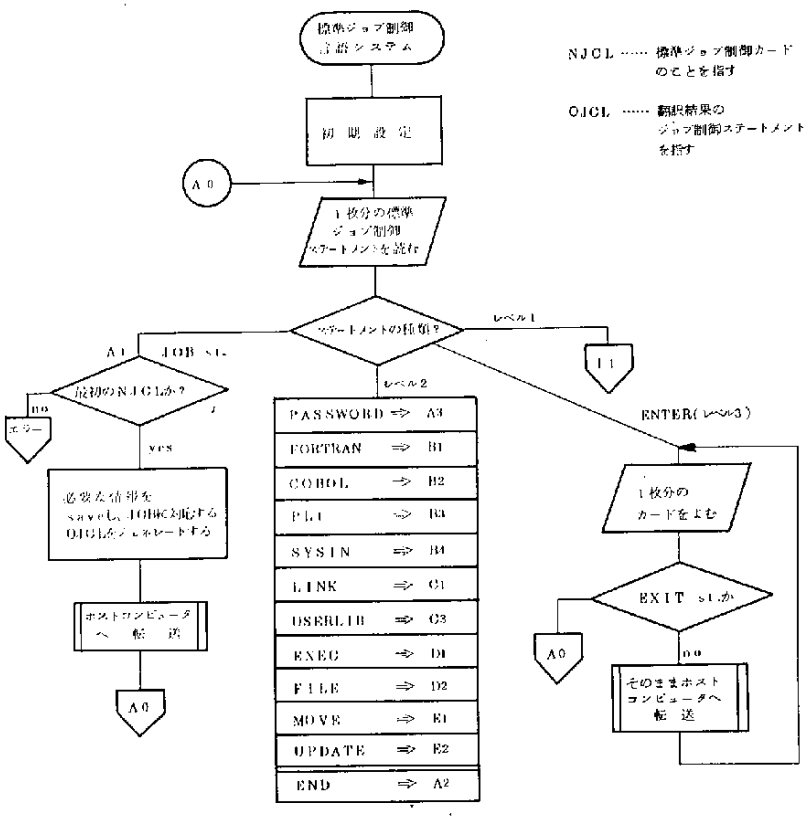

標準ジョブ制御
言語システム
NJCL …… 標準ジョブ制御カード
のことを指す
OJCL …… 翻訳結果の
ジョブ制御ステートメント
を指す
初期設定
A0
1枚分の標準
ジョブ制御
ステートメントを読む
ステートメントの種類？
レベル1
I1
A1
JOB st.
最初のNJCLか？
no
エラー
yes
必要な情報を
saveし、JOBに対応する
OJCLをジェネレートする
ホストコンピュータ
へ 転 送
A0
レベル2
PASSWORD ⇒ A3
FORTRAN ⇒ B1
COBOL ⇒ B2
PL1 ⇒ B3
SYSIN ⇒ B4
LINK ⇒ C1
OSERLIB ⇒ C3
EXEC ⇒ D1
FILE ⇒ D2
MOVE ⇒ E1
UPDATE ⇒ E2
END ⇒ A2
ENTER(レベル3)
1枚分の
カードをよむ
EXIT st.か
A0
no
そのままホスト
コンピュータへ
転 送

図 5 — 1

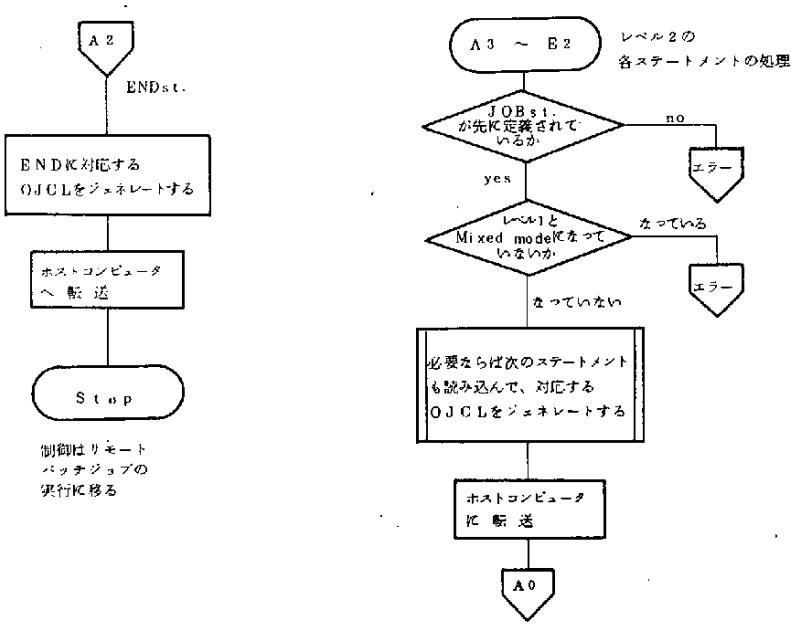

A2
ENDst.
ENDに対応する
OJCLをジェネレートする
ホストコンピュータ
へ 転 送
Stop
制御はリモート
バッチジョブの
実行に移る
A3 ～ B2
レベル2の
各ステートメントの処理
JOBst.
が先に定義されて
いるか
no
エラー
yes
レベル1と
Mixed modeになって
いないか
なっている
エラー
なっていない
必要ならば次のステートメント
も読み込んで、対応する
OJCLをジェネレートする
ホストコンピュータ
に 転 送
A0

図 5 — 2

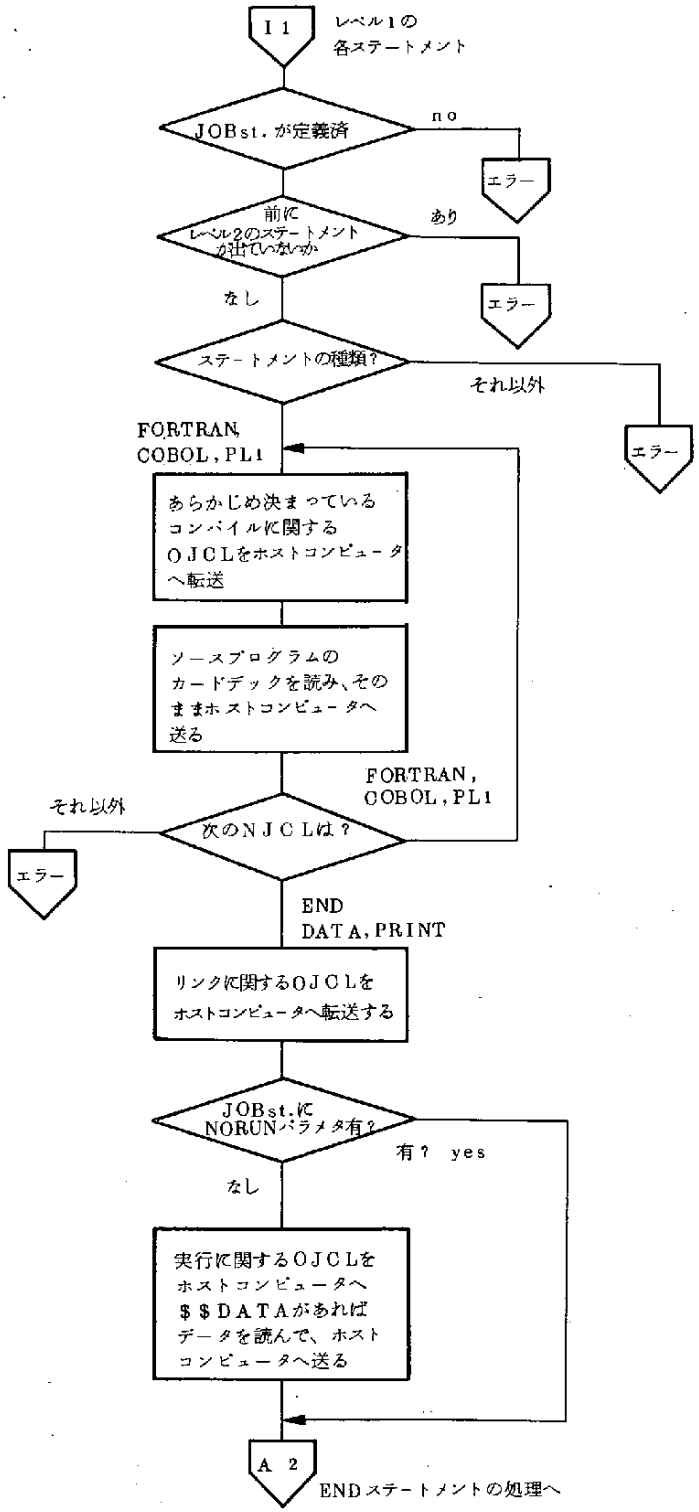

I 1
レベル1の
各ステートメント
JOBst. が定義済
no
エラー
前に
レベル2のステートメント
が出ていないか
あり
エラー
なし
ステートメントの種類?
それ以外
エラー
FORTRAN,
COBOL, PL1
あらかじめ決まっている
コンパイルに関する
OJCLをホストコンピュータ
へ転送
ソースプログラムの
カードデックを読み、そのま
まホストコンピュータへ
送る
FORTRAN,
COBOL, PL1
それ以外
次のNJCLは?
エラー
END
DATA, PRINT
リンクに関するOJCLを
ホストコンピュータへ転送する
JOBst.に
NORUNパラメタ有?
有? yes
なし
実行に関するOJCLを
ホストコンピュータへ
$$DATAがあれば
データを読んで、ホスト
コンピュータへ送る
A 2
ENDステートメントの処理へ

図 5 — 3

先ずシステムとしての各種の初期設定を行なったのち、第1枚目のジョブ制御カードを読み込む。ここで読み込んだステートメントは必らず、JOBステートメントでなければならない。JOB ステートメントのいろいろなパラメタを調べて、ホストコンピュータのジョブ制御言語（以下Object Job Control Language の略としてOJCLと記す）に翻訳し、このOJCL をホストコンピュータに転送する。この時残しておかなければならないパラメタの内容は退避しておく。

次に第2枚目以下のNJCLステートメント（標準ジョブ制御ステートメント）の処理に移る。前にも述べたようにこの標準ジョブ制御言語には3つのレベルがあり、第1レベルと第2、第3レベルは混ぜて書くことができない。したがって各ステートメントの処理のはじめでは先ずレベルが混ざって使用されていないかを調べる。（第2レベルと第3レベルは混ざっていてかまわない。）

各ステートメントの処理は個々のステートメントにより、いろいろ異なってはいるがその設計の方針としては大体同じような作り方をするのでレベル1のFORTRANステートメントについて説明することにする。レベル1のFORTRANステートメントであることがわかると、あらかじめホストコンピュータのジョブ制御言語の仕様に従って作ってあるテーブルからFORTRAN コンパイルの部分を取り出す。例えばACOS 6の場合には、

```
$    OPTION   FORTRAN
$    FORTRAN
```

であり、 FACOM OS Ⅳ／F 4ならば

```
//COMP1      EXEC     PGM=FORTGE
//SYSLIN     DD       DSN=&&OBJMOD, DISP=(NEW, PASS),
//                    UNIT=SYSDA, DCB=BLKSIZE=80,
//                    SPACE=(80, (200, 100), RLSE)
//SYSUT1     DD       DSN=UTDS1, DISP=(NEW, PASS),
//                    SPACE=(2048, (100, 50)), UNIT=DISK
//SYSUT2     DD       DSN=UTDS2, SPACE=(2048, (100, 50)),
//                    UNIT=DISK
//SYSPRINT   DD       SYSOUT=A
//SYSIN      DD       *
```

をとり出してホストコンピュータに転送する。このステートメントの場合、次にはFORTRAN ソースプログラムのカードデックが続いているはずなので、これはそのままホストコンピュータに送る。ここまでの間にFORTRANコンパイルの指示のあったことを情報として残しておく。COBOLやPL1の場合もテーブルの内容が異なるだけで手続きは同じである。

このあと次のNJCLを読んで調べる。FORTRAN、COBOL、PL1であれば、それぞれのステートメントの処理へ戻って、OJCLおよびソースプログラムの転送をくり返す。DATAもしくはENDステートメントであればJOBステートメントにNORUNパラメタがセットされているかを調べる。このパラメタはユーザがコンパイルのみ行ないたいとき指定するものなのでこの

指定があればここでジョブが打ち切られたものと考えてENDステートメントの処理へ行く。そうでなければ、リンクに関するＯＪＣＬをホストコンピュータに転送する。これもコンパイルの場合と同じくあらかじめテーブル内に作られているもので、前のジョブステップのコンパイラの種類を示す情報をもとにシステムライブラリの名前を選んでつけ加える。ただしこのジョブステップはホストコンピュータの機種によっては不要の場合もありＡＣＯＳでは必要ない。

ＦＡＣＯＭ ならば

```
//LINK1    EXEC  PGM=JQAL, PARM=(XREF, LIST),
//                COND=(4, LT, COMP1)
//SYSLIN   DD    DSN=&&OBJMOD, DISP=(OLD, DELETE)
//SYSUT1   DD    DSN=UTDS1, UNIT=DISK, DISP=(OLD, DELETE)
//SYSLMOD  DD    DSNAME=&&PROG(MAIN), DISP=(NEW, PASS),
//                UNIT=SYSDA, SPACE=(1024,(20,10,1),RLSE),
//                DCB=BLKSIZE=1024
//SYSPRINT DD    SYSOUT=A
```

（そして FORTRAN の場合は）

```
//SYSLIB   DD    DSNAME=SYS1.FORTLIB, DISP=SHR
```

以上を転送することになる。

このあと実行に関するＯＪＣＬをジェネレートする。

ＡＣＯＳの場合は

```
$    EXECUTE
```

を出しＪＯＢステートメントにパラメタの指定があれば

```
$    LIMITS    時間、メモリサイズ、出力行数
```

を転送する。

ＦＡＣＯＭの場合は、

```
//XQT1     EXEC  PGM=*.LINK1.SYSLMOD,
//               COND=((4,LT, COMP1), (4, LT, LINK1))
//FT06F001 DD    SYSOUT=A
```

を転送する。

さらに、ＤＡＴＡステートメントがあれば

（ＡＣＯＳの場合）

```
$    DATA      I*
```

（ＦＡＣＯＭの場合）

```
//FT05F001 DD    *
```

を転送し、このあとにデータ用カードデックを送る。

以上でレベル１のステートメントの処理を終え、ＥＮＤステートメントの処理として

（ACOSの場合）

```
$       ENDJOB
***EOF
```

（FACOMの場合）

```
//
```

を転送すれば、NJCLシステムによるジョブ制御言語の翻訳は終了し、コントロールをホストコンピュータによるジョブの実行に移せばよいこととなる。

レベル1のコンパイルに関するステートメントは実質的にはコンパイル・リンク・実行の3枚のカードをひとつにまとめたものなので転送するOJCLも3つの部分から成っているが、レベル2では、1つのNJCLステートメントを受け取るとテーブルから、対応する1組のOJCL の原形をとり出し、パラメタなどによって情報を補なって転送する形になる。

レベル3を示すENTERステートメントが出現した場合にはNJCLシステムではENTERの次のカードからEXITステートメントが現われるまでのすべてのカードを無条件でホストコンピュータに送り続ける。EXITステートメントを見つけた時点でこの操作をやめ、もとのレベル2の処理に戻る。

以上がこの標準ジョブ制御言語（NJCL）システムの概要である。

5.2.4 使用と変換の例

ここで、標準ジョブ制御言語を使用した例をいくつか挙げ、それを特定のシステムに固有のジョブ制御言語に変換した場合もあわせて示すことにする。

例 1

> FORTRANソースプログラムをカードデックの形でコンパイルし、リンクし、実行する。リンクの際にはユーザーズライブラリOBLIBを結合し、実行時にはカードリーダとラインプリンタを標準使用に従って用いるものとする。

この時標準ジョブ制御言語では次のように書けばよい。（レベル2）

```
$JOB     JOB01, EX1, JIPDC                    …①
($PASSWORD  ABCDEF, XYZ123)                   …②
$FORTRAN                                      …③
```

FORTRAN ソースプログラム

```
$LINK                                         …④
```

```
$USERLIB OBLIB                                        …⑤
$EXEC                                                 …⑥
```

データ

```
$END                                                  …⑦
```

これを各システム固有のジョブ制御言語に変換すると次のようになる。

（IBM OS/VS1の場合）

```
//JOB01       JOB    EX1, MSGLEVEL=(1, 1)                    …①に対応
//COMP1       EXEC   PGM=IEYFORT
//SYSPRINT    DD     SYSOUT=A
//SYSLIN      DD     DSN=&&OBJMOD, DISP=(NEW, PASS),
//                   UNIT=SYSDA, SPACE=(80, (200, 100), RLSE),
//                   DCB=BLKSIZE=80
//SYSIN       DD     *
```

③に対応

FORTRAN ソースプログラム

```
//LINK1       EXEC   PGM=IEWL, PARAM=(XREF, LIST),
//                   COND=(4, LT, COMP1)
//SYSLIB      DD     DSNAME=SYS1.FORTLIB, DISP=SHR
//SYSLMOD     DD     DSNAME=&&PROG(MAIN), DISP=(NEW, PASS),
//                   UNIT=SYSDA, SPACE=(1024, (20, 10, 1), RLSE),
//                   DCB=BLKSIZE=1024
//SYSPRINT    DD     SYSOUT=A
//SYSUT1      DD     DSN=UTIL1, UNIT=SYSDA,
//                   SPACE=(1024, (100, 10), RLSE),
//                   DCB=BLKSIZE=1024
//SYSLIN      DD     DSN=&&OBJMOD, DISP=(OLD, DELETE)
//           DD      DSN=OBLIB, DISP=(OLD, KEEP)             …⑤に対応
//XQT1        EXEC   PGM=*.LINK1.SYSLMOD,                    }⑥に〃
                   COND=((4, LT, COMP1), (4, LT, LINK1))
//FT05F001    DD     *                                       …データ用カードデックの存在により
```

④に対応

データ

```
//FT06F001    DD      SYSOUT=A                                    ⑥に対応
//                                                                ⑦  〃
```

（UNIVAC EXEC−8 の場合）

```
@RUN, /PT       JOB01, EX1, JIPDC                   …①に対応
@FOR, IS        , RPROG                             …③  〃
```

FORTRAN ソースプログラム

```
@MAP, IS        , APROG                             …④に対応
IN RPROG
LIB OBLIB                                           …⑤に対応
@XQT            APROG
```

データ	…⑥に対応

```
@FIN
```

（NEAC ACOS 6の場合）

```
$   SNUMB       JOB01                               ①に対応
$   IDENT       EX1, JIPDC
$   USERID      ABCDEF$XYZ123                       ②  〃
$   FILEDIT     SOURCE, OBJECT, INITIALIZE
$   FILE        R*, O1S, 3L                         ③に対応
$   DATA        *C, , COPY
$   FORTRAN                          (73カラムより)F01
```

FORTRAN ソースプログラム

```
$   ENDEDIT                                         ②
$   ENDCOPY                                         〃
$   FILEDIT     , OBJECT, UPDATE                    ④に対応
$   FILE        *R, O1R
$   FILE        R*, O2S, 4L
$   DATA        *C, , COPY
```

```
$   INCLUDE
$   OPTION      FORTRAN
$   LIBRARY     L0                          ←⑤に対応
$   COPY        , , F01
$   INCLUDE
$   EXECUTE
$   PRMFL       L0, R, S, ABCDEF/OBLIB      ←⑤に対応
$   ENDEDIT     END
$   ENDCOPY
$   EXECUTE                                 ←⑥に対応
$   FILE        R*, O2R
$   DATA        I*                          ←入力データの存在による
```

データ

```
$   ENDJOB
***EOF
```

（FACOM OS Ⅳ/F4 の場合）

```
//JOB01     JOB     EX1, MSGLEVEL=(1, 1)                    …①に対応
//COMP1     EXEC    PGM=FORTGE
//SYSLIN    DD      DSN=&&OBJMOD, DISP=(NEW, PASS),
//                  UNIT=SYSDA, SPACE=(80, (200, 100), RLSE),
//                  DCB=BLKSIZE=80
//SYSUT1    DD      DSN=UTDS1, DISP=(NEW, PASS),
//                  SPACE=(2048, (100, 50)), UNIT=DISK
//SYSUT2    DD      DSN=UTDS2, SPACE=(2048,(100,50)),
//                  UNIT=DISK
//SYSPRINT  DD      SYSOUT=A
//SYSIN     DD      *
```

③に対応

FORTRAN ソースプログラム

```
//LINK      EXEC    PGM=JQAL, PARAM=(XREF, LIST),
//                  COND=(4, LT, COMP1)
```

```
//SYSUT1     DD     DSN=UTDS1,UNIT=DISK,DISP=(OLD,DELETE)
//SYSLMOD    DD     DSNAME=&&PROG(MAIN),DISP=(NEW,PASS),
//                  UNIT=SYSDA,SPACE=(1024,(20,10,1),RLSE),
//                  DCB=BLKSIZE=1024
//SYSPRINT   DD     SYSOUT=A
//SYSLIB     DD     DSNAME=SYS1.FORTLIB,DISP=SHR
//SYSLIN     DD     DSN=&&OBJMOD,DISP=(OLD,DELETE)
//          DD     DSN=OBLIB,DISP=(OLD,KEEP)
//XQT1       EXEC   PGM=*.LINK.SYSLMOD,
//                  COND=((4,LT,COMP1),(4,LT,LINK1))
//FT06F001   DD     SYSOUT=A
//FT05F001   DD      *
```

④に対応

…⑤に対応

⑥に対応

データ

```
//
```

（MELCOM BPMの場合）

```
!JOB        EX1,JIPDC
!FORTRAN    LS,GO
```

…①に対応

…③ 〃

FORTRAN ソースプログラム

```
!LOAD       (GO),(UNSAT,(F4LIB)),(EF,(OBLIB)),(LMN,XQT1)
!RUN        (LMN,XQT1)
!DATA
```

…④および⑤に対応

…⑥に対応

…データの存在による

データ

```
!FIN
```

…⑦に対応

例　2

COBOL ソースプログラムをカードデックから大記憶のファイルへコピーし、改めてこのファイルをコンパイル、リンク、実行する。この時、オブジェクトプログラムおよび実行形式プログラムを保存する。

これを標準ジョブ制御言語では次のように書く。

```
$JOB          JOB02, EX2, JIPDC
($PASSWORD    ABCDEF, XYZ123)
$EDITS
$NFILE        SFILE1, NEW, DA
/$INSERT      CB01, C
```

COBOL ソースプログラム

```
$COBOL        , SAVE
$OFILE        OBJ1, NEW, DA
$SYSIN        SFILE1(CB01)
$LINK         EPROG, SAVE
$EFILE        EXEC1, NEW, DA
$EXEC         EXEC1(EPROG)
$FILE         PT, , PRINT
$FILE         CR, , CREADER
```

データ

```
$END
```

これを各システム固有のジョブ制御言語に変換すると次のようになる。

（FACOM OSⅣ/F4の場合）

```
①  //JOB02     JOB   (EX2, JIPDC), MSGLEVEL=(1, 1)
   //EDIT1     EXEC  PGM=JSEUPDATE, PARAM=NEW
   //SYSPRINT  DD    SYSOUT=A
   //SYSUT2    DD    DSN=SFILE1, DISP=(NEW, CATLG, DELETE),
   //                SPACE=(1024, (500, 200, 1)), UNIT=DISK,
```

```
//                  DCB=BLKSIZE=1024
//SYSIN     DD    *
./      ADD         NAME=CB01
```

COBOL ソースプログラム

```
/*
//COMP1     EXEC PGM=JMNC0000,REGION=128K
//SYSUT1    DD   DSN=&&UTIL1,UNIT=SYSDA,SPACE=(460,(700,100))
//SYSUT2    DD   DSN=&&UTIL2,UNIT=SYSDA,SPACE=(460,(700,100))
//SYSUT3    DD   DSN=&&UTIL3,UNIT=SYSDA,SPACE=(460,(700,100))
//SYSUT4    DD   DSN=&&UTIL4,UNIT=SYSDA,SPACE=(460,(700,100))
//SYSPRINT  DD   SYSOUT=A
//SYSLIN    DD   DSN=OBJ1(CB01),DISP=(NEW,CATLG,DELETE),
//                UNIT=SYSDA,SPACE=(80,(500,100)),
//                DCB=BLKSIZE=80
//SYSIN     DD   DSN=SFILE1(CB01),DISP=OLD
//LINK1     EXEC PGM=JQAL,PARM=(LIST,XREF),REGION=128K
//                COND=(5,LT,COMP1)
//SYSLIN    DD   DSN=OBJ1(CB01),DISP=OLD
//SYSLMOD   DD   DSN=EXEC1(EPROG),DISP=(NEW,CATLG,DELETE),
//                UNIT=SYSDA,SPACE=(1024,(50,20,1)),
//                DCB=BLKSIZE=1024
//SYSUT1    DD   UNIT=SYSDA,SEP=(SISLIN,SYSLMOD),
//                SPACE=(1024,(50,20))
//SYSPRINT  DD   SYSOUT=A
//SYSLIB    DD   DSN=SYS1.COBLIB,DISP=SHR
//XQT1      EXEC PGM=EXEC1(EPROG),DISP=OLD
//CR        DD   *
```

データ

```
//PT        DD   SYSOUT=A
//
```

ここで①はソースプログラムをファイルへコピーするジョブステップ，②はコンパイル，③はリ

ンク，④は実行である。

（ACOS－6の場合）

```
$       SNUMB     JOB02
$       IDENT     EX2, JIPDC
$       USERID    ABCDEF$XYZ123
$       FILSYS                                    ┐
USERID            ABCDEF$XYZ123                   │ SFILE1の
FCREAT            ABCDEF/SFILE1, LINKS/3, 10/     │ ファイルの創成
                                                  │ および登録
$       FILEDIT   SOURCE, NOBJECT, INITIALIZE     ┐
$       PRMFL     K*, W, S, ABCDEF/SFILE1         │ ソースプログラム
$       DATA      *C, , COPY                      │ をSFILE1に
$       COMPILE   OFF                             │ コピーする。
$       INCLUDE   SOURCE
                          （73カラムより）
$       COBOL     EALERT                  CB01
```

COBOL ソースプログラム

（73カラムより）

```
$       GMAP                              N.J.CL  ┐
        SYMDEF    N.J.C.L                         │ ダミーの
N.J.CL  NULL                                      │ プログラム
        END                                       ┘
$       COMPILE   ON
$       ENDEDIT
$       ENDCOPY
$       FILSYS                                    ┐ オブジェクトライブラリ
USERID            ABCDEF$XYZ123                   │ ファイルOBJ1の
FCREAT            ABCDEF/OBJ1, LINKS/3, 10/       │ 創成および登録
$       PROGRAM   NJCL.1                          ┐ CB01を
$       PRMFL     **, R, R, DRP/NJCLEDIT01        │ ワークファイルへ
$       PRMFL     IN, R, S, ABCDEF/SFILE1         │ 抜き出す
$       FILE      OT, S1S, 3L, DKU411             │
$       DATA      I*                              ┘
          CB01
```

```
$       FILEDIT   SOURCE, OBJECT, INITIALIZE
$       PRMFL     R*, W, S, ABCDEF/OBJ1
$       FILE      *C, S1R
$       FILSYS
USERID            ABCDEF$XYZ123
FCREAT            ABCDEF/EXEC1, LINKS/4, 12/,
                  MODE/RAND
$       FILEDIT   , OBJECT, UPDATE
$       PRMFL     *R, R, S, ABCDEF/OBJ1
$       FILE      R*, O2S, 4L
$       DATA      *C, , COPY
$       INCLUDE
$       SYSLD     CATALOG=EPROG
$       LOWLOAD
$       OPTION    COBOL
$       MODIFY    , , N.J.CL
$       INCLUDE
$       EXECUTE
$       ENDLD
$       ENDEDIT   END
$       ENDCOPY
$       SYSEDIT
$       PRMFL     Q*, W, R, ABCDEF/EXEC1
$       FILE      R*, O2R
$       PROGRAM   EPROG
$       PRMFL     **, R, R, ABCDEF/EXEC1
$       PRINT     PT
$       DATA      CR
```

ワークファイル内のソースプログラムをコンパイルする。結果はＯＢＪ１に入れる。

実行形式プログラムファイルＥＸＥＣ１の創成および登録

ＬＩＮＫに相当する。

実行形式プログラムライブラリの作成

実　行

デ　ー　タ

```
$       ENDJOB
***EOF
```

5.3. 言 語 仕 様

5.3.1 言 語 仕 様

次に標準ジョブ制御言語の言語仕様について述べることにする。前節でもたびたび述べたように、この標準ジョブ制御言語は3つのレベルから成り、第1レベルは第2レベルをカタログした形になっていて第2、第3レベルとは言語仕様上は別体系となっている。このため先ずレベル1について述べ、その次にレベル2の各ステートメントについて記すことにした。

レ ベ ル 1

標準ジョブ制御言語のレベル1はプログラミングを始めたばかりの初心者を対象にしている。また，初心者でなくてもコンパイル言語を用いて作成したソースプログラムのカードデックにデータ用カードデックをつけて（つけないことも可）コンパイル・リンク・実行を行なうことは、日常最もよく行なわれている作業であると思われる。標準ジョブ制御言語のレベル1では、1個のジョブでコンパイル言語を用いて作成したソース・カードデックを入力し、これをコンパイルする場合、およびコンパイル・リンク・実行を行なう場合のみを想定している。

レベル1においてはステートメントの種類も少なく、使用方法も簡単なので例を示すことによって個々のステートメントの説明に換えたいと思う。

例 1

FORTRANプログラムをコンパイル実行する場合は次のようになる。

```
$JOB                    オペランド
$$FORTRAN
┌──────────────────────────┐
│   FORTRAN ソース・プログラム   │
└──────────────────────────┘
$$DATA                                  ┐
┌──────────────────────────┐            ├ （もしあれば）
│   入 力 デ ー タ          │            ┘
└──────────────────────────┘
$END
```

$JOBは、ジョブ開始を示すステートメントで、各レベルに共通であるので、詳細についてはレベル2で説明するものとし、ここでは$JOBステートメントのオペランドにはユーザ名やアカウント番号を記述することになっていることを記すに留める。ただしソースステートメントのコンパイルのみを行ないたい場合にはレベル1で用いている場合に限り、キーパラメタとしてNORUNと書くことができる。通常レベル1でジョブが与えられると、標準ジョブ制御言語の

ジェネレータはこれをコンパイル・リンク及び実行の３つ以上のジョブステップから成る一連のジョブと解釈し、それぞれの機種固有のジョブ制御言語に翻訳するわけであるが、ＪＯＢステートメントのオペランドにNORUNを指定した場合に限り、コンパイルのジョブステップのみでそれ以降の処理手続きをジェネレートしない。

＄＄FORTRANはレベル１でFORTRANコンパイラを指定していることを示すもので、この他に＄＄PL1、＄＄COBOLの使用が可能である。

標準ジョブ制御システムでは、一般にステートメントは第１カラムに＄を置き、第２カラムより各ステートメントの名標となるキーワードを置くことになっているが、レベル１においてのみ使用可能なこの＄＄FORTRAN等のステートメントは第１、第２カラムに＄を続けることによってレベル１であることを示している。＄の文字は特殊記号の文字セットが機種により異なることがあり、使用不可の場合も考えられるのでそのような場合には別の特殊文字で置き換えてもかまわない。

＄＄DATAステートメントは次にあるカードデックが入力データであることを示すものであり、＄ENDは、そのジョブの終了を示すステートメントである。

レベル１では、前にも述べた通り、特に指定がなければコンパイルののちリンク・実行まで行なう。ただしコンパイル時にエラーが生じている場合には、リンクや実行は行なわない。

また、レベル１でコンパイル・リンク・実行を行なった場合は、いかなる場合にもソースプログラムやオブジェクトプログラムまた実行形式プログラムをファイルとして保存することは不可能である。こういう場合にはレベル２を使用されたい。さらに、リンクの際にユーザライブラリを使用したい場合にもレベル２を使用しなければならない。

例　２

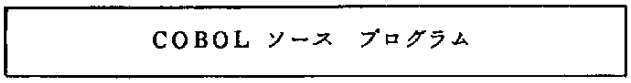
FORTRANプログラムのコンパイルのみ行なう場合は次のようになる。

```
＄JOB                    オペランド、NORUN
＄＄FORTRAN
        FORTRAN ソース プログラム
＄END
```

例　３

COBOLプログラムをコンパイル、実行する場合は、次のようになる。

```
＄JOB                    オペランド
＄＄COBOL                CR, PT
```



$$DATA

……①

$END

この場合もほとんどFORTRANの場合と同じであるが、COBOLに限り、カード・リーダおよびプリンタ出力用のファイル定義名はユーザがソースプログラム中でAssign文で指定したものと一致しなくてはならないので、COBOLステートメントのオペランドに指示すること。

$$COBOL　<カードリーダのファイル定義名>、<プリンタのファイル定義名>

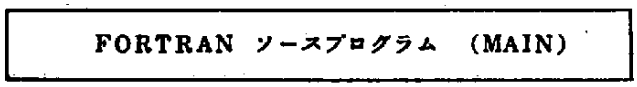
但し、この場合にも対象機種によって、カードリーダあるいは両方共標準名を使っていれば、定義する必要のないものもある。この場合は、これらを位置パラメタと考えた上で省略可能である。

例 4

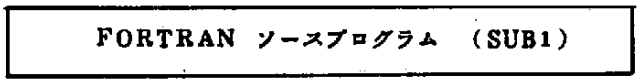
例1と同じくFORTRANによるソースプログラムをコンパイルし、実行する。ただしソースプログラムはMAIN、SUB1、SUB2の3つのサブプログラムから成る。

$JOB

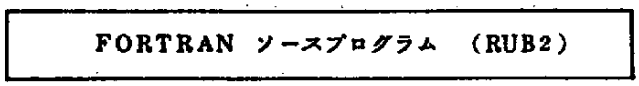
$$FORTRAN



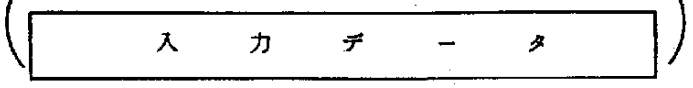
$$FORTRAN



$$FORTRAN



$$DATA



もしあれば

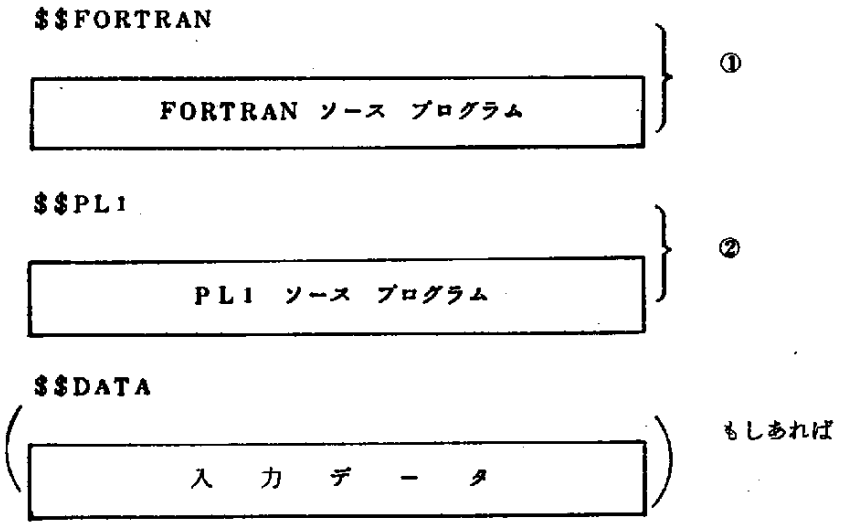
$$END

このようにソースプログラムがいくつかのサブプログラムに分かれている時には、その各々に$$FORTRAN（または、$$COBOL、$$PL１）ステートメントを入れる必要がある。

もしも入れ忘れてひとまとめにした場合にも機種によっては実行までできる場合もありうると思われるが、一般的には第１のサブプログラムのみコンパイルし、そのあとエラーとなる可能性が高いので、必らず各サブプログラムごとに制御カードを入れることが望ましい。

例　5

FORTRANプログラムとPL１プログラムを同時にコンパイルし、ひとまとめにリンクして、実行する場合は次のようになる。



①と②の順は逆でもかまわない。

この他に実際には余りないことと思われるが、COBOLとPL１、FORTRANとCOBOLのソースプログラム、あるいはFORTRAN、COBOL、PL１の３種のソースプログラムを同時にコンパイルし、リンクし、実行することも上記と同様な方法によって、理論上は可能である。

表５－１　レベル１　ジョブ制御ステートメント一覧表

ステートメント名	使　用　目　的	
$JOB	ジョブの定義	レベル２と共通
$$FORTRAN	FORTRANのコンパイルを指定する	レベル１専用
$$COBOL	COBOL　〃	〃
$$PL1	PL1　〃	〃
$$DATA	入力データ用カードデックを示す	〃
$END	ジョブの終りを示す	レベル２と共通

レベル　2

標準ジョブ制御言語の第２レベルは、コンパイル言語を用いているユーザが日常よく行なうと思われる作業内容を想定して設計したもので、コンパイル、リンク、実行およびソースプログラムの更新等をより簡単に行なえるようになっている。ここでは先ず記述上での一般的規則について述べ、そのあと各ステートメントの説明を行なうものとする。

標準ジョブ制御言語の一般的規則

この標準ジョブ制御言語の各ステートメントの一般形は次のようなものとする。

$<ステートメント名標><空白>[<パラメタ列>]

(1) この標準ジョブ制御言語は、８０欄カード上で表現される形を想定し、一般的には、カードリーダーから読み込まれるか、あるいは別の媒体から読み込まれたとしてもそれに準じるフォーマットであると考える。

(2) 先頭の$はこの１枚分のカードが標準ジョブ制御言語のステートメントであることを示すもので第１カラムに記す。特殊文字$はもし機種によって$の使えない場合には、他の特殊文字（例えば＆！／など）で置き換えてもさしつかえない。

(3) ステートメント名標とは、標準ジョブ制御言語のステートメントの種類を規定するもので、例えばJOB、FORTRAN、EXECなどのようにジョブやジョブステップに関する情報を示すものや、SYSIN、USERLIB、DATAなどのようにファイルについての記述を行なうものなどがある。

(4) ステートメント名標のあとには１個以上の空白が必要である。

(5) パラメタ列はパラメタを１個以上書き並べたもので２つ以上の場合にはカンマで区切る。ステートメントによって必要なパラメタの種類、性質、個数が決まっており、全くパラメタを必要としないステートメントもある。

(6) パラメタには、記述方法によって位置パラメタとキーパラメタの２種類がある。位置パラ

メタはパラメタ列において記入する位置が決まっているもので、省略時にも当該パラメタに対応するカンマを記入して省略していることを示す必要がある。これに対して、キーパラメタは〔キーワード＝パラメタ〕の形で記入するか、あるいはパラメタそのものがキーワードの場合で、位置パラメタがあればそのあとに、順序に関係なく記入するパラメタのことである。

(7) 一連のパラメタのストリングの中には空白を含んではならないが、パラメタの区切り記号であるカンマや等号あるいは括弧の前後には任意個の空白があっても構わない。

(8) 標準ジョブ制御ステートメントの内容はカードの第1カラムの＄で始まり、第72カラムまでに収めるものとする。パラメタ列が長すぎて1枚分のカードに入り切らない場合はパラメタ列をカンマで打ち切り、次のカードの第1カラムに＄を記したのち、第2カラムは空白とし第3カラム以降の任意のカラムから残りのパラメタを記入する。原則としては、1ステートメントが何枚のカードにわたってもよい。

(9) 第1～第72カラムにはコメントを書くことは許さない。ただし73～80カラムにシークェンス番号またはコメントを含むことはさしつかえないがシークェンス・チェックは行なわない。

A．ジョブの定義に関するステートメント

ジョブの定義に関するステートメントとしては、JOBおよびENDステートメントがある。また、ジョブの定義ではないがジョブ全体にわたってパスワードを定義するPASSWORDステートメントもこの範ちゅうに加えることにする。

次にこれらのステートメントについて詳しく述べることとする。

A－1　JOBステートメント

JOBステートメントは、ジョブの始めを示すと共に、このジョブの実行に必要な各種の情報をシステムに与えるステートメントであり、一般形は、次のとおりである。

＄JOB＜空白＞＜ジョブ名＞，＜アカウント番号＞，＜識別名＞
〔，PRTY＝α〕〔，TIME＝{ (m，n) / m }〕
〔，PAGE＝p〕〔，＜コア占有領域サイズ＞KB〕
〔，NORUN〕

(1) ジョブ名は5文字以内の英字で始まる英数字のストリングでそのジョブの名前を定義するものである。

(2) アカウント番号は会計情報を与えるための、また、識別名はプログラマ名あるいはプロジェクト名を示すコードでこれは各計算センターで定められたものに準拠するものとする。

(3) 以上の3つのパラメタ（ジョブ名、アカウント番号、識別名）は位置パラメタであり、この順番に必らず記述しなければならず、省略することは許されない。一方次に述べるパラメタはキーパラメタであり、相互の位置関係は任意であってよい。また、省略することも可能で省略時には各機種あるいはこの標準ジョブ制御言語システムの定める値がとられ

る。

(4) 優先順位　PRTY＝α

これは、このジョブの優先順位（もしくは緊急度）を示すものでαにはアルファベットの1文字を記入する。このときアルファベットのZに近い方から優先順位が上位となる。

(5) CPU使用時間　TIME＝{ (m, n) / m }

これは、このジョブに割り当てられるべき最大のCPU使用時間を指定する。分と秒を指定する場合にはTIME＝(m, n)でmは分、nは秒を示す形、分のみ指定する場合にはTIME＝mの形を用いる。機種によっては秒単位の指定ができないものがあるが、そのようなときには、標準ジョブ制御言語システムにおいて秒の単位の切り上げを行なってから、当該機種用の制御言語に翻訳する。

(6) 最大出力ページ数　PAGE＝p

これは、このジョブ全体にわたって最大の出力ページ数を指定するパラメタであり、pはページ数を示す任意の整数である。機種によっては最大出力行数を指定しなければならないものもあるが、この場合には、1ページ＝50行として換算した上で当該機種の制御言語に翻訳を行なうので実際の出力結果においては多少の誤差を生じることもありうる。

(7) ＜コア占有領域サイズ＞KB

この指定に限り、ユーザプログラムの実行時のジョブステップに対するものであり、通常は指定する必要はないが、ユーザが特に大きなプログラムを実行しようとする時、その必要な主記憶容量の最大値を10進数でキロバイト単位で表わしたものにKBを付けて記すことができる。（1ワード＝4バイトとして計算されたい）

(8) NORUN 指定

JOBステートメントはレベル1でも同様の書き方をするのであるが、NORUN 指定のみはレベル1においてのみ有効なパラメタで、レベル2の制御ステートメントで用いている場合にこのパラメタを付けても無視される。

これは、レベル1においてコンパイルのみを行ない実行をしないことを意味し、ソースプログラムをコンパイルし、リストを得たい時に無駄を省く効果がある。

A－2　END ステートメント

ENDステートメントは文字通り、そのジョブの終りを示すものであってパラメタはなく

$END

と記述し、JOBステートメントで始まる一連のジョブデックの最後に位置するステートメントである。

A－3　PASSWORDステートメント

PASSWORDステートメントはそのジョブ内で使用するカタログ・ファイルを使用するために必要なステートメントであり、一般形は次のとおりである。

$PASSWORD<空白><システム・マスタ・ディレクトリ名>,<パスワード>

このステートメントは、機種による依存性の強いもので、現在迄に調べたところではACOS－6システムにのみ有効なものである。従ってもしこのステートメントを他の機種に対して書いても今のところ全く無意味で、この標準ジョブ制御言語の目的である一般性を欠いてしまって好ましくないのであるが、ACOS－6においてユーザがカタログ・ファイルを使用する際、この指定がないとファイルが使用できず、したがって実質的な作業が何もできなくなるのでやむを得ず加えたものである。

このステートメントは、JOBステートメントの次に、あらゆるジョブ制御ステートメントに先立って記述しなければならない。（ACOS－6を対象としカタログファイルを使用する場合に限り、指定すればよい）

B．コンパイルに関するステートメント

コンパイルを行なうため各種のコンパイラを指定するステートメントにはFORTRAN、COBOL、PL1の各ステートメントがある。

一般にコンパイラへのソースプログラムの入力方法はカードによる場合とファイルからの場合とがあるわけであるが、この標準ジョブ制御言語においては、カードデックの場合にはそのままコンパイラを指定するジョブ制御ステートメントの次にそのカードデックを付ければよいが、ファイルからの場合には、カードデックの代わりにSYSINステートメントをつけてそのファイルに関する記述を行なわなくてはならない。

また、コンパイル結果のオブジェクトプログラムを残しておきたい場合には、コンパイラ指定ステートメントにSAVEというパラメタを付け加えたのち、同時にこれらを入れるファイルを指定しなければならない。このために用いるステートメントには、OFILEステートメントがある。

次にこれらの各ステートメントについて詳しく述べることにする。

B－1　FORTRANステートメント

このステートメントは、FORTRANソースプログラムのコンパイルを行なうジョブステップを定義するステートメントであり、一般形は次のとおりである。

$FORTRAN<空白>[<オブジェクトプログラム名>][,<コンパイルパラメタ・リスト>]

<コンパイルパラメタ>::=<コンパイルパラメタ>|<コンパイルパラメタ><コンパイルパラメタ・リスト>

<コンパイルパラメタ>::=SAVE[({I/A})]|NOLIST|<プログラムサイズ>

<プログラムサイズ>::=S|M|L

(1) オブジェクトプログラム名は、6文字以内の英字で始まる英数字のストリングである。ソースがカードデックの場合で、SAVEパラメータを指定する場合には、必らずこの名前を書かなければならない。ソースがファイルからの入力の場合には、オブジェクトプログラム名はソースプログラム名と等しい名前がつけられるので、この指定は不要である。ただし、このパラメタは位置パラメタなので、プログラム名を省略して、他のパラメタをそのあとに書く場合には、プログラム名に対応するカンマが必要である。(例えば、$FORTRAN ,NOLISTのようにする。)さらに、ソースがカードデックである場合であっても、SAVEパラメタを指定しない場合には、プログラム名は書いても書かなくてもよい。

(2) プログラムサイズは、ソースプログラムの大きさをカード枚数の概算で示すもので、

1000枚までならば	S
1000～ 3000枚ならば	M
3000～10000枚 〃	L

の1文字を記入する。省略時にはSとみなす。

(3) SAVEパラメタはオブジェクトプログラムを、ユーザの指定するファイル内に保存しておくことを指示するものである。このパラメタを指定した場合には、そのファイルに関する記述を与えるOFILEステートメントをこのFORTRANステートメントの直後に書かなければならない。SAVEパラメタを指定してもOFILEステートメントがなければ、SAVEパラメタは無効となり、したがってプログラムの保存は行なわれない。

SAVEのあとにサブパラメタとして括弧で囲んでAかIを付けることができる。Iはイニシャル・モード、Aはアディショナル・モードを指し、OFILEで指定するファイルがOLDファイルであるとき、Iならば、そこに今まで入っていた内容をこわして新しいオブジェクトライブラリに作り変えることを意味する。Aならば、そこに入っていた内容を破壊することなく、その続きにオブジェクトプログラムを入れることを示している。OFILEがNEWファイルならば常にIとなり、(I)は省略してかまわない。OFILEがOLDのときに、このサブパラメタを省略するとAとみなす。

OFILEの指定 / SAVEのサブパラメタ	NEW	OLD
(I)	新しいファイルを登録し、新しいオブジェクトファイルを作る	指定のファイルの内容は全く新しいものとする
(A)	ありえない。エラーメッセージを出し(I)とみなす	指定のファイルの内容に続けてオブジェクトプログラムを入れる
省略時の解釈	(I)	(A)

ただし（Ａ）を指定したとき、ＯＦＩＬＥステートメントで指定するファイル内に現在作っているのと同名のプログラムがある時、機種によっては新しいものと置き換らないで同名のものが二つ入ってしまうことがありうるので、このような場合にはＥＤＩＴＯステートメントで（後述）前以って同名のプログラムを消すか、一旦別のファイルに作ってＥＤＩＴＯステートメントファイルをマージするかのいずれかの方法をとる方が無難である。

(4) ＮＯＬＩＳＴパラメタは、コンパイルリストが不要なときに指定するパラメタである。

(5) プログラム・サイズ、ＳＡＶＥおよびＮＯＬＩＳＴパラメタは、キーパラメタなので、お互いの位置関係は任意である。またパラメタを全て指定しなくてもかまわない。

Ｂ－２　ＣＯＢＯＬステートメント

このステートメントはＣＯＢＯＬソースプログラムのコンパイルを行なうジョブステップを定義するステートメントであり、一般形は次のとおりである。

$$\$\text{COBOL}\langle\text{空白}\rangle\left[\left\langle\begin{matrix}\text{オブジェクト}\\ \text{プログラム名}\end{matrix}\right\rangle\right][,\langle\text{コンパイルパラメタリスト}\rangle]$$

パラメタについてはＦＯＲＴＲＡＮに同じなので前項を参照されたい。

Ｂ－３　ＰＬ１ステートメント

このステートメントはＰＬ１ソースプログラムのコンパイルを行なうジョブステップを定義するステートメントであり、一般形は次のとおりである。

$$\$\text{PL1}\langle\text{空白}\rangle\left[\left\langle\begin{matrix}\text{オブジェクト}\\ \text{プログラム名}\end{matrix}\right\rangle\right]\left[,\left\langle\begin{matrix}\text{コンパイル}\\ \text{パラメタリスト}\end{matrix}\right\rangle\right]$$

パラメタについてはＦＯＲＴＲＡＮ、ＣＯＢＯＬに全く等しいのでその項を参照されたい。

Ｂ－４　ＳＹＳＩＮステートメント

このステートメントは、ＦＯＲＴＲＡＮ、ＣＯＢＯＬ、あるいはＰＬ１のソースプログラムをコンパイルするにあたって、そのソースプログラムがカードデックとして与えられるのではなく、既に作成されているファイルの中にあることを示すものであり、その一般形は次のとおりである。

$$\$\text{SYSIN}\langle\text{空白}\rangle\langle\text{ファイル名}\rangle(\langle\text{プログラム名列}\rangle)\left[,\left\{\begin{matrix}\text{DA}\\ \text{MT}\end{matrix},\langle\text{ボリューム通番}\rangle\right\}\right]$$

$$\langle\text{プログラム名列}\rangle::=\langle\text{プログラム名}\rangle\mid\langle\text{プログラム名}\rangle,\langle\text{プログラム名列}\rangle$$

(1) ファイル名はソースプログラムの存在するファイル名のことであって、８文字以内の英字で始まる英数字のストリングである。

(2) プログラム名列はこのジョブステップでコンパイルしようとするソースプログラム名、あるいは同一ファイル内にコンパイルしようとするプログラムが２個以上ある時にはそれ

らの名前をカンマで区切ったものを、括弧でくくったものである。プログラム名は6文字以内の英字で始まる英数字のストリングである。

(3) プログラム名を2個以上併記する場合には、そのプログラム名の順序はそれらのプログラムがファイル内で入っている順序どおり（シークェンシャル）になっていなければならない。

(4) 第3のパラメタはそのファイルの媒体を示すものでDAは大記憶、MTは磁気テープをあらわす。MTの場合にはそのボリューム通番（volume serial number）が必要である。ボリューム通番の書き方は、その機種あるいは計算センターの指示に従うものとする。

(5) SYSINステートメントのパラメタはすべて位置パラメタなので、記述する時には指定の順序で書かなければならない。省略は許さない。ただし第3のパラメタのみは、省略すればDAとみなす。

B－5 OFILEステートメント

このステートメントはコンパイルのジョブステップ内で結果のオブジェクトファイルをユーザーの指定するカタログファイルに保存しておくためのファイルに関する情報を与えるステートメントである。

$OFILE<空白><ファイル名>[,{NEW / OLD}[,{DA / MT,<ボリューム通番>}]]

(1) ファイル名とはオブジェクトファイルを保存しておこうとするファイルの名前で英字で始まる8文字以内の英数字のストリングである。

(2) 第2のパラメタは、このカタログファイルをここで創成するのならばNEW、既に存在しているものならばOLDを指定する。もし省略した場合にはOLDとみなす。

(3) 第3のパラメタはこのカタログファイルの媒体を示すものでDAは大記憶、MTは磁気テープをあらわす。MTの場合にはそのボリューム通番が必要である。ボリューム通番の書き方は、その機種あるいは計算センターの指示に従うものとする。省略時にはDAとみなす。

(4) これらのパラメタはすべて位置パラメタである。

B－6 ステートメントの順序について

コンパイルのジョブステップは各コンパイラを指定するステートメント（FORTRAN、COBOL、PL1）で始まる。その中でSAVEを指定している場合には、OFILEステートメントはSYSINステートメントあるいはソースカードデックに先立って指定しなければならない。

例 $FORTRAN
$OFILE ファイル名, …

```
$SYSIN          ファイル名(      )…
```

B－7　コンパイルに関するジョブ制御言語の例

ここで実際にコンパイルを行なう場合に標準ジョブ制御ステートメントをどのように用いるのか例をあげて説明することにする。

例　1

FORTRANによるソースプログラムをカードデックとして入力する場合。

```
$JOB          JOB01, A001, NAME1
$FORTRAN
```

FORTRAN　ソースプログラム

```
:
```

例　2

FORTRANによるソースプログラムをカードデックとして入力する場合。（ただし例1と異なり、プログラムはMAINとSUBの2つのサブプログラムから成る。）

```
$JOB          JOB02, A001, NAME1
$FORTRAN
```

FORTRANソースプログラム　（MAIN）

```
$FORTRAN
```

FORTRANソースプログラム　（SUB）

```
.
.
.
```

例1と例2は共にソースプログラムをカードから入力する場合の例であるが、例1ではソースプログラムが一つのプログラムから成り、例2では、二つのサブプログラムに分かれている。標準ジョブ制御言語では、ソースプログラムがいくつかのサブプログラムに分かれている場合には、それらをサブプログラムごとに分けてコンパイルする必要がある。この規則に反していくつかのサブプログラムをまとめてコンパイルしようとする場合、対象となる機種によっては可能であるかもしれないが、それはその機種の特殊性によるものと解釈されたい。

ソースプログラムがPL1、COBOLの場合についても同様である。

例　3

既にユーザのカタログファイルに入っているCOBOLによるソースプログラムをコ

ンパイルする場合。

```
    $JOB            JOB03, B020, NAME2
  ($PASSWORD        ABCDE, XXYYZZ     )
    $COBOL          , M
    $SYSIN          AFILE(CPROG), DA
    :
```

これはユーザのカタログファイルAFILE内に入っているCOBOLプログラムCPROGのコンパイルを行なうものである。$COBOLのオペランドのMはCPROGが1000～3000枚程度の大きさのソースプログラムカード枚数を持つことを示している。（対象機種がACOS－6の場合、カタログファイルの識別のために$PASSWORDが必要である。）

例 4

例3と同様にユーザのカタログファイルに入っているFORTRANによるプログラムをコンパイルする。ただし、ソースプログラムはBFILEというファイル中にA1、B2、C3、D4、E5の順に入っているものの中からB2、D4、E5およびCFILEの中にXX、YY、ZZと入っている中のYYの4個を用いるものとする。

```
    $JOB            JOB04, C111, NAME3
  ($PASSWORD        ABX, X0012)
    $FORTRAN        , M                          }…①
    $SYSIN          BFILE(B2, D4, E5), DA        }
    $FORTRAN        , S                          }…②
    $SYSIN          CFILE(YY), DA                }
    :
```

SYSINステートメントでは一度に1個のファイルしか指定できないので、BFILE、CFILEに対してそれぞれ別々のジョブステップを定義しなければならない。BFILE中のサブプログラム名の指定の順序は、もともとBFILE中に並んでいる順に指定しなければならない。そうでないと対象となる機種によっては不都合が生じるおそれがある。

例 5

XFILE中に入っているAPPLEというソースプログラムおよびカードデックの形のORANGEというプログラムをコンパイルする。

```
    $JOB            JOB05, EX00, NAMEX
  ($PASSWORD        ABC, A123         )
```

```
$PL1
$SYSIN          XFILE(APPLE),DA          …①
$PL1
┌──────────────────────────┐
│  PL1ソースプログラム      │            …②
│                (ORANGE)  │
└──────────────────────────┘
:
```

ソースプログラムがカードからの場合とファイルからの場合を合わせる例で、この場合ジョブステップを分ける必要がある。①と②のどちらが先でもかまわない。

例　6

FORTRANのソースプログラムをカードから読んでコンパイルし、結果はユーザの新しいカタログファイルOBJECTFへ入れる。このファイルは大記憶に作られるものとする。

```
$JOB            JOB06,EX10,NAMEZ
($PASSWORD      ABCD,XXX)
$FORTRAN        GRAPE,SAVE
$OFILE          OBJECTF,NEW,DA
┌──────────────────────────┐
│  FORTRAN ソースプログラム │
└──────────────────────────┘
:
```

FORTRANステートメントのSAVEパラメタとOFILEステートメントによってこの結果のオブジェクトプログラムはOBJECTFに入ることになる。

オブジェクトプログラム名はGRAPEである。ここでSAVEパラメタは、SAVE(I)と書いたのと同じである。

C．リンクに関するステートメント

リンクエディットを行なうために標準ジョブ制御言語ではLINKステートメントを定義している。また、ユーザのカタログ済のライブラリを編集の際に参照する場合には、USERLIBステートメントを用いる。リンクの結果として得られる実行形式プログラムをユーザのカタログ・ファイルに保存しておきたい場合には、LINKステートメントにSAVEパラメタを付けたのち、そのファイルを指定するEFILEステートメントを与えなければならない。

次にこれらのステートメントについて述べることにする。

C－1　LINKステートメント

このステートメントは前のジョブステップでコンパイルの結果でき上ったオブジェクトプログラムを（必要ならばライブラリを参照して）リンクし、実行形式プログラムを作成するリンクのジョブステップを定義するものであり、その一般形は次のとおりである。

$LINK<空白>〔<実行形式プログラム名>〕〔，SAVE〔（{I/A}）〕〕

(1) 実行形式プログラム名はリンクの結果得られる実行形式プログラムの名前で英文字で始まる6文字以内の英数字の列である。SAVEパラメタを指定した場合は必らず書かなければならない。これは位置パラメタである。

(2) SAVEパラメタは実行形式プログラムを保存しておく場合に指定するパラメタでその場合これと共に次のステートメントEFILEでその入るべきファイル名を定義しなければならない。SAVEパラメタを指定しても、EFILEステートメントがなければ、SAVEパラメタは無効となり、したがってプログラムの保存は行なわれない。

SAVEのあとにサブパラメタとして括弧で囲んでAかIを付けることができる。Iはイニシャル・モード、Aはアディショナル・モードを示し、EFILEステートメントで指定するファイルがOLDファイルであるとき、Iならばそこに今まで入っていた内容をこわして、新しい実行形式プログラムに変えることを意味する。Aならば、そこに今まで入っていた内容を破壊することなく、その後にここで作る実行形式プログラムを入れることを示している。

EFILEがNEWファイルならば、常にIであるはずであり、（I）は省略してかまわない。

またEFILEがOLDのときには、このサブパラメタを省略するとAとみなす。

EFILEの指定 / SAVEのサブパラメタ	NEW	OLD
（I）	新しいファイルを登録し新しい実行形式プログラムファイルを作る	指定のファイルの内容は新しいものに置き換わる
（A）	ありえない。エラーメッセージを出し（I）とみなす	指定のファイルの内容に続けて実行形式プログラムを追加する
省略時の解釈	（I）	（A）

(3) この標準ジョブ制御言語のこの版では、LINKに関するジョブステップは、ジョブにおける第1番目のジョブステップとはなり得ない。すなわち、常にこの前段階でコンパイルが行なわれているものとする。

C－2　EFILE ステートメント

このステートメントはリンクの際に実行形式プログラムを保存するファイルを定義するものであり、一般形は次のとおりである。

$$\$EFILE\langle空白\rangle\langle ファイル名\rangle\left[,\left\{\begin{matrix}NEW\\OLD\end{matrix}\right\}\left[,\left\{\begin{matrix}\underline{DA}\\MT,\langle ボリューム通番\rangle\end{matrix}\right\}\right]\right]$$

パラメタ等の説明については前節のＯＦＩＬＥステートメントに同じなのでその項を参照されたい。

C－3　USERLIB ステートメント

リンクのジョブステップでユーザが前以って作成したオブジェクトライブラリを参照する時に用いるステートメントであり、一般形は次のとおりである。

$$\$USERLIB\langle空白\rangle\left\langle\begin{matrix}オブジェクト\\ライブラリ名\end{matrix}\right\rangle\left[,\left\{\begin{matrix}\underline{DA}\\MT,\langle ボリューム通番\rangle\end{matrix}\right\}\right]$$

(1) 第１のパラメタは参照するオブジェクトライブラリ名である。これは位置パラメタで勿論省略不可である。

(2) 第２のパラメタはそのファイルの存在する媒体を示すものでＤＡは大記憶、ＭＴは磁気テープファイルを表わす。省略時にはＤＡとみなす。

(3) 参照したいオブジェクトライブラリが多数ある場合はこのＵＳＥＲＬＩＢステートメントをその個数だけ並べればよい。但し機種によりその数に制限のあるものがある。

C－4　リンクに関するジョブ制御言語の例

ここで実際にリンクを行なう場合に標準ジョブ制御ステートメントをどのように用いるのか例をあげて説明することにする。

例　1

ＦＯＲＴＲＡＮプログラムをコンパイルしリンクする。この時ユーザのオブジェクトライブラリ３個を参照する。

```
$JOB            JOB201,AB01,NAME1
($PASSWORD      ABCDE,XXXZZ          )
$FORTRAN
        FORTRAN ソースプログラム
$LINK
$USERLIB        AB01,DA
$USERLIB        AB02
$USERLIB        AB03,MT,テープのボリューム通番
:
:
```

例　2

COBOLプログラムをコンパイルし、リンクする。この時できた実行形式プログラムをTEST1と名付けてカタログファイルに保存する。リンクの際ユーザのオブジェクトライブラリを参照する。

```
$JOB            JOB202, AB02, NAME2
($PASSWORD      ABCDEF, X0011       )
$COBOL
```

COBOL　ソースプログラム

```
$LINK           TEST1, SAVE
$EFILE          EXECF1, NEW, DA
$USERLIB        CB01
$USERLIB        CB02
:
```

LINKステートメントのSAVEパラメタとEFILEステートメントによってEXECF1ファイルを創成してこれにTEST1と名付けた実行形式プログラムを保存することになる。

ステートメントの順序はEFILEステートメントのある時にはこれをLINKの次に記し、そのあとUSERLIBを書き並べる。

D．ユーザプログラムの実行に関するステートメント

コンパイル、リンクを行ない、いよいよユーザプログラムを実行するにあたっては、実行ジョブステップを定義するEXECステートメントを用いる。また、ユーザプログラムがファイルを取り扱う場合には、FILEステートメントで定義する。

尚、標準的な入出力としてカードリーダとラインプリンタを用いる場合は特にFILEステートメントを用いて定義する必要はない。

次にEXECおよびFILEステートメントについて述べる。

D－1　EXECステートメント

このステートメントはユーザプログラムの実行を行なうジョブステップを定義するステートメントであり、一般形は次のとおりである。

$EXEC＜空白＞〔＜ファイル名＞(＜プログラム名＞)〕〔, PARAM='　　'〕

(1)　ファイル名とはこれから実行しようとする実行形式プログラムがユーザのカタログファイル内にある場合、その所在を示すファイル名である。

(2) プログラム名は、実行形式プログラム名であり、括弧で囲んで記す。前段階でコンパイル・リンクを行ない、特に実行形式プログラムに名前をつけていない場合には、ファイル名・プログラム名共に省略可能である。

(3) 第3パラメタのPARAM＝'　　'の引用符の内容は、実行時にユーザープログラムに渡たされる性質のものである。ただし、このパラメタは対象機種によっては、そのような機能のないものもあり、無効となることもある。

D－2　FILEステートメント

このステートメントはユーザプログラムの実行の際に入出力ファイルを指定するためのステートメントであり一般形は次のとおりである。

$FILE{<機　番>, <ファイル定義名>}, [<ファイル名>] [, {NEW, OLD, TEMP, CATLG}] [, UNIT={入出力装置種類, 〃 機番}]

[, VOL＝ボリューム通番] , DISP={KEEP, DELETE, CATLG} [, {CREADER, PRINT, PUNCH}]

[, SPACE={m [{TRK, CYL, ブロック長} [, データ領域] [, 増分, インデックス領域]]}]

[増分, インデックス領域]] [, {SIDP}] [, {UVF}]

[, RCDSIZE＝1レコードの大きさ] [, BLKSIZE＝1ブロックの最大長]

[, {BCD, JIS, EBCDIC}]

ファイルに関する記述はコンピュータによってさまざまにきまっており、これを簡単に記述する目的で制限することは、そのコンピュータの機能を著しく損うことと思われるので、このFILEステートメントにもいろいろなパラメタを用意した。

ここでどうしても必要なものは第1第2のパラメタであり、第1パラメタはFORTRANにおける処理装置を示す機番またはPL1、COBOLにおけるファイル定義名である。第2パラメタのファイル名はPRINTおよびPUNCHを指定する場合以外は必らず書かなければならないし、PRINT、PUNCHの場合も対応するカンマは必要である。S、I、D、Pはファイル編成を示すパラメタでSは順編成、Iは順インデックス、Dは直接編成、Pは分割型順編成を示す。

U、V、Fはレコード編成を示し、Fは固定長、Vは可変長、Uは不定長を示す。

SPACE＝mの指定は機種によって単位が異なる場合にこの書き方を用いる。例えばACOSではファイルスペースのとり方は３８４０ワードを１リンクとしているのでSPACE＝１０と書けば１０リンク、３８４００ワードをあらわす。

D－３　ユーザプログラムの実行に関するジョブ制御言語の例

ここでユーザプログラムを実行する場合にどのように標準ジョブ制御言語を用いるのか、例をあげて示すことにする。

例　1

FORTRANプログラムをコンパイルし、リンクし実行する。（ユーザプログラムでの実行はカードリーダとラインプリンタを標準使用で用いるものとする。）

```
$JOB            JOB301, AA00, NAMEA
($PASSWORD      ABCDE, ABC          )
$FORTRAN
```

FORTRAN　ソースプログラム

```
$LINK
$USERLIB        F100, DA
$EXEC
```

デ　　ー　　タ

```
$END
```

これはFORTRANプログラムの典型的な使用例である。ユーザプログラムの実行の際、特別なファイルを用いていないので単にデータ用カードデックを挿入するだけでよい。

例　2

FORTRANプログラムですでにユーザの実行形式プログラムライブラリFELIBに入っているEX１というプログラムの実行のみを行なう場合（実行にあたってはデータをカードリーダおよびFL１というファイルから読みFL２ファイルへ書き出すものとする。ラインプリンタ出力もある。）

```
$JOB            JOB302, ABX, NAME10
$EXEC           FELIB(EX1)
$FILE           10, FL1, OLD, UNIT=DA, DISP=KEEP     …①
$FILE           11, FL2, NEW, UNIT=DA, DISP=KEEP,    …②
                SPACE=10, S
```

デ　　ー　　タ

```
$END
```

②はACOS用ファイルでＦＬ２という１０リンク順編成の大記憶ファイルを示し、FORTRANプログラム中で機番１１でとり扱かっていることを示す。

例　3

COBOLプログラムをコンパイルし、リンク、実行する。（プリンタのファイル定義名はＰＴとする。）

```
$JOB              JOB303, ABC, NAME11
($PASSWORD        ABCDE, XXX          )
$COBOL
```

COBOL　ソース　プログラム

```
$LINK
$USERLIB          XC01, DA
$EXEC
```

デ　　ー　　タ

```
$FILE             PT, , PRINT                    …①
$END
```

①はCOBOLのプリンタ出力用ファイルを示す。ソースプログラム中のAssign文で指定したファイル定義名ＰＴをジョブ制御システムに伝えるためのステートメントである。

E．プログラムの更新に関するステートメント

ソースプログラムが大変に大きいときなど、毎回カードデックを読ませるのが煩わしい場合がある。こういう場合には、ユーザはソースプログラムをファイルに入れておき、ソースに変更が起った時にはそのファイルを更新する形をとることもよく行なわれることであろう。またでき上ったプログラムをコンパイルし、オブジェクトプログラムの形でファイルに入れたものをあとで一部削除したいという場合も生じることがある。このような時に使用するステートメントとして、標準ジョブ制御言語ではEDITステートメントを用意している。これにはソースプログラムに関するものとしてEDITS、オブジェクトプログラムに関するものとしてEDITO、実行形式プログラムに関するものとしてEDITE、さらにファイルそのものを消

去するものとしてEDITDがある。以下この各ステートメントについて述べることにする。

E－1　EDITS ステートメント

EDITS ステートメントは、ソースプログラムをファイルに移すこと、ファイル上にあるソースプログラムの更新、あるいはソースプログラム内のステートメントの更新を行なうステートメントであり、一般形は次のとおりである。

$$\$EDITS\langle \overset{空}{白} \rangle \left[\langle ファイル名1 \rangle \left[, \left\{ \begin{array}{l} \underline{DA} \\ MT, \langle ボリューム通番 \rangle \end{array} \right\} \right] \right]$$

$$\$NFILE\langle \overset{空}{白} \rangle \left[\langle ファイル名2 \rangle \left[, \left\{ \begin{array}{l} NEW \\ \underline{OLD} \end{array} \right\} \left[, \begin{array}{l} \underline{DA} \\ MT, \langle ボリューム通番 \rangle \end{array} \right\} \right] \right]$$

このあとに

EDITSに対するコントロールカードを含む 更新ソースプログラム　デック

あるいは

$$\$TFILE\langle \overset{空}{白} \rangle \langle ファイル名3 \rangle \left[, \left\{ \begin{array}{l} \underline{DA} \\ MT, \langle ボリューム通番 \rangle \end{array} \right\} \right]$$

のいずれか一方を記述する。

(1) ここでファイル名1は、ソースプログラムライブラリとなっているマスタファイルの名前であり、このマスタファイルが存在して、このジョブ・ステップ内でこれに対するプログラムの追加、置き換え、削除、あるいはこのファイル内のプログラムに対するステートメントの更新が行なわれることを意味する。このファイル名1が省略された場合は、ソースライブラリとしてのマスタファイルが存在せず、従ってこのジョブステップ内で与えられるソースプログラムカードデックあるいはトランザクションファイルから新しくソースライブラリ用のマスタファイルを作成することとみなす。

(2) ファイル名は、ファイル名1を基にするかあるいは全く新しく、作り出すソースプログラムライブラリの新しいマスタファイルの名前である。このファイル名が省略されるか、あるいはファイル1と等しい場合には、ファイル1そのものを更新することを示す。ファイル名2を別に指定する場合には、この新しいファイルがホストコンピュータの中にまだ登録したことのない新しい名前のファイルである場合にはNEW、ファイル名だけは既に登録されていてこの度のジョブステップで内容を一新する場合にはOLDを指定すること。OLDの時にはファイル2は、そのファイル自体の創成時にソースプログラムライブラリ用に登録されたものでなければならない。このパラメタは省略可能で省略時にはOLDとみなす。但しDA、MT等のパラメタと並記する時には、NEW、OLD のパラメタを先

に書かねばならない。

(3) 各ファイル名を指定する場合には、そのファイルの存在する媒体を示すパラメタが必要である。大記憶ファイルの場合はＤＡ、磁気テープファイルの場合はＭＴでこれを示し、後者のときには、ボリューム通番を添えるものとする。省略時にはＤＡとみなす。

(4) ＄ＥＤＩＴＳおよび＄ＮＦＩＬＥステートメントを用いて二つ（あるいは一つ）のマスタファイルを指定したのち、ソースプログラム自体の更新あるいは、プログラム内のステートメントの更新を行なおうとする場合は＄ＥＤＩＴＳに対する次のようなコントロールカードと共にソースプログラム（ステートメント）のカードを与えることになる。コントロールカードは以下のとおりである。

ＥＤＩＴＳに関するコントロールカードは第1カラムに／（スラッシュ）、第2カラムに＄をおき、第3カラムからコントロール用キーワードを書いたのち、1個以上の空白をあけてから、プログラム名を記し、そのあとプログラム名を書く。この後に、プログラム自体の更新の場合にはそのソースプログラムがＦＯＲＴＲＡＮかＣＯＢＯＬかＰＬ１かを示すＦ、Ｃ、Ｐの中の1文字をカンマで区切った後に書く。ステートメントの更新の場合にはプログラム名の後に、カード番号を括弧で囲んで記す。

標準ジョブ制御言語ステートメントの一般規則に準じて、プログラム名及びカード番号は、空白を含まない一連のストリングであるが、カンマや括弧の前後には任意個の空白があってもかまわない。

ここでカード番号とは、ソースプログラムのシークェンスナンバーの場合と、コンパイルリスト上にコンパイラが出力したものとの二通りの場合がありうるが、そのいずれであるかは対象機種の指定による。したがって整数の場合と頭に英文記号のついた整数の場合がありうる。

(イ) プログラムあるいはステートメントの置き換え

一般形は次のとおりである。

$$／\$\ \begin{Bmatrix}\text{REPLACE}\\ \text{R}\end{Bmatrix}\ \langle\text{空白}\rangle\langle\text{プログラム名}\rangle\left[\left(\langle\text{カード番号1}\rangle\left[,\langle\text{カード番号2}\rangle\right]\right)\right],\begin{Bmatrix}\text{F}\\ \text{C}\\ \text{P}\end{Bmatrix}$$

置き換えを示すキーワードＲＥＰＬＡＣＥは単にＲと略してよい。プログラムの置き換えを行なう場合にはプログラム名とコンパイル言語の種類を示す文字を示す。ステートメントの置き換えの場合には、その置き換える区間をカード番号1で始まり、カード番号2で終わるように括弧でくくってプログラム名と共に指定し、コンパイル言語の種類のコードをつけ加える。いずれの場合にもこのコントロールカードの後には、置き換わるべきカードまたはカードデックを付けなければならない。置き換えるカードが1枚の時にはその番号を括弧でくくればよい。

(ロ) プログラムあるいはステートメントの追加

一般形は次のとおりである。

$$/\$\ \left\{\begin{matrix}\text{INSERT}\\ \text{I}\end{matrix}\right\}\langle\text{空白}\rangle\langle\text{プログラム名}\rangle\left[\ (\langle\text{カード番号}\rangle)\ \right],\left\{\begin{matrix}\text{F}\\ \text{C}\\ \text{P}\end{matrix}\right\}$$

追加、挿入を示すキーワードINSERTは単にIと略記してもよい。プログラム自体を追加する場合、あるいは厳密には追加とは言えないが、一番はじめてマスタファイルを作る場合もこのコントロールカードを用いて、プログラム名およびコンパイル言語の種類を指定する。ステートメントの挿入の場合は、プログラム名の後に、挿入しようとするステートメントの直前になるステートメントのカード番号を括弧でくくって書き、そのあとにコンパイル言語のコードを示す。

(ハ) プログラムあるいはステートメントの削除

一般形は次のとおりである。

$$/\$\ \left\{\begin{matrix}\text{DELETE}\\ \text{D}\end{matrix}\right\}\langle\text{空白}\rangle\langle\text{プログラム名}\rangle\left[\ (\langle\text{カード番号1}\rangle\left[,\langle\text{カード番号2}\rangle\right])\ \right],\left\{\begin{matrix}\text{F}\\ \text{C}\\ \text{P}\end{matrix}\right\}$$

削除を示すキーワードDELETEは単にDと略記してもよい。プログラム自体を削除する場合には、プログラム名およびコンパイル言語の種類を指定する。ステートメントの削除の場合は、プログラム名と共に削除すべきステートメントの区間をカード番号で示し括弧でくくったものを添える。削除すべきカードが一枚のときはその番号を括弧でくくればよい。

(ニ) 注　意

プログラムに対する更新とステートメントに対する更新は、同一のEDITSに関するジョブステップで行なうことができる。ただし、ソースプログラムライブラリのマスタファイルはシークェンシャル編成になっている場合があるので、プログラム及びその中でのステートメントに関する更新情報はもともとのファイルに入っている順序（すなわち、シークェンシャル）に与えなければならない。したがってプログラムを追加する場合は、旧マスタファイル（ファイル名1で示す）の内容の直後からつけ加えられるものとする。詳しくは例を参照のこと。

(5) ソースプログラム用のライブラリファイルが2本あってこれをひとつにまとめたいときには、その一方をファイル名1で指定し、他方をTFILEステートメントで指定する。このTFILEを示した場合には、カードデックによるソースプログラムの更新を同時に行なうことはできない。TFILEには、そのファイル名及び媒体を示すパラメタを指定する。三つ以上のソースライブラリをひとつにまとめるときには、1回のEDITSジョブステップは2本ずつしかまとめられないので何回かに分けて実行するようにされたい。

またこの場合ＴＦＩＬＥで指定するファイル中にＥＤＩＴＳステートメントのオペランド部で指定したファイル１の内容と同一のプログラム名を持つものが存在する場合にはＴＦＩＬＥ内のものの方がとられ、マスタファイル内の同名のプログラムは抹消される。

Ｅ－２　ＥＤＩＴＳ ステートメントの使用例

ＥＤＩＴＳ ステートメントの代表的な使用例を以下に示すことにする。

例　１

ＦＯＲＴＲＡＮによる、ＭＡＩＮ、ＳＵＢ１、ＳＵＢ２、ＳＵＢ３、ＴＥＳＴをＳＦＩＬＥ１（未登録の新しいファイル）に入れて、新しいソースプログラムライブラリを作成する。

```
$EDITS                                  …マスタライブラリファイルなし
$NFILE      SFILE1, NEW, DA             …新しいライブラリファイルの定義
/$INSERT    MAIN, F
```

ＦＯＲＴＲＡＮ　ソースプログラム　（ＭＡＩＮ）

```
/$I         SUB1, F
```

ＦＯＲＴＲＡＮ　ソースプログラム　（ＳＵＢ１）

```
/$I         SUB2, F
```

ＦＯＲＴＲＡＮ　ソースプログラム　（ＳＵＢ２）

```
/$INSERT    SUB3, F
```

ＦＯＲＴＲＡＮ　ソースプログラム　（ＳＵＢ３）

```
/$I         TEST, F
```

ＦＯＲＴＲＡＮ　ソースプログラム　（ＴＥＳＴ）

この時にはＳＦＩＬＥ１の名前はユーザのファイルとして未定義なのでＮＥＷというパラメタを記す。

例　2

例1で作成したＳＦＩＬＥ１の中のプログラムＳＵＢ２を削除しＳＵＢ４を追加する。またＳＵＢ１プログラムの（１００～１０５）を別のカードと入れ換え、１２０以下に挿入を行ない、（１５０～１６０）を削除する。又ＴＥＳＴプログラムは同一名の別のプログラムを置き換える。新しいソースプログラムファイルはＳＦＩＬＥ２（これも未登録の新しいファイル）に入れることにする。

```
$EDITS        SFILE1, DA
$NFILE        SFILE2, NEW, DA
/$REPLACE     SUB1(100, 105), F
```

ソースカードデック（入れかえ）

```
/$INSERT      SUB1(120), F
```

ソースカードデック　（挿　入）

```
/$DELETE      SUB1(150, 160), F
/$DELETE      SUB2, F
/$REPLACE     TEST, F
```

ソースプログラム（TEST）　（入れかえ）

```
/$INSERT      SUB4
```

ソースプログラム　（SUB4）

例　3

ＦＯＲＴＲＡＮソースプログラムＭＡＩＮ、ＸＹＺ、ＡＢＣを例１で定義したＳＦＩＬＥ１の内容を消して、この中に入れる。

```
$EDITS
$NFILE        SFILE1, OLD, DA
/$I           MAIN, F
```

ソースプログラム　（MAIN）

```
/$I          XYZ, F
```

ソースプログラム（XYZ）

```
/$I          ABC, F
```

ソースプログラム（ABC）

例　4

例2で作ったSFILE2に例3で作ったSFILE1をつけ加える。

```
$EDITS       SFILE2, DA
$NFILE                                   …（SFILE2を更新する）
$TFILE       SFILE1, DA
```

この結果SFILE2には、SUB1、SUB3、TEST、SUB4、MAIN、XYZ、ABCの各プログラムがこの順でセットされる。MAINはSFILE1内のものがとられる。

E－3　EDITOステートメント

EDITOステートメントは、オブジェクトプログラムファイルに関する更新、すなわちオブジェクトプログラムの削除、および二つのオブジェクトプログラムファイルをひとつにまとめる作業を指示するステートメントで一般形は次のとおりである。

$EDITO<空白><ファイル名1> [, { DA / MT, <ボリューム通番> }]

$NFILE<空白> [<ファイル名2> [, { NEW / OLD } [, { DA / MT, <ボリューム通番> }]]]

このあとに

EDITOに対して削除するプログラム名を与えるコントロールカード

/$ { DELETE / D } <空白><プログラム名>

を続けるか、あるいは

$TFILE<空白><ファイル名3> [, { DA / MT, <ボリューム通番> }]

のいずれか一方を記述する。

(1)　ここでファイルやその他のパラメタについては、ファイルがオブジェクトファイルであ

る点を除いてほぼＥＤＩＴＳに等しいので詳しくはその項を参照されたい。

(2) ファイル名１は、オブジェクトプログラム用のマスタファイルの名前であり、EDITOステートメントにおいてはこの名前は必らずなければならない。なぜならば、オブジェクトプログラムは、コンパイルの結果として得られるものでソースプログラムとは異なりユーザーが直接与えることはできないので、ＥＤＩＴＯステートメントで取り扱うのはマスタファイルにトランザクションファイル内のプログラムを追加、あるいは置き換えを行なって新しいマスタファイルを作成する場合、およびマスタファイル内から特定のプログラムを削除する場合に限られるからである。

(3) ファイル名２は、更新の結果作られる新しいオブジェクト用マスタファイルの名前であり、ファイル名１と等しくてもよい。ファイル名１と等しい場合には省略可能である。ファイル名２を別に指定する場合には、この新しいファイルがホストコンピュータの中にまだ登録したことのない新しい名前のファイルである場合には、NEW、そうでなければＯＬＤを指定する。ファイル名２がファイル名１のファイルと異なるものでＯＬＤ である時、ファイル名２のファイルにもともと入っていた内容は、このＥＤＩＴＯの実行によって、消されて、そのあとに更新された内容が入るので注意されたい。

(4) ＄ＥＤＩＴＯおよび＄ＮＦＩＬＥステートメントを用いて二つ（あるいは一つ）のマスタファイルを指定したのち、古いマスタファイルからプログラムを削除したい場合には

／＄ { DELETE / D } ＜空白＞＜プログラム名＞

のカードを続ける。この書き方については＄ＥＤＩＴＳに対するコントロールカードの 項を参照されたい。このカードは削除するプログラムの数だけ書き並べることができるが、オブジェクトプログラム用マスタ・ファイルはシークェンシャル編成になっている場合があるので、プログラム名はマスタファイル内の順序に従って与えなければならない。

(5) オブジェクトプログラム用のマスタファイルが２本あり、これをひとつにまとめたいときには、その一方をファイル名１で指定し他方をＴＦＩＬＥステートメントで指定する。このＴＦＩＬＥ を指定した場合にはコントロールカードでプログラム削除を同時に指定することはできない。

三つ以上のオブジェクト用マスタファイルをひとつにまとめるには、１回のＥＤＩＴＯジョブステップでは、２本ずつしかまとめられないので、何回かに分けて実行するようにされたい。

またＴＦＩＬＥで指定するファイル内にＥＤＩＴＯステートメントのオペランド部で指定したマスタファイル１内のプログラム名と同じものがある場合、古いマスタファイル（ファイル１）内のものは削除されＴＦＩＬＥの方がとられる。

E－4　EDITO ステートメントの使用例

EDITO ステートメントの例を次に示すことにする。

例　1

オブジェクト用マスタファイルOBJECT1内にX、Y、Z、A、B、Cの6個のプログラムが入っているものとする。（これらが初めてOBJECT1に入るのは、コンパイルの結果としてである。）この中からZとBを削除する時は次のようになる。

```
$EDITO        OBJECT1
$NFILE        OBJECT2, NEW, DA
/$DELETE      Z
/$D           B
```

例　2

オブジェクト用マスタファイルOB001とOB002があり、OB001にはA、B、C、D、E、F、OB002にはC、E、I、J、Kの各プログラムが入っているとき、これをマージしてOB001を更新する。

```
$EDITO        OB001
$NFILE
$TFILE        OB002, DA
```

これが実行されるとOB001にはA、B、D、F、C、E、I、J、Kの各プログラムが入るものとする。（機種によっては、A、B、C、D、E、I、J、Kの形になることも、ランダムファイルの場合もある） いずれにしてもCとEはもとのOB002 内のものである。OB002は変化しない。

例　3

オブジェクト用マスタファイルOB001内にA、B、C、D、E、F、G の7個のプログラムがあり、OB002の中にX、Y、Zの3個のプログラムがある。今、OB001からA、B、Eを削除したものとOB002をマージして新しいマスタファイルOB003を作成する。

このような場合、一度にはできないので2つのジョブステップに分ける。

```
$EDITO        OB001, DA
$NFILE        OB003, NEW, DA
/$DELETE      A
/$D           B
/$D           E
```

```
$EDITO      OB003
$NFILE      OB003
$TFILE      OB002
```

これを次のように書いても結果は同じである。

```
$EDITO      OB001,DA
$NFILE      OB003,NEW,DA
$TFILE      OB002
$EDITO      OB003,DA
$NFILE
/$DELETE    A
/$D         B
/$D         E
```

E－5　EDITEステートメント

EDITEステートメントは、実行形式プログラムファイルに関する更新、すなわち、実行形式プログラムの削除および二つの実行形式プログラムファイルをひとつにまとめる作業を指定するステートメントで、一般形は次のとおりである。

$EDITE<空白><ファイル名1>[,{DA / MT,<ボリューム通番>}]

$NFILE<空白>[<ファイル名2>[,{NEW / OLD}[,{DA / MT,<ボリューム通番>}]]]

このあとに

EDITEに対して削除するプログラム名を与えるコントロールカード

/${DELETE / D}<空白><プログラム名>

を続けるか、あるいは

$TFILE<空白><ファイル名3>[,{DA / MT,<ボリューム通番>}]

のいずれか一方を記述する。

(1) EDITEステートメントについては、ステートメント名、およびファイルが実行形式プログラムファイルであり、プログラムが実行形式プログラムであることを除いて、EDITOステートメントに等しいのでその項を参照されたい。

(2) 実行形式プログラムファイルは機種によってはMT上には作成できないことがあるので

注意されたい。

E－6　EDITDステートメント

カタログファイルとして用いるソースプログラムファイル、オブジェクトプログラムファイルおよび実行形式プログラムファイルは、一旦ホストコンピュータに登録すると、ファイル内のすべてのプログラムをEDITS、EDITO、EDITEなどのステートメントを用いて消し去っても、ファイル名だけは登録されたままになっている。これを抹消したい時にはEDITDステートメントを用いる。その一般形は次のとおりである。

$EDITD　　<ファイル名>

パラメタとしては抹消しようとするファイル名のみを指定する。通常、ファイルを消し去る必要のあるのは大記憶内のファイルの場合で、磁気テープの場合は、不要と思われるので、標準ジョブ制御言語のこの版ではEDITDのファイル名は大記憶ファイルに限るものとする。

例

ソースプログラムファイルSFILE1を抹消する。

$EDITD　　SFILE1

レベル　3

標準ジョブ制御言語の第3レベルは、ユーザにそのホストコンピュータ独自のジョブ制御言語の使用を許すものである。これを用いるためには、$ENTER、$EXITの2つのステートメントの間に、使用しようとしているジョブ制御言語の仕様に従ってジョブ制御ステートメントを書けばよい。

この$ENTERおよび$EXITステートメントに挟まれた部分に対しては、標準ジョブ制御言語システムでは、翻訳を行なわず、そのままホストコンピュータに転送する。

レベル3はレベル2と共に記述することができる。ただし、$ENTERから$EXIT迄を翻訳不要のひとまとまりのステートメントとみなすので、この中にレベル2のステートメントを混ぜて書くことはできない。

5.3.2　変換規則

標準ジョブ制御ステートメントは、標準ジョブ制御言語システムによって対象となる機種のジョブ制御言語に翻訳されることになっている。ここでは、ホストコンピュータをNEAC　ACOS 6の場合にとって、その変換規則を示すことにする。さらにファイルについては大記憶の場合の例で

示すことにする。

A－1　JOBステートメントについて

JOBステートメントは

$JOB＜空白＞＜ジョブ名＞，＜アカウント番号＞，＜識別名＞

〔，PRTY＝α〕〔，TIME＝{(m，n)／m}〕〔，PAGE＝p〕

〔，＜コア占有領域サイズ＞KB〕〔，NORUN〕

のように記すがACOS 6では$JOBに該当するステートメントとして

$　　SNUMB　＜ジョブ名＞，　＜緊急度＞

$　　IDENT　＜アカウント番号＞，＜識別名＞

をジェネレートする。ジョブ名、アカウント番号、識別名は、ユーザの指定通りとし、緊急度は、Z＝1、Y＝2、X＝3、……のように数字に変換して示す。

その他のパラメタは、ACOSシステムの場合は、ユーザプログラムの実行のときしか指定できないので、この時まで保存しておき、必要になったとき

$　　LIMITS　＜CPUタイム＞，＜メモリサイズ＞，，＜行数＞

の形でジェネレートする。

CPUタイムとしてTIME＝{(m，n)／m}を用いるがACOSの単位は1／100時間なのでm、nを換算した結果を記すことにする。メモリサイズは、ワード単位なのでユーザの指定サイズを4で割って切り上げたものとする。また、出力行数はユーザの指定したpを50倍したものとする。

またNORUNパラメタは、NJCLのレベル1でのみ指定できるパラメタで、翻訳すべき性質のものではなく、NJCLシステムの中でフラグとして用いるものである。

A－2　ENDステートメント

ENDステートメントに対してはACOSでは

$　　ENDJOB

***EOF

をジェネレートする。

A－3　PASSWORDステートメント

PASSWORDステートメントはACOSでどうしても必要なために作ったもので、

$PASSWORD＜空白＞＜システム・マスタ・ディレクトリ名＞，＜パスワード＞

のように記すが、ACOS用にはこれをそのまま

$　　USERID　＜システム・マスタ・ディレクトリ名＞$＜パスワード＞

に変換する。

A－4　JOB、END、PASSWORDステートメントの変換例

```
$JOB          JOBA, X0011, JIPDC, PRTY=X,
              PAGE=200, 120KB, TIME=(4, 30)
$PASSWORD     ABCDEF, XX12AB
 .
 .
 .
$END
```

以上のようなNJCLステートメントはACOSの場合、次のように変換される。

```
$    SNUMB       JOBA, 3
$    IDENT       X0011, JIPDC
$    USERID      ABCDEF$XX12AB
     .
     .
     .
                   *1  *2
$    LIMITS      8,   40,, 10000   ←ユーザープログラムの実行時
     .
     .
     .
$    ENDJOB
***EOF
```

＊1　4分30秒は7.5単位時間となるので切り上げて8となる。

＊2　120Kバイトは40Kワードである。

但し、ACOS、FORTRANの場合は、指定を省略した場合、次の値がとられる。

CPUタイム	5単位時間　（3分）
メモリサイズ	16Kワード（64KB）
印字行数	12000行　（240ページ）

B－1　FORTRANステートメント

FORTRANステートメントは

$FORTRAN<空白>[<オブジェクトプログラム名>][,<コンパイルパラメタリスト>]

<コンパイルパラメタリスト>::=<コンパイルパラメタ>|<コンパイルパラメタ>,<コンパイルパラメタリスト>

<コンパイルパラメタ>::=SAVE[(I/A)]|NOLIST|<プログラムサイズ>

＜プログラムサイズ＞：：＝Ｓ｜Ｍ｜Ｌ

のように記す。ＡＣＯＳの場合コンパイルを行なう時カードデックからソースプログラムを入力し、それをリンク実行し、後にプログラムを残さない場合と、入力がファイルの場合や何らかの形でプログラムファイルをＳａｖｅする場合とではＪＣＬが全く異なってしまう。ＮＪＣＬをＡＣＯＳのＪＣＬに順次変換して行く場合、後のジョブステップでファイルのＳａｖｅが行なわれるとするとそれを認識するのは、不可能なことなので、カードデックからの入力の場合にも一旦これをワークファイルに移し、これをコンパイルする形に統一する。

但しレベル１では、ＮＪＣＬでも条件を設定しているのでＡＣＯＳのＪＣＬに変換する際直接コンパイルする方法を用いる。これについては後述する。

(1) プログラムサイズの指定があった場合、ファイルの大きさ（下限）をきめる時にこれを用いる。

	ソースプログラム	オブジェクトプログラム	実行形式プログラム
Ｓ	３Ｌ	３Ｌ	４Ｌ
Ｍ	９Ｌ	９Ｌ	１２Ｌ
Ｌ	３０Ｌ	３０Ｌ	４０Ｌ

単位はＬ（リンク）（３８４０ワード）である。

以下の説明ではプログラムサイズをＳとして変換の例を示していくことにする。

(2) $FORTRAN

ソースプログラム

の場合を考える。この場合にはＡＣＯＳの次のようなＪＣＬに変換する。

```
$   FILEEDIT   SOURCE, OBJECT, INITIALIZE      ①
$   FILE       R*, O1S, 3L                     ②
$   DATA       *C, , COPY                      ③
$   FORTRAN                  (73カラムより)F01  ④
      ⎧ FORTRAN ソースプログラム
$   ENDEDIT                                    ⑤
$   ENDCOPY                                    ⑥
```

図５－４（Ｂ－１）

(3) $FORTRAN　<プログラム名>

のようにプログラム名の指定がある場合には、上の6.3.2－B－1図において④の$　FORTRANの第73カラムからF01の代わりにそのプログラム名を入れる。

またNJCLの$FORTRANのあとにカードデックの続いたジョブステップが複数個続いた場合には、プログラム名の指定していないカードデックに対してF01、F02、F03、…と順に名前をNJCLシステムがジェネレートしていく。

(4) $FORTRAN　<プログラム名>, NOLIST

のときには6.3.2－B－1図の④の代わりに

$　FORTRAN　NLSTIN　（73カラムより）プログラム名

をジェネレートする。

(5) 次にソースがカードデックでなくファイルの場合を考える。このときNJCLでは、次のように書く。

$FORTAN

$SYSIN　<ファイル名>（<プログラム名列>）[,{DA / MT, <ボリューム通番>}]

<プログラム名列>::=<プログラム名>|<プログラム名>, <プログラム名列>

このときには先ずファイルの中から指定されたプログラムをとり出しワークファイルに入れるジョブステップをジェネレートしてから、コンパイルに移る。すなわち、

```
$   PROGRAM    NJCL.1                                          ⑦
$   PRMFL      **, R, R, DRP/NJCLEDIT01                        ⑧
$   PRMFL      IN, R, S, <システム・マスタ ディレクトリ名>/<ファイル名1>  ⑨
$   FILE       OT, S1S, 3L, DKU411                             ⑩
$   DATA       I*                                              ⑪
      <プログラム名1>                                          ⑫
      <  〃   2>
          .
          .
          .
```

図5－4（B－2）

このあとコンパイル用のJCLをジェネレートする。

```
$   FILEDIT    SOURCE, OBJECT, INITIALIZE    ⑬
$   FILE       R*, O1S, 3L                   ⑭
$   FILE       *C, S1R                       ⑮
```

図５－４（Ｂ－３）

ACOSにはソースプログラムをとり扱うために、ソースプログラム・ライブラリ・エディタというシステムがあるが、これを用いるためには、ファイル内のプログラムすべての名前と順序がわかっていないと必要なプログラムを選び出すことができないので、これに代わって指定ファイルから指定のプログラムのみをとり出し、ワークファイルに移すRun time Routine をＮＪＣＬ用ほ作ることにした。⑦のＮＪＣＬ．１はこのルーチン名である。⑧～⑪はこのＮＪＣＬ．１を動かすために必要なＪＣＬであり、プログラム名は⑫以下に順に書き出すことにする。

このとき作ったワークファイル（ファイル定義名Ｓ１）をソースプログラム・ライブラリ・エディタを介してコンパイルするようなＪＣＬを作り出しているのがＢ－３図である。

ＦＯＲＴＲＡＮであることの指定はソースプログラムをファイルに入れる時に既に ソースプログラムと共にファイル内にセットしてある。

(6) 次にＳＡＶＥパラメタのある場合、すなわち、オブジェクトプログラムファイルを作成する場合を考える。この時、ＮＪＣＬでは次のような書き方をする。

```
$FORTRAN   <プログラム名>, SAVE [( {I} ) ]
                                 {A}
$OFILE     <ファイル名2> [, {NEW} [, {DA                    } ]]
                          {OLD}  {MT, <ボリューム通番>}
           ソースプログラム
```

あるいは

```
$FORTRAN   , SAVE [( {I} ) ]
                   {A}
$OFILE     <ファイル名2> [, {NEW} [, {DA                    } ]]
                          {OLD}  {MT, <ボリューム通番>}
$SYSIN     <ファイル名1> …
```

ＯＦＩＬＥの指定がＮＥＷであれば先ず次のＪＣＬをジェネレートする。（ＯＬＤのときにはこのジョブステップは作り出さない。）

$　FILSYS	
USERID	<システム・マスタディレクトリ名>$<パスワード>
FCREAT	<システム・マスタディレクトリ名>/<ファイル名>,/LINKS/3,10/

図5－4（B－4）

このあと前者ならばB－1図、後者ならばB－2、B－3図のJCLをジェネレートする。ただし②または⑭のファイルを

$ PRMFL　　R*, W, S, <システム・マスタディレクトリ名>/<ファイル名2>

とする。

SAVEのサブパラメタI、Aの区別はLINKに対するJCLを作り出す時までフラグとして保存しておいて用いる。この報告書では、長くなるのでIの場合についてのみ述べることにする。

B－2　COBOL ステートメント

COBOLステートメントの場合もFORTRAN ステートメントに準じる。異なる点は

$　OPTION　FORTRAN　が
$　OPTION　COBOL　に　また、
$　FORTRAN　パラメタ　が
$　COBOL　パラメタ　に代わるだけである。

B－3　PL1ステートメント

PL1ステートメントの場合もFORTRANステートメントに準じる。異なる点は

$　OPTION　FORTRAN　が
$　OPTION　PL1　に　また、
$　FORTRAN　パラメタ　が
$　PL1　パラメタ　に代わるだけである。

C　LINK ステートメント

LINK ステートメントは単に

$LINK　…第1型

あるいはSAVE パラメタを付けて

$LINK　<実行形式プログラム名>, SAVE [({I/A})]

$EFILE　　<ファイル名>, {NEW / OLD} [{, DA / , MT, <ボリューム通番>}]　…第2型

のいずれかの形で記述し、必要があれば、

$USERLIB <オブジェクトライブラリ名> [, {DA / MT, <ボリューム通番>}]

を付け加える。前者を第1型、後者を第2型と呼ぶことにする。

(1) 第1型の場合基本的には次のようなACOSのJCLに変換する。

```
$   FILEDIT     ,OBJECT,UPDATE                                    ①
$   FILE        *R,O1R                                            ②
$   FILE        R*,O2S,4L                                         ③
$   DATA        *C,,COPY                                          ④
$   INCLUDE                                                       ⑤
$   OPTION      FORTRAN                                           ⑥
$   LIBRARY     L0                                                ⑦
$   COPY        ,,F01                                             ⑧
$   INCLUDE                                                       ⑨
   ($  OPTION   NOGO)                                             ⑩
$   EXECUTE                                                       ⑪

$   PRMFL       L0,R,S <システム・マスタディレクトリ名>/<オブジェクトライブラリ名>   ⑫

$   ENDEDIT     END                                               ⑬
$   ENDCOPY                                                       ⑭
$   EXECUTE                                                       ⑮
$   FILE        R*,O2R                                            ⑯
```

図5－4（C－1）

(イ) コンパイルの段階でオブジェクト・ファイルが指定されている場合には②が

$ PRMFL   *R,R,S,<システム・マスタディレクトリ名>/<ファイル名>

になる。また、コンパイラがCOBOLのときは

⑥が　$　OPTION　COBOL、ＰＬ１のときは

$　OPTION　PL1

となる。２種類以上のコンパイラが関与している時にはその数だけ対応するものを作り出すことになる。

(ロ) ⑦と⑫はＵＳＥＲＬＩＢで指定したライブラリに関する記述である。ＵＳＥＲＬＩＢの数だけ⑦のＬＩＢＲＡＲＹのオペランドにＬｉを必要ならばカンマで区切ってジェネレートする。このときオペランドはＬｉとしｉは０～９の数である。（ＡＣＯＳではオブジェクト ライブラリは１０個まで指定できる。）このＬｉとオブジェクトライブラリ名から⑫のＰＲＭＦＬを作り出す。

(ハ) コンパイルの段階でカードから直接入力した際には⑧をジェネレートする。プログラム名は必らずしもＦ０１ではなく、ＦＯＲＴＲＡＮでもコンパイルするジョブステップが多数あればＦｎ（ｎは２桁の整数）となることもあり、ＣＯＢＯＬならばＣｎ、ＰＬ１ならばＰｎとなる。ファイルからの入力のときは⑧の代わりに

$　MODIFY　,,N.J.CL

となる。

(ニ) ⑩はＬＩＮＫのジョブステップの次にもし$ＥＸＥＣＵＴＥが指定されず、従って実行が行なわれないときにジェネレートする。

(2) 第２型の場合には、ＥＦＩＬＥステートメントで指定するファイルがＮＥＷであれば次のＪＣＬをジェネレートする。

```
$　FILSYS
USERID　　<システム・マスタ / ディレクトリ名>$<パスワード>
FCREAT　　<システム・マスタ / ディレクトリ名>/<ファイル名>,LINKS/4,12/,
　　MODE/RAND/
```

そのあと次のようなＡＣＯＳ用ＪＣＬを作り出す。

```
$   FILEDIT     ,OBJECT                                          ①
$   FILE        *R,O1R                                           ②
$   FILE        R*,O2S,4L                                        ③
$   DATA        *C,,COPY                                         ④
$   INCLUDE                                                      ⑤
$   SYSLD       CATALOG=<実行形式プログラム名>
$   LOWLOAD
$   OPTION      FORTRAN                                          ⑥
$   LIBRARY     L0                                               ⑦
$   COPY        ,,F01                                            ⑧
$   INCLUDE                                                      ⑨
$   EXECUTE                                                      ⑪
$   PRMFL       L0,R,S,<システム・マスタディレクトリ名>/<オブジェクトライブラリ名>  ⑫

$   ENDLD
$   ENDEDIT     END                                              ⑬
$   ENDCOPY                                                      ⑭

$   SYSEDIT                                                      ⑯
$   PRMFL       Q*,W,R,<システム・マスタディレクトリ名>/<実行形式プログラムライブラリ名>  ⑰
$   FILE        R*,O2R                                           ⑱
```

図５－４（Ｃ－２）

このときは実行形式プログラム・ライブラリを作成しているのでＣ－１は少し形が異なる。但し前項(1)の(イ)(ロ)(ハ)については同じである。(ニ)の次に$EXECUTEのない場合、実行のＪＣＬがＣ－１とＣ－２では異なるのでOPTION NOGOはジェネレートしない。

Ｄ　EXEC ステートメント

EXEC ステートメントは

$EXEC　〔<ファイル名>　(<プログラム名>)〕〔,PARAM='　'〕

のように記す。

(1) 通常前段階でＬＩＮＫを行ない実行形式プログラムファイルを作成していない場合は、ファイル名およびプログラム名は指定しない。この場合およびもし指定していても、プログラムが実行形式プログラムファイルに入っていない場合には、このステートメントに対するＡＣＯＳのＪＣＬはここではジェネレートしない。

（ACOSではこのような時リンクのステップと明らかな区別がないため）。ただしFORTRAN PL.1で読み込むカードデックがあれば

```
$  DATA        I*
     ｛ データ
```

をジェネレートする。COBOLの場合にはNJCLで標準の読み込みおよびプリントアウトファイルを

```
$FILE      <ファイル名P>, , PRINT
$FILE      <ファイル名C>, , CREADER
```

と定義しているはずであるので、これに対して

```
$   PRINT      <ファイル名P>
$   DATA       <ファイル名C>
      ｛ データ
```

を作り出す。

(2) 実行形式プログラムファイルに入っているプログラムに対するEXECステートメントの場合には、次のようなACOS用JCLをジェネレートする。

```
$    PROGRAM    <実行形式プログラム名>
$    PRMFL      **, R, R, <システム・マスタ / ディレクトリ名>/<ファイル名>
```

このあとの標準的なプリントアウト及びカードデックの読み込みについては(1)に同じである。

(3) この他実行時にはFILEステートメントによっていろいろな入出力ファイルを定義するわけである。このFILEステートメントは各ホストコンピュータの機能を損なわないように多種類のパラメタを定めているため、その各々の場合について変換結果のJCLも異なるので、ここでは説明を省略する。

E－1 EDITSステートメント

EDITSステートメントは一般形としては

```
$EDITS [<ファイル名1> [, { DA / MT, <ボリューム通番> }]]
$NFILE [<ファイル名2> [, { NEW / OLD }] [, { DA / MT, <ボリューム通番> }]]]
```

このあとにEDITSに対するコントロールカードを含む更新ソースプログラムデックを置く場合を第1型とし、

$TFILE <ファイル名3>[,{DA / MT,<ボリューム通番>}]

を用いる場合を第2型とする。

(1) 先ず、いずれの場合にも、$NFILEにファイル1とは異なるファイル名2が表記されており、これにNEWというパラメタがある場合には、このファイルをシステムに登録するジョブステップ(アクティビティ)をジェネレートする。

大記憶ファイルの場合にはこのジョブステップは次のようになる。

```
$        FILSYS
USERID   <システム・マスタディレクトリ名>$<パスワード>

FCREAT   <システム・マスタディレクトリ名>/<ファイル名2>,
                                    LINKS/3,10/
```

ここでシステムマスタディレクトリ名とパスワードは、NJCLではPASSWORDステートメントで指定されているものである。

(2) 第1型でファイル名1がない場合には、新しくファイル2にソースプログラムライブラリを作成することを示している。したがって、この場合EDITSに対するコントロールカードは、プログラム単位のINSERT以外にはあり得ない。すなわち、NJCLで次の形のときである。

```
$EDITS
$NFILE    <ファイル名2>[,{NEW / OLD}][,{DA / MT,<ボリューム通番>}]

/$INSERT  <プログラム名1>,{F / C / P}
          ソースプログラム1
          .
          .
          .
```

このときACOSでは次のようなJCLをジェネレートする。

```
$   FILEDIT   SOURCE,NOBJECT,INITIALIZE

$   PRMFL     K*,W,S,<システム・マスタディレクトリ名>/<ファイル名2>

$   DATA      *C,,COPY
$   COMPILE   OFF
$   INCLUDE   SOURCE
```

```
$       FORTRAN                          (第73カラムより)プログラム名1

        FORTRAN ソースプログラム1

$       GMAP                             (第73カラムより)N.J.CL      ①
        SYMDEF      N.J.CL
N.J.CL  NULL
        END
$       COMPILE     ON
$       ENDEDIT
$       ENDCOPY
```

(注) ここで $FORTRAN は COBOL の場合には

```
$   COBOL     EALERT      (第73カラムより)プログラム名
```

PL1プログラムの場合は

```
$   PL1                   (第73カラムより)プログラム名
```

とする。以下 FORTRAN の場合を例にとって説明を行なう。

また、ここで①のACOSアセンブラGMAPによるダミーのプログラムN.J.CL を必らずジェネレートする。この理由は、ACOSでFILEDITを用いてファイルを作成する時、一番最後のプログラム名を認識していないと困ることが生じるのに、標準ジョブ制御言語システムではユーザーが作成したファイルの最後のプログラム名を認識し得ないので、やむをえずこのようなダミーを作ることにした。

(3) 第1型でファイル名1が指定された場合は、何らかの更新作業が行なわれる場合である。説明をわかりやすくするため、NJCLで次の形が指定されているものとする。

```
$EDITS      SFILE1, DA          古いソースライブラリ
$NFILE      SFILE2, OLD, DA     新しい      〃
/$・・・
                .
                .
                .
```

この部分に対してはACOSでは次のようなJCLをジェネレートする。

```
$   FILEDIT
$   PRMFL     M*, R, S, <システム・マスタ ディレクトリ名>/SFILE1
$   PRMFL     K*, W, S, <システム・マスタ ディレクトリ名>/SFILE2
```

```
$   DATA        *C,,COPY
$   COMPILE     OFF
        この間には更新に関する情報が入る
$   COPY        SOURCE,,N.J.CL
$   COMPILE     ON
$   ENDEDIT
$   ENDCOPY
```

(4) 次に上の(3)の例で、プログラムの置き換えが行なわれる場合を考える。すなわち次のようなときである。

/$REPLACE　　プログラム名1, { F / C / P }

ここで今 /$REPLACE PROG1, F であるとすると、ACOS に関しては次のようなJCLに変換する。

```
$   MODIFY      SOURCE,,PROG1
$   FORTRAN                     (73カラムより)PROG1
      { ソースプログラム (PROG1)
```

(5) プログラムの削除を行なう場合も同様にNJCLでは、

/$DELETE　　プログラム名2, { F / C / P }

であるがここで /$DELETE PROG2, C が与えられたとするとACOS については次のようなJCLをジェネレートする。

```
$   MODIFY      SOURCE,,PROG2
$   DELETE      SOURCE,,PROG2
```

(6) プログラムを追加する場合、NJCLでは次のように書く。

/$INSERT　　プログラム名3, { F / C / P }

ここで、/$INSERT PROG3, P であるとすると、ACOSでは次のようなJCLを出すことになる。

```
$   MODIFY      SOURCE,,N.J.CL
$   INCLUDE     SOURCE
$   FORTRAN                     (73カラムより)PROG3
      { ソースプログラム (PROG3)
```

N.J.CLはファイルの最後のプログラムとして標準ジョブ制御システムがジェネレートするものである。

(7) さらに、特定のプログラム内でステートメントの置き換えを行なう場合にはNJCL では次のように書くことになっている。

```
/$REPLACE    プログラム名(m[, n]), { F / C / P }
```

これ以降では、カード番号をm、nで表わすものとする。この時にはACOSでは次のようなJCLに変換する。

```
$   MODIFY      SOURCE, , プログラム名
$   FORTRAN          (73カラムより)プログラム名
$   UPDATE
$   ALTER       m, n
      〉置き換わるべきソースステートメント
```

㋑

NJCL では、置き換えるはずのカードが1枚のときには、mしか指定しないが、ACOSに変換するときには、このような場合 $ALTER m, m とする。

㋑の3つのステートメントは同一プログラム名に対してステートメントの更新が行なわれていないときのみ、ジェネレートする。このことはステートメントの追加、削除の場合についても同様である。

(8) ステートメントの削除の場合、NJCL では次のように記す。

```
/$DELETE     プログラム名(m[, n]), { F / C / P }
```

この時、ACOSでは次のようになる。

```
$   MODIFY      SOURCE, , プログラム名
$   FORTRAN          (73カラムより)プログラム名
$   UPDATE
$   ALTER       m, n
```

㋑

(9) ステートメントの追加の場合、NJCLでは次のように記す。

```
/$INSERT     プログラム名(m), { F / C / P }
```

この時、ACOSでは次のように変換する。

```
$   MODIFY      SOURCE, , プログラム名
$   FORTRAN          (73カラムより)プログラム名
$   UPDATE
```

㋑

```
$    ALTER        m'
      〈 追加すべきソースステートメント
```

ここでm'はm＋1の整数である。ACOS ではカード番号は、原則として コンパイルリスト上のものを用いることになっており、ソースカードの番号のインクリメント幅は1である。他の機種では、ほとんど、指定したステートメントの後に挿入することになっているが、ACOSのみ指定したステートメントの前に挿入するのでこのようになる。

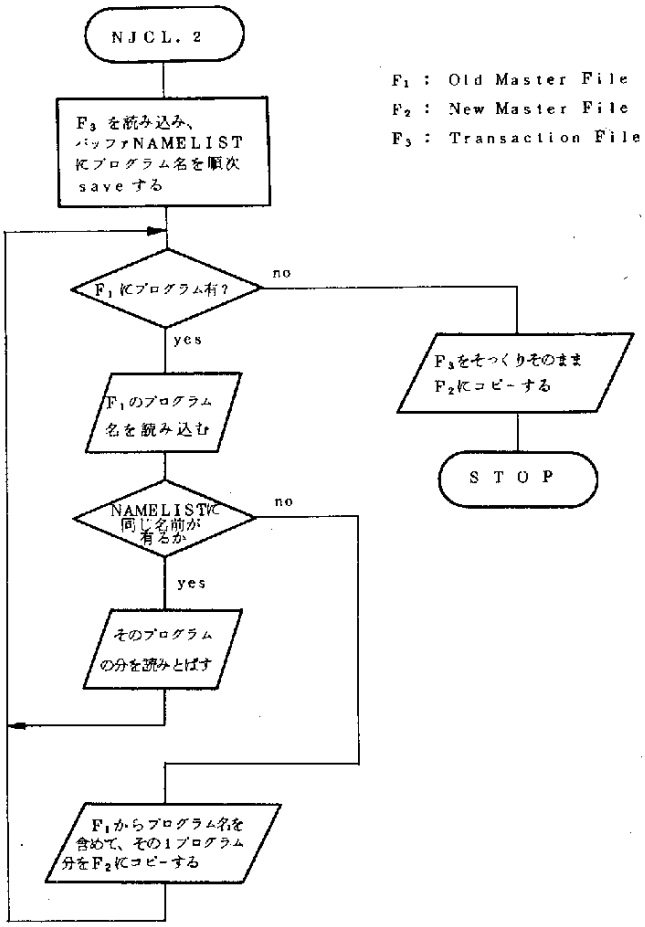
(10) 次に第2型すなわち、TFILEの指定のある場合について述べることにする。

ACOSの場合ソースおよびオブジェクトプログラムファイルはシークェンシャル編成なので、基になるマスタファイルとTFILEで指定されるトランザクションファイルに同名のプログラムがあるかどうかは、一旦どちらかのファイルを読んでプログラム名をSaveしてからでないとわからない。このあと、もう一方のファイルと比べて片方のファイルにないもののみ、新ファイルにコピーし、次に残りのファイルをコピーする手順となる。このような機能を持つACOSのユーティリティプログラムがないので、標準ジョブ制御言語では独自のRun time Routineを作成する。従って第2型のときにはこのルーチンを呼び出すことになる。

```
$    PROGRAM    NJCL.2                                          ①
$    PRMFL      **, R, R, DRP/NJCL  EDIT02                      ②
$    PRMFL      MF, R, S, <システム・マスタディレクトリ名>/<ファイル名1>   ③
$    PRMFL      TF, R, S, <    〃    >/<ファイル名3>            ④
$    PRMFL      NF, R, S, <    〃    >/<ファイル名2>            ⑤
```

上のようなJCLをジェネレートする。①はランタイムルーチン名の定義、②はその存在するファイル、③～⑤はこのジョブステップの入出力ファイル3個である。

このランタイムルーチンのフローチャートは次のとおりである。



E－2　ETITO ステートメント

ETITOステートメントは一般形としては、

$$\$\mathrm{EDITO}\quad \langle \text{ファイル名1} \rangle \left[, \left\{ \begin{matrix} \underline{\mathrm{DA}} \\ \mathrm{MT}, \langle \text{ボリューム通番} \rangle \end{matrix} \right\} \right] \left] \right]$$

$$\$\mathrm{NFILE}\quad \left[\langle \text{ファイル名} \rangle \left[, \left\{ \begin{matrix} \mathrm{NEW} \\ \underline{\mathrm{OLD}} \end{matrix} \right\} \left[, \left\{ \begin{matrix} \underline{\mathrm{DA}} \\ \mathrm{MT}, \langle \text{ボリューム通番} \rangle \end{matrix} \right\} \right] \right] \right]$$

このあとにEDITO に対して削除するプログラム名を与えるコントロールカード

$$/\$ \left\{ \begin{matrix} \mathrm{DELETE} \\ \mathrm{D} \end{matrix} \right\} \quad \langle \text{プログラム名} \rangle$$

のカードを続ける場合を第１型とし、

$$\$\mathrm{TFILE}\quad \langle \text{ファイル名3} \rangle \left[, \left\{ \begin{matrix} \underline{\mathrm{DA}} \\ \mathrm{MT}, \langle \text{ボリューム通番} \rangle \end{matrix} \right\} \right]$$

を用いる場合を第２型とする。

(1) 先ず、いずれの場合にも、$NFILEにNEWというパラメタがある場合には、このファイルをシステムに登録するジョブステップをジェネレートする。

```
$       FILSYS
USERID          <システム・マスタディレクトリ名>$<パスワード>

FCREAT          <システム・マスタディレクトリ名>/<ファイル名2>,
                LINKS/3,10/
```

(2) 第1型の場合には次のようなACOSのJCLを作り出す。

```
$   FILEDIT     NOSOURCE,OBJECT,UPDATE
$   PRMFL       *R,W,S,<システム・マスタディレクトリ名>/<ファイル名1>
$   PRMFL       R*,R,S,<システム・マスタディレクトリ名>/<ファイル名2>
$   DATA        *C,,COPY
$   MODIFY      ,,<プログラム名1>
$   DELETE      ,,<プログラム名1>
$   MODIFY      ,,<プログラム名2>
$   DELETE      ,,<プログラム名2>
      .
      .
      .
$   ENDEDIT
$   ENDCOPY
```

ここでプログラム1，2は消去するプログラム名である。

(3) 次に第2型すなわち、TFILEの指定のある場合について述べる。

ACOSの場合、EDITSのところで説明したとおり、オブジェクトファイルもシークェンシャル編成で自動的に同名のプログラムを更新する機能を持たないので、EDITSにおけるものと同じような標準ジョブ制御言語用のRun time Routineを作成し、これを呼び出すJCLを以下のようにジェネレートする。

```
$   PROGRAM     NJCL.3
$   PRMFL       **,R,R,DRP/NJCLEDIT03
$   PRMFL       MF,R,S,<システム・マスタディレクトリ名>/<ファイル名1>
```

```
$    PRMFL      TF, R, S, <システム・マスタディレクトリ名>/<ファイル名3>

$    PRMFL      NF, R, S, <   〃   >/<   〃   2>
```

このランタイムルーチンはオブジェクトプログラムファイルの取り扱いという点を除いて、EDITSにおけるNJCL.2に同じであるのでその項を参照されたい。

E－3　EDITDステートメント

カタログファイルとして登録してあるファイル，（ACOSではパーマネントファイル）を抹消するためのステートメントがEDITDであり

```
$EDITD          <ファイル名>
```

のように記す。これに対しては次のようなACOSのJCLをジェネレートする。

```
$    FILSYS
FR              <システム・マスタディレクトリ名>/<ファイル名>
```

F　レベル1のステートメントの変換

レベル1の各ステートメントはそのパターンが限定されているのでACOSでは比較的簡単なジョブ制御言語に変換できる。以下にその変換の方法を示すことにする。

F－1　レベル1の$$FORTRANステートメント

レベル1の$$FORTRANステートメントに対しては次のようなACOSジョブ制御ステートメントをジェネレートする。

```
$    OPTION     FORTRAN
$    FORTRAN
```

FORTRAN　ソースプログラム

```
$    EXECUTE
$    LIMITS     ・・・   もしJOBステートメントで
                         パラメタによる指定があれば
$    DATA       I*                                  } ①
```

データ

データがなければ①以下はジェネレートしない。またJOBステートメントにNORUNパラメタが指定してあれば$EXECUTE以下はジェネレートしない。

F－2　レベル1の$$COBOLステートメント

レベル1の$$COBOLステートメントに対しては、次のようなACOSジョブ制御ステート

メントをジェネレートする。但し $$PRINT ステートメントのオペランドはPTとする。

```
$   OPTION      COBOL
$   COBOL       EALERT
```

コボルソースプログラム （FILE-CONTROLとしてPTをリスティング用）に Assignするものとする

```
$   EXECUTE
($  LIMITS        ...       )
$   PRINT       PT
$   DATA        I*
```

JOB ステートメントにパラメタによる指定のある時

①

データ

データがなければ①以下はジェネレートしない。またJOB ステートメントにNORUN パラメタがなければ $EXECUTE 以下はジェネレートしない。

F－3　レベル1の $$PL1 ステートメント

レベル1の $$PL1 ステートメントに対しては、次のような ACOS ジョブ制御メントをジェネレートする。

```
$   OPTION      PL1
$   PL1
```

PL1　ソースプログラム

```
$   EXECUTE
($  LIMITS        ...     )
$   DATA        I*
```

JOB ステートメントにパラメタによる指定のある時

①

データ

データがなければ①以下はジェネレートしない。またJOB ステートメントに NORUN パラメタが指定してあれば $EXECUTE 以下はジェネレートしない。

5.4 JCLのサーベイ

この項では、IBM(OS/VS1)、FACOM(OSⅣ/F4)、UNIVAC(EXEC－8もしくは1100OS)、日電(ACOS－6)、MELCOM(BPM)それぞれのジョブ制御言語について、その通念と機能の面において、分類と比較を行なう。

まず、各社システムの通念とするところの、ジョブ制御言語に表われた部分を挙げ、次に各々について、極く簡単に分類を行ない、最後にやや細かく、機能的な面について比較をする。

5.4.1 通　念

A．OS/VSI

一つのジョブは、すべて何組かのジョブ・ステップの並びにより構成される、という概念が、形式的にも明瞭な形に統一されている。また各ジョブ・ステップは、一つのEXEC文と必要なだけのデータ定義文から構成される。

ファイルの動的割付けをせず、必要なものすべてを陽に記述する事により、所要ファイルおよび、ファシリティが入力ジョブ・ストリームから明らかとなるため、システム資源の割付けが効率よく行なわれる。

データ定義文のUNITの指定法の例にも見るとおり、ユーザは特殊目的のために、システムのハードウェア構成を意識してジョブを定義する事もでき、また必要がなければ、それを無視してジョブを構成する事もできる。この概念はその他のシステム資源管理にも及んでいる。

B．OSⅣ/F4

MシリーズはIBMのOS/VS系オペレーティング・システムとの相互互換性を狙ったものであり、概念的、機能的に際立った差異は認められない。

C．EXEC－8(1100OS)

ジョブ制御言語の機能として、大体において、必要十分なものが、各社を通じて最も単純な形で記述できる。ジョブの構成は、制御ステートメント、サブコマンド、データの単純な並びであり、IBM系のような、明瞭な形でのジョブ・ステップという構造を持たない。

データ・ファイルとプログラム・ファイルという属性をシステムが認識し、更に、プログラム・ファイル中の、プログラム・エレメントについて、ソース、リロケータブル、アブソリュート(実行可能)型式の区別もシステムが認識する。またデータ・ファイルは一種のデータしか収容できないが、プログラム・ファイルは複数エレメントを収容する。

ソース・プログラムとオブジェクト・プログラムの修正が、コンパイル・オプションにより同時に行なわれる。

バッチとタイム・シェアリングの両形式の処理を通じて、同一の制御言語を使用可能である。

FUR/PURと呼ばれるファイル・プログラム操作用のユティリティ・プログラムが制御ステートメントレベルで簡単に使用できる。

D．ACOS－6

情報の保護を重要視し、ファイルには、時限パスワードのような厳重なアクセス資格を設定する事が可能である。また、複数プログラムから同一ファイルへの同時アクセス時の読出し／書込み許可も、細かく指定する事が可能である。更に、ファイル、メインメモリの内容を使用後に自動的に消却することを指定することも可能である。

プログラムについては、ソース・プログラムとオブジェクト・プログラムはソース・オブジェクト・ライブラリにより同理に管理されているため、修正時ソース、オブジェクト・プログラム間の一致性が保たれている。

各ジョブ・ステップごとに、メモリ容量、プリント行数等の資源に対する制限が付けられるため、ユーザの誤りにより、資源を浪費する事が少なくなる。また、テープ、ディスクなども各ジョブ・ステップ単位に指定し、そのジョブ・ステップが終了すると特に指定の無い限り消滅するため、資源の有効利用を計りやすい。

ジョブの構成は、概念的に、ジョブがジョブ・ステップの並びであり、各ジョブ・ステップで使用するデータの定義はステップ毎になされる点で、ＩＢＭ系と近いが記法的には、それ程の統一性は無い。

ファイルの割付けについては、動的に行なうものと静的に行なうものの混合的な形態をとる。

E．BPM

制御ステートメントの形式、及びジョブの構成法はUNIVACと近いものである。またファイルの割り付けもUNIVACと同様、動的に行なわれるため、必要なものすべてを陽に指定する必要はない。

ソース入力、オブジェクト出力などの際の入出力デバイスの変更が比較的容易である。

ユーザ・プログラムを標準のシステム・プロセッサとして、システムに登録、使用することができる。

リアル・タイム処理のための機能が他社のオペレーティング・システムと比較して、制御言語、レベルで容易に使用できる。

5.4.2 分　類

各社のジョブ制御言語を分類するにあたり、それらが必ずしも単一の視点で分類できないところから、一応ジョブの構成、ジョブ・ステップの定義法、データの定義法といった視点で分類し、更にそれぞれの中の小項目についての分類を試みた。この項目においては、ジョブ制御言語ならびにそれを用いて定義したジョブに、最も根本的な差異を与える、ジョブの構成上の分類を行ない、その他については、ここでは後の〔Ⅲ〕で比較を行なうに際して着目すべき、視点を示す。ジョブの構成法上の分類は、そのジョブ制御言語自身の全般的特徴を表わし、それ以外のものは、各オペレーティング・システムの機能上の比較になると考えられる。

A．ジョブ構成

これについては、調査対象の五社のシステムについては、以下の二種類に大別可能である。

1）実行するシステムプログラム、あるいは、ユーザ・プログラムの指定とそれに付属するデータの定義を不可分のジョブ・ステップとして捕え、このジョブ・ステップの並びをジョブとするもの、この場合、プログラム指定の文とデータ定義の文の間には時間的な順序関係は無く、ジョブ・ステップ同士にのみ順序関係がある。

2）ファイルのアサインなどのデータ定義文と、プログラム起動の文は別々の制御文として捕えられ、それらはアクティビティと呼ばれる。この場合、データ定義文が現われると、その時点で資源の割付けが行なわれ、プログラム起動文ですぐに実行が始まることから、それらの文の間には時間的順序関係が存在する。

ジョブ全体としては、コマンド的な制御文の並びとなる。

上記の1）に属するものとしては

・ＩＢＭ　ＯＳ／ＶＳ１

・富士通　ＯＳⅣ／Ｆ４

・日　電　ＡＣＯＳ－６

が挙げられ、2）に属するものとしては

・UNIVAC　1100OS

・MELCOM　BPM

が挙げられる。

この分類は、オペレーティング・システムが、バッチジョブに対して、ファイル（データ）の動的割付けを許しているかいないか、という観点からの分類とほぼ一致する。1）は動的でないものの群であり、ジョブ作成の際コーディング量は多くなり、ＴＳＳ環境との言語の互換性の問題もあるが、概念的には、プログラムとデータを一体として捕える。ジョブステップが明確となる。これに対して2）は動的なものの群で、上記と逆の性質を持つものと見られる。

実際にコーディングしたジョブを見ると、1）に属するものでは、プログラムの実行指定が先に来てその後ろにデータ定義が続き、2）に属するものはその逆の順となる。

B．ジョブ・ステップの構成

ジョブ・ステップ、あるいはアクティビティの構成については、上でも少し触れたが、その機能について、更に次の二項目を考慮する。

ⅰ）システム・プロセッサとユーザ・プログラムの実行に同じステートメント形式を用いるか。

ⅱ）どのようなものがジョブ・ステップ、あるいはアクティビティとなるか。

C．データの定義法（ファイルに関して）

これについては次の各項目について比較できる。

ⅰ）アサインの方法

1）デバイスによる違い

2）有効範囲

3）ファイルの寿命による違い

ii）ファイルの概念

iii）内容の属性

iv）内容の取扱い

D．その他の機能

上記の各項目には属さないが、各オペレーティング・システムの提供する機能により、次の各項目について比較する。

i）オプション及びサブコマンドの指定法

ii）各リソースに対する制限

iii）カタログ・プロシージャ

iv）条件分岐

5.4.3 比　較

5.4.2で述べたように、ここでは、各社のオペレーティング・システムが提供する機能について、以降で挙げた各項目に従って分類と比較を行なう。

A．ジョブ・ステップの構成

i）システム・プログラムとユーザ・プログラムの呼出し法

a群：すべて同じ制御ステートメントを用いるもの

・ＯＳ／ＶＳ１

・ＯＳⅣ／Ｆ４

これらはEXEC文上のPGMパラメタで指定するプログラム名を変える形に統一され、ジョブ・ステップは必ずこのEXEC文で始まる。

b群：異なるステートメントを用いるもの

・１１００ＯＳ

・ＢＰＭ

・ＡＣＯＳ－６

ii）ジョブ・ステップの単位

a群：プログラム実行指定文と、そのプログラムで使用するファイル及びデータ定義を含めてジョブステップとするもの

・ＯＳ／ＶＳ１

・ＯＳⅣ／Ｆ４

・ＡＣＯＳ－６

b群：a群の意味でのジョブ・ステップは存在せず、データ定義はプログラム（システム・プロセッサを含む）の起動とは別個のアクティビティであるもの

・1100OS

・BPM

これらについて、プログラム起動とデータ定義は別のアクティビティではあるが、意味上の関係から、データ定義はプログラム起動より前で行なわれなければならないという制約がある。

B．データの定義法

i）アサインの方法

1）デバイスによる制御文の違い

a群：同じステートメントを用いるもの

・OS／VS1

・OSⅣ／F4

・1100OS

・BPM

これらは各々の定義上に指定するパラメタのうち、デバイス指定とそれに関するものを変更する事によってテープ、ディスクなどを使い分けられる。

b群：異なったステートメントを用いるもの

・ACOS－6

テープ装置には$ TAPE、ディスク装置については$ FILEというステートメントを用いる。

2）有効範囲

ここで言う有効範囲とは、そのファイル定義の有効範囲であり、ファイルそのものの有効範囲ではない。

a群：定義した時点以降、特に削除しない限りジョブ全体に有効なもの。

・1100OS

・BPM

b群：基本的には定義のあったジョブ・ステップ内でのみ有効なもの

・ACOS－6

次のジョブ・ステップに引き渡すことも可能で、ディスポジション・コードでセーブの指定によりなされる。

c群：ステップ内のみ有効なものとジョブ全体に有効なものの二つの機能を持つもの

・OS／VS1

・OSⅣ／F4

通常はステップ内の有効範囲であり、以下のステップへはディスポジションでパスを指定して引き渡すが、特にジョブ全体に有効としたい時には、JOBLIB機能を使う。

3）ファイルの寿命による違い

ここでは、一時的ファイルと保存ファイルの取り扱いについて触れる。

a群：それぞれの種類のファイルの作成、使用に同じステートメントを用いるもの。

・ＯＳ／ＶＳ１

・ＯＳⅣ／Ｆ４

・１１００ＯＳ

種類や使用法の区別は、オプションないしパラメタで指定するが、ＩＢＭ、ＦＡＣＯＭについては、一時ファイルはそのデータ・セット名（＆＆付き）で区別する。

b群：異なったステートメントを用いるもの。

・ＡＣＯＳ－６

一時ファイルについてはＳ　ＦＩＬＥ、保存ファイルの作成には＄　ＦＩＬＳＹＳ、その使用には＄　ＰＲＭＦＬという別々のステートメントを用いる。

ⅱ）ファイルの概念

a群：ファイルのためのスペースの確保と内容の書き込みが論理的に別であるもの。

・１１００ＯＳ

・ＢＰＭ

ファイルのアサインとプログラムの起動が別個に存在する事から当然である。

b群：内容とファイルの枠組が同時に行なわれるもの。

・ＯＳ／ＶＳ１

・ＯＳⅣ／Ｆ４

・ＡＣＯＳ－６

ⅲ）内容の属性

a群：システムが、ファイルの内容を認識し、その使用に関与するもの。

・１１００ＯＳ

データ・ファイルとプログラム・ファイルはシステムによって区別され、更にプログラム・ファイルの内容である。各エレメントについて、ソース、リロケータブル、アブソリュートの属性が設定されるため、同名三種のエレメントも作成可能である。

b群：システムが、ファイルの内容の属性には関与しないもの

・ＯＳ／ＶＳ１

・ＯＳⅣ／Ｆ４

・ＡＣＯＳ－６

ただし、ＯＳ／ＶＳ１およびＯＳⅣ／Ｆ４では、システムが保有する特定のものについては属性を持っている。カタログ・プロシージャ用ライブラリや、ＳＹＳ１．ＬＩＮＫＬＩＢと呼ばれるシステム・プログラム用ライブラリなどである。

iv）ファイル中でのデータの扱い

a群：順次編成のファイルしかないもの

・1100OS

・BPM

・ACOS－6

データ・ファイルの内容は一体のものでなければならない。

b群：区分編成可能なもの

・OS／VS1

・OSⅣ／F4

ファイル作成の際に、その内容を区分データ・セットのメンバとして登録すればよい。

v）ファイル中のプログラムの扱い

a群：ソース、オブジェクト、実行形式のプログラムを混合して書き込めるもの

・1100OS

・OS／VS1

・OSⅣ／F4

IBM、FACOMについては、実行形式プログラムは必ず区分データ・セットのメンバでなければならないが、ソースとオブジェクトプログラムは、順次データ・セットでもよい。

UNIVACでは、プログラムは各形式ともエレメントとして作成されるので、順編成は無い。

b群：形式の異なるプログラムを混合してファイルに書き込むことは不可能なもの

・ACOS－6

・BPM

これらについては、ソースとオブジェクトプログラムは順編成、実行形式プログラムは区分編成ファイルとなる。

C．その他の機能

i）オプションの指定法

a群：順序、場所の制約の無いもの

・BPM

b群：場所の制約のみあぬもの

・1100OS

オプションは、プロセッサ名指定の後にコンマを打って、一項目英字一字で指定し、順序は任意

c群：一部順序の制約のあるもの

・OS／VS1

・ＯＳⅣ／Ｆ４

・ＡＣＯＳ－６

パラメタのうち、特定のものは、それを記述する順序が定められており、他の大部分のものについては、順序、場所は任意である。

ii）サブコマンドの指定法

プログラムによっては、その動作のための特定の情報をサブコマンドによって受けるが、このコマンドの与え方を比較する。

a群：コマンドのためのデータ・セットを設けるもの

・ＯＳ／ＶＳ１

・ＯＳⅣ／Ｆ４

ユティリティ・プログラムなどへのコマンドは//SYSIN DDとして与える。

b群：プロセッサ呼び出し文の後に直接付けるもの

・１１００ＯＳ

コンパイラへのソース修正情報、リンケージエディタ（ＭＡＰプロセッサ）へのマップ情報などである。

c群：他の制御ステートメントと同じもの

・ＡＣＯＳ－６

ある特別なコマンドは直接付けるものもある。

d群：サブコマンドを用いないもの

・ＢＰＭ

ＢＰＭでは、すべてオプションで記述する。

iii）リソースの制限

実行時間、出力量などについての許容量の与え方について比較する。

a群：ジョブ・ステップ毎に指定するもの

・ＡＣＯＳ－６

b群：ジョブ全体に対して指定するもの

・１１００ＯＳ

・ＢＰＭ

c群：a、b群双方の機能をもつもの

・ＯＳ／ＶＳ１

・ＯＳⅣ／Ｆ４

ただし、出力プリント量はジョブ全体には指定できず、記憶量指定は両方を指定すると、EXEC文上の指定は無視される。

iv）カタログ・プロシージャ

ここでは、カタログ・プロシージャの機能に各社システムが、どのようなものを与えている

か比較する。

a群：カタログ・プロシージャ内で指定したパラメタの修正、セットだけが可能なもの

・ＢＰＭ

b群：上の他に、ステートメントの追加、修飾が可能なもの

・ＯＳ／ＶＳ１

・ＯＳⅣ／Ｆ４

・ＡＣＯＳ－６

ＯＳ／ＶＳ１およびＯＳⅣ／Ｆ４では、各パラメタの省略時値の設定、カタログ・プロシージャ中での特定パラメタの無視などの機能も備えている。

c群：カタログ・プロシージャの機能をもたないもの

・１１００ＯＳ

特にカタログ・プロシージャの機能は無いが、記号エレメントもしくはデータ・ファイル上に制御ステートメントの列を作成して、それを@ADD機能を用いて制御ストリーム中に挿入する事はできる。但し、パラメタなどの機能は無い。

カタログ・プロシージャについて、各社のオペレーティング・システムが提供している機能は、その制御言語を用いて直接コーディングした場合のコーディング量の多いシステムほど豊富なものとなっており、通常のルーティン・ワークにおける見掛け上のコーディング量を低減させている。

Ⅴ）条件分岐

あるジョブ・ステップや、それ以降のジョブの実行を制御する方法

a群：分岐命令を用いて、適当なレーベルの付いた文へ分岐するもの

・１１００ＯＳ

・ＡＣＯＳ－６

b群：ジョブ・ステップの実行の迂回条件を記述し、分岐命令はもたないもの

・ＯＳ／ＶＳ１

・ＯＳⅣ／Ｆ４

これらは、他に、ＪＯＢ文上で指定することにより、ジョブの実行を中止する条件の記述することができる。

6. データ共有システムの基本構想

第1章において、ユーザの視点からコンピュータネットワーク技術を議論した。このなかで、現在のネットワーク技術は、一般ユーザが容易にコンピュータネットワーク上のリソースをアクセス出来るユーザインタフェース（REX、NAM、RITA等）の開発〔ROSE76a〕に重点を置いていることをみた。このようなユーザインタフェース開発の背景として次の2つの問題を指摘出来る。

1）コンピュータネットワークが種々のリソースを持つことによる異種性問題。
2）コンピュータネットワーク上のどこに、必要なリソースが存在するかを明らかにするリソースの分散問題。

現在のコンピュータネットワーク技術は、必要なリソースの所在を知り、そのリソースのアクセス手続きを知っているならば、ネットワークを通してリソースを利用出来るようにしている。しかし、一般ユーザにとって、多くのリソースの異種性情報と分散情報とを知ることは困難な問題である。ARPANET におけるNAMをはじめとするユーザ支援システムは、異種性問題と分散問題とを解決し、一般ユーザへ対して容易なリソース利用を可能とさせようとするものとみることができる。分散型リソース共有問題において、異種性と分散との2つの問題を分離して検討する必要がある。

リソースとして、プログラム、データ、ハードウェアがあげられている〔ROBE70〕が、データが今後最も重要なリソースとなることを第Ⅰ部第3章において述べた。政府省庁、地方自治体等の行政情報、大学、研究所における学術情報等のデータが、コンピュータネットワークを通して共有される重要なリソースと考えられる。データの他のリソースに対する特徴は、大量であることと、時間とともに変化することである。この点は、データベースをネットワークを用いて共有することの1つの根拠となっている。

このようなデータリソースを検討するために、コンピュータネットワーク問題とは別個に、個々のコンピュータにおけるデータベース管理システム（DBMS）を第2章において検討した。これまで、DBMS の分類として代表的な3つのデータモデル（関係、網型、階層型）が用いられてきた。しかし、現在の DBMS は、網型モデルを基礎にして関係的な問合せ言語を備えているといった具合に、種々のインターフェースを提供するようになっている。この傾向は、ANSI／SPARC レポート〔ANSI75、TSIC76〕に述べられる様に、データベースを、効率に係る面（内部スキーマ）、ユーザのアプリケーションに依存する面（外部スキーマ）、データの意味に関する面（概念スキーマ）の3層への分離を示している。コンピュータの変化とユーザの要求の多様性とに対して、一意な概念スキーマを設定し、外部 ↔ 概念 ↔ 内部という変換を行なうことによって対処しようとするものである。このことは、ユーザのアプリケーションに適した種々のデータモデルと言語、そして個々のコンピュータに最も適した内部スキーマとの共存を許す共存アーキテクチャ（coexistence architecture）〔NIJS76b〕を、一意な概念スキーマを設定することによって実現することでもある。分散型デー

タベースを議論する場合、個々のDBMSのどの階層間で通信をおこなうかは重要である。この意味で、三層化概念に基づいたDBMSの分類検討は今後十分に行なう必要がある。

第3章では、コンピュータネットワークを用いたデータベース間の結合問題、即ち分散型データベース問題を実際のシステム例に基づいて検討した。分散型データベースへは、2つのアプローチがあると考える。1つは、ある特殊なアプリケーション専用のものである。これは、CONIT〔MARC76、77〕EURONET〔CEC77a、b、c〕に代表される実用システムである。これらのシステムは、"文献検索"というよく定義されたアプリケーションを対象としている。ユーザへ対しては共通インタフェースを与え、システムはこの共通インタフェースと各データベースのインタフェースとの変換を行なう。各データベースの大半は、ファイルシステムであることも特徴の1つである。各データベースが共通の論理構造を持ち、文献検索の様によく定義されているアプリケーションへ対してはこの様なアプローチは有効なものである。

これに対して、最近開発されているINGRES等のアプローチがある。前者のアプローチと異なり、このアプローチはアプリケーションへ対して汎用である。このため、各アプリケーションとのデータベースの意味を記述するために各データベースはDBMSを用いている。このアプローチの例としてINGRES〔STON77〕、SDD-1〔ROTH77b〕、LADDER〔SAGA77〕、POLYPHEME〔ADIB77〕について検討した。これらのシステムは、次の点について検討された。

1）ネットワーク上へ分散されたデータへ対する統一データモデルと言語とは何か。
2）データが分散していることにともなう分散問題と、分散問題のために必要な機能（分散機能）は何か。
3）異種データベースを持つことによる異種性問題と、異種性問題のために必要な機能（異種性機能）は何か。

この章の議論において、分散型データベース全体における統一データモデルと非手続的問合せ言語の必要性と、分散問題と異種性問題との分離の必要性とを指摘した。

上記の議論で明らかなように、コンピュータネットワーク上のリソースに関する異種と分散に関する情報の管理は重要となっている。第4章では、これらの情報を管理するNCC（Network Clearing Center）について検討した。NCCは、コンピュータネットワーク上のリソースアクセスのために必要となる情報 ―リソースの所在（分散）情報と個々のリソースの異種性情報 ― とを保持し、ユーザの要求に対して必要な情報を提供する機能をもっている。NCCの議論においては、分散型データベースにおいて重要となるディレクトリの構成、分散形態についてまとめた。

上記の検討をもとにして、本章では異種コンピュータネットワークにおけるデータ共有システムの基本構想を述べる。6.1節では、リソース一般（即ち、プログラム、データ、ハードウェア）の場合の異種コンピュータネットワークにおけるリソース統合問題を論じる。6.2節では、リソースとしてデータを考えた場合のデータ共有問題を論じる。この節では、汎用システムとして必要となる全体概念

（統一）モデルと言語、分散問題と分散機能、異種性問題と異種性機能について論じる。

6.1 異種コンピュータネットワークにおけるリソース統合問題

1970年代にはいってから、ARPA〔ROBE70、77〕、CYCLADES〔ZIMM77〕等のコンピュータネットワークの成果に基づいた分散型リソース共有システム（Distributed Resource Sharing System)は、情報処理の分野に大きな影響を与えてきている。コンピュータネットワーク上に分散した種々のリソースが地理的距離に関わりなく利用出来る様になった一方で、単一コンピュータシステムでは生じなかった次のような問題が発生している。

第一の問題は、リソース（プログラム、データ、ハードウェア）の異種性である。コンピュータネットワークが種々のホストコンピュータを含んでいるために、ユーザはホストごとに異なった種々のアクセス手続き（ログオン／オフ、コマンド等）を習得せねばならない。このような種々のアクセス手続きを知ることは困難である。

第2の問題は、リソースがコンピュータネットワーク上に分散していることによる分散問題である。数百のリソースを持つ分散型リソース共有システムにおいて、ユーザにとって必要なリソースがどのサイトのどのホストにあるかをユーザが知ることは困難な問題である。

上記の2点は、一般ユーザがシステムへ参加することを困難にしている。

分散型リソース共有システムにおいて、上記したリソースの異種性と分散問題がユーザへ視えないこと、即ち不可視性〔ROTH77c〕は、基本的概念である。このためには、異種性と分散問題との不可視性の概念に基づいた種々のリソースのネットワーク全体での共通アクセスが必要となる。一般ユーザに対して、分散型リソース共有システムをあたかも一つのシステムかのようにみせることである。よって、システム全体に対する統一モデルとアクセス言語とが必要となる。統一モデルには、リソースの異種性も分散問題も表われない。統一モデルにおけるリソースとは、実際のリソースのクラスまたグループの共通概念である。

上記の問題点を解決するための我々のアプローチを以下に示す。ARPAネットワークにおけるユーザ支援システム（user assistant）を第1章において議論した。ユーザ支援システムは、異種性と分散の問題を解決しようとするものである。この意味で、第1に既存のユーザ支援システムの機能の分類を試みる。この分類のなかで、異種と分散の不可視性を解決する非手続的リソース統合システムが今後の課題であることを述べる。第2に、非手続的リソース統合における必要な機能とアーキテクチャを述べる。第3に、非手続的リソース統合処理におけるリソース記述を論じる。最後にリソースの処理表現を論じる。

6.1.1 ユーザ支援システムの分類

ユーザ支援システムは、分散型リソース共有システム上のリソースへの容易なアクセス手段を提提するシステムである。この様なユーザ支援システムとして、NAM〔ROSE76b〕、REX〔BENO74〕、RITA〔ANDE76、77〕等が開発されている〔PYKE74、ROSE76a〕

ユーザ支援システムは、分散型リソース共有問題の異種性問題と分散問題との解決を目指すものであると考えられる〔ＴＡＫＩ７８〕。

既存のユーザ支援システムの機能を、次の２点について分類する。

1）ユーザへ提供されるサービスレベル

2）ユーザが、ユーザ支援システムを通して同時にアクセス出来るホストコンピュータの数

1）のサービスレベルとは、ユーザの要求が手続的言語又は非手続的言語によって記述されるかの問題である。我々は前者を手続的レベル、後者を非手続的レベルと呼ぶ。後者は前者へ対して高水準である。

2）は、ユーザ支援システムによって単一又は複数ホストをアクセス出来るかである。我々は前者を"単一リソースアクセス"後者を"リソース統合"と呼ぶ。以上の分類を図６－１に示す。

手続的リソースアクセス（procedural resource access）とは、ユーザは同時には１台のホストしか使えず、かつ手続的言語によって必要なサービスを指示するものである。例として、ＮＡＭ〔ＲＯＳＥ７５、７６〕がこれに相当する。

非手続的リソースアクセス（non-procedural resource access）とは、ユーザは同時には１台のホストしか使えないが、非手続的言語によって必要なサービスを指示するものである。例えば、ＲＥＸ〔ＢＥＮＯ７４、７５〕、ＲＩＴＡ〔ＡＮＤＥ７５、７６ａ、ｂ〕がこれに対応する。

手続的リソース統合では手続的言語によって複数ホストを利用できる。例として、ＣＯＮＩＴ〔ＭＡＲＣ７７〕、ＲＳＥＸＥＣと〔ＴＨＯＭ７３〕を上げることが出来る。リソース統合（resource integration）を、ユーザ支援システムを用いてユーザが複数ホストコンピュータのリソースをアクセス出来ることと定義する。

以上の分類より、非手続的言語によるリソース統合、即ち非手続的リソース統合（non-procedural resource integration）（図６－１の斜線部分）が未分野であることが解かる。非手続的リソース統合において、リソースの異種性と分散問題を意識することなく、ユーザは高水準な手続的言語によって自分の要求を述べることが出来る。非手続的リソース統合では、ユーザに対する異種性と分散問題の不可視性とともに、システム全体へ対する統一モデルの提供も必要である。

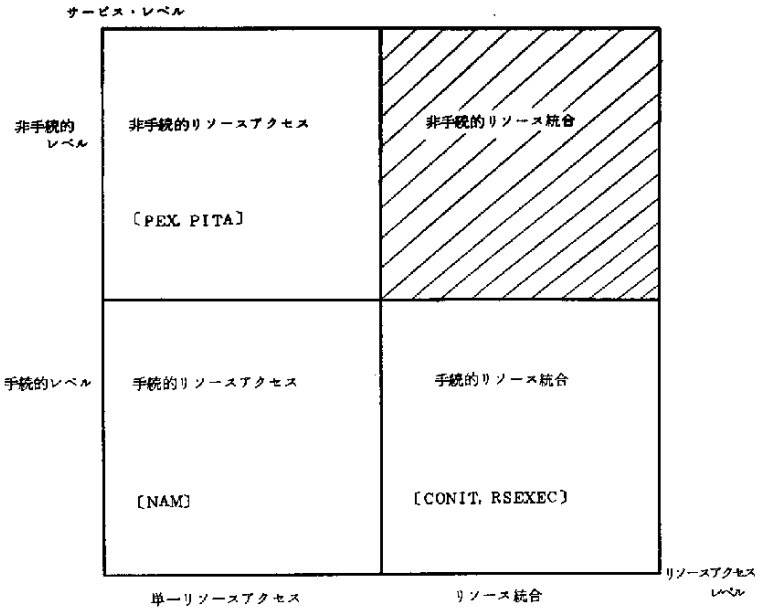


図６－１　ユーザ支援システムの分類

6.1.2　アーキテクチャ

非手続的リソース統合において次の３点が重要となる。

１）統一モデルと統一アクセス言語

２）分散問題の不可視性

３）異種性問題の不可視性

これらを実現するためユーザ支援システムは、統合プロセッサ（CP：Coordinate Processor）、ローカル情報プロセッサ（ＬＩＰ：Local Information Processor）、ネットワーク案内センタ（NCC：Network Clearing Center）の３つの論理要素から構成される。

ＣＰは、ユーザの要求を解析しＮＣＣからのリソースの分散情報に基づいて、必要なリソースを持ったＬＩＰへのアクセスを行ない、ＬＩＰからの応答結果の統合を行なう。

ＣＰの機能は次の様である。

１）ユーザの非手続的要求を、システム内での処理手順を示す手続的な標準内部形式に変換する。

２）必要なリソースの存在するホストコンピュータを決定する。ＮＣＣの保有する分散情報に基づいてこの決定は行なわれる。対応するリソースが複数存在する場合は、最適なものを決定する。

３）ＬＩＰへの処理依頼を標準形式〔ネットワークジョブ制御言語（ＮＪＣＬ）〕（第５章参照）

で行なう。

4）ＬＩＰの実行時における同期管理を行なう。

5）ＬＩＰからの処理結果に基づいて、ユーザの要求に合うような結果の合成を行なう。

ＣＰは、主にユーザへの容易なインタフェースであるとともに、分散問題の不可視性をユーザへ提供する。

ＬＩＰは、他のリソースの分散についての情報とは関係なく、ホストコンピュータ内のリソースの管理を行なう。ＬＩＰは各ホストごとに存在する。ＬＩＰは、ＣＰからのリソースアクセス要求をローカルホストで処理出来る形式への変換を行なう。即ち、異種性問題の解決をＬＩＰにおいておこなう。変換に必要となる異種性情報は、ＣＰと同様にＮＣＣから得る。

ＮＣＣは、コンピュータネットワーク上のリソースに関する分散情報と異種性情報との２種類の情報を保有している。ＮＣＣは、ディレクトリに相当するものである。ＣＰとＬＩＰの各々において必要となる分散情報と異種性情報は、このＮＣＣ内に保持されている。ネットワーク上でのＮＣＣの実際の構成としては種々の分散形態が考えられる〔 4.3 節を参照のこと 〕。分散形態は、分散情報と異種性情報の各々へ対して考える必要がある。

ＣＰ、ＬＩＰ、ＮＣＣの３つの論理要素の関係を図６－２に示す。

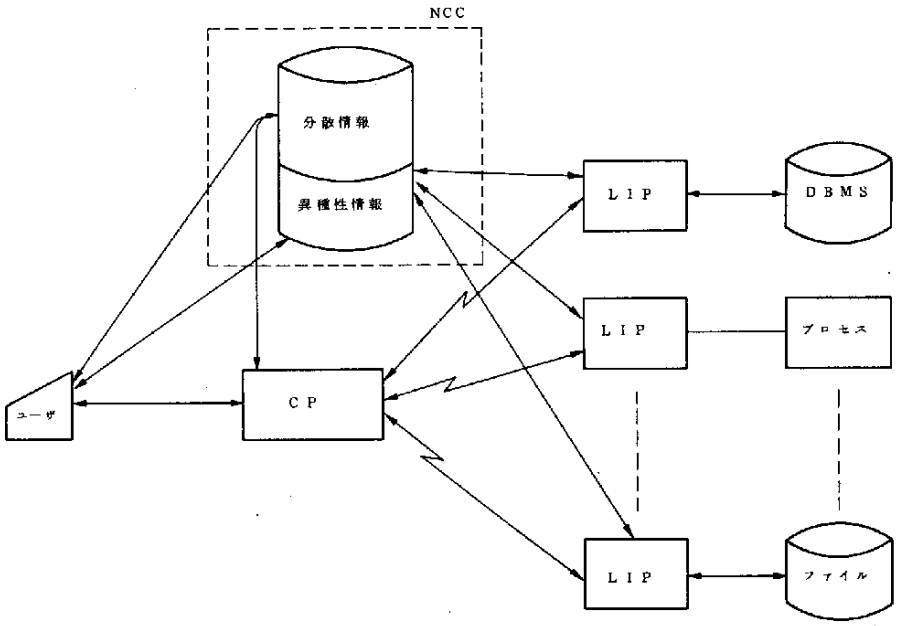


図６－２　ユーザ支援システムの論理構成

6.1.3　リソース記述

我々は先に、コンピュータネットワーク全体に対する統一的なリソースモデルが必要であることを述べた。この統一モデルは、データリソースに対しては、ＰＯＬＹＰＨＥＭＥ〔ＡＤＩＢ７６、７７〕

における様な全体概念スキーマ〔Adiba は全体視点(global view)と呼んでいる〕である。統一モデルにおいては、リソースの分散(所在)情報と異種性情報とを視ることは出来ない。統一モデルにおけるリソースは、リソースのあるクラス又は集合の共通概念である。この共通概念はある名前によって表わされる。この共通概念を表わす名前は、REX〔BENO74〕における一般リソース名(generic resource name)へ対応する。このような概念名によってリソースを表わしたものを概念リソース記述(conceptual resource description)と呼ぶ。例えば、FORTRANは、FORTRAN−H、FORTRAN−IV等のコンパイラ、又はJIS 7000 FORTRAN、JIS 5000 FORTRAN といった言語のクラスへ対する共通概念名とみなせる。よって"FORTRAN"は概念リソース記述である。統一モデルにおいて我々は、リソースが実際どこに存在し、どの様に使用するかに関わりなく"FORTRAN"という概念リソース記述によって必要なリソースを表現出来る。

更に、概念リソース記述へ対して、リソース間の関係を加えたものを処理表現(processing expression)と呼ぶ。概念リソースモデルとは、概念リソース記述と処理表現とから成る。概念リソースモデルは、システム全体へ対する統一的なモデルである。ユーザは、このモデルを用いて、リソースの分散問題と異種性問題とを意識する必要なくリソースをアクセス出来る。

概念リソース記述へ対して、実際のリソースの記述とは、それの属するリソースクラス又はグループの共通概念に個々の固有情報を付加したものである。例えば、JIS 3000 のFORTRANを概念リソース記述とした場合、JIS7000 の実際のリソースは、JIS3000 へ対する拡張機能を"FORTRAN"に付加したものである。この様に実際のリソースの詳細かつ厳密な記述を、リソース属性記述(resource attribute description)と呼ぶ。即ち、リソース属性記述とは、分散と異種性問題を含まない概念リソース記述へ対して、分散と異種性情報とを導入したものである。

一方、概念リソース記述は、リソース属性記述から異種性情報と分散情報を除去したものである。この除去の過程を概念化(conceptualization)と呼ぶ。逆に概念リソース記述へ、分散と異種性情報の導入過程を、非概念化(deconceptualization)と呼ぶ。

概念リソース記述とリソース属性記述との関係を図6−3に示す。概念リソースモデル構成のアプローチは次の2つの段階からなる。

1) ローカルサイト単位でのリソースの属性記述から概念記述への変換。異種性情報は、この変換過程で除去され、NCC情報として、NCCにおいて保有される。サイト単位のリソースの概念記述を、ローカル概念リソースモデル(local conceptual resource model)と呼ぶ。
2) ローカルな概念記述をもとに、分散情報を除去した全体概念リソース記述を生成する。分散情報は、1)と同じくNCCにおいて保有される。この全体の記述を、全体概念リソースモデル(global conceptual resource model)と呼ぶ。

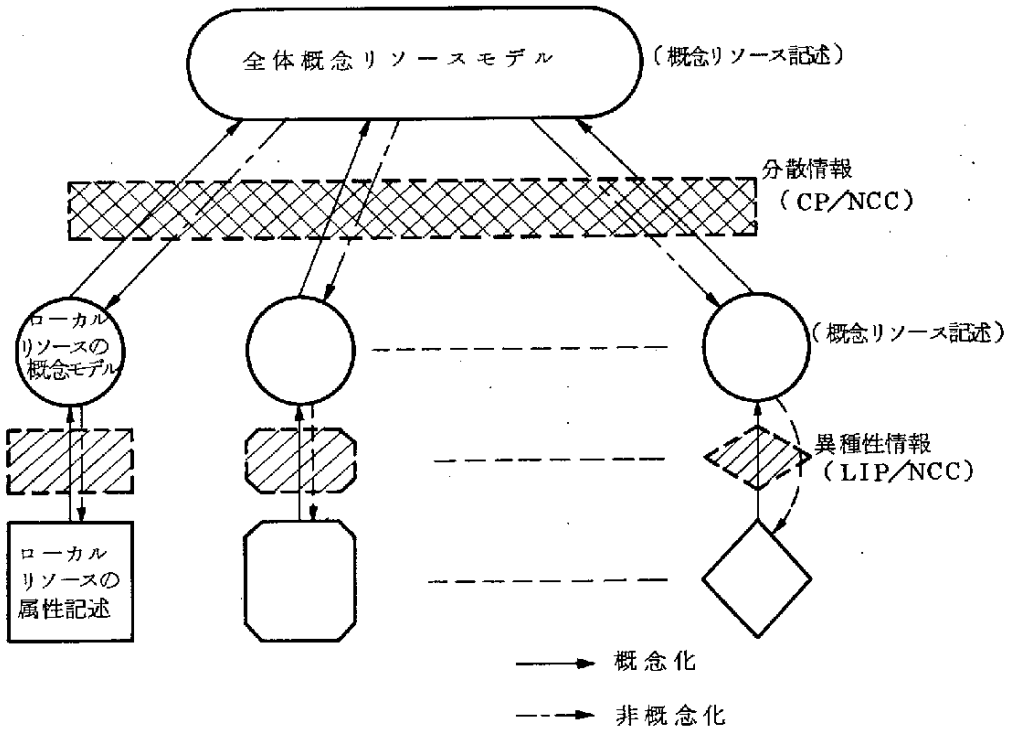


図 6 － 3　概念記述と属性記述との関係

6.1.4 処 理 表 現

ユーザは全体概念リソースモデルを用いて、自分の必要な仕事を記述する。この記述は、リソースの分散情報と異種性情報とに関係なく行なわれる。この様なリソースの処理記述（処理表現と呼ぶ）を分割して、各リソースサイトへの処理要求を生成せねばならない。

リソースの処理表現として次のものを考える。

1）ユーザ言語
2）全体処理表現(global processing expression)
3）ローカル処理表現(local processing expression)
4）ローカル実行可能表現(local executable expression)

ユーザ言語とは、非手続的な言語である。非手続的言語を、手続的なものに変換する必要がある。全体処理表現は、概念リソース記述と、概念リソース間の手続的な論理的関係性とを示したものである。この表現には、分散及び異種性情報はあらわれない。

次に、全体処理表現を、実際のリソースの存在するサイト単位の処理への分割が必要になる。ＮＣＣに保有されている分散情報に基づいてこの分割は行なわれる。この様に各サイト単位に分割された処理表現をローカル処理表現と呼ぶ。ＣＰは、このローカル処理表現を用いて、各サイトのＬＩＰへ処理依頼する。各々のローカル処理表現は、実行の同期のために半順序化されている。

ＬＩＰは、システム全体で共通な形式を用いたローカル処理表現を各ローカルコンピュータ上で実行可能な一連の表現へ変換する。この場合、ＮＣＣの異種性情報を用いて変換が行なわれる。ローカル実行可能表現も半順化され、実行の同期をとれるようになっている。

これらの表現の関係を図６－４に示す。

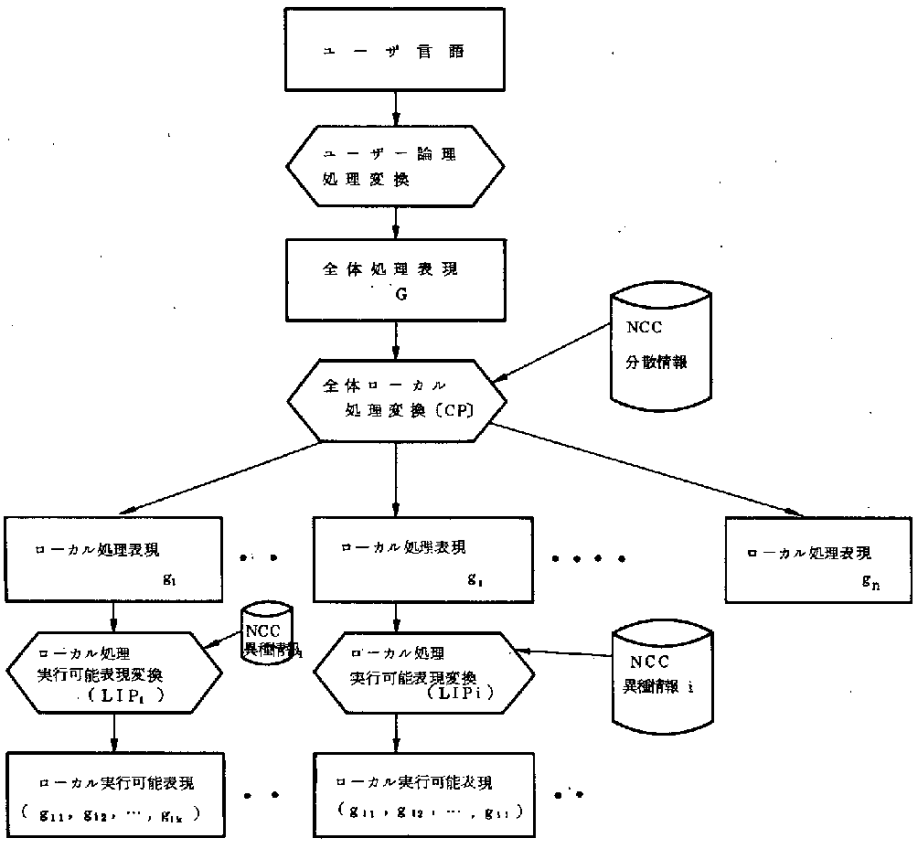


図６－４　処理表現変換

6.2　リソース統合問題におけるデータ共有問題－DBMS 統合への基本的視点

本節では、データリソースとしての DBMS の統合問題、即ち分散型データベース問題について考える。第３章において分散型データベースについての検討において、２つのアプローチがあることを述べた。１つのアプローチは、CONIT〔MARC 77〕と EURONET〔CEC 77 a、b、c〕に代表されるものである。これは、"文献検索"という限定されたアプリケーション専用のものであり、共通インタフェースから各データベースのインタフェースへの変換を行なわせるものである。

どの様にアクセスするかを視ることは出来ない。ローカル全体概念化は、システム全体の管理者（全体管理者）によってなされる。

ユーザのアプリケーションの多様化に対処するために、1つの全体概念スキーマへ対して、アプリケーションへ適したデータモデルと言語とを外部スキーマとして定めることも出来る。この外部スキーマの設定は、各アプリケーション管理者によってなされる。

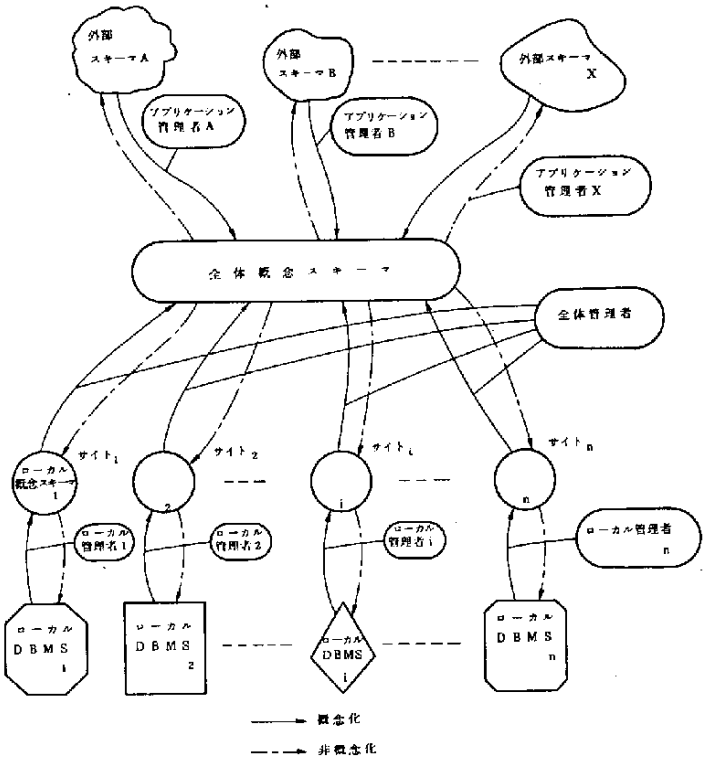


図 6－5　分散型データベースへの基本視点

概念化とは実世界からある企業体にとって必要な概念を抽出する過程である。このことは実世界から概念社会における不必要な事項の除去でもある。非概念化とは、除去された事項を概念社会へ導入することによって実世界を再現することである〔FALK 77a、b〕。ローカル全体概念化では分散情報が導入される。ローカル・ローカル概念化では異種性情報が除去され、非概念化では異種性情報が導入される。

まず、第2章における概念モデルの検討に基づいて概念モデルとしてのデータモデルについて考える。

次に、全体概念スキーマとローカル概念スキーマとの対応として分散問題を考える。分散問題とは、全体概念スキーマにおける要素間の関係性が、どの様に各サイトに分散しているかの問題である。

分散問題を実現するためのディレクトリ、一致性、問合せ分割、同時実行制御、冗長コピーの更

我々は文献検索の様な各データベースの論理構造又は意味記述が同一であるアプリケーションへ対しては、このアプローチは適していると考える。

他のアプローチは、INGRES〔STON77〕、SDD−1〔ROTH77b〕、LADDER〔SAGA77〕、POLYPHEME〔ADIB77〕にみられるもので、アプリケーションへ汎用のものである。アプリケーションごとに異なっているデータの意味を記述するために、これらのシステムの各サイトはDBMSである。この章では第2のアプローチ、即ち汎用アプローチに基づいて議論を行なう。

3.3節のDBMSから構成される分散型データベースシステムの議論において、我々は次の点が問題となることを述べた。

1）ユーザへ対する分散問題及び異種性問題を含まない統一的なデータモデルと高水準非手続的問合せ言語
2）データの分散問題と、このために必要となるディレクトリ等の分散機能。
3）DBMSの異種性問題と、このために必要となる異種性機能。
4）DBMS間の通信レベル

コンピュータネットワーク上に分散したデータベースをアクセスしようとする場合、必要なデータ項目がどこに存在し、どの様にアクセスするかを知らねばならないことは一般ユーザにとって大きな負担である。一般ユーザへの容易なアクセス手段の提供は、NAM〔ROSE76〕、REX〔BENO74〕、RITA〔ANDE76、77〕、NSW〔SCHA76〕等の目的と同様である。上記の問題点を解決する1つの方法は、ローカルDBMSの異種性とデータ項目の分散形態との情報を除外し、データベースの意味（semantics）を記述した統一的モデルをネットワーク全体へ設定することである。この統一モデルを全体概念モデル（global corceptual model）と呼ぶ。

我々の全体概念モデルの構築に対するアプローチは、異種性の除去と、分散情報の除去との2つの過程から成っている（図6－5）。第1の過程は、各データベースサイト内に格納されているデータの意味のみを記述したローカル概念スキーマの構成である。この過程をローカル・ローカル概念化と呼ぶ。各サイトは、そのDBMSのデータモデルによって、自分のサイトが保有しているデータの記述、即ちローカル内部スキーマを備えている。ローカル・ローカル概念化とは、各サイト単位で、そのサイト固有の異種性情報を除去し、ローカル概念スキーマを形成することである。ローカル概念スキーマは、各サイト共通のデータモデルによって記述される。ローカル・ローカル概念化は、各サイトのローカル管理者によってなされる。

第2の過程は、ローカル概念スキーマから、分散情報を除去して、全体概念スキーマを構成するものである。この過程をローカル・全体概念化と呼ぶ。全体概念スキーマは、ローカル概念スキーマと同じデータモデルによって記述される。よって、全体概念スキーマでは、ある企業体にとって関心のあるデータの意味のみが記述されている。このスキーマにおいて、データがどこに存在し、

新、ローカル問合せの半順序化等を議論する。

次に、ローカル概念スキーマとローカル内部スキーマとの変換として異種性問題を考える。異種性問題は、共通問合せから各ローカル DBMS 問合せ（request）への変換及び逆変換である。

問合せの変換、ローカル DBMS への問合せ同期、DBMS の起動等の必要な機能について論じる。

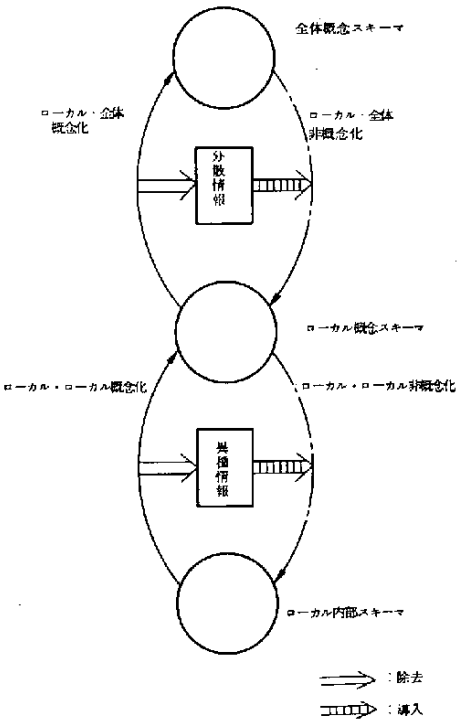


図６－６　概念化と非概念化

最後にDBMS間の通信レベルとしては、DBMS のアクセス方法等の内部機構と独立な親言語と問合せ言語レベルと考えることにする。

6.2.1　概念モデルと言語

概念モデルへ対して要求される事項は下記の点である。

1）データの意味の完全で形式的な記述

2）定常性

3）変換容易性

4）発展性

5）理解容易性

概念モデルにおいて第1に重要な点は、これがデータベースの意味の記述であることである。種々のローカルDBMSに格納されているデータの意味を、完全に形式的に記述出来ねばならない。種々のローカルDBMS間のモデル変換において、データベースの意味は概念モデルにおいて保存されていなければならない。

第2の点は、システムを構成するローカルDBMSの変化に対して、概念モデルは一定であることである。この様なモデルの定常性は、システム全体の長寿命性をもたらす。

第3の点は、概念モデルから、ローカルDBMSのデータモデルへの変換が容易であることである。ローカル概念モデルから、ローカル内部モデルの変換は、ローカル問合せの処理のたびに必要である。よって変換コストを少なくするために、変換の容易性は重要である。

第4の点は、概念モデル自身の発展性があることである。概念世界が変化しても、他のスキーマへの影響は可能な限り少なくなければならない。

最後の点は、概念モデルが一般ユーザにとって理解しやすい自然なものでなければならないことを示している。

現在までに、概念モデルとして種々のデータモデルが提案されてきている。これらの幾つかについては、2.4節において議論した。これらのモデルの概念に関する議論は、哲学的で難解なものである。Biller〔BILL77〕は、現在提案されているモデルが、同一の概念へ対してモデルごとに異なった用語を用いていることを指摘し、データモデルの静的側面の議論は止めてデータ操作等に関する動的な環境下での議論を行なうべきことを述べている。"もの"に対する人間の概念が形式化されない限り、どの概念が最適であるかを決定することは不可能である。

概念モデルの関係性として、大きく2つの流れがあると考える。1つはCodd〔CODD70〕によるn一項関係モデルに基づくものである。n一項関係モデルの欠点である意味問題を取扱うために、事象(entity)を関係の属性とは独立に扱かえ、関係の組が実世界の基本的事実を表わすように改良していくものである。Hallの縮退不能関係モデル(irreducible relational model)〔HALL76〕、Chenの事象・関係性モデル(entity－relationship model)〔CHEN76〕等である。Hallのモデルは、n項関係モデルへ事象の概念を導入し、n項関係モデルの欠点である事象の独立的な取扱いを可能とさせようとしている。Hallのモデルにおける関係は縮退不能関係と呼ばれる。これは、関係の組が実世界の基本的事実を表現出来るように、より小さな関係へ分割したものである。縮退不能関係は、単一の事実のみを表わす。Chenのモデルは、2.4.2項に詳述してある。

他は、Abrial〔ABRI74〕、Adiba〔ADIB76、77〕、Senko〔SENK76、77〕等による2項関係モデルである(2.4.2項参照)。このモデルにおいて、基本的事実は対象(object)間の2項関係によって表わされる。データの意味は、2つの対象間の両方向の関係(2項関係)によって記述される。モデルの基本要素を、他のモデルの要素と対応づけ、各要素間の関係を2項関係で記述することによって、種々のモデルを2項関係モデルによって記述できることは、Adiba〔ADIB76、77〕によって示されている。このモデルは、分散型データベースPOLYPHEME

〔ＡＤＩＢ７７〕において、ローカルデータベースと全体視点の記述のために用いられている。Senkoは、ＤＩＡＭ Ⅱ における情報論理層におけるデータモデルとして用いている〔ＳＥＮＫ７６ａ, ｂ〕。

２項関係モデルは、対象の個々の集合の対へ、実世界の意味に対応した２項関係性を導入している。このため、２項関係モデルは、モデルの変換性と記述性は高いものと思える。この意味で、概念モデルのよい候補であろうと思われる。一方で、各要素間に２項関係が存在しているために、再構成の困難さにともなう発展性の欠如が指摘されている〔ＦＡＬＫ７７ａ〕。

6.2.2 分 散 問 題

分散問題とは、全体概念モデルとローカル概念モデルとの対応である（図６－７）。即ち、全体概念モデルの各要素と要素間の関係と、ローカル概念モデルの要素と関係との対応である。

全体概念モデルにおける要素のローカル要素からの構成方法として次の基本形態がある。

１）ローカル要素の１部
２）ローカル要素の集合
３）ローカル要素は冗長

実際は、これらの基本形態の複合した形態である。

関係モデルを概念モデルとした場合を考えよう。関係モデルの要素として関係をあげることができるＳＤＤ－１では、関係は、２）と３）の形態をとり、ローカル要素はフラグメントと呼ばれる〔ＲＯＴＨ７７ｂ〕。関係のフラグメントへの分割形態は、具現（materialization）として表わ

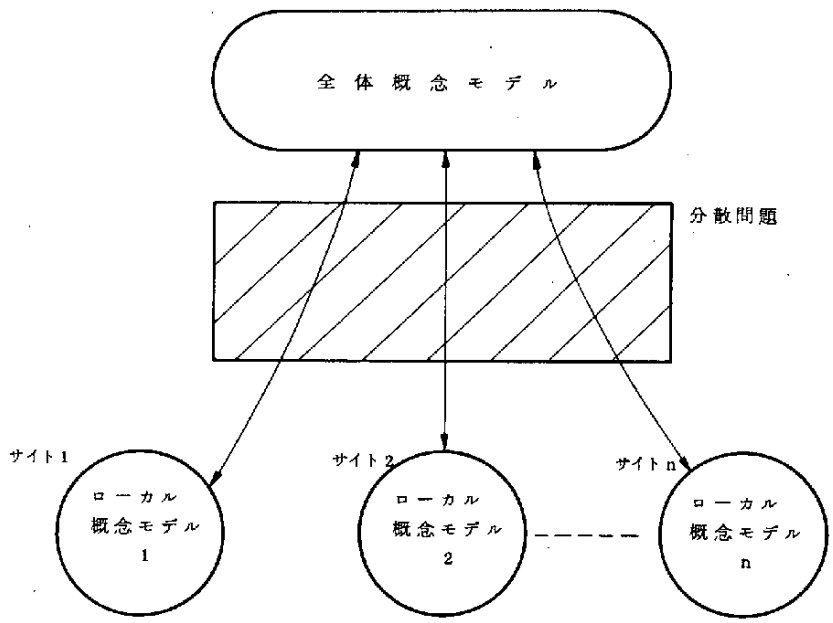


図６－７　全体概念モデルとローカル概念モデルとの対応（分散問題）

される。1）の形態は、全体概念モデルにおける関係があるローカル関係の射影又は制約である場合である。この3つの基本形態を図6－8にまとめる。

Adiba〔ADIB77〕のモデルの要素は、有形カテゴリ、抽象カテゴリ、アクセス関数の3つ〔ABRI74〕である。よって、分散の問題は、全体概念モデル（Adibaの全体視点へ対応する）における各カテゴリが、ローカル概念モデルのどの対象（object）から構成されるかである。3.3.4項において述べたように、フィルタ部によって1）、COMPOSE句によって2）、OCCUR句によって3）の分散形態を表現している。全体概念モデルにおける全体アクセス関数は、各ローカルアクセス関数によって記述されるプログラムである。

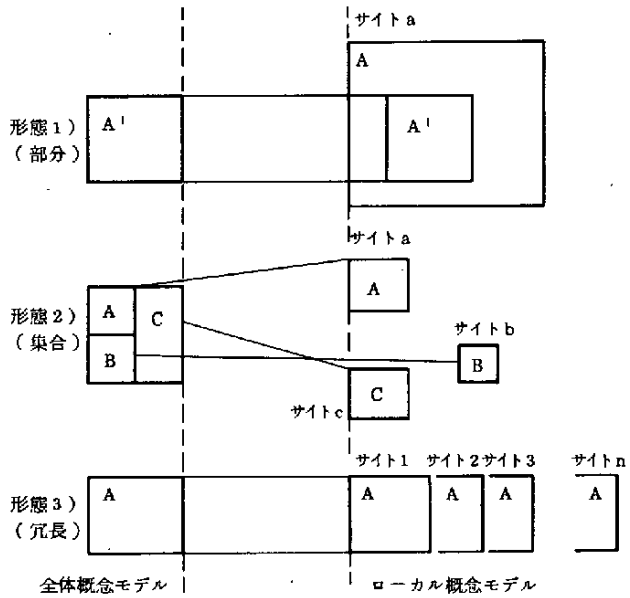


図6－8　全体関係とローカル関係との対応

分散問題は、概念モデルとして用いるデータモデルに大きく依存しているが、基本的分散形態（1）～3））を明らかに表わせることが必要である。各サイトのローカル概念モデルと全体概念モデルとが同一のデータモデルを用いていることは、分散問題を単純化する。図6－8のような3つの基本形態とこれらの組合せを表わしたものが全体概念モデルである。

6.2.3 分散機能

分散問題を解決するために必要となる機能として次のものがある。

1）ディレクトリ

2）全体問合せの分割

3）分割されたローカル問合せの半順序化と同期

4）一致性制御

5）同時実行性制御

ディレクトリは、全体概念モデルにおける要素がどのサイトに存在するかの情報、即ち分散情報を格納している。ディレクトリでは、その情報内容の種類と各々の種類についての分散形態を議論する。

全体問合せ（global query）は、分散問題と異種性問題を意識することなく、全体概念モデルにもとづいて述べられる。システムは、この全体問合せを、各サイト単位のローカル問合せ（local query）へ分割させねばならない。この様に分割されたローカル問合せは、次に半順序化（partial ordering）され、ローカル問合せの実行の同期を考える必要がある。問合せ分割と半順序化を次に議論する。

冗長コピーの存在は、高信頼性と高パフォーマンスをもたらすが、一方では更新に対する一致性の保持という困難な問題を生じさせる。このために一致性制御について議論する必要がある。

最後に、同時実行性の問題として、デッドロックの検出及び防止について考える。

A. ディレクトリ

ディレクトリは、その内容の種類ごとの分散について議論されねばならない。Rothnie〔ROTH77c〕は、ディレクトリに含まれるべき情報として、論理構造定義情報、物理構造定義情報、ファイル情報、会計情報の4種をあげている。分散型データベースにおける分散情報は、この4つへ加えられねばならない。この意味で上記4つの情報は、ローカル DBMS の異種性に関するものであるが、分散形態の議論は同一に行なう。ディレクトリの分散形態を議論する場合に重要な点は、ディレクトリ全体の分散ではなく、ディレクトリを構成する情報の種類ごとに分散形態を考えねばならないことである。ディレクトリ情報を階層化した分散問題は、Stonebraker〔STON77〕のシステムカタログの議論にみられる。ディレクトリ情報の種類によって、更新頻度、検索頻度、トラフィックの局所性、ディレクトリの蓄積コスト等が異なっている。分散形態がこの様な要因によって決まる以上ディレクトリの情報内容ごとに分散形態を考えるべきと考えている。

先に述べたように更新頻度、検索頻度、トラフィックの局所性、信頼性、蓄積コストが分散形態の主要な要因となる。

各ローカルサイトが自分自身のディレクトリだけを持つ局所型、システムのマスタディレクトリが1つのサイトだけにある集中型、全サイトがマスタディレクトリを持つ多重マスタ型の3つを典型的な型としてあげることができる。局所型、集中型及び多重マスタ型との選択はトラフィックの局所性についてなされる。トラフィックの局所性が高く、ネットワーク上にトラフィックがほとんど発生しないならば局所型が適している。集中型と多重マスタ型は、ネットワークを通したサイト間のトラフィックがかなり存在している場合適している。集中型と多重マスタ型の選択は、更新頻度と検索頻度とについてなされる。更新頻度が高い場合は集中型が適し、検索頻度が高い

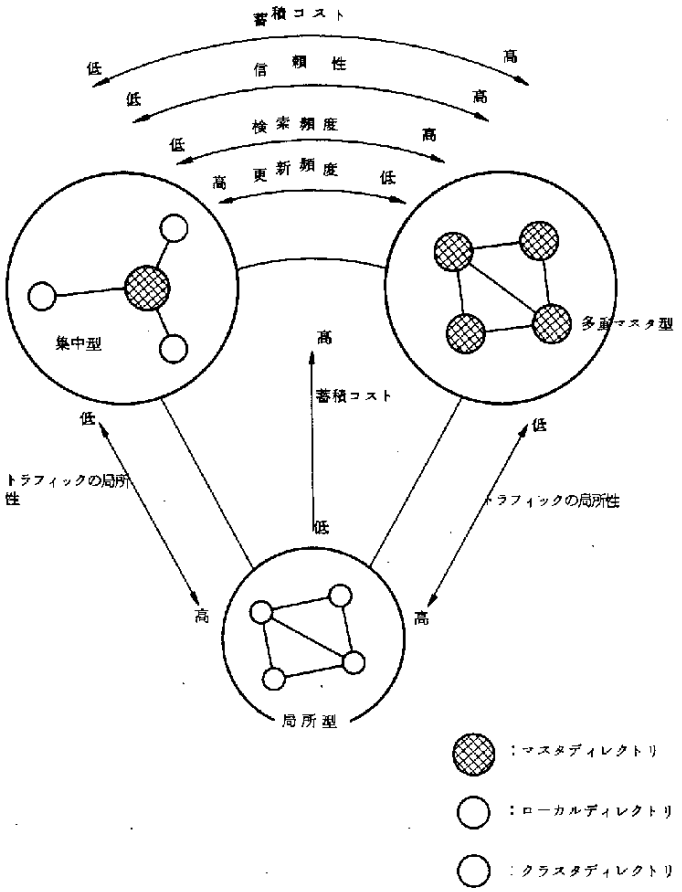


図６－９　分散形態の決定要因

場合は多重マスタ型が適している。他に考えるべき要因としてディレクトリの蓄積コストがある。システム全体での蓄積コストは、局所型、集中型、多重マスタ型の順に高いものとなる。

上記した点をまとめると図６－９のようになる。図６－９のように局所型、集中型、多重マスタ型を頂点とする三角形が形成される。この三角形内は、３つの型の複合形態の存在を示している。

ディレクトリの情報内容は大別して分散情報と異種性情報の２種がある。一般的に述べて分散情報は、多重マスタ型の形態、異種性情報は局所型の形態をとるだろう。

B. 全体問合せの分割

全体概念モデルに基づいた全体問合せ（Ｒ）を、各サイト単位のローカル問合せ（r_1, r_2…,r_n）に分割する必要がある（図６－１０）。この過程は、非手続的な全体問合せを、ローカル問合せからなる手続への変換でもある。

関係モデルにおける問合せ分割は、Wong〔WONG７６、７７〕において論じられている。これはサイト間にまたがった多変数ＱＵＥＬ問合せを、一連の１変数問合せへの最適な分割アルゴ

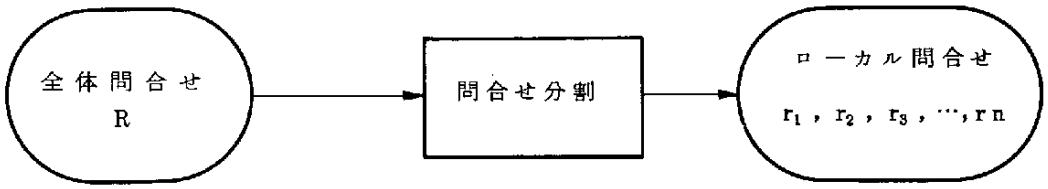


図６－１０. 問合せ分割過程

リズムに関する議論である。全体問合せのローカル問合せへの分割形態はいくつか考えられるが、通信コストを最少となるような最適化を行なっている。複数のサイトへまたがっている問合せの処理において、各サイトでの処理コストに対して、サイト間のデータ転送コストが支配的と考えられる〔WONG７７〕。この意味で通信コストを目的関数とした最適化は有効である。

C．ローカル問合せの半順序化と同期

全体問合せの各サイト単位のローカル問合せへの分割は、非手続的問合せからローカル問合せより成る一連の手順の生成である。このため問合せ分割によって生成されたローカル問合せ r_1, r_2, ..., r_n の半順序化が必要になる。例えば、ローカル問合せ r_2 は r_1 の完了を待ち、r_4 は r_2 と r_3 の完了を待ち、r_5 は r_4 の完了を待たねばならないとすると、問合せの実行順序は図６－１１のようになる。問合せ r_3 は r_1 と r_2 と同時並行して処理してもかまわないが、r_4 の処理以前に r_3 と r_2 の実行完了は同期されねばならない。

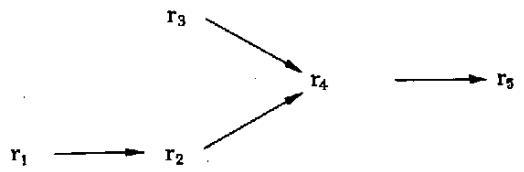


図６－１１　ローカル問合せの半順序化

問合せ分割において、ローカル問合せの半順序化を表現出来ることが重要である。

D．一致性制御

一致性問題として次のものがある〔STON７７〕。

1）データベースは１人の DBA をもつ。複数サイトにまたがる場合も、各サイトは異なった DBA をもてない。

2）２人の DBA が同一名のデータベースを同時に生成しようとした場合、これを解決せねばならない。INGRES では、データベースを生成しようとするメッセージに時間を印すことによってこの問題を解決している。

3）同一名のデータベースは、同一の構成をもたねばならない。即ち、関係モデルでは同一名の関係は、同一の属性構成をもたねばならない。

4）障害へ対して、システム全体で一致性をもったものへ回復せねばならない。ネットワーク内での回復は困難な問題である。個々のコマンドの処理が、各サイトでの処理を節とする完全な木構造をとっているならば、障害からの回復は容易に行なえる〔STON77〕。どれかの節での処理が障害を起こした場合、この節から根まで、各ノードの処理を撤退させることによって、全コマンド処理を撤退できる。

障害を起こしたホストの障害／回復は、ディレクトリ情報の更新を必要とする。ディレクトリが冗長である場合、ディレクトリの一致性を保たねばならない。

5）バックアップコピーも一致性が保障されねばならない。更新は主コピーに対してのみなされ、主コピーからこの更新をバックアップコピーへ対しても行ない全コピーの更新終了を待つ方式をINGRESはとっている。この方式は、更新のために大きなオーバヘッドと遅延とをもたらす。しかし、トラフィックの大半が検索トラフィックであることから、利用頻度高い地域に一致性の保たれた冗長コピーを設け、この地域のユーザはこのコピーをアクセス出来る。

図6－12にEllis〔ELLI77〕による多重コピーの更新プロトコルを示す。この図は、ペトリネット〔HOLT70、PETE77〕の改良である評価ネット(evaluation net)〔NUTT72〕に基づいている。このプロトコルは、多重コピーの更新において一致性を保つものである。

この図の円（ロケーションと呼ばれる）は、プロセス状態を、四角（メッセージロケーション）は到着メッセージの待ち行列を、水平線は転送を表わしている。INT REQは、ローカルユーザによる更新要求である。EXT REQは他のサイトからの外部的更新要求である。ACK+は正常応答、ACK-はNAKを表わし、ACK dは外部要求へのACKである。passiveは制御プロセスが何の内部更新要求（INT REQ）をサービスしていないことを示す。

D. 同時実行制御

同時実行制御は、デッドロックの防止と検出である。デッドロックは、ある処理に必要となる全部のリソースを要求し、獲得されたならばこれらのリソースをロックすることによって防止できる。全てのリソースを獲得できないならば、獲得されたリソースのロックは解かれ、ある時間後に再度リソースの要求を行なう。

デッドロックの検出は、ネットワーク全体のリソースについての"ロックアウト図（lock-out diagran）〔MARC76〕をつくり、循環部分を見つけることによって行なわれる。デッドロックの検出において重要な点は、ネットワーク全体のリソース情報を集める必要があることである。このために、集中型の中央制御センタをもつことは有利である。

デッドロックが検出されるならば、どれかのプロセスを犠牲として撤退（backout）させる

ことによってデッドロック状態を解除する。

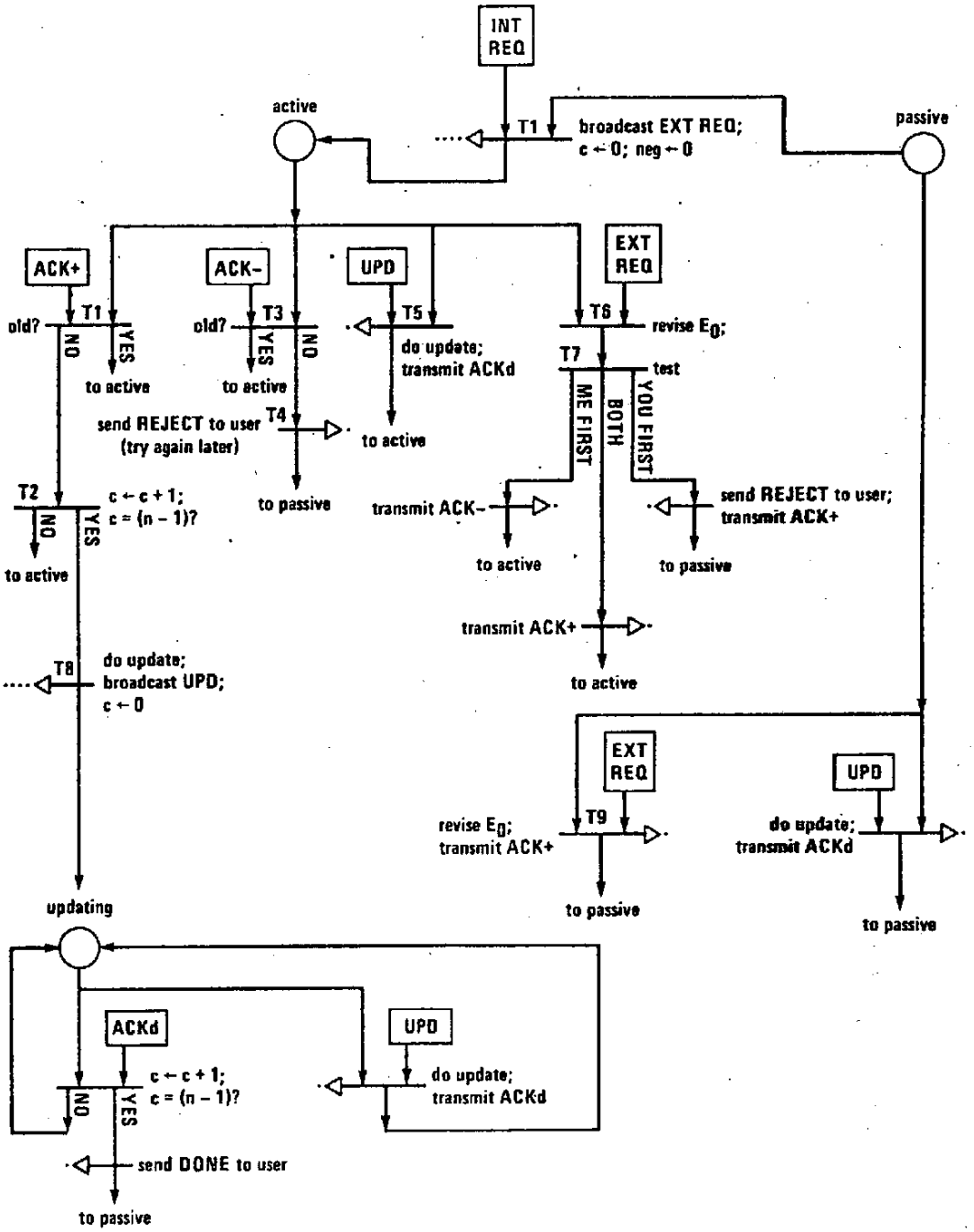


図６－１２　非集中更新プロトコル〔ＥＬＬＩ７７〕

6.2.4 異種性問題

異種性問題は、ローカル概念モデルとローカル内部モデルとの対応である〔図６－１３〕。即ち、ネットワーク全体で共通なデータモデルを用いているローカル概念モデルと、各ローカルＤＢＭＳとの変換問題である。各サイトのローカル管理者は、概念モデルを用いて、各サイトのデータベー

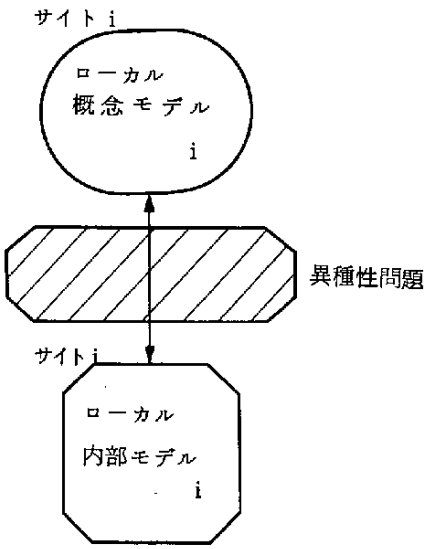


図６－１３　異種性問題

スを記述することが必要である。即ち、モデル変換が必要である。ローカル問合せから、ローカルＤＢＭＳ の対応する通信レベルへの変換と、ローカルＤＢＭＳからの応答の統合が必要となる。

6.2.5 異種性機能

異種性問題を解決するための機能として次の点を考える必要がある。

1）ローカル問合せ（local query）からローカルDBMS問合せ（request）への変換
2）ローカルDBMS問合せの半順序化と同期
3）ローカルDBMSの起動
4）ローカルDBMSからの応答の変換と合成

ＤＢＭＳ 間の通信レベルとして、ＤＢＭＳの内部機構に独立な親言語レベル及び問合せ言語レベルを用いると述べた。ネットワーク全体で共通な言語を用いたローカル問合せから、ローカルDBMSの親言語又は問合せ言語への変換が必要になる〔図６－１４〕。この様にして生成されたローカルＤＢＭＳ 問合せも、分散機能の問合せ分割と同様に、半順序化される。　半順序化されたローカル

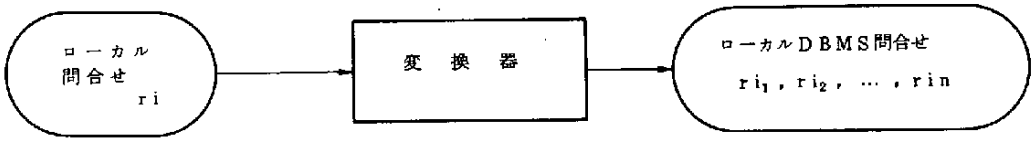


図６－１４　ローカル問合せ－ローカルDBMS問合せ変換

DBMS 問合せの実行同期を管理せねばならない。ローカル DBMS 問合せに対するローカル DBMS からの応答を合成して、ローカル問合せの応答を生成せねばならない。又、ローカルDBMS の起動も行なわねばならない。

異種性機能は、共通－ローカルの変換機能である。変換問題は困難な問題である。データベース全体の変換問題は〔SHOS７５、SHU７５、HOUS７７〕等によって扱われている。問合せ変換については〔NAHO７６〕によって扱われている。通信レベルとしては、問合せ言語レベルを考えることが、DBMS の内部機構的な異種性から独立であるために望ましい。

6.3　ま　と　め

本章では、データ共有システムに対する基本構想を論じた。コンピュータネットワークを通してリソースをアクセスする場合の問題点は、リソースの異種性と分散との２点である。一般ユーザが容易にリソースをアクセス出来るためには、この２つの問題の不可視性を獲得することが必要である。我々のリソースの統合的利用に対するアプローチは、上記２点の不可視性を備えた統一的リソースモデルと非手続的共通アクセス言語をユーザへ提供する非手続的ユーザ支援システムを構築することである。このために、異種性問題と分散問題を明確に分離して議論した。１つは異種性問題を除去する概念化であり、他は分散問題を除去する概念化である。前者によって生成されるものをローカル概念リソースモデルと呼ぶ。この各サイト単位のローカル概念リソースモデルから分散問題を除して生成されるものを全体概念リソースモデルと呼ぶ。こうすることによって、異種性と分散の２つの問題を明確に分離して議論出来る。

次に、データリソースにおける共有問題を考えた。即ち分散型データベース問題である。議論の前提として、汎用のシステムを考えるために、各データベースサイトは DBMS であるとした。前述した概念化に対応して、ローカル内部スキーマ、ローカル概念スキーマ、全体概念スキーマとの三層のスキーマを導入した。ローカル内部スキーマは、既存の DBMS へ相当し、通信レベルとしては内部機構に独立な問合せ言語又は親言語レベルを考える。ローカル概念スキーマは、各データベースサイト単位に異種性を除去し、システム全体で共通な概念モデルによって、各サイトの保有するデータの意味を記述したものである。全体概念スキーマは、各ローカル概念スキーマから企業

体にとって関心のあるデータの意味を記述したものである。このスキーマにおいて分散情報は除去されており、ローカル概念スキーマと同じ概念モデルを用いて記述されている。ユーザは、この統一的な全体概念モデルと問合せ言語を用いて、データの分散と異種性を意識することなくデータをアクセス出来る。他に全体概念スキーマから、各アプリケーションに適したデータモデルと言語とを定義することも出来る。このようにして定義されたものを外部スキーマと呼ぶ。

我々は、データ共有の問題点として次のものがあると考えている。

1）統一的なデータモデルと言語
2）分散問題
3）分散機能
4）異種性問題
5）異種性機能

この各々について、問題点の概要を述べた。意味記述のためにどのようなデータモデル（概念モデル）を用いるかは重要であり、今後十分な検討が必要である。分散問題は全体概念モデルとローカル概念モデルとの対応である。全体レベルにおけるデータ項目とサイトとの対応である。この対応は、概念モデルとしてどのようなデータモデルを考えるかに依存したものである。この様な分散問題のために必要となる機能（即ち、分散機能）として必要なものとして、ディレクトリ、問合せの分割と判順序化、一致性制御、同時実行制御について論じた。これらの機能は、分散問題において重要であり、今後の課題である。

異種性問題は、共通的なローカル概念スキーマとローカル内部スキーマ（DBMS）との変換問題である。この問題において、各データベースサイトのDBMSをどのようにみるかは重要となる。次はDBMSのどの通信レベルを用い、このレベルの特徴（データベースモデル）は何かの問題である。DBMSは最近種々のインタフェースを提供するようになり、既存のDBMSの分類は混乱を示している。今後、DBMSの分類は重要である。

本章では、汎用のアプローチについて検討した。この検討において、問題点の概要の整理を行なった。このアプローチに含まれる問題は膨大なものであり、今後多くの検討が必要となろう。文献検索の場合の様な専用システムに対しては、CONITにみられるようなインターフェース変換は有効で簡単なアプローチと考えている。

〔参考文献〕

[ABRI 74]
Abrial, J.R., "Data Semantics,"
Proc. IFIP TC-2 Working Conf. on DBMS, Cargese, Corsia, France, April 1974, pp. 1-60 (also in [KLIM 74]).

[ADIB 76]
Adiba, M., Delobel, C., and Leonard, M., "A Unified Approach for Modelling Data in Logical Data Base Design,"
Proc. IFIP TC-2 Working Conf. on Modelling in DBMS, Freudenstadt, Germany, Jan. 1976, pp. 311-338 (also in [NIJS 76a]).

[ADIB 77]
Adiba, M., and Delobel, C., "The Problem of the Cooperation between DBMS,"
Proc. IFIP TC-2 Working Conf. on Architecture and Models in DBMS, Nice, France, Jan. 1977, pp. 165-186 (also in [NIJS 77a]).

[ANDE 75]
Anderson, R.H., "Advanced Intelligent Terminals as a User's Network Interface,"
Proc. IEEE Compcon '75 Conference, Sept. 1975, pp. 25/4·1-6.

[ANDE 76a]
Anderson, R.H., and Gillogly, J.J., "Rand Intelligent Terminal Agent (RITA): Design Philosophy,"
R-1809-ARPA, Rand Corporation, Santa Monica, Feb. 1976, p. 48.

[ANDE 76b]
Anderson, R.H., and Gilogly, J.J., "Rand Intelligent Terminal Agent (RITA) as a Network Access Aid,"
AFIPS Conference Proceedings, Vol. 45, May 1976, pp. 501-509.

[ANDE 77]
Anderson, R.H., Gallegos, M., Gillogly, J.J., Greenberg, R., and Villanueva, "RITA Reference Manual,"
R-1808-ARPA, Rand Corporation, Sept. 1977, p. 68.

[ANSI 75]
"Interim Report of the Study Group on Data Management Systems,"
ANSI/X3/SPARC DBMS Study Group, Report 75-02-08 Feb. 1975.

[ANSI 76]
"Discussion of Future Directions of ANSI SPARC DBMS Study Group," Proc. 2nd Share Working Conf. on DBMS, Montreal, Canada, April 1976, pp. 207-220.

[ASTR 76]
Astrahan, M.M., et al., "System R: Relational Approach to Database Management," ACM TODS, Vol. 1, No. 2, June 1976, pp. 97-137.

[BACH 64]
Bachman, C.W., and Williams, S.B., "A General Purpose Programming System for Random Access Memories,"
Proc. AFIPS, Vol. 26 (FJCC 1964).

[BACH 77]
Bachman, C.W., and Dale, M., "The Role Concept in Data Models," Proc. 3rd International Conf. on VLDB, Tokyo, Oct. 1977, pp. 464-476.

[BEER 77]
Beeri, C., Fagin, R., and Howard, J.H., "A Complete Axiomatization for Functional and Multivalued Dependencies in Database Relations," Proc. ACM SIGMOD Conf., Toronto, August 1977, pp. 47-61.

[BELF 75]
Belford, G.G., "Research in Network Data Management and Resource Sharings,"
CAC Document No. 161, JTSA Document No. 5508, Center for Advanced Computation, Univ. of Illinois at Urbana-Champaign, May 1975, p. 27.

[BENO 73]
Benoit, J.W., "Evolution of Network User Service--The Network Resource Manager,"
Computer Networks: Trends and Applications, IEEE Inc., New York, 1973, pp. 21-24.

[BENO 74]
Benoit, J.W., and Graf-Webster, E., "REX: A Resource Location and Acquisition Service for the ARPA Computer Network,
MTP-387, The MITRE Corporation, Jan. 1974, p. 23.

[BERN 75]
Bernstein, P.A., Swenson, J.R., and Tsichritzis, D., "A Unified Approach to Functional Dependencies and Relations,"
Proc. of ACM SIGMOD Workshop on Management of Data, San Francisco, May 1975, pp. 237-245.

[BERN 76]
Bernstein, P.A., "Synthesizing Third Normal Form Relations from Functional Dependencies," ACM TODS, Vol 1, No. 4, Dec. 1976, pp. 277-298.

[BERN 77]
Bernstein, P.A., Shipman, D.W., Rothnie, J.B., and Goodman, N., "The Concurrency Control Mechanism of SDD-1: A System for Distributed Databases," Technical Report CCA-77-09, Computer Corporation of America, Cambridge, Dec. 1977, p. 172.

[BILL 77]
Biller, H., and Neuhold, E.J., "Concepts for the Conceptual Schema,"
Proc. IFIP TC-2 Working Conf. on Architecture and Model in DBMS, Jan. 1977, pp. 1-30 (also in [NIJS 77a]).

[BIRS76]
Birss, E.W., and Fry, J.P., "Generalized Software for Translating Data,"
AFIPS Conference Proceedings, Vol. 45, May 1976, pp. 889-897.

[BLAS 77]
Blasgen, M.W., and Eswaran, K.P., "Storage and Access in Relational Data Bases,"
IBM Systems Journal, Vol. 16, No. 4, 1977, pp. 363-377.

[BOBR 75]
Bobrow, D.G., and Collins, A., "Representation and Understanding," Academic Press, Inc., New York, 1975, p. 427.

[BOYC 75]
Boyce, R.F., Chamberlin, D.D., King, W.F., and Hammer, M.M., "Specifying Queries as Relational Expressions: the SQUARE Data Sublanguage,"
CACM, Vol. 18, No. 11, Nov. 1975, pp. 621-628.

[BRAC 76]
Bracchi, G., Paolini, P., and Pelagatti, G., "Binary Logical Associations in Data Modelling,"
Proc. IFIP TC-2 Working Conf. on Modelling in DBMS, June 1976, pp. 125-148 (also in [NIJS 76a]).

[BUBE 76]
Bubenko, J.A., Jr., and Berild, S., "From Information Requirements to DBTG-DATA Structures,"
Proc. Conf. on Data: Abstraction, Definition and Structure, Salt Lake City, March 1976, pp. 73-85.

[BUBE 77a]
Bubenko, J.A., Jr., "The Temporal Dimension in Information Modeling,"
Proc. IFIP TC-2 Working Conf. on Architecture and Models in DBMS, Jan. 1977, pp. 93-118.

[BUBE 77b]
Bubenko, J.A., Jr., "IAM: An Inference Abstract Modeling Approach to Design of Conceptual Schema,"
Proc. ACM SIGMOD International Conf. on Management of Data, Toronto, Canada, August 1977, pp. 62-74.

[CANA 74]
Canady, R.H., Harrison, R.D., Ivie, E.L., Ryder, J.L., and Wehr, L.A., "A Back-end Computor for Data Base Management,"
CACM, Vol. 17, No. 10, Oct. 1974, pp. 575-582.

[CANN 76]
Canning, R.D., "Distributed Data Systems," EDP Analyzer, Vol. 14, No. 6, June 1976.

[CANN 78]
Canning, R.D., "Installing a Data Dictionary,"
EDP Analyzer, Vol. 16, No. 1, January 1978, p. 13.

[CARD 76]
Cardarin, G., and Spaccapietra, S., "Integrity of Data Bases: A General Lockout Algorithm with Deadlock Avoidance,"
Proc. IFIP TC-2 Working Conf. on Modelling in DBMS, June 1976, pp. 395-411.

[CARL 74]
Carlson, W.E., and Crocker, S.T., "The Impact of Networks on the Software Markerplace,"

[CARL 76]
Carlson, C.R., and Kaplan, R.S., "A Generalized Access Path Model and Its Application to a Relational Data Base System,"
Proc. International Conf. on Management of Data, ACM-SIGMOD, June 1976, pp. 143-154.

[CASE 72]
Casey, R.G., "Allocation of Copies of a File in an Information Network," AFIPS Conference Proceedings, SJCC, 1972, pp. 617-621.

[CASE 73]
Casey, R.G., "Design of Tree Networks for Distributed Data,"
AFIPS Conference Proceedings, Vol. 42, May 1973, pp. 251-257.

[CASH 74]
Cashin, P., "DATAPAC Standard Network Protocol," INWG General Note No. 77, Nov. 1974.

[CEC 76]
Commission of the European Communities, "EURONET NEWS ISSUE No. 2," August 1976, p. 6.

[CEC 77a]
Commission of the European Communities, "EURONET NEWS ISSUE No. 7," July 1977, p. 6.

[CEC 77b]
Commission of the European Communities, "EURONET NEWS ISSUE No. 8," October 1977, p. 6.

[CEC 77c]
Commission of the European Communities, "EURONET NEWS ISSUE No. 9," December 1977, p. 6.

[CHAM 74]
Chamberlin, D.D., "SEQUEL: A Structured English Query Language," Proc. 1974 ACM SIGMOD Working Conf. on Data Description, Access and Control, May 1974,

[CHAM 75]
Chamberlin, D.D., Gray, J.N., and Traiger, I.L., "Views, Authorization, and Locking in a Relational Data Base System," AFIPS Conf. Proc., Vol. 44, May 1975, pp. 425-430.

[CHAM 76a]
Chamberlin, D.D., "Relational Data-Base Management Systems," Computing Surveys, Vol. 8, No. 1, March 1976, pp. 43-66.

[CHAM 76b]
Chamberlin, D.D. and Boyce, R.F., "SEQUEL: A Structured English Query Language,"
Proc. ACM-SIGMOD Workshop on Data Description, Access, and Control May 1976, pp. 249-264.

[CHAM 76c]
Chamberlin, D.D., Astrahan, M.M., et al., "SEQUEL 2: A Unified Approach to Data Definition, Manipulation and Control,"
IBM Journal of Research and Development, Vol. 20, No. 6, Nov. 1976, pp. 560-575.

[CHEN 73]
Chen, P.P.S., "Optional File Allocation in Multi-Level Storage Systems," AFIPS Conference Proceedings, May 1973, pp. 277-282.

[CHEN 76]
Chen, P.P.S., "The Entity-Relationship Model - Toward a Unified View of Data," ACM TODS, Vol. 1, No. 1, March 1976, pp. 9-36.

[CHIB 77]
千葉恭弘，"データ模型の分類学、データベース・モデル研究会委員会中間報告"
データベース管理システム研究会資２－３，IPSJ, July 1977, PP. 1－10.

[CHU 73]
Chu, W.W., "Optional File Allocation in a Computer Network," Computer-Communication Networks (N. Abramson and F. Juo, eds.), Prentice Hall, 1973, pp. 82-94.

[CHU 75]
Chu, W.W., and Nahouraii, E., "File Directory Design Considerations for Distributed Data Bases,"
Proc. International Conf. on 1st VLDB, Farmingham, Massachusetts, 1975, pp. 543-545.

[CHU 76]
Chu, W.W., "Performance of File Directory Systems for Data Bases in Star and Distributed Networks," AFIPS, NCC, Vol. 45, 1976, pp. 577-587.

[CHUP 76]
Chupin, J.C., Seguin, J., and Sergeant, G., "Distributed Applications on Heterogeneous Networks,"
Proc. 3rd Annual Symposium on Computer Architecture, Florida, U.S.A., Jan. 1976,

[CLIP 76]
Clipsham, W.W., Glave, F.E., and Narraway, M.L., "Datapac Network Overview," ICCC 1976, pp. 131-136.

[CODA 71]
Data Base task Group, "Data Base Task Group to the CODASYL Programming Language Committee," ACM, April 1971, p. 273.

[CODA 73]
CODASYL, "CODASYL Data Description Language Journal of Development," June 1973, NBS Handbook 113 (1974).

[CODA 75]
"A Progress Report on the Activities of the CODASYL END User Facility Task Group," June 1975.

[CODA 76]
CODASYL, "COBOL Journal of Development 1976," Canadian Government 110-GP-1d, 1976.

[CODD 70]
Codd, E.F., "A Relational Model of Data for Large Shared Data Bank," Communications of the ACM. Vol. 13, No. 6, June 1970, pp. 337-387.

[CODD 71a]
Codd, E.F., "Further Normalization of the Relational Data Base Model," Courant Computer Science Symposium 6, "Data Base System," New York City, May 1971, pp. 33-64.

[CODD 71b]
Codd, E.F., "Relational Completeness of Data Base Sublanguages," Courant Computer Science Symposium 6, "Data Base Systems," New York City, May 1971, pp. 65-98.

[CODD 74a]
Codd, E.F., "Recent Investigations into Relational Data Base Systems,"
Proc. IFIP Congress 1974, Aug. 1974, pp. 1017-1021.

[CODD 74b]
Codd, E.F., "Seven Steps to Rendezvous with the Casual Users," Proc. IFIP Working Conf. on Data Base Management, 1974, pp. 179-200.

[COMB 75]
Comba, P.G., "Needed: Distributed Control,"
Proc. International Conf. on VLDB, 1975, pp. 364-375.

[CUAD 75]
Cuadra, C.A., "SDC Experiences with Large Data Bases, Journal of Chemical Information and Computer Sciences,"
Vol. 15, No. 1, 1975, pp. 48-51.

[CURT 76]
Curtice, R.M., "The Outlook for Data Base Management," Datamation, April 1976, pp. 46-49.

[DALE 76]
Dale, A.G., and Lowenthal, E.I., "End-User Interfaces for Data Base Management Systems," Proc. 2nd Share Working Conf. on DBMS, Montreal, Canada, April 1976, pp. 81-99.

[DATE 77]
Date, C.J., "An Introduction to Data Base System," (Second Edition), Addison-Wesley, June 1977, p. 536.

[DELO 73]
Delobel, C., and Casey, R.G., "Decomposition of a Data Base and the Theory of Boolean Switching Functions,"
IBM Research and Development, Vol. 17, No. 5, Sept. 1973, pp. 374-386.

[DEPP 76]
Deppe, M.E., and Fry, J.P., "Distributed Databases: A Summary of Research," Computer Networks, Vol. 1, No. 1, Sept. 1976, pp. 130-138.

[DOUQ 75]
Douqe, B.C.M., and Nijssen, G.M. (eds.), "Data Base Description," (Proc. IEIP TC-2 Special Working Conference on Data Base Description, Wepion, Belgium, Jan. 1975), North-Holland, 1975, p. 382.

[EGEL 75]
Egeland, J., "The SUNY Biomedical Communication Network: Six years of Progress in On-Line Bibliographic Retrieval," Bull. Med. Libr. Assoc. 63(2), April, 1975, pp. 189-194.

[ELLI 77]
Ellis, C.A., "A Robust Algorithm for Updating Duplicate Databases," Proc. 2nd Berkeley Workshop on Distributed Data Management and Computor Networks, May 1977, pp. 146-158.

[ELIZ 76]
Elizabeth, J., Feinter, J., et al., "ARPANET Resource Handbook," SRI, Network Information Center, Menlo Park, Calif., Dec. 1976, p. 858.

[ESRO 73]
ESRO/ELDO, "The Space Documentation Service - A Progress Report," ESRO/ELDO Bulletin No. 23, Nov. 1973, p. 8.

[ESWA 76]
Eswaran, K.P., Gray, J.N., Lorie, R.A., and Traiger, J.L., "The Notions of Consistency and Predicate Locks in a Database System," CACM, Vol. 19, No. 11, Nov. 1976, pp. 624-633.

[EUSI 77]
EUSIDIC, "NEWSIDIC Quarterly No. 25," Autumn 1977, p. 18.

[FADO 75a]
Fadous, R., "Mathematical Foundations for Relational Data Bases," Ph.D. Dissertation, Department of Computer Science, Michigan State University, 1975, p. 88.

[FADO 75b]
Fadous, R., and Forsyth, J., "Finding Candidate Keys for Relational Data Bases,"
Proc. 1975 ACM SIGMOD International Conf. on the Management of Data, May 1975, pp. 204-210.

[FAGI 77a]
Fagin, R., "Multivalued Dependencies and a New Normal Form for Relational Databases," ACM TODS, Vol. 2, No. 3, Sept. 1977, pp. 262-278.

[FAGI 77b]
Fagin, R., "Functional Dependencies in a Relational Database and Propositional Logic," IBM Journal of Research and Development, Vol. 21, No. 6, Nov. 1977, pp. 534-544.

[FAGI 77c]
Fagin, R., "The Decomposition Versus Synthetic Approach to Relational Database Design,"
Proc. 3rd International Conf. on VLDB, Tokyo, Oct. 1977, pp. 441-446.

[FALK 76]
Falkenberg, E., "Concepts for Modelling Information,"
Proc. IFIP Working Conf. on Modelling in DBMS, Freudenstadt, Germany, Jan. 1976, pp. 95-109 (also in [NIJS 76a]).

[FALK 77a]
Falkenberg, E., "Concepts for the Coexistence Approach to Data Base Management,"
Proc. IFIP TC-2 Working Conf. on Architecture and Model in DBMS, Nice, France, Jan. 1977, pp. 39-50 (also in [NIJS 77a]).

[FALK 77b]
Falkenberg, E., "Coexistence and Transformation of Data,"
Proc. 3rd International Conf. on VLDB, Tokyo, Oct. 1977, pp. 314-317.

[FALK 77c]
Falkenberg, E., "Concepts for the Coexistence Approach to Data Base Management,"
Proc. International Computing Symposium, Liège, Belgium, April 1977, pp. 39-50.

[FARR 76]
Farrell, J., "The Datacomputer - A Network Data Utility," Proc. Berkeley Workshop on Distributed Data Management and Networks, Berkeley, May 1976, pp. 352-364.

[FRY 76]
Fry, J.P., and Sibley, E.H., "Evolution of Data-Base Management Systems," ACM Computing Surveys, Vol. 8, No. 1, March 1976, pp. 7-42.

[FUJI 77]
富士通，"富士通のオンラインデータベースシステム 開発の歴史とＡＩＭ，"ＦＡＣＯＭジャーナル，Ｖｏｌ．３，№１１，１９７７，ＰＰ．１５－２５．

[GHOS 77]
Ghosh, S.P., "Data Base Organization for Data Management," Academic Press, New York, 1977, p. 376.

[GRAD 75]
Gradwell, D.J.L., "Why Data Dictionaries?," Database Journal, Vol. 16, No. 2, pp. 15-18.

[GRAY 75]
Gray, J.N., Lorie, R.A., and Putzolu, G.R., "Granularity of Locks in a Shared Data Base,"
Proc. International Conf. on VLDB, Framingham, MA., Sept. 1975, pp. 428-451.

[GRUB 77]
Grubb, D.S., and Cotton, I.W., "Criterra for Evaluation of Data Communications Services," Computer Networks, Vol. 1, No. 6, Nov. 1977, pp. 325-340.

[GUID 75]
GUIDE, "Data Dictionary/Directory Requirements," Data Dictionary/-Directory Project Report, May 1975.

[GYOS 73]
行政情報システム研究所，"行政情報案内センターに関する調査研究報告書"，昭和４８年３月，Ｐ．２７８．

[GYOS 74]
行政情報システム研究所，"行政情報案内センターに関する調査研究報告書"，昭和４９年３月，Ｐ．２１６．

[GYOS 75]
行政情報システム研究所，"行政情報案内センターに関する調査研究"，昭和５０年３月，Ｐ．２８５．

[HABE 69]
Haberman, A.N., "Prevention of System Deadlocks,"
CACM, Vol. 12, No. 7, July 1969, pp. 373-377.

[HAIN 74]
Hainaut, J.L., and Lecharlier, R., "An Extensible Semantic Model for Data Base,"
Proc. IFIP Congress 1974, North-Holland, Amsterdam, 1974, pp. 1022-1025.

[HAIN 77]
Hainaut, J.L., "Some Tools for Data Independence in Multilevel Data Base Systems,"
Proc. IFIP Working Conf. on Architecture and Models in DBMS, Jan. 1977, pp. 187-211 (also in [NIJS 77a]).

[HALL 76]
Hall, P., Owlett, J., and Todd, S., "Relations and Entities,"
Proc. IFIP Working Conf. on Modelling in DBMS, Jan. 1976, pp. 201-220 (also in [NIJS 76a]).

[HAYA 76]
早田君子，"ＮＴＩＳの概要 ― ＡＤ・ＰＢリポートを中心に ― (1)，"科学技術文献サービス，№４４，１９７６，ＰＰ．３８－４０．

[HAYA 77]
早田君子，"ＮＴＩＳの概要 ― ＡＤ・ＰＢリポートを中心に ― (2)，" 科学技術文献サービス，№４６，１９７７，ＰＰ．３５－４０．

[HELD 75]
Held, G.D., Stonebraker, M.R., and Wong, E., "INGRESS - A Relational Data Base System,"
AFIPS Conf. Proc., May 1976, pp. 409-416.

[HELD 78]
Held, G., and Stonebraker, M., "B-trees Re-examined,"
CACM, Vol. 21, No. 2, Feb. 1978, pp. 139-143.

[HEND 77a]
Hendrix, G.G., "LIFFER: A Natural Language Interface Facility,"
Proc. 2nd Berkeley Workshop on Distributed Data Management and Computer Network, Berkeley, May 1977, pp. 196-201.

[HEND 77b]
Hendrix, G.G., Sacerdoti, E.D., Sagalowicz, D., and Slocum, J., "Developing a Natural Language Interface to Complex Data,"
Proc. 3rd International Conf. on VLDB, Tokyo, Oct. 1977, pp. 37-58.

[HEND 77c]
Hendrix, G.G., "THE LIFFER MANUAL: A Guide to Building Practical Natural Language Interfaces,"
Artificial Intelligence Center Technical Note 138, SRI International, Menlo Park, California, Feb. 1977, p. 68.

[HOLT 70]
Holt, A.F., Commoner, C., "Events and Conditions,"
Record of the Project MAC Conference on Networks, 1970, pp. 3-52.

[HOTA 77a]
Hotaka, R. & Tsubaki, M., "Self-Descriptive Relational Data Base,"
Proc. on 3rd Very Large Data Base, Tokyo, 1977, pp. 415-426.

[HOTA 77b]
穂鷹良介，"実践講座データベース(I)，データベースとは，" bit，Vol．９，№１，１９７７，PP．３１－３６.

[HOTA 77c]
穂鷹良介，"実践講座データベース（ＩＩＩ）ＤＢＭＳ具体例（その１），" bit，Vol．９，№３，１９７７，ＰＰ．２４６－２５３.

[HOTA 77d]
穂鷹良介，"実践講座データベース（Ｖ） ＤＢＭＳ具体例（その３），" bit，Vol．９，№６，１４７７，ＰＰ．６０６－６１４.

[HOTA 77e]
穂鷹良介，"実践講座データベース（ＸＩ），データベースの運用（その２）データ・ディクショナリ／ディレクトリ，" bit，Vol．９，№１３，ＰＰ．１４３４－１４３９.

[HOTA 77f]
穂鷹良介，"実践講座データベース（完），データベース，モデルと将来展望，" bit，Vol．９，№１４，１９７７，ＰＰ．１４６３－１４６９.

[HOUS 77]
Housel, B.C., "Unified Approach to Program and Data Conversion,"
Proc. 3rd International Conf. on VLDB, Tokyo, Oct. 1977, pp. 327-335.

[HUMP 74]
Humpherey, S.M., "Searching the MEDCARS Citation File On-Line Using ELHILL 2 and STAIRS: A Comparison,"
Information Storage and Retrieval, Vol. 10, 1974, pp. 321-329.

[INOU 77]
井上如，"情報サービス機関案内 ── 講座：初心者のための情報管理（第２４回）── "情報管理，Vol．１９，№１２，Mar．１９７７，ＰＰ．９４０－９４４.

[IPSJ 76]

データベース言語研究委員会，"データベース言語研究委員会報告．１９７４－１９７５年度，"
情報処理，Vol．１７，№６，１９７６，ＰＰ．５３６－５４６．

[IRIA 74]
"Presentation du Reseau Cyclades," IRIA, Feb. 1974, p. 10.

[ISHIDA 77]
石田晴久，"ベル研究所の軽装ＯＳ－ＵＮＩＸ，" 情報管理，Vol．１８，№９，Sept．１９７７，
ＰＰ．９４２－９４９．

[ISHII 77a]
石井義興，"ＡＤＡＢＡＳモデル，"情報処理学会，データベース管理システム研究会資料１，
１９７７．５．１２．

[ISHII 77b]

石井義興，"データベース管理システムＡＤＡＢＡＳの機能，"情報管理，Vol．２０，№１０，
Jan．１９７８，ＰＰ．７９８－８０５．

[ISHIHA 76]
石原和郎，"ＩＢＭの分散処理の考え方について，"情報科学，Vol．１２，№８，Aug．
ＰＰ．１４－１９．

[ISO 77]
"Basic Architecture for End to End Protocols (French Contribution on End to End Protocols - Project 17),"
ISO/TC97/SC6, Data Communications, Jan. 1977, p. 8.

[JEFF 76]
Jefferson, D.K., "The Role of the External Schema,"
Proc. 2nd Share Conf. on DBMS, Montreal, Canada, April 1976,
pp. 67-79.

[JEID 73]
日本電子工業振興協会，"データ・ベース管理システム── GUIDE／SHAREの要望書と
CODASYLレポートとの対比 ──（国際動向専門委員会報告書），４８－Ｃ－２５６，
昭和４８年３月，Ｐ．８７．

[JEID 74]
日本電子工業振興協会，"データベース管理システムに関する調査 ── 評価，運営，理論の考察
（データベース専門委員会報告書），"４９－Ｃ－２７５，昭和４９年３月，Ｐ．１０８．

[JEID 76]
日本電子工業振興協会，"データベース・システムに関する調査 ── データ・ディクショナリ／
ディレクトリ （データベース専門委員会報告書），"５１－Ｃ－３０３，昭和５１年３月，
ＰＰ．８３－１０２．

[JICS 77a]
日本科学技術情報センター，"ＪＯＩＳコマンド解説集－補遺－" Jan．１９７７

[JICS 77b]
日本科学技術情報センター，"科学技術情報ハンドブック，"科学技術庁振興局監修，Oct．１９７７，Ｐ．４２２．

[JIPD 77]
日本情報処理開発協会，"コンピュータネットワークＪＩＰＮＥＴの研究開発，" ５１－Ｓ－００３ 昭和５２年３月，Ｐ．３６７．

[KAMB 77]
Kambayashi, Y., Tanaka, K., and Yajima, S., "A Relational Data Language with Simplified Binary Relation Handling Capability," Proc. 3rd International Conf. on VLDB, Tokyo, Oct. 1977, pp. 338-350.

[KAWA 77]
川島勝，"サービス活動(4)レファレンスサービス　講座：初心者のための情報管理（第２２回），" 情報管理，Vol．９，№１０，Jan．１９７７，ＰＰ．７８６－７９２.

[KENT 73]
Kent, W., "A Primer of Normal Forms," IBM Technical Report TR02.600, IBM Res. Lab., San Jose, Dec. 1973, p. 23.

[KENT 76]
Kent, W., "New Criteria for the Conceptual Model," Proc. 2nd International Conf. on VLDB, Brussels, Belgium, Sept. 1976, pp. 1-12.

[KERS 76a]
Kershberg, L., Klug, A., and Tsichritzis, D., "A Taxonomy of Data Models," Computer Systems Research Group, University of Toronto, Technical Report CSRG-70, May 1976 (also available from IEEE Computer Society, R-77-47).

[KERS 76b]
Kershberg, L., Klug, A., and Tsichritzis, D., "A Taxonomy of Data Models," Proc. 2nd International Conf. on VLDB, Brussels, Belgium, Sept. 1976, pp. 43-64 (also appeared in [LOCK 77]).

[KILG 75]
Kilgour, F.G., "Computerized Library Networks,"
Second - UJCC, 1975, pp. 166-171.

[KIMB 75]
Kimbleton, S.R., and Schneider, G.M., "Computer Communication Networks,"
Computing Surveys, Vol. 7, No. 3, September, 1975, pp. 129-173.

[KING 73]
King, P.F., and Collmeyer, A.J., "Database Sharing - An Efficient Mechanism for Supporting Concurrent Processes,"
AFZPS Conf. Proceeding, May 1973, pp. 271-275.

[KIRS 76]
Kirshenbaum, F., "Panel Discussion on ANSI/X3/SPARC DBMS Study Group Report,"
The ANSI/SPARC DBMS Model, Proc. on 2nd Share Working Conf. on DBMS, 1976, pp. 23-34.

[KLIM 74]
Klimbre, J.W., Koffman, K.L., "Data Base Management,"
(Proc. IFIP TC-2 Working Conf. on Data Base Management, Cargese, Corsia, France, April 1974), North-Holland, 1974, p. 423.

[KUNI 77]
Kunii, T.L., and Kunii, H.S., "Design Criteria for Distributed Database Systems,"
Proc. 3rd International Conf. on VLDB, Tokyo, Oct. 1977, pp. 93-104.

[LEVI 75a]
Levin, K.D., and Morgan, H.L., "Dynamic File Assignment in Computer Networks under Varying Access Request Patterns,"
Decision Sciences Department, Univ. of Pennsylvania, Philadelphia, April 1975, p. 21.

[LEVI 75b]
Levin, K.D., and Morgan, H.L., "Optimizing Distributed Data Bases - A Framework for Research," AFIPS Conf. Proceedings, May 1975, pp. 473-478.

[LIS 76]
Lockhead Information Systems, "DIALOG Information Retrieval Service, Specializing in Computer Searching of Data Bases in Science, Technology/Engineering, Social Sciences, and Business/-Economics," May 1976.

[LOCK 77]
Lockemann, P.C., and Neuhold, E.J., (eds), "Systems for Large Data Bases,"
(Proc. the 2nd International Conf. on VLDB, Brussels, Sept. 1976), North-Holland, Amsterdam, 1977.

[MAHM 76]
Mahmoud, S., and Riordon, J.S., "Optimal Allocation of Resources in Distributed Information Networks,"
ACM TODS, Vol. 1, No. 1, March 1976, pp. 66-78.

[MANN 77]
Manning, E.G., and Peebles, R.W., "A Homogeneous Network for Data-sharing Communications,"
Computer Networks, Vol. 1, No. 4, May 1977, pp. 211-229.

[MARC 69]
Marcus, R.S., Kugel, P., and Kusik, R.L., "An Experimental Computer - Stored, Augumented Catalog of Professional Literature,"
AFIPS Conference Proceedings, Vol. 34, 1969, pp. 461-473.

[MARC 75]
Marcus, R.S., "Network Access for the Information Retrieval Application," IEEE Intercon Conference Record, April 1975, pp. (25/4) 1-7.

[MARC 76]
Marcus, R.S., and Reintjes, J.F., "The Networking of Interactive Bibliographic Retrieval Systems,"
Massachusetts Institute of Technology, Electronic Systems Laboratory, Report ESL-R-656, March 1976, NTIS Order No. PB 252, 407, p. 164.

[MARC 77]
Marcus, R.S., and Reintjes, J.F., "Computer Interfaces for User Access to Heterogeneous Information - Retrieval Systems,"
Massachusetts Institute of Technology, Electronic Systems Laboratory, Report ESL-R-739, April 1977, p. 84.

[MARI 75]
Maril, T., and Stern, D., "The Datacomputer - A Network Data Utility,"
AFIPS Conf. Proc., May 1975, pp. 389-395.

[MCFA 75]
McFarland, M.E., "The National Referential Center,"
Special Libraries, Mar. 1975, pp. 126-132.

[MCGE 76]
McGee, W.C., "On User Criteria for Data Model Evaluation,"
ACM TODS, Vol. 1, No. 4, Dec. 1976, pp. 370-387.

[MEDI 77]
医療情報システム開発センター，"医療ネットワークコマンド設計書（ネットワークプロセシングシステム），"昭和52年5月，P．237．

[MERL 74]
Merlin, P.M., "A Study of Recoverability of Computing Systems,"
Ph.D. Dissertation, Dept. of Information and Computer Science,
University of California, Irvine, 1974, p. 165.

[MERL 76]
Merlin, P.M., and Farber, D.J., "Recoverability of Communication Protocols - Implications of a Theoretical Study,"
IEEE Transactions on Communications, Sept. 1976, pp. 1036-1043.

[MICH 76]
Michaels, A.S., Mittman, B., and Carlson, C.R., "A Comparison of the Relational and CODASYL Approaches to Data Base Management,"
ACM Computing Surveys, Vol. 8, No. 1, March 1976, pp. 125-151.

[MILL 75]
Miller, L.A., "Naive Programmer Problems with Specification of Transfer-of-control,"
Proc. NCC 75, Vol. 44, 1975, pp. 653-657.

[MORG 77]
Morgan, H.L., and Levin, K.D.,
"Optimal Program and Data Locations in Computer Networks,"
CACM, Vol. 20, No. 5, May 1977, pp. 315-322.

[MORR 77]
Morris, P., and Sagalowicz, D., "Managing Network Access to a Distributed Database,"
Proc. 2nd Berkeley Workshop on Distributed Data Management and Computer Network, May 1977, pp. 58-67.

[MOUL 76]
Moulin, P., Randon, J., Teboul, M., et al., "Conceptual Model as a Data Base Design Tool,"
Proc. IFIP TC-2 Working Conf. on Modelling in DBMS, Freudenstadt, Germany, Jan. 1976, pp. 221-238 (also in [NIJS 76a]).

[MCQU 77]
McQuillan, J.M., and Walden, D.C., "The ARPA Network Design Decisions,"
Computer Networks, Vol. 1, No. 5, Aug. 1977, pp. 243-289.

[NAHO 76]
Nahouraii, E., Brooks, L.O., and Cardenas, A.F., "An Approach to Data Communication between Different Generalized Data Base Management Systems,"
Proc. 2nd International Conf. on VLDB, Sept. 1976, pp. 117-142 (also in [LOCK 76]).

[NEGU 77]
Negus, A.E., "EURONET Guideline: Standard Commonds for Retrieval Systems,"
Final Report on a Study Carried Out for the Commission of the European Communities, DG XIII, December 1977, p. 66.

[NIH 76]
National Institute of Health, "Communication in the Service of American Health ... A Bicentennial Report from the National Library of Medicine,"
Development of Health, Education, and Welfare, Public Health Service, 1976, p. 198.

[NIJS 76a]
Nijssen, G. (eds.), "Modelling in Database Management Systems (Proceedings of the IFIP TC-2 Working Conference on Freudenstadt, Jan. 1976)," North-Holland, 1976.

[NIJS 76b]
Nijssen, G., "A Gross Architecture for the Next Generation Data base Management Systems,"
Proc. IFIP TC-2 Working Conf. on Modelling in Database Management Systems, Freudenstadt, Jan. 1976, pp. 1-24 (also in [NIJS 76a]).

[NIJS 77a]
Nijssen, G. (eds), "Architecture and Model in Database Management Systems,"
Proc. of the IFIP TC-2 Working Conf., Nice, France, Jan. 1977, North-Holland, 1977.

[NIJS 77b]
Nijssen, G., "Current Issues in Conceptual Schema Concepts,"
Proc. IFIP Working Conf. on Architecture and Model in Database Management Systems, Jan. 1977, pp. 31-66 (also in [NJJS 77a]).

[NIJS 77c]
Nijssen, G., "A Gross Architecture for the Next Generation DBMS," Information Processing 77 (Proc. IFIP Conf.), 1977, pp. 327-335.

[NIKK 65]
日刊工業新聞社，"情報の提供とサービス — 情報管理実務講座６，"情報管理実務講座編集委員会編，１９６５.

[NISI 76]
西野博二，"データベース・システムの研究開発動向，"情報処理，Vol.１７，№１０，１９７６，ＰＰ.８７８－８８２.

[NIPD 66]
日本ドクメンテーション協会，"科学と政府と情報 — 米国政府に対するワインバーグ報告，"ＮＩＰＤＯＫシリーズ１，１９６６，Ｐ.５０.

[NIPD 70]
日本ドクメンテーション協会，"ＮＩＳＴとその周辺 —科学技術会議の答申を中心として —，"ＮＩＰＤＯＫシリーズ１２，１９７０，Ｐ.８１.

[NIPD 77]
日本ドクメンテーション協会，"農林水産試験所課題 機械検索システム（ＲＥＣＲＡＳ）開発報告 —農林省農林水産技術会議事務局編 —，"ＮＩＰＤＯＫシリーズ２０，１９７７，Ｐ.９９.

[NUTT 72]
Nutt, G.J., "Evaluation Nets for Computer System Performance Analysis,"
AFIPS Conference Proceedings (FJCC), 1972, pp. 279-286.

[ODA 66]
小田泰正編，"レファレンスワークシリーズ— 図書館の仕事１４，" 日本図書館協会，１９６６，Ｐ.１２.

[OHIS 77]
大石雅彦，"複合構造を持つデータベース ＳＹＳＴＥＭ２０００について，"情報処理学会，データベース研究会資料１，１９７７.５.１２.

[ORGA]
Organik, E.J., "The MULTICS System: An Examination of Its Structure,"
MIT Press, 1972, p. 392.

[OWLE 77]
Owlett, J., "Defferring and Defining in Databases,"
Proc. IFIP Working Conf. on Architecture and Models in Data Base Management Systems, Jan. 1977, pp. 277-291.

[PAXT 77]
Paxton, W.H., "A Framework for Speech Understanding,"
Artificial Intelligence Center, Technical Note 142, SRI, June 1977, p. 280.

[PETE 77]
Peterson, J.J., "Petri Nets,"
ACM Computing Survey, Vol. 9, No. 3, Sept. 1977, pp. 223-252.

[PLAG 72]
Plagman, B.K., and Altshuler, G.P., "A Data Dictionary/Directory System within the Context of an Integrated Corporate Data Base,"
AFIPS, FJCC, Vol. 40, 1972, pp. 1133-1140.

[POUZ 75]
Pouzin, L., "The CYCLADES Network - Present State and Development Trends,"
RES 505.2, July 1975, p. 11.

[POUZ 77]
Pouzin, L., "Packet Networks - Issues and Choices,"
Proc. IFIP Congress 77, Toronto, Aug. 1977, pp. 515-521.

[PYKE 74]
Pyke, T.N., Jr., "Network Access Techniques: Some Recent Development,"
Proc. 3rd Annual Texas Conference, Oct. 1974, pp. 2-2-1 - 2-2-3.

[REIN 69]
Reintjes, J.F., "System Characteristics of Intrex,"
AFIPS Conference Proceedings, Vol. 34, 1969, pp. 457-459.

[RIES 77]
Ries, D.R., and Stonebraker, M., "Effects of Locking Granularity in a Database Management System,"
ACM TODS, Vol. 2, No. 3, Sept. 1973, pp. 233-246.

[RISS 77]
Rissan, J., "Independent Components of Relations,"
ACM TODS, Vol. 2, No. 4, Dec. 1977, pp. 317-325.

[RITC 74]
Ritchie, D.M., and Thompson, K., "The UNIX Time-sharing System," CACM, Vol. 17, No. 7, July 1974, pp. 365-375.

[RIYO 77]
電子計算機利用に関する技術研究会周辺問題分科会制度研究班，"昭51年度　制度研究班報告書，" 1977.3，P.171.

[ROBE 70]
Roberts, L.G., and Wessler, B.D., "Computer Network Development to Achieve Resource Sharing," AFIPS Conf. Proc. (FJCC), 1970, pp. 543-549.

[ROBE 77]
Roberts, L.G., "Packet Network Design - The Third Generation," Information Processing 77, North-Holland, 1977, pp. 541-546.

[ROSE 68]
Rosenfeld, A., "An Introduction to Algebraic Structures," Holden-Day, San Francisco, California, 1968, p. 285.

[ROSE 74]
Rosenthal, R., and Watkins, S., "Automated Access to Network Resources: A Network Access Machine," Computer-Networks - Trends and Applications, NBS-ICST and IEEE Computer Society, May 1974.

[ROSE 75]
Rosenthal, R., "Accessing On-line Network Resources with a Network Access Machine," IEEE Intercon Conf. Record., April 1975, pp. (25/3) 1-4.

[ROSE 76a]
Rosenthal, R., "Network Access Techniques - A Review," AFIPS Conference Proceedings, Vol. 45, May 1976, pp. 495-499.

[ROSE 76b]
Rosenthal, R., "A Review of Network Access Techniques with a Case Study: The Network Access Machine," NBS Technical Note 917, NBS, July 1976, p. 29.

[ROSENK 77]
Rosenkrantz, D.J., Stearns, R.E., and Lewis, P.M., "A System Level Concurrency Control for Distributed Database Systems," Proc. 2nd Berkeley Workshop on Distributed Data Management and Computer Networks, Berkeley, May 1977, pp. 132-145.

[ROTH 74]
Rothnie, J.B., "An Approach to Implementing A Relational Data Management System,"
Proc. ACM SIGMOD Workshop on Data Description, Access, and Control, 1974, pp. 277-294.

[ROTH 75]
Rothnie, J.B., "Evaluating Inter-entry Retrieval Expressions in A Relational Data Base Management System,"
AFIPS Conf. Proceedings, Vol. 44, May 1975, pp. 417-423.

[ROTH 77a]
Rothnie, J.B., Goodman, N., and Bernstein, P.A., "The Redundant Update Methodology of SDD-1: A System for Distributed Databases (the Fully Redundant Case),"
CCA Technical Report CCA-77-02, Computer Corporation of America, June 1977, p. 70.

[ROTH 77b]
Rothnie, J.B. and Goodman, N., "An Overview of the Preliminary Design of SDD-1: A System for Distributed Databases,"
Proc. on 2nd Berkeley Workshop on Distributed Data Management and Computer Networks, UC Berkeley California, May 1977, pp. 39-57.

[ROTH 77c]
Rothnie, J.B., and Goodman, N., "A Survey of Research and Development in Distributed Database Management,""
Proc. 3rd International Conf. on VLDB, Tokyo, Oct. 1977, pp. 48-62.

[ROUS 75]
Roussopoulos, N., and Mylopoulos, J., "Using Semantic Networks for Data Base Management,"
Proc. International Conf. on VLDB, Framingham, MA., Oct. 1975, pp. 144-172 (also in [DOUQ 75]).

[SACE 77]
Sacerdoti, E.D., "Language Access to Distributed Data with Error Recovery,"
Proc. 5th International Joint Conf. on Artificial Intelligence, MIT, Cambridge, August 1977, pp. 196-202.

[SAGA 77]
Sagalowicz, D., "IDA: An Intelligent Data Access Program,"
Proc. 3rd International Conf. on VLDB, Tokyo, Oct. 1977, pp. 293-302.

[SAKA 76]
酒井博敬，"データ・ディクショナリ／ディレクトリ（ＤＤ／Ｄ）の動向，"情報処理学会，データベース研究会資料ＤＢ２７－１，１９７６．３．１１，Ｐ．１０．

[SAKA 77]
酒井博敬，"データ・ディクショナリ／ディレクトリの動向，"情報処理，Ｖｏｌ．１８，№５，Ｍａｙ１９７７，ＰＰ．４９１－４９８．

[SAKA 78]
酒井博敬，"リレーショナルモデルの動向を探る，"ビジネスコミュニケーション，Ｖｏｌ．１５，№２，１９７８，ＰＰ．４９－５５．

[SCAN 74]
Scantlebury, R.A., and Wilkinson, P.T., "The National Physical Laboratory, Data Communication Network,"
Proceedings of the Second ICCC, Stockholm, August 1974, p.223.

[SCHA 76]
Schantz, R.E., and Millstein, R.E., "The FOREMAN: Proving the Program Execution Environment for the National Software Works,"
BBN Report No. 3266, March 1976, p. 58.

[SCHM 75]
Schmid, H.A., "On the Semantics of the Relational Data Model,"
Proc. International Conf. on Management of Data, ACM SIGMOD, San Jose, May 1975, pp. 211-223.

[SCHM 77]
Schmid, H.A., "An Analysis of Some Constructs for Conceptual Models,"
Proc. IFIP TC-2 Working Conf. on Architecture and Model in DBMS, Jan. 1977, pp. 119-148.

[SCHU 77]
Schussel, G., "The Role of the Data Dictionary,"
Datamation, June 1977, pp. 129-142.

[SENK 73]
Senko, M.E., Altman, E.B., Astrahan, M.M., and Fehder, P.L., "Data Structures and Accessing in Data Base Systems,"
IBM Systems Journal, Vol. 12, No. 1, 1973, pp. 30-93.

[SENK 75a]
Senko, M.E., "The DDL in the Context of a Multilevel Structured Description: DIAM II with the FORAL,"
Proc. IFIP TC-2 Working Conf. on Data Base Description, Jan. 1975, pp. 239-257.

[SENK 75b]
Senko, M.E., "Specification of Stored Data Structures and Desired Output Results in DIAM II with FORAL,"
Proc. International Conf. on VLDB, Sept. 1975, pp. 557-571.

[SENK 76a]
Senko, M.E., "DIAM as a Detailed Example of the ANSI SPARC Architecture,"
Proc. IFIP Working Conf. on Modelling in DBMS, Jan. 1976, pp. 73-94 (also in [NIJS 76a]).

[SENK 76a]
Senko, M.E., "DIAM II: The Binary Infological Level and Its Data Base Language - FORAL,"
Proc. Conf. on Data: Abstraction, Definition, and Structure, Salt Lake City, Utah, March 1976, pp. 121-140.

[SENK 76c]
Senko, M.E., and Altman, E.B., "DIAM II and Levels of Abstraction, The Physical Device Level: A General Model for Access Methods,"
Proc. 2nd International Conf. on VLDB, Brussels, Sept. 1976, pp. 79-94 (also in [LOCK 76]).

[SENK 77a]
Senko, M.E., "Conceptual Schemas, Abstract Data Structures, Enterprise Descriptions,"
Proc. International Computing Symposium, Liège, Belgium, April 1977, pp. 85-102.

[SENK 77b]
Senko, M.E., "Data Structures and Data Accessing in Data Base Systems: Past, Present, Future,"
IBM Systems Journal, Vol. 16, No. 3, 1977, pp. 208-257.

[SHAR 76]
Sharman, G.C.H., "A Constructure Definition of Third Normal Form,"
Proc. ACM SIGMOD International Conf. on Management of Data, Washington, D.C., June 1976, pp. 91-99.

[SHAW 74]
Shaw, A.C., "The Logical Design of Operating Systems,"
Prentice-Hall, Inc., 1974, p. 306.

[SHEM 72]
Shemer, J.E., and Collmeyer, A.J., "Database Sharing: A Study of Interference, Roadblock and Deadlock,"
Proc. ACM SIGFIDET, May 1972, pp. 147-163.

[SHEN 77]
Shenk, K.L., and Pinkert, J.R., "An Algorithm for Servicing Multi-relational Queries,"
Proc. ACM SIGMOD Conf. on Management of Data, Toronto, Canada, Aug. 1977, pp. 10-19.

[SHOS 75]
Shoshani, A., "A Logical Level Approach to Data Base Conversion,"
Proc. SIGMOD Conf., San Jose, California, May 1975.

[SHIM 77]
島内武彦，北川敏夫（編），"広域大量情報の高次処理"，東京大学出版会，Jan.1977，P.1253.

[SHU 75]
Shu, N.C., Housel, B.C., and Lum, V.Y., "CONVERT: A High Level Translation Definition Language for Data Conversion,"
CACM, Vol. 18, No. 10, Oct. 1975, pp. 557-567.

[SIBL 73]
Sibley, E.H., and Taylor, R.W., "A Data Definition and Mapping Language,"
CACM, Vol. 16, No. 12, Dec. 1973, pp. 750-759.

[SIBL 76]
Sibley, E.H., "The Development of Database Technology,"
ACM Computer Surveys, Vol. 8, No. 1, March 1976, pp. 1-5.

[SOFT 77a]
Software AG, "ADABAS 概説書," 1977.

[SOFT 77b]
Siftware AG, "ADABAS Reference Manual," 1977.

[SOFT 77c]
Software AG, "ADABAS Utilities Manual," 1977.

[SOFT 77d]
Software AG, "ADASCRIPT Manual," 1977.

[SOFT 77e]
Software AG, "ADAMINT Reference Manual," 1977.

[SOFT 77f]
Software AG, "ADAWRITER Reference Manual," 1977.

[SOFT 77g]
Software AG, "Programmer's Introduction to ADABAS," 1977.

[SORI 74]
総理府統計局，"昭和４８年度統計データバンク－統計データの収集利用に関する調査研究報告，"統計データバンク研究資料（３３），昭和４９年３月，Ｐ．７９．

[STEE 75a]
Steel, T.B., "Data Base Standardization - A Status Report," Proc. ACM-SIGMOD 1975, Conf., San Jose, California, May 1975, pp. 65-78.

[STEE 75b]
Steel, T.B., Jr., "Data Base Standardization: A Status Report," Proc. IFIP TC-2 Working Conf. on Data Base Description, Wepion, Belgium, Jan. 1975, pp. 183-198 (also in [DOUQ 75]).

[STEE 76]
Steel, T.B., Jr., "Formalization of Conceptual Model," Proc. 2nd Share Working Conf. on DBMS, Montreal, April 1976, pp. 181-206.

[STON 73]
Stonebraker, M., "High Level Integrity Assurance in Relational Data Base Management Systems," ERL Report ERL-M473, Electronic Research Lab., College of Engineering, UC Berkeley, Aug. 1973, p. 40.

[STON 75]
Stonebraker, M., "Implementation of Integrity Constraints and Views by Query Modification," Proc. International Conf. on Management of Data, May 1975, pp. 65-78.

[STON 76]
Stonebraker, M., Wong, E., Kreps, P., and Held, G., "The Design and Implementation of INGRES," ACM Transactions on Database Systems, Vol. 1, No. 3, September 1976, pp. 189-222.

[STON 77]
Stonebraker, M., and Neuhold, E., "A Distributed Data Base Version of INGRES," Proc. on 2nd Berkeley Workshop on Distributed Data Management and Computer Networks, UC. Berkeley, California, May 1977, pp. 19-36.

[SUMM 75]
Summit, R.K., "Lockhead Experience in Processing Large Data Bases for Its Commercial Information Retrieval Service,"
Journal of Chemical Information and Computer Sciences, Vol. 15, No. 1, 1975, pp. 40-42.

[TAKI 78]
Takizawa, M., Hamanaka, E., and Ito, T., "Resource Integration and Data Sharing on Heterogeneous Resource Sharing System,"
submitted in ICCC'78, Kyoto, Oct. 1978.

[TANA 76]
田中功，"クリアリング情報 ── 講座：情報の効果的な入手と利用法（第7回），"情報管理，
Vol.№19～Vol.19，№7，Oct.1976，PP.543－548.

[TAYL 76]
Taylor, R.W., and Frank, R.L., "CODASYL Data-Base Management Systems,"
ACM Computing Surveys, Vol. 8, No. 1, March 1976, pp. 67-103.

[TEOR 77]
Teorey, T.J., "The Database Design Evaluator, Part I: Overview,"
Working Paper DE 7.2-1, Data Translation Project, Graduate School of Business Administration, The University of Michigan, Ann Arbor, Michigan, July 1977, p. 35.

[THOM 73]
Thomas, R.H., "A Resource Sharing Executive for the ARPANET,"
AFIPS Conference Proceedings, Vol. 42, May 1973, pp. 155-163.

[THOM 75]
Thomas, D.R., "DATAMANAGER - A Free Standing Data Dictionary System,"
Database Journal, Vol. 16, No. 2, pp. 14-17.

[THOM 76]
Thomas, R.H., "A Solution to the Update Problem for Multiple Copy Data Bases which Used Distributed Control,"
BBN Report No. 3340, BBN, July 1976, p. 50.

[TODD 76]
Todd, S.J.P., "The Peterlee Relational Test Vehicle - A System Overview,"
IBM Systems Journal, Vol. 15, No. 4, 1976, pp. 285-318.

[TOTT 77]
ビル・トッテン，"ＴＯＴＡＬ，"情報処理学会，データベース管理システム研究会資料１，１９７７．５．１２．

[TSIC 75]
Tsichritzis, D.C., "A Network Framework for Relation Implementation,"
Proc. IFIP TC-2 Working Conf. on Data Base Description, Wepion, Belgium, Jan. 1975, pp. 269-282.

[TSIC 76]
Tsichritzis, D., and Lochovsky, F., "Views on Data,"
Proc. the Second Share Working Conf. on Data Base Management Systems, Montreal, Canada, April, 1976, pp. 51-65.

[TSIC 77]
Tsichritzis, D.C. and Lochovsky, F.H., "Data Base Management Systems,"
Computer Science and Applied Mathematics, A Series of Monographs and Textbooks, University of Maryland, 1977, p. 388.

[UEMU 72]
植村俊亮，"オンライン文献検索システム――イントレックスを中心に――ＩＲ講座第２４回，"情報管理，Ｖｏｌ．１５，№８，Ｎｏｖ．１９７２，ＰＰ．５５９－５６６．

[UEMU 76]
植村俊亮，"ＣＯＤＡＳＹＬ方式のデータベース・システム，"情報処理，Ｖｏｌ．１７，№１０，１９７６，ＰＰ．８９２－９０３．

[UEMU 78]
植村俊亮，"ＣＯＤＡＳＹＬデータベース用共通言語の進展　'７８，"情報処理学会，データベース管理システム研究会資料５－３，１９７８．１．１７．

[UHRO 73]
Uhrowczik, P.P., "Data Dictionary/Directories,"
IBM Systems Journal, Vol. 12, No. 4, 1973, pp. 332-350.

[UTSU 76a]
宇津野宏二，"ことばの泉　ＥＵＲＯＮＥＴ，"情報管理，Ｖｏｌ．１９，№１，Ａｐｒ．１９７６，ＰＰ．５９．

[UTSU 76b]
宇津野宏二，"ヨーロッパにおける科学技術情報ネットワーク，"情報管理，Ｖｏｌ．１９，№２，May　１９７６，ＰＰ．６９－８０．

[WAGN 76]
Wagner, F.V., "Is Decentralization Inevitable?,"
Datamation, Vol. 22, No. 11, Nov. 1976, pp. 86-97.

[WANG 75]
Wang, C.P., and Wedekind, "Segment Synthesis in Logical Database Design,"
IBM Journal of Research and Development, Jan. 1976, pp. 71-77.

[WATA 76]
渡辺純一，"階層構造のデータベース，"情報処理，Vol.17，№10．Oct．1976，PP．883－891.

[WEYL 75]
Weyl, S., "An Interlisp Relational Data Base System,"
Technical Report II, SRI, Nov. 1975, p. 35.

[WHIT 75]
Whitby-Strevens, C., "Current Research in Computer Networks,"
ACM Computer Communication Review, April 1976, Vol. 6, No. 2
pp. 13-40.

[WINO 75]
Winograd, T., "Frame Representations and the Declarative - Procedural Controversy,"
Representation and Understanding (Bobrow and Collins [eds]),
Academic Press, 1975, pp. 185-210 (in [BOBR 75]).

[WONG 76]
Wong, E., and Youssefi, K., "Decomposition - A Strategy for Query Processing,"
ACM TODS, Vol. 1, No. 3, Sept. 1976, pp. 223-241.

[WONG 77]
Wong, E., "Retrieving Dispersed Data from SDD-1: A System for Distributed Databases,"
Proc. 2nd Berkeley Workshop on Distributed Data Management and Computer Network, Berkeley, May 1977, pp. 217-235.

[WOOD 75]
Woods, W.A., "What's in a link: Foundations for Semantic Networks,"
Representation and Understanding (Bobrow, D.G., and Collins, A., [eds]), Academic Press, 1975, pp. 35-82.

[YAMA 77]
山口義一，"世界の図書館<米> LC National Referal Center，"
科学技術文献サービス，№48，Dec.1977，PP．30－34.

[YORM 76]
Yormark, B., "The ANSI/X3/SPARC/SGDBMS Architecture,"
Proc. 2nd Share Working Conf. on DBMS, Montreal, Canada, April 1976, pp. 1-21.

[ZANI 76]
Zaniolo, C.A., "Analysis and Design of Relational Schemata for Database Systems,"
Ph.D. Dissertation, Computer Science Dept., Univ. of California, Los Angeles, 1976.

[ZIMM 77]
Zimmermann, H., "The Cyclades Experience - Results and Impacts,"
Proc. IFIP Congress 77, Toronto, Aug. 1977, pp. 465-470.

[ZLOO 75]
Zloof, M.M., "Query By Example," AFIPS Conf. Proceedings, May 1975, pp. 431-437.

[ZLOO 77a]
Zloof, M.M., "Query-By-Example: A Data Base Language,"
IBM Systems Journal Vol. 16, No. 4, 1977, pp. 324-343.

[ZLOO 77b]
Zloof, M.M., and de Jong, S.P., "The System for Business Automation (SBA): Programming Language,"
CACM, Vol. 20, No. 6, June 1977, pp. 385-396.

〔英文資料〕

An Approach to Resource Integration and Data Sharing

Abstract:

One of the problems confronting us today concerning the resource sharing system, in which resources are distributed over a wide area, is how to integrate them distributed among heterogeneous computers. For casual users in particular, who constitute a great majority of the system's users, it is required that all the resources within the system be integratedly managed and that easy access to its full functions thereof can be guaranteed.

This paper presents one approach toward solving that problem. The approach we propose herein is intended to provide casual users of the system with the function of the user assistant for resource integration.

In this paper, therefore, an outline of the user assistant is considered while emphasis is placed on data resource sharing.

CONTENTS

1. Introduction

Since the turn of the '70s the distributed resource sharing system has made a great impact in the field of information processing. But a number of problems have yet to be solved before the distributed resource sharing system reaches its final stage for practical application.

One of the problems is how to overcome the heterogeneity of host computers. Today, users have to master various access procedures due to the heterogeneity of host computers in order to make full use of the system. Under such situation it is very difficult for casual users, who form a great majority of the system's users, to join the system.

Another big problem to be solved in the future is how to achieve the so-called distributed database management[6) 19)] which would make it possible to provide uniform access by logically integrating the data distributed among multiple heterogeneous computers.

First of all, we must define the distributed resource sharing system as a system which can integrate various resources distributed among multiple heterogeneous computers, and obtain the desired results by making these computers share the processing responsibility through using the integrated resources. The purpose of the user assistant is to provide users with a means of easy access to resources by integratedly managing the resources distributed among heterogeneous computers in the form of generic resources.

In order to integrate these various resources, the following three functions are required (See Fig. 2.2).

1) CP (Coordinate Processor)

*The function to analyze the content of the user's request, distribute the necessary processing into various resources on the system, and execute them.

*The high-level, non-procedural interface for the casual users.

2) NCC (Network Clearing Center)

*The information center to integratedly manage the resources on the distributed resource sharing system.

3) LIP (Local Information Processor)

*The function of efficient translation between resources distributed among heterogeneous computers.

We consider the data to be the most important resource on the distributed resource sharing system. We see many great needs for this data service in the database of the public information service, the service system which provides scientific informations,[25)] etc. In the CYCLADES[9)] the database is also considered very important. Due to the importance of data

as a resource as seen above the data sharing problem is particularly important in considering resource integration.

In this paper we will discuss 1) the outline of the user assistant for casual users, and 2) the data sharing problem in the resource integration. Description of the user assistant for resource integration is given in Chapter 2. In that chapter, first of all, the classification of the existing user assistants is made in the light of the important role played by casual users on the distributed resource sharing system, and through that classification it is pointed out that non-procedural resource integration is still an unresolved field and is of importance.

When we consider the resources distributed among heterogeneous computers, it is important how to describe them. In this regard, we would like to introduce two concepts, that is, 1) the explanatory description of resources as a common concept of class or group of resources, and 2) the attribute description of resources as detailed and precise description of actual resources. This kind of method of defining resources is our fundamental way of thinking. Let us later consider the outline of the user assistant structure for the sake of the non-procedural resource integration. In describing the user assistant, the user's requirements must be analyzed and the expression thereof must be formulated in a manner enabling them to be processed in the actual system. Finally, processing expression and its translation in the user assistant will be taken up.

In Chapter 3 the data sharing problems in WPE (the processing expression executable on the actual system) are treated vis-a-vis the following three points:

1) DC location and identification,
2) DC transfer and access, and
3) DC item translation.

DC (data collection) is composed of database, file record and item.

2. User Assistant for Resource Integration

In this chapter, the outline of the resource integration system for casual users is covered.

2.1 The Classification of User Assistants

We define the user assistant as a system which enables users to make easy use of services provided by the service vendor of the distributed resource sharing system. Such user assistants as NAM,[17] REX[3] and RITA[1) 2)] have been developed so far.[14) 18)]

Here, we will classify the functions of these user assistants with regard to the following two points:

1) The service level provided to users, and

2) The number of hosts to which users can access concurrently through the user assistant.

Item (1) represents the service level, that is to say, whether the required services can be instructed to the user assistant by way of the procedural language or the non-procedural one. We call the former case the "how-level (procedural level)" and the latter the "what-level (non-procedural level)." The "what-level" is higher than the "how-level" service.

As for Item (2) the question is whether users can access to a single or multiple host with the help of the user assistant. We call the former "access to a single resource" and the latter "resource integration." The above-mentioned classification is shown in Fig. 2.1.

The "procedural resource access" is a system in which users cannot use more than one host at a time, and can only instruct the required service by way of "how-level" language. An example of this is NAM. The "non-procedural resource access" is a system in which users cannot use more than one host at a time, but can instruct the required service by way of "what-level" language, such as REX and RITA.

In the case of "procedural resource integration," users can make use of multiple hosts by way of the how-level language. Examples of this case are CONIT[10] and RSEXEC.[24] Herein we can define the resource integration system as a system in which users can access resources on multiple hosts through the user assistant.

Judging from the classification as shown above we can conclude that the resource integration of the what-level, or "non-procedural resource integration (part of the oblique lines in Fig. 2.1)," is a still unresolved field. The non-procedural resource integration is a field remaining unresolved but at the same time if it were developed it would be the highest level of application for users.

2.2 The Description of Resources

There are two methods of resource description regarding the distributed resource processing system as shown below.

1) explanatory description, and

2) attribute description

The explanatory description corresponds to the common concept of class or group of resources. We call this common concept the "generic resource.[3]" Meanwhile, the attribute description is the detailed and precise description of actual

resources. In the explanatory description the intended meaning is rather obscure. But users tend to view resources in the system by way of the explanatory description. On the other hand, the attribute description is precise. But it is difficult for users to understand the information on resources in the system by its description alone.

For instance, "FORTRAN" is an explanatory description. But in actuality, the FORTRAN program is executed by such compilers as FORTRAN-H and FORTRAN-IV. The description on the FORTRAN system is, therefore, the attribute description.

Thus, compared to the attribute description on resources, the explanatory description is not clear. The problem with the explanatory description is, therefore, how to classify or group actual resources. As for DC resources, the data key corresponds to the generic resource name. (See Section 2.3)

For users a more explanatory description is desirable, but the actual system is described by the attribute description. In this sense we have to consider both the explanatory language (or logical processing expression) and the attribute language (or whole processing expression). It can be said that the language problem with non-procedural resource integration lies in the translation from the explanatory description into the attribute expression (L-W translation).

The resource considered here includes data, programs (called process) and hardware. As mentioned in Chapter 1 data is the most important among the resources.

2.3 The Overall Structure of User Assistant for Non-Procedural Resource Integration

The user assistant has such functions as analyzing users' requests, selecting the most appropriate ones from resources in the distributed resource sharing system, allocating necessary resources, generating the processing expression executable on the actual system, allocating necessary jobs to each host computer and monitoring the whole jobs.

The logical elements of the user assistant are the following three factors:

1) CP (Coordinate Processor)
2) NCC (Network Clearing Center)
3) LIP (Local Information Processor)

The overall structure of the user assistant is shown in Fig. 2.2.

2.3.1 CP (Coordinate Processor)

CP is the decision process for resource integration. CP is generated by each user, especially at the predefined host. The place where the CP is generated is called the home host,

which keeps users' profiles. Details of the user profile are covered in 2.3.2.

CP has the following functions:

1) To translate the user's request or objective into an expression of the whole processing that is executable on the actual system,

2) To assign jobs to each host computer processing necessary resources by making a selection from the processing expressions, and

3) To monitor the job execution on each host computer.

CP obtains information necessary for the execution of the above-mentioned functions from NCC.

2.3.2 NCC (Network Clearing Center)

NCC holds resource information, called NCC information, necessary for the distributed resource sharing system. We divide these resources into two categories as shown below from the viewpoint of utilization.

1) public resources

2) private resources

Public resources are shared by the whole system and used without restriction. Private resources are usable only by predefined users or groups of users under a certain access mode. The resources managed by NCC are public resources, and NCC possesses the resource directory as shown in Fig. 2.3. With the help of this directory detailed information on resources such as the actual location of resources and the access mode can be obtained from the generic resource names.

The reason why the concept of the resource description has been adopted herein is stated below. In the distributed resource sharing system, which is aimed at resource sharing, there exist various kinds and a wide range of heterogeneous resources. We consider that detailed information on individual resources should be independent of the resources themselves. If so, it would become possible to handle the information and its management uniformly and it would become easy to cope with various kinds of processing and queries.

Private resources are managed by each individual user by the private resource directory (See Fig. 2.4) according to their own user profiles. That is to say, the directory comprises: resource names → host names (→ resource description) (See Fig. 2.5). The user profile is kept in the home host. Since public resources are to be used by other persons there should be a resource description, and part of which is in the resource directory. On the other hand, many cases of private resources do not require the possession of resource description as they are usually utilized by predefined users or predefined

applications. In many instances, private resources do not need to notify the whole system of information on whether or not the resource exists and on its location of the resource. Resources such as these also could give instruction on the location of resources by designating users' names, host names and resource names which would enable users to access by using NCC.

2.3.3 LIP (Local Information Processor)

LIP translates from expressions in common with the network into local expressions at each host. What should be taken into consideration are the five categories of translation as shown below.

1) data translation,
2) query translation,
3) data model translation,
4) command language translation, and
5) programming language translation.

LIP is generated at the instruction of CP.

2.4 Processing Expression and Translation

The function of CP is to translate users' requirements into the expression executable on the actual system by using NCC information (See Fig. 2.6). In this section we cover the generation process of the executable expression. The generation process is shown in Fig. 2.7.

In the generation process of the executable expression, let us consider the following four necessary forms of expression.

1) user language,
2) LPE (Logical Processing Expression),
3) WPE (Whole Processing Expression), and
4) executable expression.

The user language is a non-procedural language. Casual users can state their requirements to the user assistant by using the user language. LPE expresses resources by the generic resource name, a common concept corresponding to a certain resource, and expresses relationships between resources by the input and output to the process. WPE is an expression which is executable on the actual system and must express the whole processing. The executable expression is to express what has been described by WPE in the form of the most optimal schedule for each host computer.

Important points in resource integration are LPE, WPE and LPE-WPE (L-W) translation. User language is the interface to casual users, and is not taken up in this paper. The role of the executable expression is to allocate WPE to each host.

Hereinafter, we discuss LPE, WPE and the L-W translation.

2.4.1 LPE (Logical Processing Expression)

LPE is produced by the U-L translator with user language being used as input (See Fig. 2.8). It expresses generic resources, and logical and semantic relations between generic resources. As shown in Fig. 2.9, LPE is composed of the combination of process and data. Process and data in LPE are generic resources themselves. In other words, they are a common concept of the class or group of resources. A process communicates with another process through data collection (DC). A primitive unit of LPE, as shown in Fig. 2.10, is composed of a single process and n-multified ($n \geqq 1$) DCs. The process and DC are defined in terms of an access mode from the process and the data organization of DC. These relations do not appear in LPE. They become necessary in the process of translation from LPE into WPE.

2.4.2 WPE (Whole Processing Expression)

WPE is the executable expression as a whole on the actual system. As shown in Fig. 2.11, WPE is generated by the L-W translator by inputting LPE. In WPE actual resources on the system are allocated optimally, and the translation process can be automatically incorporated, if necessary.

The following points should be taken into consideration with WPE.

i) translation, and

ii) synchronization and concurrency between processes.

We have considered the PERT-like notation in expressing these things. (See Fig. 2.12)

Examples of the executable expression translated from WPE in Fig. 2.12 are shown in Fig. 2.13.

2.4.3 L-W Translator

The L-W translator generates WPE by using NCC information and inputting LPE. Translation is conducted in 1) description of resources and 2) the relationship between resources. In the translation of the description of resources the location of the actual resources is determined from the generic resource name in LPE by using the NCC resource directory. The resources in WPE are expressed by the resource description of the NCC resource directory. Some actual resources correspond to a generic resource name. Therefore, it is required to determine which actual resources are most optimal. The information (on cost, processing speed, resource status, etc.) necessary to determine the most optimal resource is in the resource description of NCC.

When translating the inter-resource relationship of LPE to WPE, required translation processes, LIPs, are generated by using the resource descriptions of each resource, and the synchronization and concurrency between processes are also taken into account.

3. Data Sharing Problems in WPE

In the actual system, WPE is an expression executable on the actual system. The translation and the synchronization and concurrency between processes in WPE are covered in Chapter 2. In this chapter, let us discuss the transport problems of data, namely, data sharing problems in the distributed resource sharing system. The heterogeneity of distributed resource sharing system and the importance of data as the resource make the transport function important.[4) 8) 11) 23)]

Relating to WPE, the relationship between actual resources is defined in terms of access mode to DC (Data Collection) and the data organization of DC. DC is composed of database, file, record and item.

The database mentioned in this chapter is the data managed by DBMS. In other words, access to database is offered by query language which is provided by DBMS. On the other hand, file is the data managed by file systems in OS of each host computer. The record and item explained here are included in the file mentioned above. Item is the smallest named unit of data stored in the file and record is a named collection of associated items.

In this chapter, following three points are considered.

1) DC location and identification
2) DC transfer and access
3) DC item translation

(1) is an explanation where the required DC is located in the system and how to gain access to it. Information relating to the location and identification of DC is kept partly in NCC (Section 3.1). The transfer and access of DC are considered in (2). DC as an object of transfer, here, is file (See Section 3.2). The effective DC item translation method by setting virtual attribute description is discussed in (3). (See Section 3.3)

3.1 DC Location and Identification

When accessing DC, it is required to know the presence of data and its location at first. In this section, the location and identification of DC are considered.

Corresponding to the fact that there exist two kinds of

resources, there are also two kinds of DC, private and public DCs. Answering to the two kinds of DC, various resource directories have been composed. The resource directory (DC directory) related with DC and resource description (DC description) are considered in the following.

3.1.1 NCC DC Directory

Prior to accessing to data, users are required to know the presence of needed DC and the host who has the DC. The query for DC location must be proceeded through database, file, record and item.

Public DC holds such a directory as indicated in Fig. 3.1 in NCC. NCC DC directory is data key → DC name, DC name → host → a part of DC description. Through this directory, NCC answers queries from users and CP (Coordinate Processor). The NCC DC directory covers such fields as data structure and access methods and DC status information in the DC description. In DC description, each host keeps information relating to security, which means that each service site has to protect its important information from being abusively accessed, intentionally or not. It is a point of compromise between the service site, which does not want to allow abusive access to its data, and users, who want to make use of the resource information in the system.

The private DC directory makes clear the location of private DC. It is managed in the user profile by the user. It is a directory of DC name → host. As mentioned in the private resource, by specifying user name, host name and DC name, the location of DC is available through NCC. DC which does not have DC description is transferable only among homogeneous hosts. Furthermore, in that time, making access is not also available by specifying only special items within the record.

3.1.2 DC Description (DCD)

DC description can be said as a kind of DC to maintain and manage information (meta data) concerning DC. And DC description is composed of the following three factors.

1) data definition,
2) cross reference, and
3) DC information

As for items of data definition, the following are taken into account.

Database name	record format
permission mode	block length
database owner	record length
file owner	record name

file organization item name
kinds of keys item attribute

Cross reference is a description considering permission relating to process, database, file, record and item as shown in Fig. 3.2. DC information is data needed to decide optimal resource relating to cost, speed and status.

In order to obtain information including DC description, the following access methods are considered:

1) on-line query facility,
2) on-line update facility, and
3) query facility by key word.

3.2 DC Transfer and Access

In this section, the file is considered as DC. As for database, it is considered that the data or group data is available through query language supported by DBMS. As a result problems concerning database are making translation of local query language related with query language and each database. In this sense, the database access can be considered as problems of DC access.

The translation of file organizations as DC transfer is mentioned as follows. Taking into consideration the access of record as DC access, access is available through common command within the distributed resource sharing system.

3.2.1 File Transfer

In file transfer, the translation of file organizations is explained (See Fig. 3.3).

Concerning organizations translation, (1) one-way (source $\rightarrow$ target) and (2) round-trip (source $\not\rightleftarrows$ target) are considered. In (2), the possibility of restoration toward the initial file when re-translation is made for file organization, using the transferred file for the target file organization through the same translator, is considered. As file organization, sequential organization (S), partitioned sequential organization (PO), indexed sequential organization (IS) and direct organization (D) are considered.

Basically, we are able to conclude that the translation among all organizations is possible from one-way and round-trip points. We summarize the results of problems in organization translation as follows.

(1) As for IS and D, the correspondence of key in IS and address in D is needed. (2) The translation is possible concerning the PO by corresponding one member to one file in other organization. (3) The translation relating to D, is carried out including writing dummy records.

In the translation among file organizations, the

correspondence in the identification method of records within the file is an important problem. There are two methods concerning the identification of the file -- by accessing ordered records in turn and by examining address information such as the key for identification of records. The identification methods are shown in Table 3.1.

3.2.2 DC Access

As DC access, we take up the access of record level to the file. Fig. 3.4 shows DC access. In DC access, request to the translator is processed by using a common command within the distributed resource sharing system. The translator translates request accessible to file. The response from file is translated to the common response through the translator and proceeded to the process. In other words, the translation in the access of record level is the translation of the common access command to the local access command in file, and the translation of local response to common response.

As for the common command, the following are needed:

(1) open,
(2) access (READ and WRITE), and
(3) close.

At the time of accessing file, the target file is accessible in the process without considering the organization of the file. Process can access file normally by an identifier relating to the needed record. The signification of the identifier in the process is shown Table 3.2. However, at the time of opening file for the new registration, the process is required to get inherent information of the file including the organization of file needed and its capacity.

3.3 DC Item Translation

In this section, we discuss DC item translation. The virtual attribute description is used here for the sake of an effective translation.

Where the data value in the real world is expressed in a computer, descriptions that are convenient to the data processing are used. This procedure makes data have various attributes. An attribute means binary number or decimal number, fixed point number or floating point number in item and the file organization in file and so on. Concerning the data description in the real world, there are two descriptions --a computer description and a computer internal description. The computer description means the virtual description of data on the basis of the common concept of "computer." The data is able to have some computer descriptions. The computer internal description is the data description within respective

computers, and the same data have their different descriptions on their respective computers.

The translation requires the following two functions (See Fig. 3.5).

(1) The translation among computer internal descriptions which have the same data attribute.

(2) The translation among different data attributes.

The collection of items is considered as record and the collection of records is considered as file as mentioned before. The translation is required to consider three levels relating to the item, record and file. The data attribute in computer description is summarized in Table 3.3.

The translation of items means to translate data in the real world (computer description D^1) from internal description of computer A (D_A^1) to internal description of computer B (D_B^1). In other words, it indicates $D_A^1 \rightarrow D_B^1$ (See Fig. 3.6).

As the translation processes, the following three methods are explained:

(1) $D_A^1 \rightarrow D_B^1$ (Direct translation)

(2) $D_A^1 \rightarrow D^1 \rightarrow D_B^1$

(3) $D_A^1 \rightarrow D^{1'} \rightarrow D_B^1$

(1) is the direct translation from D_A^1 to D_B^1. And (2) supposes virtual computer description of D^1. It is translated twice: from D_A^1 to D^1 ($D_A^1 \rightarrow D^1$) and from D^1 to D_B^1 ($D^1 \rightarrow D_B^1$). Then (3) supposes virtual computer description $D^{1'}$ as (2). $D^{1'}$ is the virtual description most accessible to D_A^1. It is translated to D_B^1 through $D^{1'}$. In that time, the translations from D_A^1 to $D^{1'}$ are not processed in most cases.

Firstly, the number of translations is considered on the three methods mentioned above. Its results are shown in Table 3.4. Here, m indicates the number of computers which have different internal descriptions.

Secondly, where n (n > 1) computer descriptions exist, its result is shown in Table 3.5 and Fig. 3.7. Here, n indicates the number of computer descriptions and m indicates the number of computers which have different internal descriptions.

As a result, the following points are concluded.

(1) The translation method of data item adopts the virtual internal description method. (2) In order to decrease the number of translations, attributes are translated while the data item is not.

4. Summary and Discussions

The user assistant for non-procedural resource integration is a system which is accessible by integrating heterogeneous resources through non-procedural language. Here, a summary of this system is mentioned and the data sharing problem in question is discussed.

The description of resource is one of the most important problems in the resource integration. Concerning the description of resources, two concepts were introduced--(1) explanatory description and (2) attribute description. Explanatory description is a common concept relating to classes or groups of resources. On the other hand, attribute description is a detailed and precise description concerning actual resources. By using an explanatory description of resources, actual heterogeneous resources are accessible in one united body. As for the descriptions corresponding to the two descriptions of resources, LPE for explanatory description and WPE for attribute description are considered and the translation of LPE to WPE is discussed. At the time of considering the explanatory description, how to group actual resources and how to classify each groups are important problems yet to be solved.

Regarding user language and U-L translator, these problems were not discussed here. One of the purposes of non-procedural resource integration is the offering of high-level and non-procedural interface for the resource integration for the casual user. Hereafter, the investigation of interface for casual users and the investigation of translation from user language to formal LPE[7) 21)] are required to be studied.

Concerning the data sharing problem, the following three points--(1) DC location and identification, (2) DC transfer and access and (3) DC item translation--were discussed. In the discussion, by NCC DC directories, virtualization of resources was proceeded. In the translation of DC items, by setting virtual attribute description, an effective translation method is cleared.

With database as DC, the translation of query is a principal problem. The query is able to correspond to the access of each internal DCs. Accordingly, there should be a common query language at this stage.

As resources, we consider both public and private ones. Regarding use of the whole system, NCC manages the resources which have no limitation in use as public ones. On the other hand, private ones which have some regulations are managed by users at the local level.

From now on, resources which own security, in particular, DC, will become more important. The security and integrity in DC, database, file, record and item are required to be considered at each level. It is required to consider the correspondence between access mode (READ, WRITE, etc.) at each level, and users or user groups who are authorized to access.

Reference

1) Anderson, R.H., and Gilogly, J.J., "Rand Intelligent Terminal Agent (RITA): Design Philosophy," R-1809-ARPA, The Rand Corp., Santa Monica, Jan. 1976.

2) Anderson, R.H., et al., "RITA Reference Manual," R-1808-ARPA, The Rand Corp., Santa Monica, Sept. 1977.

3) Benoit, J.W., Grif-Webster, E., "REX: A Resource Location and Acquisition Service for the ARPA Computer Network," MTP-387, MITRE Corp., Jan. 1974.

4) Briss, E.W., and Fry, J.B., "Generalized Software for Translating Data," Proc. NCC'76, May 1976, pp. 889-897.

5) Date, C.J., "An Introduction to Database Systems," 2nd Edition, Addison-Wesley, 1977.

6) Deppe, M.E., and Fry, J.P., "Distributed Database: A Summary of Research," Computer Networks, Vol. 1, No. 2, September 1976, pp. 130-138.

7) Hendrix, G.G., Sacerdoti, E.D., Sagalowicz, D., and Sbcum, J., "Developing a National Language Interface to Complex Data," Proc. 3rd International Conference on VLDB (Part II), Tokyo, Oct. 1977, pp. 37-58.

8) Housel, B.C., "Unified Approach to Program and Data Conversion," Proc. 3rd International Conference on VLDB, Tokyo, Oct. 1977, pp. 327-335.

9) IRIA, "Presentation du Cyclades," IRIA, Fevrier, 1974, p. 10.

10) Marcus, R.S., and Reintjes, J.F., "The Networking of Interactive Bibliographic Retrieval Systems," ESL-R-656, Electronic System Laboratory, MIT, March 1976, p. 174.

11) Merten, A.G., and Fry, J.P., "A Data Description Language Approach to File Translation," Proc. ACM SIGMOD Workshop on Data Description, Access, and Control, Ann Arbor, May 1974, pp. 191-205.

12) Morris, P., and Sagalowicz, D., "Managing Network Access to a Distributed Database," Proc. 2nd Berkley Workshop on Distributed Data Management and Computer Networks, May 1977, pp. 58-67.

13) Organik, E.J., "The MULTICS System: An Examination of its Structure,"
MIT Press, 1972.

14) Palmer, I., "Distributed Data Base," On-Line Database, Infotech, 1977, pp. 203-226.

15) Peebles, R.W., "Design Consideration for a Distributed Data Access System," Ph.D. Dissertation, Wharton School, University of Pennsylvania, Philadelphia, PA. 19174, May 1973.

16) Pyke, Jr., T.N., "Network Access Techniques: Some Recent Development,"
Proc. Third Annual Texas Conference, Oct. 1974, pp. 2-2-1 - 2-2-3.

17) Rosenthal, R., "A Review of Network Access Techniques with a Case Study: The Network Access Machine," NBS Technical Note 917, July 1976.

18) Rosenthal, R., "Network Access Techniques - A Review,"
Proc. NCC'76, Vol. 45, May 1976, pp. 495-499.

19) Rothnie, J.B., and Goodman, N., "A Survey of Research and Development in Distributed Management,"
Proc. 3rd International Conference on VLDB, Tokyo, October 1977, pp. 48-62.

20) Rothnie, J.B., and Goodman, N., "An Overview of the Preliminary Design of SDD-1: A System for Distributed Databases,"
Proc. 2nd Berkeley Workshop on Distributed Data Management and Computer Networks, May 1977, pp. 39-57.

21) Sagalowicz, D., "IDA: An Intelligent Data Access Program,"
Proc. 3rd International Conference on VLDB, Tokyo, Oct. 1977, pp. 293-302.

22) Severance, D.G., "Some Generalized Modeling Structures for Use in Design of File Organization," Ph.D. Dissertation, University of Michigan, 1972.

23) Shu, N.C., Housel, B.C., and Lum, V.Y., "CONVERT: A High Level Translation Definition Language for Data Conversion,"
CACM, Vol. 18, No. 10, Oct. 1975, pp. 557-567.

24) Thomas, R.H., "A Resource Sharing Executive for the ARPANET,"
Proc. NCC'73, May 1973, pp. 155-163.

25) "EURONET On-Line Information Network," Commission of the European Communities, 1977, p. 13.

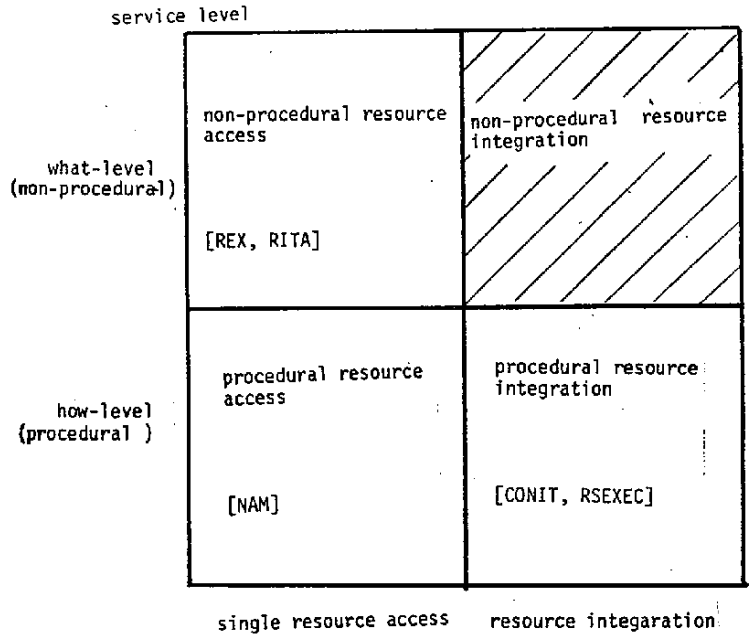


Fig. 2.1. The Classification of User Assistants.

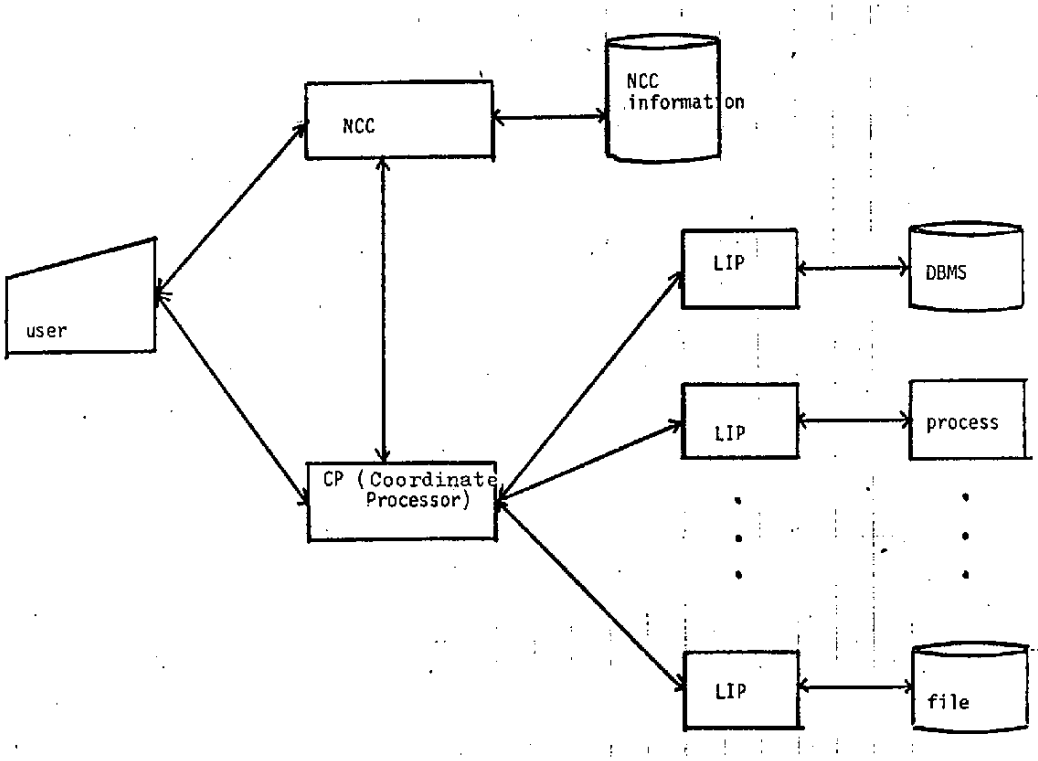


Fig. 2-2. The Logical Structure of User Assistant

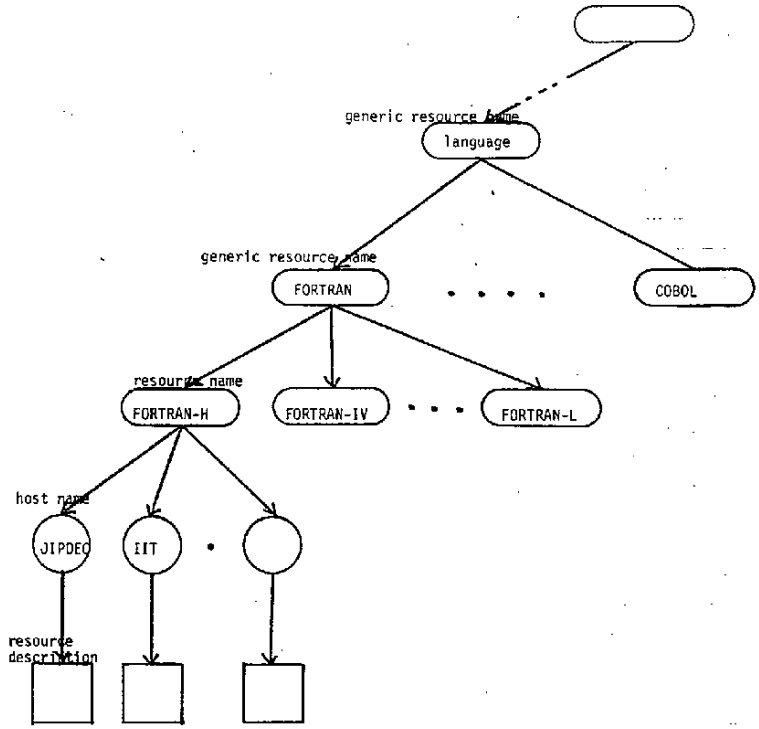


Fig. 2-3. NCC Resource Directory

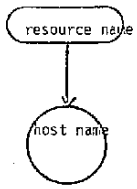


Fig. 2-4. Private Resource Directory in User Profile

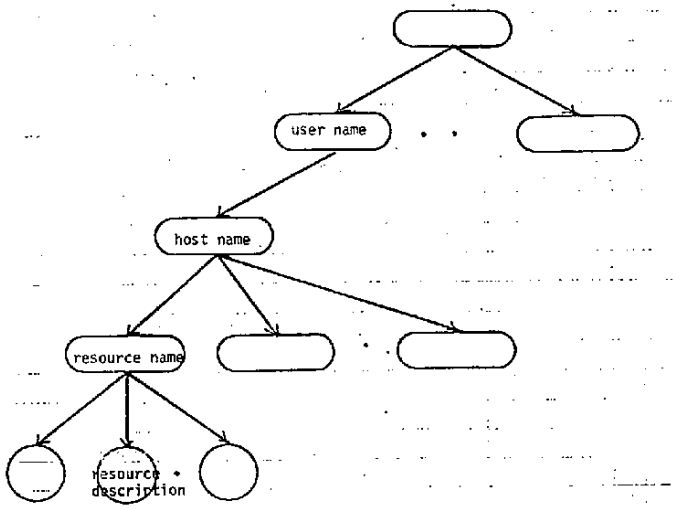


Fig. 2-5. Private Resource Directory in NCC

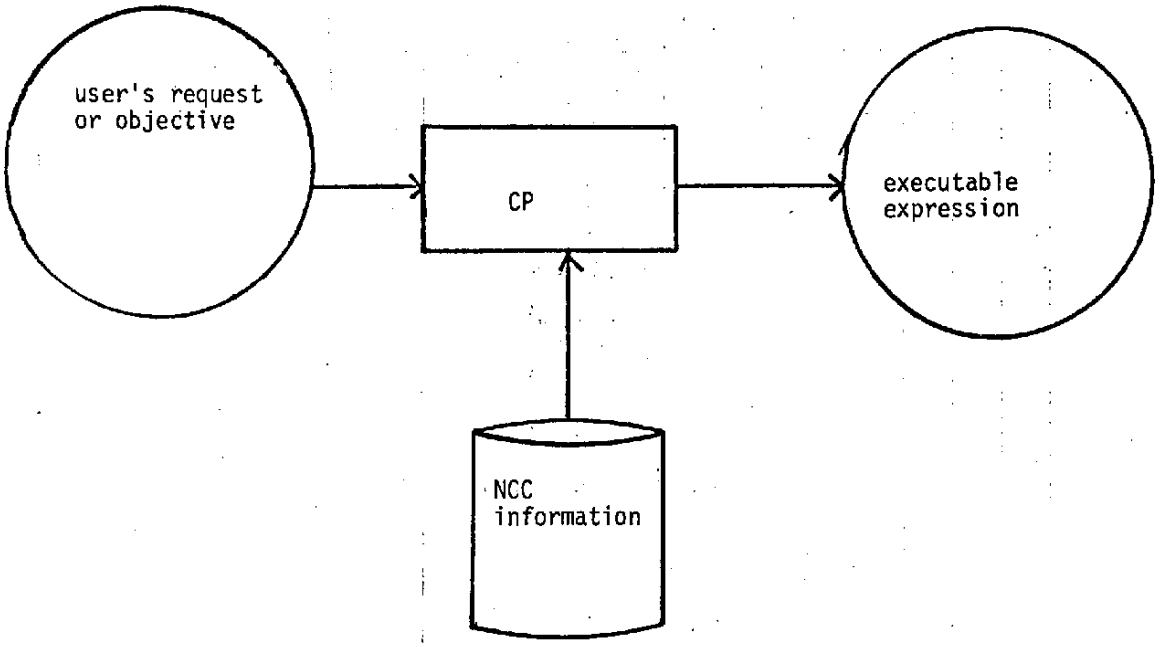


Fig. 2-6. The Translation of user's request to executable expression

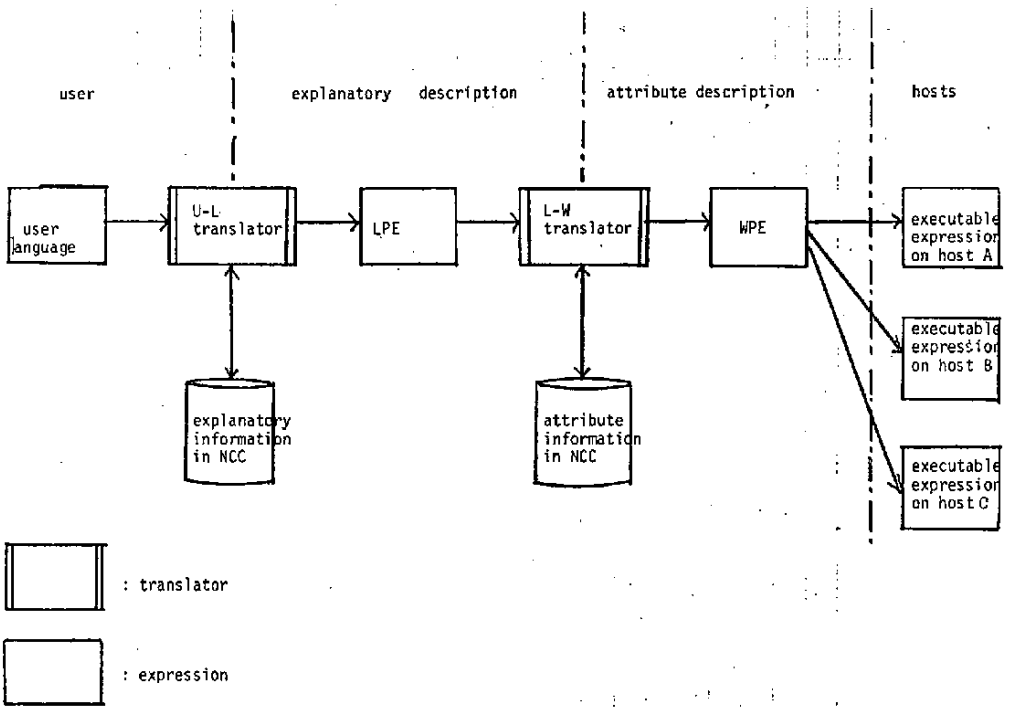


Fig. 2-7. The Translation of Expressions.

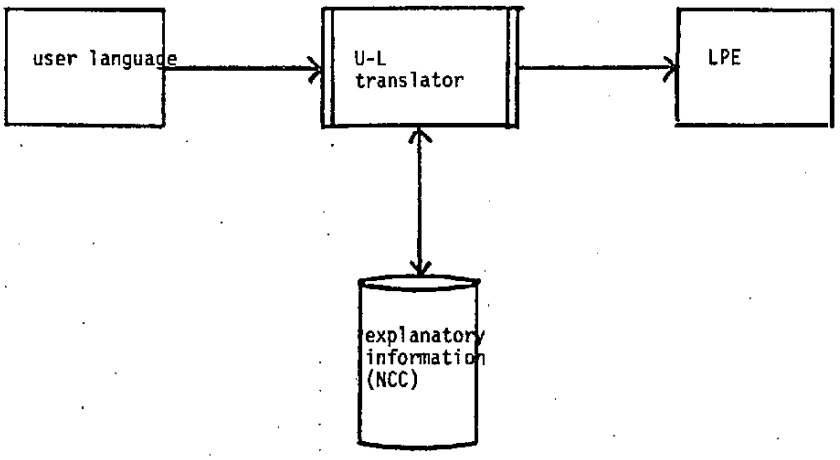


Fig. 2-8. U-L Translator

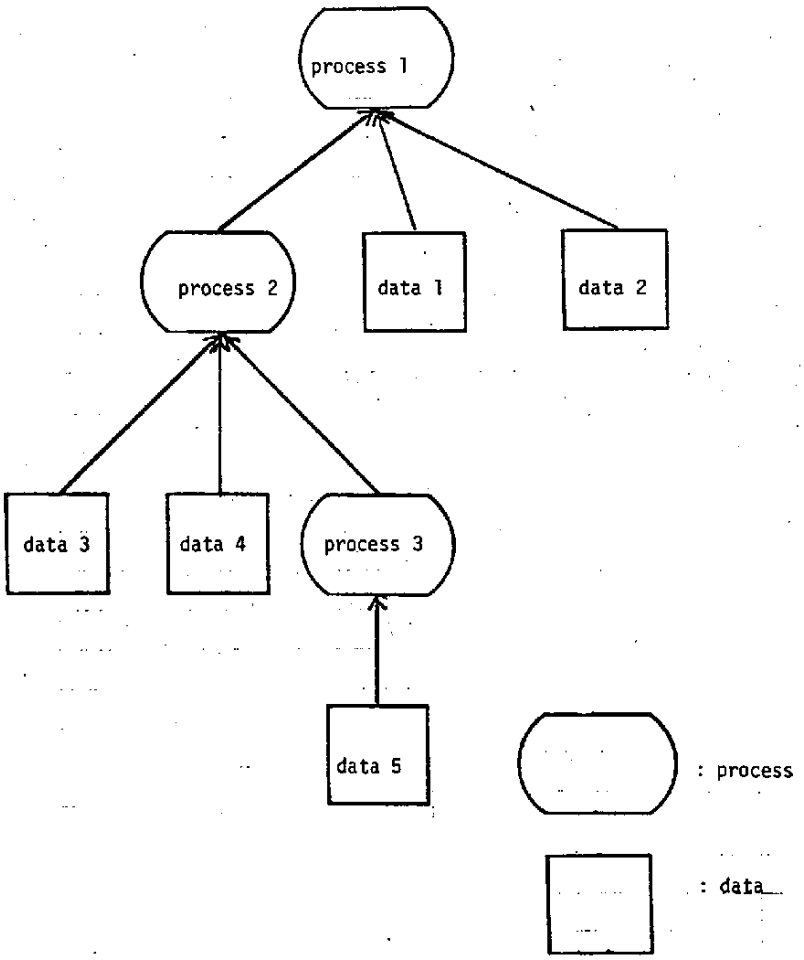


Fig. 2.9. The Example of LPE.

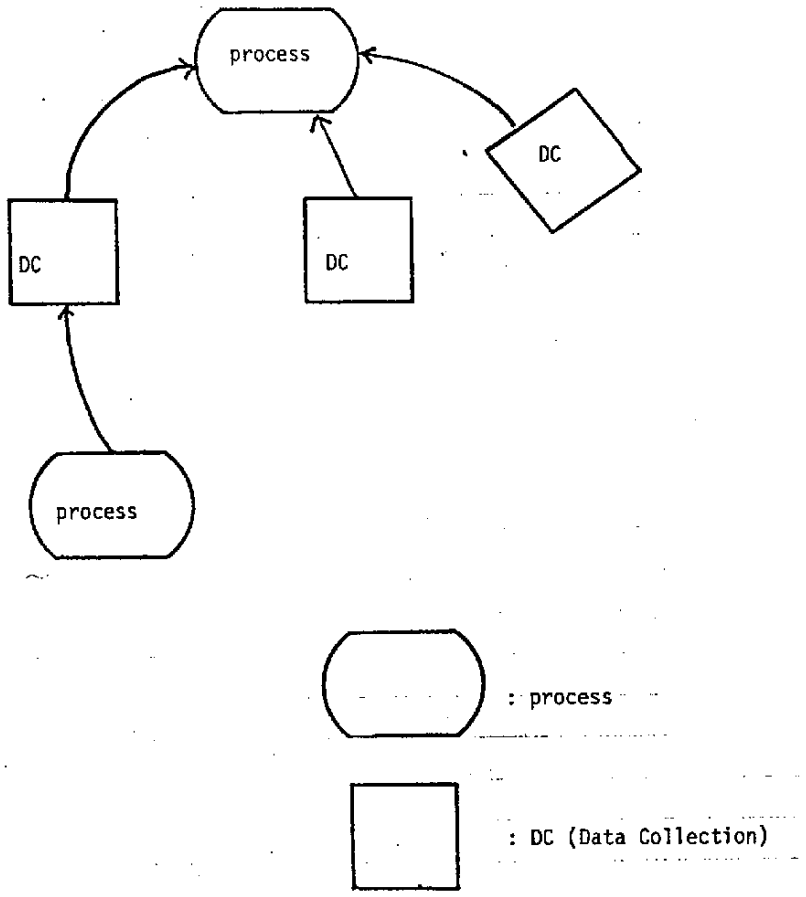


Fig. 2-10. Process-DC Relation.

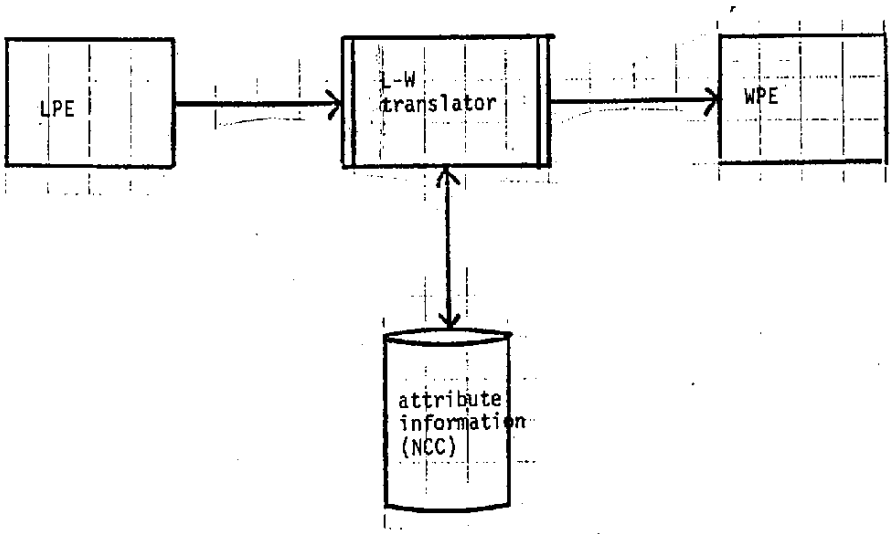


Fig. 2-11. L-W Translator

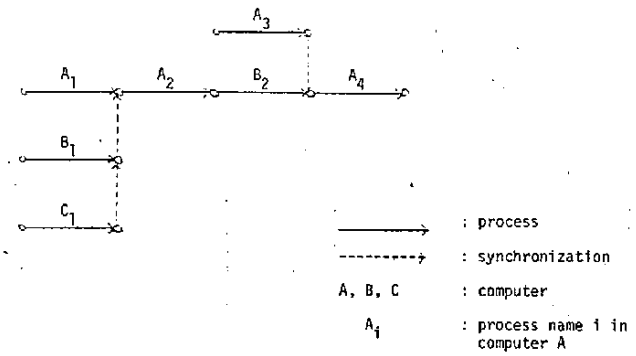


Fig. 2.12 The Example of WPE

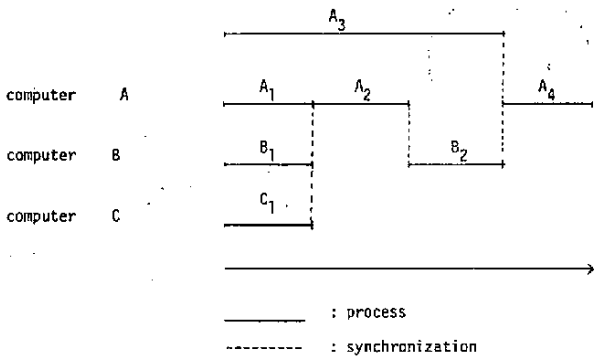


Fig. 2.13. The Example of Executable Processing Expression

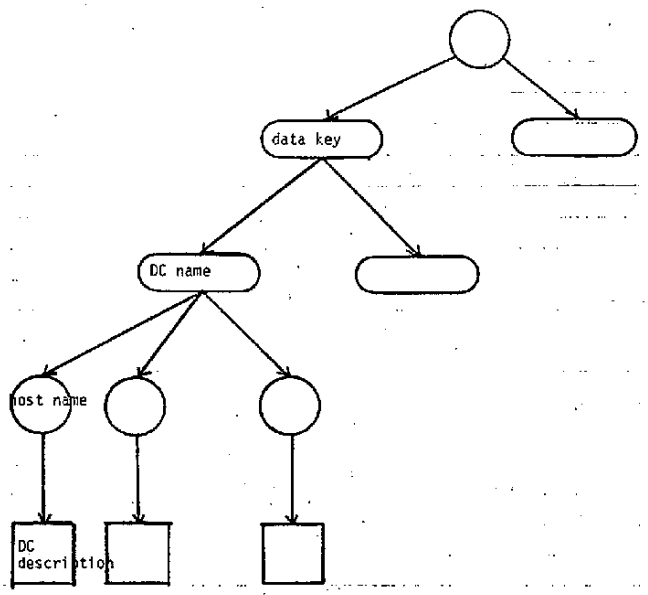


Fig. 3-1. NCC DC Directory

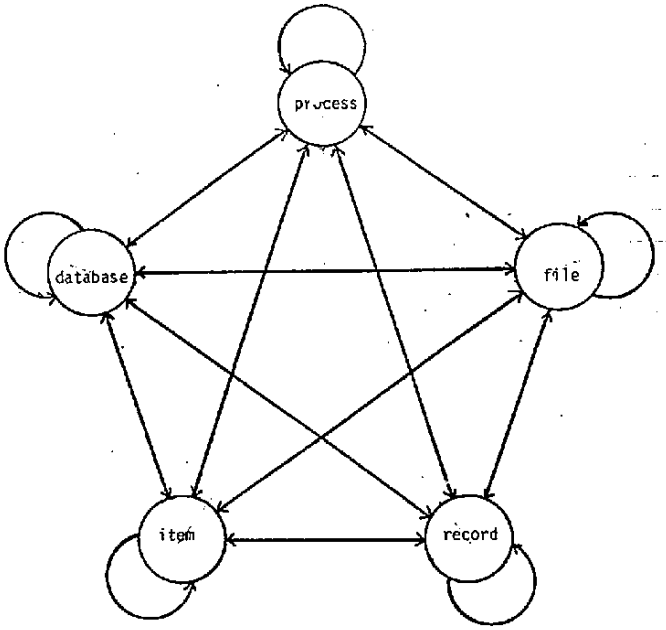


Fig. 3-2. Cross Reference

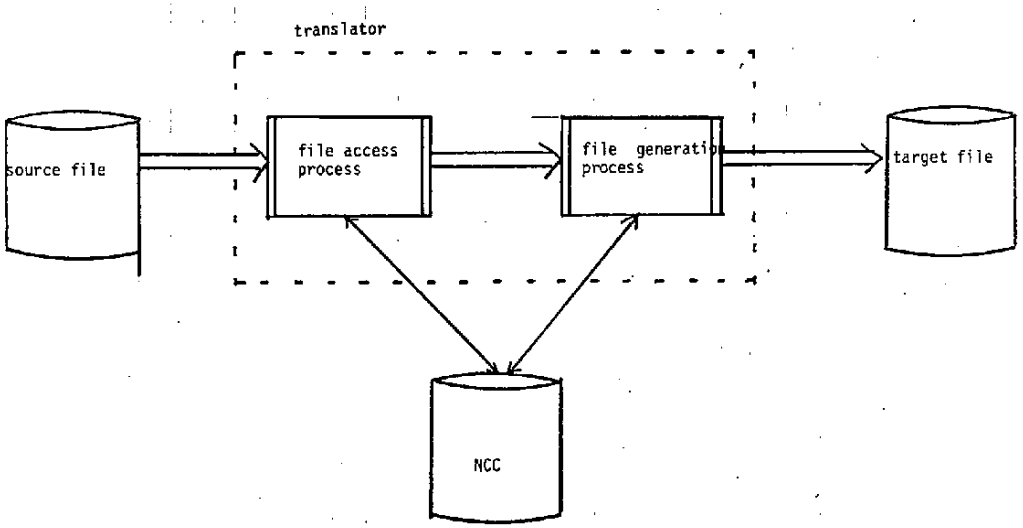


Fig. 3-3. File Transfer

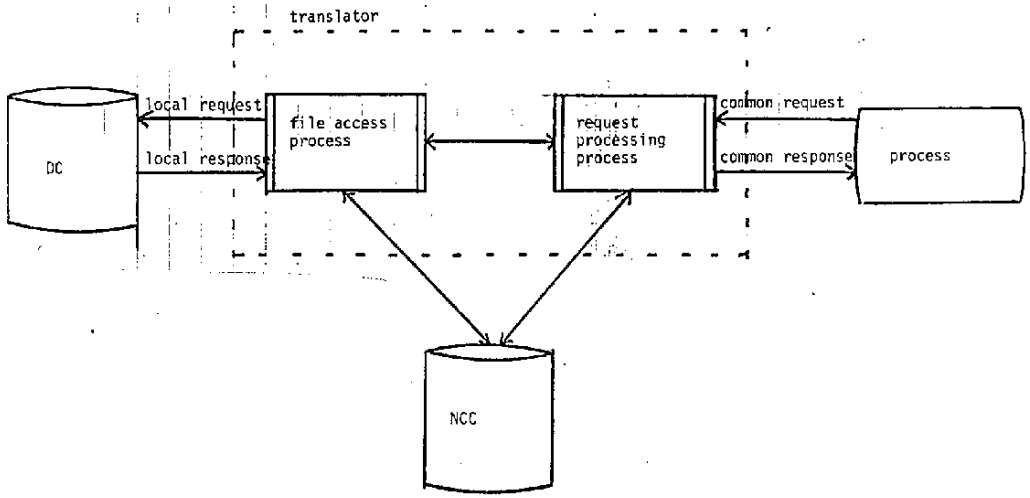


Fig. 3-4. DC Access

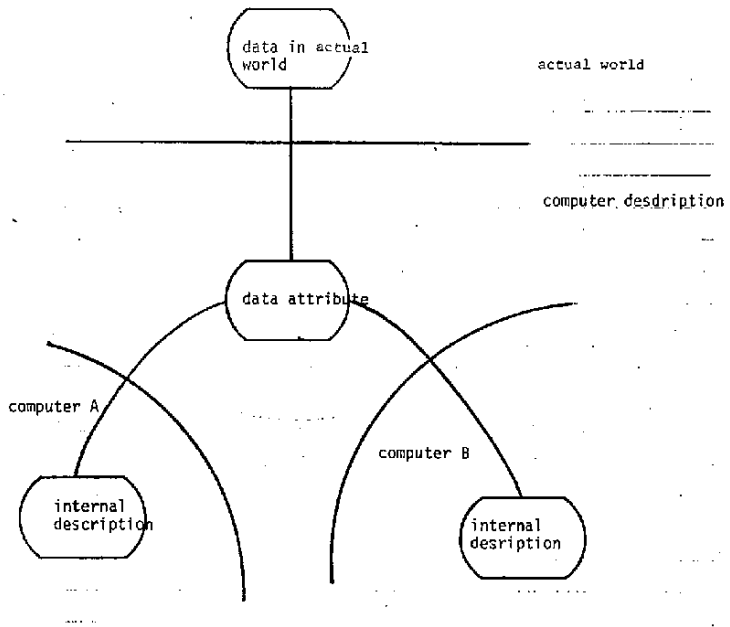


Fig. 3-5. Data Description

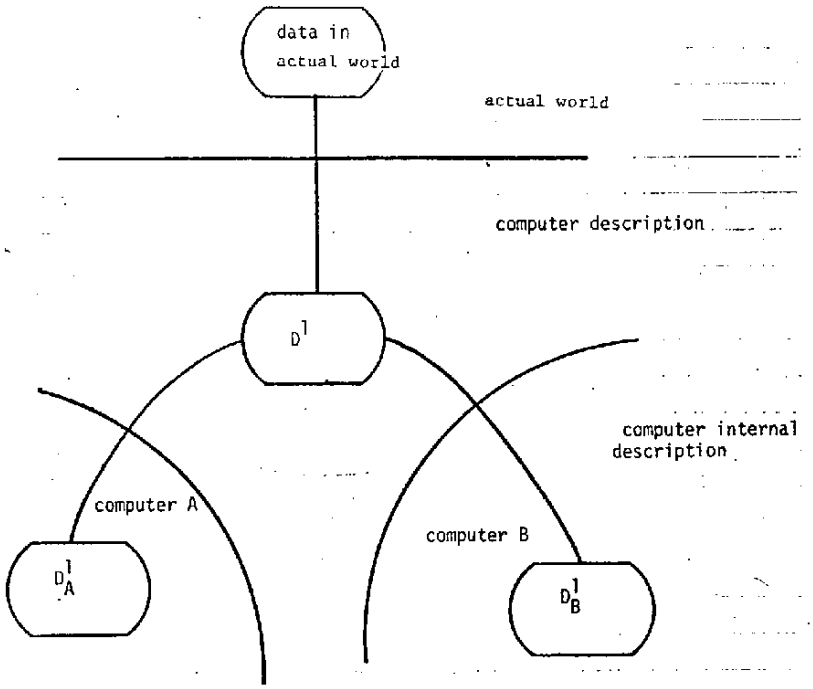


Fig. 3-6. Data Description

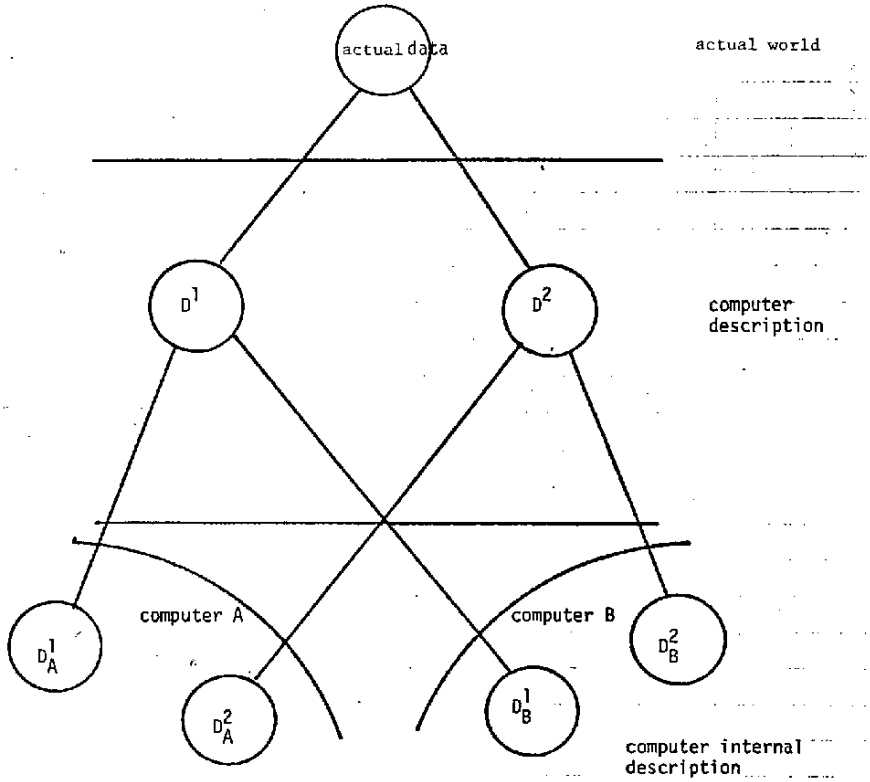


Fig. 3-7. Data Description in the case that there exist two computer descriptions.

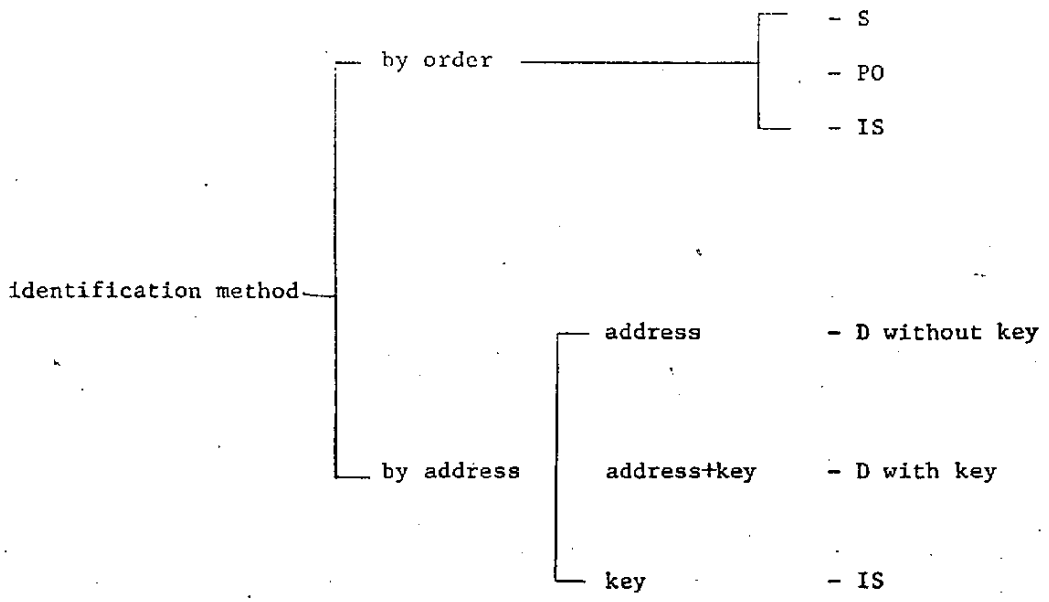


Table 3-1. The Identification Method of Record in File.

file organization	identifier
sequential	nothing
partitioned sequential	member name
indexed sequential	key
direct without key	address
direct with key	address + key

Table 3-2. The Identification in File.

translation process	the number of translations	the number of translation processes
1) $D_A^1 \longrightarrow D_B^1$	normal 1 max 1	$2m^2n^2$
2) $D_A^1 \rightarrow D^1 \rightarrow D_B^1$	normal 2 max 2	$2m^2n^2$
3) $D_A^1 \rightarrow D^{1'} \rightarrow D_B^1$	normal 1 max 2	$2n(n+1)m$
4) $D_A^1 \longrightarrow D_B^2$	normal 1 max 1	$2m^2n^2$
5) $D_A^1 \rightarrow D^1 \rightarrow D_B^2$	normal 2 max 2	$2m^2n^2$
6) $D_A^1 \rightarrow D^2 \rightarrow D_B^2$	normal 2 max 2	$2m^2n^2$
7) $D_A^1 \rightarrow D^1 \rightarrow D^2 \rightarrow D_B^2$	normal 3 max 3	$2n(n+1)m$
8) $D_A^1 \rightarrow D^{1'} \rightarrow D^{2'} \rightarrow D_B^2$	normal 1 max 3	$2n(n+1)m$

Table 3-5. The Number of Translations.

DC element	file	record	item
data attribute in computer description world	organization	item organization	arithmetic base binary or decimal scale mode precision
	key to record	key to item	string string attribute number, bit, or character length

Table 3-3. Data Attribute.

translation process	the number of translations	the number of translation processes
1) $D_A^1 \dashrightarrow D_B^1$	normal 1 max 1	2m
2) $D_A^1 \rightarrow D^1 \rightarrow D_B^1$	normal 2 max 2	2
3) $D_A^1 \rightarrow D^{1'} \rightarrow D_B^1$	normal 1 max 2	2

Table 3-4. The Number of Translations.

III. 日本語情報処理技術の研究開発

1. 総　　論

1. 1. 研究の背景と目的

従来，わが国の情報処理分野では、日本語情報の取扱いを、カタカナもしくはローマ字（英字）によって代用してきた。そのため、コンピュータを利用した経営情報システムにおいて、意思決定に必要な情報は数値情報を中心にカタカナやローマ字表現で情報伝達され、意味にあいまいさが伴なって適確な情報交換を行なうことができなかった。それればかりか、我々日本人にとっては、漢字かな混り文でないがために読みずらさも加わって、情報の価値、理解度、認識度という観点からも満足な情報とはいえなかった。

一方、日本語情報を処理するための入出力装置、いわゆる漢字入出力装置の開発は、コンピュータ周辺・端末装置の急速な発展と機能向上の中で、あいかわらず低速度の進行状態である。そこにはあらゆる面で文字種の多さによる原因があった。普及の最重点的要素である入出力装置の価格が、従来の英数字カナ文字用に比べてあまりにも高価であったこと、漢字入力・データ作成に費用が多くかかること、などすべてが多字種によるものである。

日本語を構成している文字は、ひらがなやカタカナをはじめ、新旧漢字・俗字などを含めると最大５万字種近くもあると言われている。もっとも一般的な制定漢字として当用漢字があるが、この制限枠内の漢字でさえも1,８５０字種（昭和５２年１月に国語審議会が発表した改定案は1,９００字）である。また、新聞・雑誌に使用されている漢字の種類は、およそ3,０００字種前後といわれ、日本人の姓名に至っては約１万字種を使用しないと書き現わすことができない。

このように、欧米諸国での使用文字種が数十種というのに比較して、日本ではその数十倍から数百倍の文字種を扱わなければならないというギャップがある。このことは、コンピュータ処理において文字コード・配列順の問題が生起して、入力側では入力キーボードへの文字配列の問題が、出力側では文字発生機構の複雑化等の問題が派生してくる。こうした日本語情報の取扱い上の問題がある中で、日本語情報（漢字）処理の先駆的役割りを果したのが、新聞・通信業界であった。当業界では昭和３４年に文字種・文字コードの制定を行なって（CO－5.9コード）、新聞記事・原稿の通信印刷の手段に漢字入出力機器を使用するようになった。この当時開発されたのが漢字テレタイプライタ（漢テレ）であり、２千数百の漢字を一面のキーボード上に配置したフルキーボード型の入力機器が用いられた。

その後、日本の経済成長にともなってコンピュータの普及と利用分野の拡大もあいまって、コンピュータからの出力に漢字かな混り文字を印字したいという要望が高まってきた。新聞・通信業界に続いて出版・印刷業界で漢字入出力機器の導入気運が高まり、次第に官公庁関係機関が導火線となって民間企業へと普及するようになった。とくに姓名や住所、会社名などのあて名書きに漢字化が急速に進められた。

当初は、漢字入出力機器の製作・販売は数メーカに限られていたが、大手コンピュータ・メ

ーカをはじめ周辺機器メーカもあいついで漢字処理システム機器の開発に参入してきた。その結果、出力装置に関しては従来のラインプリンタ以上の機能を有した、高速・高精度の装置が開発されて、ユーザの利用目的に合った選択ができるようになってきた。しかし一方の入力装置に至っては、現在でも相変らずフルキー方式が主流を占めている。両手を使った2打けん式の入力操作から、和文タイプ型、タブレット型、ペンタッチ型へと操作が容易になって素人でも扱えるようになり、記録媒体も紙テープから、カセットテープ、フロッピー・ディスクなどへと後処理がしやすいように変ってきたが、数千字の漢字が配置された盤面から目的とする漢字を探す方法はほとんど変わっていない。また操作性の向上を目指して、ローマ字入力・カタカナ入力方式も研究開発されてきたが同音異義語（同音異字）、分ち書きの問題をかかえて実用化されるまでに至っていない。

このような漢字入出力装置の開発動向に加え、それぞれの入出力方式に付随した制御ソフトウェア、各種変換プログラムなどに関する研究開発も進められている。入力方式ではカタカナやローマ字入力によって操作性を向上させようとすれば、コンピュータへの依存度が高い変換ソフトウェアが必要である。また出力方式では漢字パターンの記憶・アクセス（復元）方法、文字情報の表示・出力方法などにソフトウェア制御部分が多い。そのため、制御プロセッサにミニ・コンピュータ規模の処理機能が要求されていた。とくに出力装置・表示装置には大規模な文字発生装置が必要となって、ほとんどのものがミニ・コンピュータを制御装置に内蔵していた。これは明らかに高価格化をうながす要因であり、ユーザにとってはニーズがありながらも漢字化をすすめることができない大きな障害となっている。

以上のような日本語情報処理技術の研究背景からして、今後の日本語情報処理の普及拡大をはかるには、さらに安価な漢字処理装置の開発が望まれる。そして日本語情報処理の適用分野も、そのニーズから広範囲の需要が見込まれており、利用目的に応じた入出力装置の出現が期待されている。

新聞・出版・印刷関係の業務では、高品質文字を扱う電算写植システムとしての利用が主であり、そこでは多少の高価格化（もちろん低価格にこしたことはない）よりも多彩な機能と高速化が要求されて、脱鉛化、CTS（Cold Type System）化、などの省力化を目的としている。そのため出力装置の適用方向は、安価・簡易化をめざすものと機能向上・高速化をめざすものとに大別されよう。しかし入力装置への期待はどの分野で利用しようとも、簡易入力が最大の課題である。

このようなことから当事業で研究開発を目指すことは、コンピュータ端末装置としての日本語情報処理アプリケーション向け入出力方式の確立をはかることである。コンピュータ周辺端末装置は、これまでとくに漢字化が遅れている部分であり、ここにも高価格な漢字入出力装置が大きな要因である。そのため当研究開発では、日本語情報処理端末の最適な入出力構成方法と、その入出力制御方式の確立をめざし、ユーザにとって操作のしやすい方式と満足のいく出力結果が得られるように検討を進めていく計画である。

なお、当事業「分散型リソース処理技術の研究開発」においては、わが国のリソース・シェアリング・ネットワークの特色の1つとして日本語情報の処理機能が必要不可欠なことである。とくに今後のネットワーク・システムには、データベースが重要なリソースとして期待され、そこには日本語情報のデータベースが構築されることになる。この日本語情報のリソースを利用するためには、日本語情報を処理するソフトウェアと、入出力を行なう日本語情報処理用ハードウェアによって構成されなければならない。

そのためネットワーク端末として日本語情報を扱うことができる装置、いわゆる日本語端末を構築して、日本語情報リソースの共有化をはかっていくことが主要な目標である。

1.2 本部の概要

第Ⅲ部は、「分散型リソース処理技術の研究開発」の一環として推進している、日本語情報処理技術に関する研究開発についてとりまとめたものである。

第Ⅲ部の構成は、5つの章からなっており、それぞれ次のような内容について記述している。

第1章「総論」では、当研究開発の背景とその目的について述べている。

第2章「日本語情報処理技術の調査分析」の第1節(2.1)では、日本語情報処理技術の現状として、漢字入出力装置の調査分析と適用分野の事例紹介を行なっている。第2節(2.2)では、漢字入出力装置をとくにオンライン処理という観点からその現状分析を行なうとともに、オンライン漢字処理の事例を紹介している。第3節(2.3)では、日本語情報処理に応用される関連技術について、その動向を述べている。

第3章「日本語情報処理技術の考察」では、第2章の現状分析結果から、日本語情報処理全般の問題点、入力に関する問題点考察、出力に関する問題点考察、についてそれぞれ述べている。

第4章「日本語端末の構成方法と処理機能」では、第2章の調査分析と第3章の問題点考察の中から、とくにコンピュータ端末としての日本語情報処理に視点をあてた、日本語端末モデルの構築を目指して、その入出力構成と処理機能の最適化について検討した結果を述べている。

第5章「日本語端末の課題」では、日本語情報処理全般の問題点および日本語端末の検討・考察から、今後の研究課題と展望に対して述べている。

2. 日本語情報処理技術の調査分析

2.1 日本語情報処理の現状

2.1.1 漢字入出力装置の分析

(1) 漢字入力装置

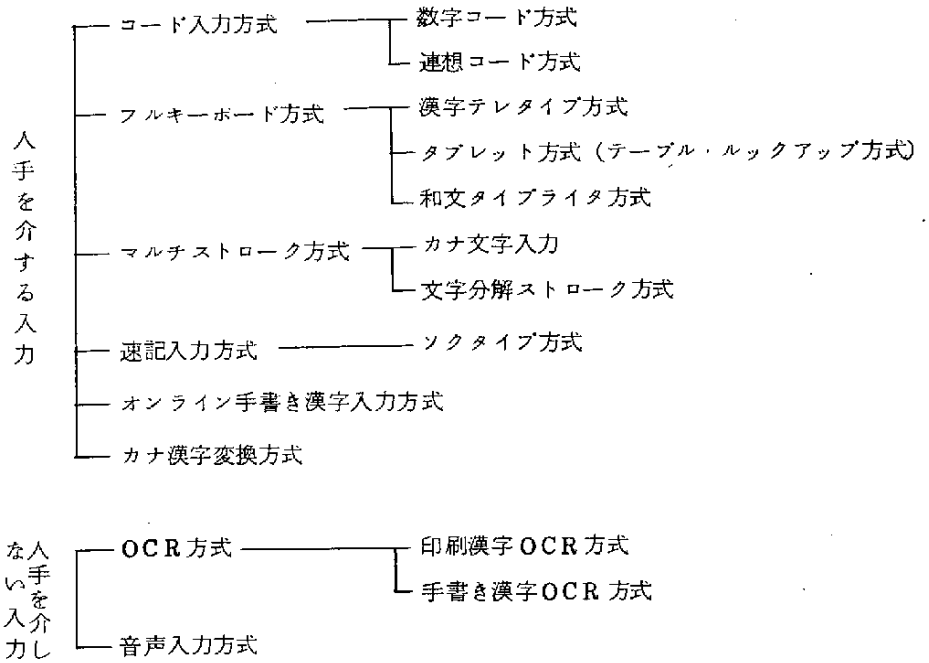
漢字をコンピュータで処理しようとした場合、もっとも問題となることは漢字の文字種が多いことである。しかも、適用分野によって必要な文字種にバラツキがあり、また、漢字コードや鍵盤上の配列順序が各メーカごとに異なったまま使用されてきた。このうち後者については、ようやく昭和５３年１月１日付をもって「情報交換用漢字符号系」の日本工業規格（ＪＩＳ）が公示されるに至った。しかしながら、漢字入力の最大の問題は、何といっても、数千文字の中から入力したい文字をどのように選択するかにある。漢字入力を考える場合には、一般のデータ入力と同様あるいはそれ以上に、誰が使っても同じような速度で入力でき、しかも、簡単な操作で扱える装置が望まれる。。一般の事務員がそのままオペレータとして使えるものとして欧米では英文タイプライタがあるが、日本ではこのようなものは見あたらない。このように、漢字入力装置として決定的なものが存在しないため、現在、各メーカでそれぞれ工夫をこらした各種の入力方式が発表されている。以下に入力方式の分類を試み、各方式の特徴を概述する。

（１－１） 漢字入力方式の分類

漢字入力方式の分類法には、そのとらえる観点の違いから種々の方法があるが、ここで述べる分類はあくまでその１つの見方を示したのにすぎない。

漢字入力装置をみてみると、まず、人手を介して行なう間接入力と人手を介しない直接入力とに分けることができる。人手を介在するとは、文字１字１字が人間の手によって入力操作されることである。人手を介する入力方式には数字コード方式、フルキーボード方式、マルチストローク方式、速記入力方式、オンライン手書き漢字入力方式、カナ漢字変換方式などがある。人手を介しない入力にはＯＣＲ方式（印刷漢字ＯＣＲ方式、手書き漢字ＯＣＲ方式）、音声入力方式などがある。

表２－１　漢字入力方式の分類



人手を介する入力

①　コード入力方式

a）数字コード方式

前もって準備した漢字と数字コードとの対応表を用いて、人手により漢字を１０進コードに変換してテンキーから入力する方式。現在では主に外字処理（盤面上にない文字の入力）に使われるほかはあまり使われていない。しかし、この方式では特殊な装置を用いないでテンキーだけで入力を行なうことができるため、低コストの入力を望む場合に、一部で使われている。この方式のコードとして漢和字典「新字源」の各漢字につけられている4桁の数字コードが使われることが多い。

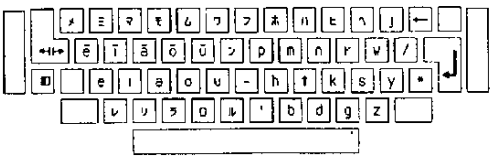
b）連想コード方式

ある漢字について、オペレータがその漢字を連想しうるようなコードをカナや英字で表現して入力する方式である。この方式では連想コードを文字種分記憶しなければならないため、オペレータは、かなりの熟練を必要とする。この方式の例としては、ラインプット（ラインプット社）、**KANTEX**（カンテック社）、**K.I.S.**（九段コンピュータサービス）方式がある。また、この方式の1種として、「へん」、「つくり」などの漢字の構成要素に数字コードをあてはめて入力するワング方式（ワング・コンピュータ社）もある。これら連想コード方式では、１文字を２タッチ程度で表わし、かつ、オペレータに憶えやすいコードをつけなければならないため、文字種の数が制限される。この方式の例として

ラインプット方式の特殊キーボードと印字例を図２－１に示す。

漢字	1,987 字
ひらがな	76 字
カタカナ	80 字
ローマ字	56 字
数字	20 字
符号	85 字
計	2,304 字

a．ラインプット方式で使われる文字種



b．キーボード

つぎの文は、ラインプットで記録したものである。	tugino ﾌ ﾆha'-rā'ī ﾝ pū wūtōde ﾉ ﾃ ﾛ psitamonode 'aru-'
世の中には、規格品ぎらいという思想があつて。それはおそらく個性尊重主義とつながつているのだろうが、とくに高級知識人によくみられる考え方である。しかし問題はどんな規格品をつかつても、それによつてつくりだされる知的生産物が個性的創造的であればいいのである。	j-noānniha'-ﾚﾚō ﾘ ﾛ ﾆgi ra'ito'i'uz-dyga'awu te'-soreha'osorakumb j ﾍ dw ﾍ ﾋ ūzr ﾑ totunagawu te'irunodaro'uga'-to kùnikn ﾗ bsz ﾆ ﾏ ﾍ -niyoku mirarerumg'eāide'aru-' sikasi ﾃ kkthadon-na ﾚﾚ ō ﾘ ﾛ ﾆ wotukawutemo'-so reniyowutetukuridasa reruszyrā ﾏ u ﾑ ﾌ hgamb] ﾍ yrē ﾘ ﾑ dyrde'areba'i'i node'aru-'

c．印字例

図２－１　ラインプット方式

② フルキーボード方式

フルキーボード方式とは入力する漢字を鍵盤上などに並べておき、打鍵あるいはライトペン等のタッチにより、漢字１文字を選択指示する方式である。

a）漢字テレタイプ方式

文字キー１個に対して４～１５個程度の漢字を割り当てておき、この中から文字をシフトキーにより選択する方式。文字の入力は右手で文字キー、左手でシフトキーというように両手を使って行なう。文字の配列には使用頻度、使用分野による配列や音訓順、部首別といった配列が使われており、文字種は、通常２５００～５０００字種が収容されている。この方式はもっとも早くから使用されて、現在でも広く漢字せん孔業務に用いられている。新聞社・通信社が通信回線を使って記事を電送するために開発したのがはじまりである。一般にこの方式は入力の際にモニタ出力が得られないために、入力文字の確認が行なえず、誤入力の発見が困難である。

b）タブレット方式

テーブル・ルック・アップ方式とも呼ばれているこの方式は、漢字文字盤上に配置された漢字を特殊なペンなどで押えることにより文字を入力する。機構としては静電容量結合方式、電磁方式、光電方式、超音波方式等がある。文字を盤面上に配置する点では、漢字テレタイプ方式と類似しているが全文字がＡ３版大にまとめられて、目視する面積を少な

くして、オペレータの文字の探索を簡単にしている。ワンタッチで盤上の各文字を入力できるので漢字入力に慣れない人間にも簡単に操作ができ、短時間で習熟することができる。

c）和文タイプライタ方式

和文タイプライタを応用したものであり、約５０万人という層の厚い和文タイピストをそのまま漢字入力のためのオペレータとして使うことができる利点を持っている。そのため、オペレーション技能の汎用性を大きな特長としてあげることができる。また、基本的には和文タイプライタを応用しているため、全ての印字モニタがとれるのでミスパンチは大幅に減らすことができる。この型式の入力装置はさらに次の４種類に分類することができる。

イ）和文タイプライタ搭載型

和文タイプライタをそのまま搭載する。活字盤上の活字を打字機構が１字拾うとその活字の位置関係からポジショニングにより漢字コードに変換する。この漢字コードを媒体に記録したり、コンピュータに直接入力することも可能である。

ロ）バーコード型

和文タイプライタの活字上の文字の上側または下側にあらかじめ楕形のバーコードを活字といっしょに鋳造しておき、文字の打字機構によって印字と同時にバーコードを印刷する。バーコードの印字精度を上げるために、電動式の打字機構を持つものもある。印字用紙をそのまま媒体としてバーコード・リーダに読みとらせて入力する。活字を打った表面の各行間に楕形のバーコードが並ぶので多少読みづらくなる。

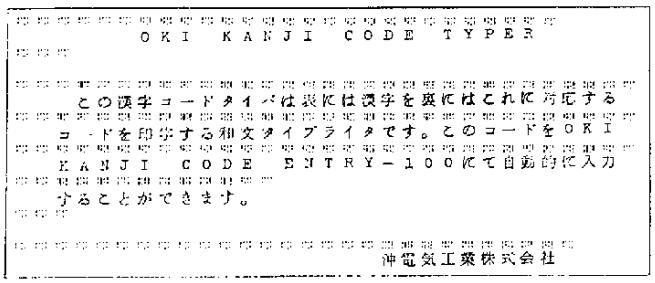


図２－２　バーコード型和文タイプライタの使用例

ハ）活字コード型

和文タイプライタの活字の側面に、その文字に対応した漢字コードをバーコードなどで表わして印刷した銘板を張付けておく。タイプライタの打字動作中に、拾った活字の側面のコードを光学的に読みとり、その漢字コードを紙テープまたはカセット・テープなどの媒体に記録する方式。

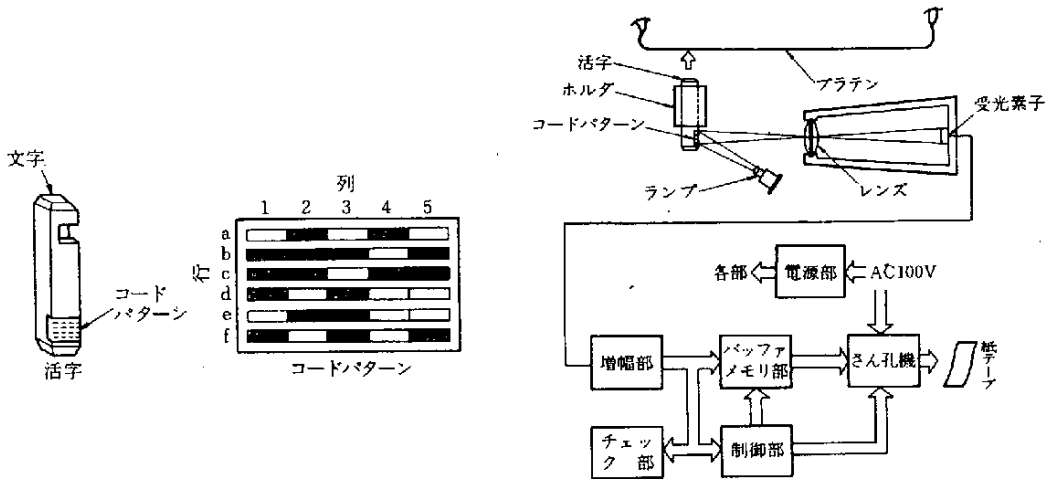
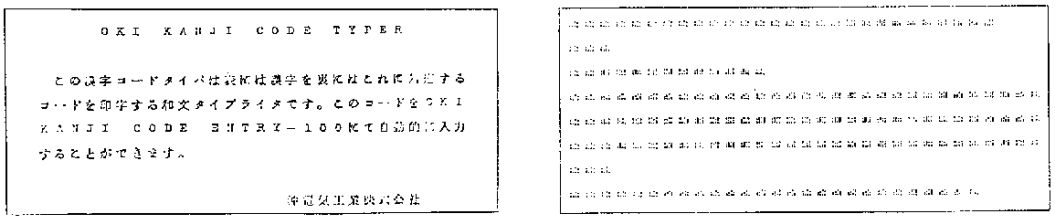


図２－３　活字コード型和文タイプライタのコード読取原理

ニ）裏打ち型

和文タイプライタの打字機構と活字盤を用いて、同様の操作により用紙の表に文字を印刷する。その時、印刷と同時に、印字した文字に対応する漢字コードを用紙の裏面に、４×５程度のドット・マトリクスなどで印刷する。この方式では、ドット符号を表に打つ場合に比べてエリアを大きくとることができる。そのため、S/N比（Signal - to - Noise Ratio）が非常に高く安定しているので約１０万分の１程度の誤字率を保つことができる。



（表）　　（裏）

図２－４　裏打ち型和文タイプライタの使用例

③ マルチストローク方式

a）カナ文字入力

この方式はカナ・タイプライタ鍵盤を用いて漢字を入力しようとするものである。この方式には大きく分けて２つの方法がある。その１つは漢字の音訓をもとにして漢字を一意に定めるようにするもので、漢字１文字に対して音に２字、訓に２字、を組合せてカタカナ文字で表わす方法と、漢字が一意に定まるまで音訓を順次入力する方法がある。この方法は当用漢字程度の文字種に限って考えれば、かなり効果があるがそれ以上の字種まで考慮すると音訓読みに無理がある。もう１つの方法は音訓いずれかを入力して、その読みに対する漢字を全てディスプレイに表示させ、表示された中から目的の漢字を選択するものである。この方式はカナ・キーボードを使って簡単に入力できる点がよいが、入力速度が遅くなる。

b）文字分解ストローク方式

この方式は、漢字をへん、つくり、……という構成要素に分解して入力するものである。漢字の読み書きになじみの少ないアメリカで研究されてきた。この方式の入力装置としては、アメリカのITEK社が開発したChicoder（中国語の機械翻訳用の入力機器）がある。具体的には「安」という文字は「宀」と「女」から構成されているので、最初に「宀」を次に「女」を打鍵すると、ディスプレイにこれらの構成要素を組合せて作られる漢字を全て表示する。この中から目的とする「安」を選び出し、その位置に対応する鍵盤を打鍵して「安」を入力する。速度としては簡単な訓練で２０～３０文字／分の速度に達するといわれている。日本人には回りくどい方法であるため、外字処理等に一部使われている。

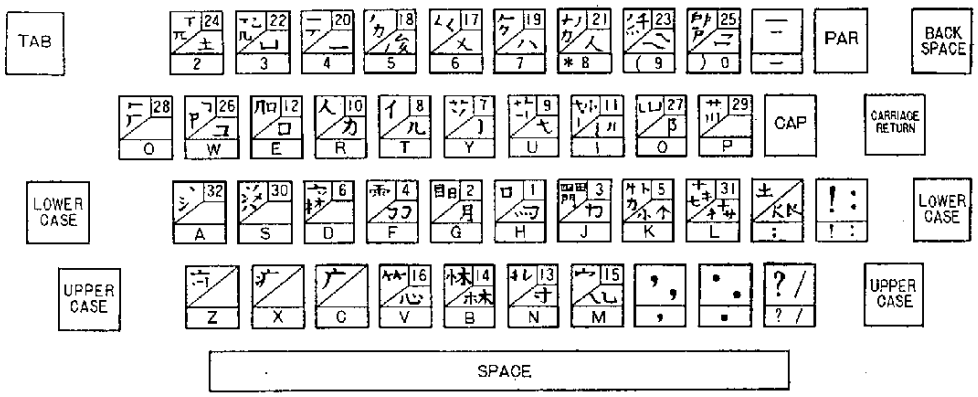
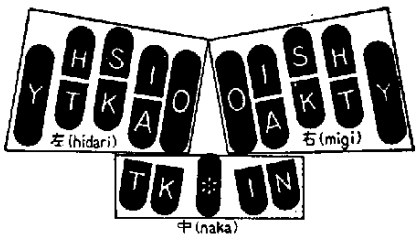


図２－５　文字分解ストローク方式の鍵盤配列

④ 速記入力方式

この方式は速記用鍵盤装置（ソクタイプ）を漢字を入力するために使うものである。ソクタイプとは、現在、全国の裁判所で使っている速記用機械であり、機械の構造は２１の活字

が一列に並び、それぞれに連結したキーを打鍵すると、特定の場所（印刷される位置が活字ごとに定まっている。）に印字される。この機械では同時にいくつでもキーを打ってよく、一列に並んで印字される。活字は固有の位置に印字されるため、２１孔のせん孔テープと考えてよい。実際に紙テープにせん孔し、その内容を解析させれば、この方式は日本語の入力手段としては速度の速いすぐれたものとなる。２１キーのうち中央の「＊」のキーが比較的使われていないため、このキーを漢字かどうかの識別に使えば、漢字かなまじり文を入力することができる。なお、ソクタイピストの養成は裁判所で行なわれており、２０年間に1,000 人の卒業者を出している。ソクタイピストの養成には２年間かかり、卒業すれば２００単語／分の速度で打つことができる。



土　tuti　T　T *T
度　do　THKS　*
怒　怒号　THKS　*　TA KH
努　努力　TK*　SKHT　(ryakugo)

T　IOTK INDIASKHTY　(ということになっても)
T S IOTK INO KH　(つずけるのでございます)
T KSAIOTK IN IAS H　(ことわらなければならない)

図２－６　ソクタイプのキーボードと使用例

⑤　オンライン手書き漢字入力方式

タブレット（一定のスペース上に筆点座標を設けたもの）の上に文字を手書きすると、その座標と筆順をコンピュータが認識して、漢字を選択する方法である。この方式では楷書体で漢字を書く場合の基本ストローク（画）を１１種類に分類し、その組合せで文字を認識している。

⑥　カナ漢字変換方式

漢字かなまじり文をカナ文字（ローマ字）によって入力し、コンピュータで漢字かなまじり文に変換する方式。変換は一般に単語辞書、文法辞書等を用いて、構文解析や意味解析等が行なわれるが、これには自然言語処理の研究成果が適用されている。同音異義語の処理には種々の方式があり、処理を１字ごとに行なうものと単語ごとに行なうもの、あるいは分節単位に行なうものとに分類できる。この方式は機器のハードウェアよりもむしろ変換プログラム（ソフトウェア）と変換辞書が重要である。方式としては、入力と変換をディスプレイを使った対話型で同時に行なう方式と、入力、変換、出力を分離した一括処理による方式とがある。

人手を介しない入力

① OCR 入力方式

イ）印刷漢字OCR

すでに印刷された漢字を直接OCRで読みとって漢字を入力する方式である。現在、工業技術院・パターン情報処理システムのプロジェクトをはじめ、メーカの研究所等で研究開発が進められている。各社の認識方法はそれぞれ異なるが、2,000字を対象とした印刷漢字OCRは、現在、研究段階を終えて商品化の時期に入っており2～3年以内には数社の製品が市場に送り出されることが予想される。

表２－２　印刷漢字認識方式の比較

	東芝総研	富士通研	日立中研
①候補の選択法	複雑指数、四辺コードによる分類	帯パターンによる分類	ボカしたパターンの総当たり
候補数	複雑指数だけ 平均190字 四辺コードだけ 平均207字 両方使用 平均 98字	平均５０字	１０字以下
②マッチング手法	複合類似度	重ね合わせ、位置ずれ補正	重ね合わせ、位置ずれ補正
類似文字の区別	混合類似度	差分マスク	対判定加重相関

ａ）手書き漢字OCR

すでに手書きされた漢字をパターン認識技術によって直接入力しようとするもの。手書き漢字認識に力を注いで研究しているメーカはほとんどなく、公的な研究所・大学で、少数の研究者によって地味な研究が続けられている。用途を限定して小数の漢字を対象とすれば、かなり高い認識率が得られる。

② 音声入力方式

一般的な音声入力または認識は、学問的な興味から数多く研究されているが、実用化を目的として開発されるには至っていない。ただし、日本語文の音声というより、数値情報や単純な命令的情報の音声入力に限れば、一部実用化をめざして実験中である。音声入力は情報処理の終局的な夢であり、これが実用化された場合には、人間は音声による情報伝達に慣れているため非常に有効であろう。最近、音声認識処理システムとしてオペレータの音声を３２語まで認識できるものが出てきているので、これからの発展が期待できる。

（１－２）　外字入力方式

現在、多くの漢字入力装置は 3,000 字前後の文字を収容しているが、実際には、まれに使用される漢字も含まれると種類は、はるかに多い。字種の多い例では 50,000 字近くを収容.

した字典もある。

しかし、普通の漢和字典は 10,000 字前後のものが多く、また研究機関等における漢字調査結果からみてもおよその上限が 10,000 字くらいだと推定できる。しかし漢字入力装置の盤面の規模からしても 3,000 字種前後しか収容できない場合が多いため、他の 7,000 字種の入力が必要になってくる。これが外字である。外字入力の方法は一般にテンキー方式、ブック型バーコード入力方式、文字合成方式、イメージ方式などがある。その概略を以下に述べる。

① テンキー方式

この方式は外字に対して、漢字と数字コード（10進4桁のコードが多い）との対応づけがなされた索引を準備しておき、その索引を使って目的とする漢字の数字コードを探して、これをテンキーボードから入力するものである。

② ブック型バーコード入力方式

外字に対して索引を用意しておくことはテンキー方式と同様であるが、この方式では数字コードのかわりにPOSに使われているバーコードを印刷したものを分野ごとに分類し、本にしておき、これをPOS用ハンドスキャナなどで走査して漢字に対応したバーコードを入力する。

③ 文字合成方式

漢字を構造面からみると、いくつかの構成要素（へん、つくりなど）に分解することができるが、これらの構成要素を組合せて漢字を表現する方法である。この方法は外字に対して直視的に判断することができる利点がある。しかしながら、漢字の特性として、その構成要素を少なくして入力しようとすると、その組合せの記述が複雑となり、入力キーの数は少なくなるが、入力タッチ数が増加する。一方、構成要素の数を多くすると入力キーは多くなるが入力タッチ数が減り、入力の判断が容易となる。このため外字入力用としては後者の構成要素の数を多くした方が有効であるとされている。

④ イメージ方式

所定の用紙上に手書きで目的の文字を描き、これをイメージ入力装置に読み込ませる。イメージ入力装置では、この手書き文字を標準ドット・マトリックスに変換してパターン・ジェネレータの書き込み可能エリアに記録する。そして、文字コード・テーブルから1組の使われていないコードを選び出し、そのコードをオペレータに知らせる。この様にオペレータに新しい文字パターンを入力させて、フォントを作成、登録し、それに付られた文字コードを入力する方式である。

表２－３　漢字テレタイプ方式

（メーカ）	（機　　種）	（入力速度）	（収容文字種）	（ベリファイ機構、モニタ）	（外字処理方式）	（コード）	（備　考）
アルプス電気	SCK5A00L		2643　５０音順（１３９キー）			８ビット／２列	
日本IBM		６０字／分	3,600　（２４０キー）	モニタ無	数字コード入力	２行カードコード EBCDICコード IBM漢字コード	
沖	GL	５０字／分	2,400　（６００キー） 2,600～5,200　（６５０キー）	ベリファイ無 ディスプレイにモニタ		５、６、８単位／２列	
写研	NI 201	６０字／分	2,784　（２３２キー）			CO 59コード	
高千穂交易	T4112（T 4011） T 4012		3,072～8,192　（２６５キー） 3,032～8,262（２６５キー）		コード入力 コード入力	８ビット T400漢字コード	
谷村新興	SC－４		2,304　（１９２キー）	モニタプリント可		６単位／２列 CO 59コード	
	KB 202A		2,688～7,188（１９２キー）	モニタ無	パターン入力	８ビット JEM漢字コード	日本電子産業と協同
	S 4000		4,096（４８キー）	モニタ可		６ビット、８ビット	
	SCC 4000		4,000～8,000（１９８or４８キー）	打鍵直後の１文字修正、１ブロック単位抹消			
	SCK200D（SCK200K）	６０字／分	2,850　（１９２キー）	モニタプリント可		CO 59コード	
	SC 4000		4,049～8,192			CO 59コード	
東京機械	FAM漢字鍵盤		2,376（1,188キー）			６単位／２列 FAMコード	
東レ	TORAY 8510（8511）	６０字／分	3,240　（２７０キー）	ベリファイ無		１６ビット／字 東レ・コード	
日本電子産業	KB－２０２A	６０字／分	2,688　（１９２キー）部首順	ベリファイ無			
富士通	FACOM 6801A	６０字／分	5,376×2書体（４８０+１９２キー）	モニタ無	システムにより異なる	８ビット 富士通漢字コード	
	FACOM 6802A，B，C	６０字／分	2,880　（２８１キー）	モニタ無	テンキーによる符号入力	６ビット CO 59コード	
昭和情報機器	S5011／5111		3,072		TKコード	８単位	
日電漢字システム	C 5130J		2,688　（１９２キー）			８ビット×２列 EXOK	

表２－４　テーブル・ルック・アップ方式

（メーカ）	（機種）	（入力速度）	（収容文字種）	（ベリファイ機構、モニタ）	（外字処理方式）	（コード）	（備考）
日本IBM			3,520字以上	モニタ無			対座標入力
ゼネラル	KX202		2,826			6，8ビット×2列	静電結合式
谷村新興	S－1184D		差しかえ 2,000＋2,000			6ビット×2列	
東芝	KIM－100（KIM200）		4,049			6ビット×2列	電磁式
東京機械	FAN5040	40字/分	7,000	ベリファイ無			新聞社専用
日電漢字システム	C－5120J		2,976（音訓順）	紙テープ読取りにより1字ずつ文字を発光	自由文字部分(220文字分)を指定する	8ビット×2 FXOK－2	光電式
日本電子産業	KB－301	50字/分	2,976	ベリファイ有			
日立			2,400	モニタ無		12単位（6単位×2列）	座標入力
ベンテル	PK1000	40字/分	1,536			6 × 2	静電結合式
	PK2000C		2,880				
	PK2000D		2,880×3シート				
	PK3000B		3,591			6単位2列 CO59拡張文字コード	
	PK4000		4,096				
	PK7000		7,412				
富士通	FACOM 6578C		3,072		4桁の10進数	富士通漢字コード	
三菱電機	M8230		3,840				光電式
東レ	TORAY 8514		4,096				静電結合式
オールシステム	AS－1101		4,096				静電結合式
	AS－1102A		4,096				光電式
	AS－1102B		4,096				電磁式
沖	KTE－500		3,304				静電結合方式
	KTE－600		3,342				静電結合方式
日立	9425ビデオ・ターミナル		4,096				
日本ユニバック	J 1797		2,816(＋1,024)		コード入力、（オクタル5桁）	16ビット ユニバック漢字コード	
日電漢字システム	7100（7200）		3584				

表２－５　和文タイプライタ方式

（メーカ）	（機種）	（入力速度）	（収容文字種）	（ベリファイ機構、モニタ）	（外字処理方式）	（コード）	（備考）
沖	GP		2,304		192字まで		ジャーナルテープ印字
沖東京磁気印刷			2,145			ラインバーコード	バーコード型
東京電機	FAN300M		2,304	ベリファイ無	4,500字可能		新聞社専用
日本システム技術	KIPROS5010（漢タイパー）		3,132			6／8単位	
日本タイプライタ	KB－7	30字／分	2,304				
日電	C－5500	30字／分	2,940	バックスペースでベリファイ	ファンクションキー	JIS 8単位	
	C－5110	30字／分	50音順 2205～7,875	バックスペース可	コード入力	NEAC漢字コード	
	C－5510		2,940（50音順）		予備の活字盤使用		バーコード方式
富士通	FACOM6805		5,182		4桁の10進数	富士通漢字コード	活字コード型
PDC		30～60字／分	3,259	ページ内文字任意修正可能	HEX入力		
学研	GKS－111		2,205			8ビット×2	
沖			2,200	誤字修正、脱字追加		16ビット	裏打ち型

表２－６　マルチストローク、連想記憶方式

（メーカ）	（機種）	（入力速度）	（収容文字種）	（ベリファイ機構、モニタ）	（外字処理方式）	（コード）	（備考）
沖			7,488	光学的表示			カナキーボード音節選択方式
谷村新興	SCW－4000		4,096			CO－59コード	カナキーボード
カンテック		120字／分	8,464	ベリファイ機能有			カナキーボード
ラインプット			2,304				

(2) 漢字出力装置

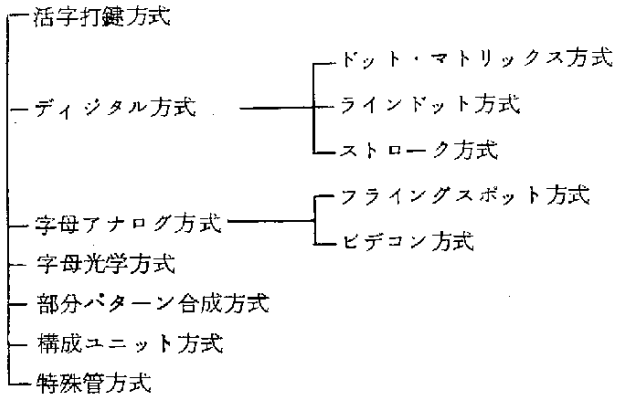
コンピュータを利用した情報処理において出力結果は重要であり、正確な出力と、それを利用するものにとっては、みやすいものでなければならない。従来、ラインプリンタをはじめ、どんな出力装置にしてもコンピュータ処理の出力情報は英・数・カナ文字等で表わされ、我々日本人に最も親しみやすい漢字かなまじり文で出力することはできなかった。カナ文字列では読みにくく、せっかくコンピュータで出力した情報も確実・正確性をもって情報交換することができない。そのためコンピュータで漢字かなまじり文を出力するために開発されたのが漢字出力装置である。昭和30年代、新聞・印刷業界での経験を基にして漢字出力機器が各メーカから多数開発されるようになった。その後の技術の進歩は目ざましく、漢字情報処理の出力側の大きな問題は、コスト面をのぞき、ほぼ解決されたように思える。

小さな問題としては、印字方式・用紙の問題などがいくつか残されている。

このようなことから、漢字出力装置の分類方法が種々ある中で、コスト・印字品質の面から検討が必要な文字発生方式の分類を試みる。つづいて、漢字出力装置の利用・運用の面からハード・プリンタとソフト・プリンタの分類を行なってその特徴を述べることにする。

（2－1） 文字発生方式

表2－7 文字発生方式の分類



① 活字打鍵方式

一字ずつ活字をハンマで打って印字する純機械式の方式。この方式は多字種とインパクト型のため印字速度が遅く、騒音も問題となる。また、字体、ピッチ、文字サイズ等も一定で融通性がなく現在ではこの方式の出力装置はほとんど使われていない。

② ディジタル方式

漢字パターンをディジタル化することにより磁気コア、ディスクなどのコンピュータの一般的な記憶装置に記憶させることができる。

a) ドット・マトリックス方式

文字を点（ドット）の集まりとして表現する方式。英数字の場合には5×7ドット程度で

表現でき、キャラクタ・ディスプレイの文字発生方式として広く用いられている。漢字を表現する場合、最低15×18ドットは必要である。格子を密にすれば、いくらでも高品質の文字を得ることができ、明朝体やゴシック体の区別など文字の微妙な点まで表現できる。しかし、文字パターン（フォント）を記憶するのにメモリを大量に必要とし、文字発生時にも大量にメモリを読み出さねばならないので、出力速度に影響する。

b）ライン・ドット方式

この方式は文字を縦あるいは横のラインで分割してメモリに記憶しておく。メモリには、縦に分割する場合、ラインのX座標とライン上での白（地）と黒（文字部分）との境界の座標が記憶されている。文字を発生させる場合には、この座標をメモリから読み出して画像信号に変換しCRTに表示する。この方式を用いたものにはビデオ・コンプ（Videocomp）がある。この方式はXY方向のいずれかがドットではなく直線で再現されるため文字品質はきわめて優れている。この方式もメモリを大量に必要であり、分割を横方向にした場合でも100×120分割で1字当り平均700バイトを必要とする。

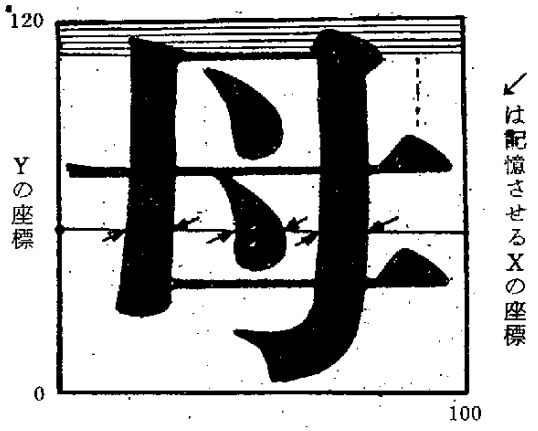


図2－7　ライン・ドット方式の原理

c）ストローク方式

漢字をドットではなくストローク（直線）に分解して表現する方式。ストロークはグラフィック・ディスプレイと同様に、始点と終点を指示するか、始点の座標と長さと方向を与えて表現する。太い部分はストロークを数本重ね合わせ、曲線は折線で表現する。この方式は直線部分が多いほど有利であるが、ドット・マトリックス方式と比べて必要なメモリ量はほとんど変わらなく、各文字パターンの記憶容量はそのストローク数によって可変である。出力装置はストローク信号によって動作することが要求されるため、装置が複雑になる。多くは、グラフィック表示機能をもったCRTディスプレイやCOM装置などに用いられる。

③ 字母アナログ方式

a）フライング・スポット方式

この方式では文字マトリックス（高解析度写真乾板）に文字パターンが収容されている。入力された文字コードによりフライング・スポットCRTのビームが文字コードと対応した文字マトリックス板上の文字を選択走査する。選択走査はXY偏向回路によってCRTビームの位置を指定することで行なわれる。このCRT走査により選択された文字形の光信号は光電子増倍管によって増幅され、整形回路を経てプリントCRTに1字ずつプリントされる。フライング・スポット方式の文字信号発生は文字パターンのアナログ情報をそのまま出力するのが特徴となっている。しかし、読み出し部の構造が複雑となる欠点がある。この方式では多種文字を高速に、高品質で、任意の形・大きさで発生させるのに適している。

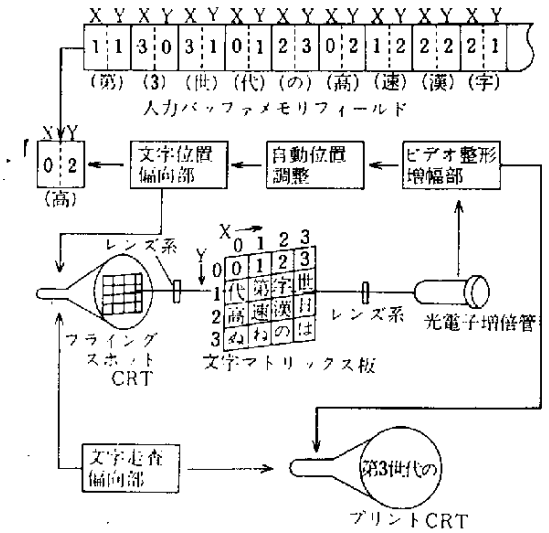


図2－8　フライング・スポット文字発生方式

b）ビデコン方式

この方式は文字円板、フラッシュ・ランプ、ビデコン（Vidicon）の組合せで文字を発生させている。文字円板を回転させておき、フラッシュ・ランプの閃光により円板上の1列の文字パターンをビデコン上に蓄積する。この中から必要な1字を選択走査して文字の走査信号を得る。この方式は文字を読み出す際に円板の1部しか使用しないために解像度が比較的よく、1枚の円板上に数千の文字を収容できるため文字パターン・メモリを小さくできる。フライング・スポット方式と比べると速度も遅く、解像度も悪くなるが偏向系が簡単であり、解像度もドット・マトリックス方式と比べると、アナログ方式のため、自然な文字が出力されて可読性がよい。

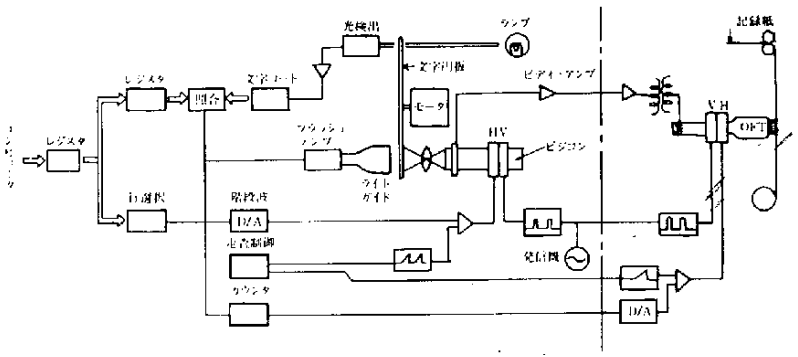


図２－９　ビデコン方式の文字発生原理

④　字母光学方式

この方式は回転している文字盤にフラッシュランプを照射して、必要とする文字をレンズ系を通して、プリント面に撮影するものである。文字盤の種類によって回転円板型、回転円筒型、固定文字盤型などに分類される。例えば、回転円筒型の原理は、文字盤上にある出力したい文字が規定の位置（フラッシュ・ランプの正面）に来た時、フラッシュ・ランプが点灯され、所望の文字を含む円周方向一列の文字が選択される。この文字の光学像は光学セレクタに入り文字１字１字に対応した反射鏡のうち所望の文字が入射する反射鏡だけ曲げられ、その光学像だけセレクタを通過する。その後、プリズムで反射されフィルム上に結像される。プリズムの移動により行内の送り、フィルムの移動により行送り、光路の途中にあるレンズを変えることによりポイント・サイズの変更を行うことができる。

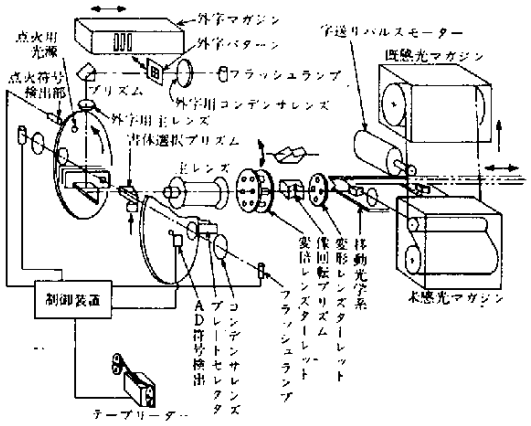


図２－１０　字母光学方式原理図

⑤　部分パターン合成方式

漢字をその構成要素から組立てて発生させる方式。同一部分が多くの漢字に共通に用いられることから、発生効率を向上させるのに役立つ。漢字を「へん」、「つくり」、「かんむり」、

「たれ」などに分解して構成要素を抽出するとだいたい250 程度のパターンで当用漢字を表現することができ、これらの配置関係を表わすオペレータとして10個程度あれば十分である。そのため、パターンメモリの面からは大きなメリットが考えられる。実験によるとドット・マトリックス式、ストローク方式と比べて $\frac{1}{2}$ ～ $\frac{1}{3}$ になると報告されている。同一パターンでも文字の中の相対的な位置関係により形が異なってくるため、文字品質の面に問題がある。パターン自体の印字・表示方式はドット・マトリックス 方式、ストローク方式、字母方式等による。

⑥ 構成ユニット合成方式

部分パターン合成方式では漢字を構成パターンに限定したが、この方式では文字を絵とみて、構成ユニットを組立てて、忠実に絵を再現しようとするものである。英文字の場合には、平均10個のユニットで合成でき、9種類のユニットで表現することができる。この方式では、明朝体特有の「はね」や「おさえ」用の構成ユニットを作っておけばよいため、明朝体等を表現するのが有利である。この方式は、いわば、ディジタル・アナログ混合方式ということができる。

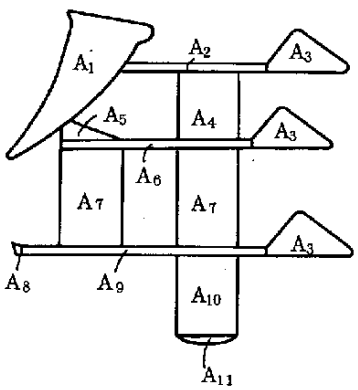


図2－11　構成ユニット合成方式の例

⑦ 特殊管方式

文字を発生させるために特殊管を用いる方式。英数字を発生させる特殊管としては、CRTの中に文字板を封入したキャラクトロン管、ビデコンに文字選択格子を封入したライノトロン管がある。英数字は字種が少なく字画も簡単なために早くから実用化されてきた。漢字用の特殊管には富士通のモノスコープ管がある。これは1つの管に 384 文字収容しており、幾本かの管をいっしょにして、プリンタやディスプレイの文字発生装置として使われている。

（2－2） ハードプリンタ

漢字プリンタの分類

漢字プリンタを大別するとインパクト型とノンインパクト型になる。

① インパクト型

a）活字方式

活字をハンマで打鍵する純機械式のプリント方式。この方式は漢字プリンタとして最初に用いられ、漢字テレタイプの受信印字装置として使われていた。そのため機械は安定しているが100～300字/分と出力速度がきわめて遅く、印字も不揃いである。

b）ワイヤ・ドット方式

文字をドット・マトリックスで表現し、各ドットに対応した細いワイヤピンを配置し、これらのワイヤでインクリボンを叩き印字する。この方式はワイヤが機械的に使われるため、磨耗その他の機械的トラブルの心配がある。また、インパクト方式のため速度が遅く、騒音の心配があるが、最近は、音も気にならないようになった。

② ノン・インパクト型

a）ノンインパクト型のドット方式

文字をドット・マトリックスで表現するのはワイヤ・ドット方式と同じであるがノン・インパクトのため機械的な部分がほとんどなく、速度も速く、騒音等の心配もない。ノン・インパクト式のドット方式は多くの出力機器で使用され、多くの印字方式で実現されている。

c）ライン・ドット方式

この方式は文字を縦または横のラインに分割して、そのライン上の白（地）と黒（文字の部分）の境界の座標をメモリに記憶しておき、必要に応じて画面情報に変換して文字を再現する。

d）ストローク方式

文字をストローク（直線）に分解して表現する方式。直線を表現する場合、始点と終点の座標を指示するか、または、ベクトルで始点の座標と方向、長さを指示して表わすため、ドット・マトリックス方式のように中間の点を指定する必要がない。しかし、曲線は折線で表わすため記述が大変である。

e）字母アナログ方式

この方式にはフライング・スポット方式とビデコン方式がある。フライング・スポット方式は文字板を内蔵していて、その文字板が固定されている。ブラウン管上の文字位置から、スポットで文字盤上の所要の文字を照らし、そのブラウン管と反対側の受像管が受けてプリントする。途中にメモリを置かないため、タイムラグがなく、また電子式であるので、印字速度もきわめて速く、高品質の文字を出力できる。

ビデコン方式は、同心円状に文字を6,000字程度配列しているプラスティック円板を持っており、この円板を回転させる。フラッシュ・ランプのせん光により ビデコン上に1列の文字パターンを蓄積し、必要な文字1字を選択走査してビデオ信号を得ている。この方式は、偏向系が簡単で、価格的にも、比較的安価である。

f）字母光学方式

この方式は写真植字機からきている。コードが与えられると自動的に写真植字機が動作して印画紙に漢字を投影し、それを現像してプリント・アウトする。投影される文字品質はきわめて良く、印刷用としても効果がある。機械的な部分があるため、プリント速度は遅い。

印字方式の分類

漢字プリンタで採用されている印字方式は種々の方式が存在する。印字する際に使用する用紙も、普通紙、静電記録紙、酸化亜鉛紙、銀塩写真紙、感熱紙、アルミ蒸着紙などがある。ここでは、印字方式を印字の際に使用される用紙側から分類してみる。

① 普通紙を使用する印字方式

普通紙を使用する方式には、乾式電子写真方式、インクミスト方式、インクジェット方式、ワイヤドット方式などがある。

a）乾式電子写真方式

酸化亜鉛（ZnO）または、硫化カドミウム（CdS）等の記録媒体を帯電・露光し静電潜像を形成させ、磁気ブラシ法により現像トナーを普通紙に転写し、定着させ、プリント結果を得る方式。露光にはOFT（Optical Fiber Tube）またはCRTを用いているため、解像度がよく、文字サイズの変更が容易であり、高品質、高速印字も可能である。最近は、レーザ・ビームをオン・オフ制御して、回転しているドラム上に光導性の薄膜に潜像を作り、トナーをつけて紙に転写し印字するレーザ・ビーム・プリンタが現われてきた。このプリンタでは1分間に数千行も高速印字することができる。乾式電子写真方式では、基本的にはゼロックス社あるいはキャノン社などの複写機メーカが開発したPPC（Plain Paper Copy）方式をコンピュータの出力機器として利用できるように機構を改良して、耐久性、信頼性を上げたものである。

b）インクミスト方式

超音波によりインクを攪拌し、インクミストを発生させる。このインクミストの中に、コロナ放電で発生させたイオンを電界で制御し、形成したイオン流を流入させる。そして、インクミストの周囲にイオンを付着させ、加速して印字する方式。イオン流を個別の小孔から供給するため、解像度に制限があり、文字サイズが大きくなるが、高速印字が可能である。

c）インクジェット方式

この方式はインクの噴射原理により、電界制御型と荷電量制御型とに大別される。電界制御型はノズルより微小圧力と高電圧を与えたインクを電界の静電力により誘引し分裂させて粒子列を形成する。このインク粒子を電界により偏向させて印字する。荷電子制御型はインクに対して高電圧と超音波を与え、噴射させ、インク粒子列を飛翔中に高電圧で帯電させる。このインク粒子を偏向電極で偏向させて印字する。インクジェット方式は、シ

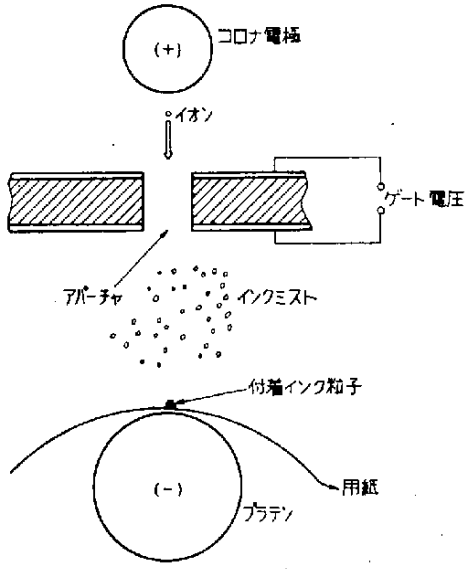


図2－12　インクミスト方式の原理

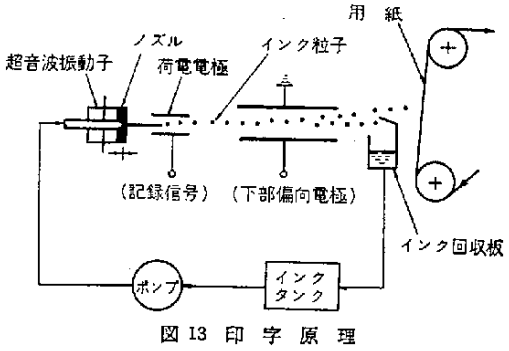


図13 印字原理

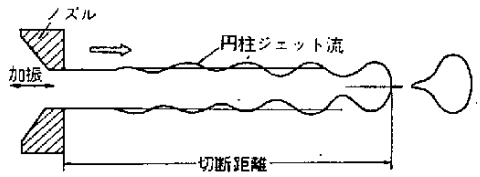


図2－13　インクジェット方式の原理

リアル・プリンタに使用され、印字速度は荷電子制御方式で300～500字/秒、電界制御型で30字/秒程度である。（英・数・カナ文字の場合）。電界制御・荷電子制御方式のほかに、ファクシミリへの適用をねらったものとしてオン・オフ制御方式のものがある。漢字プリンタとしてこのオン・オフ制御方式が検討されており、32×32 ドットで60/秒の速度である。

d）ワイヤピン方式

文字をドット・マトリックスで表現して、各ドットに対応した細いワイヤピンを配置する。文字発生回路からの信号を受けて、タイミングよくワイヤを突き出してインクリボンを叩き用紙の上に文字を印刷する。印字機構はプランジャー型とクラッパ型に大別できる。

② 静電記録紙を使用する印字方式

この方式は、低抵抗の基紙に誘電体（樹脂）をコーティングした記録紙（静電記録紙）に電圧を印加して、静電潜像を形成させ、現像して定着させる。この方式には光静電記録方式、静電記録方式がある。

a）光静電記録方式

OFT（Optical Fiber Tube）、透明電極、光導電体膜、静電記録紙の順に配置する。透明電極に印加されている電圧をOFT で露光した光導電体膜を経由して静電記録紙に与える方式。OFT 上に解像度の高い光線を利用できるので、次に説明する静電記録方式で針電極を使用する場合よりも印字品質が向上する。

b）静電記録方式

針電極により静電記録紙上に直接、電圧を印加し電荷を与える方式、針電極の形状には、千鳥配列のもの、回転ドラムにらせん状に配列したもの、マトリックス状に配列したものがある。この方式は機構が簡単であり、多層構造の静電記録紙を用いればコピーがとれる。しかし、針電極の寸法、耐圧から解像度が悪くなる。

③ 酸化亜鉛紙を使用する印字方式

○ 湿式電子写真方式

紙に酸化亜鉛（光導電体）を塗布した感光紙（酸化亜鉛紙）を一様に帯電し、OFT で露光して静電潜像を形成させ、湿式現像を行なってトナーを付着させ定着させる。露光はOFT を使って行なわれるため、解像度は高く印字品質は良好であるが、酸化亜鉛紙自体の色が印字用紙のベースの色として残るため、普通紙への印字と比べると印字のコントラストが高くとれない。

④ 銀塩写真紙を使用する印字方式

○ 銀塩写真方式

この方式は、銀塩写真紙にOFT で露光し、現像・定着させる。解像度は高いが、用紙が高価で、装置の取り扱いも難かしい。

⑤ 感熱紙を使用する印字方式

○ 感熱記録方式

この方式の原理は、サーマルヘッドと感熱紙が基本要素となっている。サーマルヘッドでは、数ミリ秒の間、200～300℃位の温度に発熱された熱潜像を発生させる。この熱が感熱紙に与えられ、感熱紙はこの熱を受けて発色し、目に見える像としての印字となる。機構は簡単であるが、解像度が低く、速度も遅い。また、感熱紙を使用するために、いつでも熱を加えれば書き換えられ（改ざん性）、日光の照射等により、かっ色化すること（保存性）がある。

⑥ アルミ蒸着紙を使用する印字方式

○ 放電破壊式印字方式

この方式は古くからファクシミリに用いられてきた。プリント・ヘッドで、アルミ蒸着紙に放電し、印字する。記録中に、わずかな臭いとカスを発生するが、騒音の発生が少なく、保守取扱が容易で、記録速度が比較的速い。特殊紙を使用するため、記録紙のコストが高く、大量に記録紙を使う用途にはむかない。

表 2 － 8　印字方式の比較

プリント用紙	プリント方式	概要	備考
普通紙	乾式電子写真方式（PPC方式）	帯電 → OFT —露光→ [ZnO] →乾式現像 —転写→ [普通紙] → 定着	
		帯電 → CRT＋レンズ —露光→ [CdS] →乾式現像 —転写→ [普通紙] → 定着	
	インクミスト方式	イオン流 —電界制御→ インク加速ミスト → [普通紙]	
	インクジェット方式	ノズル→インク粒子列 —電界で偏向→ [普通紙]	
	ワイヤピン方式	ワイヤピン —インクリボン→ [普通紙]	インパクト方式
静電記録紙	光静電記録方式	OFT —露光→ 光導電体膜 —電圧印加→ [静電記録紙] → 湿式現像 → 定着	
	静電記録方式	針電極 —電圧印加→ [静電記録紙] → 湿式現像 → 定着	
酸化亜鉛紙	湿式電子写真方式（湿式エレクトロファックス方式）	帯電 → OFT —露光→ [酸化亜鉛紙] → 湿式現像 → 定着	
銀塩写真紙（安定化処理）	銀塩写真方式	OFT —露光→ [銀塩写真紙] → 湿式現像 → 定着	
感熱紙	感熱記録方式	サーマルヘッド —加熱→ [感熱紙]	低速
アルミ蒸着紙	放電破壊方式	プリントヘッド —放電→ [アルミ蒸着紙]	

表２－９　電子写真方式

（メーカ）	（型）	（速　度）	（字／行）	（用　紙）	（CG）	（ドット構成）	（字　種）	備　考
富　士　通	F－6504A	2,000行／分	68～136	普通紙	モノスコープアナログ型	母型　外字用 32×32	3,479	
日　立	H－8195 H－8196	12枚／分		普通カット紙			8,000～ 23,000	レーザビーム ページプリンタ
日電漢字システム	JEM－NEAC 7300	2,600行／分	68、34	普通紙	磁性メモリ	ストロークドット	4,000～ 10,000	CRT
三　菱	M－8220							
	8240							
	8250							
	8251	3,600行／分	136	スプロケット付 一般連続用紙 （Z紙）		32×32	2,048， 16,384	OFT乾式
沖	オキレーザビームプリンタ	2,700行／分	113～136	普通紙		16×18 24×24 32×32	65,536	レーザ ドット・ノンインパクト
昭和情報	S－5400							
	SP－5400							
	S－5421	180,000字／分	90			32×32		OFT
東　レ	TORAY8570							
	TORAY8572	1,500行／分	80	普通紙	ホログラムメモリ	32×32	7,000， 10,000	液乾電子写真
日本ユニバック	漢字プリンタ	2,800行／分	40～80	普通紙	LSIメモリ	32×32	2,048～ 16,348	OFT乾式
日本　I.B.M	IBM3800							
キ　ヤ　ノ　ン	LBP－2000							
	LBP－4000							
オールシステム	AS－1201							
富士ゼロックス	ゼロックス－1660							
大　倉　商　事	40T1J							
	40T2J							

表 2 － 10　静電記録方式

（メーカ）	（型）	（速　度）	（字／行）	（用　紙）	（CG）	（ドット構成）	（字　種）	備　　考
富　士　通	F－6501B	121.5行／分		静電記録紙	磁気ドラム	16×15	2,688＋384	多針電極、コピー
	F－6502A	600　行／分	68	静電記録紙	磁性線ROM	15×18	2,400～7,200 3,479	光静電、コピー
	F－6521B	100　字／秒	46	静電記録紙	字円板ビジコン撮像		5376	光静電
	F－6521C	286　行／分	21	静電記録紙	回転文字円板	母型20本／㎜	3,387～4,096	光静電
日　　電	C－5210	1,600　行／分	64～80	静電記録紙	ワイヤメモリ（磁気ディスク）	24×24	2,961～7,875	高密度多針電極
	C－5210D	1,800　行／分	64～80	静電記録紙		24×24	2,961～7,875	高密度多針電極
	C－5230（4タイプ）	170～230 行／分 120～170 行／分	46～96 36　75	静電記録紙		17×18 24×24	3,967～7,875	
日電漢字システム	JEM－NEAC 7150	400　行／分		静電記録紙			7,875	
東　　芝	KF 3000 KF 4000	600　行／分 700　行／分	24	静電記録紙	磁性線ソレノイド	24×24	2,362～4,076	
	KF　100 KF　200	26～53 行／分	16	静電記録紙	変成器形ROM	24×22	2,362～2,842	多針電極
伊藤忠データシステム	CP－4000							
	CP－8000							
理経 Varsatec	モデル200A	600 行／分		静電記録紙				
日本システム技術	KIPROS－5030	100 字／秒		静電記録紙				
昭　和　情　報	S－5200	14,000 字／分		静電記録紙				
	S－6200	14,000 字／分		静電記録紙				
谷　村　新　興	S－6200	14,000 字／分		静電記録紙				

表－11　ドット・インパクト方式

（メーカ）	（型）	（速　度）	(字／行)	(用　紙)	（CG）	(ドット構成)	(字　種)	備　考
富士通	F－6523A	30字／秒	68	普通紙 感圧紙	ディジタル・メモリ	15×18	5,376	ワイヤインパクト
沖	オキタイバ 8500Ⅰ	60行／分	60	普通紙	回転文字板	16×18	1,500×α	ワイヤピン
	〃 8500Ⅱ	1,200字／分	30、66	普通紙		24×24	4,000	
日本 I.B.M	IBM－2245	300行／分	16	普通紙	磁気ディスクドラム	18×22	3,600～ 10,000	ワイヤピン
日電	C－5220	100行／分	26	普通紙 感圧紙	ワイヤメモリ 磁気ディスク	17×18	2,961～ 3,969	
谷村新興	KPS－7200	120字／分						ワイヤ・ドット
	SC －8000	600字／分	66		ディスク	26×26	4,000	データ・ターミナル
東レ	TORAY－8575	34字／秒				16×18	3,500	ワイヤ・ドット
ワング・コンピュータ	2211／WK	60字／秒				16×18	10,000	ワイヤ・ドット
電々武蔵野通研		60字／秒				18×16		データ通信端末 ワイヤ・ドット

表２－12　ノンインパクト（インクジェット、インクミスト）方式

（メーカ）	（型）	（速　度）	(字／行)	(用　紙)	（CG）	(ドット構成)	(字　種)	備　考
沖	エレクトロ・プリンタ	4,000行／分	66	普通紙	コア・メモリ ワイヤ・メモリ	16×18	2,048～ 65,536	インクミスト
東レ	TORAY－8576	60字／秒				22×24	5,632	インクジェット
電々武蔵野通研		30～ 120字／秒		普通紙		32×32 27×20 18×16		データ通信用端末 インクジェット

（2－3）　ソフトプリンタ（ディスプレイ）

漢字ディスプレイも一般のディスプレイと同様にコード・リフレッシュ型とパターン・メモリ型に分けることができる。

① コード・リフレッシュ型

コード・リフレッシュ型は、一般の英・数・カナのキャラクタディスプレイで多く使われている方式である。文字コードをコード・リフレッシュ・メモリに入れておき、そのコード・リフレッシュ・メモリを一定周期（普通1／30秒）で読出し、その都度、文字発生装置により該当する文字パターンを発生させ文字を表示する。

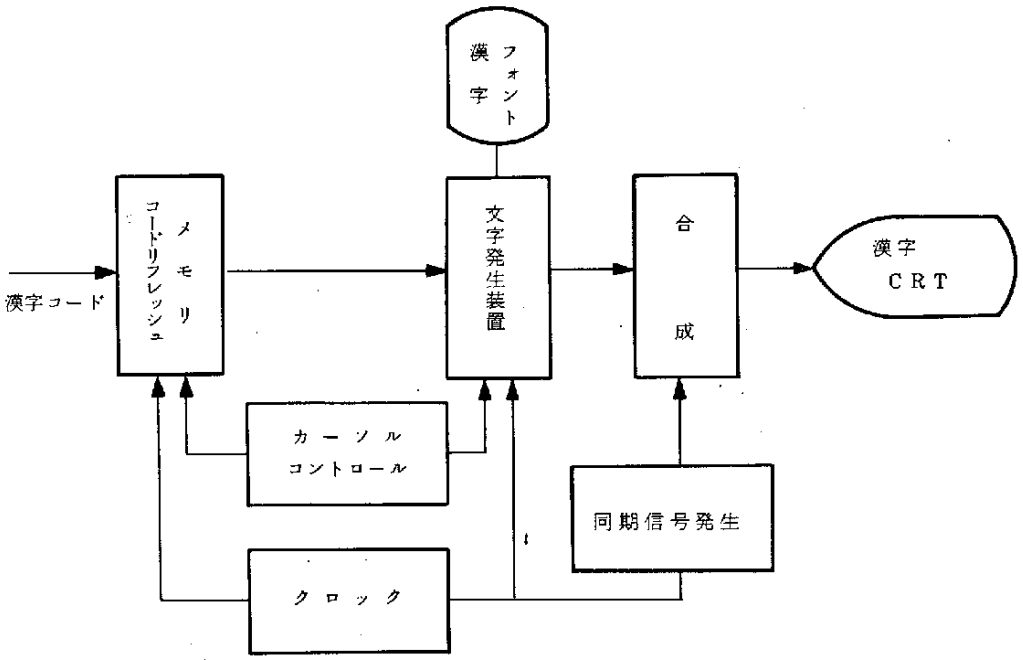


図 2 － 14　コード・リフレッシュ方式のブロック図

このコード・リフレッシュ方式は周期的に文字パターンを発生させなければならないため、一般に1台のディスプレイ装置で1つの文字発生装置を専有する。そのため、文字発生装置が高価な漢字ディスプレイではあまり用いられていない。また、漢字のパターンが複雑であることから、文字発生速度もかなりの高速性が要求される。

② パターン・メモリ型

パターン・メモリ型は表示したい漢字のコードが入っているコード・バッファから1字ずつ読み出し、該当する文字パターンを漢字文字発生装置で発生させ、そのパターンをパターン・メモリに記憶させる。この方式はパターン・メモリと表示用 CRT を兼用する蓄積管型と、それぞれを別物で構成したパターン・リフレッシュ型に分けることができる。商品化されている多くはパターン・メモリ以下を複数台接続したマルチ・ステーション方式のもので

表２－１３　ディスプレイ方式の比較

メーカ（型名）	表示部							キーボード		文字発生部				最大設置台数	備考
	形式	表示方式	表示文字数 横書	表示文字数 縦書	画面の大きさ	画面メモリ	ライトペン	入力方式	入力文字数	方式	文字パターン	記憶素子	収容文字数		
日電 N7872-11		ドット リフレッシュ	40字×20行 (17×18) 32字×15行 24×24	24字×27行 (17×18) 18字×21行 (24×24)	12″	MOS RAM	可	タブレット式	3584	ドット	17×18 24×24	MOS RAM	最大8K字	16	
日電 N7872-21	高解像型	ドット リフレッシュ	35字×24行	36×23行	14″	MOS RAM	可	タブレット	3584	ドット	24×24	MOS RAM	最大8K字	16	
日電 N7872-31	カラー	ドット リフレッシュ	40×20行 (17×18) 32×15行 (24×24)	24×27行 (17×18) 18×21行 (24×24)	14″	MOS RAM	可	タブレット	3584	ドット	17×18 24×24	MOS RAM	最大8K字	16	
日電 C5310	高解像型	TV スキャニング	26×16行	24×16行	12″	スキャンコンバータ		タブレット	2205	ドット	17×18		2961/3969	4	
昭和情報機器 S820/S821	高解像型	ドット リフレッシュ	32×20行	——	14″	MOS RAM		多段シフト	3072	ドット	32×32	ディスク	最大16K	64	グラフィック機能可
ゼネラル KK202		コード リフレッシュ	18×24行	16×13行 ×2段	14″			タブレット	2791	ドット	16×16		4K字	8	
三菱 M2365	カラー	ドット リフレッシュ	32×12行	12×32行	14″		可	タブレット	(*1) 4096	ドット	15×30	ディスク 又は MOS RAM	最大16K	16	(*1) 内の256字は単漢語入力用野線，グラフィック機能可。
沖		ドット リフレッシュ	32×16行	——	16″	コア		多段シフト	2600	ドット	18×18	UIコア	最大2.5K	1	
沖 OKISCOPE - 3010		ストレッジ	32×32	——	150×200 mm					光ディスクメモリ			3072		
沖 BT-1009		コード リフレッシュ	32×16	——	16″	コア			2578 (3240可)	ドット UIコア トランスレータ	18×18				
日立 H9915		リフレッシュ	40×12行	——	14″			タブレット	3072	ドット	16×18	ディスク	最大12K	12	
富士通 F6570		ドット リフレッシュ	32×16行 32×16行	——	17″	スキャンコンバータ チューブ		多段シフト	7680	文字母型式	活字文字	回転文字円板	最大8K	4	
	ツインネック	ドット リフレッシュ	64×16	——	20″	スキャンコンバータ チューブ		多段シフト	7680	文字母型式	活字文字	回転文字円板	最大8K	4	

メーカ（型名）	表示部							キーボード		文字発生部				最大設置台数	備考
	形式	表示形式	表示文字数 横書	表示文字数 縦書	画面の大きさ	画面メモリ	ライトペン	入力方式	入力文字数	方式	文字パターン	記憶素子	収容文字数		
東芝 DDS150		コード リフレッシュ	20×20行		16″	ワイヤメモリ				ドット	22×24	磁性線 ソレノイド	2.5K字	1	
東芝 DDS250K		ストレッジ	32×26		11″	—				ドット	22×24	磁性線 ソレノイド	4K字	1	
東芝 SB8102			2944		19″					ドット	32×32				
東芝 SB8103			832		11″					ドット	22×24				
三菱 M8210			32×4行					タブレット		ドット	32×16		最大16K	1	
東レ T8582C	カラー	リフレッシュ	30×20行		20″			多段シフト	3240	ドット	24×24		4K字	1	
東レ T8582		リフレッシュ	30×20行		14″			多段シフト	3240	ドット	24×24		4K字	1	
東レ T8589		ストレッジ	30×20行	—	11″			多段シフト	3240	ドット	32×32	ホログラム	最大10K		
東レ T8583			720		20″					ドット	16×18				
東レ T8580		ストレッジ	74×54		11″					ドット	32×32	ホログラム	7000 (10000可)		
昭和情報機器 S830			640		14″					ドット	32×32				
学研 GKS121			16×8							ドット	24×30		5000		
東洋通信機 D1501			32×8	—	12″ (P—37)	MOS IC	可	多段シフト		ドット	18×18		最大4000		フォーマット，棒グラフ表示
日本電子産業 JKM3100S		蓄積管型	32×16	—	10″ (P— 1)			多段シフト	2688	母型方式 FSS+PM			最大10000	32	
日本電子産業 JEM2420			18×8	—	16″ (P—24)			多段シフト	2688	フライングスポット		半角板	最大3000	3	
日本ユニバック 3550			384	—	130×210㎜					ドット	30×15				
ワング・コンピュータ2226A			42		9″					ドット	16×18				

ある。この方式では一旦パターン・メモリに記憶してしまうと新しい文字を発生させるまで文字発生装置は必要なく、容易に文字発生装置を幾つかのディスプレイ装置で兼用することができる。

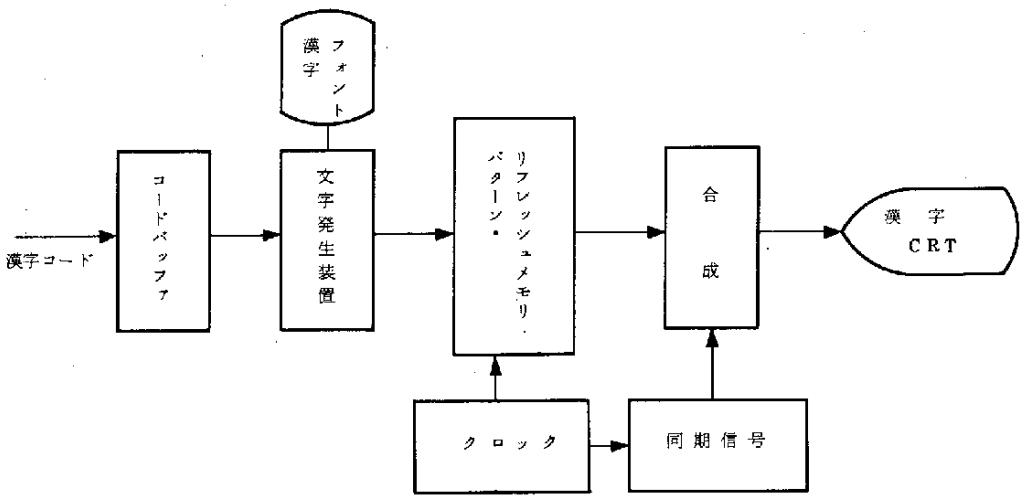


図２－１５　パターン・メモリ方式（パターン・リフレッシュ）のブロック図

2.1.2　日本語情報処理の適用分野

日本語情報処理用入出力機器が実用化されて、日本語（漢字）の処理がコンピュータで行なえるようになってきた。しかし、日本語情報処理は通常の英数・カナを使用する処理と比べて、字種が多く字形も複雑なことにより、ハードウェア、ソフトウェアとも高価なものとなっている。コンピュータの出力を漢字かなまじりで行なうことは我々、日本人にとって大変望ましいことであるがコストの面で、全ての業務に適用することができない。カナ文字によってデータ処理や出力を行なった場合に、漢字データが持った正確さが必要な業務、コンピュータで漢字が使えなかったために、人手により作業を行なってきたが、コンピュータで処理することができるようになり作業のスピード化や作業環境の改善などを行なえる業務、漢字による出力がサービス向上につながって営業成績にも影響するような業務などが漢字処理の対象として適当であると考えられる。

(1)　編集・印刷業務のＥＤＰ化

グーテンベルグ以来、印刷の技法は活字を用いた活版印刷が中心であった。この活版印刷においては、最初に文選が行なわれる。つぎに、文選の工程でひろわれた活字を、割付け（レイアウト）に従って体裁を整えていく。植字あるいは整版といわれる作業がある。この２つの作業は一見、単純であるが、かなり熟練を要する作業であり、これらの作業に従事する労働者は年々減る傾向であり、新しく養成することが困難となっている。また、活字は鉛を主体とした合金で作ら

れているため、それを溶かすための熱、それから発生する鉛蒸気、機械的な騒音、直接人手による作業、活字自体の要量など、鉛公害といわれる悪労働条件の問題がある。このため、印刷業界では古くから脱活字化または鉛熱を使わないという意味のコールド・タイプ・システム（CTS）が提唱されてきた。このため、写真植字機がつくられ、さらに電算植字システムが誕生するに至った。

出版印刷関係の電算写植は基本的には「データ作成」「自動編集」「版下作成」の3段階から構成されている。データ作成は原稿を漢字穿孔機で作成して、原稿ファイルを作成する。校正用にはモニタ出力して、それに対する赤字校正を行ない修正を完了する。自動編集では、棒組編集、割付編集の指示を与え、シュミレーションを行なって、組版処理を終了する。この自動編集は、ほとんどソフトウェアで行なわれている。自動編集用のソフトウェアは字詰用、割付用、体裁処理用などから構成されている。字詰用では棒組に必要なパラメータを指定することにより棒組を行ない、とくにルビ付漢字にその力を発揮するものである。校正用では、原稿の削除、入替え、追加などを処理し、割付け用では割付けにおいて段の位置を自由に指定でき、割付けのやり直しを容易に行なうことができる体裁を整えるためのものでは、頁の指定、ノンブル、柱の揃えの指定を行なうことができる。組版処理が終了した後に、その結果に従ってポジフィルムを作成・現像し、原版作成に使用する。

この種のアプリケーションとして、新聞原稿収配信・印刷システムがある。これは一般の編集印刷業務よりも高速性が要求される。ニュースの生命は速報性にあり、それは直ちに新聞紙面に反映されなければならない。ここでは、電子式写植機とCRTディスプレイなどを使って、新聞制作工程を機械化している。工程を大別すると集信システムと編集および組版写植システムとになる。まず、集信システムは別の地域からの出稿記事を一度紙テープに穿孔するか、または直接打鍵で伝送するもので、この分野は、漢字を含む日本語情報の機械処理をはじめて実現したものである。それは、いわゆる漢テレによる記事集配信システムであり、漢字テレタイプライタ（送信側）と漢字テレタイプライタ（受信側）とを通信回線で結んだ通信システムである。これは、やがて日本語情報システムの基礎となって現在も直接・間接的に大きな影響を与えている。

編集および組版写植システムでは、上記のようにして他の地域から集められた記事と自社取材の出稿記事とをコンピュータに入力して漢字プリンタにモニタ・プリントを得る。これを校閲して赤字訂正し、整理・編集を行なって記憶媒体に蓄積する。この校閲作業においては、CRTディスプレイを用いて訂正処理が行なわれる。続くレイアウト作業では、蓄積された記事情報を読み込み、小組、校閲、前組、大組を経て1ページのレイアウト出力を行なう。この編集処理と組版が新聞作成システムの中で最も中心的な部分である。完了した組版情報により出力する情報を決定し、写植装置を使ってフィルムに印字して現像処理後に原版が作成される。編集および組版においては、一般の編集および組版業務と似ているが、新聞特有の組版処理と時間に関するシビアな要求があるということで異なっている。

新聞原稿収集印刷システムは現在、地方新聞社ではすでに全面的にこのシステムに移行した所も多いが、

大手の新聞社では、家庭欄や、日曜版、選挙速報などに利用されているものの、一般の記事などには、とび込みの記事などの処理がありシステムが複雑になるために適用されていない。

また、一般の編集・印刷業務の機械化は、漢字入出力装置自体が高価なために、組合を作って共同で運営している場合が多いようだ。

(2) 一般EDPへの漢字の導入

この種のアプリケーションは、従来、カナなどによりEDP化されていたものを、カナでは見にくいということからカナを漢字でおき換えたものである。アプリケーションそのものとしては、従来から存在しているものである。この分類におけるアプリケーションは、事務処理の分野のものが多い。人事管理、顧客管理などでは、従来は人名とか住所（地名）をカタカナで入出力していたが漢字入出力装置が利用できるようになり、本来の漢字で出力するようになった。販売・資料あるいは統計業務といった分野では、商品や物品の名称、その属性、調査項目のデータなどを漢字で表わすようになった。また、レセプト（健康保険業務における診療報酬請求明細書）の作成では、患者の氏名・病気などは漢字で表記しなければならないようになっている。そのため、EDP化した場合にも、これらの部分はフリガナをつけたりあけておき、後から、人手で書き込まなければならなかったが、漢字処理装置の出現ですべてがEDP化できるようになった。これと同じような特徴をもっているのは、株式会社の株主名簿である。これは登録してある名簿台帳と同じでなくてはいけないところから今までEDP化することができなかった。このような業務もコンピュータで漢字かなまじり文の処理ができるようになり作業を高速化できるようになった。

この種のアプリケーションの例としては、他に各種証明書交付事務、物品管理、貨物輸送管理、証券代行業務、ダイレクト・メール、不動産流通情報提供業務などをあげることができる。

(3) 漢字情報のEDP化

この分野に入るアプリケーションは、漢字データ・ファイルを作成しておき、このファイルに対して情報検索やファイルの更新および管理を行なうものである。この種の適用業務としては生命保険会社における顧客管理のための漢字名寄せシステムや地方自治体の住民管理などがある。これらは人名・住所等をあつかうため、漢字の字種を多く必要とする。名寄せ業務においてはカナ文字で姓名ファイルが作成してある場合は、カナ姓名のほかに生年月日を加えて探索しても名寄せ率を95%程度しか期待できないといわれる。そのためさらにサブキーとして住所などのほかの要素を加える必要がある。これに対して、漢字姓名ファイルを使うと、姓名と生年月日で、ほぼ完璧な名寄せが期待できる。また、住民管理システムでは、1つの住民基本ファイルを作っておけば、市民台帳、各種証明、人口推計・予測などの各種統計、各種通知書など多くの用途に使え、しかもインプットやメインテナンスは1つの基本ファイルに対して行なえばよいように効率がよい。この種類のアプリケーションは、ほかに、住民記録管理、各種台帳・議事録検索、文献検索、判例検索、特許情報検索などがある。このような検索システムでは、大量の漢字データを使用するため、このデータ作成が問題となる。一般に、漢字データ作成は英数・カナ系のデータと比べて入力コストがかなり高くなり、入力業務も専門オペレータを必要とするなど特殊な性

格を持っている。このため、文献検索データなどの共用できるようなものについては、公共の機関でデータを作成して、安価にデータを供給することが考えられる。

また、この分野では、漢字ディスプレイを用いれば、オペレーションの指示や選択項目を漢字で表示できるため、業務担当者や顧客が直接システムと対話できマン・マシン・インタフェースが向上し、窓口などでの待ち時間を短縮できサービスの向上に図れる。文献、文書などの情報検索などでは、漢字キーワードを使用することにより検索精度が上がり、索引誌、配布リストなどの最終アウトプットが直接得られるなど、漢字ディスプレイを用いることによって、日常言語に近い質問応答が可能となり、検索が極めて容易となる。

○　事例1（信託銀行）

主要信託銀行では、証券代行業務に漢字処理システムを導入している。証券代行業務とは、株式を発行する会社に代わって株式に関係する事務を代行することである。ここで言う株式に関連する事務とは株主の会社に対する権利、つまり、利益配当金受領・経営参加・新株引受権取得等を確定し、個々の株主に対する各種書類を作成・送付することである。証券代行業務での日本語情報処理の対象は、主に株主に対して送られる各種書類のあて名印刷である。この導入の過程についてみると、まず最初、あて名処理はステンシルを使って行なわれていた。この際、送付書類とステンシルの対応づけは人間がチェックし、ステンシルで印刷していた。これでは、スピードが遅く、作業者も熟練者を必要とし、まちがいも多かった。

次に使われたのが、スクリプト印刷機である。これでは、あて名と書類との対応づけは簡単になったが、字がかすれたり、つぶれたりしてあて名が読みにくく、速度があまり速くなかった。そのため、コンピュータであて名を印刷することが考えられた。カナ文字であて名を印刷した場合、このことから、誤配送があってはこまること、漢字を使用できるようにすれば、あて名以外の処理にも使えることから漢字処理装置を導入した。導入してみて、正確さ、出来上り、スピードなどの点で満足されている。あて名以外の処理として、増資の際の割当名簿の作成と、株主の管理の一手段として、株主名簿をディスクに記憶しておき、この台帳と印鑑書（住所・氏名も書かれている）の管理とに使われている。

今後の漢字処理の応用として考えられているのは、オンライン化して株主情報の検索や株券の印刷、パンフレットの配布のあて名書き（現在、ステンシルを使用）、預金者の利率の通知のあて名書き（現在、カナ）などがある。

○　事例2（高島屋）

高島屋では、ダイレクト・メールのあて名の印刷のために漢字処理システム（現在、FACOM6504A漢字プリンタを中心に構成）を導入して約4年経過している。関東用には、本店に機械を設置し、関西用には大阪で外注して処理している。導入の目的としては、多くのダイレクト・メールの中から顧客に封筒の中身をより多く見てもらうためのイメージ・アップと回収率を上げることにある。顧客数は100～200万人程度で、ダイレクト・メールの場合あまり即時性は要求されないのでバッチ処理で行なわれている。

ダイレクト・メールの発送は毎週行なっており多い人で月に1回程度、普通は3,4カ月に1回程度であり、条件により送る顧客を決めている。データ作成はすでに外注で昭和47、8年頃から始められ、その当時はマスタの大きさが50～60万人分であった。現在では毎週1～2万件の新規登録があるが、削除、修正などファイルのメンテナンスを行なっているのでマスタはそれほど大きくならない。使用文字種としては6,000字種程度を持っている。関東と関西では必要文字種が異なり関東では約4,000字種、関西では約8,000字種必要としている。使用頻度の少ない文字についてはカナで印刷したり、打たないで手で書き入れるなどの処理を行なう。あて名印刷までの顧客抽出処理は、抽出条件を企画部門で決定した後、EDP部門で処理している。マスタ・ファイルのメンテナンスは2カ月に2～3回程度一括して行ない、削除は随時行なっている。年に3,4回顧客台帳を出力しているが、これに時間がかかるため、将来オンライン化して台帳の出力をなくしたい希望がある。漢字システムを導入して、回収率があがった。

○ 事例3（倉敷中央病院）

倉敷中央病院では、昭和45年に委託形態での給与計算、給食カロリー計算を手初めに、EDP化に積極的に取組み、昭和47年にコンピュータを初導入し、現在では業務拡張に伴なってNEAC2200/250（2台）からNEAC・ACOS/300および500へと移行中（昭和52年12月現在）である。処理業務は、即時処理による入院、外来窓口会計業務、一括処理によるレセプト業務、医事統計、薬剤管理、給食管理、検査業務、病床管理などの病院事務に関する各業務が稼動している。これらのうち漢字処理が適用されているのは、レセプト業務と医事統計業務の一部である。

レセプト（診療報酬請求明細書）の特徴として、現在の医療保険制度では、患者の氏名、保険の記号番号、病名をカナ文字で提出することが認められていないことから、これまで多勢の職員を投入して手書きしてきた。このためコンピュータでレセプトを作成する場合、英数字カナ文字で記入して良い項目だけをコンピュータ処理し、漢字項目は符号やフリガナを印字しておいて後でマスタ台帳より探して手書きしなければならなかった。レセプトの作成は、月末締め切りで7日提出という短期間で処理しなければならず、そのわずか1週間にぼう大な要員と人件費を費やすことから、当病院では漢字処理装置を導入することになった。

漢字化にあたっては、まず病名マスタの作成と患者台帳ファイルの作成にとりかかった。病名は約15,000件といわれ、更新処理も多少あって随時行なっている。病名マスタは病名コードと漢字仮名病名からなり、レセプト用入力データでは病名コードが使われて、出力時に漢字仮名病名が印字される。患者台帳は、患者名と住所に漢字化がはかられ、とくに初診患者・入退院患者の更新は毎日行なわれている。住所については住所コードによって処理される出力時に漢字出力がなされる。マスタ・ファイルの作成までは外注で、日毎の更新は内部でデータ作成している。

当病院では出力装置の要件として、レセプトの印刷用紙が各保険機関によって定型が異なる

ためにフォーマットの融通性が要求される。また印字方式として、インクの改ざん性が重要である。これはレセプトの内容が書き換えられないようにするためである。

当病院でのレセプト処理量は、毎月22,000件で、病名は 51,200件の使用がある。

○ 事例4（地方自治体）

群馬県太田市では、全国に先がけて"住民情報漢字処理システム"の導入を図り、当市の将来の「総合情報管理システム」の確立をめざし、より一層の住民福祉の向上に十分な役割を果すべく、システム開発に取組んでいる。当市では昭和44年から税業務をはじめ、国保、水道、国民年金と段階的に大量反復作業を中心としてコンピュータ利用が進められてきた。

その後、住民情報管理システムの確立をめざして、調査検討を行なった結果、他の自治体においてカナ文字によるコンピュータ利用に対する批判があることなどから、漢字処理システムの導入を決定した。この時点でそれまで3社に委託していた計算センターを1社にし、計算センターの協力を得ながら漢字調査を進め約1万字種を決定するとともに、住民基本台帳をもとにした住民マスタ作りを行ない、昭和51年末をもってマスタが完成した。52年1月以降通常の異動処理が行なわれている。

住民マスタのなかの漢字項目は、住所、世帯主氏名、本人氏名、さらに肩書き欄を設けており、その他、国民健康保険（給与）関係において、レセプト処理が図られるように基礎的な項目が設けてある。

すでに漢字処理で打ち出されたものとして、市民税の申告書、台帳、市民課の市民台帳、人口ピラミッド（姓 300傑）、国民健康保険の保険証、その他各種業務における台帳などのチェック用リストがあり、活用分野も除々に拡大されつつある。今後、対外的な期待効果としては、漢字処理を中心とした窓口での住民票の交付、証明書の発行などを、また即時処理としてのデータベース構築を基本とするオンライン・システムの活用を目指して端末装置を導入し、市民サービスはもとより効果的な行政の展開を目標としている。

○ 事例5（明治生命）

明治生命では、従来は顧客宛の諸リストのうち数字項目だけを機械プリントした上で姓名、住所はアルバイトなどによって手書きで作成していた。顧客から契約内容の照会を受けた場合には、本社にある顧客姓名の索引カード（被保険者名、契約者名から証券番号の割り出しができる単票式のカードで24台の回転式カードケースに約 1,000 万枚を整備保管）を手作業で抽出し、その証券番号から契約内容の検索を行なっていた。このような方法では、手作業のため顧客の照会や問合せに対する回答に時間がかかり、また作業自体も慣れるまでに一定期間を要するとか、多くの人員を必要とするとか、立作業だから疲労するなど労務管理上の問題を伴っていた。そのための解決策として開発されたのが漢字名寄せ索引システムである。

この漢字索引名寄せシステムは姓名および生年月日をキー項目として、同一契約者または被保険者の複数の保険契約をひとまとめにして検索するもので、漢字ディスプレイ・キーボードから顧客の姓名と生年月日を入力するだけで契約情報（加入している保険の種類、期間、満期、

保険金額、その他）をディスプレイ画面上にカラーで即座に映し出すシステムである。

表2－14　日本語アプリケーション例
（実動中または開発中・計画中も含む）

官公庁関係機関	行政管理庁法制局	法令検索システム（省令以下は各省システムとなる）
	裁判所	判例検索システム
	大蔵省（主計局）	国家議事録（予算）検索システム
	法務省	不動産登記情報システム
	通産省（情報管理課）	情報検索システム（経済情報・法令）
	特許庁	特許情報検索システム、特許出願事務処理、商標検索システム
	総理府（統計局）	国勢調査・各種統計資料
	外務省	外電・通達文書管理システム
		国会議事録・調査等の情報検索
	厚生省	社会保険（徴収・裁定・支給事務）
	運輸省	自動車登録
	国税庁	名簿リスト
	防衛庁	庁内資料情報検索と印刷
	内閣調査室	総合情報管理システム（オンライン情報検索システム）
	文部省	人事関係事務処理、庁用品（消耗品）管理事務
	文化庁国立国語研究所	用語用字実態調査研究
	電子技術総合研究所	日本語索引情報検索システム
大学・	広島大学 原爆放射能医学研究所	情報検索
	青山学院大学	図書目録・索引、事務処理
	国際基督教大学	漢字CAI
	東海大学	図書管理
	早稲田大学	図書管理
	慶応大学	同窓生名簿管理
	東邦大学	名簿管理
	実践女子大学	事務処理
	共立女子大学	事務処理
	東京農大学	事務処理
	日本体育大学	事務処理
	京都産業大学	図書管理
	茗渓会	名簿管理

大学・公的機関・地方自治体	日本科学技術情報センター	文献速報編集システム、文献検索システム
	日本放送協会	放送番組製作管理、選挙開票速報
	(財) 日本特許情報センター	特許情報検索システム
	全国共済農業協同組合連合会	共済契約証書
	(財) 日本情報処理開発協会	日本語情報文献検索システム
		カナ漢字変換システム
	国立国会図書館	索引編集システム、国会会議録索引システム
	兵庫県立図書館	図書目録作成
	国立国文学研究資料館	文献資料管理、検索・索引システム
	倉敷中央病院	レセプト、病院事務
	岩手県立病院	レセプト
	国立がん研究所	レセプト
	東京医科歯科大学	レセプト
	岐阜市民病院	窓口会計事務、レセプト、薬剤在庫
	北品川総合病院	レセプト
	大阪市役所	住民記録
	太田市役所	住民情報管理、健康保険、税務処理
	静岡県三島田方行政情報センター協議会	住民情報管理、健康保険、選挙人名簿
	大阪府警察本部	自動車運転免許
	富山県警察	自動車運転免許
	愛知県警察	自動車運転免許
	日本機械学会	会員管理システム
	日本化学会	会員管理システム
民間企業	住友信託銀行	漢字情報処理システム (証券代行業務・あて名印刷)
	東洋信託銀行	証券代行業務システム
	三菱信託銀行	あて名印刷、証券代行業務
	明治生命保険	総合漢字情報処理システム (顧客印刷、統計資料作成、保険契約内容照会)
	旭電算製版センター	印刷組版
	オール出版印刷	新聞・商業印刷、組版
	学習研究社	電算植字編集システム、印刷組版、漢字ラベル自動発行
	竹田印刷	商業印刷組版
	千代田コンピュータ印刷	印刷組版

民間企業	東京ジェー・ビー	印刷組版
	凸版印刷	一般書籍文字組版、株券氏名刷り
	カシヨ印刷	印刷組版
	日本コンプ	印刷組版
	ペテイ	印刷組版
	東京放送（TBSテレビ）	テレビ選挙速報
	日本教育テレビ（テレビ朝日）	テレビ選挙速報
	フジテレビ	テレビ選挙速報、野球放送
	朝日新聞社	漢字集信システム（原稿受信） NELSON（新聞編集、写植・組版）
	上毛新聞社	新聞組版
	中日新聞社	漢字集信システム（原稿受信）
	日刊スポーツ印刷社	NEPS（新聞案内広告）
	毎日新聞社	漢字コード変換システム（原稿編集・組版） Mainichi Automated Comosing System（原稿編集、組版）
	山梨日々新聞社	新聞原稿編集・組版
	読売新聞社	高速漢字通信処理システム（原稿集配信）
	大倉商事	和文自動組版ディジセット・ビデオコンプ・システム（出版印刷・情報検索）
	共同通信社	ローマ字・漢字かな変換処理システム（電文集配信）
	日本経済新聞社	電算写植システムANNECS
	神奈川新聞社	新聞編集、組版
	高知新聞社	新聞編集、組版
	日本海新聞社	新聞編集、組版
	北日本新聞社	新聞編集、組版
	サンケイ新聞社	SUCCESS電算写植システム
	佐賀新聞社	新聞編集、組版
	中国新聞社	新聞編集、組版
	西日本新聞社	新聞編集、組版
	関西テレビ放送	番組編集
	CTS大日本（大日本印刷）	一般商業印刷、組版
	ダイヤモンド社	雑誌編集、組版
	黒船印刷	印刷組版
	三菱商事	人事情報

民間企業	丸紅	人事情報
	三菱電機	人事情報
	三井金属	人事情報
	キャノン	在庫管理、個人記録
	日立製作所	日立ドキュメンテーション・エディット・システム（ドキュメントの組版）
	北海道拓殖銀行	株式業務、会員名簿
	朝日生命	顧客管理、保険契約、その他
	三井銀行	人事管理、顧客管理、その他
	野村証券	人事管理、顧客管理、その他
	大成建設	帳票印刷
	日本出版販売	帳票印刷
	日比谷コンピュータ・システム	あて名印刷
	東京新聞社	集配信業務
	全国商品取引計算センター	あて名印刷（株式・DM）、外務員登録、医療レセプト、販売促進業務
	特許データセンター	特許情報検索、GIPSY工業技術情報
	松下電器産業	技術情報管理
	高島屋	あて名印刷
	日本ユニバック	あて名印刷
	紀伊国屋書店	文献検索、図書目録・索引
	アメリカン・インターナショナル・アンダーライターズ（A.I.U）	保険契約案内
	旭アド	棚札作成業務、ダイレクトメール
	鹿島建設	見積書、請求書
	下野新聞社	新聞編集組版
	長野コンピュータ印刷	出版編集
	デンショク	出版編集
	安田信託銀行	証券代行業務
	小野田セメント	
	資生堂	
	日刊福井新聞社	新聞編集
	東京出版販売	出版情報サービス
	帝国興信所	企業情報サービス
	日本テレビ放送網	テレビ選挙速報
	イトーヨーカ堂	棚札作成システム
	ダイエー	漢字入り価格表示カードシステム
	日本生命	社員名簿、顧客管理

2.2 オンライン日本語情報処理技術の動向

日本語情報処理の適用は、コンピュータ利用分野の拡大と利用形態の多様化に伴って今後ますます需要が高まると予測されている。その中で今日のオンライン化やコンピュータ・ネットワークの進展にともなって日本語情報処理のオンライン利用も期待されている。これまでの日本語情報処理は、印刷や新聞の編集をはじめとして事務処理の漢字化まで、いずれもオフライン一括処理がほとんどであった。そこにはニーズはありながらも漢字入出力装置の高価格化に最大の原因があることは言うまでもない。

そのためオンライン処理ばかりかオフライン処理も含めて日本語情報処理のユーザは官公庁や大企業レベルに限られてきた。中小企業レベルでは高価な漢字化には EDP 化以上に導入機運は少ないはずである。

官公庁におけるオンラインでの日本語情報処理は、法令や判例および国会議事録、特許情報などの情報検索システムに適用されている。また地方自治体では、住民管理の名寄せシステムのオンライン化への機運がみられ、一般企業では経済情報や文献情報の検索システムと、顧客管理における名寄せシステム、テレビの選挙速報、新聞原稿の自動組版・編集処理システムなどにオンライン化がみられる。

2.2.1 オンライン用漢字入出力装置

(1) オンライン用入力装置

前節に述べた漢字入力装置の分類、各種入力方式の特徴概説は、とくにオフライン／オンラインとを問わずに現状分析を行なった。ここではオンラインで使用する入力方式という観点から現状をみることにする。

漢字入力装置の多くは、主にデータ作成を目的としたオフライン装置として使用されるが、コンピュータとの接続を考えた場合にそのインタフェースが必要である。換言すればインタフェースを作りさえすればオンライン装置としての利用も可能といえる。オンライン用入力方式という立場から前述の入力方式を、フルキーボード方式（和文タイプ方式、タブレット方式も含む）、マルチ・ストローク方式（カナ文字入力方式、カナ漢字変換方式などを含む）、コード入力方式（数字コード方式、連想コード方式）、その他の方式、とに分類してみる。

フルキーボード方式は、漢字テレタイプの入力けん盤を代表的に、和文（邦文）タイプライタなどに見られる方式で、けん盤の一面に全漢字を配置するという特徴を持っている。そのためけん盤が大きく、操作のための動作（目視動作や腕の動作）が激しいことから、オペレータの疲労度が高いこと、文字の配置を覚える訓練が必要なことなどに難点があると指適されている。しかし装置自体の操作法は簡単で、該当文字を探しさえすれば確実に入力できることからオンライン入力装置として効果的な面もある。とくにタブレット方式に分類されるペンタッチ入力などは簡易な入力方式として素人向けである。

マルチストローク方式（多段入力方式）は、フルキーボード方式の" けん盤上の文字配置を記

憶しなければならない”ということに対する反動として考えられた方式で、コンピュータの力をかりた各種の方式がある。その中で2打ないし3打で漢字1字を選択入力するカナ文字入力方式に分類される装置は、入力操作にコンピュータを利用していることから、即オンライン化できる入力方式である。また、漢字1字ごとに多段階の操作を必要とするこの入力方式に、漢字の単語・熟語ないし短い文単位に入力できるようにしたカナ漢字変換方式も、オンライン入力する情報量と一致することから有効な方式とされている。この方式も同音異義語の処理があることから対話式で利用されることが一般的である。なお自然言語処理的な複雑な変換方法を採用して一意的に変換するのは特殊なオンライン入力に限られよう。

コード入力方式は、オンライン入力装置としてもっとも簡易なキーボード構成であることからオンライン向きといえる。しかしこの方式もコードから漢字（かな混り文）への変換にコンピュータを使用する。もっとも難点とされるのは、取扱う全漢字を符号化するために、漢字とコードとの対応表をオペレータが記憶しなければならないことである。そのため字種を極限したり、覚えやすい符号を対応させるなどの工夫をこらせば、入力速度の点ではもっとも優れているという評価を得ている。

その他の方式では、音声入力方式がオンライン入力にもっとも期待されているが、現状の技術レベルでは実用化は無理である。

(2) オンライン用出力装置

漢字出力装置として、各種の分類方法に基づいた特徴については前述したが、ここではとくにオンラインで使用する場合の出力方式という観点から現状をみることにする。

漢字出力装置は、ソフト・コピー装置としての漢字ディスプレイと、ハード・コピー装置としての漢字プリンタという分類がなされるが、一般にオンライン処理を目的とした端末装置においては、入力装置とともに表示装置が欠くことのできない基本的な装置である。漢字を使う端末装置の場合、とくに大量の文字フォントを含む表示装置の性能およびコストが、端末全体の使いやすさ、作業能率およびコストを決める重要な要素である。

漢字ディスプレイおよび漢字プリンタをオンラインで使用するには、漢字出力のための文字発生装置の機能と構成法が重要である。そのためオンライン用出力装置としての出力方式は、そのまま文字発生方式のいかんにかかってくる。

文字発生方式の分類とその特徴はすでに述べたとおりであるが、オンライン用という立場から再度分類を試みると、活字打鍵方式、ディジタル方式、アナログ方式、その他の方式とに大別できる。

活字打鍵方式は、和文（邦文）タイプライタ式に1字ずつ活字をハンマで打って印字する機械式のものであり、漢字プリンタのもっともオーソドックスな方式である。漢字テレタイプライタが漢字印刷電信装置として使用されたようにオンライン出力装置して可能であるが、低速度で印字が不ぞろい、騒音がひどいなどの欠点がありオンライン向きでない。

ディジタル方式には、ドット・マトリックス方式、ラインドット方式、ストローク方式があり、

いずれも文字フォントをディジタル化して記憶する方式である。これらの方式では多字種の漢字を扱うことから記憶容量がぼう大になり、メモリ・コストが高くなる、文字発生速度が遅くなるなどの欠点を有するが、多重処理が容易であるためもっともオンライン向きといわれる。ただしストローク方式は、前二者のドット方式にくらべ表示・出力の走査方式が異なって、その変換が複雑である。

アナログ方式は、フライング・スポット方式やビデコン方式など光学的に文字発生する方式である。出力文字の品質はもっとも良好であるが、装置の規模が大きいこと、多重処理に制限を受けるなどの難点があって、オンラインには向いていない。

その他の方式としては、部分パターンや構成要素の合成方式などがあるが、これらは個々の要素パターンをディジタルないしアナログ的に記憶して、出力時に合成印字する方式である。ディジタル合成方式ならばオンライン処理としても有効である。

以上が文字発生方式からみたオンラインに対する可能性である。次に印字方式、記録用紙の特徴からオンライン利用を検討してみると、端末装置の運用面から普通紙への印字と、装置管理の容易な乾式電子写真方式、ワイヤピン方式、インクジェット方式などが採用されそうである。

また漢字ディスプレイの表示方式は、高価な文字発生装置を共有するという面から、パターン・メモリ型がほとんどであり、表示機能の面からはさらにパターン・リフレッシュ型と蓄積管型の表示装置とに使い分けされている。

(3) オンライン端末構成

各メーカから発表されている漢字入出力装置の中からオンライン処理が可能とされる装置をいくつかとりあげて、その構成の特徴をみると以下のようである。

a）ゼネラルの漢字編集ターミナル

入力装置はタブレット型のもので2,862字種の入力が可能。漢字ディスプレイの1画面には18字 × 24行 = 432字が表示でき、各文字は 16 × 16 のドット・マトリックスで構成されている（ドット・マトリックス方式）。制御編集用にマイクロ・コンピュータが内蔵され、接続ターミナル数は標準8台となっている。

b）三菱・ユニバックの漢字ディスプレイ・ターミナル

入力装置はライトペン式の入力キーボード（フルキーボード）で、最大16,384字種の入力が可能。漢字ディスプレイには、256文字から最大3,200文字まで表示可能で、赤・緑・白の三色をカラー表示できる。制御部にはMELCOM−70ミニコンピュータが採用されており、最大16台まで制御可能になっている。文字発生はドット・マトリックス方式である。

c）東芝の漢字人名検索用ターミナル

入力装置はカタカナ・キーボードであり、姓名の読みカナを入力して漢字姓名に変換するカナ漢字変換方式を採用している。姓名の変換辞書には、それぞれ 6,000種が外部記憶装置（2.4Mバイト）に登録されている。検索結果は漢字ディスプレイに表示される。制御用にTOSBAC−40Cミニコンピュータが使用されている。

d）日立の日本文編集処理端末

入力は回転和文タイプ式（メニュー選択型）の小形漢字入力装置を使って 2,600 字種の入力が可能（外字はコード入力）。漢字ディスプレイには 40字 × 16行 ＝ 640 文字が表示できる。文字パターンはドット・マトリックスで外部記憶に記憶され、漢字プリンタとディスプレイで共有する。制御用にミニコンピュータが使用されている。

e）日本ＩＢＭの漢字映像表示装置

入力は英数字キーボード、漢字キーボードいずれも接続可能。その他プログラム機能けん盤とジョイスティック（操作桿）によって画面の制御が行なえる。映像画面は 864 × 864 のドット・マトリックスになっており、漢字かなの他図形表示も可能になっている。システム／7 の制御のもとに最大 14 台の漢字映像表示装置を接続することができる。

f）DEC・学研のオンライン漢字処理システム

漢字ディスプレイには 16字 × 8行 ＝ 128字の漢字が表示できる他に、英数字のキャラクタ・モードの文字が同一画面に表示できる。文字発生は 30 × 24 ドットの文字フォントを外部記憶装置（ディスク）にもち、PDP － 11 ミニコンピュータの制御のもとに、他のコンピュータ（ホスト）とのオンライン処理が可能になっている。ディスプレイ・キーボードは英数字キーボードであり、コンソール・ディスプレイと併用できる。ディスプレイには簡易なハード・コピー装置が装着される。

g）グラフテック社の漢字ディスプレイ端末

マイクロ・コンピュータを組込んだ漢字ディスプレイ端末は、漢字キーボードとディスプレイ画面から構成される。2,400 字種の漢字パターンを内蔵し、最大 16,774 字の漢字・数字・カナを発生できる。専用プリンタを接続することにより漢字プリンタとしてハードコピーが得られる。電通国際情報サービスの商用 TSS「MARKⅢ」用の端末として接続できる。

2.2.2 オンライン日本語情報処理の事例

オンラインで日本語情報処理を行なっているアプリケーション・システムをあげると以下のようである。

(1) 印刷・出版・新聞編集業務

この分野は従来は人手による作業がほとんどであったが、漢字入出力装置の発展によって電算写植システムが実用化した。電算写植システム自体はバッチ処理されるが自動組版システムのオンライン化と、編集校正システムのオンライン化が一部実動している。

ここでのオンライン用漢字入出力装置は漢字ディスプレイと漢字キーボードの構成である。編集や組版のソフトウェア規模は他のオンライン・アプリケーションよりも大きく、多機能を有する。また扱う文字種もかなり多く文字発生装置の規模は大きい。印刷用にはフィルム化して版下を作成するが、その文字品質は高度であり、字母アナログ方式が採用されることが多い。

そのためシステム構成は高価であるが、無鉛化－CTS 化のメリットは高く、他のアプリケーションを含めても、もっとも多く導入されている分野である。

(2) 事務処理の漢字化

これまで英数カナ文字で EDP 化されていた事務処理業務に漢字を導入し、サービスの改善と事務処理の能率向上を図っている。主に宛名印刷（DM ラベル）や医療請求業務、各種統計管理表作成業務などで、漢字化項目は姓名、企業名、住所、商品名などの固有名詞である。

ほとんどはバッチ処理形態で運用されるが、一部オンライン利用されている。それはマスタ・ファイルの管理をオンラインで更新処理する場合や、台帳管理上で名寄せや索引をオンライン処理する場合である。このようなオンライン処理では、ほとんどが漢字ディスプレイ中心の構成である。しかし端末数はそれほど多くなく、窓口受付業務以外では高スピード化も要求されず、品質などの要求も低い方である。

オンライン化は、どちらかというとバッチ処理で作成したデータベースやマスタ・ファイルの有効利用をはかる目的で行なわれていると言える。

(3) 漢字情報の EDP 化

これまで英数カナ文字の EDP 化が許されずにいた分野が、漢字入出力装置の出現で漢字 EDP 化へと進んだ。

官公庁、地方自治体などにおける住民記録や不動産登記業務、さらには特許、法令、判例、文献などの情報検索業務に漢字化が行なわれた。この分野ではバッチ処理形態よりも逆にオンライン処理形態での利用方法が多く、固有名詞情報に加え文章（文書）情報をも漢字を必要とする。とくに情報検索システムには、日本語情報が必須であり、即時性をも伴なって漢字ディスプレイによるシステム構成になっている。またソフト・コピーばかりか入手情報としてハード・コピーも要求されるため、ハード・コピー装置あるいは漢字プリンタ装置などが利用されている。

以上の3分野の適用を実際のシステム例からその特徴をみることにする。

① サンケイ新聞社の漢字処理システム

サンケイ新聞社では、新聞製作工程の合理化と近代化、そして無公害化と職場環境の改善を目指して、鉛活字による活版工程からコンピュータと写真植字機を組合せた電算写植システム、いわゆるコールドタイプ・システム（CTS）化へと変革をはかってきた。このシステムをサクセス（SUCCESS）と呼び、ミニコンピュータをネットワーク化した方式で、故障に強い、拡張性がある、廉価であるなど数多くの特徴をもっている。

サクセスの第1ステップのシステム構成は次のようである。

- メインコンピュータ ―――――― FACOM230－25　1式
- 処理用コンピュータ ―――――― FACOM U－200　2式
- デバイス制御用コンピュータ ―――――― HITAC 10Ⅱ　1式
- 入力ステーション（タイプライタ、紙テープ読取装置、ターミナル・コントローラ（制御ボード、フレキシブルディスク装置内蔵））―――――― JEM・NEAC製　1式

・ディスプレイ・ステーション（グラフィック・ディスプレイ装置、漢字キーボード、ターミナル・コントローラ）――――――――――――――JEM・NEAC製　1式

・プルーフ・プリント・ステーション（プリンタ制御装置、プルーフ・プリンタ、ターミナル・コントローラ）――――――JEM・NEAC製　1式

・フィルム・プリント・ステーション（プリンタ制御装置、フィルム・プリンタ、ターミナル・コントローラ、磁気テープ装置、紙テープ読取装置）

――――――JEM・NEAC製　1式

・モニタリング・ステーション――――――FACOM製　1式

サクセスでの新聞製作工程は、編集局で書かれた原稿が、図2－16に示すような流れで処理される。まず①漢字キーボードで紙テープにパンチされ、②それを入力ステーションでフロッピーディスクに変換する。この時点でコードコンバートや編集に必要な情報がつけられる。③ついで小組の必要がある原稿はディスプレイ・ステーションで小組処理がなされる。④つづいてプルーフ・プリント・ステーションへ流す。大組段階で流し組みとなるものは入力ステーションで処理されたままプルーフ・プリント・ステーションで校正刷を必要枚数とる。そして校閲され、⑤その指示に従って訂正、挿入、削除、移動などをディスプレイ・ステーションで修正する。その後、⑥面ごとにまとめて大組（編集処理）をする。面編集された後は、⑦フィルム・プリント・ステーションで印画紙にプリントされる。小組処理や面編集処理をせずに印画化も可能である。

漢字キーボードでパンチする際に、複雑な組版のためファンクションのパンチがしにくいものは、ディスプレイ・ステーションでエディット・パンチ・モードによって、原稿から直接フロッピーディスクへ変換される。サクセスのディスプレイ・ステーションは、小組、大組、校正、エディット・パンチ・モードのいずれも処理できる機能をもっている。それは校正段階でレイアウトに影響する訂正があったときは小組の機能が必要であり、大組段階で文章の訂正や削除が必要なときは校正機能がなければならないこともあるため、それぞれの機能を独立させたのでは効率が悪い。そのためディスプレイ装置はグラフィック・ディスプレイを採用している。

一方、モニタリング・ステーションの機能は、大阪本社、支局、国会その他予備線を含めて50ボー回線が28回線、共同通信から200ボー回線が2回線、これらの回線から漢字電信で原稿を受け、モニタ・プリントを平均2～3枚出力する。また電文の蓄積・検索の機能をもち、検索の際に電文を修正して、出力することもできるようになっている。

② 国立国文学研究資料館の漢字処理システム

国立国文学研究資料館は、国文学に関する文献・資料の調査研究、収集、整理などを行なうことを目的として、昭和47年に設立された国立大学の共同利用研究機関である。

現在、古典文学に関する資料は全国に60万点あるといわれ、このうち当資料館では3万点を収集しているが、今後とも年間5,000点の増加が見込まれている。また論文でもすでに6千点を収集している他、単行本や逐次刊行物も数万点を収集して、昭和52年7月から一般に

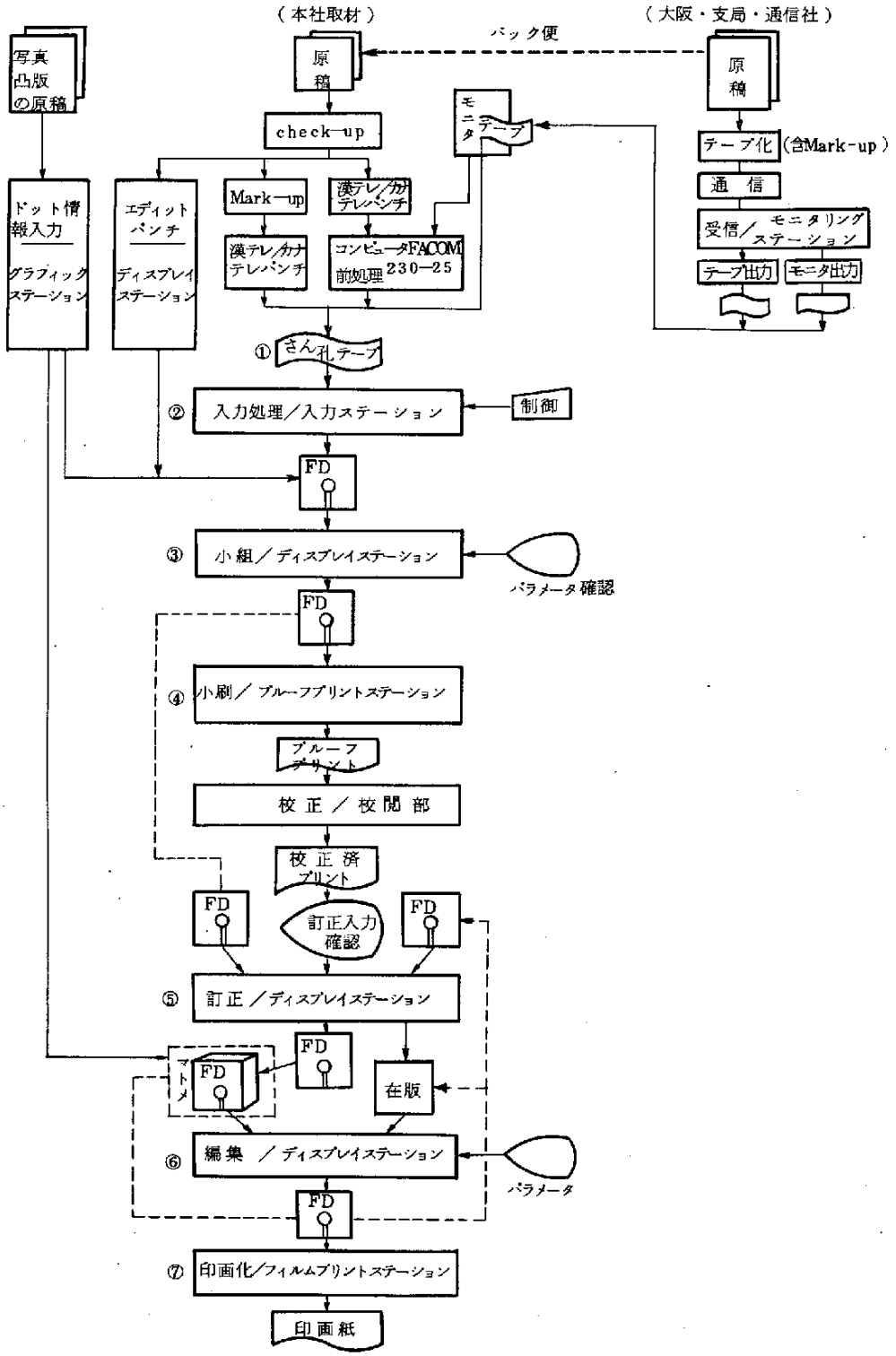


図２－１６　サクセスの新聞製作工程

も公開している。これら文献・資料の目録については、館内の研究情報部情報処理室が東大計算センターを利用して作成にあたっていた。しかし今後の情報量の増大にともない独自にコンピュータを導入して処理することになった。

昭和52年12月に大型コンピュータ HITAC・M160 II を中核とする漢字情報処理システムを導入し、その構成は漢字プリンタ制御装置H8191、漢字ディスプレイ装置H9915を2台、ビデオ・データ・ターミナルH9415 を3台などである。漢字ディスプレイ装置の1台は一般閲覧室に、もう1台をパンチ室に設置し、ビデオ・データ・ターミナルは1台を閲覧管理用に、2台を研究用に使用している。

当資料館ではこのシステムを使って、文献・資料、マイクロフィルムや単行本、逐次刊行物の目録作成を手始めに、オンラインによる情報検索や閲覧管理を行なっていく。その他にも次のようなシステムを開発する予定である。

a）文献・資料システム（文献・資料の書誌的事項に対する検索）

b）研究情報検索システム（文献への研究論文の書誌的事項の検索）

c）図書情報検索システム（当資料館で所蔵しているマイクロフィルムで保管してある文献、単行本および刊行物に対する受け入れ管理、問い合せ、貸し出し、返却など）

d）各種目録作成システム（マイクロ資料目録、研究論文目録、刊行物目録などの作成）

e）語い索引システム

オンラインによる情報検索システムについては、将来全国各地の大学に端末機を設置して、遠隔地の国文学研究者にも利用の便を図ることを計画している。

③ 学習研究社のラベル印刷システム

学習研究社では昭和40年にコンピュータを利用した編集業務を開始して以来、昭和45年にはメーカと共同開発した商業印刷用漢字システム、JEM－3850 電算植字システムを導入して、これまで各種のアプリケーション・システムを開発し運用している。その後 JEM 7300 漢字プリンタの導入、VIDEO COMP 500 電算植字システムの導入へと拡張して社内の出版・編集業務ばかりでなく、漢字処理に関するシステム全般のほか、ミニコンピュータのソフトウェア、漢字処理用のハードウェアの販売に至る幅広い活動を行なっている。

ここに紹介するラベル印刷システムは、昭和52年10月にディジタル・イクイップメント日本支社（DEC日本支社）から導入したもので、IBM370／148 のホスト・コンピュータとDEC の PDP 11／70 によるラベル印刷システムをオンラインで結んだ構成になっている。このシステムは、書籍や雑誌の配送用ラベル、伝票などを印刷するシステムで、学研第2ビル（東京・仲池上）にある大型コンピュータ IBM370／148 がまとめた発送・管理用のデータを、即時処理で学研平和島流通センター（東京・平和島）に送る。流通センターでは、これをミニコンピュータPDP11／70で受け、米国ブリントロニクス社製プリンタによって、バーコード、漢字、カタカナ、アルファベット、数字などの混合したラベル、伝票が打ち出される。これによって、これまで人手に頼っていた出荷が完全に自動化された。

このシステムのオンライン化には、DICAM（データシステム・インタラクティブ・コミニケーションズ・アクセス・メソッド）と呼ばれる新しいソフトウェア技法が利用され、ホスト・コンピュータには全く負担がかからず、通常の CRT ディスプレイ端末とみなしてラベル印刷システムが接続されている。

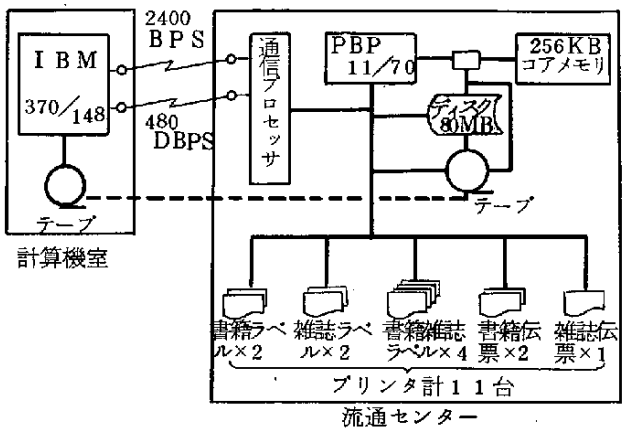


図２－１７　学習研究社のオンライン・ラベルシステム

④　広島大学原爆放射能医学研究所の大容量映像情報検索システム

広島大学原爆放射能医学研究所では、かねてから被爆の影響に関する正確な情報を収集・処理、保存するため「原医研データ・バンク」の構想が検討されてきた。昭和４６年３月より、生物統計学、疫学、社会医学研究部門と原爆医学標本センター（現原爆被災学術資料センター）とが協力して、その基盤となる「原爆被爆情報マスターファイル」の作成に関し、重点的・精力的に試行を重ねてきた。その後各関係機関とのデータ・リンケージが確立されることによって、経時的に被爆者に関する諸情報の把握ができ、その科学的研究や健康管理に十分役立てることが可能となった。そして「原医研データ・バンク」のより具体的な展開を計るために、その第１段階として、昭和４９年６月の NEAC コンピュータ・システムの導入決定とともに、ネットワーク的な被爆者情報をコンピュータ・システムにのせる最適なソフトウェア開発に着手した。

当システムは昭和５０年９月に至り、漢字情報処理を含むインライン型情報検索システムとして完成した。主な特徴は以下のようである。

a）　ホスト・コンピュータ（NEAC 2200／250B）と漢字処理用サブ・コンピュータ（NEAC 3200／30）をオンライン・アダプタで結合したインライン処理方式による対話型および一括処理型情報検索システムである。

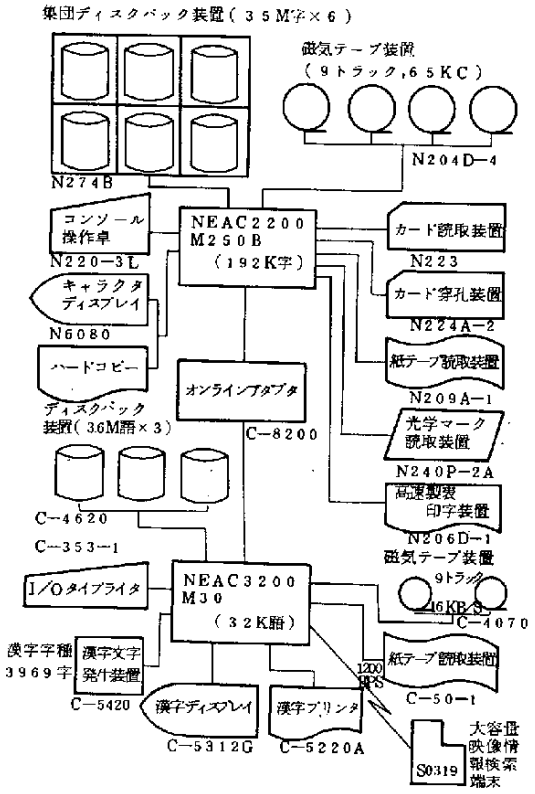
b）質問は、対話型検索を行なう場合はデータ・スコープを使用し、一括処理型検索を行なう場合はカード読取装置を使用して行なう。

c）検索結果は使用目的により、ラインプリンタ、データ・スコープ、磁気テープ、漢字ディスプレイ、漢字プリンタ、の５出力媒体に自由に選択して出力可能である。

d）被爆情報の蓄積・更新時、漢字ディスプレイ装置を使用して容易に漢字イメージ情報を作成し、ファイルの蓄積・更新を行なうことができる。

このシステムで使用している漢字種は、氏名や住所などを中心とした漢字情報であるため、NEAC 宛名漢字文字セットを採用し、これに６０字を追加した ３９６９字種が扱える。さらに当システムは拡張が行なわれて、診療カルテやオリジナルの各種調査票、文献情報などを大容量マイクロフィルム装置に収録した、オンライン・マイクロフィルム検索システムの開発が行なわれた。これが「大容量映像情報検索システム（ＩＲＡＢＳ－３）」である。

ＩＲＡＢＳ－３では、映像情報（静止画像：超マイクロフィッシュ）と漢字情報のハイブリッド（重畳）と、表示・重畳されたハードコピーが容易にとれる、映像情報の大容量化（最大３００万頁）が付かれる、という大きな特長が加わった。そのために用いられるのは大容量映像情報検索端末であり、フロント・エンド・プロセッサ（Ｎ３２００／３０）に接続されている。この端末装置には、映像表示画面、漢字表示画面、仮名・英数字・記号表示画面、という３つの画面が装着され、マイクロフィルム・マガジンが内蔵されている。



図－１８　ＩＲＡＢＳ－３

2.3. 関連技術の研究動向

2.3.1. 自然言語処理の研究

自然言語処理の概念は広く、計量言語学的な統計処理から日本語文の自動整形・編集、機械翻訳、カナ漢字・漢字カナ変換、自然言語理解、質問応答系など多岐にわたっている。その研究の流れは、自然言語の基礎研究から機械翻訳を原点にして進められ、人工知能、オートマトン、学習機械、パターン認識などとの関連で研究開発が行なわれている。

以下に機械翻訳の研究の流れから自然言語処理、日本語情報処理への経過を述べる。

機械翻訳は世界的には古くから研究が行なわれ、１９３３年にソ連のスミルノフ・トロヤンスキが、いくつかの人間言語を同時に翻訳して遠方まで送信する装置に特許を得ている。これはコンピュータ出現以前に機械装置のみを使った翻訳であった。その後最初のコンピュータＥＮＩＡＣが誕生した１９４６年にアンドリュー・ブース（Andrew D. Booth イギリス）はコンピュータの記憶部分に単語辞書を作って、Ａ言語の単語をＢ言語の単語に逐語訳的に置きかえる方法を提案した。同年にワレン・ウィーバー（Warren Weaver ，アメリカ）は言語の翻訳操作を暗号解読と同じような性格のものであると考えた。

そして１９４８年にリチェンズ（Ｒ．Ｈ．Ｒichens）は単語の文法変化語尾の自動解析（Ｓtem －ending method）を提案し、１９４９年にはオスワルド（Ｖ．Ａ．Ｏswald）とフレッチャー（Ｓ．Ｌ．Ｆletcher）はドイツ語の文章論のしくみを機械的に分析する方法を研究した。１９５２年、機械翻訳に関する最初の会議がＭＩＴで開かれた。

それ以降、理論を実際に適用する試みがなされ、１９５４年にドスタート（Ｌeon Ｄostert ジョージタウン大）とシェリダン（Ｐ．Ｓheridan，ＩＢＭ）はＩＢＭ７０１を使って２５０の単語と６個の文章規則によってロシア語から英語への機械翻訳に着手し一応の成果があった。

１９５６年から１９５８年にかけて機械翻訳に関係した研究計画がアメリカの大学やＩＢＭなどで本格的に開始された。

表２－１５ 機 械 翻 訳

研　究　機　関	研 究 内 容
ハーバード大学（エティンガー）	ロシア語→英語
ＭＩＴ（ロック　バー・ヒレル　ユングヴ）	ドイツ語→英語
ジョージタウン大学（ドスタート　ガーピン）	ロシア語→英語
ミシガン大学（クートスーダ）	ロシア語　単語の多義性
ワシントン州立大学（ライフラー　マイクルセン）	ロシア語 中国語 ドイツ 語

日本では１９５９年に電気試験所（現電子技術総合研究所）で「やまと」と名付けられた翻訳機械が試作されて簡単な英文和訳の実験が始められ、さらに九州大学でも「ＫＴ－Ｉ」という機械を作って日本語・英語・ドイツ語の３国語相互翻訳の試みが行なわれるに至った。同時期に京都大学

でも研究が開始された。

この時期に注目された理論は、機械翻訳を直接意図したものではないがＭＩＴのノーム・チョムスキー（Ｎoam　Chomsky）による新しい言語理論である。これは生成変形文法（Ｇenerative　transformational Grammar)と呼ばれ、人間言語のもつ抽象的な一面を形式化することに成功した。この考え方はわが国の言語学研究者にも、また情報処理の研究者にも大きな衝撃を与え、１９６０年頃から日本でもコンピュータによる言語研究熱が高まって、上記研究所・大学などで研究が開始されたものである。

その頃日本では計量国語学会が発足（１９５７年）して、その後も地道な研究が続けられている。アメリカでは１９６４年にＡＬＰＡＣレポートと称する調査報告書が出され、機械翻訳は人間の翻訳には決して及ばないとして研究活動が急速におとろえてしまった。その後言語に対する新しいアプローチは人工知能研究の分野から出されて、コンピュータと自然言語によって対話する研究へと進んでいった。

その後フランスやドイツなどでは着実な研究が行なわれてきており、最近はカナダにおいても英仏翻訳が研究されて天気予報という限られた範囲で実際にシステムが動いているといわれる。

日本では、国立国語研究所をはじめとする言語学系統の研究者と九州大学の研究者達は、機械翻訳よりもむしろ日本語の構文解析の研究に力点がおかれていった。これは必ずしもコンピュータを用いない日本語の構造的研究とともに次第に基礎研究の方向へと進んでいる。また京都大学や電子技術総合研究所などでは質問応答システムの研究へと進んで、人工知能の研究に接近していった。現在言語構造の研究と人工知能的な研究へと至っている。そして日本語の構文分析の研究は、日本語情報処理技術への適用として、カナ漢字変換やその逆の漢字カナ変換の技術へと利用されている。そこでは日本語構文から単語への自動分ち書きが可能になりつつある。

自然言語処理からの影響でカナ漢字変換（ローマ字漢字変換も含む）は各所で研究が行なわれ、主なものでも表２－１６に示すようなシステムが開発されている。カナ漢字変換の技術は、日本語情報（漢字かな混り文）のデータ作成法の１方式として利用されることは前述した。そこにはカタカナ（読み）から漢字（あるいは単語）への変換で、分ち書きと同音異義語の問題が大きい。カタカナ文が分ち書きされていることを前提にしたカナ漢字変換であれば、変換辞書の作り方次第でかなりの変換率が得られる。しかしカタカナ文の連続データから漢字かな混り文へ変換するには、まず基本的な単語・分節単位に自動分ち書きすることでさえ難問で、日本語文の構文解析や文法、意味処理までも行なわなければならない。ここには機械翻訳の技術が導入される。

表２－１６　カナ漢字変換システム

開発機関名	入　力	入力方式	処理方式	同音異義語(字)処理
ＮＨＫ総合技術研究所	カタカナ文 ローマ字文	自立語・分ち書き	最長一致法 単語辞書 付属語辞書	代表漢字表示 〔カッコ〕内に他を表示
沖　電　気	カタカナ文 ローマ字文	特殊記号・分ち書き	漢字辞書 文法辞書	代表漢字出力 同音異義語表出力(訂正用)
東　　レ	カタカナ姓名 カタカナ住所	分ち書き	姓名辞書 住所辞書	全漢字を(カッコ)内に出力して後処理
日本ユニバック	カタカナ姓名 カタカナ住所 カタカナ企業名	分ち書き	姓名辞書 住所辞書 企業名辞書	マークシートに出力して選択入力
学習研究社	カタカナ姓名 カタカナ住所	分ち書き	姓名辞書 住所辞書	代表漢字出力 誤変換を漢字コード入力して訂正
共同通信社	カタカナ文 ローマ字文	文節分ち書き	最長一致法 入力変換辞書 単語辞書 文法辞書	代表漢字出力 〔カッコ〕内に他を表示
電子技術総合研究所	ローマ字文	字／語分ち書き	固定漢字辞書 浮動単語辞書	会話型選択
国立国語研究所	カタカナ文	分ち書き	ヨミガナ辞書	代表漢字出力
九州芸術工科大学	カタカナ文	文節分ち書き	見出語辞書 係り受け辞書	代表漢字出力
日本情報処理開発協会	カタカナ文 カタカナ固有名詞	特殊記号・分ち書き	専門分野別辞書 固有名詞辞書	カタカナ入力時に選択番号を付加

2.3.2. 関連機器の応用研究

現在、漢字入出力装置として実用化されている入出力機器とその入出力方式については前述した。この項では日本語情報処理の技術としては直接的なつながりはなくとも、日本語情報の根本要素である漢字かな混り情報が入出力できる他の分野の技術についてその研究動向を述べることにする。

その中心となるのは画像処理の技術である。現在の情報処理技術の高度化が進む中で、情報伝達

と情報処理システムという観点に立ってみるとコンピュータ・電話・テレビという三者の技術結合の高度化から生まれる技術が大きな推進力になっている。コンピュータと電話から生れる技術がデータ通信であり、電話とテレビからはテレビ電話が、そしてこれを機能的に拡大したものとして画像通信がある。さらにコンピュータとテレビを結合することによって生まれる技術が画像処理技術である。

これらの技術のうち、データ通信はすでに実用化に入っている技術であり、画像通信ではファクシミリ、テレビ電話、ＣＡＴＶ、静止画伝送などの各種技術要素がそれぞれ一応の段階に達し、その一部の技術は、個別に実用化に入るとともに、さらにこれらが総合された技術へと高まりつつある状態である。

画像処理技術は、一部に実用化されたものがあるとはいえ全体的にまだ将来技術である。その現状は各研究機関に大容量の記憶装置を有するコンピュータと、各種の光学的・電子的手段を用いた画像入出力装置を組合せた画像処理装置が設置されて、各種の応用分野に対する画像処理の研究開発が進められている段階としてとらえることができる。

画像処理の方式は、光学方式、写真方式、ビデオ方式などのアナログ処理と、コンピュータ処理によるディジタル方式とがある。一般にディジタル処理はアナログ処理に比し融通性、精度、調整の容易さ、再現性などの点で優れているが、画像のもつ膨大な情報量をコンピュータに記憶するための記憶容量およびそれを直列に演算していくための時間が多くかかるという障害があった。しかしながら最近のＩＣ、ＬＳＩ技術の進歩によって、高速記憶と低価格化が実現され、ディジタル画像処理の最大の障害はとりのぞかれつつある。

また一方では、画像を処理する入出力装置が特殊で高価格であるという障害は、マイクロ・プロセッサの導入によって制御機能の大幅なソフトウェア化と、撮像、表示素子の進歩、ＩＣリフレッシュ・メモリの登場などによって明るい見通しがえられ、画像処理のディジタル化の傾向が強まっている。しかしアナログ処理の大面積、高速処理などの特徴を活かして組合せていくことも必要であり、入出力処理ではとくにその必要性は高い。

このような画像処理の技術は、すでに一部が日本語情報処理技術として導入され、入力装置には文字認識技術として、出力装置では文字パターン発生、表示・出力技術に、さらには伝送技術などに応用されており、今後ますますその傾向が強まりそうである。

もう少し身近に画像処理をとらえて、画像情報システム、画像処理システムあるいは画像通信システムといわれる分野をみることにする。

まず漢字かな文字を含めて、図形や映像などの情報を出力するための機器、装置としてテレビやテレビ電話、ファクシミリ、ＣＡＴＶ、そしてマイクロ・フィルム、スライド機器などがあげられる。

テレビ電話は本来、対面通話を主体に開発された装置であるが、これのもつ押しボタン・ダイヤル機能や簡易キーボードを付加することで、交換網を介してコンピュータにアクセスすることがで

き、その処理結果をテレビ画面上にディスプレイすることができる。

これと同様にして、文書・絵画の複写伝送を目的に開発されたファクシミリについても、適当な入力手段をもうければ必要に応じてコンピュータの遠隔端末となって処理結果を記録することが可能である。

また放送網の難視聴対策にはじまったＣＡＴＶ（Common （or Community） Antenna Television）も、最近では多種多様なシステムが開発されており、とくに双方向ＣＡＴＶの出現により、コンピュータによる情報サービスが不可欠の要素になってきた。これには家庭用のテレビやファクシミリが情報端末となって、通常の一斉放映のサービスのほかに、端末で必要とする情報をコンピュータから受けることが可能になる。

このような画像情報システムでは、端末・周辺装置を操作するのは、コンピュータの専門家よりもさらに不慣れな一般大衆であり、またサービス内容も簡易なものに限定される。したがって情報サービスとして表示あるいは印刷記録するにも、従来の英数字、カナ文字だけでは不足し、漢字・ひらがなどの文字や、棒グラフ・マス目などの簡単なグラフィック表示、さらにはマイクロ・フィルムに記録されている映像情報の表示なども必要になってくる。

これらの情報サービスの形態は、日本語情報処理システムのサービス形態においても、全く同じような機能が必要である。このことは逆にテレビやファクシミリを、日本語情報の処理結果を表示・出力する装置として使用することができ、コンピュータ内部で処理された符号情報から、視覚・図形情報への変換処理を施すことによってそれが可能になる。

(1) テレビへの漢字表示

日本語情報処理において漢字ディスプレイ装置の利用価値は高く、とくにオンライン処理を目的としたシステムには不可欠な装置である。しかしながら現状では、文字種の多さと文字パターンの複雑さのために文字発生装置が高価格化をきたし、英数カナ文字用のＣＲＴディスプレイ装置の普及のようにはいかない。

一方、画像情報の表示装置としては、グラフィック・ディスプレイとテレビが代表的である。グラフィック・ディスプレイは図形や文字の表示を可能とするため、漢字ディスプレイ装置としての機能はすべて備えている。テレビにいたってもその機能を有しているが、テレビ信号の送受信は一般にアナログ情報の放送電波である。

最近のテレビジョン技術の動向は、テレビ画像に字幕スーパーを流したり、テレビ信号の空時間に番組とは別の画像信号を挿入したりする情報サービスの技術が進められている。
このような研究は、静止画像システムとか、文字放送（文字情報放送）と呼ばれる分野で行なわれている。

文字放送とは、文字や簡単な図形で構成された画像情報をディジタル信号の形で放送電波に多重して伝送し、受信側ではこれを記憶装置に記憶したのちテレビ信号に変換して通常のテレビ受像機で受像する方式の仮称で、静止画放送の呼び方も一般的に使われている。静止画放送の利用形態としては、テレビ番組と関連する文字や図表などの情報や、時々刻々と発生する最

新のニュース情報、天気予報、交通情報などの比較的変化の少ない各種の情報が一定時間の繰返しでたえず送出されており、受信者には従来のテレビ受信機に付加したアダプタを使用して、任意の時刻に希望する情報を選択するとテレビ画面上に表示される。また耳の不自由な人を対象とする字幕放送への応用も考えられている。

文字や図形をコード化して伝送する方法（コード伝送）は受信側に文字発生装置を必要とするが、文字や図形からなるパターンを直接伝送する方法（パターン伝送）に比べ情報の伝送速度が数倍～数十倍になるから、文字発生装置を受信側に備えることが経済的に可能であれば静止画放送には最も適した方法といえる。英数字を使用する欧米諸国では、すべてコード伝送方式を採っている。

これに対し、我が国のように多字種を扱うために、テレビ受信側に文字発生装置を備えることは経済的にみて困難である。静止画放送の伝送方式においてもパターン伝送を使用せざるをえない。しかしこの場合でも、文字や図形を始めからドット構成でディジタル的に作成し、ドット・パターンを必要最小限度まで粗くすることにより、メモリ容量を通常の画像よりはるかに少なくすることができる。

日本語情報の表示端末としてテレビを使用する場合も、漢字ディスプレイ装置の文字発生方式にあるコードリフレッシュ型と、パターン・メモリ型のいずれかを採用しなければならない。一般のテレビの画素数は、およそ２４０×２１０＝５０,４００ドットであり、１５×１８ドットの漢字パターンで約９８文字を表示できるとすると、コード・リフレッシュ型で１９６バイト、パターン・メモリ型で６３００バイトのメモリをテレビ受信機に付けなければならない。そして文字発生装置の扱う文字種が２０００字としてもその記憶容量は膨大である。

テレビと接続する文字発生装置は、ディジタル型、アナログ型の文字発生方式のうち、機能や規模を考えると、テレビの走査方式にマッチしたドット方式が最も適することになる。また高価な文字発生装置を経済性から多重利用する場合にもドット方式が適している。

その他、テレビジョン信号処理の中心となる帯域圧縮ディジタル符号化処理に関する研究論文が数多く発表されている。

(2) ファクシミリ・プリンタ

ファクシミリ（facsimile）は文書・図形の伝送用に開発された装置であるが、通常のプリンタに比べ漢字や図形も記録できることから、日本語情報の安価な出力装置として広範な利用方法が期待されている。記録方式は、テレビや漢字ディスプレイ、漢字プリンタと同じように文字発生装置や図形発生装置を接続すれば、コンピュータで処理された符号パターン情報をファクシミリの走査方式に変換することで漢字プリンタの代用が可能となる。

ファクシミリの画像処理装置としての研究開発動向はおよそ以下のようである。

１枚の画像は多くの情報をパターンとして含んでいるが、これを電気信号に変換して伝送するために画素に分解すると10^6～10^7という多量のデータとなる。伝送には電話回線を使用することが主であり、３KHz帯域幅では伝送に多くの時間を要し、遠隔地への通信コストは高い

ものとなる。このため時間短縮に関する技術開発が進められ、ファクシミリ信号のデータ圧縮符号化によるディジタル伝送は飛躍的な高速化を可能にした。またデータ圧縮はファクシミリの蓄積交換システムや、画像情報サービスにおける情報ファイルとしての蓄積においても、記憶容量の低減のために重要な技術である。

ファクシミリでは通信の基本理念である信号を忠実に早く伝送することが前提になっており、そこで使われている技術は、画像を扱うとはいえパターン処理とは異なり、むしろ通信理論を基本とした信号処理技術の性格が強い。ファクシミリ信号のディジタル伝送技術は、アナログ画信号をディジタル化する処理と、符号化する処理が重要である。符号化はディジタル化された画像の画素パターンをより少ない符号で表示するための符号変換である。またデータ圧縮符号化は情報点の位置（アドレス）をできるだけ少ないビットで効率よく表すための符号変換であり、ファクシミリではランレングス符号化が代表的である。

ファクシミリのデータ圧縮伝送はこれまでに数多くの方式が提案されてきたが、符号化方式の大部分はランレングス符号化を基本とする種々の変形で、最近の実用機では複数ライン一括処理モード・ランレングス方式が多い。ランレングス符号化は白および黒画素の継続する長さ（Run　Length）を符号化する方式である。ファクシミリ信号のランレングスの平均値は通常数十ビット（白の場合１０～１００、黒の場合２～６程度）とかなり長いのでビットパターンを送るよりもランレングスを２進数で表示した符号を伝送する方が平均的に情報量が少なくなる。

２ライン一括処理モード・ランレングス符号化は、ファクシミリ信号の走査線（ライン）間の相関が強いことを利用してデータ圧縮を改善する方式であり、２ラインの上下の２画素の白（Ｗ）黒（Ｂ）の組合せで生じる４つのモード（ＷＷ、ＷＢ、ＢＷ、ＢＢ）のランの長さを符号化する。またライン間の相関をさらに効果的に利用するため前ラインの情報を逐次的に使用する方式も研究されている。この種の方式は伝送符号誤りの影響が２次元的に伝搬する欠点を有するが、前述の一括処理方式に比較して圧縮率は高くできる。

その他、多段分割符号化は一種のブロック符号化であり、その符号化のアルゴリズム、符号形式が単純なので、コンピュータによる信号処理に適するという特徴をもっている。以上のように符号化方式には種々多様な形式があるが、実用においては圧縮率ばかりでなく、符号誤りの影響度、回路構成の複雑さ、経済性、ソフトウェア処理では論理処理の単純さなど様々な面からの総合的判断が必要である。

一方、ファクシミリ通信のネットワーク化も進んでいる。従来は加入電話回線や専用回線を利用して、Ａ地点からＢ地点へと１対１のメッセージ交換が主であった。しかしファクシミリの使用台数が増加し利用率が高まるにつれて、ユーザ側からファクシミリの使用効率、回線の利用効率の向上、新サービスの要求となって、メッセージ・スイッチング・ネットワークが開発・提供されるようになってきた。このネットワークは電話の交換機にあたる部分にコンピュータによるファクシミリ交換装置を置いて、ファクシミリおよび回線を接続して相互に通信するものである。そこにはメッセージを一度記憶装置に蓄積し、コンピュータの指示によって、要求されたサービ

スに従い、同報サービス（１から複数への送信）、代行サービス（使用中・故障中の代行あるいは後刻送信）や優先サービス（サービスの優先順で送信）などを行なって、ファクシミリ端末および回線の利用効率を改善する。ネットワークはその構成により、加入電話回線を使って同報サービスなどのメッセージを即時で転送・交換する「回線交換方式」と、メッセージを一度交換用コンピュータに蓄積し、この蓄積されたメッセージを新たに構成されたネットワークで相手方に送る「蓄積交換方式」に大別される。

以上、ファクシミリ技術の動向を概説してきたが、ここでファクシミリをコンピュータの入出力装置としたときの観点にたって特徴をみることにする。

一般に情報処理用の入出力装置は、あらかじめ定められた符号を発生したり、定められた符号を受信してそれをプリンタなどに出力するという形式である。一方、ファクシミリは、白と黒が平面的に組合せてできるパターンならば原理的にはどのようなものでも扱うことができる。これは従来の入出力機器が扱ってきた英数字やカタカナ以外の多くのパターンを扱えるという拡張性が大いに期待できる。

ファクシミリの受信機については、コンピュータで送信機と同じような信号を作成して送り出してやれば、受信機には漢字や図形が記録でき、漢字プリンタや図形プロッタとして使用することができる。

また送信機については、画像情報の入力に適用でき、マークシート・リーダや文字読取り装置として、また汎用の線図形入力装置として応用できる。

ファクシミリ受信機の応用は、とくにファクシミリ・プリンタとして期待できる。コンピュータ処理は、文字コードや図形コードなどの符号化された情報を文字パターンや図形パターンに変換して、それをファクシミリ信号として送信すれば、ドット・プリンタ、グラフィック・プリンタとして有効である。

電電公社武蔵野電気通信研究所で研究開発しているファクシミリ応答装置は、ファクシミリ応答制御装置と呼ばれる共通制御部と、複数個のファクシミリ応答端末装置とよばれる端末対応部からなっている。後者のファクシミリ応答端末装置は、中央処理装置から転送された文字情報１行分の文字コード列および１走査線分のドット・パターン列を記憶しておくメモリを有し、そのドット・パターンをファクシミリ信号に変換する機能をもっている。

このようにファクシミリ受信機をプリンタとして使用しているケースは数少ないが、それは解像度とスピードに難点があるからといわれている。しかしながらファクシミリの需要は高く、各メーカでもますます高解像度と高速処理をめざしてファクシミリの開発が行なわれており、他の情報処理用出力機器に比べて低価格に魅力もあって、今後日本語情報処理用の簡易な遠隔漢字プリンタとしても大いに期待されている。

また前述したファクシミリの各種技術も他の情報処理技術に利用されているものもあり、日本語情報処理分野でもデータ圧縮技術として一部利用されている。

(3) マイクロフィルムの応用

コンピュータによる情報の管理は、大量のデータを処理できるという特徴がある。しかしその情報はディジタル化できるものに限られて、高速処理と正確性をもって利用されてきた。それは人間が通常使っている文字や図形をコンピュータで扱うというよりは、コンピュータで扱いやすいディジタル情報のみに範囲がしぼられ、人間がコンピュータに歩みよって使っているにすぎない。これに対して、マイクロフィルムはディジタル化できない情報、いわゆる文献・資料・図面・写真などの非ビット情報の処理に適している。

こうしたディジタル処理のコンピュータとアナログ処理のマイクロフィルムをつないでいかに効率化をはかるかが利用面での課題である。コンピュータとマイクロフィルムの結合は、1つはCOM(Computer Output Microfilm)システムであり、もう1つはマイクロフィルム検索システムである。

マイクロフィルムの特徴は、文献や図形の縮小管理と、原本どおりに記録できる正確性とにあり、フィルムの形態としてはロール状のものとカード状のものに大別される。

① ロールフィルム……100フィートのフィルムをリールに巻いたもので、普通2000コマの写真を入れることができる。

② マイクロフィッシュ……4×6インチのサイズのシート状のフィルムに、60、72、98、あるいは224コマを収容したものでダイレクト検索ができる。

③ アパーチュア・カード……ロールフィルムを1コマあるいは数コマにカットし、紙カードに窓をあけたところにはりつけて使用する。

④ PCMI(Photochromic Microimage)……超マイクロフィッシュ、スーパ・マイクロフィルムともいわれ、4×6インチのフィッシュに3000コマ以上を収容できる。

これらのマイクロフィルムの作成と利用手順は一般に、

撮影 → 現像 → 保管 → 検索 → 映写 → 映読 → 複写

という工程を経る。マイクロフィルムの検索は、カード状のものであれば、表題とか番号が人間の目で判別できるように工夫されており、検索は目録カードと同様な方法でランダム・アクセスができる。これには手動検索方式と自動検索方式とがある。

ロールフィルムの検索方法には次のような種類がある。

① ミラコード方式

ミラコードとはMicrofilm Information Retrieval Access Codeの略で、その機能はデータをマイクロフィルムに撮影すると同時に、その内容を記述する検索用コード(キーワード、分類コードなど)も撮影する。検索コードには透明、不透明の長方形のコードを使用し、バイナリ・コードで表現して、専用のリーダのロジック回路によって、求めるフィルムをアクセスする。

② イメージ・コントロール方式

別名ブリップ方式ともいわれ、フィルム1コマごとにブリップと呼ばれる矩形のマークをフィ

ルム上に撮影し、専用のリーダで求めるフィルムが何コマ目かを指示することにより、リーダの光電管装置でブリップ・マークを計算してアクセスする。

③ コードライン方式

フィルムの各コマとコマの間にコード・ラインを撮影し、リーダのスクリーンの端に取付けられたスケールにその線が合致するかどうかでアクセスする。

④ フラッシュ・ターゲット方式

マイクロフィルムの内容は肉眼で判別はできないが、フィルムの内容を見やすくするためにフィルムの始めや途中に肉眼でも判別できる各種のコマ（ターゲット）を挿入する。これを検索するとき、リーダでフィルムを送るとフラッシュ・ターゲットが通過することによりフィルムの区分を知る方法である。

⑤ カウンタ方式

フィルムの長さを目やすとして検索する方法で、リーダにフィルムの長さと対比したコマ番号を指定する。

このように各種の形式のフィルムとそれに合った検索方法が考案されて、マイクロフィルムの検索が行なわれている。しかしながら大量の情報を管理して、その中から必要な情報を探し出すために、マイクロリーダを使うことは容易でない。ここにコンピュータの高速処理機能を導入して、マイクロフィルム検索システムを構築する必要性がでてくる。実際にミニコンピュータを組込んだマイクロリーダも数社から開発されており、アナログ処理とディジタル処理を効果的に利用した例といえる。この検索方式では一般の情報検索システムと同様に、検索情報や条件をコンピュータと対話しながらキーボードから入力し、検索条件に適合した情報をマイクロフィルムからダイレクトにアクセスしてサービスする。またマイクロフィルムのソースをコンピュータから直接アクセスしないで、マイクロフィルムのキー番号を提供して、これを利用者がマイクロリーダに入力してアクセスする方法もとられる。

また変った利用方法としては、マイクロフィルムに記録されたアナログ情報をリーダに表示してサービスするばかりでなく、コンピュータ内部で処理した数値情報を画面に二重表示するシステムも研究開発されている。これは、出力に要求されるケイ線やワク組、図形情報などをマイクロフィルム化してもち、内部処理で得られる数値情報をそのフォーマット・パターン内に挿入して表示する方式である。

以上のマイクロフィルムの応用は、フィルム形態や検索方法は異なっても、フィルムに蓄積する過程はカメラを使ってのオフライン処理である。これに対してＣＯＭは、コンピュータで処理された情報をハードコピーとしてプリンタなどに出力せず、マイクロフィルムに直接あるいは磁気テープを介して記録させることができる。

ＣＯＭは、コンピュータからディジタル信号をアナログ信号に変換し、それをさらに光に変換してマイクロフィルム上に記録する。電気信号から光への変換にはＣＲＴ（Ｃａｔｈｏｄｅ　Ｒａｙ Ｔｕｂｅ）が多く使用されている。マイクロフィルム上の記録は、目にみえない潜像であるため、これに

現像装置が必要であり、CMCで中に現像装置を内蔵したものもあるが、一般には別になっている。

ＣＯＭの機能を大別するとＢｕｓｉｎｅｓｓ　ＣＯＭ（事務用ＣＯＭ）とＧｒａｐｈｉｃ ＣＯＭ（科学用ＣＯＭ）の２種類になる。前者はラインプリンタのモードで使え、特殊なものとして漢字ＣＯＭが含まれる。後者は文字以外に精密な図を描くことができ、ＸＹプロッタと同様に図形処理用として使用される。

漢字ＣＯＭは、現在数社のメーカで開発されているが、国産メーカよりも外国のＧｒａｐｈｉｃ ＣＯＭ開発メーカのものが多い。実際に使用しているユーザもわずかである。それは漢字出力装置の文字発生器が高価であるのと同じで、ＣＯＭのＣＲＴ上に多字種の漢字をもたなければならないからである。ましてＣＯＭ装置自体はさらに高価であり特殊な分野でしか利用されない。

(4) その他

前述のマイクロフィルムと同じような機能をもったものにスライドフィルムとＶＴＲがある。映像情報の検索システムとして、スライドプロジェクタによる検索システムとか、ＶＴＲ映像検索システム、さらにはビデオファイルの検索システムなどの研究開発がある。これらはいずれもアナログ的に映像情報として扱うと、情報の量が少なく、蓄積量も増加し、検索速度もはやくなるというメリットを生かしたシステムである。この映像の中には漢字かな混り文や図形情報が含まれており、多くの分野で適用可能とされている。たとえば医学における各種の写真やカルテ、グラフなどの蓄積・検索に、また教育方面ではＣＡＩ（Computer Assisted Instruction）とともに視聴覚教育において有効な利用方法と目されている。

この他、日本語情報処理に関連する技術として、パターン認識技術がもっとも大きな影響を与えることはまちがいない。音声認識・文字認識などはいちはやく研究開発から実用化へ進んでほしい技術である。

3. 日本語情報処理技術の考察

3.1 日本語全般の問題点

日本語情報をコンピュータで処理する場合に、最も大きな問題は日本語文を記述・構成する文字種が非常に多いことである。日本語文を表記する文字は、中国から伝わる漢字と、日本で生れ育った和字・仮名文字（ひらがな、カタカナ）と、英字をはじめとした欧米文字、それに数字と、多種多様である。とりわけその中でも漢字（和字も含む）の種類が最も多く、制定漢字の当用漢字で1,850字、通常の新聞・雑誌にはおよそ3,000字種が使用されている。また姓名・地名などの固有名詞を書き現わすには約1万字種が必要とされており、一般の漢和字典でも1万字前後の文字が収録され、最も多くの漢字を収録しているといわれる康熙字典では5万字近くもあるという。文字種の少ない数字でさえも、アラビア数字、ローマ数字、漢数字などが日本語文中に出現する。

このような多くの文字種を扱わなければならない日本語情報処理は、欧米諸国の言語情報処理に比較して大きなハンディキャップをもっている。コンピュータ処理のために日本語情報を入力するには、英数字（カナ文字）用の入力キーボートに比べて、キー数が増大し、その中から該当文字を選択（探索）して打鍵する操作は大変であり、その結果入力速度も遅く、パンチャ（オペレータ）の疲労度も高くなる。またコンピュータ処理した日本語情報を出力するには、漢字・仮名・英数字を印字できる出力装置が必要であるが、これらの文字を出力するための文字発生装置の規模が巨大化する。英数字用の文字発生は文字種が少なくタイプライタ型の活字印字も容易だが、数千字の漢字を文字発生するには同一機構ではとても実用は無理である。

さらに文字種の影響は、文字コードに及んでくる。英数字の表現は、64種を6ビットで可能であり、8ビット（1バイト）では256種の表現まで可能となってカタカナまでも扱える。これに対して、数千字の漢字をコード化しようとすれば、4,096字種を12ビットで、14ビットでは16,384字種まで表現できる。コンピュータの処理単位としてバイトの扱いがもっとも一般的であることから16ビットの漢字コードが多く、最大65,536字種まで可能である。しかしながら実際には、上記の文字種から、5,000～10,000 字種を上限とすれば14ビットで十分である。制定されたJIS コード（情報交換用漢字符号系JIS C 6226—1978）でも、第1水準漢字集合2,965字＋第2水準漢字集合3,384字＝6,349 字種を14ビット体系にしている。これを16ビット処理系にして使用するには上位2ビットを0にして処理するようになっている。これまでJIS コードのような標準コードが存在しなかったために、各メーカ、各ユーザの利用目的によって自由なコード体系を作って使用していた。そのため現在では、まだJIS コードを使用せずに、従来からのコードを使っているユーザが大部分である。今後、新機種の入出力機器や新しい情報処理システムに除々にJIS コードが採用されるであろう。

次に文字種・文字コードの設定にあたって文字の配列順が問題になる。英字はABC順、仮名は五十音順に並べることができるが、漢字には、このような誰もが手軽に並べられる方法がない。このことは、コンピュータ処理におけるソート／マージに規則性を設けることができず、出力結果の利用方

法にも問題が残ってしまう。一般には、漢和字典の部首順や代表音訓読による索引、総画数による索引などが用いられている。部首順は一般に康熙字典方式を準用しているが、これが必らずしも検索に便利とはいえず、この修正版も出るようになった。それでもなお部首順による配列は引きやすいという感じを与えていない。また代表音による五十音順という配列も多く利用されている。この場合、音読のないものあるいは使用されないものはどうするか、多音読のある漢字はどうするか、という問題がある。しかも同音の漢字が多数あることも問題である。訓読にしても同様である。このように配列にいくつかの方式があることは、コンピュータ処理に際して不便を生じる。この解決をはかるべくＪＩＳ コードが制定されたがそこでは使用率の高い第１水準漢字集合を音読順で、使用率の低い第２水準漢字集合を部首順で配列している。この配列法では、第１水準と第２水準の漢字を混合して使用したときに分類は完全に二分されることになる。

こうした配列は、同時に入力キーボードの漢字配列に関連し、フルキーボード方式の入力で、パンチャ／オペレータの操作性に大きな影響を与える。そのためＪＩＳ コードでも字種とコードは制定しているが、けん盤の文字配列までは規定していない。よってメーカおよびユーザとも、適用業務・利用者によって文字配列を設定することになる。

一方、出力に関する日本語文（文字）の特質として、字体（新字、旧字、古字、俗字）と書体（明朝体、教科書体、ゴシック体など）がある。字体は、当用漢字字体表が制定されてはいるが、その他の字種については規準がなく、必要に応じていくつかの字体が用いられている。コンピュータ処理においては、姓名や地名などの固有名詞の扱いでは旧字体などを使用せざるを得ないが、その他の文書ではほぼ当用漢字字体を中心に使用されている。

書体は、漢字プリンタの出力文字に対するデザイン上の字形として、モニタ・プリント用の簡易型と、一般印刷に用いられている電算植字用の字形とがある。簡易型は多くがドット字形であり、細ゴシック体や明朝体のドット文字がデザインされている。電算植字用の書体としては、明朝体、ゴシック体、教科書体、新聞書体が、一般印刷とおなじく用いられている。漢字プリンタは、簡易型を除いて多くは明朝体を主体にして使用されており、その特徴としては、横線が細く、縦線が太い。線の末端には毛筆書きのトメ、ハネ、オサエの名残りがある。いずれも線の太さ、末端のアクセントが可読性を高める要素となっている。また次に多く用いられるゴシック体の特徴は、横線と縦線がほぼ同じ太さで線が太い。線の末端に付加物がない。とくにハネを少し細め、線端の切り方によって文字の表情を出している。

書体の使用にあたっては、本文用と見出し用、補助用と使い分けられ、本文用は長い文章を読むための書体であり、要求される条件がきびしい。見出し用書体は、さらに見出し用と帳票用とに分かれ、見出し用は線の太さ、長さ、フトコロの広がりに注意をはらい、一見して読めるものでなければならない。帳票用は正方形に近いバランスで、天地のラインをそろえた規則性のよい書体が必要である。

補助用は本文用と同じ書体を用いてふりがなどに使われる。これら何種かの書体を必要とするのは適用業務が限られ、コンピュータ処理の漢字プリンタでは、明朝体を目指したものがほとんどである。

文字サイズにおいても、多種多様な活字が存在するが、一般の漢字プリンタでは多種機能を有するものは少なく、電算植字用のプリンタなどに限られて、大小様々な文字が使われている。

表３－１　文字種のレベル

①	教育漢字	８８１字
②	当用漢字	1,８５０字
③	ＪＩＳ 第１水準漢字集合	2,９６５字
④	常用漢字（昭和４３年全日本漢字配列協議会）	4,００３字
⑤	ＪＩＳ 第１水準＋第２水準漢字集合	6,３４９字
⑥	角川新字源	9,９２１字
②′	基本的漢字	2,０００字
③′	分野別漢字	3,０００字
⑥′	全分野・固有名詞	１０,０００字

表３－２　情報交換用漢字符号系（ＪＩＳ Ｃ６２２６－１９７８）の文字種

特殊文字	１０８	間隔	1
		記述記号	３６
		括孤記号	２２
		学術記号	１５
		単位記号	9
		一般記号	２５
数字	１０		
ローマ字	５２		
平仮名	８３	新、旧仮名	４８
		濁、半濁音	２５
		拗促音	１０
片仮名	８６	新、旧仮名	４８
		濁、半濁音	２６
		拗促音	１２
ギリシャ文字	４８		
ロシア文字	６６		
漢字	6,３４９	第１水準	2,９６５
		第２水準	3,３８４

3.2 入力の考察

漢字入力装置の現状については前述したが、ここでは日本語情報、とくに漢字の入力に関する種々の問題点をとりあげて考察することにする。日本語情報の入力方法を述べるには、入力装置の操作性と、適用業務における利用方法、さらには入力を行なう利用者（オペレータ、パンチャ）のレベル、といった観点から考察する必要があろう。

(1) 入力装置の操作性

入力装置の操作性に関連する要因として、

・文字種・キー数

・文字の配置

・打鍵・操作方法

・モニタ表示・印字

・修正方法

などをあげることができる。これらの要因はそれぞれに相関して影響するが、前述した日本語全般の問題点から直接にも間接にも影響を受けている。

① 文字種・キー数

漢字キーボードからの入力では、入力装置が扱える（入力できる）文字種の数によって、入力の操作性は大きく異なってくる。文字種が多くなればなるほどに鍵盤に配置するキー数が増えるため、オペレータ（パンチャ）が文字位置を覚えることが負担である。一般に英文タイピストの養成にくらべて、和文タイピストの養成には数倍の訓練が必要である。しかも熟練者の入力速度は、英文タイプ２５０字／分（Ａ級）、和文タイプ５０字／分（１級）と５分の１であり、フルキー方式の漢字入力装置すべてが、この程度である。また漢字テレタイプ方式ではキーの種類に、文字キーとセレクト・キー（シフト・キー）があって、これを両手で操作して入力するため操作性は落る。しかし両手を使ったバランスのよさはオペレータの疲労度を少なくしている。

② 文字の配置

漢字入力鍵盤への文字の配置は、ＪＩＳ 化されておらず、メーカおよびユーザの利用目的にまかされている。英文タイプライタでは汎用性が高く、また仮にＡＢＣ順に並び変えたとしても文字種が少ないので影響はそれほどない。しかし多字種を扱う漢字入力では、その配列方法によって影響が大きい。一般に音訓読順、部首別順、画数順、頻度順などで配置することができるが、これらを組合せた方法も多い。たとえば、使用頻度の高い漢字群（当用漢字など）を音訓順に、低頻度の漢字群は部首別順に配分する方法であり、高頻度漢字群を操作しやすい手前側に、低頻度漢字群は左右・前方の離れた場所に配置するなど工夫する必要がある。一方、英数字・カナ文字の配置は、素人にとってＡＢＣ順、アイウエオ順の方が探しやすい。和文タイプライタなども英数字・カナ文字部はそうなっている。

③ 打鍵・操作方法

英文タイプライタのように両手を使って打鍵するか、和文タイプライタのように片手で打鍵するかはオペレータの疲労度に関連する。漢字テレタイプ方式の場合には、右手で文字キーを、左手でセレクト・キーをというように両手を専用化して操作する方法である。タブレット方式（テーブル・ルックアップ方式）では、右手にライトペンなどを持って文字を指示するペンタッチ型と、指でキーを押下するフィンガー・タッチ型とがある。その他の方式は、音声入力、OCR 入力方式を除いて、英文タイプライタやカナ文字タイプライタなどと同じシフト型である。入力のスピードの点では両手の方が向上し、両手のバランスの点でも疲労度の面でもすぐれている。これは専門パンチャにとって重要なことであるが、素人のオペレータにとっては片手で指示・入力する方が簡単で操作性にすぐれている。

④ モニタ表示・印字

入力の正確性の面から、オペレータが打鍵・入力した文字を確認できる機能が必要である。専門パンチャの場合には、入力文字の確認を入力中にすることはほとんどないようだが、これは後の作業で検孔や修正工程がある時に限られる。漢字入力装置には、この検孔機能がほとんどなく、モニタ印字を行なって校正するという工程が一般的である。和文タイプ方式の漢字入力装置にのみ、この印字機構が装備されているが、他はデータ作成とモニタ印字は分離しており、漢字プリンタを介してモニタをとっている。もっとも多く実働している漢字テレタイプ方式では、専門パンチャが使用することもあって検孔機能はなく、モニタ印字・モニタ表示装置を使用しなければならない。入力装置にモニタ表示・印字装置をつけるには文字発生装置が必要となって高価格化をきたし、データ作成業務用にはほとんど装着することはない。とくにモニタ表示用漢字ディスプレイは修正・更新用に使用されている。モニタ表示装置と操作性との関連は、修正のしやすい画面制御機能と、編集機能の充実をはかることが要求される。

⑤ 修正方法

漢字入力装置の多くは、これまで紙テープを使ってデータ作成を行なってきた。しかし紙テープの取扱いは効率が悪く、磁気テープにメディア変換したうえで修正処理が行なわれている。最近の、データ・エントリ機器は、紙テープからカセット・テープやフロッピー・ディスクなどに変ってきており、修正はしやすくなっている。漢字テレタイプ方式で入力されたデータは、モニタ印字がとられて、これに人手で赤字校正をして訂正データを作成し、修正処理を繰返し行なって正しいデータを完成する。この校正作業は漢字入力の特異なもので（印刷の校正と同様）、パンチャとともに校正作業の人員確保もおろそかにできない。これに対して漢字ディスプレイを使った修正作業は、修正のための編集操作が簡単で、修正箇所がすぐに確認でき、正確性が高い。しかし漢字ディスプレイのコストと、コンピュータ利用のコストが余分にかかって、データ作成・修正が高価格になることが多い。そのためデータ作成業務は自社処理よりもパンチ・センターへ外注することが多くなっている。即時性のあるデータや更新処理にのみ漢字ディスプレイが使われている。

以上の操作性に関する考察は、主に漢字キーボードからの入力方法に対してである。これに対して、漢字の入力をできるだけ無くそうという観点から、英数字キーボードやカナ・キーボードを使って、漢字に代替するコードや読み仮名による入力方法がある。入力の操作性は、入力速度、訓練の容易性、正確性などによって議論されるが、このうち前二者は一般的に相反することが多い。たとえば入力速度がもっとも速いといわれる連想コード方式では、漢字に対応するコードを記憶するのに時間がかかる。またディスプレイなどを使って会話型で入力するマルチストローク方式は操作は簡単であるが、応答に時間がかかって入力速度は遅くなる。カナ漢字変換方式でも同様な特徴を有する。

一方、人手によるキーボード操作を全くなくすのが、パターン認識によるOCR 入力方式（印刷漢字、手書漢字）と音声入力方式であるが、現状では実用化はむづかしい。

(2) 適用業務からみた入力

日本語情報処理の適用分野は、今後ますます拡大されるであろうが、適用業務によっては専任オペレータを雇用して大量データの入力を行なう場合もあれば、外注して一括入力する場合、あるいは少量データを非専門者が入力する場合など種々の運用形態がとられている。また即時性が要求されるものと、ある一定期間まとめて入力処理する場合もあり、入力装置に要求される機能も異なることが多い。

① 印刷・編集分野

この分野は、本来日本語情報処理が主要業務であるため、大量の入力データがあり、一連の工程は、熟練した専門のオペレータによって行なわれている。入力装置としては、入力速度が速いことが第1の条件であるが、そのためには漢字の配置あるいは対応するコード付けに十分な配慮が必要である。しかし専門のオペレータがスピードをあげて入力するため、同時モニタやガイダンス機能などは不要である。専門オペレータはほとんどモニタを見ずに打鍵してしまい、かえってモニタを見ながら入力作業を行なうことは、複雑化してオペレータの負担を大きくする。

この分野で扱う文字種は多く、基本文字を5,000字以上にして、他に外字入力が要求される。記号や約物、編集用ファンクションの入力も行なわなければならない。このため入力方式としては、フルキー方式を使用し、その中でも漢字テレタイプ方式がもっとも多く適用される。和文タイプライタ方式では入力速度がいく分落ち、テーブル・ルック・アップ方式では文字盤の間隔が狭いため、1字1字文字盤を見なければ入力できない。そのため入力速度が遅くなってしまう。また記憶方法にユニークさをもった連想コード方式では、覚えてしまえばスピード・アップがなって理想的であるが、コード設定のために文字種に制限を受けることが多く、多字種の扱いには欠点を有する。漢字テレタイプ方式では、両手を使って入力するので長時間の作業に向いており、専門オペレータによってかなりの入力速度が期待できる。

一方、この分野では印刷・編集業務ばかりでなく、新聞社・通信社などでのメッセージ集配信業務が含まれる。メッセージの集信（原稿収集）は、国内の場合は漢字テレタイプを使った

入力がほとんどであるが、国外からの集信の場合には、テレックスのキーボードが英数字のため、英文やローマ字文で入力されることが多い。そのためカタカナやローマ字（英字）で受信して、漢字かなまじり文に変換する、いわゆるカナ漢字変換方式を採用することも有効である。この場合、受信文が即新聞原稿となるわけでないことから、あえて高度な変換、１００％の変換率は必要なく、編集者にとって内容がわかりやすくなればよいであろう。また、メッセージ集配信にファクシミリを利用する方法もある。

② 事務処理分野

この分野における日本語情報のコンピュータ処理は、すでにカタカナやローマ字によってＥＤＰ化されているものが多い。そのため漢字化したいものが、カナ・ファイルなどですでに存在している。これを漢字化する方法としては、新たに漢字データを作成する方法と、カナ漢字変換処理によって自動変換する方法とがある。前者の方法は大量データの作成であることから、専門のオペレータによって漢字入力する必要がある。しかし、この分野で大量データの作成が必要になるのは、マスタ・ファイルを完成するまでであり、運用時には少量の修正、追加データを入力するだけとなる。このような場合には、専任オペレータを自社で管理することをしないで、一般には外注によってデータ作成を行なうことが多い。

事務処理で取扱う漢字データの内容は、主に姓名・住所・会社名あるいは商品名といった固有名詞であり、社内文書の漢字化はコンピュータ処理するほどに需要は少ない。固有名詞用の漢字入力は、一般文書よりも漢字の種類が多いために、フルキーボード方式の漢字けん盤に、外字入力ができる機能が必要になる。漢字けん盤の規模は、無制限に漢字を増加することはできても、オペレータの操作性からそれほど大型化することは好ましくない。うで、身体の動きが大きくなって疲労度が増してしまうからである。そのため、一般には使用頻度からけん盤文字の選定を行なって最適化をはかり、低頻度の漢字は外字入力による方法がとられる。

一方、カナ漢字変換処理によって、カナ・ファイルを漢字化する方法は、新規データの作成コストがぼう大になることから対処されて、効果をあげることができる。この場合、扱うデータの種類・内容・精度などによって効果に差が出る。たとえば住所や姓名のカタカナ表現において、ある短い単位で分ち書きされているような場合には、変換辞書の作成も容易で、その変換アルゴリズムも簡単なマッチング程度で処理することが可能である。カナ漢字変換の最大のネックは、同音異義語の処理が多いために一義的に漢字を対応づけることがむずかしい。そのため固有名詞処理では、変換結果をモニタ・リストして、人手で台帳と照合を行ない、訂正あるいは選択データをおこしてマスタ・ファイルを完成させるという手順がとられる。また少量データの場合には、漢字ディスプレイを使用した会話型で同音異義語処理を行なうことも有効である。

事務処理分野で漢字入力を自社で行なうとすれば、運用時における修正、追加データ用であることから、キーボード・ディスプレイ型の入力装置が最適であろう。一般事務員が手軽に扱えて、操作性のよい装置が望まれる。

③ 情報検索分野

この分野では、技術文献情報などの文書管理や、顧客情報、住民記録情報などの管理を行なって、利用者からの問合せに対して提供サービスする業務を主としている。これらの業務は、これまで漢字かなまじりの日本語情報をコンピュータで扱えなかったことからＥＤＰ化されずにいたものと、すでに英数字やカナ文字によって処理されていたものとがある。この分野の大きな特徴は、提供サービスするためのデータ管理がぼう大なことである。そのためデータ作成も大量入力が必要である。

技術文献情報では、専門分野ごとの文献にまとめられるため、専門用語は多いが、漢字の種類はほぼ平均的に3,０００字くらいが使用されている。また文献情報の発生は定期刊行物が多いため、一定期間ごとにまとめてデータ作成およびデータ・ファイルの更新が行なわれる。文献情報の提供は、問合せに適合する文献を早く正確に行なうことが要求される。そこには漢字記述の原文そのままを提供する必要はなく、一部が仮名文字に置き換えられてもそれほど影響ない。そのため入力の問題は、基本漢字として約3,０００字を入力できる装置を使用し、データ発生量と即時性とから、自社入力するか外注するか判断すればよいであろう。また英数字やカナ文字で処理してきたデータを漢字化するには、カナ漢字変換処理も有効に使用すべきである。なお文献情報には英数字項目が多くあるため、漢字項目と分離したデータ作成が可能である。

一方、顧客・住民情報といった固有名詞を扱った情報検索は、これまであまりＥＤＰ化されなかった分野であるが、漢字の扱いが可能になって、これらの情報提供サービスができるようになった。顧客・住民情報の特徴は、一度マスタを漢字化してコンピュータに記憶してしまえば、後の運用では少量の更新データの作成のみになる。そのため外注による場合が多く、そこでは固有名詞を扱うために文字種が多く、フルキーボード方式に外字入力機能が付いたものが適用されよう。

技術文献・固有名詞情報を問わず、情報検索の利用面からは、一般にオンライン会話型で問合せ入力するが、そのための入力は少量のキーワードやパラメータに限られる。入力装置あるいは入力キーボードの操作性は、入力速度よりも利用者にとって親しみやすい操作の簡単なものが好まれる。会話型であることからカナ漢字変換方式やマルチ・ストローク方式の入力方法も有効で、タブレット方式なども適用されよう。

(3) 利用者からみた入力

一般に入力装置に望まれる条件は、①操作性に優れ、②入力速度がはやく、③正確に入力できる、ことがあげられる。入力操作性は、誰もが手軽に操作することができて、操作のための訓練を必要としない、あるいは簡単な訓練で操作できること、操作のための疲労が少ないこと、そして熟練者にとってはメクラ打ちが可能なことが要求される。入力速度は操作性がよければ速く入力できることになるが、そこには正確性も要求される。正確さは入力文字の確認・検孔と、修正のしやすさなどが要件となっている。これらの条件が全て満されることが理想的であるが、日本語情報処理においては漢字の字種が多いために、実現がむずかしい。そのため入力装置を取り扱

う人間（利用者）のレベルによって、どういう入力方式の装置が適するのかを判別しなければならない。

漢字入力を専門の業とするような人にとっては、操作の訓練に時間が多少余計にかかっても、入力速度の速いことが第1の条件になる。そのためには操作性が高く、文字種・文字配列が汎用的な方式がよい。操作性に優れた入力方式としては、ワンダ・コンピュータ社やラインプット社などで実用化されている連想コード入力方式や、マルチ・ストローク方式、カナ漢字変換方式などがあげられよう。これらの中で入力速度のもっとも速いのが連想コード入力方式である。しかしこの方式は、誰もが入力できるというものでなく、長期間（他の方式にくらべて）の訓練を必要とする。それは各漢字に対応した連想コードを字種分覚えなければならないからである。さらに文字種が制限されることも欠点である。正確性の面では、コードのモニタをとっても専門家でなければチェックすることができず、コードから漢字へ変換を施した後に漢字モニタをとって、ようやく校正・訂正が可能になる。マルチ・ストローク方式では入力速度が遅く、カナ漢字変換方式でも同音異義語（異字）の処理が必要になって速度が遅くなる。

このようなことから、コード入力方式よりも入力速度は幾分落るが多字種が容易に入力できるフルキーボード方式の漢字テレタイプ方式がもっとも多く使用されている。また一種の専門オペレータである和文タイピストを使って、和文タイプライタ型の入力装置も事務処理などで利用されている。和文タイピストを漢字入力のオペレータとして使用できる場合には、もっとも効果的な入力装置である。しかも汎用性がもっとも高い。

これに対して、漢字入力を素人が行なう場合には、入力速度よりも操作性や正確性の面を重視して、入力装置の選択を行なわなければならない。この場合の正確性は、素人が入力するために入力文字の確認ができることと、入力ミスに対して修正が容易であることが要求される。入力文字の確認は、モニタ表示あるいはモニタ印字を入力と同時にとれることが必要であり、漢字ディスプレイや印字機構の付いた入力装置が望まれる。そして確認によって入力ミスを発見したとき、修正を行なうための制御機能が要求される。ディスプレイ・キーボードでは、画面制御機能が、キーボード・プリンタには、訂正・編集機能が必要である。

素人による入力のため、操作はワンタッチ式の入力装置が簡単であり、フルキー方式の中でもテーブル・ルック・アップ（タブレット型）方式がもっとも適用しやすい。また会話型の入力方式として簡易なカナ漢字変換方式やマルチ・ストローク方式も効果的である。

タブレット型の入力装置には、入力文字の確認機能として、各文字キーに発光ダイオード（LED）やランプを付けて、入力打鍵時に点灯されるものもある。また入力したい文字がさがしやすいようにガイダンス機能をもった入力装置がある。漢字入力用ではないが、同じタブレット型でオフィス・コンピュータの入力に多く使用されているコードレス入力方式も、商品名や会社名をワンタッチ入力できることから、日本語情報の入力方式として、たとえば漢字単語の入力や固有名詞の入力あるいはキーワードの入力といった利用に、大いに活用できる方式である。ブック型になったシート上に頻度順や五十音順、項目別配列などに工夫すれば、漢字1字1字を探す

フルキー方式よりも、素人にとってはスピード・アップがはかれることが期待される。

マルチ・ストローク方式の入力装置は、入力の簡易化をはかるために各種の工夫がなされており、漢字を探すことなく、読み仮名の第1字を入力すれば、それに対応した漢字がすべて表示され、その中から該当する漢字を指示すればよい。また読みがわからない漢字は部首などを指示することによって、それに類する漢字群が表示されて、指示・選択すれば入力できる。あるいはまた、これらを組合せて同類漢字を減少させて該当文字に近づける方法をとっているものもある。この方式の欠点は各漢字ごとに多打鍵（2打以上）を必らず必要とするために入力速度が遅いことである。そのため漢字入力を専門とする業務には不適当で、素人の一般事務員が少量のデータを入力する場合に有効である。

表3－3　漢字入力方式の比較

方式		速度	訓練度	特徴
フルキー方式	漢字テレタイプ方式	40～80字/分	中大	漢字入力装置としては最も古く、現在、最も使われている。モニタ印字が得られず専門オペレータ向き。
	和文タイプライタ方式	30～50字/分	中中	和文タイプライタのオペレータがそのまま使用できる。モニタ印字が得られるので誤入力が少ない。
	テーブル・ルック・アップ方式（タブレット）	30～70字/分	中小	方式としては新しく、全文字がコンパクトにまとまっており、目視面積が少なく、片手で入力できる。入力の確認もできる。
連想コード方式		60～100字/分	大	記憶訓練がむずかしく、収容文字数が限定される。
速記入力方式		200語/分	大大	
マルチストローク方式表示選択型（音訓・部首）		20～30字/分	小	一般素人向き、ディスプレイ装置を使用するため装置がやや高価
カナ漢字変換方式		ソフト依存度により異なる。	小	会話型にすると素人向けになるが、ソフトの規模・機能によって入力速度・価格は異なる。

3.3　出力の考察

日本語情報処理の主要な機能は、漢字の出力であり、漢字入力装置のハードウェア・コストも出力装置が占める割合が大きい。その中でも、多字種の漢字を扱うために文字発生装置が高価格をきたし、各メーカ・研究機関における技術開発も、いろいろな角度から対策をはかっている。またユーザ側か

らは多種多様な要望もあって、その対処法も様々である。漢字出力装置の現状については前述したが、ここでは出力の問題点をとりあげて考察することにする。

(1) 文字発生方式と記憶素子

漢字出力装置を構成する機器の中で、もっとも重要なものとして文字発生装置がある。漢字は英数字やひらがな、カタカナなどと比較して文字の種類が多く、しかも個々のパターンが複雑であるために、文字発生器に漢字パターンを記憶(記録)するのに膨大な容量を必要とする。そのためコスト的にも出力装置の大きな部分を占めることになる。

漢字出力装置に使用されている文字発生方式は、第2章で述べたように多くの種類がある。文字発生方式を大別すると、アナログ型とディジタル型に分類することができる。アナログ型の文字発生方式は、字母型(字母アナログ方式、字母光学方式)や活字打鍵方式のように文字パターンをアナログ的に、フィルムや活字などの形で持つ方式である。アナログ型の文字発生方式の特徴は、一般にディジタル型に比べて高品質の文字を印字することが可能であるが、装置は大型化して価格も概して高い。しかも文字発生装置のマルチ・ステーション化は困難である。

一方のディジタル型の文字発生方式では、コンピュータの記憶装置に使われているディジタル・メモリを使用することができるという特徴がある。そのため最近のLSI 技術の進歩や、周辺記憶装置の高密度・大容量化も進んで、ビット当りの単価が下降しつつあり、低価格を目指す漢字出力装置への適用が注目されている。ディジタル型の文字発生方式は種々あるが、いずれも多量のメモリを必要とする難点がある。その中でもっとも期待されるのはパターン・メモリ圧縮技術が利用できるドット方式である。ドット(・マトリックス)方式は、表現の自由度が高く、コンピュータ処理で扱いやすいことから、今後の漢字出力装置の主流となるであろう。

ディジタル型の文字発生方式で使用されるフォント用記憶素子として、磁性線、コア・メモリ、ICメモリ、ホログラム・メモリ、磁気バブル・メモリ、磁気ディスク、磁気ドラム、光ディスク・メモリなどがある。これら記憶素子の特徴を比較検討すると以下のようである。磁性線、コア・メモリは最も古くから使われていて、高速読み出しに適しているが、メモリ容量が大きくなると価格的に問題がある。ICメモリ、特にマスクROM は量産効果が最も期待できるが、少量多品種を必要とする漢字パターン・メモリでは、あまり需要が期待できないが、磁気ディスクの大容量性とMOS LSI の高速性を組合せた使い方が実用になっている。しかし、MOS LSI の容量によって価格が左右されるため、その利用分野は限定されてしまう。また、ホログラム・メモリは大容量化の容易さと光制御の高速性を利用して実用化されているが、光電変換部において処理に時間がかかり、実用的には読み出し速度は毎秒1,000字程度である。しかも、記録媒体の書き込み技術、あるいは複写技術について問題が残っている。磁気バブル・メモリは将来のメモリとして、大容量化の可能性と、高信頼性から最も期待されており、価格的にもVLSI(Very Large Scale Integration)と同程度かそれ以上の見通しがある。磁気ディスク、磁気ドラム、および光ディスク・メモリなど、回転しているメモリ媒体と、静止している読み取りヘッドから構成されている記憶素子では、一定の速度で回転している回転体あるいは回転板から信号を

読み出す関係から高速性は望めなく、通常、毎秒60～180 字程度が限界である。しかし、現状では最も低価格のメモリである。

表3－4　パターン・メモリと読み出し速度

読み出し速度（字／秒）	接　続　機　種	メモリの種類
高速（1,000以上）	漢字ディスプレイ レーザ・プリンタ 光静電プリンタ	磁性線、コア マスクROM IC+磁気ディスク
中速（数百～1,000）	静電プリンタ	（フライングスポット） ホログラム 磁気バブル
低速（数十～数百）	インク・ジェット・プリンタ ワイヤ・ドット・プリンタ サーマル・プリンタ	磁気ディスク／ドラム 光ディスク

ドット・マトリックス方式で漢字を表現しようとすると、少なくとも16×18ドットは必要であり、5×7ドットの英数字や7×9ドットのカナ文字に比べて大量のメモリを必要とする。各漢字を表現するのに、ドット数の増減は文字品質に直接えいきょうしてくる。このドット数と文字の品質との関係を表3－5に示す。

表3－5　ドット数と文字品質の関係

ドット数	文　字　の　品　質
18×18ドット	文字の骨組を表現することしかできない。画数の多い字に対しては略字体を用いなければ表現できない。たとえば、当用漢字の中でも「曇」「警」「響」「襲」の4字は表現できないほか、約4,000字の漢字に対して約30字の漢字が略体となる。この文字は帳票用を主とした特殊用途の文字である。
24×24ドット	ほとんどの文字を略字体なしで正確に表現することができ、明朝体での表現もある程度できる。
32×32ドット	全ての文字を正確に表現することができ、明朝体、ゴシック体の区別が行なえる。しかし、20画以上の文字になるとこの区別が難かしい文字もでてくる。斜線、曲線の表現が難かしい。
64×64ドット	書体の区別は確実に行なえるが、明朝体の線端の表現および、ひらがなの曲線の微妙な差異が表現しにくい。一般の事務用文書や帳票の版下として使用できる。
96×96ドット以上	線の太さ、線端のアクセント、曲線なども十分に表現できる。商用印刷にも使用することも可能である。

もし、２４×２４ドットで漢字パターンを表現すると、2,000字分のメモリ容量は、およそ１４４Ｋバイトになる。このような大量のメモリを必要とするために、文字発生装置、しいては漢字処理装置の価格を高める原因になっている。そのため、フォント・メモリをできるだけ少なくしようと、文字パターンの圧縮技術が研究されている。しかしここで注意しなければならないことは、パターン・メモリの圧縮技術の方式評価は、単に圧縮率ばかりでなく、その復元、変換方法の簡易さと、出力文字に要求される各種機能をいかに備えているかも判断しなければならない。

(2) 印字方式と記録紙

漢字プリンタに出力するための印字方式は、記録紙と密接な関係をもっている。印字方式の分類は、インパクト方式とノンインパクト方式とに大別されて、それぞれの特徴を有している。ノンインパクト方式は、騒音が少なく、低速から高速まで適用範囲が広い。しかも高品質文字用から簡易な低品質文字用のプリンタもあり、多種多様な方式をもって、センタのバッチ処理にも、また簡易・小型の端末用にも使用されている。これに対して、ノンインパクト方式ではコピーがとれないという特徴があるために適用業務によってはインパクト方式が利用される。インパクト方式には、活字インパクト方式とドット・インパクト方式があるが、前者では出力速度が遅く、後者ではワイヤ・ピンの摩耗とドット文字の精度に限界がある。

一方、記録される用紙の面からは、次のような比較ができる。

表３－６　印字方式と記録紙

印字方式		記録紙	関連文字発生方式	速度（字／秒）	解像力（本／mm）
インパクト方式	活字インパクト	普通紙	活字	2	
	ドット・インパクト	普通紙	ドット	１０～１００	4
ノンインパクト方式	感熱記録	感熱紙	ドット	１０～３００	4～20
	インクジェット	普通紙	ドット，ストローク	１０	
	静電記録	静電記録紙	ドット	１００～1,000	4～8
	銀塩写真	銀塩写真紙	字母	～2,000	１５
	湿式電子写真	酸化亜鉛紙	ドット	～2,000	１０～１２
	乾式電子写真	普通紙	ドット	～4,000	6～１２
	インクミント	普通紙	ドット	～8,000	4～8
	放電破壊	アルミ蒸着紙	ドット	２０～３００	

普通紙に印字できる方式は、用紙が安価で運用コストを低くおさえることができ、しかもコントラストが良い、プリプリント用紙を自由に使用できるなどの長所を有している。漢字プリンタの使い分けは、センタ用に高速出力と普通紙への印字が要望され、また端末用には低価格で小型

の装置が要件となり、できれば操作管理の容易な普通紙への印字ができることが望まれる。

一方、これらの印字方式によって出力される文字品質の要因としては、①解像度（文字パターン発生部の絵素数、出力印字部の解像力）、②濃度（均一性、絶対値、コントラスト）、③幾何学的歪（字並び、直線性、大きさ歪）、④品位（文字のデザインの美しさ、デザインのフォント内のバランス、線画の太さの均一性）、があげられる。これらの要因は、漢字プリンタの適用業務によって要望値が異なり、一般情報処理では低品質であってもよく、商業印刷・写真植字用では高品質文字の印字が要求される。

表３－７　印字方式の特徴

センタ（高速大形向き）	インクミスト	(利)・普通紙を使用、約9,000字／秒の高速 (欠)・ドット密度があらい
	乾式電子写真方式（PPC, Plain Paper Copy）	(利)・普通紙を使用、印字品質良好、高速 (欠)・装置が大型になりやすい
端末（小形向き）	湿式電子写真方式（湿式エレクトロファックス）	(利)・印字品質良好、かなり高速 (欠)・紙が高い（酸化亜鉛紙）
	静電記録方式	(利)・印字品質良好 (欠)・紙がやや高い
	光静電記録方式	(利)・文字メモリ安価、印字品質良好 (欠)・低速度、紙がやや高い
	感熱記録方式	(利)・装置簡単、紙安価 (欠)・低速度、記録の保存性に難
	インクジェット	(利)・普通紙を使用、装置簡単 (欠)・低速、ドット密度があらい
	ワイヤインパクト	(利)・普通紙を使用、装置簡単、コピー可 (欠)・ドット密度があらい、低速、騒音やや大
	活字インパクト	(利)・普通紙、コピー化、印字良 (欠)・騒音、極端に低速（３字／秒程度）
	放電破壊方式	(利)・装置の保守取扱が容易で小型 (欠)・特殊紙を使用、記録中に臭いとカスを発生

表３－８　印字品質にたいする性能

分類	解像力	文字寄り引き	濃度	１文字に必要なドット数		
				プリンタ解像力	８ポイント	１２ポイント
一般用モニタ	４本／㎜以上	1.0㎜以下	2.0～2.5	４本／㎜	１１×１１	１７×１７
				６〃	１７×１７	２５×２５
				８〃	２２×２２	３２×３２
軽印刷	１０〃	0.2㎜以下	2.2～2.5	１０〃	２８×２８	４２×４２
				１５〃	４２×４２	６３×６３
植字用版下	２０〃	0.1㎜以下	2.5±0.1	２０〃	５６×５６	８４×８４

(3)　適用業務からみた出力

①　編集・印刷分野

この分野におけるコンピュータを利用した漢字処理は、漢字プリンタから出力される印字物がそのまま出版・印刷されるために、高品質の印字が要求される。人手による鉛活字や写真植字と同程度の品質で組版されなければならない。そこには文字ソフトウェアと呼ばれる、文字の種類、大きさ、字形、書体、などの要求も多様である。これらの条件に適した出力方式は、字母アナログ方式、多くのドットを扱ったドット・マトリックス方式、ラインドット方式、それに字母光学方式などである。

字母アナログ方式のフライングスポット方式は、多種文字を高速に、高品質で、任意の形、大きさで文字発生・印字するのに適している。ドット・マトリックス方式では商業印刷用として、９ポイント文字を１００×１００ドット程度の高品質文字によって適用可能である。またライン・ドット方式も、分解能を上げて、ライン分割数を増せば非常に高品質の文字を出力できる。字母光学方式は、本来電算写植用に開発されたものであり、文字品質は高い。しかし光学機械部分を含んでいるために、字母アナログ方式に比べて速度が遅くなる。

一般に高品質文字を印字でできる出力装置は高価であり、そのため最終原稿の印刷時にのみ使用して、校正用モニタには高速で普通紙に印字でき、運用コストの安い簡易なドット式の出力装置を使うなど工夫する必要がある。

②　事務処理分野

事務処理における漢字化項目は、姓名や住所、社名などのあて名書き用と、商品や物品の名称、あるいは病名、薬品名といった固有名詞が多い。そのため出力に対する要求は、文字種が他の分野に比べて多く必要とすることである。印字品質は、公的な書類、証書類にはある程度の高品質文字が要求されるが、社内文書や顧客サービスを目的とした漢字出力では、それほど品質に対する要求は少ない。とくに後者に至っては、これまでカナ文字やローマ字、英字などで処理していたものが漢字化されたということだけでも大きな効果があるからである。これに

対して前者の場合は、逆に印刷活字・タイプ活字などで処理していたために、コンピュータ処理の漢字化にあたっても高品質文字の要求が高い傾向にある。

この分野の漢字出力の特徴は、ある周期をもって、一時的に大量出力があって、それ以外はあまり使用率が少ないことである。そのため、一時的使用のために高速な出力装置が必要となる。たとえば病院のレセプト業務のように、月末から翌月1週間に大量出力が集中するようなケースである。またあて名印刷なども、発送時期直前に集中処理しなければならない。このようなことから、印字品質はドット方式で十分とされ、高速印字のできる漢字ラインプリンタの使用が適している。この場合の記録紙は普通紙の要求が高い。

また、漢字出力装置の有効利用をはかるために、他の業務の漢字処理が要求される場合には自社導入し、特定業務以外は使用されることがない場合には、共同導入や計算センターの利用が効率がよい。

③ 情報検索分野

この分野の漢字出力は、オンライン端末を利用して、センタのコンピュータに問合せた結果が出力されるという使用法が多い。そのため漢字プリンタ主体というよりも漢字ディスプレイを使用する場合が多くなり、そのハードコピーとして漢字プリンタないし画面のハードコピー装置を接続使用するようになる。

出力機能は、文字種もそれほど多く必要としないで、それよりも技術文献情報などの出力では出力文字数が多いため、ディスプレイ画面への表示文字数を多く必要とされる。また漢字名寄せなどの利用では、姓名索引が必要となって文字種を多く必要とする。その他、文字品質については中程度以下でよく、また出力速度も端末装置としては中速、低速で十分である。情報検索結果を一括大量に出力する場合には、センタに高速プリンタを置いて、共用する方が効果的である。ディスプレイ画面のハードコピー装置としては、画面に表示された情報のメモ書き程度のもので十分とされ、低速で安価なものでよい。

このようなことから、情報検索分野における漢字出力装置は、事務処理分野と同様に、ドット方式の出力装置で十分である。適用業務によっては、複数コピーのとれるインパクト型のものも利用価値が高い。

(4) ディスプレイ方式の考察

2章ですでに述べたように漢字ディスプレイ装置はコード・リフレッシュ型とパターン・メモリ型に分けることができる。コード・リフレッシュ方式では文字発生装置を専有してしまうため、複数台のCRTに対し同数の高価な漢字文字発生装置を必要とする。また、コード・バッファを一定周期（普通1/30秒に1回）ごとに読み、その都度文字発生装置から文字パターンを発生させることから、文字発生装置に対して厳しい仕様が要求される。

一方、パターン・メモリ型では文字発生装置を共用させて何台かのディスプレイを接続することもできる。パターン・メモリ型はパターン・リフレッシュ型と蓄積管型とに分けられる。蓄積管型では、リフレッシュさせていないため、ちらつきはなく見やすいが、画面に表示された文字

の修正が難かしい。一方のパターン・リフレッシュ型は1画面分のパターン・リフレッシュ・メモリを個々のディスプレイに持たせなければならない。パターン・リフレッシュ・メモリは1画面に何字表示するかによって決まり、何画面分持ってもよい。1画面に、２００字表示し、1文字を24×24ドットで表わすとすると、1画面では、約１４Ｋバイトを必要とする。ここでは1画面２００字としたが、一般的には６００字表示できるのが理想とされるため、この3倍の４２Ｋバイトを持つことも考えられる。

表示画面の品質として1画面に表示できる文字数は500～600 字程度表示できること。表示速度は使用者から見て遅いと感じない程度であること。画面のフリッカは目を疲れさせる原因になるため、極力、押えることなどが望まれる($^{1}/_{30}$ 秒に1回リフレッシュ)。また、印刷物のレイアウトを画面上で決める場合などでは、文字品質より表示文字数が重視される。その場合、文字サイズを縮小して文字数を増加できる機能があると便利である。たとえば英数字を主体とした文書の場合、ハーフ文字(縦は同じだが、横が普通の半分)で表示でき、これらの文字と通常の文字を混在表示できると便利である。また、画面制御としては、画面上の編集効率を上げるために、カーソル制御や画面上の文字の抹消や訂正を行ないやすくするためのリフレッシュ機能を持つこと、同一画面上に図形と文字を混在表示できることが望まれる。以上の機能は利用者にとって使いやすい漢字ディスプレイの機能であるが、制御部分が大きくなり、ミニ・コンピュータを専用に使ったり、メイン・コンピュータのプログラムに端末制御のかなりの部分をまかせなければならない欠点がある。

4. 日本語端末の構成方法と処理機能

前章までに日本語情報処理に関連した技術動向と、その技術的・総合的な問題点の考察を述べてきた。その結果として、今後の日本語情報処理の適用で最も期待されるのは、コンピュータ端末装置の漢字化であり、そこには漢字入出力装置の低価格化が大きな課題としてとらえることができよう。これまでの漢字入出力装置を分析してみると、日本語の特質から、取扱い文字種の多さにともない、従来の英数カナ文字の入出力装置に比較して、装置の価格が非常に高価であり、それに係わるデータ作成費用も倍加している。今日、日本語情報処理の普及が遅れているのもそこに大きな原因がある。

一般の英数カナ文字用の周辺端末装置の分野でも、最近のコンピュータの普及にともない、低価格の機器・装置や、簡単に操作できるものへとその傾向を示しており、市場におけるニーズも当然その方向を示している。またバッチ処理ばかりでなく、オンライン化傾向が強まってきており、同時に高速処理、データ処理内容の向上、スループットの向上を可能とする製品の開発が進んでいる。とくにオンライン化への進展により、データの分散処理が必要となり、端末機器のインテリジェント化が急速に進んでいることが大きな傾向としてみられる。分散処理端末化は、専門でない人たちが必要に応じていつでも操作を行なうことが前提となり、だれでも容易に操作できる端末機が望まれる。

これらのことは、日本語情報処理においてもいえることで、これまで実例の少なかった日本語のオンライン処理も、低価格の漢字入出力装置が出現することによって大きく変化するにちがいない。

コンピュータ周辺端末装置の漢字化にあたって、装置を構成する漢字入出力機器に要求されることは、“簡易な入出力方式によって端末装置が構成でき、そこには操作性と処理機能の最適化をはかって、低価格の日本語端末を構築する”ことである。

ここで我々が研究開発を目指す日本語端末は、この目標に一歩でも近づけるために、入出力装置の構成方法と処理機能に検討を加えて、モデル構築のための基本的事項をとりまとめたものである。特に重点的課題としてとりあげたのは、第1に日本語端末全体の低価格化を目指すことであり、次に、入力方式の簡易化、文字パターン発生方法の最適化、出力・表示方式の簡易化などであり、それぞれについて検討した結果を述べている。

以下本章で記述する内容は、まず第1節（4.1）で、日本語端末がもたなければならない基本的機能について述べる。第2節（4.2）では、日本語端末の基本機能を発揮するための構成要素として、入力機器、出力機器、文字発生器、制御装置の組合せ選択方法を検討し、その構成方法による処理機能について各々の特徴を述べる。第3節（4.3）では、日本語端末の主要な機能として必要不可欠な文字発生器の機能を検討し、とくに文字パターンの記憶方法とその復元に関する処理機能を分析する。第4節（4.4）では、オンライン対話型で日本語情報を得るのに主要な構成要素である漢字ディスプレイの表示機能と、ハードコピーとしての漢字プリンタの出力・印字機能について考察を述べる。

4.1 日本語端末の基本機能

日本語情報の入出力処理を行なうための端末装置を"日本語端末(装置)"と定義して使用する。漢字かなまじりの日本語情報をコンピュータに入力する方法は、漢字そのもののパターンを人手を介さないで入力する方式と、漢字を何等かの識別コードに符号化して人手を介して入力する方式とに分けられている。前者は、漢字パターンを直接OCR装置などから入力するものと、音声パターンから日本語情報を認識して入力するものとがある。これらはパターン認識技術を利用した入力方法であるが、現状では実用化されていない。後者の方式は、漢字を数字や英数字・カナなどで組合せたコードで入力するものと、漢字キーボードから漢字コードを発生させて入力するものとがある。これらはいずれも人間(オペレータ)がキーボードを操作して入力する方法であり、多字種の漢字の中から1字1字を探しながら、あるいは各漢字に対応したコードを記憶の中から探し出して入力しなければならない。実用化している入力装置のほとんどが漢字キーボードから入力する方式であり、一部にコード入力が使用されている。

一方、日本語情報の出力は、文字発生装置を介して、漢字コード/英数・カナ・コードから漢字パターンを発生させ、漢字プリンタあるいは漢字ディスプレイに出力・表示させる方法がとられている。漢字パターンの発生方法は光学的な技術を利用したアナログ型のものと、漢字を点素や線素などの画素に分解してオン/オフ情報あるいはベクトル情報などに数値化して記憶し、これを復元して文字発生させるディジタル型のものとに大別される。日本語情報を出力する装置は、日本語端末の利用者に漢字かなまじりの日本語情報を提供サービスするために必要不可欠な要素である。

オンライン端末の使用にあたっては、一般に入出力される情報が短かいという特徴と、頻繁なメッセージ交換が長時間にわたって繰り返されるという特徴がある。これは英数カナ文字処理用端末、日本語端末にかかわらず、マン・マシン・システム(質問応答系の会話型システム)としての特徴といえる。これらの特徴によって端末装置に要望されることは、端末利用者にとって、入力の面では操作が容易であること、出力の面では見やすく理解しやすいこと、そして長時間使用しても疲れないといった機能を考慮しなければならない。とくに出力機能では、日本語情報を出力するという、日本語端末の主要な目的を達成することができる。

日本語端末の入力機能は、オフラインの入力装置のような漢字入力専用端末でなく、また校正や修正・編集用に用いられる漢字入力モニタ表示装置といったものでもない。人間とコンピュータとの間の活発な情報交換を行なうための入力であることから、人間(端末利用者)がコンピュータ内に格納されている情報を探索するのに使いやすい便利な機能を有さなければならない。一方の出力機能は、入力に対するモニタ表示装置とかモニタ印字装置として使用するのが目標ではなく、コンピュータから得られる日本語情報を利用者に正確に伝えることを前提にし、そのため見やすく利用しやすい出力機能でなくてはいけない。

また日本語端末の利用者は、漢字入力せん孔業務の専任パンチャや専任オペレータでなく、多くはコンピュータ業務に無縁な技術者であったり事務員であり、日本語端末の入力操作性が重視される。一般に入力装置の要件は、入力操作性の他に、入力速度、正確性、および修正のしやすさなど

があり、日本語端末のように"誰もが簡単な操作で入力できる"ことに重点を置けば入力速度はダウンする傾向にある。しかも不なれな人が操作するために修正機能やガイダンス機能などの要求もある。これらは入力機能としてばかりか、表示・出力機能として動作するキーボード・ディスプレイやキーボータ・プリンタ、タイプライタに装備される機能でもある。この場合には、入力モニタ表示(印字)機構も含まれて、文字発生器が必要である。

日本語端末の機能は、こうして入力機能、出力(表示)機能、そして出力のための文字発生機能と、端末の制御機能、とに大別できる。この中から日本語端末の基本機能あるいは最小機能を設定しようとすれば、日本語端末の定義から、漢字かなまじりの日本語情報を提供サービスするために出力(表示)機能が必須条件である。そして出力・提供サービスを受けるための会話・問合せのできる入力機能があれば、日本語端末を構築することができる。

4.2. 日本語端末の構成法

日本語端末の機能を発揮するためには、各種の漢字入出力機器および関連機器を組合せて端末装置を構築しなければならない。日本語端末の構成機器を、入力機器、出力機器に大別すると別表のような各種の入出力機器が適用可能である。このうち日本語端末の出力機能は、文字発生機能が主要部分を占めており、文字発生方式の分類によって文字パターン(フォント)の記憶形態も様々である。そのため日本語端末の入出力機器の構成のほか、文字発生器とその制御装置の機能とが重要な関連性をもってくる。

日本語端末の構成を、基本機能をもとに簡易化と低価格化を目指して最適化をはかろうとすると、コンピュータ端末、コンピュータ処理を大前提にするため、必然的に処理しやすい入出力方式と文字発生方式が選択されることになる。文字発生方式を大別するとディジタル型とアナログ型に分けられるが、文字発生機構が簡単で小型化の容易なディジタル型の文字発生方式の採用が有望視されている。しかも、最近のＬＳＩ技術や周辺記憶装置の急速な発展ともあいまって、ディジタル記憶の大容量化、高密度化がはかれ、ますます低価格化が期待できる。さらに文字パターンの記憶にも、融通性、精度、調整のしやすさ、などの点の優れた特長をもつディジタル方式を、この日本語端末に採用することにする。

文字パターンをディジタル方式で記憶するには、記憶方法とその特徴とにより、また記憶素子・装置の選択・組合せによって、各種の文字発生装置を構成することができる。その基準は、主に記憶容量、アクセス速度、価格の3点に、端末装置としての記憶部と考えて、高密度、小型化の可能性を検討しなければならない。

表 4 － 1　入力機器

入力機器	入力方式
テンキー プッシュホン ダイヤル	数字コード入力
英数字・キーボード	(英数字) コード入力 ローマ字入力
英数字カナ・キーボード	(英数字カナ) コード入力 カタカナ入力 (カナ漢字変換) 連想コード入力
タブレット・キーボード	テーブル・ルックアップ入力 ペンタッチ入力 コードレス入力 (項目入力) 単語 (キーワード) 入力
漢字キーボード	漢字入力 フルキー入力
OCR	OCR入力 印刷漢字認識 手書　〃
音声入力装置	音声入力

表 4 － 2　出力機器

出力機器	出力形態
漢字ディスプレイ グラフィック・ディスプレイ テレビ マイクロフィルム表示 (リーダ) スライド映写 電光掲示	漢字 (図形) 表示 (ソフトコピー)
漢字プリンタ 漢字ラインプリンタ 漢字タイプライタ 画面ハードコピー装置 XYプロッタ ファクシミリ受信機 マイクロフィルム・プリンタ	漢字出力 (ハードコピー)
音声出力装置	音声出力

表４－３　文字発生形態

文字発生方式	フォント形態
活字打鍵方式	活字母型
ディジタル方式	ドット、ストローク
アナログ方式	字母アナログ、光学母型
合成方式	構成要素（母型，ドット）

日本語端末の構成は、入力機器、出力機器および文字発生装置（文字発生方式）とその制御装置（制御プロセッサ）とを種々組合せて構築することができる。

まずこれまでの日本語情報処理でオンライン端末あるいはそれに類似した使い方をしているユーザの実例と、研究機関における構成例を参考（2.2節参照）にして構成法を分類してみると以下のようになる。

〔構成①〕入力機器……漢字キーボード

出力機器……漢字印字機構

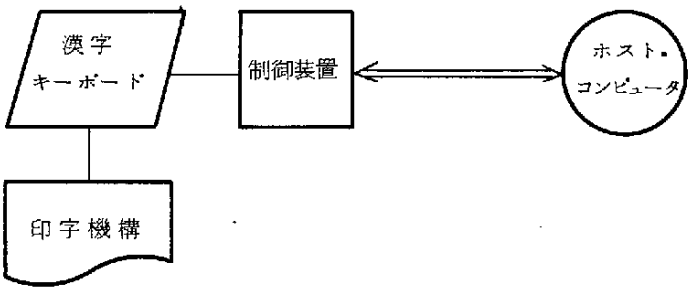


図４－１　構成①

この構成は漢字テレタイプライタ（漢テレ：漢字印刷電信装置）型のもので、和文タイプライタの応用的なものも含まれる。漢字キーボードからの打鍵と同時に印字とコード発信を行なう機械的制御機構のもので、プロセスとしての制御機能はほとんどない。印字速度も機械式のため極端に遅い。

〔構　成　②〕　入力機器……漢字キーボード

（フルキー／タブレット）

出力機器……漢字表示機器

（漢字ディスプレイ）

〔オプション〕……ハードコピー装置

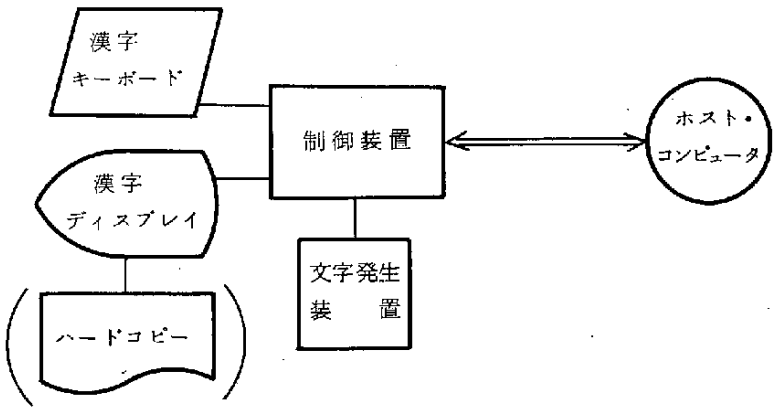


図４－２　構成②

この構成は、漢字ディスプレイ端末の標準的なもので、オプションとして表示画面のハードコピー装置が接続される型である。制御装置の機能としては、漢字キーボードから打鍵発信した信号から文字パターンを発生して、漢字ディスプレイに表示するとともに、ホストコンピュータ（あるいは他のプロセッサ）に入力情報を転送する。ディスプレイの画面の制御に対しても制御コードの入力によって機能する。文字発生装置の機能は、キーボードからの入力コードあるいはホストからの受信コードによって文字パターンをアクセスして、ディスプレイの表示走査方式に合せてパターン情報を発生する。

〔構　成　③〕　入力機器……漢字キーボード

（フルキー／タブレット）

出力機器……漢字表示機器

（漢字ディスプレイ）

漢字出力機器

（漢字プリンタ）

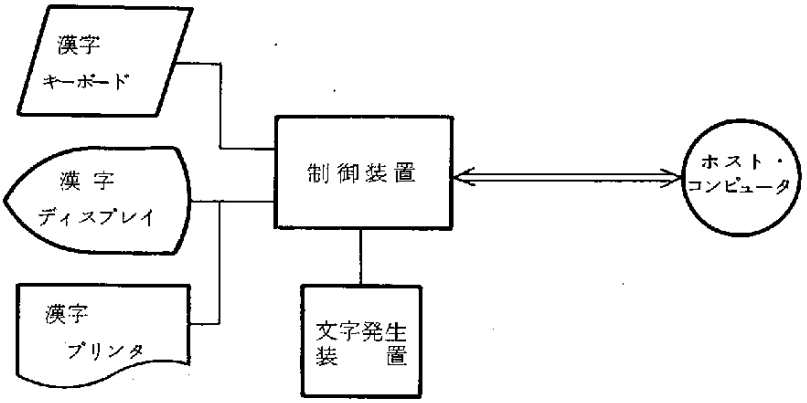


図 4 － 3　構成③

この構成は、漢字入出力端末装置としての標準タイプであり、構成②のハードコピー装置から漢字端末用プリンタに機能アップした構成である。制御装置・文字発生装置のそれぞれの機能は構成②とほぼ同様であり、相異点は、漢字ディスプレイと漢字プリンタの両装置を制御して文字発生装置を共用させることである。この場合とくにディスプレイ、プリンタともに同一走査方式でないと文字発生装置の共有化は当然はかれない。

ハードコピー装置と漢字プリンタの目的は漢字ディスプレイに表示された情報を印字出力することであるが、後者の場合端末での出力編集機能が高度化される。

〔構　成　④〕　入力機器……英数カナ・キーボード

出力機器……漢字表示機器

（漢字ディスプレイ）

〔オプション〕……漢字出力機器

{ ハードコピー装置
{ 漢字プリンタ

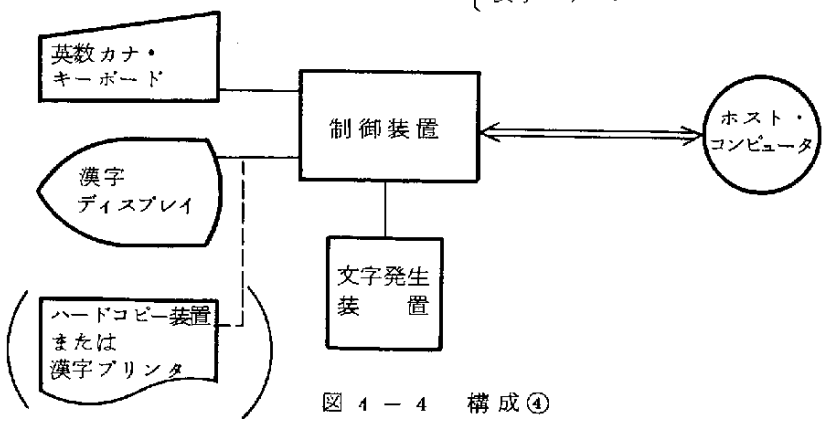


図 4 － 4　構成④

この構成は、簡易な漢字ディスプレイ端末の標準的なもので、①~③との相異点は入力方式の簡略化、換言すれば漢字入力の皆無化である。よって漢字入力の不要なアプリケーション端末に向いている。また特異な使用法としては、漢字のコード入力方式も構成法を同様にして適用できるが、コード変換機能が必要となる。その他、カナ漢字変換方式も同様な構成で可能だが、制御装置の処理機能は②，③の構成よりもカナから漢字へのコード変換機能が加わって多少大きくなる。ただしキーボードからの入力情報を漢字化する必要もなく、ホストへ英数カナ・コード（８ビット・コード）をそのまま転送する処理方式だと簡易化がはかれる。オプションの接続目的は②，③の場合と同様である。

以上の構成①~④は、いずれも各端末装置に対してそれぞれに文字発生装置が接続して、制御装置によってコントロールされている（①は異質）。文字発生装置の規模・機能については、英数カナ・モードの処理系ではその存在すらもあまり問題にされずに各出力・表示装置に組込まれている。それに対して漢字モードの処理系では、漢字という多字種・複雑なパターンを扱わねばならないことから文字発生装置の規模・機能は巨大化し、漢字出力装置、漢字表示装置での占める割合はコスト的にも規模・機能的にも大きくなっている。

文字発生装置の規模・機能を単にディジタル方式の記憶容量としてとらえてみても、英数カナ・モード（７×９ドット／文字×２５６字種）と漢字モード（２４×２４ドット／文字×3,000字種）の比率は１対１００以上にもなる。これが漢字処理装置の高価格化の直接の原因になっている。この高価格化をおさえる技術開発は、現在、ＬＳＩをはじめとした記憶素子の低廉化であり、また文字パターンの圧縮と復元技術の開発であるといわれる。しかしながら現状の文字発生装置（漢字パターン発生）からその対策と対処方法を求めるならば、高価な装置の共有化が最短距離である。上記の構成法でも、その第１段階として漢字ディスプレイと漢字プリンタで文字発生装置の共用がはかられている。

そして次の第２段階としては、日本語端末を複数台で構成する場合には、端末の入出力構成は②~④いずれかで、文字発生装置と制御装置（プロセッサ）を複数端末で共用する、いわゆるマルチステーション化をはかることである。

〔構　成　⑤〕

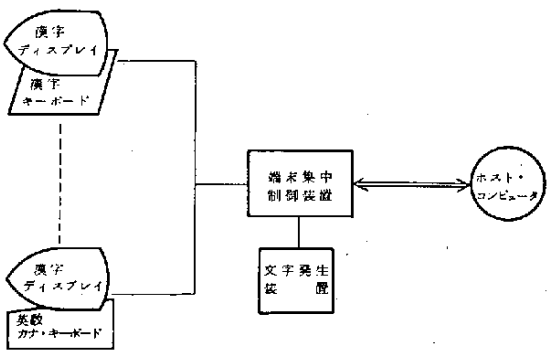


図 4 － 5　構成⑤

この構成では、入出力機器およびオプション装置は構成②～④のいずれかで端末を構成し、いずれも文字発生および出力・表示・走査方式が同一でなければならない。制御装置の機能は複数台端末を集中制御する機能が加わるだけで他は同内容であり、文字発生装置も同様である。端末の接続台数は４～１６台が一般的であるが、複数端末の制御によって、文字発生処理（漢字コード→アドレス変換→パターン・アクセス）の高速化が要求されることになる。もっとも影響するのは文字パターンの記憶方法と記憶素子であり、高速アクセスの可能な記憶装置を採用するとともに、記憶方法も複雑な圧縮技術を安易に採用することは復元の処理が増大することもあり多重利用に不向きな面が生じる。

構成⑤の端末構成と集中制御装置との処理関係から、端末側に入出力に係わる処理機能を分離させて、端末にインテリジェンス性を持たせた構成をとると、次の⑥の構成が可能である。

〔構　成　⑥〕

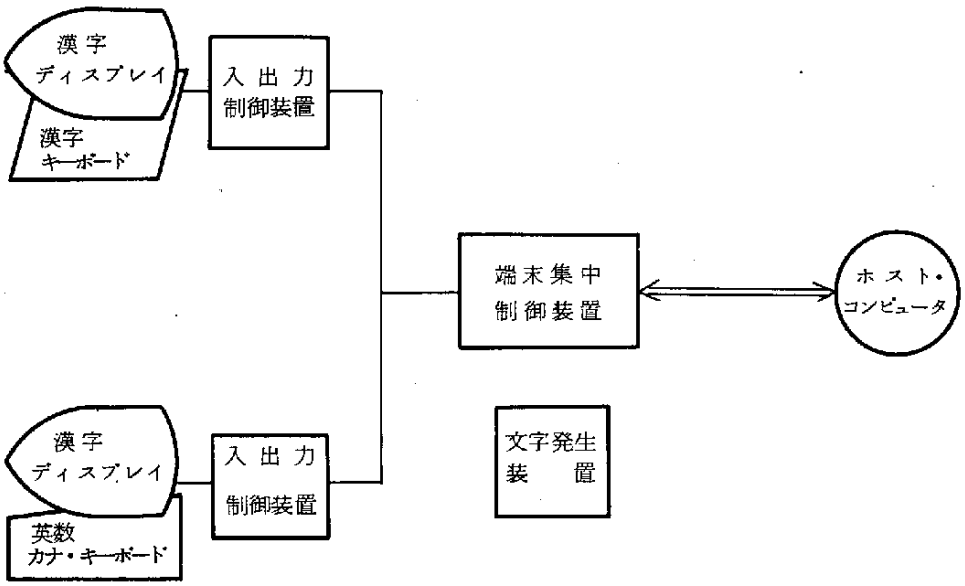


図４－６　構　成　⑥

この構成では、入出力機器は構成⑤の場合と同様に②～④の適用であり、相異点は入出力制御装置（入出力制御プロセッサ）を端末に持たせたことである。入出力制御機能とは前述のコード入力方式やカナ漢字変換方式などのコード変換機能や、出力編集機能など、あるいはそれらの入出力処理の一担をプロセッサにより制御することである。

こうすることで、集中制御装置の負担を軽減し、多重処理、高速化をはかることができるようになる。ただし端末の機能がレベル・アップする反面、端末個々の価格はいくぶん高額になる。

以上、構成②～④をもとにして構成⑤と⑥の構成法とその基本的機能を検討してきたが、これらの構成法はいずれも原則的には文字発生装置と制御装置の処理機能に依存しているといってよい。

そして文字発生装置と制御装置に果せられた処理機能の規模から、これまで制御用プロセッサはミニ・コンピュータが内蔵されるケースがほとんどであった。そのプロセッサのメモリ規模は、光学式の文字発生制御には4KW～8KW(16ビット/W)のものが多く、ディジタル式には16KW～32KWのものが多い。光学式の場合はレンズ機構など複雑な機械制御部が多くソフトウェア制御による部分が少ないために小規模になっている。またディジタル式では、文字パターンのアクセス過程がすべてソフトウェアに依存しており、またアクセス・バッファや入出力バッファのメモリ・サイズが大きいためにソフトウェア・メモリの規模が大きい。さらにディジタル式では文字発生装置のパターン・メモリの記憶容量を節約するために圧縮技術を使用することも多くあり、文字発生の際にはこれを復元する処理が必要である。

このように漢字処理装置の制御用にはミニ・コンピュータ規模の処理機能と処理速度が要求されてきたが、日本語端末の処理機能を、オフライン漢字処理装置と切り離して考え、その機能を制限することによって簡易化をはかることが可能である。

まずその第1は文字種の制限である。簡易な日本語端末の1つの条件として文字種の制約を行ない、専用アプリケーション端末として必要最小限の文字のみを漢字化すればよく、文字発生装置の規模を小型化できる。第2の簡易化は処理機能の縮小化をはかって小規模にする。ディスプレイ画面の制御や出力編集機能をユーザ・サービス機能を落すことのない程度に限定する。第3は適用アプリケーションに適した構成と処理機能の設定を行なうことである。

こうした簡易化を実現する構成を検討すると次のようである。

〔構　成　⑦〕

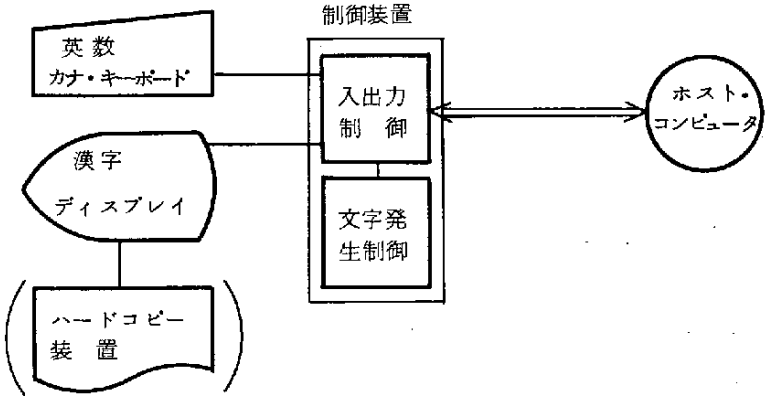


図4－7　構　成　⑦

この構成の入出力機器は④を適用し、制御装置は、入出力制御と文字発生制御を機能するプロセッサで構成される。プロセッサはミニ・コンピュータあるいはマイクロ・コンピュータで制御し、マイクロ・コンピュータの場合はミニ・コンピュータ規模の処理機能を果すためにマルチCPU構成や規模の大きいマイクロ・コンピュータが必要になるであろう。

その可能性を検討するために、構成⑦の入出力構成をモデルにして、全体的な機能・規模を具体化してみると以下のようである。

◎ 文字発生部の構成

- 漢字パターンのドット構成：２４×２４＝５７６ビット＝７２バイト
- 文字種：当用漢字１８５０字＋英数カナ２００種＝２０５０字
- メモリ容量：７２バイト／文字×２０５０字＝147,600バイト

◎ 入力キーボード

- 英数カナ・キーボード：６３キー

◎ 漢字ディスプレイ

- テレビ・モニタ表示：１６字／行×１２行＝１９２字
- パターン・リフレッシュ・メモリ：｛（２４＋4間隔）×１６字／行｝×｛（２４＋8間隔）×１２行｝＝４４８×３８４＝１７2,0 3 2ビット＝２1,５０４バイト

◎ 入出力制御部

- 送受信バッファ：３８４バイト
- プロセッサ：入力制御、送受信制御、文字パターン・アクセス、画面制御……約８Kバイト
  ーマイクロプロセッサ

この構成で使用するメモリは、総計１７７Kバイトになる。このうち主記憶として使うのはプロセッサ部分の８Kバイトであり、他は外部メモリで可能である。とくに文字パターンのメモリはフロッピーディスクが記憶容量の面からも適当な装置として使用できる。またディスプレイのリフレッシュ・メモリは、リード・ライト・メモリのＲＡＭが最適であろう。全体のコスト的な見積りは、ほとんどメモリ価格ともいえるほどで、単純計算で総計約１２０万円と概算できる。実際にはこれにソフトウェア開発費、文字パターン作成費などが加算されよう。

文字発生装置の構成と文字パターンの記憶方法などについては次節（4.3）で、漢字ディスプレイの構成とその機能については4.4で述べる。

4.3. 文字パターン発生方法

4.3.1. 文字パターンの記憶

日本語情報処理全般としての考察の中でも、また漢字出力装置・漢字出力方式に関する問題点考察の中でも、最も大きな問題としてとらえられているのが、文字発生装置の高価格化である。多量な文字種と複雑な文字パターンで構成される漢字を取扱うことから、どのような文字発生方式を採用しようとも、その構成は大規模化し、高度なハードウェア技術をもってようやく現在の日本語情報処理が可能になっている。

このような現状の中で、日本語端末としてその主要な役割りを果す文字発生装置を簡易化し、低価格化をはかるには、様々な要因がからんでくる。その中で最も大きな要因としてあげられるのが文字パターン（フォント）の記憶方法に係わることである。文字パターンの記憶には、文字発生方

式によって種々の技術が採り入れられており、その中から日本語端末が採用する方法を見い出さなければならない。

文字発生方式の分類とその特徴・評価については前述してあるが、ここでは字母アナログ型とディジタル型とにしぼって日本語端末用の文字発生方式を検討してみる。

字母型の文字発生方式には、フライング・スポット方式、ビデコン方式、その他特殊管方式などがあったが、これらの共通的な特徴はディジタル型に比べて、高品質文字の印字ができることである。そこには、光学的な技術が採用されており、装置自体の規模が大型化して、総じて高額である。また日本語端末を想定した場合、複数端末を制御するようなマルチステーション化をはかろうとすると、字母型の文字発生装置では共用が困難とされている。汎用コンピュータのディジタル処理という観点からも、字母フォントへのアドレッシングにはディジタル処理ができても、それ以降の字母のスキャンから印字・表示機構にはコンピュータ処理はあまり生かされないという結果になる。

一方のディジタル型では、文字パターンをディジタル情報に変換して処理するために、磁気コア、磁気ディスク、その他ＩＣ、ＬＳＩメモリなどの一般の記憶素子が使用できるという特徴をもっている。しかも最近の半導体技術の進歩は急速で、今後ますます高密度・大容量メモリの開発が進んで、コンピュータ・メモリとしても普及すれば、記憶素子の低廉化が大いに期待できる。ディジタル型の文字発生方式には、ドット・マトリックス方式、ラインドット方式、ストローク方式があることは前述した。このうち文字パターンの記憶方法が容易で、文字のアクセス表示速度がはやく、表現の自由さ、コンピュータ処理の扱いやすさ、さらには固定長記憶、表示・出力媒体の選択範囲とその多様さなどに特徴を有するドット・マトリックス方式がもっとも有効な方式として適用されよう。

ドット・マトリックス方式の最大の問題点はメモリを大量に使用しなければならないことである（これはディジタル型の文字発生方式すべてにいえる）。２４×２４ドットの文字パターンを例にとれば、日本語端末の取扱う文字種を2,０００字に制限したとしても、（２４×２４ビット／文字）×2,０００字で1,１５２Ｋビット、約１３２Ｋバイトという記憶容量になる。これは英数カナ文字のみを扱ってきた従来の処理系（７×９ドット／文字×１２８種＝8,０６４ビット≒1Ｋバイト）よりも１００倍以上ものメモリを必要とする。このメモリ規模だけでも文字発生装置の価格は高額になってしまう。そのため各種の対処方法が考案されて、各メーカから漢字出力装置・文字発生装置が開発されてきた。

対処方法の代表的なものは、まず低価格の記憶素子を用いることであった。それはビット当りの単価が高いコア・メモリに代って外部記憶媒体を利用することである。しかし単に価格面を重視した装置が望まれるばかりでなく、これまでの漢字出力装置の開発は逆にどちらかというと高速化、多字種、多目的利用を目指して進んできたようである。ここに日本語端末の出現・普及が遅れている原因がある。最近ようやくこの傾向から脱皮して、オンライン利用向けに簡易で低価格の漢字処理装置が出現してきた。そこには前述したような半導体技術の進歩によるところが大きい。また周辺装置としてダイレクト・アクセスが可能で、小型であることから端末用周辺機器として発展のめ

ざましい、フロッピー・ディスク（ディスケット）、ミニ・フロッピー・ディスクの出現が、今後とも高度な技術力をもって適用されるであろう。その他にも、高速不揮発性半導体メモリ、ＣＣＤ（電荷結合素子）、磁気バブル・メモリ等が期待されている。またホログラフィ技術を応用したホログラム方式は、ディジタルな磁気記録よりもメモリの高密度化が期待できる点で将来性があるといわれている。

もう一方の対処方法は、文字パターンのドット・マトリックスを圧縮して記憶する、いわゆるパターン圧縮・復元の技術を適用することである。パターンの圧縮が可能であればメモリの節約ができて、大量メモリの必要性もなくなり低価格化がはかれる。

パターンの圧縮方法は、現在のところ実用例はまだ数少なく、いくつかの方法が用いられて実験研究され、検討されている段階といえる。その代表的な方法は、漢字パターンの圧縮符号化法であり、他にラン・レングス・コード（Run−Length Code）法を利用したもの、漢字イメージのサブパターンを利用したものなどがある。これまでの研究報告によると圧縮率は最高５０％といわれている。

もう１つの圧縮法は、“漢字は縦線よりも横線が多い”、“漢字には１ドットの点は存在しない”という漢字パターンの特性を利用した方法で、この成果は圧縮率５０％といわれる。また漢字の構造面からの特徴を利用して、“へん”や“つくり”など漢字部首や漢字の主要な構成要素をパターンとして用意しておき、これらのパターンを組合せて文字を表わそうという部分パターン合成方式などがある。この方式では圧縮率１０～２５％という研究報告がなされているが、この場合印字品質が極端に悪く自然性を欠き、しかも出力速度が遅いという短所があって実用的でないといわれている。これらの圧縮法の中で、サブパターンを利用した漢字パターンの圧縮符号化法が、アルゴリズムの容易さと、コンピュータ処理（ソフトウェア化）のしやすさから、日本語端末のソフトウェア技術として採用できそうである。この方式を以下に概説する。

この圧縮符号化法では、“漢字パターンは全くランダムなパターンではなく一定の傾向を持っている”という特徴を利用している。文字を表わすドット・マトリックスの任意の一つのドットの状態（白または黒）を考えた場合、その周囲のドットの状態に対して互いに独立ではなく、極めて強い相関関係が存在している。そこで文字パターンを２×２の部分マトリックスに分解してみると、ドット間の相関関係により、２×２のドット・マトリックス１６種の出現する率は確率的に定まってくる。この１６種の文字パターンに対して符号化を行なえば良いのだが、これでは漢字パターンの冗長性を十分利用しているといえない。ある任意の部分パターンを考えた場合、周囲の部分マトリックスの状態に依存していると考えられる。そこで、ある部分マトリックスにおいて特定の部分パターンが生じた場合、その右隣りの部分マトリックスに生じる部分パターンの条件付き確率を考える。これは漢字の特性（縦線より横線の方が多い）を考慮して上下より左右の相関の方が強いことからきている。このようにして得られた確率を利用して、Huffmanの符号化法によりコード化していけば、平均符号長は最小となる。これが漢字パターンの圧縮符号化法である。24×24 ドットの漢字パターンをこの方式で圧縮して、約５０％のフォント・メモリの節減になったとの報告がある。

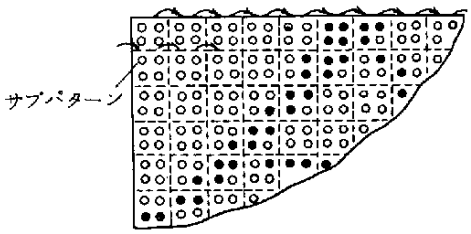


図 4 － 8　2×2の部分マトリックスと符号化順序

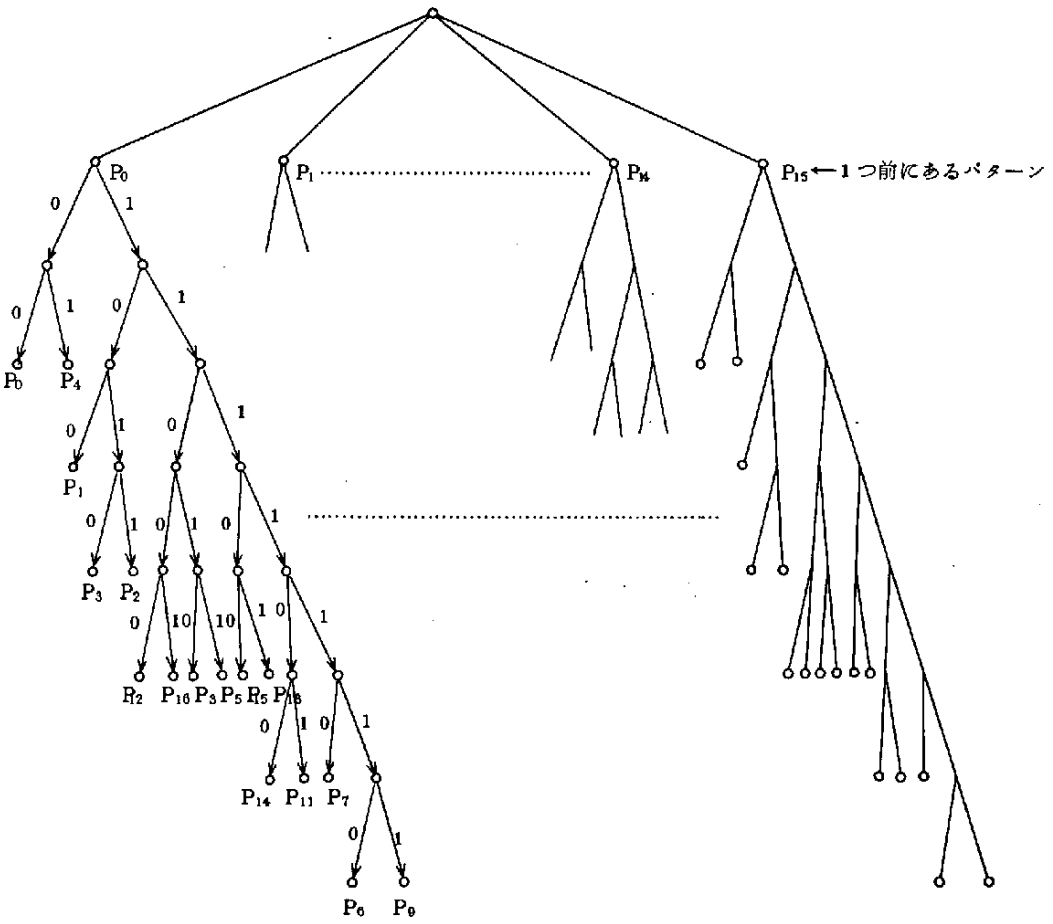


図 4 － 9　先験確率を考えた復号 Tree

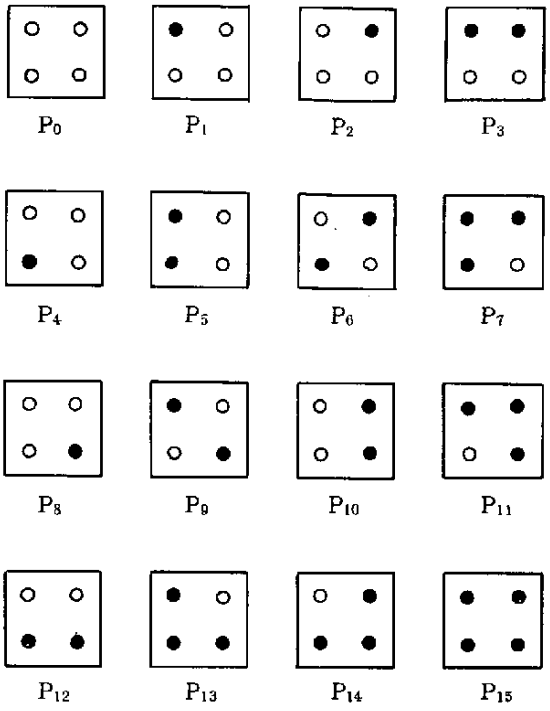


図 4 － 10　　16 個の部分パターン

この圧縮符号化法による文字パターンの圧縮は、文字パターンを記憶装置（素子）に書込んでいく際に、符号化アルゴリズムをソフトウェア化して記憶していく。書込まれた文字パターンは、出力の際に文字発生処理が行なわれ、漢字コードに対応した記憶場所から圧縮パターンをアクセスし、これを復元アルゴリズムによってフルパターンに復元してから出力・表示装置に送られる。この復元のためのソフトウェアは、文字発生装置あるいは制御装置いずれかのプロセッサに持たなければならないが、そのためのソフトウェア規模がどれくらいのものかは明らかにされていない。これは実験研究を通して評価しなければならないであろう。

このように文字パターンの圧縮方法は、単にメモリ節約のための圧縮率ばかりでなく、その復元・変換方法の容易さと、出力文字に要求される文字ソフトウェアの機能をいかに備えているかという総合的な評価が必要である。

4.3.2. 文字発生装置の機能

日本語情報の処理結果を出力・表示する装置には多字種の漢字を扱うために、漢字文字発生装置が分離して独立するほどに規模が大きい。英数字処理系の文字発生器は出力・表示装置に内蔵・組込まれて、その存在すら見つけられないほどに小規模である。

文字発生装置は、制御装置のコントロールのもとに、漢字コードから漢字パターンを発生する機

能を持っている。文字発生方式の分類は前述してあるように各種の方式によって漢字の文字パターンを発生して、出力・表示することができる。文字パターンの発生には、文字パターンを記憶したメモリあるいは機器がなければならないが、前項の考察・検討から、ここでとりあげる日本語端末用の文字発生方式はドット・マトリックス方式にする。

ドット・マトリックス方式における文字パターンの記憶方法は前項で検討したように、ディジタル記憶装置に、漢字のドット・パターンを記憶する方法をとる。ドット・パターンの構成を２４×２４ドット・マトリックスに分解しフルパターンの７２バイト単位で各漢字をメモリに記憶しておく。日本語端末の取扱い文字種を2,０００字に限定すると、その記憶容量は１４４Kバイトである。アドレッシングは2,０００字分のスタート・ポイントがつけられれば実行可能であるが、一般の漢字コードの構成が１２ビットから１６ビット構成であるために、2^{12}＝４０９６字分から2^{16}＝６５,５３６字分のアドレス・ポイントが可能である。（漢字符号の標準化がはかられて、制定されているが、そのコード体系は１４ビットないし１６ビット構成になっている。）また漢字コードと文字パターンの記憶アドレスの関係は、文字種と漢字コードの間には連続コードが付られていないのがほとんどで、漢字コードのシーケンスには空がある。そのた　　コードに対応した記憶場所がメモリの損失になるために、何等かのアルゴリズムで無駄のないコード→アドレス変換ができることが望まれる。

文字発生装置の構成は、図４－１１のようにコード・アドレス変換部、文字パターン用メモリ、制御回路、座標信号発生部から成る。コード・アドレス変換部では、漢字コードから文字パターンの記憶アドレスを求め、このアドレスから文字パターン・メモリをアクセスし、該当する文字パターンをバイト単位にバッファにもってきて、この情報を座標信号発生部で求めた出力座標とともに送出する。制御回路では、これらの文字発生過程を同期をとりながらコントロールしている。

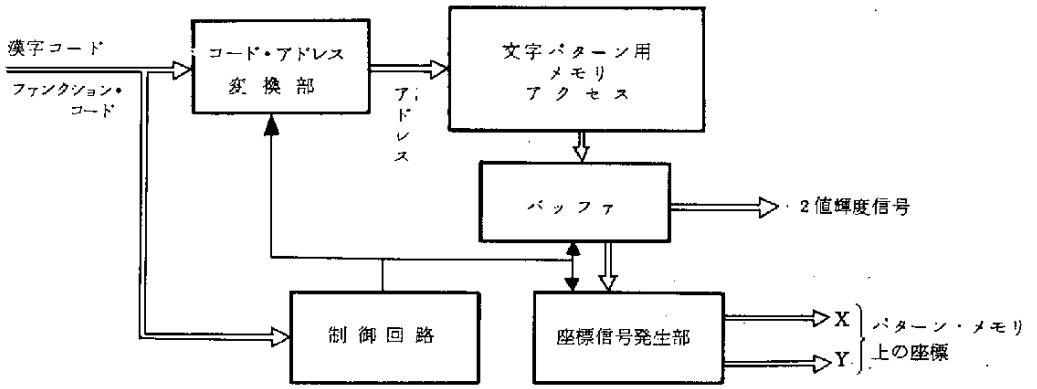


図４－１１　文字発生装置の構成図

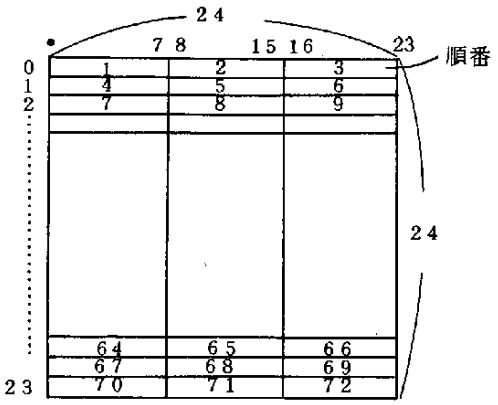


図 4 － 12　文字パターンの記憶順

4.4. 表示・出力機能

日本語端末を利用して情報サービスを受ける方法は、日本語端末を構成する入出力機器によって異なるし、日本語アプリケーションのサービス形態によっても様々な方法がある。日本語端末の基本機能と構成方法から、利用者が情報サービスを受けるためには、主に漢字かなまじり文の表示ができるディスプレイ装置を利用する。また基本機能にオプション付属した漢字プリンタやハードコピー装置には、ディスプレイ画面に表示された情報のハードコピーがとれる他、ソフトコピーなしに直接漢字プリンタへ出力して利用することも可能である。

ソフトコピーの代表的装置である漢字ディスプレイ装置は、コンピュータ周辺端末装置として幅広く利用したい装置でありながら、漢字を扱うために文字発生装置が高額な装置となって、ニーズの高さほどに普及するに至っていない。簡易日本語端末を構成する漢字表示装置としても高額な漢字ディスプレイを使用することは好ましくないが、日本語端末としての最小機能、文字種の限定により、どうにかして低価格の漢字ディスプレイを構築したい。

またハードコピーの情報サービスも要求は高く、大量出力（バッチ処理ほど大量でない程の出力）が要求されるアプリケーション端末ならば、中速以下の漢字プリンタを設置する必要があろうし、小量コピーで使用頻度もそれほど高くないながらもハードコピーがどうしても必要な場合には、低速のプリンタやディスプレイ画面に接続されるハードコピー装置が要求される。この場合でも日本語端末の低価格化をはかるために、簡易なハードコピーが得られるように構築したい。

このようなことから、ここでは日本語端末の基本機能と低価格化を目指した検討を加えて、表示機能・出力機能の最適化について考察することにする。

(1) 表示機能

日本語端末を構成する表示装置として、低価格でコンピュータ制御も可能なことから家庭用テレビまたはモニタ・テレビを適用することにし、その場合の表示機能を設計してみるとおよそ以

下のようである。

テレビ画面の分解能は、一般にX軸（横）方向が２５６～５１２ドット、Y軸（縦）方向が１９２～３８４ドット、にして使用するケースが最も多い。ここに漢字表示させようとすると、２４×２４ドット構成の漢字で、字間、行間を多少とって表示・可能文字数を算出すると、1行当り９～１８字、行数は６～１２行くらいまで可能である。よって画面表示文字数は、９字／行×6行＝５４文字～１８字／行×１２行＝２１６文字になる。

日本語端末を利用するアプリケーションは多方面にわたり、そこに要求される表示文字数も様様であるが、日本語端末の低価格化を目指してテレビ画面への表示を適用するために、ここでは表示可能文字数から制約しておく。なおテレビ画面の分解能は、インタフェース回路その他の改善で１０００文字以上の表示も可能とされている。

よって当日本語端末の表示画面を横に５１２ドット、縦に３８４ドットに分解して、１８字／行×１２行＝２１６文字を表示することにする。これを漢字表示装置あるいは漢字ディスプレイとし、次にその表示方式を検討する。

一般にディスプレイ装置（キャラクタ・ディスプレイ、漢字ディスプレイ）の表示方式はコード・リフレッシュ型とパターン・メモリ型に分けられ、それぞれの特徴についてはすでに前述してある。そこでは文字発生装置を専有化するディスプレイと、共有化できるディスプレイという観点から考察を加えているが、文字発生装置の共有化による低価格化については第２段階のモデル設計で検討することにして、ここでは日本語端末個々に文字発生装置を持つ方式にして検討する。そのためコード・リフレッシュ型、パターン・メモリ型とも対象になるが、文字発生速度の点でパターン・メモリ型が優れている。これは漢字コードから文字パターンへの変換過程をリフレッシュのために繰り返し処理するコード・リフレッシュ型に対し、変換過程は１度のみで、文字パターン以降を繰り返し表示すればよいパターン・メモリ型の方が速度への要求が低く、画面のフリッカの問題も対処できるからである。

文字発生速度の比較評価は、日本語端末の構成要素から重要である。それはメモリ・コストの低減、プロセッサ・コストの低減と直接関係することであり、一般に低価格のメモリ、プロセッサを採用しようとすればアクセス・処理速度は低下する。ＩＣ・ＬＳＩメモリからフロッピー・ディスクの採用、ミニ・コンピュータからマイクロ・コンピュータの採用と、日本語端末の構成要素個々を低価格化の目的で採用せざるを得ないことなどからも、文字発生速度への影響が少ない表示方式をとるべきである。

またパターン・メモリ型は、表示画面サイズ（分解画素）と一致したパターン・メモリを有さなければならないが、パターン・メモリと表示用ＣＲＴを兼用する蓄積管型と、それぞれを別物で構成したパターン・リフレッシュ型に分類される。蓄積管型、パターン・リフレッシュ型それぞれに特異な特徴を有しているが、日本語端末用に適用容易なテレビを用いるためにはパターン・リフレッシュ型に制限される。蓄積管はフリッカがないなどの特徴を有してオンライン端末に向いているが、特殊管を用いた機器のためにテレビに比べ、入手方法も容易でない。

パターン・リフレッシュ方式による漢字ディスプレイの構成は、大よそ図４－１３のようである。

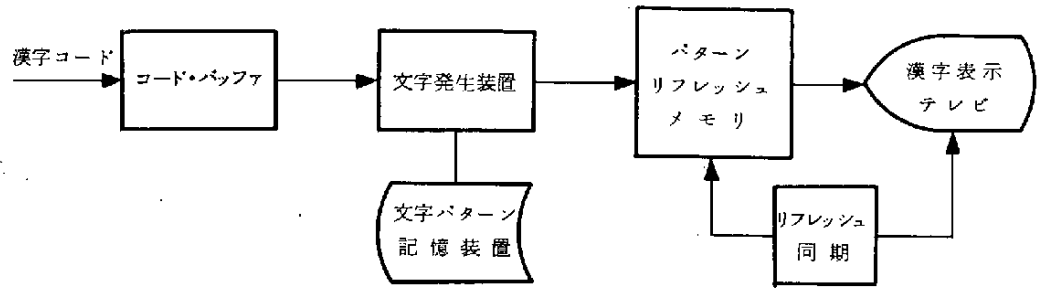


図４－１３　漢字ディスプレイ構成

コード・バッファは漢字ディスプレイに表示する文字分の漢字コードを受けるためのバッファ・メモリである。最大２１６文字を表示するディスプレイであるから、１文字２バイトの漢字コードで、バッファ・メモリ・サイズは４３２バイト必要である。実際にはこの他に画面制御コードなども挿入されて入力するので多少増分される。このバッファ・メモリは制御プロセッサの入出力バッファとしてとられる。

文字発生装置・文字パターン記憶装置については前節でその構成・規模・機能を検討してあるので、それを導入する。文字パターンは、約２０００字分をフロッピーディスクにもつ。

パターン・リフレッシュ・メモリは、画面の分解ドット数分もち、ここでは５１２(横)×３８４（縦）ドット構成にしているので、およそ２５Ｋバイトのメモリが必要である。メモリの機能は、リード・ライト・メモリでランダム・アクセスのできる記憶素子に限定される。メモリのアクセス速度は、テレビ画面への表示にリフレッシュ機能が必要であり、そのリフレッシュ・サイクルが一般に１／３０秒といわれることから、それ以上の速度が当然要求される。

以上のようなメモリ構成の他に、日本語端末の漢字ディスプレイ装置としてもつ、いわゆる画面操作機能や編集機能によってメモリ容量が増加する。メモリ容量に影響するもっとも大きな機能は、画面の保存機能である。画面に表示した情報をどれくらい保存しておいて再表示可能とするか、またその保存情報をコードバッファにもたせるか、パターン・リフレッシュ・メモリにもたせるか、あるいは外部メモリ（フロッピー・ディスク）に退避しておくか、などの方法があり、その選択によってメモリ容量や再表示速度などが異なってくる。

次に、画面操作機能・編集制御機能を検討する。標準的なディスプレイが持つ機能は別表４－４に示す。日本語端末が持たなければならない操作・制御機能は、日本語端末を利用するアプリケーションによって、あるいは日本語端末に採用する入力方式によって、規模・機能の範囲がちがってくる。後者の入力方式で採用するキーボードは、英数カナ・キーボードであり、ディスプレイ制御キーボードを付属しなければ機能を発揮することはできないが、そこに割当てる最小機能としては、カー

ソル移動のためのいくつかのキーであり、挿入・削除・復改・送信・ハード・コピーの各キーが必要であろう。

またディスプレイの表示機能として効果的なものは、漢字モードと英数字カナ・モードとの扱いを混在できることである。漢字モードの表示は２４×２４ドットのフル・パターンで表示し、英数字・カナ・モードの場合は半角文字として約１０×１８ドット構成で表示できるとよい。また画面を入力モニタ部と応答表示部とにわけて使うことが有効である。とくに英数字カナ・キーボードを使用する場合には、入力は英数字・カナのみであり、これを漢字パターンをアクセスしてモニタ表示するのは効率が悪い。

表示機能としてさらに望まれる機能は、ケイ線表示機能である。漢字ディスプレイをドットに分解して機能するのであるから、簡易な図形表示も可能であり、アプリケーション端末としても簡単な作表・枠組に図形表示機能を要求されることもあろう。

このような各種の表示機能は、ソフトウェア規模によっては持つことができないこともあり、プロセッサの機能と文字発生機能などの総合的な実験をふまえて決定することになろう。

表４－４　ディスプレイの制御キー

全画面消去
行消去
文字削除
文字挿入
行挿入
復帰改行
カーソル移動
- １字前進（→）
- １字後退（←）
- １行前進（↓）
- １行後退（↑）
- ホーム・カーソル（↖）
- ニュー・ライン（↙）
- タブ前進
- タブ後退

タブ・セット
タブ・クリア
ヘッディング・マーク
シフト
送信
ハード・コピー
前画面表示
前行表示

(2) 出力機能

日本語端末の出力機能は、漢字ディスプレイ画面のハードコピー装置と、日本語情報の印字機能としての漢字プリンタとに大別できる。

日本語端末の基本機能は、漢字かな混りの日本語情報を提供サービスすることであったが、その中心となる構成は漢字ディスプレイであり、前述の表示機能だけでも十分である。しかしながら、コンピュータと対話型で得られた情報をソフトコピーにとどめて満足することは少なく、対話途中でどうしても必要な情報はその場でコピーをとっておきたいことが多分にある。そのようなときのハードコピー機能はどの程度であればよいか。

またディスプレイ画面のハードコピーばかりでなく、もう少し多量の出力情報を得るようなアプリケーションでは、ディスプレイ表示とハードコピーを繰り返し操作するのでは処理時間が長くなって、オンライン日本語端末の特徴が生かされない恐れもある。ホスト・コンピュータへの問合せにおいて、データベースからのアクセス情報が大量な場合、一括出力をセンターに依頼して、オンライン出力はアクセス結果のメッセージに留めるなどの対処方法も考えなければならないが、情報内容によって、あるいは即時出力が要求されるアプリケーションでは遠隔端末にハードコピー機能を備える必要がある。そのときの出力・印字機能をどう設定すればよいであろうか。

日本語端末の基本機能に価格面と処理（出力・印字）機能を含めて全体的機能・構成を設計する必要がある。漢字ディスプレイ・キーボードを中心にした日本語端末にハードコピー機能を容易に装備できるのは、ハードコピー装置を漢字ディスプレイのパターン・メモリに直結するのが最も簡単である。漢字ディスプレイとして適用したテレビ画面の構成に、テレビ画面の分解能に合わせたパターン・リフレッシュ・メモリが装着されており、メモリのビットが画面の画素（点素）と対応して、ドット・マトリックス・ディスプレイを構成している。そのドット・マトリックス構成は前述のように５１２（横）×３８４（縦）ドットを想定して設計されている。このドット構成にリード・ライト・メモリが適用されて、その走査・配列順は、図４－１４のようになっている。ここに文字パターンが記憶されて、テレビ画面に表示される。ハード・コピー装置の

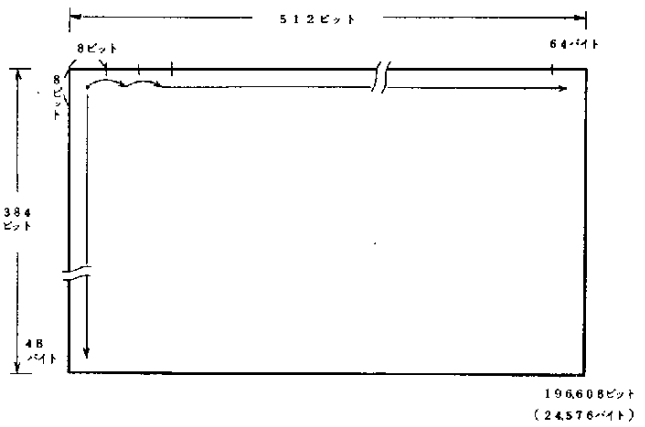


図４－１４　パターン・リフレッシュ・メモリ構成

機能は、このドット・マトリックスのメモリを図中、左上から右へ、上から下へと走査して右下までを走査していく。当然ハード・コピー装置もドット・マトリックス・プリンタ、ラインドット・プリンタ類に限られ、メモリ走査とドット印字走査が同一形態でなければコピーは難かしい。

次に日本語端末用の漢字プリンタの機能を検討する。日本語端末の文字発生方式は、ディジタル型のドット・マトリックス方式であることから、ここに適用する漢字プリンタも同方式の印字ができる機能を有する必要がある。それは日本語端末の出力機能を、漢字ディスプレイと漢字プリンタに装着するにあたって当然のことながら高価な文字発生装置を共用させることである。漢字プリンタへ出力させる構成は下図4－15のようである。

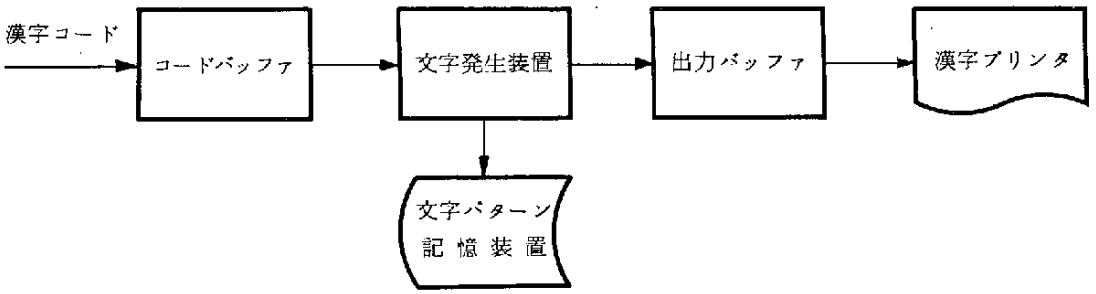


図4－15 漢字プリンタの構成図

コード・バッファ、文字発生装置および文字パターン記憶装置の機能は、漢字ディスプレイのパターン・リフレッシュ方式の構成で述べたことと同様である。相異点は漢字プリンタへ出力するための出力バッファと印字機構である。

漢字プリンタの出力方式はドット・マトリックス方式を採用するが、その印字方式がシリアルな印字機構を有するのか、ラインプリンタの印字機構を有するのかで、出力バッファの持ち方はちがってくる。シリアルなドット・プリンタでは、文字パターンの発生後、1字ごとに印字していくために、出力バッファは最小1字分でよい。ライン・ドット・プリンタでは、文字パターンの発生を1行分まとめてバッファに入れ、その1行分のバッファから、水平方向に1ビット行を走査しながら印字していく。文字単位・行単位の印字方向を図4－16に示す。

端末装置に使われるプリンタは漢字出力を問わず、シリアル・プリンタが多い。シリアル・プリンタは、ライン・ドット・プリンタよりも一般に速度は遅いが、印字機構が簡単で安価なものが多い。しかし、ファクシミリがコンピュータの出力機器に使われるようになれば、ライン・ドット・プリンタも数多く出現するにちがいない。

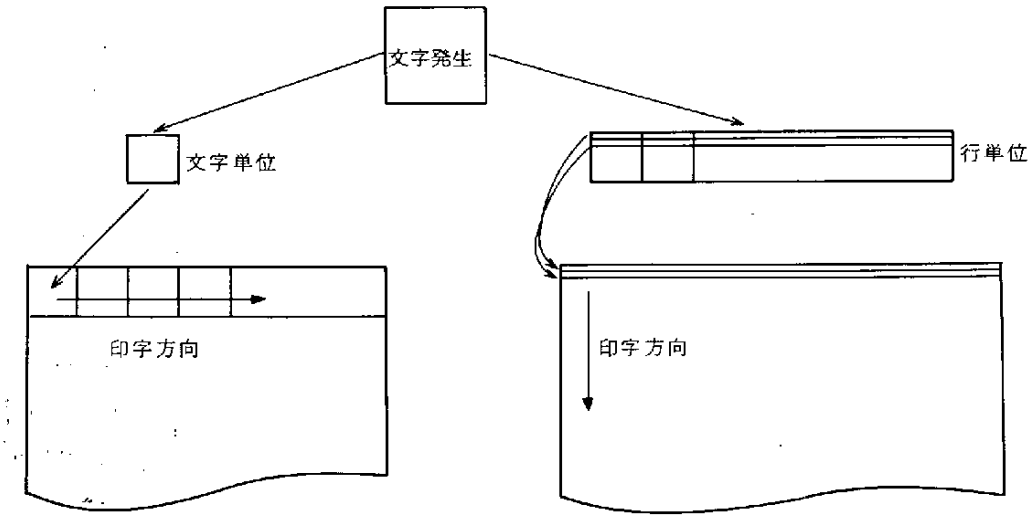


図 4 － 16　文字パターンの印字

日本語端末用の漢字プリンタとして要求されるのは、安価で小形であることを第1の条件に、その他運用にかかわる印刷用紙の低価格、事務所に設置するような場合の低騒音であることなどが要件としてあげられる。これらの要件に適用できる印字方式は、インクジェット方式をはじめに、感熱記録方式、静電記録方式、コピーのとれるワイヤインパクト方式などが有効であろう。いずれも日本語端末の文字発生方式に採用するドット・マトリックス方式に適合する印字方式である。とくにインクジェット方式とワイヤインパクト方式は普通紙に印字記録できることからも満足されるであろう。また低価格化がはかれるならば、乾式電子写真方式の高速性・普通紙印刷がバッチ処理をも含めて、日本語情報処理の出力装置として期待される。

一方、日本語情報の出力処理には、編集機能の有無が問題にされることがある。主にバッチ処理の出力制御装置に課せられる機能であって、日本語独特の編集規則が多数存在する。主なものだけでも、字詰・禁則・ルビ・ジャスティフィケーション・行頭・行末・中央揃え、分離禁止・数式合成などの棒組調整処理や文章全体の組版編集処理など様々である。こうした編集機能は出力装置のハードウェア機能によっておよそ制限をうけるが、ソフトウェア技術によるところも多分にある。

日本語端末としては、これらの編集機能は一般には不要で、ディスプレイ画面の編集機能にとどまることが多い。その他出力装置がもつ機能に、ページング機能がある。一般のラインプリンタは連続用紙を使ってページ単位に切り離すことができるが、漢字プリンタではロール紙が使われるケースが多くあり、この場合には自動カッテング機能が装備されていないと不便である。

以上(1)表示機能、(2)出力機能について、検討事項を述べてきた。ここで日本語端末の表示・出力機能と構成を再度考察してみると、表示装置と出力装置の文字発生は、1つの文字発生装置を共用

できることである。その構成を次図４－１７に示す。

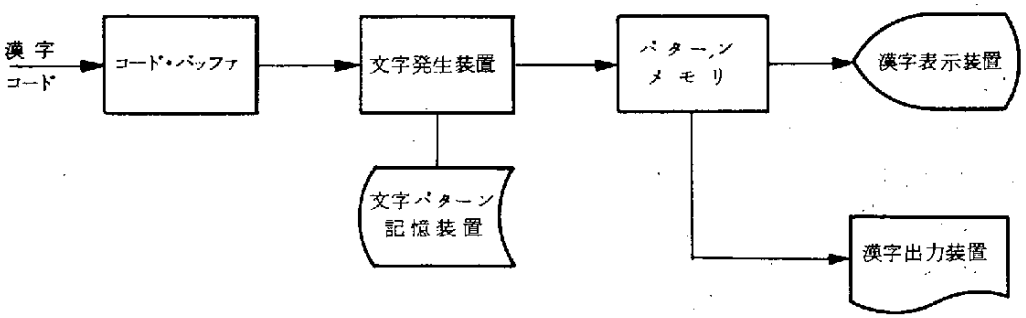


図４－１７　文字発生装置の共有図

こうした共用化は、日本語端末の表示・出力機能をバックアップし、文字発生装置の二重化による価格向上を抑えた、効率の良い日本語端末を構築することができる。

5. 日本語端末の課題

最近のコンピュータ技術、周辺、端末装置の発展はあらゆる分野において情報処理のコンピュータ化を急速に拡大してさた。そのため従来コンピュータ処理といえば演算処理を目的として利用されてきたが、適用分野の拡大によって一般事務処理へのコンピュータ化が進むにつれて、コンピュータの出力結果に対する不満と、日本人であるがゆえに、コンピュータから出力される文字に漢字かなまじりの日本語情報をという要望が強まってきた。とくに官民・業種を問わず、事務所にある文書はすべて日本語文で記述されていることはもちろんのこと、人事管理・顧客管理における姓名・住所その他もコンピュータ処理といえども本来は漢字かな混りの日本語文である。これまでコンピュータ化されても数値以外は手書きを加えたり、カナ文字やローマ字・コードなどに置きかえられたまま使用せざるをえなかった。

これらは従来の情報処理が処理面の効率追求にあったためである。この追求心が今日のコンピュータの進歩をもたらしてきたけれども、欧米と日本との比較では文字表現の問題が大きな相違である。欧米諸国で日常使用している文字は数十種で十分であり、そのコンピュータ化も容易で、文章・人名・住所・社名などいずれもコンピュータで扱われている数十種の文字で表現できる。コンピュータ用のキーボードも、欧米人は日常から欧文タイプライタを使いなれており、そのタイプライタのキーをそのまま入力キーや端末キーボードに使うことで、素人向けも専門オペレータ向けも区別なく適用可能である。これに対して我が国で使用されている文字種は数千字から数万字ともいわれ、欧米諸国の文字種に対し１００倍以上の多さである。しかも邦文（和文）タイプライタは一般への普及もなく印刷業務や事務処理業務において熟練した専任オペレータのみしか使用できない。そのため和文タイプライタのキー配置を、コンピュータ処理の入力キーボードにそのまま適用することはできない。現在の日本語情報処理では、和文タイプライタ方式も含めフルキーボード方式が主流を占めているが、その入力操作訓練に、およそ３ケ月以上必要といわれている。そして熟練したオペレータでも、入力速度は欧文タイプライタの３分の１以下である。この差は、欧米と日本との文字種の相違から当然のことであり今後も差をつめることは不可能であろう。

それでは日本のメーカ各社はいかにして日本語情報処理に対処していけばよいであろうか。上記の文字種のギャップは、日本語情報処理用の各装置の価格を高額にしている。とくに出力装置に至っては、数千万円規模のものが多く、普及をさまたげている大きな原因になっている。出力装置の高価格化の最も大きな要因は文字種の多さであり、この要因は出力装置の主要機構である文字発生器の規模を大型化している。ディジタル型にしてもアナログ型の文字発生方式にしても、出力のために文字パターン（フォント）を字種分記憶（記録）しなければならず、英数字用の文字発生にくらべ、これだけでも１００倍の文字種分が必要である。それに加えて、漢字のパターンは複雑で、字画、点素でとらえても漢字表現のためには多くのパターン情報が必要である。漢字を格子に通して点素（ドット）に分解しようとすれば、約１万字種を表現しようとすると最低２０×２０以上、標準的には２４×２４あるいは３０×３０、３２×３２といったドット構成が多い。２４×２４ドッ

トで５７６ビット、英数文字を５×７ドットで３５ビットとしても、記憶するメモリ容量は１００倍もちがう。ということは単にメモリ容量の比較をしてみても、５７６ビット×１００００字＝5,760,000ビット、３５ビット×６４字＝2,240ビット、となっておよそ2,500倍のちがいがある。文字発生器のメモリ・コストもこの差がそのまま現われることになるが、実際には、他の要素（字種を制限したり）があるため数倍から数十倍程度の差になっている。文字発生器のメモリ・コストは、そのまま漢字出力装置のコストに大きくえいきょうし、さらに日本語情報処理全体のコストにも及ぶことになってメーカ各社の苦心がここにある。ディジタル型の文字発生器による場合、ディジタル・メモリのコストが主要であり、その低廉化がはかれることが期待される。最近の半導体素子、ＬＳＩ技術の進歩によって、高密度化、大容量化がはかられつつあり、低廉化も進んでいる。また、周辺記憶装置もますます大容量化が進み、ビット当りの単価も次第に下がってきている。さらにプロセッサの機能向上と、ＬＳＩ化による低廉化も進んで、ディジタル型の文字発生器がもっとも有望な方式とみられている。

ディジタル型の文字発生器への期待は、さらに文字パターンの圧縮技術によって倍加する。現在のところ文字パターンの圧縮技術を優先させるか、ディジタル・メモリの低価格化を率先させるか判断に迷う段階である。実際に低価格をうたった漢字出力装置の例をみてもパターン圧縮によるものと、低価なメモリを採用したもの、とに大別される。そして両者の制御には、機能面からミニコンピュータを使用し、マルチステーション化によって低価格を前面に出す場合と、機能を多少落しても低価なマイクロコンピュータを導入して低価格を打ち出す場合がある。ここには、発展途上にあるマイクロコンピュータが、どこまで制御することができるか判断しにくい面がある。マイクロコンピュータも汎用コンピュータと見なすことができるか、あるいは現在多くの例にあるように、各種の機器・装置に組込まれて利用されるようにコントローラ用のプロセッサとしかないと判断するかである。ミニコンピュータ・メーカが主記憶をＬＳＩ化してマイクロコンピュータとするケースもあればマイクロコンピュータの機能を向上させてミニコンピュータを代用するという双方から近づきつつある。しかし現段階では、汎用コンピュータというよりも、ＯＥＭ用の制御プロセッサといった方が適当のようである。とはいえ、マイクロコンピュータの出現は、日本語情報処理はもとより、他の周辺、端末機器、各種アプリケーション・システムにおいても、注目されることである。

低廉化を進めるための要因は上記のようなことであろうが、漢字入出力機器の各メーカからは、現在のところ低価格製品は数少ない。漢字出力装置の低価格製品は、低速なシリアル・プリンタ、低品質文字のプリンタ、文字種の少ないプリンタと欠点ばかりが目につき、漢字出力機能として基本的・標準的機能を装備したプリンタの出現はない。日本語端末用にも簡易化をはかって精度を落すばかりでは需要があるかどうか疑問点が残ってしまう。メーカに対しては、需要が大きければ大量生産がはかれて低額になるというが、日本語情報処理の現在の普及状況からは、早く低価格製品を出して需要を喚起する必要があるようだ。

最近、ミニコンピュータによるオンライン漢字処理システムを発売したＤＥＣ社（学研と提携）や、ＴＳＳ端末に接続できるマイクロコンピュータ内蔵の漢字ディスプレイ端末を発表した米グラフテッ

ク社（アルプス電気と提携）など、低価格を前面に出したことは、ユーザに大きな反響を呼んでいるようだ。しかし残念なことは、これらのメーカが国産メーカでないことである。日本語情報処理は日本で生れて日本で育ってほしいと願われ、その技術開発も国産メーカが先導して、これに外資系メーカが追随するのが本来の姿ではなかろうか。旧来、新聞社のシステムも日本ＩＢＭが最初に開発している。情報処理システムの中でも、わが国独特な情報処理、それが日本語情報処理であり、この分野こそ国産メーカがもっと力を入れてほしいと願われる。

低価格の漢字入出力装置が出現し、実用化されるようになれば、コンピュータ周辺端末が漢字化され、オフイス・コンピュータも事務室内で漢字かなまじり文を得ることができるようになる。その結果、コンピュータ処理というイメージも、英数字やカナ文字しか入出力できないために異和感や機械による冷たさから開放されて、社会・家庭の中にもとけこめる柔らかさが得られることだろう。

従来の英数字処理は、本来日本語の処理の一部であり、日本語の文書の中には、漢字の他に、仮名文字、数字、そして英字その他の外国文字も含まれることもある。その一部をコンピュータ処理してきたが、これが日本語情報処理によって、果ては広く世界語の処理を可能とするであろう。

次に低価格化の課題とともに、日本語端末に重要なことは、ソフトウェアの処理機能の最適化をはかる点である。日本語端末の構成と処理機能は強い相関にあり、構成要素の入出力機器の選択はソフトウェア機能の増減に大きくえいきょうする。簡易な入力方式の採用は、内部でのコード変換処理を増加させ、出力機能を多様化すればソフトウェアも増加するように、相反するものと同調するものがある。また端末自身ばかりでなく、ホスト（センタ）側の処理機能との関連も強く、端末のインテリジェント化とホストの集中処理が相関する。一般にホスト側にアプリケーション・システムがあってデータベースを有し、端末側には端末の入出力機能・制御機能を持つことが多い。このセンタと端末の処理機能は、合せもってこそ日本語情報処理が可能であり、一方のみが漢字処理ができても実動することはできない。たとえば、日本語のデータベースを持つホストの処理は、データベースのアクセスを端末からの問合せに対して行ない、その結果を端末へ送る、といった機能をもって、一方の端末では、送られてきた情報を利用者に出力提供する機能をもたなければならない。その場合、入力する情報は問合せ情報で少量であるが、応答情報はそれよりも多量なことが一般的である。その出力情報が日本語情報であった場合、漢字コード情報なのか漢字パターン情報であるか、あるいはまた英数・カナ・コードであるかによってそれぞれの機能が大きくちがってくる。それは端末の機能によるところが大きく、文字発生機能、英数カナ・コードからの変換機能、などが端末のプロセッサに必要である。そしてこれはまた、伝送速度の問題もあり、一般には伝送するには簡易なコードで行ない、端末で文字発生（コードからパターンへの変換）するようでないと実用的でない。カナ・コード伝送では端末にソフトウェア負担が多くなりこれも実用的でない。

このようにセンタと端末との間に、処理機能を分担させることは、多数端末を接続するセンタのコンピュータ負荷を平均化するという効果がある。

また、端末の用途も、アプリケーションによって種々であり、センタと端末との機能分担も、適用分野ごとに設計する必要もある。

その場合は、日本語端末の入出力構成も機能を十分発揮できるように構築しておかなければその効果を活用することはできない。

以上述べてきたように、日本語情報処理全般からみた課題と、日本語端末としての課題とがあるが、早くその解決をはかって日本語のコンピュータ処理を可能とし、普及してほしいものである。そのためにも、大手メーカ各社が率先して低価格製品を出して需要を高め、また一方では官公庁機関で日本語情報処理の導入に先べんをつけ、一般企業への導火線となるように各種アプリケーションの漢字化を推進されることを期待する。

公的研究機関、メーカ系研究機関においても、これまで以上に漢字入出力装置の研究開発およびソフトウェア開発に力を注がれることが期待される。

当時における日本語端末の研究開発も、日本語情報処理の普及の一端を担うべく、低価格化、簡易化を目指して、入出力処理機能の最適化検討を進めていく計画である。そして日本語端末のモデル構築を行ない、ネットワーク端末として日本語アプリケーションの事例研究を行なっていく予定である。なおこの日本語端末の構成ハードウェア機器は、すでにメーカで開発されている入出力機器を利用していくことになろう。

〔参考文献・資料〕

（日本語全般）

(1) 漢字処理とハードウェア

松田功、石井淳（富士通）, FUJITSU, '76, Vol 27, №3, PP.53—63

(2) わが国における周辺端末機器の発展〔日本編〕

事務と経営 3月臨時増刊号、1977, Vol.29, №353, PP.2—64

(3) 漢字情報処理の現状と展望（明るい見通し ソフト開発 コンピュータと漢字の対話）

菅井和郎（JIPDEC）、印刷界、'776、№283, PP.40—47

(4) 日本語情報処理の調査分析

工技院利用研報告書、昭和52年1月、P.155

(5) 情報交換用漢字符号系 JIS C 6226—1978

日本工業標準調査会、日本規格協会発行

(6) 日本語情報処理の技術動向調査報告書

JIPDEC、47—R—001.'783、P.282

(7) 文書情報処理に関する調査研究

JIPDEC、48—S—004. '743、P.206

(8) 文書情報処理に関する研究開発

JIPDEC、49—S—006、'753、P.151

(9) 日本語情報処理システムの研究開発

JIPDEC、50—S—001、'763、P.300

(10) 日本語情報処理システムの研究開発

JIPDEC、51—S—001、'773、P.321

（入出力関係）

(11) 日本語・インプットの実際

高橋達郎（JICST）、事務と経営、'716、PP.97—101

(12) 日本語・インプットの実際（続）

高橋達郎（JICST）、事務と経営、'717、PP.43—48

(13) タッチ打法による漢字入力

川上晃、川上義（ラインプット）、情報処理、'74、Vol.15、№11、PP.863—867

(14) ソクタイプによる漢字処理法

佐伯功介（裁判所書記官研修所講師）、情報処理、Vol.11、№81、Aug.1970、PP.491—494

(15) 2000字種を100字／秒で読む印刷漢字OCRの開発
日経エレクトロニクス、1977.10.31、PP.102—128

(16) パターン合成による漢字入出力処理
長谷川実郎（日本電気）、情報処理、'75.9、PP.808—817、Vol.16、№9

(17) ユーザのための漢字入出力装置の選定
草野玄三（沖電気）、事務管理、Vol.16,№8,Aug.'77, PP.25—35

(18) 日本語・アウトプットの実際（上）
高橋達郎（JICST）、事務と経営、'71.9、PP.59—64

(19) 日本語・アウトプットの実際（下）
高橋達郎（JICST）、事務と経営、'71.10、PP.97—102

(20) 漢字のドット字形
成田勝（富士通）、FUJITSU、Vol.27、№3、PP.129—136

(21) 部分パターンによる漢字の合成
坂井利之、長尾真、寺井秀一（京大）、情報処理、'69、Sept、Vol.10、№5、PP.285—293

(22) 漢字モノスコープ管C—3M06
大森浩志、森本義春、安積輝（富士通）、富本斉（富士通テン）、FUJITSU、Vol.27
№3（1976）、PP.531—538

(23) 高速漢字プリンタシステムについて
黒崎悦明（沖電気）、情報処理、'75.9、PP.802—807、Vol.16、№9

(24) 電々公社が開発急ぐ漢字情報処理システム用プリンター
編集部、コンピュートピア、Vol.11、№132、Sep、'77、PP.75—79

(25) 放電破壊式プリンタの特徴とその応用
上野正明（日本ハムリン）、電子科学、1977.3、PP.63—67

(26) FACOM 6570 漢字ディスプレイ
萩原洋一、嶋村敏雄、堅月忠夫、大津信一、柿沼梯司（富士通）、木下幾夫（F.H）、
FUJITSU、Vol.26、№1（1975）、PP.119—130

(27) 高品質を目的とした漢字情報処理機器
柴田信之、畑中靖通、小畑甫（三菱電機）、情報処理、'75.12、PP.1084—1091、
Vol.16、№12

(28) 漢字システムの入出力装置、ユーザ一覧
小森一男（EDP評論家）、コンピュートピア、Vol.11、№132、Sep.'77、PP.69—72

(29) メーカ別漢字入出力装置一覧表
事務管理、Vol.16、№8、Aug.'77、PP.48—51

(30) 最新のプリンタ技術
データ通信、1977、Oct、PP.30—67

(31) 各種漢字パターン・メモリの実例とその得失

小谷進太郎、大迫昭和、安孫子一松（沖電気）、日経エレクトロニクス、1978.2 20 PP.126−148

(32) PROMによる漢字発生器

内藤祥雄、南谷崇（日本電気）、電子科学、1976、5月号、PP.37−43

(33) 文字放送

沼口安隆（NHK）、電子通信学会誌、5／'77、Vol.60、№5、PP.526−528

(34) 仮名漢字変換による文字放送用の文字編集装置を開発

日経エレクトロニクス、'76 823

(35) 情報処理における機械技術

中川三男、金子礼三（武蔵野通研）、電子通信学会誌、7／'77、Vol.60、№7、PP.802−810

(36) オンライン和文処理システムNOWS

真子ユリ子、内田俊一（電総研）、須賀英興（三菱電機）、島田幹郎、右上正之（慶大）、情報処理学会マンマシンシステム研究会資料、1976.1.20，PP.1−10

(37) オンライン和文端末のための表示装置

真子ユリ子、内田俊一（電総研）、須賀英興（三菱）、電子通信学会論文誌、'76.6 Vol.J59−D、№6.PP.383−390

(38) 画像端末制御おけるミニコンの応用

釜江尚彦、谷口道夫（武蔵野通研）、情報処理、Apr.1974、Vol.15、№4、PP.310−315

(39) 簡易形遠隔図形表示方式

吉田春彦、栗原定見、中根一成、伊藤博（武蔵野通研）、情報処理、Vol.17、№9、Sep.1976、PP.796−804

(40) 印刷とコンピュータ

近藤厚美（千葉大）、データ通信、'71、Feb. PP.29−33

（アプリケーション関係）

(41) 明治生命の漢字情報処理システム

田附俊雄（明治生命）、事務と経営、1978.1、PP.60−63

(42) 明治生命の漢字名寄せ索引システム

奥山貞夫、小川孝一郎（明治生命）、コンピュートピア、Vol.11、№132、Sep.'77 PP.41−45

(43) 証券代行業務における漢字プリンターの利用

久保田博（東洋信託）、FUJITSU、'76、Vol.27、№3、PP.5−12

(44) 証券代行業務における漢字情報処理システム

長屋康生（東洋信託）、地方自治コンピュータ、'77．6、Vol.7、№6、PP．28−35

(45) 漢字レセプトシステムにおける病名管理

野村優、塩飽春典、山部裕、武部伸、三宅成幸（倉敷中央病院）、金井一成、小林啓二（日電）、昭和52年度情報処理学会第18回全国大会論文集、PP．743−744

(46) 太田市（群馬県）における漢字情報処理について

渡辺耕造（太田市）、地方自治コンピュータ、'77．6、Vol.7、№6、PP．19−27

(47) 地方自治体（群馬県太田市）における住民情報漢字処理システム

渡辺耕造（太田市役所）、コンピュートピア、Vol.11、№132、Sep.'77,PP．47−51

(48) 広島大学原爆放射能医学研究所に於ける「大容量映像情報検索システム」について

栗原登、務中昌己、上岡洋史（広大）、橋本昌幸、黒沢兵夫（日電）、昭和52年度情報処理学会第18回全国大会論文集、PP．677−678

(49) 広島大学原爆放射能医学研究所の情報検索システム

務中昌己、栗原登（広島大学）、橋本昌幸、高沢良一（日電）、情報管理、June1976、Vol.19、№3、PP．195−204

(50) サンケイ新聞社の漢字処理システム

米村一介（サンケイ）、FUJITSU、'76、Vol.27、№3、PP．22−38

(51) ミニコンネットワーク新聞電算写植システム−サンケイ新聞社におけるトータルシステムSUCCESSの開発と実用化−

米村一介（サンケイ新聞社）、情報管理、Dec.1976、Vol.19、№9、PP．718−727

（関連技術）

(52) 自然言語処理

雨宮真人（武・通研）、電子通信学会誌、5／'77、Vol.60、№5、PP．533−535

(53) 言語情報処理の過去・現在・将来

長尾真（京大）、情報処理、Feb．1978、Vol.19、№2、PP．106−112

(54) 機械翻訳序説〔1〕歴史的背景とその現状

原田哲夫（日大）、事務と経営、'71．5、PP．110−115

(55) テレビやテレビ電話による日本語の表示

麻生哲（日立）、"情報処理における日本語（漢字）の取扱い"〔日本経営科学協会主催セミナー〕、PP.101−115

(56) 期待されるCOMシステム／機器

水沢元典（フジコーポレーション）、事務と経営、'78．3、PP．40−43

(57) MISとCOMシステム

立花正毅（吉沢ビジネス・マシンズ）、データ通信、'71．Apr、PP．67−70

(58)　画像処理技術特集

電子通信学会誌、11／'76、Vol.59、№11、PP.1159－1319

(59)　特集・ファクシミリを検討する

季刊輸送展望、'77 winter、PP.15－79

昭和５３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（代表）

印刷所　株式会社　三　州　社
東京都港区芝大門１－１－２１
ＴＥＬ　（４３３）１４８１

52−S001