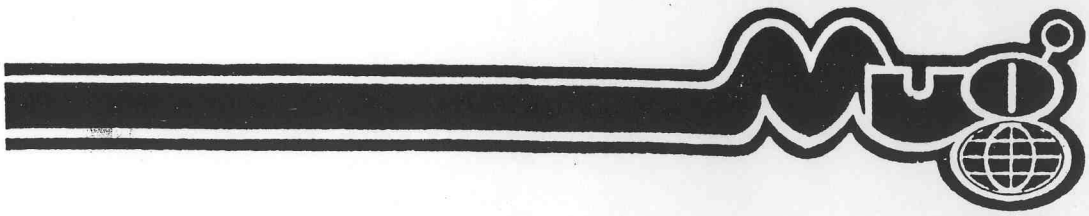
日本エム・テクノロジィ学会事務局
〒259-1193　神奈川県伊勢原市望星台
東海大学医学部・基礎医学系
大櫛陽一
Tel: 0463-93-1121 ext.2140
Fax: 0463-96-4301
Email: youichi@keyaki.cc.u-tokai.ac.jp

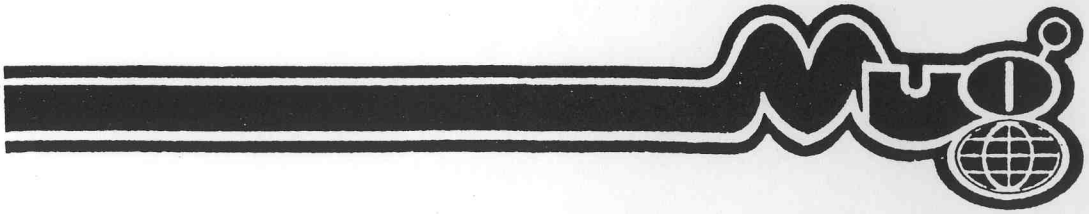


第12回日本MUG学術大会

論　文　集

1985

福　　井



第12回日本MUG学術大会

論　文　集

1985

福　　井

序

第１２回日本ＭＵＧ学術大会は、「考えるマンプス」というメインテーマのもとに、昭和６０年８月２４日～２６日に、福井県芦原研修会館で開催されましたが、多数の方々が御参加下さり、盛大のうちに終了することができました。すばらしい演題を多数ご応募いただいたこと、プログラム作成に、大会の運営に、多大の御指導、御援助をいただいたこと、そして、はるばる福井までお越し下さいましたことをここに厚くお礼申し上げます。大会の集大成として、論文集を発行いたしましたので御覧下さい。

マンプスは、医療の場から生まれた言語でありますが、使いやすさが好まれ、医療以外の分野にも広く用いられています。マンプスを搭載する国産のメーカーも出現し、今や、市販のパソコンでもマンプスが動くようになりました。かくして、マンプス・ユーザーは一挙に拡大されました。最近では、他言語との連係とか、マイコンのネットワークも可能になりつつあるようです。コンピュータを利用することは非常に便利なことではありますが、人間に与える影響も大きいものです。近年にみられる急激な新機種の出現は、一つ間違えば、人間生活にとってマイナスになる危険性も含んでいます。私たちは、積極的に経験を話し合い、知識の吸収、交換をし、どんどんアイデアを出し、プログラムを開発し、多くのユーザーが利用できるように、開発されたプログラムの流通をはかって行きたいと思います。この論文集が、情報処理の仕事に携わる人々の、また、コンピュータを利用して新たな仕事を始めようとされる人々の一助となることを期待しますとともに、今後共、なお一層の御指導、御鞭撻をお願い申し上げます。

昭和６０年９月

第１２回日本ＭＵＧ学術大会長

山　本　　和　子

第12回日本MUG学術大会

論　文　集

目　　次

招待講演

「知識工学と医療への応用」・・・・・・・・・・・・・・・・・・・・・・・・1
開原成允（東大・医・中央医療情報部）

特別講演

1　マンプスによる輸出貨物における書類作成と情報検索システムの
適用例－－ネットワーク化への対応・・・・・・・・・・・・・・・・3
服部泰夫（日本通運国際輸送事業部）

2　Ｆｒａｍｅ　Ｗｏｒｋ（システム開発支援システム）・・・・・・・・7
藤江　昭（日本ＳＭＳ・システム開発部）

3　Ｐｒｏｌｏｇとは何ぞや・・・・・・・・・・・・・・・・・・・・・11
井手真理子（日本ＤＥＣ）

4　ＭＵＭＰＳのＰｒｏｌｏｇシンタックスの包含装備・・・・・・・・17
内田達弘（マンプスシステム研究所・名城大学）

5　ＭＳＭ－Ｗｈｅｒｅ　Ｔｏ　Ｆｒｏｍ・・・・・・・・・・・・・・33
David J.Marcus Ph.D(Micronetic Design Corporation)

6　新時代のネットワークを迎える漢字ＤＳＭ－１１・・・・・・・・・39
小林泰造（日本ＤＥＣ）

一　般　講　演

Ｉ－Ⅰ：Implementation Ⅰ

1　Ｍ／ＶＸについて・・・・・・・・・・・・・・・・・・・・・・・・４９
　　吉田欣弥（三井造船システム）
2　コンパイラ型ＭＵＭＰＳを用いたシステムの導入例紹介・・・・・５１
　　石丸径美（三井造船システム）
3　Ｕ－ＭＵＭＰＳの機能強化・・・・・・・・・・・・・・・・・・５３
　　小林　勝、久江　正、上戸　隆、阪倉　明（住友電気・ＭＥ）

Ｉ－Ⅱ：Implementation Ⅱ

1　マルチウインドー機能を持った日本語ＭＵＭＰＳ・・・・・・・・５７
　　鈴木利明（アレフコンピュータ）
2　ＭＵＭＰＳ下で実現したニューメディアシステムの紹介・・・・・５９
　　山口光大、柿崎賢一（高岳製作所）
3　（株）高岳製作所・タイムリーシステム・・・・・・・・・・・・６５
　　重松郁也、佐々木一郎（高岳製作所）

Ｓ：システムデザイン

1　スモールコンセプト（第二報）・・・・・・・・・・・・・・・・・７１
　　田久浩志、馬場謙介*
　　（産業医大　振動研究室，* 第二病理学教室）
2　時系列システムにおける保守時の診断仕様書・・・・・・・・・・７５
　　今泉幸雄（日本アップジョン）

ＭＤ：医療へのアプリケーション

1　汎用性を重視した臨床検査コンピューターシステムの設計・導入
　について－－第三報　特に検査前処理業務について－－・・・・・９１
　　長原三輝雄、長谷川俊雄、黒田満彦（福井医大検査部）
2　福井医大病院における栄養管理システムについて・・・・・・・・９７
　　本間富士子、西秋人、関山明彦、佐藤裕保（福井医大医事課）
3　ＭＵＭＰＳの健診支援への人工知能的応用の検討・・・・・・・１０１
　　林恭平、六反奈利子、森田益次、山口希、青池晟、川井啓市
　　（京都府立医大公衆衛生）

MC：マイコン・アプリケーション

1　マイクロマンプスによる眼科救急外来患者の統計的観察処理・・１０５
　　木村一元、西村雅晴*、小暮文雄**、関　亮**
　　（独協医大　総研ＭＥ室，* 公衆衛生学，**眼科）
2　マイクロコンピューターによる未熟児データーベースの作製・・１１３
　　土屋喬義、田中吾朗*、木村一元**、本間　道
　　（独協医大　第一小児科，* 第二小児科，**総研ＭＥ室）
3　入院台帳プログラム・パッケージ・・・・・・・・・・・・・・・１１９
　　林寺　忠、和田行文、西角　淳、上坂邦夫
　　（国立京都病院小児科）

D：デモンストレーション

1　ＳＥＴ　ＧＮＯＳＩＳ　＝　ＭＵＭＰＳ　＋　Ｐｒｏｌｏｇ・・１４５
　　Donald A. Smith, William A. Ackerman, 若井一朗
　　（マンプスシステム研究所）
2　ＳＰ－ＭＵＭＰＳ・・・・・・・・・・・・・・・・・・・・・・１５５
　　小林勝、久江正、米田研、阪倉明（住友電気・ＭＥ）
3　ＮＥＣ９８０１とＳＨＡＲＰ　Ink Jet プリンターをつないで
　　カラーグラフィックスをハードコピーする・・・・・・・・・・１５９
　　林寺　忠、和田行文、西角　淳、上坂邦夫（国立京都病院小児科）
4　研究用患者データ管理システムのマイクロマンプスへの変換・・１６１
　　本多正幸、里村洋一（千葉大）、北村嘉章（神戸大）

W：ワークショップ「ＤＳＭ－Ｖ３をめぐる諸問題」

1　ＤＳＭ－Ｖ３による業務スピード向上評価・・・・・・・・・・１６５
　　大槻陽一、古林栄次郎、野口　弘、寺村昌文（羽曳野病院）
2　ＤＳＭ－Ｖ２とＶ３のスピード比較・・・・・・・・・・・・・１７５
　　山本和子、須藤正克（福井医大情報処理部）
3　ＤＳＭ－Ｖ３によるパーフォーマンスの向上の評価・・・・・・１８３
　　服部敏夫、上田清治、青木浩二、入江真行、三嶋博昭、
　　河村徹郎、野村　裕
　　（大阪成人病センター／大阪母子医療センター）

第12回日本MUG学術大会

論　文　集

招待講演　　第１２回日本マンプスユーザーズグループ学術大会（１９８５年８月）

知識工学と医療への応用

開原　成允

東京大学医学部附属病院　中央医療情報部
東京都文京区本郷７－３－１

「要約」　知識工学の考え方を実際のシステムを例にとりながら検討し、次にそれを医療に応用する場合の目的について考察した。

知識工学の医療への応用が盛んになり医療用のエキスパートシステムが多くつくられるようになっている。しかし、これらのシステムが１９６０年代から７０年代にかけてつくられたいわゆる「自動診断システム」とどのように違っているのかについては必ずしもはっきりしているわけではない。従って、本講演では、これまでに発表された内外のシステムをレヴューしながら、「知識工学」の考え方とその医療の中でもつ意義について考えて見たい。

知識工学を使ったシステムには大きく分けて二つの考え方があるように思われる。第１は、人間の経験的な知識を重視する考え方であり、第２はコンピュータの技術的側面を重視する考え方である。第１の考え方に立てば、医師の経験的な知識をコンピュータに入れたものはなんでも「知識工学を応用したシステム」ということになる。例えば、これまでに最も成功した診断論理の研究としては心電図のコンピュータによる解析がある。この中で使われている診断論理は医師の経験を論理化したものに他ならない。これは最近のいわゆる知識工学的なシステムとは異なっているのであろうか？

また、第２の立場に立てば例えば LISP または PROLOG を使ったプログラムならばすべて知識工学の応用ということになりかねない。PROLOG で書いたプログラムの中に知識工学とは全く関係のないものが存在することは論をまたない。

この両者の考え方はいずれも極端で、重要なことは、「知識」をコンピュータで扱う手法が発達し、人間の経験的知識までを含めたシステムがつくり得るようになったことである。知識工学の最大の貢献は、おそらく「知識」をこれを操作する部分から独立させ、「知識ベース」という考え方を普及させたことにあるのではないかと思う。医学の世界では、知識の内容は多彩であるから、その表現形式も多彩であってよいし、また推論のもつ意義もシステムによって異なっている。

第２の問題として、医療においてはかかるシステムを研究する目的についても考えておかなければならない。しかし、この医学的な目的にも実はさまざまなものがある点は注意を要する。

すなわち、第１に医学的な事実を探求するためにこのようなシステムを使う場合がある。たとえば、同じ統計的な手法を用いた研究でも、その目的は新しい疾患の要因を見付出すためのものであることもある。また、目的は診断論理の研究であっても更にその中にもさまざまなものがある。第１は人の思考過程を研究することが目的の研究がある。これは診断論理の研究というよりは「認知科学」の領域に属するものである。第２に方法論の研究を目的としたものがある。この時は医学的なデータは単に例として使うわれているにすぎない。また、第３に新しい診断の論理を研究するための場合もある。そして第４に実用的なコンサルテーションシステムをつくるための研究がある。

目的は何であってもよいが、研究をする人も、それを評価する人もその目的をはっきり定めた上でそれを行なわなければならない。方法論を目的とした研究に対して応用範囲が狭いから役にたたないと批判することは当を得ていないし、また実用を目的とした研究に対して、診断の論理が単純すぎると批判することも当たらない。今後の研究はこれらの点を明確にした上で行うべきであろう。

特別講演 1　　第１２回日本マンプスユーザーズグループ学術大会（１９８５年８月）

マンプスによる「輸出貨物の書類作成と情報検索システム」適用例
ーーネットワーク化への対応ーー

服部　泰夫

日本通運株式会社　国際輸送事業部　管理課
東京都千代田区外神田 3-12-9 日通ビル　　℡　03-253-1111 内 2807

「要　約」
医療用に開発されたマンプスを一般ビジネス分野に利用し成功した。
業務の特異性から機械化困難と言われていた分野の、書類作成や情報検索であった。
現在では、このシステムと COBOLを中心とした当社メインフレームとのリンクに力を入れており、来春には海外ネットワークも稼働予定である。
また、日本語処理もマイクロマンプス利用で着実に進んでいる。

はじめに

日本通運株式会社では、東京、横浜、名古屋、大阪、神戸の各海運貨物取り扱い店所に於いて、57年 4月から順次、輸出船積システムを稼働させております。
機械化以前にはペーパージャングルと言われるほどの業務の繁雑さゆえ「まず機械処理不可能であろう」と実務担当者から言われていた、通関書類作成が、ほぼ100%機械打ちしている現状を見るにつけ MUMPSの威力を感じざるを得ません。
さて、輸出船積業務は、営業、通関、倉庫部門の連携作業により進行しますが、通常この部門は地理的に分散しています。
これら各部門で発生するデータを、端末から即時入力することにより、リアルタイムで処理し、輸出船積に必要な船荷証券、輸出申告書、保税帳票を始めとするドキュメントに展開して行きます。
また、このシステムには、日本海事検定協会、および新日本検定協会との共同開発による、検量データシステムが組み込まれており、検量データを発生時点で、即時に入力することにより、船積書類に早期展開することができ、従来方式に比べて 2～3 日の船積時間の短縮になりました。
当初は、神戸の一上屋から始まったローカル・システムが、現在では国内主要港をネットワークで結ぶまでに爆発的に広がっております。　それにつれ、当社の中に於ける一部門である当事業部のシステムと、メインフレームである FACOMを中心としたシステムとのリンクが急速にクローズ・アップされてきており、独自の世界を持った MUMPSと、一般的な COBOLの世界を如何にリンクするかが課題になってきました。　この点に関しては VAXを通信制御用に導入することによってカバーしてきました。　さらに、米国日通（IBM機使用）を中心とした海外ネットワークへの接続も来春までには一部開通させる予定です。

また、 MUMPSの日本語処理もマイクロマンブスの利用により、海外に於ける引越貨物の通関事情のファイルなど、その用途が急速に拓けてきております。

1．ハードウェアーの概要

－－－－（国際輸送事業部　MUMPS 系統）－－－－

A．PDP11/44　　7 SETS

①東京国際輸送支店　　4 SETS

②横浜海運支店　　1 SET

③神戸海運支店　　2 SETS

B．VAX11/750　　1 SET

①東京国際輸送支店

C．NEC PC9801E　　19 SETS

①東京国際輸送支店　　13 SETS

②横浜海運支店　　2 SETS

③海外プラント輸送支店　　1 SET

④名古屋港支店　　1 SET

⑤神戸海運支店　　1 SET

⑥事業部自体　　1 SET

－－－－（国際輸送事業部　MUMPS 系統以外）－－－－

D．IBM S/36,S/34

①東京国際輸送支店　　1 SET　(S/34)

②神戸海運支店　　1 SET　(S/34)

③横浜海運支店　　2 SETS (S/36)

E．FACOM V830,9450

①東京国際輸送支店　　1 SET　(V830)

②東京国際輸送支店　　2 SETS (9450)

③横浜海運支店　　1 SET　(9450)

－－－（事業部以外　メインフレーム）－－－

H．FACOM ／M380R,M360,M340 などメインフレーム

①本社・各地方支店　　8 SETS 以上

2．システム概要

A．営業部門

①SHIPPING INSTRUCTION（S/I)の登録

荷主より受け取ったS/I に基づいて、そのデータを入力する。

②搬入確認、および貨物引き当て

S/I で指示された貨物の搬入確認をディスプレイで行ない同時に引き当てをおこなう。

③船積予定連絡票の作成

通関、倉庫、両部門に対して社内指示書をアウトプットする。

④オーシャン・フレートの登録

あらゆるタイプのフレートに対応でき、またマスター化も可能となっている。

⑤BILL OF LADING(B/L), DOCK RECEIPT (D/R) のプリント

船社毎にフォームが異なり現在約 100社、数百フォームのプリントが可能である。

⑥請求書発行

請求項目、金額を手入力したり、また、荷主の様式に合わせて発行も可能である。

⑦その他

記憶されているデータを利用し、各種統計表を作成する。

Ｂ．倉庫部門

①入庫登録

貨物が入庫すると、送り状からマーク等のデータをインプット。

②船積マスター・ファイルの作成

現品引き当て後作成する。以後は船積み単位での貨物データの参照が可能となる。

③出庫指示書の作成

引き当てられたデータを基に、出庫指示書を作成する。

④取り扱い、改品登録

取り扱い、改品があったとき、入力をする。

⑤保税台帳の作成

貨物の状況を保税台帳としてプリントアウトし、税関への報告資料とする。

Ａ．通関部門

①輸出申告書 (E/D)の作成

引き当ての完了している貨物については輸出申告書をプリントする。

②通関台帳の作成

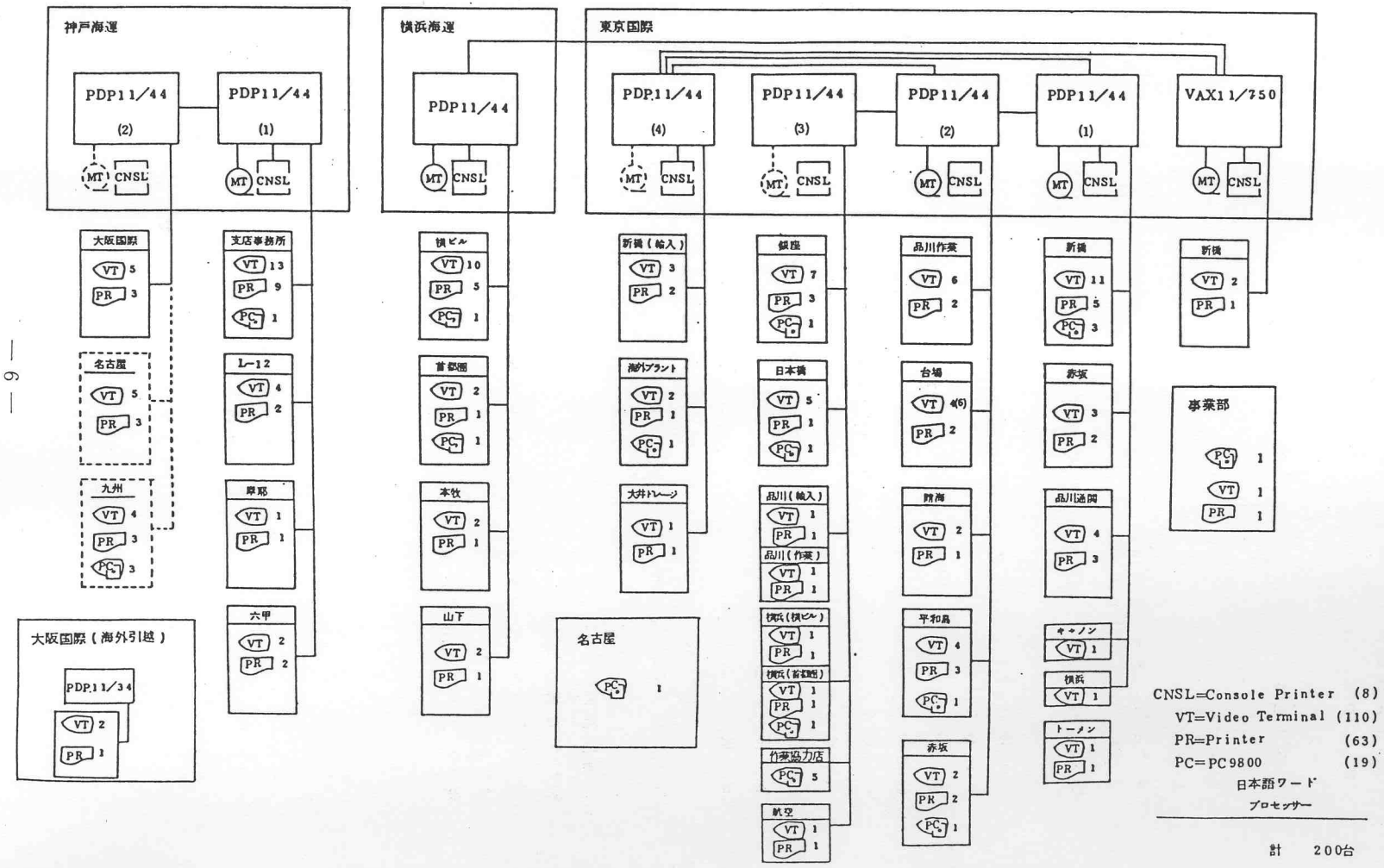
税関に提出する通関台帳をプリントする。

おわりに

この様なシステム開発成功以上に大事なことは、コンピュータを利用者に開放したことであります。　利用者が問題意識をもち、考え、作り、自分の責任で利用することが出来るのは、マンブスならではの事であります。　この結果、コンピュータ・アレルギーの解消はもとより、低コストのシステム開発ができました。

以　上

A機器（ハードウェアー）一覧表1（PDP/VAX/PC系統） S.60. 4. 1 現在



FRAMEWORK－－－アプリケーションプログラム開発支援システム

藤江　昭

日本エスエムエス（株）　システム開発部

東京都港区西麻布４－１２－２４　第３８興和ビル
Ｔｅｌ．０３－４９９－１７５１

MUMPSは優秀なプログラミング言語であるが、その特徴が、システムを維持する観点からは大きな障害となっている。MUMPSの良さを生かしつつ、大規模システムを長期間サポートしていくには、システムの開発環境を整える何らかの上位システムが必要である。FRAMEWORKはこのために作られたシステムであり、単なるユーティリティーの集合ではなく、第４世代言語のレベルまで機能が高められている。

１．緒言

MUMPSは次の様な多くの優れた特徴を持ったプログラミング言語である。

○容易に学ぶことができる高級言語。
○リアルタイム処理向きの高効率データベースマネージメントシステム。
○事前定義不要、可変長、可変構造、スパースアレイのデータベース論理構造。
○プログラム開発、バグフィックスが容易かつ、早い。
○高効率のマルチユーザ・インタラクティブシステム。

しかし、あまり語られない事であるが、この優れた特徴が、次の様な問題点を引き起こしているのも事実である。

○システムの開発変更が早く容易→大規模システムや、数多くのユーザーをサポートする時、個々の更記録が自動的に残らないため時間がたつ程メンテナンスが困難となる。
○コーデングと修正が早い→しばしば読みにくく、メンテナンスが困難なプログラムが作られる。ドュメンテーション性が弱い。

これらの問題点は、ユーザーもメーカーも痛感している事であり、管理の対象となるプログラム数が多くなればなる程重要なものとなる。またこれは時に、MUMPSをあきらめさせる原因にもなり得る。多くの大規模システムを長期に渡り維持していくには、MUMPSが本来持っている特徴だけでは明らかに不十分である。MUMPSの良さを生かしつつ上記の問題点を根本的に解決するには、MUMPSの上位に何らかの管理システムを置き、アプリケーション開発の環境を整える必要がある。

２．FRAMEWORKの設計基調

８年以上に渡って世界に病院情報システムを提供してきたSMS（Ｓｈａｒｅｄ　Ｍｅｄｉｃａｌ　Ｓｙｓｔｅｍｓ）は、同時にMUMPSユーザーでもあり、長年この問題と直面してきた。その結果、より高品質で安定したシステムを長期間サポートしていくために、新たにアプリケーションプログラム開発支援システムとして、FRAMEWORKを開発した。Ｖ１は１００人月以上の工数をかけて１９８３年にリリースされた。

FRAMEWORKは単なるユーティリティーの集まりではなく、アプリケーションプログラムの開発、及び実行に必要な総てを含んだＡｌｌ　ｉｎ　ｏｎｅ　システムとして設計された第４世代言語である。各種の仕様及び高級言語で記述された処理フローにもとずいて、MUMPSプログラムをオブジェクトコードとして出力するコンパイラーを備えており、FRAMEWORK自体もMUMPSで記述されている。FRAMEWORKの設計基調は次の様である。

○プログラミングをアートから工学へ。

プログラミング経験の自動蓄積を図るため、プログラムのモジュール化をシステム化すると同時に、各モジュールの照会機能、他のモジュールに対する影響の自動分析機能をもうける。

○詳細なコーディングからの解放

ＣＲＴからのデータ入力やＤＢへのアクセス時にしばしば用いられるプログラムの基本的な制御構造を自動生成することにより、プログラマーの生産性を良くすると同時に、プログラム自体の品質を向上させる。

○プログラム開発を、問題を最も良く理解している人の側に移動させる。

伝統的なシステム開発過程を見直し、仕様に関する情報の伝達漏れや無駄をはぶく。このため、アプリケーションプログラムの開発・保守が非プログラマーでも行えるようにし、ユーザーアナリストの働きを重視する（図１）。

○システムの変更を早くかつコントロールされた方法で行う。

長期間システムを保守するためには、あらゆるアプリケーションプログラムの変更は自動記録されなければならない。また、変更分を既存のシステムに適用する時も自動的に行われるべきである。

○セルフ・ドキュメント機能を充実させる。

ＤＢの論理構造、各変数の意味・属性などは、常にシステムから引き出されるべきものであり、システム変更と供にドキュメントも自動修正されなければならばい。

３．ＦＲＡＭＥＷＯＲＫの構成

ＦＲＡＭＥＷＯＲＫを形成している各コンポーネントの関係を、図２に示す。主なコンポーネントの機能は次のようである。

○データ・ディクショナリー：アプリケーションプログラム内で使用される総てのデータフィールドは、このＤＤに登録されなければならない。ＤＤには、各フィールドの名前、短縮名だけでなく、データ長、データタイプ、デフォールト値、バリデーションの方法、などのシステムに直接関係したものや、そのデータフィールドの目的、使い方、意味などのドキュメンテーションに関する情報も含んでいる。

○スクリーン・ビルダー：ユーザーはシステムとの対話をくり返すことにより、アプリケーションプログラムで使用するスクリーンのフォーマットや機能を定義することができる。また、同時に標準ファンクションキーのサポート機能を含めるかどうかを選択することができる。

○レポート用高級検索言語（Ｒ／ＱＬ）：データベースの論理構造を全く知らなくとも、各フィールドの名前のみで必要とするデータを取り出すことができる、英語形式の情報検索言語である。目的のフィールドまでのアクセスパスをシステムが自動的に見つけるため、Ｒ／ＱＬで書かれたプログラムは、データベースの物理構造から独立しているばかりでなく、論理構造の変更に対しても独立している。

○データベース・マネージャー：データベースの論理構造と物理構造を分離し、両者間のマッピング処理を行う。このため、従来よりＭＵＭＰＳのＤＢＭＳとしての致命的欠陥と言われていたプログラムのデータ独立の問題が完全に解決された。ユーザーは、データベースの論理構造を自由に定義することができ、物理構造は、技術部門のスタッフが、アクセス効率、ディスクスペースなどを考慮しながら、後で定義することができる。

○プロセス・ビルダー：非テクニカルスタッフが使用するための各種のモジュールを作ることができる。

○ファンクション・ビルダー：ファンクションは従来の概念ではメインプログラムに相当する。対話型式によりスクリーン・セグメント、Ｒ／ＱＬのコール、データベースへのアクセスなどの各種のモジュールを組み合わせることにより、一つのアプリケーションプログラムを作成することができる。

○エグゼクティブ・システム：システムへのログオンの管理、各エンドユーザごとのメニューの管理、各プログラムの実行管理などを行う。

○クロスリファレンス・システム：総てのコンポーネントの相互関係を管理しており、システムを変更する時の影響範囲などの分析機能を備えている。

○言語変換システム：アプリケーションシステムのテキスト部分を英語→日本語、日本語→独語のように変換・逆変換する効率のよい言語変換機能を提供する。

○チェンジ・マネージメントシステム：アプリケーションシステムに加えられた総ての変更・修正を自動記録し、「アップデートパッケージ」としてまとめることができる。また、すでに稼動しているシステムに、このパッケージを適用する時には、インストール前に変更対象のコンポーネントがユーザーによって修正されているか否かを自動チェックする。インストール後も、修正の行われたコンポーネントやファンクション

に対して、ジェネレーション、コンパイルを自動的に行い、人手の介入を極力防ぐことができる。

4．考案

ＦＲＡＭＥＷＯＲＫを用いることによる利点は、次の様に考えられる。

○開発の早さ：インプリメンターは、データのバリデーションやヘルプ機能のサポート、アボートキーのサポートなど、プログラムの基本的な部分の開発が不要である。

○より効果的なアプリケーションプログラム：ＦＲＡＭＥＷＯＲＫにより、プログラミングやコンピュータの経験は少ないが、豊富な業務知識を持った「ユーザーアナリス」と呼ぶべき人々がシステム開発に大幅に参加できる。

○少ないバグ：アプリケーションの９５％以上が、ジェネレーター、コンパイラーにより作成されるため、ケアレスミスによるバグはほとんど無い。

○プロタイピングが可能：デフォールト機能が強力、かつ、大部分のコンポーネントがすでに用意されているため、プロトタイプの作成が極めて容易である。

○メンテナンスが少ない：バグが少ないこと、クロスリファレンスの完備、強力な各コンポーネント照会機能などにより、非常に楽に早くアプリケーションプログラムを修正することができる。

○統合性に優れる：バラバラのユーティリティではなく、まとまった一つの開発トゥールとして設計されているため、プログラマー間の矛盾やプログラマーとユーザー間の誤解が少なくなる。

○実行時の効率が良い：多くのアプリケーション開発トゥールは、実行時にも制御用の各ファイルにアクセスさせるため、効率が悪くなるものが多い。しかしＦＲＡＭＥＷＯＲＫで作られたアプリケーションプログラムは、ジェネレーション、コンパイル時に、必要な情報の総てが、プログラム内に埋めこまれてしまうため、実行時の効率が極めてよく、最良の手書きプログラムと比べても約１５％減の速度比をもっている。

5．結言

ＭＵＭＰＳは優れたプログラミング言語であり、早く、容易に、アプリケーションシステムを作ることができる。しかし、これだけでは、大規模システムのソフトウェア開発や保守には、不十分である。長期間品質の良いシステムを維持するにはＦＲＡＭＥＷＯＲＫのように、ＭＵＭＰＳの良さを生かしつつ、それが提供できないオート・ドキュメンテーション機能や、その他多くの制御機能を可能にする上位システムが必要である。

Ａｄａの言語設計が行われたとき、プログラム開発環境も同時に定義されたように、ＭＵＭＰＳも、今後同様の努力がなされないと、アプリケーション開発言語として、十分な位置をしめることができなくなるかもしれない。

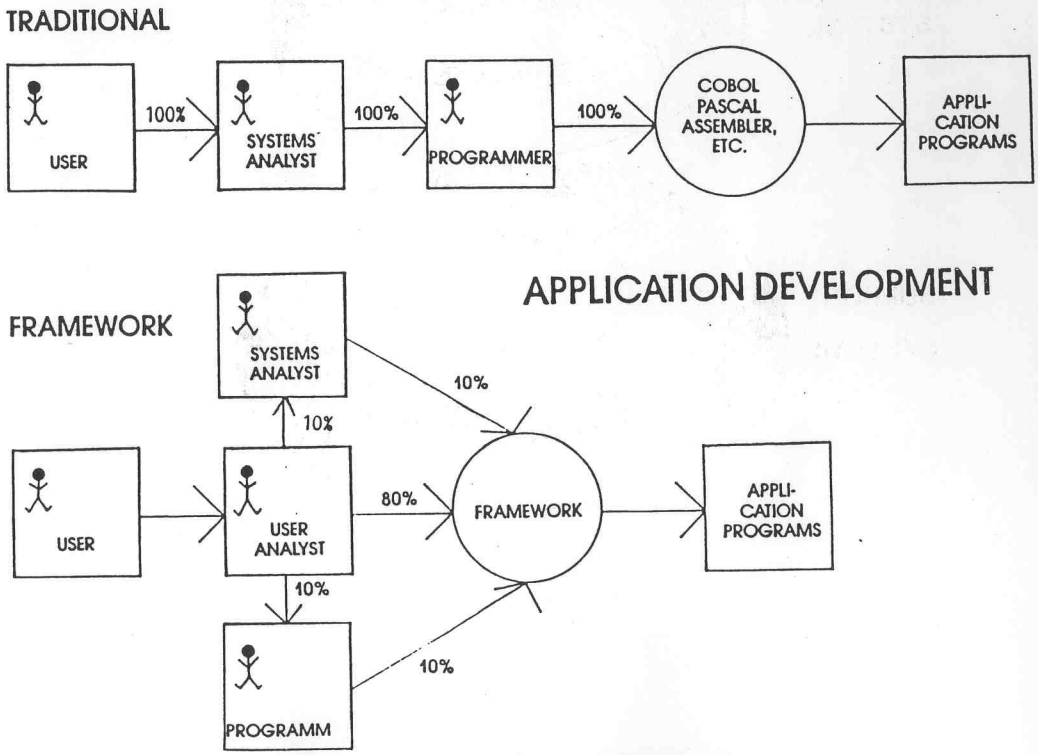


図１．　アプリケーションシステム開発過程の比較

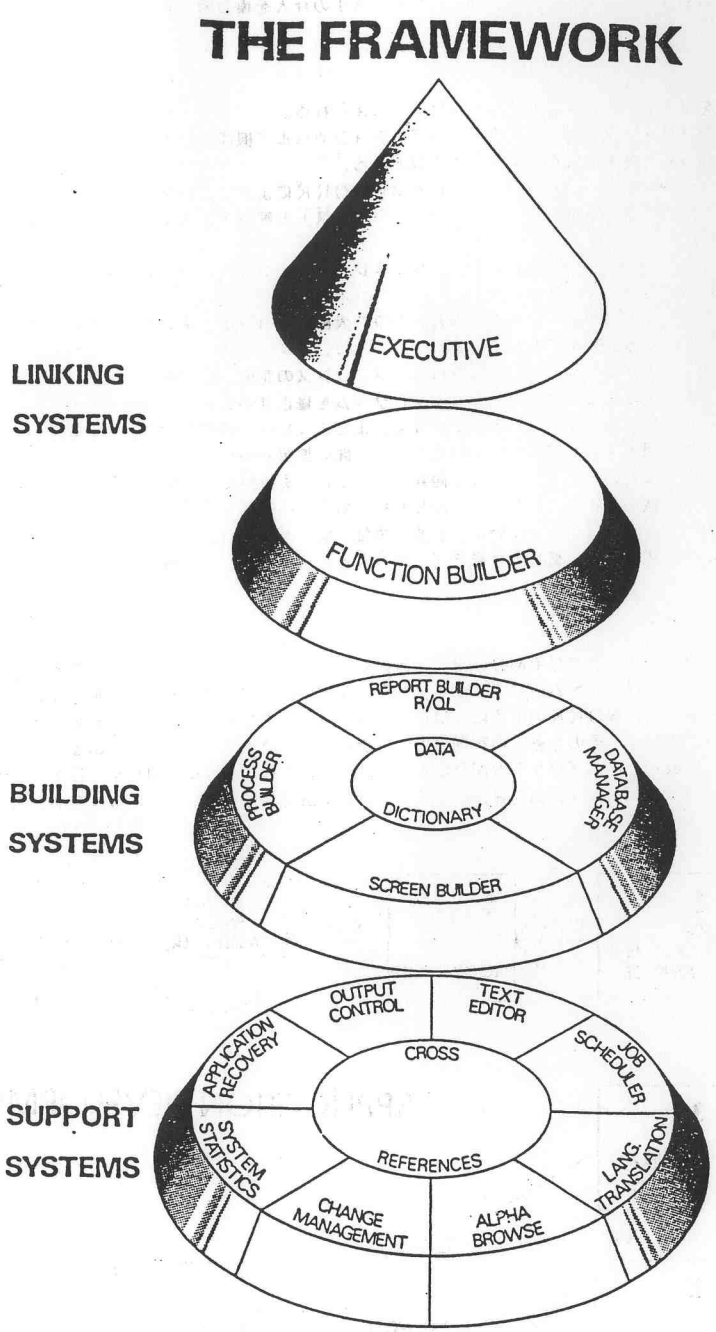


図2．　FRAMEWORKのモジール構造

特別講演３　第１２回日本マンプスユーザーズグループ学術大会（１９８５年８月）

ＰＲＯＬＯＧ とは何か？

○井手 眞理子

日本ＤＥＣソフトウェアサービス部システム第２課
大阪市北区中之島２－２－２ ニチメンビル３Ｆ

【 要約 】

ＰＲＯＬＯＧとは論理プログラミング（ＰＲＯｇｒａｍｍｉｎｇ ｉｎ ＬＯＧｉｃ）の略です。
Ｐｒｏｌｏｇプログラムは述語の定義の集まりで 以下の３形式のいずれかをとります。

(1) Ｐ。　（Ｐは常に成立する）　「事実」
(2) Ｐ :- Ｑ１,Ｑ２,…,Ｑｎ.　（Ｑ１,Ｑ２,…,Ｑｎが成立するときＰが成立する）「規則」
(3) :- Ｑ１,Ｑ２,…,Ｑｍ.　（Ｑ１,Ｑ２,…,Ｑｍは成立するか？）「質問」

【 本文 】

(1) はじめに

１９７１年にフランスのマルセイユ大学のＡｌａｎ Ｃｏｌｍｅｒａｕｒは自然言語の構文解析のために試行錯誤過程を組み込んだ ＳＹＳＴＥＭ Ｑ というプログラムを作成しました。

そして後にその動作が述語論理における証明過程として説明できることからＰＲＯＬＯＧ(Programming in Logic)と改名され、これがそもそものＰｒｏｌｏｇのおこりといわれています。

１９７７年には英国のエジンバラ大学で D.Ｗａｒｒｅｎらにより ＤＥＣ－１０ Ｐｒｏｌｏｇが開発されました。このＤＥＣ－１０ Ｐｒｏｌｏｇが 現在のＰｒｏｌｏｇの標準とされています。

その後 幾つかのシステム上で多くのＰｒｏｌｏｇが開発され １９８２年にはＰｒｏｌｏｇの生れ故郷のマルセイユで第１回ロジック・プログラミング国際会議が開催されるまでに成長しました。
ロジック・プログラミングというのは Ｐｒｏｌｏｇに代表される 論理学をベースとした言語によるプログラミングの総称です。

１つのプログラミング言語が世の中に広まるまで約１０年はかかるといわれていますが Ｐｒｏｌｏｇの場合もほぼ同じような経過をたどっているといえるでしょう。

本発表で使用する言語はＣ－Ｐｒｏｌｏｇです。

Ｃ－Ｐｒｏｌｏｇは ＥｄＣＡＡＤ(エジンバラ大学建築学部) で開発されたＤＥＣ－１０ Ｐｒｏｌｏｇ準拠のＰｒｏｌｏｇインタプリタです。
ソースはＣ言語で書かれており、ＶＡＸ／ＶＭＳ、ＶＡＸ／ＵＮＩＸで動作します。

(2) Ｐｒｏｌｏｇの特徴

「Ｐｒｏｌｏｇを教えるのはむずかしい。特に実用的なＰｒｏｌｏｇプログラミングのできる人材を育成するには 論理学を半年、定理証明を半年、ユニフィケーションを半年ぐらい学んでから初めてＰｒｏｌｏｇのコースに入るのが正式なのかもしれない。」…と これはロジック・プログラミング国際会議の席上でＡ.Ｃｏｌｍｅｒａｕｒが述べた言葉です。

それでは 私たちがよく気にする 一般的なプログラミング言語についての知識は？ というと全く必要ないといえましょう。むしろＦＯＲＴＲＡＮやＢＡＳＩＣなどといった言語の知識があるのが かえってＰｒｏｌｏｇを学ぶときの妨げになるかもしれません。

Ｐｒｏｌｏｇは 事物(Object)間の関係を 式(述語式)で表わします。

例えばこういう文があったとします。

花子は映画が好き。

これをＰｒｏｌｏｇで表現すると 以下のようになります。

好き(花子,映画).

Ｃ－Ｐｒｏｌｏｇは 日本語を述語やアトムとして用いることができますので上のような表現ができます。

一般形としては

述語(事物A,事物B).

となります。これは「 事物A と 事物B の間には 述語 の関係が成り立つ。」と読みます。

ひとつの文は必ず .(ピリオド)で終ります。述語式を ,(コンマ)で複数つないでひとつの文にすることも可能です。大文字で始まるものは 変数として扱われますので 下記のような書き方もできます。

好き(X,映画). (Ｘは映画の好きな人を表わします。)

好き(花子,X). (Ｘは花子さんの好きな物を表わします。)

これが「事実の定義」とよばれるものです。Ｐｒｏｌｏｇには もう１つ表現法があります。

「ルール」といわれるものがそれで 例えば 述語式

同一嗜好(X,Y,Z) :- 好き(X,Z),好き(Y,Z).

は「ＸがＺを好きでかつ ＹがＺを好きであるならば ＸもＹも同じようにＺが好きである。」という関係を表わします。

先程の例では

好き(花子,映画).

好き(太郎,映画).

同一嗜好(X,Y,Z) :- 好き(X,Z),好き(Y,Z).

が定義されていると

同一嗜好(花子,太郎,映画).

が事実として成り立ちます。

これが いわゆるＰｒｏｌｏｇにおける推論機構で 「Ａ→Ｂ かつ Ｂ→Ｃ ならば Ａ→Ｃ である」の三段論法と考えたほうがわかりやすいかもしれません。

それでは次に Ｐｒｏｌｏｇによるプログラム例を考えてみることにしましょう。

プログラムは前述の「事実の定義」と「ルール」に加えて「システムに対する質問(プログラムの実行)」の３形式からできています。以下はプログラムとその実行結果を示しています。

《 ソース・ファイル 》

```
/* 人の定義 */

人(ジュリエット,a,女性).                    /* ジュリエットはＡ型で女性の人。*/
人(ノンノン,o,女性).
人(ロミオ,o,男性).
人(ムーミン,b,男性).

/* 相性の定義 */

相性(a,o).                                  /* Ａ型の女性はＯ型の男性と相性がいい。*/
相性(ab,a).
相性(b,ab).
相性(o,b).

/** プログラム **/

相性のいい人(X,Y) :- 人(X,BW,女性),人(Y,BM,男性),相性(BW,BM).
        /* Ｘさんが女性 Ｙさんが男性 かつ血液型の相性がいいならばＸさんとＹさんは相性がいい */
相性のいい人(X,Y) :- 人(X,BM,男性),人(Y,BW,女性),相性(BW,BM).
        /* Ｘさんが男性 Ｙさんが女性 かつ血液型の相性がいいならばＸさんとＹさんは相性がいい */
```

《 実行結果 》

```
| ?- 相性のいい人(ロミオ,X).     /* ロミオと相性のいい人は誰ですか？ */
X = ジュリエット                 /* ジュリエットです */
yes
| ?- 相性のいい人(ノンノン,Y).
Y = ムーミン
yes
| ?
```

(3) Ｐｒｏｌｏｇはどういったプログラミングに適しているか

Ｐｒｏｌｏｇはリスト処理の得意な強力なプログラミング言語で前述のサンプルプログラムの例をみてもわかるようにプログラムとデータベースが融合しています。また文法規則の数が少なくわかりやすいために極めてプログラミングしやすい言語といえるでしょう。

Ｐｒｏｌｏｇを利用した応用分野としては 自然言語処理・エキスパートシステム・人工知能などが挙げられます。ＬＩＳＰがアメリカ系の言語といわれるのに対して Ｐｒｏｌｏｇはヨーロッパ系の言語といわれ ヨーロッパでの研究がさかんに行われています。

日本でもＩＣＯＴ(新世代コンピューター技術開発機構)がプログラミング言語としてＰｒｏｌｏｇを採用したことからＰｒｏｌｏｇでのエキスパートシステムが注目を集めています。

しかし ひとつ注意してほしいことはＩＣＯＴは 決してＰｒｏｌｏｇで直接 エキスパートシステムを作っているのではなく、知識表現言語を記述するためにＰｒｏｌｏｇを使用しているにすぎない… ということです。

Ｐｒｏｌｏｇは 使いやすい言語ですが 決して万能ではなく 欠点も当然 持っています。
例えば Ｃ－Ｐｒｏｌｏｇの場合は インタプリタであってコンパイラは持っていません。そのため 他言語とのインターフェイスは自ずと限られたものになってきます。即ち Ｃ－Ｐｒｏｌｏｇはオブジェクトイメージを作らないので異種言語で記述されたルーチンをオブジェクトレベルでリンクすることはできません。
しかし サブプロセスを起動することができますので 以下にその例を紹介することにしましょう。
(他言語としては ＶＡＸ／ＤＳＭを使用しています。)

例１) ＤＳＭのルーチン TEST を起動します。 この例では 引数はなくＰｒｏｌｏｇにより起動させられたＤＳＭのルーチンはただターミナル上にメッセージを出力するだけです。

《 ＤＳＭでのサンプルプログラム 》

```
12-AUG-1985 12:51:41.15
Saved by ^%EDT from USR$USRROOT:[OS1]ROUTINES.DSM;
TEST
TEST    ;
        W !,"We are in dsm routine",!
        Q
```

《 Ｐｒｏｌｏｇでのサンプルプログラム 》

```
sample :- write('WE ARE IN PROLOG'),    /* ターミナル上にメッセージを出力します。*/
          system("DSM ^TEST"),          /* サブプロセス中で ^TEST を起動しています。*/
          write('WE ARE NOW IN PROLOG AGAIN').
```

《 実行結果 》

```
$ PROLOG                                /* Ｐｒｏｌｏｇの起動 */

C-Prolog version 1.5b on VAX/VMS

        EdCAAD  &  Nihon DEC
yes
| ?- ['sample.pl'].                     /* Ｐｒｏｌｏｇのプログラムのロード */
sample.pl consulted 108 bytes 2.100086e-02 sec.
yes
| ?- sample.                            /* プログラムの実行 */
WE ARE IN PROLOG
We are in dsm routine
WE ARE NOW IN PROLOG AGAIN
yes
| ?-
```

例２） ファイルを介して引数の受け渡しを行います。

《 ＤＳＭのサンプル・プログラム 》

```
TEST    ;
        W !,"WE ARE IN DSM ROUTINE"
        W !,"PARAMS ="
        S FILE="PARAMS.DAT"
        O FILE
        U FILE R A U 0 W A,!
        C FILE
        Q
```

《 Ｃ－ＰＲＯＬＯＧのサンプル・プログラム 》

```
sample(PARAMS) :-       write('WE ARE IN PROLOG'), /* ターミナル出力 */
                        tell('params.dat'),     /* ファイルのオープン */
                        params(PARAMS),         /* 関数paramsの呼出し */
                        told,                   /* ファイルのクローズ */
                        system("DSM ^TEST"),    /* DSMプログラムの実行 */
                        write('WE ARE NOW IN PROLOG AGAIN'),/* ターミナル出力 */
                        rename('params.dat',[]).

params([]).                                     /* 関数paramsの定義 */
params([X|Y])   :-      write(X),put(32),params(Y).
```

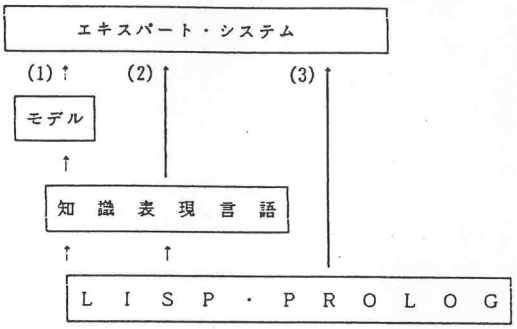
《 実行結果 》

```
C-Prolog version 1.5b on VAX/VMS

EdCAAD  &  Nihon DEC

yes
| ?- ['sample.pl'].            /* プログラムのロード */
sample.pl consulted 460 bytes 3.400087e-02 sec.

yes
| ?- sample(['arg1',2,3]).     /* 実行 */
WE ARE IN PROLOG
WE ARE IN DSM ROUTINE          /* ＤＳＭ内で出力 */
PARAMS = arg1 2 3
WE ARE NOW IN PROLOG AGAIN     /* 制御がＰＲＯＬＯＧに戻る */

yes
| ?- halt.                    /* ＰＲＯＬＯＧインタプリタの停止 */

[ Prolog execution halted ]
```

(4) 人工知能研究分野におけるＰｒｏｌｏｇの役割と今後の動向

人工知能研究用言語の中で ＬＩＳＰがアセンブラであるのに対し Ｐｒｏｌｏｇは ＦＯＲＴＲＡＮのような高級言語の位置にあるといえるでしょう。そのため ＬＩＳＰほど汎用性には富んでいませんが学びやすくある程度の推論機構を持った ユーザーフレンドリィな言語です。ここでは 人工知能の中で現在最も脚光を浴びているエキスパートシステムを例にとって話を進めることにします。

エキスパートシステムを作成するときには 下図のような３レベルが考えられます。



(1) ＬＩＳＰ・ＰＲＯＬＯＧで知識表現言語(ＡＩツールと呼ばれるのもの)を記述しそのツールを用いてプロトタイプを作成していくもの。

(2) 知識表現言語をＬＩＳＰやＰＲＯＬＯＧで記述するところまでは(1)と同じで ツールからいきなりエキスパートシステムを作成しようとするもの。

(3) ＬＩＳＰやＰＲＯＬＯＧを用いてダイレクトに エキスパートシステムを作成するもの。

(3)が最も時間を必要としかつ 失敗する危険性も大きいというのは もうおわかりでしょう。最も確実な方法と思われるのは (1)で ＩＣＯＴなどが成功したのもこの方法をとったからだと考えられます。

現在 人工知能分野で必要なのはしっかりしたユーザーインタフェイスです。それをＰｒｏｌｏｇで全て記述することは無理であるとしても Ｐｒｏｌｏｇの持つ特性を最大限に利用し人工知能研究に役立てていってほしいと思います。

また ＰｒｏｌｏｇでＰｒｏｌｏｇコンパイラを記述することも確かに可能ではありますが そのための時間やマンパワーを考えると ＫＥＥや ＢＲＡＩＮＳ、ＯＰＳ５ といった市販のツール類を使用することをお勧めします。

最後になりましたが ＤＥＣでは 近くコンパイラ付きの Ｑ－Ｐｒｏｌｏｇを販売する予定にしております。

特別講演４　第１２回日本マンプスユーザーズグループ学術大会（１９８５年８月）

ＭＵＭＰＵＳのＰｒｏｌｏｇシンタックスの包含装備

内田達弘

名古屋市東区代官町３９番１５号　マンプスシステム研究所

本文は，マンプスシステム研究所で研究・開発中の知識情報処理システムのプログラミング言語として開発したＧＮＯＳＩＳ，すなわちＭＵＭＰＳ＋Ｐｒｏｌｏｇの概要を述べている．

１．はじめに．

最近知識情報処理とか人工知能などの高度な情報処理に関心が高まっている．

ＭＵＭＰＳを使って，知識ベースを作り上げ，それを利用して行くことがどこまで可能であるか気になるところである．

今回は，この問題を考えながら，ＭＵＭＰＳがＰｒｏｌｏｇの機能を持ったとき，知識情報処理に対し，どれだけの事ができるか，その可能性の一つを，医療診断を例に取り上げ，出来るだけ平易に説明する．

２．知識ベースに対するＭＵＭＰＳの限界．

医療診断プログラムをＭＵＭＰＳで作る場合，必要なデータとして少なくとも病名とその属性，症状，性別，年齢，検査などが必要である．そのうちの症状一つとっても一つの病気に対し，大きく分類しても１０個前後はあり，それらを特定の病気，例えば感染症に限ったとしても，その症状の種類はかなりの数にる．

ＭＵＭＰＳにとって，一つの病名から，その症状などを調べることは非常に簡単なことである．その逆の，いくつかの症状から，それらを属性として含む病名を特定するためには，一般に症状をキーとする逆ファイルを作り，つまり各々の症状別のファイルの全てに属する病名を求めればよい．しかし実際は，このような単純なものではない．

症状というものは，例えば発熱一つをとってみても，急な発熱，悪寒を伴った発熱，弛張熱などいくつかの型があり，場合によっては，発熱が無いことも重要な診断の要素となる場合もある．しかもそれらが感染もしくは，発病後の時間経過や年齢，性別などにより変化することも多く，それらの要素を含めると，病名と症状，症状と症状の関係を表すのに必要なキーやポインタが縦横に必要となる．

さらにまた，ポインタだけではすまない量的または組合せ的関係も診断に必要な情報である．例えばある病気の発病時の体温の範囲，蛋白尿が痕跡程度か何十mg/dl か，それ以上かなど単に有るなしでは判断のつかない事項について調べるには，手続き型言語の特徴として，一般にＩＦがよく使われます．すなわち，知識ベースの一部をルーチン

の中に入れておくことになる．このことは，常にデータベースとルーチンが一対となって存在することを必要とし，新たな関係を登録するごとに，ルーチンの変更が必要となる．

もちろん，ＭＵＭＰＳは間接実行ということができるため，これらの条件を必要とする処理をデータベースに入れておくことは可能であり，実際に用いられている方法である．

しかしながら，個々のルーチンをその都度作成し，テストを行い登録するための労力がどの程度必要かは，十分に推察できる．

結局，行き着く所は，Ｑｕｅｒｒｙのようなプログラムジェネレータ的な方法を必要とし，そして知識が増えるにつれ，ルーチンやポインタのための変数の数が算術級数的に増えて，しかもデータベースの変更があるたびに膨大な「ポインタの付け直し」という作業が始まり，これに要する時間も比例して長くなる．

医学でみられるように日毎，時には時間毎に変化し更新されるべき知識というものにこの方法でどこまで対処できるであろうか，たとえ知識の最先端を作り上げても翌日からは段々と古く（化石）なってゆく運命は避けられません．

３．知識ベースとＰｒｏｌｏｇ．

医療診断にＰｒｏｌｏｇを使用した場合，今の問題の幾つかは，Ｐｒｏｌｏｇにとって全く問題にならない．Ｐｒｏｌｏｇでは，病名とその属性は「事実」として登録するだけで済み，個々の症状によって異なる条件な関係式などは，症状固有の「規則」として登録するだけ済む．しかもそこには何等かのポインタや手続き的なプログラムは一切不用である．

Ｐｒｏｌｏｇは非常に頭が良く，考えるということにかけては他のプログラミング言語に比べ，際だって優れていると言えますが，残念なことに，単に考えるということが得意なだけである．

今大会の『”考える”ＭＵＭＰＳ』の”考える”だけの部分がＰｒｏｌｏｇといえるでしょう．

Ｐｒｏｌｏｇとは対照的に，ＭＵＭＰＳは外界との接触のうまさ，すなわち人間とのコミニュケーションやＭＵＭＰＳとＭＵＭＰＳとの会話，そして実時間での処理，また文字列に対しての細かい処理，すなわちあちらこちらのデータを取り出してつないだり，分割したり，比較したりすることを得意とするまさに”何でも屋”的能力を持つ．しかし，その分読みにくいプログラム，すなわちドキュメントがないと解読困難といったプログラムができてしまうことがある．

そこで，この頭だけで，しかも手も足も耳も口もないＰｒｏｌｏｇをＭＵＭＰＳの中に組み込んだ場合の相乗効果を期待し，開発したのがＧＮＯＳＩＳである．

４．ＭＵＭＰＳ＋Ｐｒｏｌｏｇ＝ＧＮＯＳＩＳ

ＧＮＯＳＩＳのシンタックスおよび用法を感染症の医療診断プログラム（実際には，医療以外でも同様な複雑な判断を必要とする場合にも使えますが）に使われているシンタックスとセマンティックスを中心に説明する．

（注）ＧＮＯＳＩＳはＭＵＭＰＳの拡張である．したがって説明を要する部分は，拡張した部分のみで充分ある．

```
>SET ^DIS("CHOLERA",["cyanosis"])=""                    (1)
```

このプログラムは，”法定伝染病の一つコレラには，その症状の一つとしてｃｙａｎｏｓｉｓがある”という「事実」をデータベースとして^ＤＩＳというグローバル変数に登録するものである．

ＧＮＯＳＩＳは，値を持つグローバル変数およびローカル変数の全てが「事実」として扱われる．但し後で説明するリストを値として持つ変数を除く．したがって，

```
>SET A=0,B=1,C="comment"
>SET Food("fruit","apple")=""
```

の A,B,C および Food("fruit","apple") は「事実」である．

「事実」を常に「真」であるものとして定義すると，

```
>KILL X,^Y(1)
```

の X ,X(1),...,^Y(1),Y(1,2)... などは逆に常に「偽」であるものとして扱われる．

さて，(1) で添字にＭＵＭＰＳにはない始めてみるシンタックス

["cyanosis"]

がある．この鍵括弧 [] はリストを表す記号である．

リストはＧＮＯＳＩＳにとって重要な役を演じるだけでなく，ＭＵＭＰＳにとっても，使い方によっては非常に役にたつ．

リストの例を示す．

```
[]                                  空リストまたはＮＩＬＬと呼ぶ．
["A","B","C"]
["DIGIT",[0,1,2,3,4,5,6,7,8,9]]
```

リストを操作するための関数を４個用意した．どの関数も基本的なものであり，リストに対しての殆どの処理がこれらの組み合わせで可能である．

$ＬＡＰＰＥＮＤ
$ＬＥＸＴＲＡＣＴ
$ＬＦＩＮＤ
$ＬＬＥＮＧＴＨ

ここでは，その概略を示す．詳細は資料Ｉを参照のこと．

１）　＄ＬＬＥＮＧＴＨ

　要素の数を値とする．

```
>WRITE $LLENGTH(["A","B"])
2
>WRITE $LLENGTH([])
0
>WRITE $LLENGTH([[1,2,3,4],5)      リスト[1,2,3,4]も一つの要素である．
2
```

２）　＄ＬＡＰＰＥＮＤ

　リストに要素を加える．

```
>WRITE $LAPPEND([1,2],[3])
[1,2,3]
>WRITE $LAPPEND([1,2],[3,4])
[1,2,3,4]
>WRITE $LAPPEND([1,2],[])
[1,2]
>WRITE([],[1,2])
[1,2]
```

３）　＄ＬＥＸＴＲＡＣＴ

　リストの要素を取り出す．

```
>WRITE $LEXTRACT(["A","B","C"])
A
>WRITE $LEXTRACT(["A","B","C"],1)
A
>WRITE $LEXTRACT(["A","B","C"],2)
B
>WRITE $LEXTRACT(["A","B","C"],4)
       (null)
```

新しいコマンドＰＲＯＶＥを一つ追加した．ＰＲＯＶＥは推論機能を実行するためのコマンドである．

```
>PROVE [#^DIS("CHOLERA",["cyanosis"])]          (2)
```

このプログラムは，データベースの中に ^DIS("CHOLERA",["cyanosis"]) が「事実」として登録されているかを調べる．

「事実」であれば，＄ＴＥＳＴに１がＳＥＴされる．そうでなければ，＄ＴＥＳＴに０がＳＥＴされる．

＃記号は，Ｐｒｏｌｏｇでいう述語（ｐｒｅｄｉｃａｔｅ）を明示するためのもので＃の後に続く変数名が述語を表す．したがって #^DIS("CHOLERA",["cyanosis"])は述語である．

```
>PROVE [#^DIS("CHOLERA",*SYMPTOM)]              (3)
>WRITE SYMPTOM
["cyanosis"]
```

このプログラムは，データベースの中に第一添字がＣＨＯＬＥＲＡであるものが登録されているかを調べる．

＊ＮＡＭＥはＰｒｏｌｏｇでいう変数である．ＧＮＯＳＩＳではこれを論理変数と呼ぶ．論理変数は，対応する相手があれば何とでもユニファイ（一種のパタンマッチ）する．但しユニファイが可能な相手が複数存在する場合＄ＯＲＤＥＲの順とする．

また，ＰＲＯＶＥコマンドのアーギュメント内の論理変数はユニファイした相手が値を持つ場合，論理変数と同一名のローカル変数にその値がＳＥＴされる．したがって（2)の例ではローカル変数ＳＹＭＰＴＯＭにユニファイした値がＳＥＴされる．

この応用として，

```
>PROVE [#^DIS(*name,["cyanosis"])]              (4)
>WRITE name
CHOLERA
```

というように，論理変数をうまく使うと，こんな単純な使い方で，添字のサーチが，その位置に関係なく簡単にできる．同じことをＭＵＭＰＳで行わせると，＄ＯＲＤＥＲや＄ＤＡＴＡを何度も繰り返し使った結構めんどうなプログラムになる．

ＭＵＭＰＳにとってもっと複雑なことが簡単にでる．例えば，

```
>PROVE [#^DIS(*name,["cynosis"]),#^DIS(*name,["eruption"])]
>WRITE name
TYPHUS FEVER
```

は，添字の２番目の値として ["cyanosis"]と ["eruption"]を持つグローバル＾ＤＩＳの中から調べるプログラムである．いいかえると，症状としてｃｙａｎｏｓｉｓがありｅｒｕｐｔｉｏｎもある病気は何かを探すプログラムとみることができる．

先ほどのプログラム(4)

```
>PROVE [#^DIS(*name,["cyanosis"])]
>WRITE name
CHOLERA
```

では，*nameがユニファイする事実は，ＣＨＯＬＥＲＡであった．しかし，この後で，アーギュメントが空のＰＲＯＶＥ

```
>PROVE
```

を行えば，

```
>WRITE name
THYPUS FEVER
```

となる．すなわちアーギュメントが空なＰＲＯＶＥコマンドは，直前の推論を否定し，新たに別の候補を見つけ出す．（もしあれば）

したがって，全ての病名とその症状を表示するには，次のようにすればよい．

```
>PROVE [#^DIS(*name,*symptom)]
>FOR I=1:0 QUIT:'$TEST WRITE !,name,?20,symptom PROVE
CHOLERA             ["cyanosis"]
CHOLERA             ["diarrhea"]
 . . .
 . . .
```

次は規則の例である．これは非常にＰｒｏｌｏｇらしい使い方の例でもある．

```
>KILL ^member
>SET ^member(*x,[*x:*y])=""
>SET ^member(*x,[*y:*z])=[#^member(*x,*z)]
```

これは ^member という規則の登録である

ｍｅｍｂｅｒまたは，これと同じ機能の例はたいていのＰｒｏｌｏｇの教科書に載っているので詳細についてはそちらを見ることを勧める．

ここでは実際の使用例のいくつかをあげる．

```
>PROVE [#^member(3,[1,2,3,4])]
>WRITE $TEST
1
>PROVE [#^member(2,[1,3,5,7])]
>WRITE $TEST
```

```
0
>PROVE [#^member(*x,["A","B","C"])]
>WRITE x
A
>PROVE
>WRITE x
B
>PROVE
>WRITE x
C
>PROVE
WRITE $TEST
0
```

^ｍｅｍｂｅｒの定義のなかで，リストの中に：という記号がでてきた．：は２進木リストの節にあたる記号である．

：は可能な限りその右のリストの記号と共に省略され，かわりに"，"で表す．

"，"のない表現を内部表現，：を全て省略した表現を外部表現と仮に呼ぶとするとその対応は次の例のようになる．

外部表現	内部表現
[]	[]
[1]	[1:[]]
[1,2]	[1:[2:[]]
[[1],2]	[[1:[]]:[2:[]]
[[]]	[[]:[]]

この内部表現で式を記述することができる．

外部表現と混在してもよい．次はその例である．

```
>WRITE [1:[]]
[]
>WRITE [1:[2:[]]
[1,2]
>WRITE [1,2,3:[4,5,6]]
[1,2,3,4,5,6]
>SET X=["a","b"]
>WRITE [1,2:X]
[1,2,"a","b"]
```

定義の上では，２進木の節の右側には，リストもしくは空リストしか現れないが，論理変数に限って許す．

```
>WRITE [1:*x]
[1:*X]
```

これは次の用法ができるようにするためである．

```
>SET X([2,3,5,7,11])=""
>PROVE [#X([*CAR:*CDR])]
>WRITE CAR,!,CDR
2
[3,5,7,11]
```

例にあげたプログラムの中にＭＵＭＰＳに組み込んだ新しい機能の殆どが使われている．

以上が主な新しいシンタックスとセマンティックスである．

５．ＧＮＯＳＩＳのインプリメント．

ＧＮＯＳＩＳはマンブスシステム研究所において，研究開発を進めている言語である．

ＧＮＯＳＩＳシステムの仕様（１９８５年８月）を簡単に述べる．

データ構造は一階層のＢ－ｔｒｅｅである．

メモリ管理はＢＵＤＤＹシステムを使用し，キャッシュはＬＲＵで行っている．

実行形式は，中間コードを作るいわゆるプレコンパイラである．

６．おわりに．

現在ＧＮＯＳＩＳは，Ｕ－ｓｙｓｔｅｍというオリジナルＤＯＳ上で動くが，ＭＵＭＰＳのユーザにＧＮＯＳＩＳの研究利用について広く協力を求めるためマイクロコンピュータの汎用ＤＯＳに移植する予定である．

参考文献．

Programing in prolog	W.F.Clocksin C.S.Mellish	Springer-Verlag
micro-PROLOG:Programing in Logic	K.L.Clark F.G.Mccabe	PHI
Prolog	中島秀之	産業図書

資料　Ｉ

Ｉ　　この資料は，ＭＵＭＰＳにＰｒｏｌｏｇを組み込んだことにより，追加または拡張された部分のｓｙｎｔａｘ及びｓｅｍａｎｔｉｃｓの概要である．
ＭＵＭＰＳは拡張の場合Ｚを用いることになっているが，リストの構造記述などは，ＭＵＭＰＳのルールを出ているので敢えて，Ｚを用いない事にし，呼び名も拡張ＭＵＭＰＳではなく，ＧＮＯＳＩＳと呼ぶことにする．

Ｉ．１　リスト（ｌｉｓｔ）

ＧＮＯＳＩＳでは，２進木リストをリストとする．

Ｉ．１．１　リストの例．

[”Ａ”]	”Ａ”は要素と呼ぶ．
[]	要素のないリストを空リストと呼ぶ．
[”Ａ”，”Ｂ”]	”Ａ”と”Ｂ”がこのリストの要素である．
[[１]，２]	[１]と２がこのリストの要素である．
[[]]	内側の[]が要素である．

Ｉ．１．２　リストの評価．

```
>Ｗ　[１＋２]
[３]
>Ｗ　[”Ａ”_”Ｂ”]
[”ＡＢ”]
```

Ｉ．１．３　リストに関する演算．

：は２進木の節．

```
>Ｗ　[１：[]]，！，[１：[２]]，！，[１：[２，３]]
[１]
[１，２]
[１，２，３]
>Ｓ　Ｘ＝[”ａ”，”ｂ”]　Ｗ　[１：Ｘ]
[１，”ａ”，”ｂ”]
```

＆は単項演算子．　強制リスト解釈．

```
>W  &"[1,2+1]",!,&"ABC"
[1,3]
[]
```

I.1.4 リストに関係する関数.

（listはリストを値とする式，nは整数を値とする式，eは式）

・$LEXTRACT(list)　リストの先頭の要素を取り出す.

```
>W  $LEXTRACT([])
    ← NULL
>W  $LEXTRACT(["a","b"])
a
>W  $LEXTRACT([["x","y"],"z"])
["x","y"]
```

・$LEXTRACT(list,n)　リストのn番目の要素を取り出す.

```
>W  $LEXTRACT([1,3,5],3)
5
```

・$LEXTRACT(list,n1,n2)

リストのn1番目の要素からn2番目の要素をリストにして取り出す.

```
>W  $LEXTRACT([2,4,6],2,2)
[4]
>W  $LEXTRACT([1,3,5,7,11],2,4)
[3,5,7]
```

・$LAPPEND(list1,list2)

list1にlist2の要素を追加する.

```
>W  $LAPPEND([1,2,3],[4,5,6])
[1,2,3,4,5,6]
>W  $LAPPEND([1,2,3],[])
[1,2,3]          変化しない.
>W  $LAPPEND([],[1,2,3])
[1,2,3]
```

・$LFIND(list,e)

リストの要素の中でeの値と同じものの位置＋1を値とする.

```
>W  $LFIND([1,2,3,4],2)
3
>W  $LFIND([1,[2],2,3],2)
```

```
4
>W  $LFIND([1,2,3],5)
0
```

・$LFIND(list,e,n)

リストの要素の中でn番目の要素からみて，始めてに現れたeの値と同じものの位置＋1を値とする．

```
>W  $LFIND(["A","B","A"],"A",2)
4
```

・$LLENGTH(list)

リストの要素の数を値とする．

```
>W  $LLENGTH(["A","B","C"])
3
>W  $LLENGTH([1,[2,3,4]])
2
```

・$LLENGTH(list,e)

リストの要素の中でeの値と同じものの数＋1を値とする．

```
>W  $LLENGTH(["A","B","A"],"A")
3
>W  $LLENGTH([1,2,2,[2]],2)
3
```

I．2　述語（perdicate）．

I．2．1　次ぎのような形の表現を述語とよぶ．

```
#A      #A(1)      #^X      #Y([1],2)
*B("a")    !    (  !  をカットオペレータと呼ぶ．)
```

I．2．2　述語の評価．

```
>W  #X,!,#A(1),!,#B(1+1,3)
#X
#A(1)
#B(2,3)
```

I．3　論理変数．

Prologでいう変数に相当する．

Ｉ．３．１　＊を変数名につけて表す．

＊Ｘ　　＊％１

Ｉ．４　ＰＲＯＶＥコマンド．

ＰＲＯＶＥコマンドは，Ｐｒｏｌｏｇの推論機能を開始するコマンドである．
アーギュメントの有無により，処理が異なる．

Ｉ．４．１　アーギュメントが有る場合．

```
>Ｓ　Ａ＝””
>ＰＲＯＶＥ　［＃Ａ］　Ｗ　＄ＴＥＳＴ
1
```

述語と同一名の変数が存在する場合，＄ＴＥＳＴに１がセットされる．（真）
存在しない場合は，＄ＴＥＳＴに０がセットされる．（偽）

```
>Ｋ　Ｂ
>ＰＲＯＶＥ　［＃Ｂ］　Ｗ　＄ＴＥＳＴ
0
>Ｓ　Ａ＝１，Ｂ＝１　Ｋ　Ｃ
>ＰＲＯＶＥ　［＃Ａ，＃Ｂ］　Ｗ　＄ＴＥＳＴ
1
>ＰＲＯＶＥ　［＃Ａ，＃Ｃ］　Ｗ　＄ＴＥＳＴ
0
```

述語の数が２つ以上のとき，すべての述語がそれぞれ真のとき＄ＴＥＳＴに１がセットされる．

Ｉ．４．２　ＰＲＯＶＥコマンド実行中の論理変数の意味．

論理変数は，このコマンド実行中では，Ｐｒｏｌｏｇでいう変数と同じ働きをする．

```
>Ｋ　Ａ　Ｓ　Ａ（”ａ”）＝１
>ＰＲＯＶＥ　［＃Ａ（＊ｓｕｂ）］　Ｗ　ｓｕｂ
a

>Ｋ　Ａ　Ｓ　Ａ（＊ｘ，＊ｘ）＝””
>ＰＲＯＶＥ　［＃Ａ（１２３，＊Ｙ）］　Ｗ　Ｙ
１２３
```

Ｉ．４．３　アーギュメントが無い場合．

直前に行ったＰＲＯＶＥコマンドの終了状態を強制的に否定し別の推論すなわちバックトラックを行う．

```
>Ｋ　Ｘ　Ｓ　Ｘ（”Ａ”）＝１，Ｘ（”Ｂ”）＝１
>ＰＲＯＶＥ　［＃Ｘ（＊ｓｕｂ）］　Ｗ　ｓｕｂ
Ａ
>ＰＲＯＶＥ
>Ｗ　ｓｕｂ
Ｂ
```

Ｉ．４．４　組み込み述語．

・！　カットオペレータ．　バックトラックを止める．

・＃＄ＥＶＡＬ（式）
　式の値が真（０以外）のとき，この述語は真になる．
　式は，ＰＲＯＶＥコマンド実行時以外には，評価しない．

・＃＄ＷＲＩＴＥ（式）
　ＰＲＯＶＥコマンド実行中に，この述語が現れると，式の値を表示する．
　この述語は常に真である．
　式は，ＰＲＯＶＥコマンド実行時以外には，評価しない．

資料　ＩＩ

ＩＩ　サンプルプログラム

ＩＩ．１　論理変数と値のユニファイ．

```
>Ｋ　Ａ　Ｓ　Ａ（”ａｐｐｌｅ”）＝１
>ＰＲＯＶＥ　［＃Ａ（＊ｘ）］　Ｗ　＄ＴＥＳＴ，！，ｘ
１
ａｐｐｌｅ

>Ｋ　Ｘ
>Ｓ　Ｘ（＊ｘ，＊ｘ）＝””
>ＰＲＯＶＥ　［＃Ｘ（＊ｓｕｂ，”ｏｒａｎｇｅ”）］　Ｗ　ｓｕｂ
ｏｒａｎｇｅ
```

```
>K  X
>S  X("color","white")=1
>S  X("color","blue")=1
>S  X("food","apple")=1
>PROVE  [#X(*attribute,"blue")]
>W  attribute
color
```

II.2 ：論理変数とリストとのユニファイ.

```
>K  X  S  X([1,2,3,4,5])=1
>PROVE  [#X([*CAR:*CDR])]
>W  CAR,!,CDR
1
[2,3,4,5]
```

II.3 memberの定義と使い方.

```
>K  member
>S  member(*x,[*x:*y])=1
>S  member(*x,[*y:*z])=[#member(*x,*z)]
>PROVE  [#member(2,[1,2,3,4])]  W  $TEST
1
>PROVE  [#member(2,[1,3,5])  W  $TEST
0
>PROVE  [#member(*x,["A","B","C"])]
>W  x
A
>PROVE
>W  x
B
>PROVE
>W  x
 C
>PROVE
>W  $TEST,!,x
0
C
```

II．4　リストの反転　reverse．

```
>K reverse,rev1
>S reverse(*list,*tsil)=[#rev1(*list,[],*tsil)]
>S rev1([],*result,*result)=1
>S rev1([*x:*y],*tsil,*result)=[#rev1(*y,[*x:*tsil],*result)]
PROVE [#reverse([1,2,3,4,5],*x)]
>W x
[5,4,3,2,1]
```

II．5　否定　not．

```
>K　not
>S　not(*p)=[*p,!,#$EVAL(0)]
>S　not(*p1)=1

>K　X　S　Y
>PROVE　[#not(#X)]　W　$TEST
1
>PROVE　[#not(#Y)]　W　$TEST
0
>PROVE　[#not(#not(#Y))]　W　$TEST
1
```

II．6　グローバル　リスタ．

```
>PROVE [#^DIS(*name,*symptom)]
>F I=1:0 Q:'$TEST  W !,name,?20,symptom,!," comment ",^DIS(name,symptom) Q
CHOLERA             ["cyanosis"]
 comment
CHOLERA             ["diarrhea"]
 comment Generally painless diarrhea.
  .
  .
  .
```

ＩＩ．７　医療診断プログラムの例(一部)．

規則の登録．

```
K ^member S ^member(*x,[*x:*y])=1,^member(*x,[*y:*z])=[#^member(*x,*z)]
K ^not S ^not(*p)=[*p,!,#$EVAL(0)],^not(*q)=1
K ^include
S ^include(*x,[])=1,^include(*x,[*y:*z])=[#^member(*y,*x),#^include(*x,*z)]
K ^intersection,^int
S ^intersection(*x,*y,*z)=[#^int(*x,*y,*z,[])]
S ^int([],*x,*y,*y)=1
S ^int([*x:*y],*z,[*x:*w],*t)=[#^member(*x,*z),#^int(*y,*z,*w,*t)]
S ^int([*x:*y],*z,*w,*t)=[#^int(*y,*z,*w,*t)]
K ^exclude
S ^exclude(*x,*y)=[#intersection(*x,*y,*z),!,#$EVAL(*z=[])]

DIAGNOSIS W #,!!!!!,?30,"*** DIAGNOSIS ***",!!!
 S SYMPTOM=$LEXTRACT(^GN($LEXTRACT(^NAMEL)))
 S QUESTL=[],TRUEL=[],FALSEL=[],X=^NAMEL
 D DIAG1 Q:ANS="*"  G DIAGNOSIS
DIAG1 W !!,?10,"<< ",SYMPTOM,"  >>  (Y/N/?) " R ANS,!
 I ANS="E" R !,"  << ? symptom ? >>  == ",SYMPTOM G:SYMPTOM="" DIAG1 S ANS="Y"
 S QUESTL=$LAPPEND(QUESTL,[SYMPTOM])
 I ANS="Y" S TRUEL=$LAPPEND(TRUEL,[SYMPTOM])
 I ANS="N" S FALSEL=$LAPPEND(FALSEL,[SYMPTOM])
 G:ANS="?" DIAG2 Q:ANS="*"  S Y=X,X=[]
 F I=1:1:$LLENGTH(Y) D DIAG4
 I X=[] W !,"Not found" Q
DIAG2 F I=1:1:$LLENGTH(X) D DIAG3 G:$TEST DIAG1 Q
DIAG3 S SLIST=^GN($LEXTRACT(X,I))
 P [#^member(*SYMPTOM,SLIST),#^not(#^member(*SYMPTOM,QUESTL))] Q
DIAG4 S NAME=$LEXTRACT(Y,I)
 P [#^include(^GN(NAME),TRUEL),#^exclude(^GN(NAME),FALSEL)] Q:'$TEST
 S X=$LAPPEND(X,[NAME]) W !,"There is a possibility of",?30,NAME,"."
 Q:ANS'="Y" W !," ",^DIS(NAME,[SYMPTOM]) Q
  ;   ^ＧＮ（病名）＝症状のリスト
  ;   ^ＮＡＭＥ＝病名のリスト
```

特別講演５　第１２回日本マンプスユーザーズグループ学術大会（１９８５年８月）

MSM－Where To From

David J. Marcus
Vice－President, Micronetics Design Corp.
Member of MDC

（内容要約）

MSM(Micronetics' Standard MUMPS) は、以下の設計思想のもとに開発されている。

① 当初からマルチユーザシステムを実現する。
② 高いパフォーマンス。
③ エンドユーザに対してはＤＥＣ（ＤＳＭ）との互換性を保つ。
④ 種々のハードウェアに対し移植性を高める。
⑤ マルチ言語環境での稼動。（ＵＮＩＸ，MS／DOS）
⑥ マルチプロセッサ・分散データベース、ネットワーク等のマルチノードオペレーションを可能とする。

移植性を高かめるためＣ言語を採用し、現在では、モトローラー６８０００をベースにしたＵＮＩＸシステム（住友電工製Ustation，ＡＴ&Ｔ製３Ｂシリーズ等）およびインテル８０８Ｘをベースにしたｍｓ／ＤＯＳシステム（ＩＢＭ製ＸＴ，ＡＴ等）で稼動する。

今後、ＩＢＭ４３００等のメインフレームやＨＰ３０００等への移植拡大をするとともに、マルチノードオペレーションの開発を行なうことが大きなテーマである。マルチノードオペレーションは以下のサポートを行なう予定である。

① ネットワークサポート、イーサーネット等のＬＡＮ，ＳＮＡ・Ｘ．２５等の広域ネットワークのサポート。
② 分散データベースのサポート。
③ マルチ・プロセッサオペレーションのサポート、複数ＰＣがデータベースを共有するファイルサーバ機能および、複数プロセッサが同じタスクを共有する形や、複数プロセッサで機能を分担する様な形でのマルチ・プロセッサオペレーションのサポート。

MSM, Micronetics' Standard Mumps, has enjoyed a carefully planned growth path. From its inception to its present form, MSM has been carefully designed and implemented to anticipate and stay ahead of the needs of its users. I would like to briefly review the requirements that formed the foundation of MSM, the system architecture that supports it, and the road map which we are following.

MSM began in late 1980. The general guidelines that shaped the foundation and future of MSM were:

- Multi-User from initial stages (not a single-user which is then redesigned and converted to multi-user)
- High performance
- DEC (DSM) compatibility for the end-user
- Easy portability to different hardware
- Co-exist in multi-lingual environments (Ex: UNIX, MS/DOS, ...)
- System architecture to allow fail-safe multi-node operation (multi-processors, multiple distributed data bases, networking, etc)

It was clear when we began in late 1980 that some of these objectives could not be attained with the then commercially available hardware. It was also immediately clear that these objectives would have to wait for the Motorola 68000 processor to come of age. We therefore embarked on a two phase MSM growth path. The first stage was MSM-09 which ran as a dedicated MUMPS interpreter on Motorola 6809 processors. To gain the required performance we implemented MSM-09 in machine language, which knowingly violated our original guideline of easy portability to other hardware. However, that was not a serious flaw since except for DEC mini-computers there was not a strong market presence for any other hardware. We thus met the first three of our objectives:

- Multi-User
- High performance
- DSM externals compatibility

We were indeed pleased to benchmark faster than a PDP 11/34 (without cache, DSM 1.0). A 6809-2 processor was capable of supporting 16+ users for a small fraction of the cost of a PDP 11/34.

One very beneficial byproduct of MSM-09 for us is that we gained significant insights and knowledge as to what needs to be very efficient in a high

performance MUMPS system. This knowledge provided us with a crucial advantage when we entered the second phase of MSM. The architecture we designed was one of "modularity segregated by functionality". MSM was designed as three separate components which communicated along very well defined interfaces. They were:

- Language processor (hardware/system independent)
- Database management (hardware/system independent)
- Operating system management (dispatching, i/o drivers, etc)

The Language processor performed all the MUMPS commands. The Database manager handled the logical structure of the database (globals/routines). The Operating system manager performed all the task management, timer management, and I/O drivers. This last component contains all the hardware and host operating system dependencies.

With the advent of the Motorola 68000 and Intel 808x processors we undertook the effort to meet the remaining objectives. To significantly improve on performance we expanded the architecture to that of a compiler. To allow for easy portability to other hardware we used the C language. Our internal architecture already allowed us to exist in a multi-lingual environment. To run under a host operating system we merely had to replace the Operating Systems component with host-specific routines and lo-and-behold we are operational under a new host. Thus due to very careful original planning and designing we have by now achieved all but the last of our goals.

This brings us up todate on MSM. It is very clear that our decisions early in the product cycle of MSM have paid off. Today MSM is available on a wide variety of hardware and host operating systems. These include:

- Ustation from Sumitomo (under UNIX III / UNIX V)
- Motorola 6600 (megaframe) (UNIX V / Stand-alone)
- IBM/PC/XT/AT and other PC's (MS/DOS)
- Perkin Elmer 3200 series systems (UNIX V / OS32/RT32)
- Wang/VS (VS)
- AT&T 7300 (UNIX V)
- AT&T 3B family (UNIX V)
- NCR Tower (UNIX V)
- Tolerant 32032 based systems (UNIX V)
- VAX (UNIX V)

- PLEXUS (UNIX III)
- PIXEL (UNIX VII)
- Etc.

The decision to use the C language allowed us to implement a Kanji version of MSM for the Ustation in several months instead of years. The decision to use C also solved for us one the thorniest problems of porting software. That is: how to maintain multiple versions for different systems at the same upgrade level. By using the conditional compilation features of the C language we have just one copy of the source which generates MSM for each of the above systems.

I have just briefly described MSM's growth to its present state. The 'Where to from here?' is a two pronged effort. The first is an on-going effort at Micronetics to port to new machines. Our next major port is onto the IBM mainframes and the Hewlett Packard 3000 systems. Both of these systems represent a big step for MSM. Todate, most of the systems installed in the field are in the 20 and under user range. By going to the IBM 4300 and the HP 3000 systems, MSM can compete in the multiple VAX market. On these systems the number of concurrent users is significantly increased.

The second prong is to implement the last of the objectives: the multi-node operation. The multi-node operation includes:

- networking
    - + LAN (Local Area Network, ex: Ethernet)
    - + WAN (Wide Area Networkd, ex: SNA, X.25)
- distributed database
    - + multiple independent databases across machines
    - + single database segemented across multiple machines
- multi-processor operation
    - + independent processors sharing a common database (ex: multiple PC's sharing a common file server)
    - + homogeneous multi-processor systems - multiple processors each performing similar tasks (ex: multiple 68000 processors sharing a common task load)
    - + heteregeneous multi-processor systems - multiple processors each performing different tasks (ex: terminal processor, disk processor, applications processor, etc)

Each of the above modes overlaps the other and yet presents problems which are unique to each mode. The challenge was to develop an architecture that allows each mode to operate efficiently. We believe we have designed such an architecture. It will allow us to support full multi-node operation. The approach we took was to look at multi-node operation from a different perspective. We carefully analyzed the various proposed configurations of each multi-node operation and extracted the common elements from each mode.

With careful observation we saw that the total system operation can be looked upon as a set of 'm' processes executing on a set of 'n' processors with some inter-process/inter-processor communications and where 'm >= n'. The aspects that distinguish one mode of multi-node operation from another is the ratio of processes to processors and their inter-connectivity as well as the bandwidth of their communication. Complicating the picture may be the restriction that certain processors can only perform certain processes (ex: a terminal processor capable of only terminal I/O). As an example, an IBM/AT system of today has a single processor running multiple processes (the 80286 performs the I/O as well as execute the application). More sophisticated system may have a set of

differentiated processors each performing a special subset of the processes (ex: applications processor, terminal processor, disk processor, etc). Some of the multi-processor systems becoming available on the market support multiple applications processor sharing a part of memory (tightly coupled). Networking can be seen as just a mixture of processors with varying bandwidths of communications and limited connectivity between nodes.

Rather than design a separate architecture for each mode we designed a general architecture which lends itself to the general management of 'm' processes on 'n' processors. The customization to a particular mode becomes mostly a choice of plug-in modules (drivers). As a result, MSM will be capable of supporting the following configurations:

- Tightly coupled multi-processor (MP) systems where 2 or more processors share a common memory and each performing some of the MUMPS tasks (Ex: Motorola 6600 with multiple Application Processors)
- Loosely coupled multi-processor systems where multiple processors perform differentiated tasks (Ex: Tolerant 32032 based systems, where I/O Processors are programmable)
- Independent processors sharing a common database (ex: multiple PC's sharing a database in a file server)
- Independent processors each with a private database forming a larger logical total database in a ring/star arraingement (ex: multiple NCR Towers communicating over LAN)
- Permutations of the above

In all of the configurations I have reviewed, we will continue to maintain the performance and reliability levels that our users have come to expect. There are exciting times ahead for MSM. Based on our plans we will advance the usage of MUMPS into areas never before exposed to MUMPS. The distributed database approach will allow hundereds of networked users to access common data in a fail-safe environment. By pioneering the multi-node operations arena, MSM promises to be the defacto standard for MUMPS for the years to come.

特別講演６　　第１２回日本マンプスユーザーズグループ学術大会（１９８５年８月）

新時代ネットワークを迎える漢字ＤＳＭ－１１

Ｏ小林　泰道

日本ディジタルイクイップメント（株）
大阪支店 ソフトウェアサービス部 システム第一課
大阪市北区中之島２－２－２　ニチメンビル

「要約」

現在のコンピュータ・システムにＬＡＮ（Ｌｏｃａｌ　Ａｒｅａ　Ｎｅｔｗｏｒｋ）が多く取り入れられるようになりました。これは単に流行といったものではなく、真にその効果が認められているからでしょう。漢字ＤＳＭ－１１、日本語ＶＡＸ／ＤＳＭでは、ＬＡＮをＥｔｈｅｒｎｅｔで実現しました。以下はその概要を御紹介します。

「序論」

最近、情報伝送技術の向上には目をみはるものがあります。

例えば、光ケーブルによる高速ディジタル伝送方式、又Ｅｔｈｅｒｎｅｔ同軸ケーブルによる高速パケット伝送方式等々があげられます。

特にＥｔｈｅｒｎｅｔは、１９８０年９月にＤＥＣ社、Ｉｎｔｅｌ社、Ｘｅｒｏｘ社が共同で開発することを発表した画期的な伝送技術です。

Ｅｔｈｅｒｎｅｔは、オフィス・ビルディングのような環境下における通信手段として、又資源共有手段として設計開発されており、限られたエリア内での高速同軸ケーブルを介してネットワーク下の全資源を利用できます。

このＥｔｈｅｒｎｅｔが漢字ＤＳＭ－１１　Ｖ３．１Ｊでサポートされます。

本文では、このＶ３．１ＪのＥｔｈｅｒｎｅｔに焦点をあて、その全容を御紹介したいと思います。

又、以後の説明のために２つの用語を定義します。

用語１

Ｎｏｄｅ：：＝Ｅｔｈｅｒｎｅｔ同軸ケーブルに接続された通信機器で、
コンピュータ・ワークステーション等がこれに当ります。

Ｓｅｇｍｅｎｔ：：＝物理的に一本の同軸ケーブルで構成される単位

第１章　Ｅｔｈｅｒｎｅｔとは

序論でお話ししました通り、Ｅｔｈｅｒｎｅｔは１９８０年９月にＤＥＣ社、Ｉｎｔｅｌ社、Ｘｅｒｏｘ社が共同で開発することを発表したローカル・エリア・ネットワーク（ＬＡＮ）アーキテクチャです。

Ｅｔｈｅｒｎｅｔの語源は、Ｅｔｈｅｒ［ｉ：θəｒ］、つまりエーテルからきています。

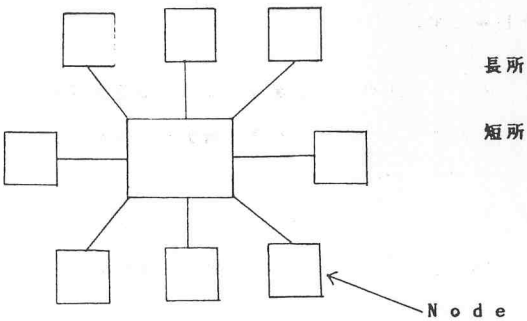
エーテルは、光を波と考えた時、その波を伝える媒質として仮想的に考えられた物資です。当然現在はこの仮説は覆えされていますが、光速伝送のごとく情報を高速伝送する意味からこの名前が付けられました。（参考文献１）

ここでＥｔｈｅｒｎｅｔの概念をお話しするためには、ＬＡＮ（Ｌｏｃａｌ Ａｒｅａ Ｎｅｔｗｏｒｋ）について述べる必要があります。

ＬＡＮは、１つのビルディング・オフィスの規模を単位とし、複数のコンピュータをケーブル上に接続し、これらのコンピュータの間で高信頼性、高速性の情報伝送を行うコンピュータ通信ネットワークのことです。

ＬＡＮのネットワーク構成から分類すると、次の３種類が考えられます。（参考文献２）

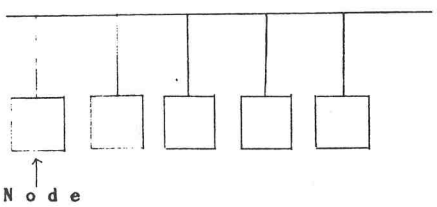
１）スター型



長所：管理しやすい。

短所：ワークステーションの増大に伴い、ケーブル長が増大する。

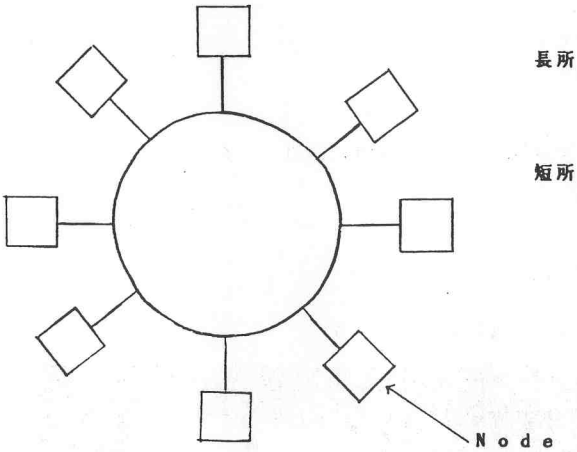
２）バス型



長所：ケーブルが短かくて良く、コンピュータの増設が容易。

短所：送信度数が増すと、衝突が増えて応答時間が増大する。

3）リング型



長所：送信度数が増しても、衝突が
ないのでスループットが良い。

短所：リング上のコンピュータの故障が、
他のコンピュータに与える影響が強く、
リング全体のダウンにつながりやすい。

ＤＥＣでは、この２）のバス型を採用しており、このバス型方式にＥｔｈｅｒｎｅｔの名が付けられました。

Ｅｔｈｅｒｎｅｔの特性を箇条書きにしますと次の通りです。（参考文献３）

- Topology - Branching bus
- Medium - Sheilded coaxial cable, Manchester encoded digital base - band signaling
- Data Rate(Physical Channel) - 10 million bits per second (maximum)
- Maximum Separation of Nodes - 2.8 kilometers(1.74 miles)
- Maximum Number of Node - 1,024
- Network Control - Multiaccess - fair distribution to all nodes
- Access Control - Carrier Sense, Multiple Access with Collision Detect(CSMA/CD)
- Packet Length - 64 to 1518 bytes(includes variable data field of from 46 to 1500 bytes less 8-byte preamble).

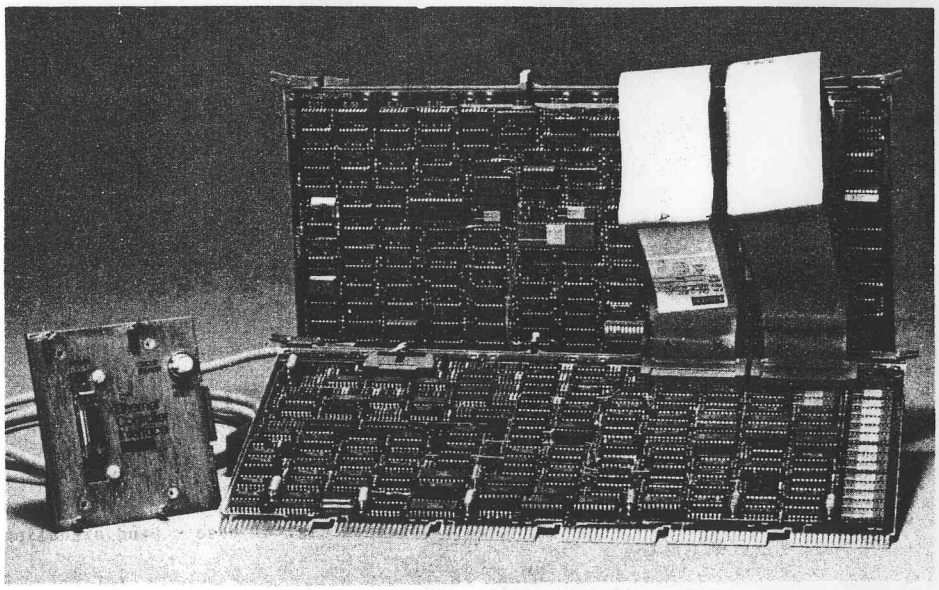
第２章　Ｅｔｈｅｒｎｅｔをサポートするハードウェア

漢字ＤＳＭ－１１でのＥｔｈｅｒｎｅｔをサポートするハードウェアで主要なものとしては次の製品があげられます。

この章では、これらの製品の役割とその特性について簡単に説明します。

１）ＤＥＵＮＡ（又はＤＥＱＮＡ）インターフェース

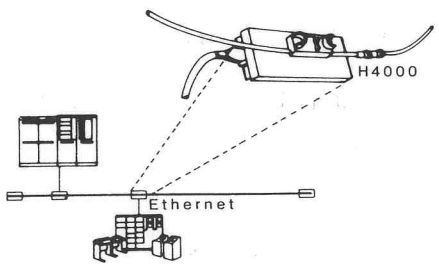
ＵＮＩＢＵＳ－Ｅｔｈｅｒｎｅｔ通信コントローラ



２）Ｈ４０００　Ｅｔｈｅｒｎｅｔトランシーバ

Ｅｔｈｅｒｎｅｔ同軸ケーブルとの接続を行う装置です。

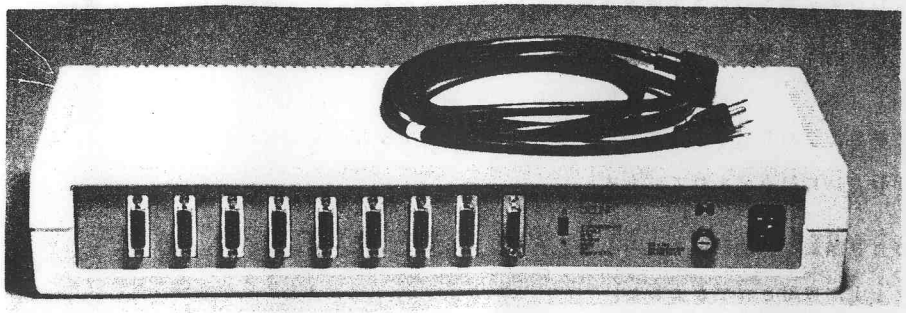
非割込式のタッピング機構の結果、Ｅｔｈｅｒｎｅｔケーブルを切断せずに接続できます。

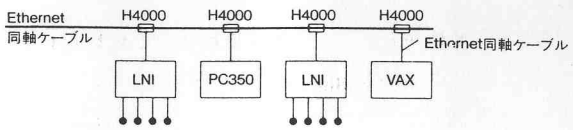


H4000トランシーバ

3) DELNI

ローカルネットワークインタコネクト





接続LNIネットワーク

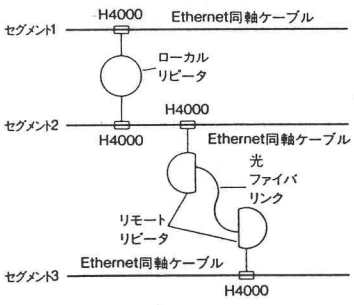
4) Ethernet同軸ケーブル

システムの接続に使用する信頼性の高い５０Ωのシールドケーブルです。

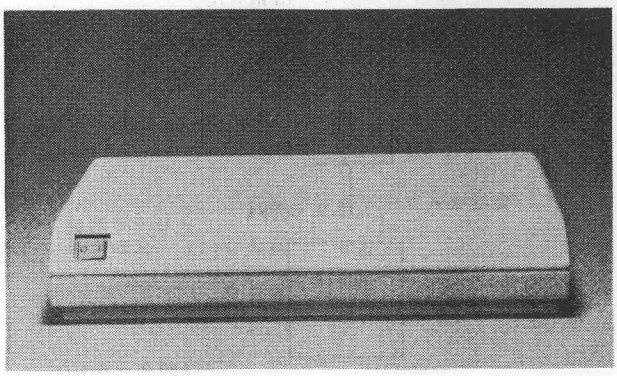
ケーブルは最大５００mの長さのSegmentで設置できます。

5) Ethernetリピータ

ネットワークの拡張のため、複数のEthernet同軸ケーブルのSegmentを接続するテーブルトップ式の装置です。



Ethernetリピータ



第３章　漢字ＤＳＭ－１１でのＤＤＰ機能とＥｔｈｅｒｎｅｔの関係

ＤＳＭはその誕生時の基本理念から、ＴＳＳによる分散型データベースを、より簡単に構築できることが考えられていました。

その結果、ＤＤＰ（Ｄｉｓｔｒｉｂｕｔｅｄ　Ｄａｔａｂａｓｅ　Ｐｒｏｃｅｓｓｉｎｇ）機能がＤＳＭに組み込まれています。

このＤＤＰ機能を可能にするハードウェアインターフェースに従来からＤＭＣ１１、又はＤＭＲ１１が採用されてきました。

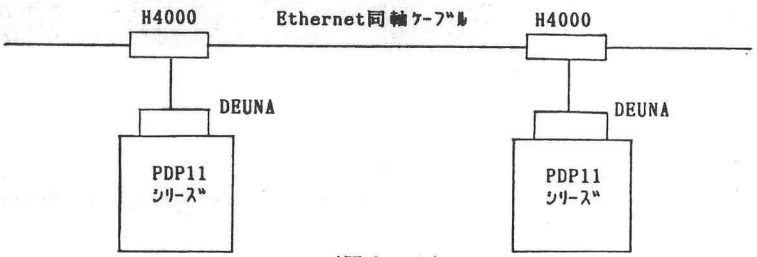
ＤＭＲ１１でのハードウェア特性から、データ伝送速度は、１Ｍｂｉｔｓ／ｓｅｃが最大となります。

ＴＳＳで分散型データベースをアクセスするには充分な速度ですが、より大量のデータを転送する場合には、これ以上の高速性が要求されてきます。

第２章「Ｅｔｈｅｒｎｅｔをサポートするハードウェア」で述べました通り、Ｅｔｈｅｒｎｅｔによる方式では、このＤＭＣ１１、又はＤＭＲ１１にかわり、ＤＥＵＮＡ（又はＤＥＱＮＡ）と呼ばれるインターフェースが使われます。
（図３－１を御参照下さい。）

従来の方式

Ｅｔｈｅｒｎｅｔによる方式



（図３－１）

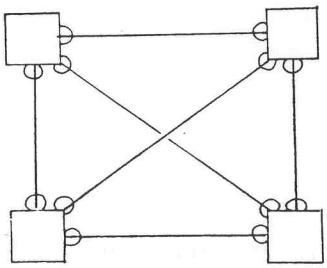
この方式ですとＥｔｈｅｒｎｅｔ同軸ケーブル間を１０Ｍｂｉｔｓ／ｓｅｃの転送を可能とします。（但し、Ｈ４０００トランシーバからＤＥＵＮＡに至る間では、１．２Ｍｂｉｔｓ／ｓｅｃの転送速度となります。）つまり、Ｅｔｈｅｒｎｅｔ方式によって、約１０倍もデータ量を多く転送することを可能にしたと言えます。

次に、ＣＰＵ間の接続方式の面でも、格段の差でＥｔｈｅｒｎｅｔ方式の方が柔軟性に富んでいると言えます。第５章「漢字ＤＳＭ－１１でのＥｔｈｅｒｎｅｔ方式によるネットワーク構築」で詳しく述べますが、ＤＭＲ方式では不可能であったＭｕｌｔｉ－Ｄｒｏｐ接続方式がＥｔｈｅｒｎｅｔ方式では可能となります。

従来のＤＭＲ方式は、Ｐｏｉｎｔ－ｔｏ－Ｐｏｉｎｔ接続方式と呼ばれ、一つのＤＭＲに接続される相手は一つと限られていました。このＰｏｉｎｔ－ｔｏ－Ｐｏｉｎｔ接続方式の欠点は、ＣＰＵの数が一つ増す毎にｎ本（ｎは既接続ＣＰＵ台数）の接続ケーブルを設置しなければなりません。

これに対し、Ｅｔｈｅｒｎｅｔ方式によるＭｕｌｔｉ－Ｄｒｏｐ接続方式ですと、ＣＰＵの数が増しても、Ｅｔｈｅｒｎｅｔ同軸ケーブルとの接続ケーブル１本だけで済みます。これは、あたかも既に張り巡らされた電線から分岐して我が家に電気を供給している姿に似ています。図３－２でこれらのことをまとめてみます。

・Ｐｏｉｎｔ－ｔｏ－Ｐｏｉｎｔ接続方式



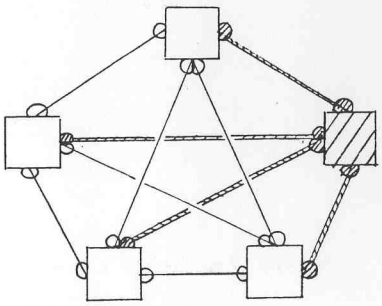
４台のＣＰＵの場合

公式 $\frac{n(n-1)}{2}$ から

６本の接続ケーブルが必要

○の数　１２

----------〉
１ＣＰＵ増設



５台のＣＰＵの場合

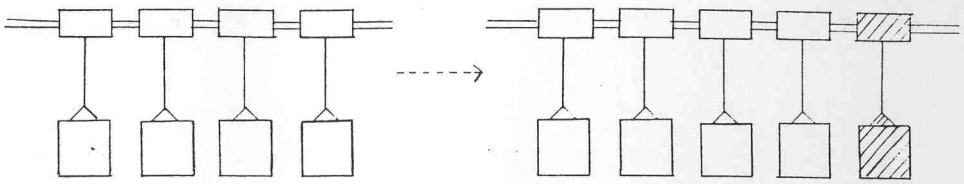
公式 $\frac{n(n-1)}{2}$ から

----------〉 １０本の接続ケーブルが必要

接続ケーブル
４本増加　○+◎の数　２０

◎の数　８

・Ｍｕｌｔｉ－Ｄｒｏｐ接続方式



記号説明

- 既ＣＰＵ
- 増設ＣＰＵ
- 既ＤＭＲ
- 増設ＤＭＲ
- 既ＤＥＵＮＡ
- 増設ＤＥＵＮＡ
- 既Ｈ４０００トランシーバ
- 増設Ｈ４０００トランシーバ
- ＤＭＲ接続ケーブル
- Ｅｔｈｅｒｎｅｔ接続ケーブル
- ＤＭＲ接続ケーブル増設分

図３－２

このように優れたＥｔｈｅｒｎｅｔをアクセスする言語面ですが、既存のＤＭＲ方式によるものとＥｔｈｅｒｎｅｔ方式によるＤＤＰシンタックスとの間は１００％互換性があり、全くそのままのプログラムでＥｔｈｅｒｎｅｔ方式に切り換えることができます。

この結果、ＤＭＲ方式とＥｔｈｅｒｎｅｔ方式を混在させることも当然可能となる訳です。

第４章　漢字ＤＳＭ－１１でのＥｔｈｅｒｎｅｔ接続方式の制限

第３章で述べたＥｔｈｅｒｎｅｔ接続には、次のような制限事項があります。

１）最大Ｎｏｄｅ数

ＣＰＵ台数と考えれば良く、最大１０Ｎｏｄｅです。

２）ＤＭＲとＤＥＵＮＡの混在数

第３章でも述べた通り、ＤＭＲとＤＥＵＮＡは混在可能ですが、１ＣＰＵ当り最大４枚迄となります。

３）Ｅｔｈｅｒｎｅｔ同軸ケーブルの物理最大長

同軸ケーブルの物理最大長は５００ｍ迄です。
従って、これ以上の距離を必要とする場合は、リピータが必要となります。詳しくは、第５章で触れます。

４）Ｎｏｄｅ間の最小長

ＮｏｄｅとＮｏｄｅの間は、少なくとも２．５ｍ以上は間隔をあける必要があります。

５）Ｎｏｄｅ間の最大長

ＮｏｄｅとＮｏｄｅの間は、１５００ｍ以上間隔をあけることはできません。

６）トランシーバとコントローラ間の最大長

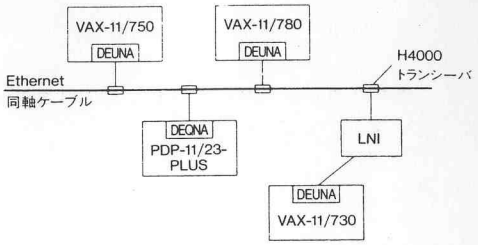
Ｈ４０００トランシーバとＤＥＵＮＡコントローラの間は、最大５０ｍです。

第５章　漢字ＤＳＭ－１１でのＥｔｈｅｒｎｅｔ方式によるネットワーク構築

最後にこの章では、数々の特性を有するＥｔｈｅｒｎｅｔの接続方法で代表的なものを数種類あげて説明します。

１）一本の同軸ケーブルによる接続

Ｅｔｈｅｒｎｅｔ同軸ケーブルは、物理最大長が５００ｍ迄となっています。従って、この長さの範囲内で一本の同軸ケーブルで各ＣＰＵを接続しようとするものです。



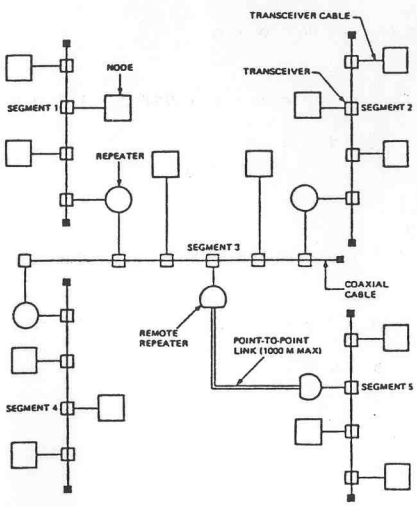
この一本の同軸ケーブル（つまり　１　Ｓｅｇｍｅｎｔ）内とは、最大１０　Ｎｏｄｅ（ハード上では、１００　Ｎｏｄｅ）迄接続できます。

２）２本以上の同軸ケーブルによる接続

コンピュータが設置されている建物が少し離れた所に点在し、その間を接線する場合は、第２章「Ｅｔｈｅｒｎｅｔをサポートするハードウェア」の５）Ｅｔｈｅｒｎｅｔリピータを使用します。

Ｅｔｈｅｒｎｅｔリピータは、ローカルリピータとリモートリピータがあります。

リモートリピータの間は、光ファイバーケーブルで接続されます。（但し、リピータ間は最大１ｋｍ迄です。）



参考文献

１）小学館発行「大日本百科事典ジャポニカ」
２）高木　英明著「トークン・リング方式によるＬＡＮ」共立出版株式会社　ｂｉｔ　1984年3月号
３）「ＤＥＵＮＡ　ＵＳＥＲ'Ｓ　ＧＵＩＤＥ」ＤＥＣ社製　発注番号 EK-DEUNA-UG-001

Ⅰ — Ⅰ — 1　第12回日本マンプスユーザーズグループ学術大会（1985年8月）

M／VXについて

○吉田欣弥，石丸径美，大垣伸二

三井造船株式会社システムエンジニアリング事業部
東京都中央区築地5－6－4

M／VXのベンチマークテストを行い、その結果を考察した。
VAX／DSMに比してかなりパフォーマンスの良いことが判明した。

MUMPSは、それ自身でOS，言語，DBの3つの機能を持つており、PDPシリーズのような優秀なミニコンには最適なシステムである。利用される分野も広い。しかし最近では、FORTRANやCOBOLも同時に使いたい、あるいは、GRAPHIC機能も使いたい、あるいはLANもとニーズが多様化してきており、従来のままではUNIX等の汎用機能型OSに遅れをとりつつあるのも否めない現状である。

ハードウエアも次々とLSI化されてきており、小型化が進み、VAXシリーズもMICRO VAX-Ⅱなど魅力的なマシンが誕生してきている。そこで、このようなハードにMUMPSを使いたいというケースが生じるのは、当然の理である。DEC社はDSMを、VMSのもとでインタプリテイブ言語として使つている。これにより、上記のニーズはほとんどカバーされ、応用分野も広がつている。

しかしながら、そのパフォーマンスはPDPにDSMを用いた場合よりも遅くVAXとしての効果が得られない。そこでMUMPSをインタプリテイブ言語としてではなく、コンパイラー言語として使えないかという要求が生じ、米国のINTERSYSTEMS社により開発されたものがM／VXである。

M／VXがVAX／DSMに比してどのくらいパフォーマンスが良いかを調べる為に、当社では下記の条件にてベンチマークテストを行つた。

使用機種：　VAX−11／750　2MB
OS　：　DSM V.2，M／VX
場　所：　三井造船 東京本社
期　日：　昭和60年8月1日
テストプログラム：　コマンド及びファイルI／O，107項目
総処理時間：　約60分

結果は次の表のとおりであり、上段がDSM，下段がM／VXの所要時間である。単位が秒なので、1秒に満たない場合は4捨5入されて、0又は1となり区別が難しい面もあるが、かなり正確な値が得られている。

次の５つの場合について、テストの内容を示す。

Ⓐ（試験番号１１３）　　１倍　　１３０文字のセットとキル

Ⓑ（　〃　１２０）　　４２倍　　単純グローバルセット

Ⓒ（　〃　１５１）　　３倍　　グローバルの$Ｄ関数

Ⓓ（　〃　３６）　　２倍　　Ａ＞１００　比較

Ⓔ（　〃　８９）　　３倍　　グローバルのセット（レベル１０まで）

Ｍ／ＶＸはＤＳＭに比して何倍速いと断言するのは、ベンチマークテストの種類にもより異なるので難しい。しかし、アプリケーションの作り方によりかなりの差があることが歴然とした。

Ｍ／ＶＸは漢字処理が未完成なので、やや応用面で制限が生じるが、必要ない場合は非常に効果があり、当社でも安全性試験システムMiTOXに使つている。

VAX／DSM	1	2	12	13	13	15	16	1	1	1	1	1	2	3	2	27
M／VX	1	1	4	2	2	2	4	0	1	0	1 Ⓐ	1	0	2	1	7
	85	85	86	9	84	10	83	25	3	6	1	2	16	18	0	0
	3	2 Ⓑ	2	1	2	1	2	11	2	3	0	0	4	5 Ⓒ	0	0
	1	0	1	1	1	1	1	1	14	2	15	2	17	2	17	2
	0	0	0	0	1	0	1	0	4	0	5	0	5	0	6	—
	79	2	2	2	3	2	2	11	3	3	2	3	3	3	2	3
	—	—	—	2	2	2	1	4	2	2	1	2	2	2	1	2
	3	3	4	3	3	4	5	5	5	1	5	6	3	3	3	4
	2	1	2	2	2	2 Ⓓ	1	2	1	2	2	2	2	2	2	2
	3	4	5	5	2	8	8	5	5	16	3	2	3	2	6	15
	2	2	3	2	2	4	4	3	2	6	2	2	2	1	3	3
	14	15	15	15	17	18	19	22	28	9	13					
	3	4	3	3	4	5	5	6	9 Ⓔ	3	2					

Ⅰ — Ⅰ — 2　第12回日本マンプスユーザーズグループ学術大会（1985年8月）

コンパイラ型MUMPSを用いたシステムの導入例紹介

○ 石 丸 径 美

東京都中央区築地5－6－4
三井造船株式会社　システムエンジニアリング事業部

ＤＥＣ社のMUMPS DSM−11V2.0ARを用いて構築されていたアプリケーションシステム（弊社製パッケージシステム「安全性試験システム」）を、Intersystem社のコンパイラ型MUMPS M/11＋ にコンバージョンした際の手順および経過について報告する。尚、当システムはM/11＋ が会話モードでのDMR−11（高速通信制御装置）をサポートしていないため、現在まだ実業務としては稼動していない。

1. M／１１＋を採用するに致つた理由

① アプリケーションシステムの特徴としてある一時に多量のバッチ処理が稼動するため、その時期オペレータの負担が大きい。

② ホストコンピュータPDP 11/24を上位機種に変える余裕がすぐにはない。

③ 漢字は当面使わない。

2. 主たるコンバージョン内容

① プログラムの分割

M／１１＋では実行はすべてオブジェクトコードであり、それはソースコードに比較して３０～５０％容量がふえる。またオブジェクトコードは8KB 以内という制限があるためコンパイル後オブジェクトコードが8KB 以上となるルーチンについてはその分割が必要となつた。従つて、DSM でほぼ5KB 以上の容量のルーチンについてはその分割を行なつた。

② コマンドおよび関数の変更

(1) 無条件に変換したもの（ユーテイリテイプログラムの利用）

・$ZSORT→$ORDER ,　$ZERROR→$ZTRAP

・ZJOB→JOB

・W ＊13 → W $C(13) …… ラインプリンタのみ

・W # → W #,$C(13) …… ラインプリンタのみ

・U 3 → U 3:132 …… 起動端末のマージンとなるため

(2) 変換の際，調査を要したもの

・VIEW , $VIEW

使用が許されるのはマネージャＵＣＩからのみであつたため使用している部分をマネーヤＵＣＩ下に％付きのルーチンとして作成した。またシステムテーブルに関係している内容についてはその違いによる調整を行なつた。

・OPEN , USE

パラメータ付きの場合の違いによる調整を行なつた。

・WRITE

MTのコントロールコード，$X，$Yとの関係があるカーソルの動きの違いによる調整を行なつた。

③ 機能がないため削除

ZUSE , $ZNEXT

④ コメントの表記法変更

M／11十ではコンパイルの際にコメントはオブジエクトコードにおとさないため $TEXT を用いてその行を実行に用いている場合、ソースコードがないと正常作動しなくなる。このため、コメントの「；」を「；；」に変更した。

3. コンバージョンに要した時間

約800本のプログラムで約2ケ月（50日）程要した。

4. M／11十の長所・短所

① 長　所

(1) グローバルのセーブ，リストアが早い。

ノードを指定しないグローバルのセーブ，リストアはDSMと比較し非常に早い。これは、ノードを順次追う方式ではなく、デイスクブロツクをそのまま扱かう方式のためである。約2MBの容量のグローバルを30秒弱でセーブ出来る。

(2) 各ラインごとにコントロールシーケンスが設定できる。

端末のコントロールシーケンスが機種によつて異なる場合などもアプリケーションで対応する必要がない。

② 短　所

(1) DMR－11の会話モードをサポートしていない。

複数のMUMPS システム間はDDP機能で対応できるが、他のOSたとえばRSXなどとのデータのやりとりがDMR－11を介しては出来ない。

(2) システムジエネレーションが頗雑である。

DSMのようにSGの内容をグローバルファイルに持つてはいないため、オンラインでのSGができず、時間もかかる。（DSM－11V1.0 に似ている。）

5. 今後の予定

DMR－11をDL－11に変えて実業務稼動する計画である。

Ⅰ — Ⅰ — 3　第12回日本マンプスユーザーズグループ学術大会（1985年8月）

U－MUMPSの機能強化

小林　勝、　久江　正、○上戸　隆、　阪倉　明

住友電気工業株式会社ME開発室
〒554　大阪市此花区島屋1丁目1番3号

筆者らは、昨年、住友電工製ワークステーションUSTATION上で稼動するマルチユーザMUMPS（U－MUMPS）を開発し、報告を行なった。[1]その後、種々の実用システムを納入するとともに、下記の機能強化を行なった。①稼動機種の拡大：UNIX SYSTEMV版MUMPSの稼動　②パソコンMUMPS開発：MS－DOS版MUMPSの稼動　③DDP機能開発：USTATION間の分散データベース処理の実現　④日本語処理機能の充実：各種漢字端末の接続サポート，ユーティリティメッセージの日本語化。

1. はじめに

U－MUMPSは、下記の特徴を有している。

(1) 1984年ANSI標準に準拠し、日本語処理もサポートしている。

(2) UNIXOSというマルチ言語環境下において稼動する。

(3) プリコンパイル方式等の採用により、処理速度の向上を図っている。

(4) プログラム開発保守・システム運用管理のための豊富なユーティリティライブラリを持つ。

今回、筆者らは、U－MUMPSに対して、さらに使い易く、柔軟なシステムにするため、①稼動機種の拡大　②パソコンMUMPSの開発　③分散データベース処理機能の開発　④日本語処理機能の充実といった機能強化を行なったので、ここに報告する。

2. 稼動機種の拡大

UNIX SYSTEMⅢ Versionで稼動するUSTATION E－10用U－MUMPSに加え、SYSTEMV Versionの上位機種E－15，下位機種E－5用U－MUMPSを開発し、機種の拡大を行なった。

当社のベンチマークテスト結果では、従来機種のE10の処理速度比を1とした場合、E－5では約1.0，E－15で約2.0である。さらに、今秋稼動予定の32ビットマイクロプロセッサ68020をベースにしたE－20では約4.0～6.0の処理速度比を有すると推定される。

表1に、USTATION シリーズに対するU－MUMPSのシステム構成を示す。

表1. USTATIONシリーズに対するU−MUMPSのシステム構成

項目＼機種	USTATION/E5	USTATION/E10	USTATION/E15	USTATION/E20
使用プロセッサ	MC68000	MC68000	MC68000	MC68020
クロック	10MHz	10MHz	12.5MHz	16.6MHz
キャッシュメモリ	無し	無し	有り(8kバイト)	有り(16kバイト)
最大実装可能メモリ	4Mバイト	8Mバイト	8Mバイト	32Mバイト
スロット数	6	14	14	18
サイズ(H,W,D,mm)	180×540×500	700×400×700	700×400×700	700×557×940
外部記憶(ディスク)(アンフォーマット)	5インチ 51Mバイト	8インチ 80Mバイト 拡張ディスク80Mバイト または 160Mバイト	8インチ 80Mバイト または 160Mバイト 拡張ディスク80Mバイト または 160Mバイト	8インチ 160Mバイト または 340Mバイト 拡張ディスク160Mバイト または 340Mバイト
フロッピディスク	8インチ 1Mバイト	8インチ 1Mバイト (拡張フロッピディスク 1Mバイト)	8インチ 1Mバイト (拡張フロッピディスク 1Mバイト)	8インチ 1Mバイト (拡張フロッピディスク 1Mバイト)
U−MUMPSでログオン可能なユーザ数	8	16	16	40
OS	UNIPLUS (SYSTEMV)	UNIPLUS (SYSTEMIII)	UNIPLUS (SYSTEMV)	UNIPLUS (SYSTEMV)
言語(標準)	C, FORTRAN, Assembler	C, FORTRAN, Assembler	C, FORTRAN, Assembler	C, FORTRAN, Assembler

3. パソコンMUMPSの開発

U−MUMPSの開発をもとに、NEC9800シリーズ，N5200モデル05／mKⅡ IBM5550上で稼動するシングルユーザMUMPS(SP−MUMPS)を開発した。表2にSP−MUMPSの仕様を示す。

表2. SP−MUMPSの仕様

対象機種	NEC9800シリーズ、N5200モデル05／05mKⅡ、IBM5550、メモリ384kB以上
フロッピーディスク	5インチ(2DD／2HD)または8インチ(2D)
オペレーティングシステム	日本語MS−DOS
言語	ANSI標準MUMPS(日本語対応)、プリコンパイル方式、論理ツリー型(内部B−tree構造)可変長フィールド
データベース	可変長レコード、スパース構造対応、データ階層設計(親子・構造体反復)可

＊固定ディスクがない場合、フロッピーディスクが2台以上必要です。この場合1MBフロッピーのご使用をおすすめします。

SP−MUMPSは、U−MUMPSのシングルユーザ版であり、プリコンパイル，ディスクキャッシング等の処理を高速にするための手法を用いて、ハイパフォーマンスなパーソナルユースのMUMPSを実現している。詳細は本大会のデモセッション「SP−MUMPS」に述べる。[2)]

4. 分散データベース処理（ＤＤＰ）機能の開発

分散データベース処理（DDP）機能は、自システムから、他システムのグローバルにアクセスする機能である。U－MUMPSでは、各システム間のスター型接続をサポートしており、そのＤＤＰシンタックスは、下記のとおりである。

```
^[ UCI , SYS ]GLOBAL
```

但し、UCI　　：アクセスするグローバルのＵＣＩ名
SYS　　：グローバルが存在するシステム名
GLOBAL：グローバル名称

例　システム“E15”上のＵＣＩ“MGR”内のＦＩＬＥ（１，２）というグローバルの参照

```
S  B=^[ “MGR”,“E15” ]FILE(1,2)
```

ＳＥＴコマンドに加えて、ＫＩＬＬ，ＲＥＡＤ，ＷＲＩＴＥコマンドや、$NEXT，$ＤＡＴＡ関数にも、ＤＤＰシンタックスは使用可能である。ＬＯＣＫコマンドのサポートは開発中である。

ＤＤＰ機能を用いた納入システムの構成例を図１に示す。生化学検査・血液検査・病棟・人間ドックの対象業務をＵＳＴＡＴＩＯＮ　Ｅ１０，Ｅ１５の２システムで実現し、病棟から検査データをＤＤＰ機能を用いて検索を行なっている。

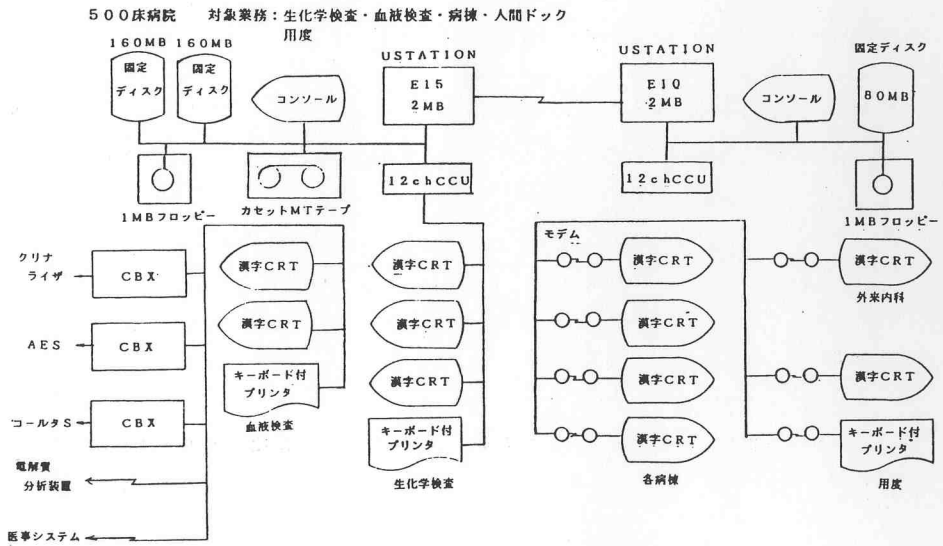


図１　ＤＤＰ機能を用いたシステム構成例

5. 日本語処理の充実

より扱い易い日本語処理ＭＵＭＰＳをめざして、下記の２項目の充実を図った。

(1)　漢字端末に対する各種パラメータの指定：漢字端末に対して、①漢字コードタイプ（ＪＩＳ漢字コード，シフトＪＩＳ漢字コード，ＤＥＣ漢字コード）、②漢字シフトイン／シフトアウトコード、③漢字サイズ（半角文字に対する全角文字の文字幅）をシステムジェネレーションおよびＯＰＥＮ／ＵＳＥ　コマンドにおける引数として指

定することで、その設定を可能としている。①，②は各種漢字端末の接続を可能とし、③は書式制御（タブレーション）を漢字に対しても有効に動作させるために必要なパラメータである。漢字端末登録の例を図２に示す。

装置番号	装置タイプ	装置名	ライトマージン	初期化パラメータ	属性
1	PC/DOS	CON	0	Parallel	/エコー /漢字出力可 /漢字入力可 /シフトJISコード /半角x2
3	PC/DOS	LPT1	132	Parallel	/出力専用 /漢字出力可 /JISコード /シフトイン=1B4B /シフトアウト=1B4E /半角x1.5
4	PC/DOS	COM1	255	1200,8B+1S	/エコー /漢字出力可 /漢字入力可 /DECコード /シフトイン=1B2440 /シフトアウト=1B2848 /半角x1

注　シフトＪＩＳコードの場合は、漢字シフトイン／シフトアウトコードは不要
　　ＪＩＳ漢字，ＤＥＣ漢字コードの場合は必要

図2. 漢字端末の登録例

(2)　エラーメッセージ・ユーティリティの日本語化：例に示すように、ＳＰ－ＭＵＭＰＳでは、各種メッセージを日本語にて出力する。また主要ユーティリティのメッセージを日本語化した。

例
<INRPT> 割込み要求が発生しました
<UNDEF> 定義されていない変数が参照されました
<SYNTX> シンタックスが間違っています

6. おわりに

Ｕ－ＭＵＭＰＳを、ＵＳＴＡＴＩＯＮシリーズおよびパーソナルコンピュータに装備することで、稼動機種の幅が広がり、システムの規模に応じた構成の選択が可能となった。また、日本語処理機能の充実により、よりユーザフレンドリーな処理系に近づいた。

今後の展望としては、当社の開放型光ＬＡＮシステムＳＵＭＩＮＥＴ３３００やパソコンネットＳＵＭＩＮＥＴ３２００等を利用した、より有機的なＭＵＭＰＳネットワークの実現や、ユーザのより容易なプログラミングを可能とするＭＵＭＰＳソフトウエアツールの開発を行なっていく計画である。

〔参考文献〕

1）小林　勝、久江　正、上戸　隆：ＵＳＴＩＯＮ上のＭＵＭＰＳの開発，Ｐ.11～Ｐ.14、第１１回　日本ＭＵＧ学術大会，１９８４年１２月

2）小林　勝、久江　正、米田　研、阪倉　明：ＳＰ－ＭＵＭＰＳ，第１２回　日本ＭＵＧ学術大会，１９８５年８月

Ｉ — Ⅱ — １　　第１２回日本マンプスユーザーズグループ学術大会（１９８５年８月）

マルチウインドウ機能をもった日本語ＭＵＭＰＳ

鈴木　利明

アレフ　コンピュータ

〒440　豊橋市花田三番町９４番地

ＭＵＭＰＳにマルチウインドウ機能を付け加えると、画面の表示がシンプルになり対話プログラムが書きやすくなる。仮想画面毎に書くプログラムは、画面を単位としたモジュールを構成する。

ユーザは、必要な情報が目の届く範囲にあるので対応が速く出来る。

ＭＵＭＰＳの特徴は、データベース処理能力が極めて高いこと、文字列処理の強力なこと、そして、対話プログラムの作りやすいことなどであるます。

対話プログラムにおいて、近年ワークステーションを中心にマルチウインドウという手法が広がっています。

マルチウインドウは、多種類の情報を欲しいときに一度に提供することが簡単に出来ます。

マルチウインドウを使用するのに便利な具体例は、

Ａ　　プログラムの比較

Ｂ　　多数のルーチンを一度に参照したいとき　（エディタ）

Ｃ　　ひとつの画面で多数のジョブを管理するとき

Ｄ　　通信の様に、外部から割り込んで来るメッセージを現メイン処理中に表示したいとき。これが出来ないと手紙と電話の差がつきます

Ｅ　　デバッグ時に画面を壊したくないとき

Ｆ　　主処理から補助画面処理を呼び出すとき。

例、ＨＥＬＰメッセージ、過去の入力データ参照、メニュウ表示

などがあります。

特に、Ｆのケースは、よく使用されます。これを今までのＭＵＭＰＳで行なったときは、次の様になります。

補助画面処理は、メイン画面の一部もしくは全部を消して補助メッセージを表わし、入出力を行ないメイン処理に戻りメイン画面をもう一度表示しなおします。

このため、画面を元に戻すための情報を変数などに持つ必要があり、表示しなおすプログラムも必要としていました。

これは、”いつでもどこででもＨＥＬＰをおこなうという”、ことを難しくしていたし、メニュウ方式の場合は、切り換え時間を長くしていた。

Ｆをマルチウインドウで行なうと、次の様になります。

補助画面処理は、まず自分専用のウインドウを開いて、補助メッセージを表わし、入出力を行ない、離脱時に自分のウインドウを閉じてメイン処理に戻り次の命令から再開する。

ウインドウを閉じるときに画面がメインに戻るため、メイン処理において画面を表示しなおす必要はありません。ここはいままでとは大きく異なります。これによりプログラムも変数も少なくて済みます。

マルチウインドウでは、いくつでもウインドウを開いて多種類の情報をユーザに与えることが出来ます。また、ウインドウの生成と消滅はプログラムで出来ますので必要なものは長く、いらないものはすばやく消滅させることができます。

システムでは、ウインドウを開くときに、画面上にエリアを確保してその下になる画面情報は待避しておきます。閉じるときには、待避してあった画面情報などを参考にして元の画面に戻します。

日本語表示に伴い注意したのは、カーソルは文字の先頭に来ること、全角文字を不連続にしないこと、半角と全角文字の混在が出来ることです。

ＡＬＥＦ日本語ＭＵＭＰＳの仕様については、現在日本語ＭＵＭＰＳ標準化委員会が精力的に標準化作業を推し進めていますので、ここでは控えさせて頂きました。

マルチウインドウに伴い拡張したものは、次のものです。

```
OPEN n:(VX:VY)              n：仮想画面番号
                           ＶＸ：仮想画面の横幅
                           ＶＹ：仮想画面の縦幅
USE n:(ORDER:RX0:RY0:VX0:VY0:RX:RY)
                              n：仮想画面番号
                        ＯＲＤＥＲ：画面表示の優先順序
                           ＲＸ０：実画面上の表示開始位置Ｘ０
                           ＲＹ０：実画面上の表示開始位置Ｙ０
                           ＶＸ０：仮想画面上の表示開始位置Ｘ０
                           ＶＹ０：仮想画面上の表示開始位置Ｙ０
                             ＲＸ：表示横幅
                             ＲＹ：表示縦幅
CLOSE n                       n：仮想画面番号
```

最後にマルチウインドウの短所は、表示にかかる絶対処理量が単純表示のときよりもはるかに多いためスピードがおちることです。これを避けるためには、速いアルゴリズムの開発が必要です。

現在のＡＬＥＦのマルチウインドウは、１００ｍｓ単位のタイムスライスでウインドウ表示を管理しています。

システム紹介

MUMPS下で実現したニューメディアシステム

山口光大，柿崎賢一
(株) 高岳製作所
システム事業部
東京都千代田区

1. 要約

本システムは、(財) 電力中央研究所殿に納入した高度経営情報システム（DEMANDS）であり、各電力会社のパイロットモデルとして位置付けられている。この中でMUMPS下でビデオテックス画面等のデータベースを管理し、LANを通して様々なメディアを提供している。

2. 特徴

本システムの特徴は、下記のとおりである。

(1) CATV方式LANを使ってビデオ情報，ビデオテックス情報，文字情報を1本のLANを通して伝送できること。
(2) MUMPSの長所を生かし、ビデオテックス画面をコンパクトにデータベース化したこと。
(3) 1台の端末でビデオ情報，文字情報，ビデオテックス情報を表示できること。

3. システム構成概要

本システムは、大別して下記の2つのサブシステムとワークステーションからなる。

(1) ビデオイメージファイルサブシステム
(2) フレームクリエーションサブシステム
(3) ビデオテックスワークステーション
(4) スタッフワークステーション

図1にシステム構成の概要を示す。

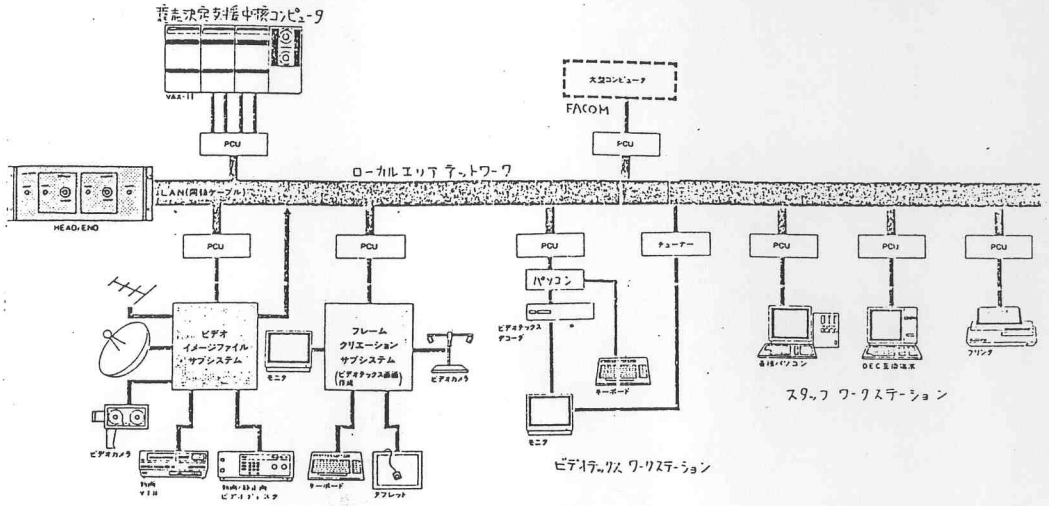


図1　システム構成図

4. ビデオイメージファイル　サブシステム

本サブシステムは、VTR，光動画ファイル（NTSCにて書き込みが可能なタイプで、動画13分20秒，静止画で24000コマ利用できるもの）を複数台接続し、端末より任意の動画・静止画をMUMPS下で選択可能にしたもので、図2に構成図を示す。

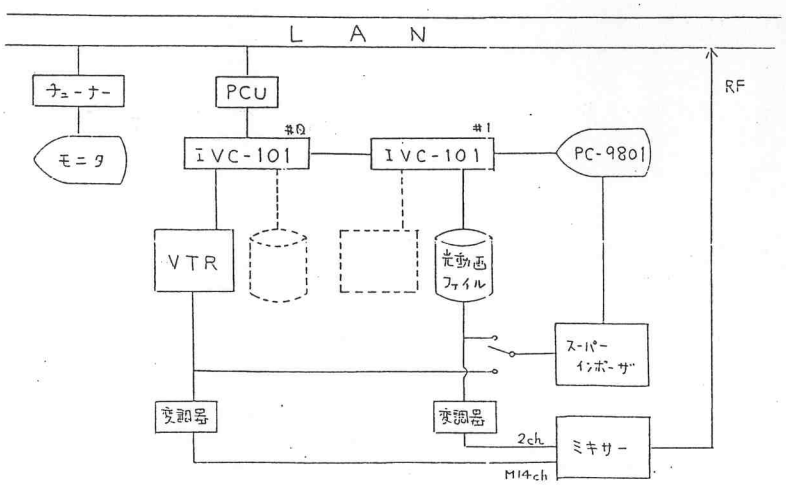


図2　ビデオイメージファイルシステムの構成

4-1. インテリジェント・ビデオ・コントローラ（IVC-101）

ここで、このサブシステムの中核となるインテリジェントビデオコントローラ（IVC-101）について紹介しておく。

このIVC-101は、ホストコンピュータから送られてきたデータの中でVTR、光動画ファイル等の映像制御用のコマンドを区別し出力するためのアダプタ機器で、次のような特徴をもっている。

（1） ホストまたはパソコン上のデータベースによる画面検索ができる。
（2） ホスト，パソコン，VTRの機種を問わない。
　　ただし、VTRにはステレオ録音，アフレコ，リモコン機能を有すること。
（3） ホストまたはパソコンからの1ラインにつき、IVC-101を最大4台まで接続できる。
（4） 通常のホスト，パソコン間の通信ができる。

4-2 ビデオ情報データベースの検索・管理

VTR，光動画ファイルのビデオ情報データは、ホストコンピュータにて管理され、次の機能がMUMPS下で制御できる。

（1） ビデオテープのアドレス付け

あらかじめ録画してあるVTR用のビデオテープをアドレス付けする時間を指定することによって0.1秒単位にアドレス付けすることができる。

（2） ビデオ情報アドレスファイルの作成

ビデオ情報の検索に必要なスタートアドレスとエンドアドレスを読み込み、それをファイルに格納することができる。

（3） ビデオ情報の検索

上項で作成されたアドレス情報をもとに、メニュー画面で番号選択することにより任意の映像を見ることができる。

また実際の表示される映像は、コンピュータの文字，図形情報とともにスーパーインポーズして表示することも可能であり、機能保護を考え通常のテレビでは見れないような周波数帯域にして表示することもできる。

5. フレームクリエーション サブシステム

本サブシステムは、次の3つの機能からなっており、ホストコンピュータ上でビデオテックス画面をMUMPS下で管理できるようになっている。

5-1 ビデオテックス画面の作成

図3に示すように、ビデオテックス画面はフレームクリエーションシステムにより ビデオカメラから入力した原画を2048色中16色にディジタル化し、それをもとに 様々なエディタ機能を使って作成し、NAPLPS（North American Presentation Level Protocol Syntax）コードに自動変換することができる。また、タブレットからの任意図形の入力やジオメトリック図形の入力，キーボードからの文字（英数字・記号・日本語等）の入力も可能である。

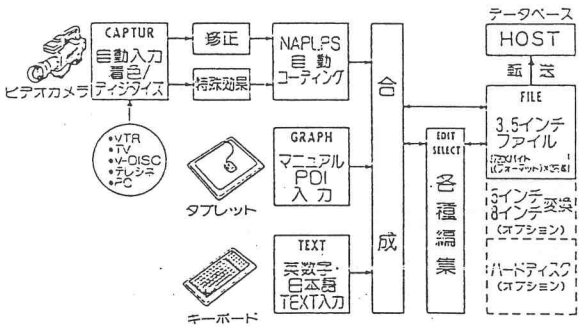


図3 ビデオテックス画面の作成

5-2 ビデオテックス画面の送受信

フレームクリエーションシステムで作成されたビデオテックス画面データをホストコンピュータのMUMPSグローバルデータとして転送したり、逆にフレームクリエーションシステムへ転送して再編集することができる。

5-3 ビデオテックス画面データベースの検索・管理

ホストコンピュータ上のMUMPS下で、ビデオテックス画面の管理を行なうことができ、次の機能がある。

(1) 画面名の登録
(2) 画面体系の登録・削除
(3) 画面情報の削除
(4) 画面のコピー
(5) 画面ディレクトリィ表示
(6) 画面名の変更
(7) 画面検索
(8) 画面表示

5-4 ビデオテックスの構成

ビデオテックスの利用については、図4に示すような構成をとっている。

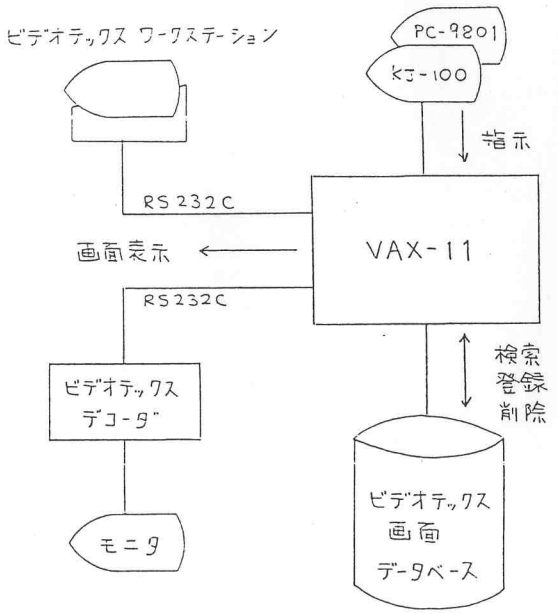


図4 ビデオテックスの構成

6. ニューメディア対応多機能端末

役員向けの情報表示は、コンピュータ情報・TV・ビデオ・ビデオテックスを1台の端末で利用できる多機能端末を通じて行なう。各機能の切替は簡易キーボードでキーを選択することにより可能となっていて、図5にその構成例を示す。

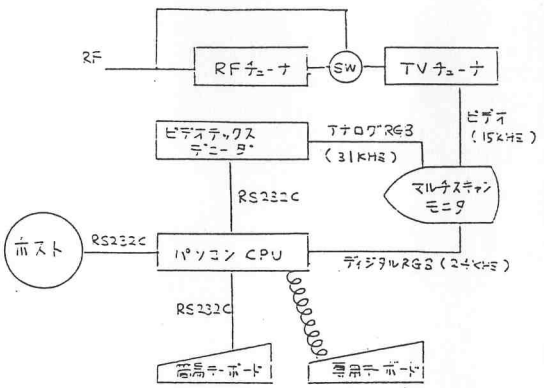


図5 ニューメディア対応多機能端末の構成例

7. あとがき

ニューメディア時代を前に当社MUMPS派はMUMPSシステムの今後の応用の手段としてニューメディア対応を考えており、このシステムを基盤として商品化したwiDE-kTOSを母体としたニューメディア統合情報システムを目指していく方針である。アプリケーション例としては次のものが考えられる。

- 展示場のデモンストレーション
- 企業の紹介
- 役員システム
- 意志決定支援システム
- 医療情報
- カルテ・レントゲン保管
- 新製品紹介
- 不動産の物件案内
- 地図・文献
- 天気・証券・市況情報
- 教育
- 店頭広告・案内・販売
- 結婚式場・観光案内
- 電子メイル・ボイスメイル

最後に本システムの開発にあたり、多大な御指導、御助力さいただいた電力中央研究所各位に深く感謝の意を表する次第である。

システム紹介

(株)高岳製作所・タイムリーシステム

直松郁也，佐々木一郎
(株)高岳製作所
システム事業部
東京都千代田区

1、要約

本システムは、営業活動における情報伝達の迅速化・精度の向上，生産活動における生産性・効率の向上，そして企業の意志決定の支援を主目的として構築された製造業における総合情報管理システムである。

2、特徴

本システムの特徴は、下記のとおりである。

(1) 広域・ローカルのネットワークを利用した分散処理システムであること。
(2) 異機種間結合（IBM，TANDEM，DEC，NCR，パソコン）を実現していること。
(3) 異種プログラム言語［MUMPS（M/11＋，M/11，DSM-11，Micro-MUMPS），COBOL，RPGⅡ）の特徴を生かしシステムを構築していること。

3、システム構成概要

本システムは、大別して下記の3つのサブシステムになる。

(1) 営業システム
(2) 名古屋事業所生産管理システム
(3) 小山事業所生産管理システム

図1は、タイムリーシステムのネットワークを示すものである。

4、営業システム

営業システムは、タイムリーシステムの中核システムとして、主に次の4つの機能より構成されている。

(1) 営業情報サブシステム
オーダエントリ，顧客情報管理，見積作成等の営業活動支援のためのサブシステム。

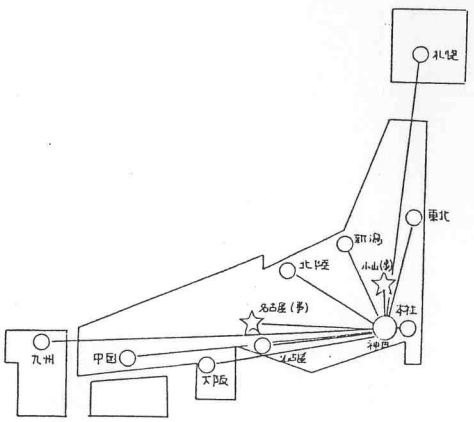


図1　ネットワーク概要図

(2) 経理サブシステム
経理情報の管理を行なうサブシステム。

(3) 総合経営情報サブシステム
全社レベルの情報をもとに企業の意志決定の支援を行なうサブシステム。

(4) 事業所インターフェイスサブシステム
名古屋，小山両事業所の中核システム（TANDEM, NCR）とのインターフェイスをとるためのサブシステム。

これらのサブシステムは、図2で示されるネットワーク上に分散データベースの形で構築されている。
このネットワークの中核は神田分室システムで、その構成は、図3に示すとおりである。

図4は、支社・営業所用のシステムであるμ-TAKACを示すものである。

これらからわかるように、営業システムは、IBMシステムとM/11，M/11+，DSM-11をOSとして使用するMUMPSをベースとするKTOS-11システムの異機種間結合システムとなっている。

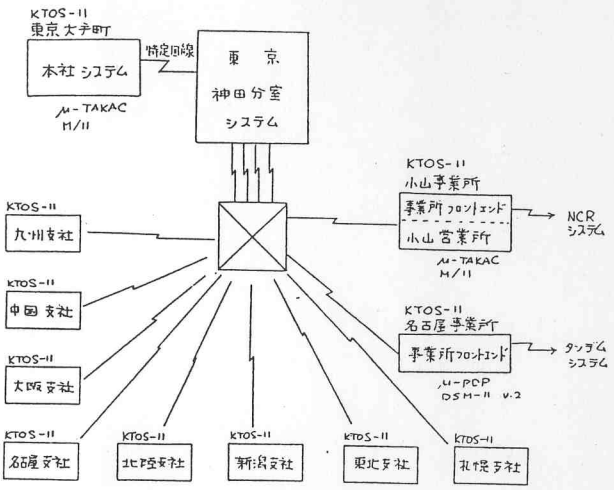


図2 営業システムネットワーク

IBMシステムとKTOS-11システム間の通信データ処理は、図5に示す形態で行っており、通信手順としては日本チェーンストア協会（JCA）が制定しているJCA手順に準拠している。

なお、営業システム内の全てのアプリケーションソフトウエアは、RPGⅢ（IBMシステム）とMUMPS（KTOS-11システム）の2つの言語によって記述されている。

5. 名古屋事業所生産管理システム

名古屋事業所の生産管理システムは、ノンストップシステムであるTANDEMシステムを使用しており、そのシステム構成は図6のとおりである。

図7は、生産管理システムの概要を示している。このサブシステムは、ノンストップシステムのもとに集中データベースを構築している。

使用言語は、主にCOBOLを使用しており、他に簡易言語のENFORM, ENABLE及びMUMPSも一部使用している。

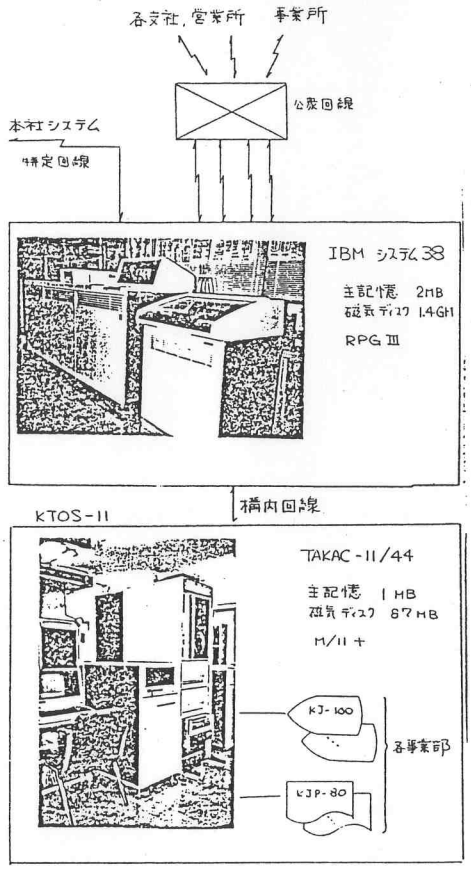


図3　神田分室システム

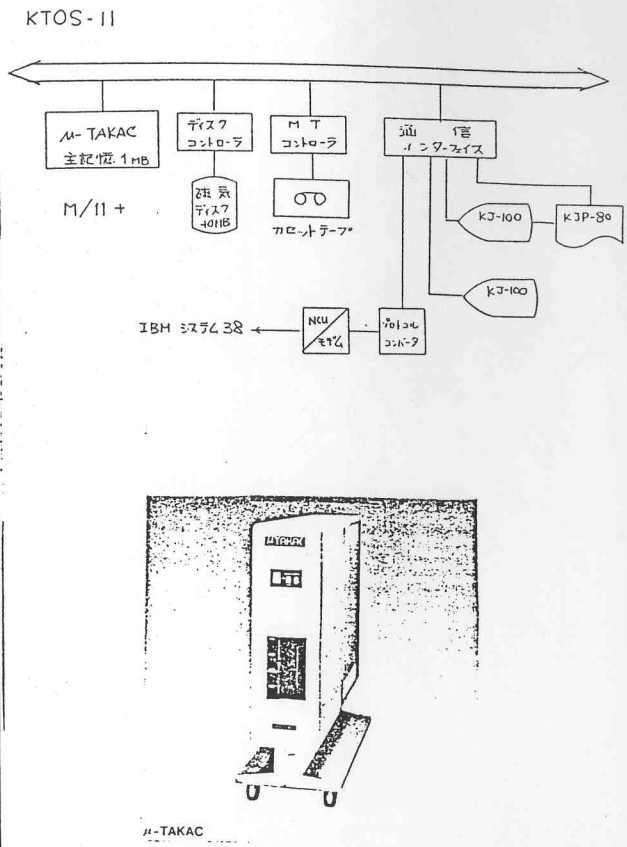


図4　支社・営業所システム

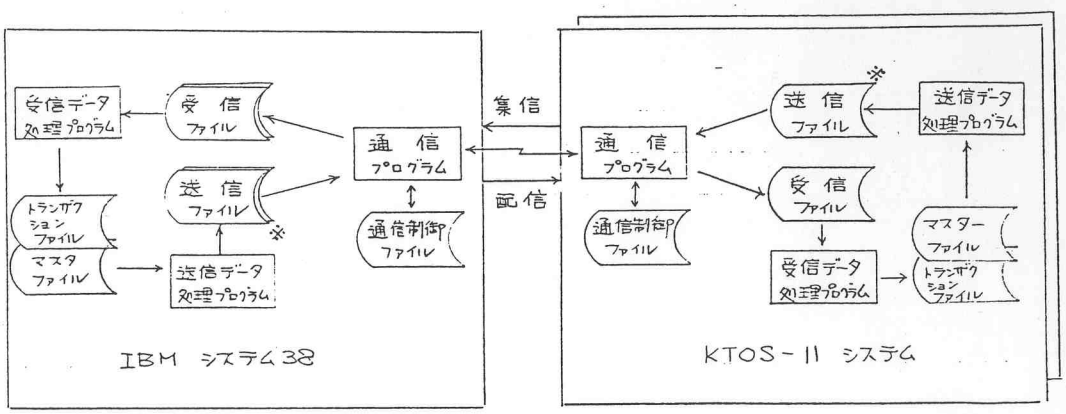


図5　営業システム 通信データ処理

※ 送信データは、バックアップのための10日間分保持

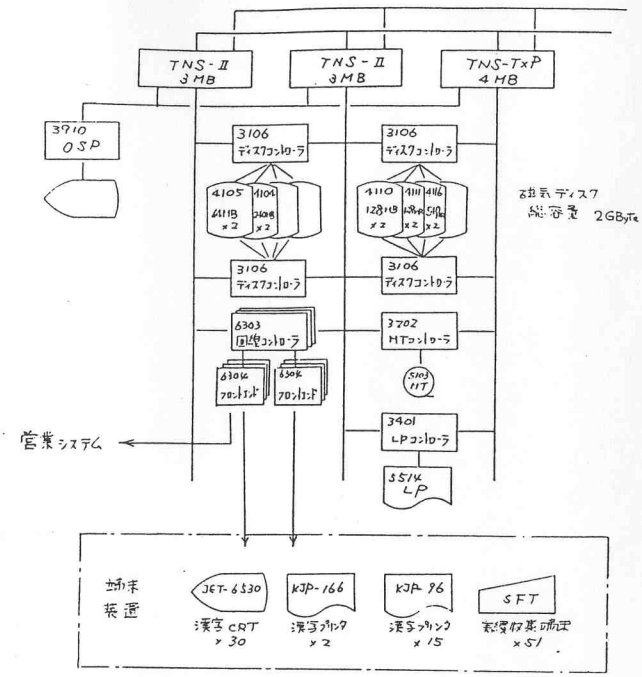


図6　名古屋事業所生産管理システム
ハードウエア構成図

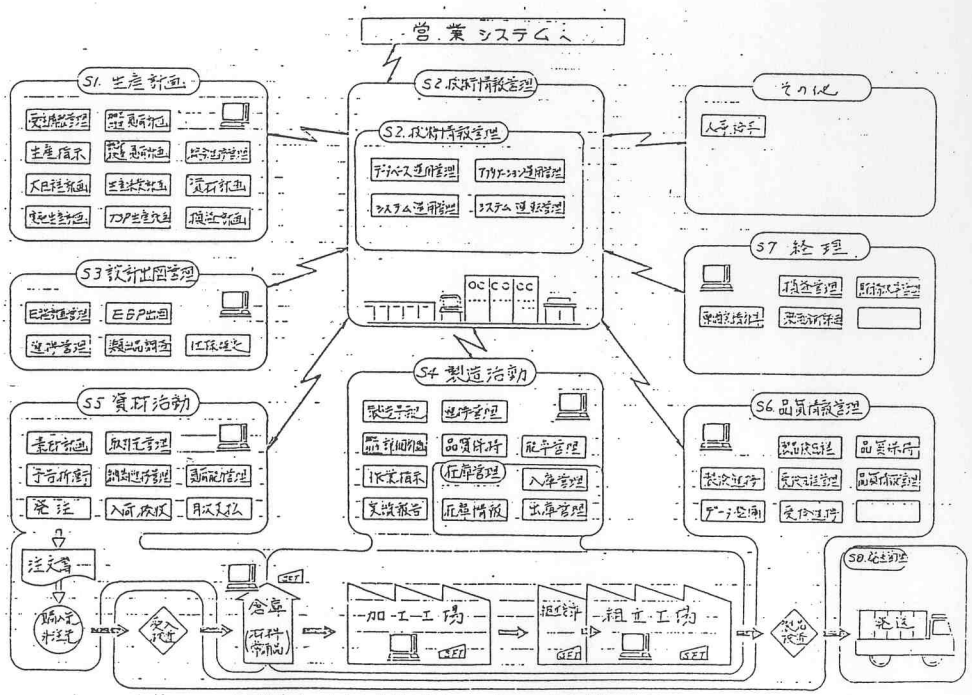


図7　名古屋事業所生産管理システム概要

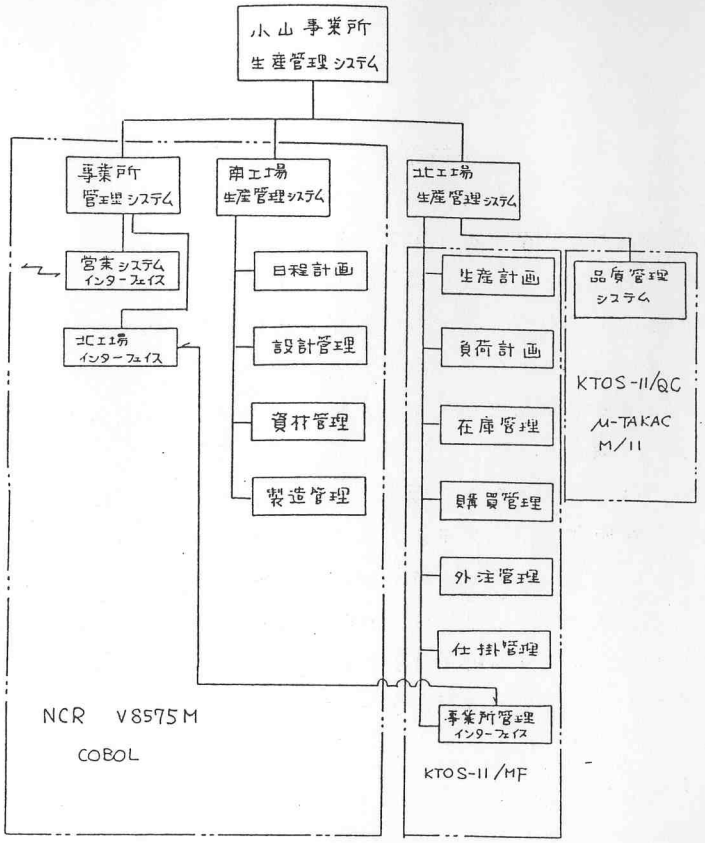


図8 小山事業所生産管理システム概要

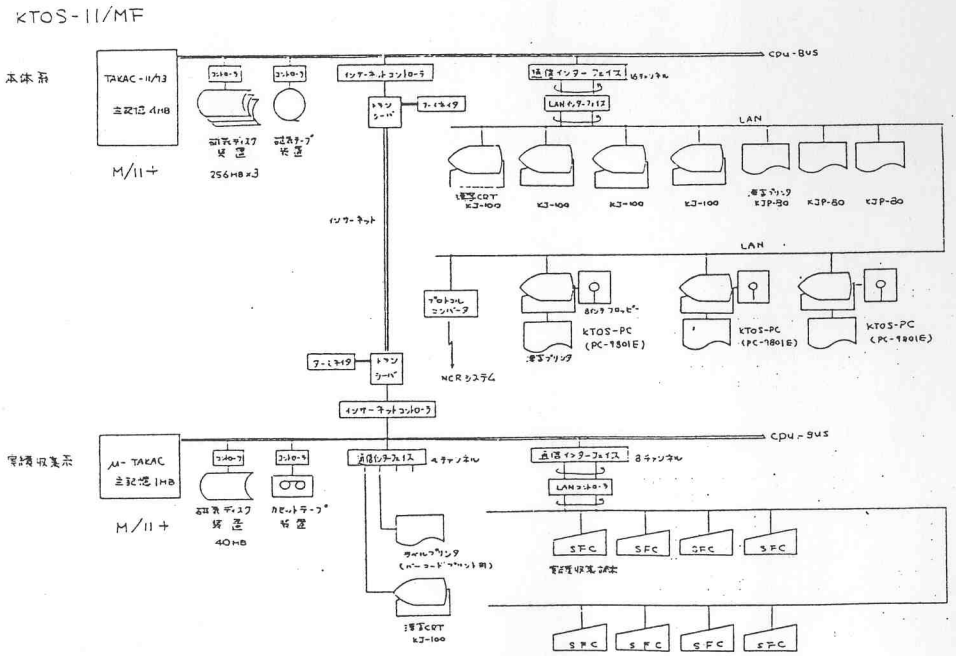


図9 北工場生管理システム構成図

6. 小山事業所生産管理システム

小山事業所の生産システムの概要は、図8に示すとおりである。この図に示されるとおり、小山事業所のシステムは、NCR V8575Mを中核として、KTOS-11/MFとKTOS-11/QCより構成される分散型のシステム形態をとっている。

NCR・V8575Mのシステムは、磁気ディスク容量約2.5GBによって、小山事業所全体の管理情報と大形変圧器を扱かう南工場の生産管理情報を保持している。このシステムの業務プログラムは、COBOLで主に記述されている。

北工場は、柱上変圧器を主に扱かう工場で、KTOS-11/QCシステムとKTOS-11/MFシステムによって構成される。

KTOS-11/QCは、当社の品質管理システムパッケージで、柱上変圧器に対する各種の試験を行なう自動検査サブシステムと結合されている。KTOS-11/QCでは、主に検査データの品質基準との照らし合わせ、統計処理、試験成績書発行などの業務を行なっている。QCシステムの全てのプログラムは、MUMPS(M/11)によって記述されている。

KTOS-11/MFも同様にMUMPSによって記述されている生産管理システムパッケージである。北工場においては、このパッケージを図9のようなTAKAC-11/73とμ-TAKACとをインサーネットで結合したシステム上に適用している。OSは、コンパイル型MUMPSをサポートするM/11+で、インサーネットを使用するDDP機能にてMUMPSネットを実現している。

また、MFシステムでは、ワークステーションとしてKTOS-PC(パーソナルコンピュータとしては、NECのPC-9801を採用)を使用している。このワークステーションでは、μ-MUMPSを使用したスタンドアロン処理と当社のターミナルエミュレータ(VT80, KJ-100, テクトロ4010/4014, Regisサポート)を使用するターミナル処理の2機能を提供している。むろん、スタンドアロン処理で作成されたデータのアップローディングとミニコン系からのダウンローディングは、通信ユーティリティによって行える。

7. あとがき

ここで紹介したシステムは、異機種間結合を実現したネットワークシステムであり、その中で大きな役割を果しているMUMPSシステムを中心に述べた。異機種を結合し、有機的にネットワーク化する技術は、増々コンピュータシステムが多様化する中で極めて重要と言える。

タイムリーシステムでは、関連会社へのネットワークの拡張、ニューメディアの取り込みなど今後もシステム機能を充実してゆく予定である。

スモールコンセプト（第二報）

田久　浩志(1) 馬場　謙介(2)

(1) 産業医科大学振動研究室　(2) 産業医科大学第二病理学教室
(1),(2) 北九州市八幡西区医生ヶ丘１－１

要　約

プログラムの小さな部品である『Small concept 』の概念を解説紹介し、その実例を示した。Small concept の概念をシステム設計に導入すると変数名の衝突が発生せず、複雑高度な処理が簡素化できることなど、多くの利点について述べた。

はじめに

MUMPS は高い移植性をほこっている。ところが、プログラムパッケージを部分的に自分のプログラムに取り込もうとする場合、一般的に変数名の衝突が生じるため移植が困難である。そこで、我々はこの問題を解決するためにスタック変数を定義し、これを変数の添字とすることにより原理的に変数名の衝突が発生しないスモールコンセプトの概念を第11回MUG 学術大会で発表した。今回は、我々がSmall concept に用いている変数名の制限、スタック変数の使用方法、プログラムの構造などを実例を挙げて具体的に紹介する。その上で我々が統一的に使用しているSmall concept の仕様の特徴、改良すべき点について述べ、Small concept が移植性をほこるMUMPS の本来の性質を引き出せることを強調する。

Small concept の概念

従来のUtility は、定型業務に対する汎利用性を目的とした独自の閉じた世界で構成され、他のプログラムより利用される事は考慮されていない。その結果、汎用性を増すためにパラメータをふやしたり、面倒な操作手順をユーザーに要求せざるを得なくなってしまった。また、一般的にサブルーチンはUtilityより汎用性が低い。これに対して、Small concept は、ライブラリに登録されたサブルーチンの性格に近いが、どんなプログラムの中でも使用できる一種のまとまった動きをするプログラムの部品であると言えよう。Small concept は、あくまで他のプログラムにおいて利用される事を前提として汎移植性を追求しているので、必然的に記述規則は不変的でかつ簡明となる。

Small concept の仕様

Small concept の仕様としては以下の基本条件を満たすことが必要である。

(1) Small concept と他のプログラムとの間で変数名の衝突がおきない。
(2) Small concept は一般的に他のSmall concept より呼ばれる。

基本条件を満たした現在我々が使用しているSmall concept の仕様の例を以下に示す。

変数の使い分け

Small concept はプログラムに組み込まれて使われるのが本来の姿であるので、そのメインプログラムの変数名と衝突が起きてはならない。この問題の解決のために、ルーチン内だけで有効な自動変数をMUMPS にもうける提案がこれまでに何度となく公式の場で検討されてきたが却下されてきた。Small concept では、変数名の衝突を避ける手段として、スタック変数を定義して使用することにより他の変数と区別する方法を採用した。これによりMUMPS の規約を変える事無く、自動変数のようにそのSmall concept 内部のみで有効な変数の概念を実現した。この結果、変数の取り扱いは以下のようになる。

1．スタック変数:
スタック変数は、Small concept のシステム変数の役割りを果し、Small concept の深さを情報として持つ。スッタク変数として変数% を使用する。
2．受渡し変数:
ルーチンよりSmall concept へ、又、Small concept よりルーチンへの値の受渡しをする受渡し変数に、%0(%1 - %9) を割りあてる。受渡し変数はFORTRAN の引数にあたる。
3．限局変数:
Small concept の内部のみで使用される自動変数を限局変数と呼び、% 英大文字で始まりスタック変数を第一添字に持つ変数で定義する。限局変数はスッタク添字の値によって区別されるために、Small concept 間での変数名の衝突は原理的にありえない。

Small concept の構造

1. Small conceptにおける開始処理の定型:
Small concept が起動された時に、はじめに行なわれる処理を定式化すると以下のようになる。
(1) 受渡し変数が未定義ならデフォルト処理またはエラー処理をする。
(2) 受渡し変数を任意の限局変数に渡す。
(3) スタックが未定義ならスタック変数を0 とする。
(4) スッタク変数に1 を加算する。

2. Small conceptのメインの処理:
Small concept のメインの処理では、変数に原則として限局変数を使用し、スッタク変数を絶対に修正しないことの二点を念頭に置いてプログラムを作成する。これによって、ユーザーより見たSmall concept は、受渡し変数さえ与えれば何の副作用も起さずに処理を行なうブラックボックスとなる。このため、Small concept より他のSmall concept を呼ぶネスティングをする時は、受渡し変数に値を入れてからSmall concept を呼べば、スタック変数の管理は自動的によばれたSmall concept 内部で行なわれる。また、連続してSmall concept を呼ぶパイピングをする時は、受渡し変数に一度値を与えたのち、Small concept を続けてコーディングするだけで良い。また、自分で自分自身を再起的に呼出すことも可能となり、再帰呼出しを用いた知識データーベースの開発などが楽にできるようになる。

3. Small conceptの終了処理の定式:
Small concept を終了する時に行なわれる処理を定式化すると次のごとくである。
(1) Small concept の処理結果を受渡し変数に設定する。
(2) スッタク変数より1 を減じてスタック変数に設定する。

(3) 使用した限局変数を全てKILLする。

具 体 例

パイピングの例として16進数10進数変換、10進数 8進数変換の使用例とプログラムリストをFig.1 に示す。また、ネスティングの例として、再起呼出しを用いたグローバルリストの使用例とプログラムリストをFig.2 に示す。

考 察

色々と利点のあるSmall concept であるが、限局変数に(%) をつけなければならないのが、煩わしい。これを避けるために、我々はSmall concept の開発時には(%) を無視して開発し、全てのデバッグの終了時点で、Change everyなどで簡単に(%) をつけるようにしている。自動的に、スタック変数、受渡し変数、以外の変数に(%) をつけるSmall concept があればより便利であろう。

受渡し変数によるエラーの処理を Small conceptでは統一的にどのようにしたら良いかと言う点を議論する。Small concept の種類によって、その処理の重要度が異なるため、エラーの時にはBreak 命令を実行して人間の支持をあおぐ方法がまず考えられる。しかし連続して処理をする場合、処理が中断されるのは望ましくない。そこで、エラー処理のデフォルトをヌル等にしておき、各Small concept の開始定型処理で、受け渡し変数がヌルの場合は何もせずに抜け出て、一連ののSmall concept を使用した後、エラーが生じたか否かを検査する方法や、何らかの方法でエラー情報を蓄積する方法などが考えられ実現もできるが、実際に必要性があるか否かは疑問がある。

限局変数の名前は、後の保守のために具体的な意味を持たせる事が望ましい。しかし、Small concept の作成を便利にするため、スタック添字を付けずに、自由に変数名を使用したくなることがある。このような変数の使用方法は、他のSmall concept を呼ばないSmall concept の中に限り許され、この時のみ限局変数に(%) のスタック添字はつけなくとも良い。

む す び

Small concept の目的は、ユーザーがシステムを構築するときのプログラムの部品を提供することである。この手法で最も効果的なことは、通常の方法で生ずる変数名の衝突を回避できる事である。これにより、従来、あまりにも私物化され使い捨てになっていたプログラムに関して、ソフトウェア部品の蓄積、共有化、再利用が可能となる。このため、大規模なシステムの構築や、研究用のシステム開発のような多品種少量のシステム開発の時などにも、規格が統一されたSmall concept は大きな武器となるであろうことを強調したい。今後、MDC のA 型草案として採択された、変数をスタックするNEW 命令が多くのMUMPS に装備されるようになると、Small concept の作成はより簡単になるであろう。また一般のルーチンの作成の時もSmall concept の規則に添ってコーディングを行なえば、プログラムの開発は非常に楽になる。今後は、Small concept 自身が他のSmall concept を呼ぶ事が頻繁にあるため、Small concept の名称を何らかの方法で統一しないと、混乱が生じるであろう。そのためにも、MUG でSmall concept に関するソフトウェアカタログを

整備し、共有財産とすることが望まれる。

```
>S %0="100"
>D ^HEXDEC,^DECOCT
>W %0,!
400

>S %0="FFF"
>D ^HEXDEC,^DECOCT
>W %0,!
7777
```

```
DECOCT
DECOCT  ;       Decimal to Octal
        ;
INIT    Q:$D(%0)=0
        Q:%0=0
        I '(%0?.N) U 0 W "Illigal character at DECOCT",! K %0 Q
        S:$D(%)=0 %=0
        S %=%+1
        ;
MAIN    S %QUOT(%)=%0,%0=""
        F %I(%)=0:0 Q:%QUOT(%)=0  S %REST(%)=%QUOT(%)#8,%QUOT(%)=%QUOT(%)\8
S %0=%REST(%)_%0
        ;
END     K %QUOT(%),%REST(%),%I(%)
        S %=%-1
        Q

HEXDEC
HEXDEC  ;       Hex Decimal to Decimal
        ;
INIT    Q:$D(%0)=0
        Q:%0=0
        I '(%0?.AN) U 0 W "Illigal character at HEXDEC",! K %0 Q
        S:$D(%)=0 %=0
        S %=%+1
        ;
MAIN    S %RESULT(%)=0,%STR(%)="0123456789ABCDEFabcdef"
        F %I(%)=1:1:$L(%0) S %CH(%)=$F(%STR(%),$E(%0,%I(%))) G:%CH(%)=0 ER
S %CH(%)=%CH(%)-2 S:%CH(%)>16 %CH(%)=%CH(%)-6 S %RESULT(%)=%RESULT(%)*16+%CH(%)
        S %0=%RESULT(%)
        ;
END     K %RESULT(%),%I(%),%CH(%),%STR(%)
        S %=%-1
        Q
        ;
ER      U 0 W "Illigal character at HEXDEC",! K %0
        G END
```

Fig.1　16進数10進数変換、10進数 8進数変換の例

```
>S ^A(1)=2
>S ^A(1,2)=3
>S ^A(2)=0
>
>D ^GBLIST
GLOBAL =^A(
A(1)=2
A(1,2)=3
A(2)=0

>
```

```
GBLIST
GBLIST  ;       PUT DATA LIKE ^A( ,^A(1,1,
        ;
        R "GLOBAL =",%0,!
        D GBL
        Q
        ;
GBL     ;
INIT    Q:$D(%0)=0  Q:%0=""  S:$D(%)=0 %=0 S %=%+1,%SEED(%)=%0,%NOD(%)=-1
        ;
MAIN    F %I(%)=0:0 S %KEY(%)=%SEED(%)_%NOD(%)_")" S %NOD(%)=$N(@%KEY(%)),
%TEMP(%)=%NOD(%) Q:%TEMP(%)=-1  D SUBMAIN
END     K %KEY(%),%SEED(%),%NOD(%),%TEMP(%),%DATA(%),%I(%)
        S %=%-1
        Q
        ;
SUBMAIN S %KEY(%)=%SEED(%)_%NOD(%)_")",%DATA(%)=$D(@%KEY(%))
        I (%DATA(%)=1)!(%DATA(%)=11) W %KEY(%),?15,@%KEY(%),!
        I %DATA(%)>1 S %0=$P(%KEY(%),")",1)_"," D GBL
        Q
```

Fig.2　再帰呼出しを用いたグローバルリストの例

時系列システムにおける保守時の診断仕様書

今泉幸雄

アップジョン・ファーマシュウティカルズ・リミテッド
総合研究所　研究データ管理室
高崎市大八木町168

1. はじめに

一般に、コンピュータを導入した研究の複合システムには、利用者とシステム開発者の業務が分かれているため、プログラムの品質保証、評価が正しくなされていない場合が多い。このような場合、保守に際してシステムを熟知していない利用者にも知識を提供し、品質保証が理解できる方法として、診断仕様書を立案したので紹介したい。

2. 目的と分類

目的としては、障害、プログラム、グローバル・ファイルの変更、及び業務の追加が発生した時に、最小のテスト・ケースで品質を最大に保証し、既存ソフトウェアとの互換性を維持、システム構造・知識を利用者へ提供する。さらに時系列上で流れているシステムの停止時間を最小にするために、標準データの常備、履歴管理などを用意し、回復や保証を効率化する。

2.1　障害時　障害が発生した時点で、障害レポートに状況を記入して、保守担当者に連絡する。該当する試験データ・運用データを保存し、修正環境に、移送して障害回復テストを行う。回復終了後に関連する標準データのテストをして副次障害がないことを確認した後、発生した環境の箇所に修正プログラム・グローバルファイルを移送する。同時に障害レポートには原因・処置について、詳細に記入するようにし、障害管理DataBaseに登録する。処理後の資料として、障害ごとの平均故障間隔、平均回復時間、プログラムごとの故障率・信頼率、どの開発フェイズでの不良かなどが計算され、かつ障害発生成長モデルの近似関数を求めて変曲点と総障害発生数の予測値などにも役立てる。

2.2　プログラムの変更時　プログラム全体を三つに分類して、①初期処理、全体の制御、終了処理　②ジョブ・メニュー単位の集合　③共通ルーチン　各プログラムごとのハイアールキィーをDataBaseに登録する。またグローバル・ファイルを含めて、1プログラムごとに使用しているグローバル・ファイル情報、逆に1グローバル・ファイルごとに使用しているプログラム情報、さらにシステム全体で重要であるグローバル・ファイルのノード情報をDataBaseに登録しておき、プログラム変更時に活用する。

2.3　グローバル・ファイルの変更時　グローバル・ファイル間においても、変更前後に機械的に得る方法があると良いが、実存するファイルよりは難しい。しかし設計者が知識として持っている内容を事実DataBaseとしてあらかじめ登録して

おき、誰でも参照できるようにする。方法としては、事実DataBaseは、各々のグローバル・ファイルのnode情報とdata情報の関係をサブスキーマとして下記の様に表現し、(関係式を)対話形式にてアクセスできる様にする。

```
↑FACTDB("TRSIX",1,"NODE",5) = 1;RSC;record sheet code@
↑FACTDB("TWORK",1,"KBDA",4) = "NODE";TRSIX;;1;RSC@ ・・・@
↑FACTDB("TWORK",1,"NODE",4) = 1;WCD;VA(3);work item code@
```

2.4　業務の追加(新規のプログラム、グローバル・ファイルの追加)
新規のプログラム、グローバル・ファイルの追加などによって、共通に影響を受けると予想される所は、DataBaseに登録しておき、作成者への資料提供をする。この作業をした後は、2.2、2.3、2.5、2.6の事項を更新することが必要である。

2.5　標準データの確認　例えば動物毒性試験システムでは、Study-Protocolに基づき、生データの入力・変更、最終報告書作成までのデータを時系列的に保持・解析するのが目的であるが、実際の試験過程に於いてどのステップでの環境をもOne-shotで設定するのは不可能である。そこでケース・スタディにて、Study-Protocolを作成し、各生データを入力し、最終報告までのデータを時系列に保存して(実際上で予想される再処理ポイントを設定して)メニュー単位で再実行可能にする。この確認作業は、障害、業務の追加などによって利用され、『診断テスト実施』によって記録として残され、履歴管理に保存され、品質の低下を防ぐ一つの指標となる。

2.6　履歴管理　障害、プログラム、グローバル・ファイルの変更、業務の追加などによって変更された履歴を管理する履歴DataBaseを登録しておき、利用者を支援する。履歴DataBaseには、作業の分類、要求ごとの分類、メニューNo、プログラム名、グローバル・ファイル名(各々詳細情報を含む)、作業者名、期日などのデータを登録しておき、利用者が対話形式にて、検索条件によって、過去のシステムの変化をいつでも参照できるようにし、保守作業の効率を高める。

3.　おわりに

保守時期になると、開発時期に比べて担当者数は少なくなり、実際開発時に担当しなかったプログラムも担当するのが現実である。この診断仕様書の作成によって本システムに対し、初めての保守担当者でも、コンピュータを通して過去の経験・知識を得られ、経験者の man to man の指導が最小となるように支援できる。

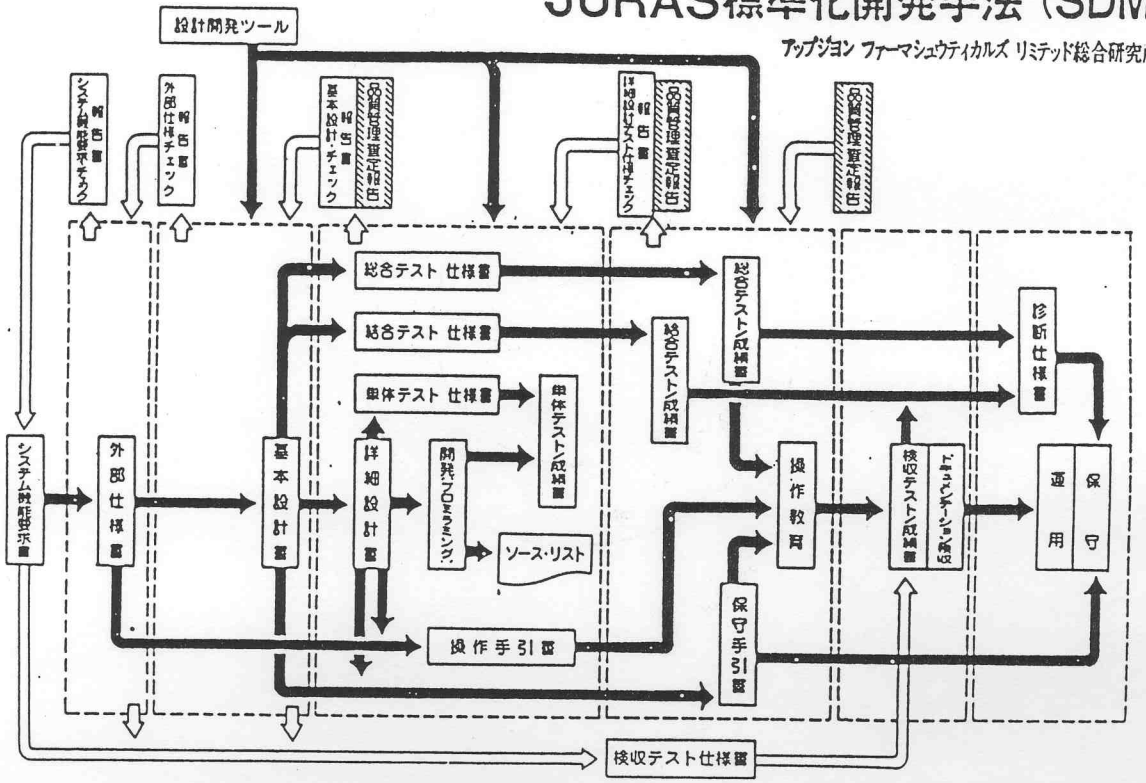
参考文献

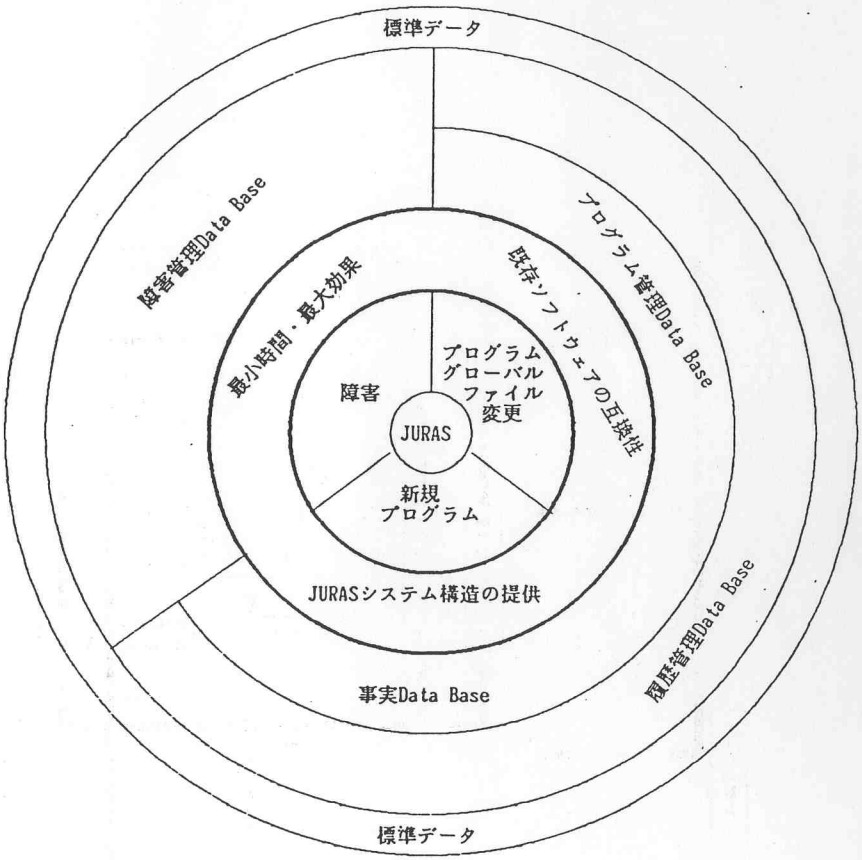
[1] 横井俊夫　人工知能向言語(1)
情報処理　Vol.17 ,No. 17 (1976) pp577-586

[2] 今泉幸雄　複合設計・開発における品質管理
第11回日本ＭＵＧ学術大会

JURAS標準化開発手法（SDM）

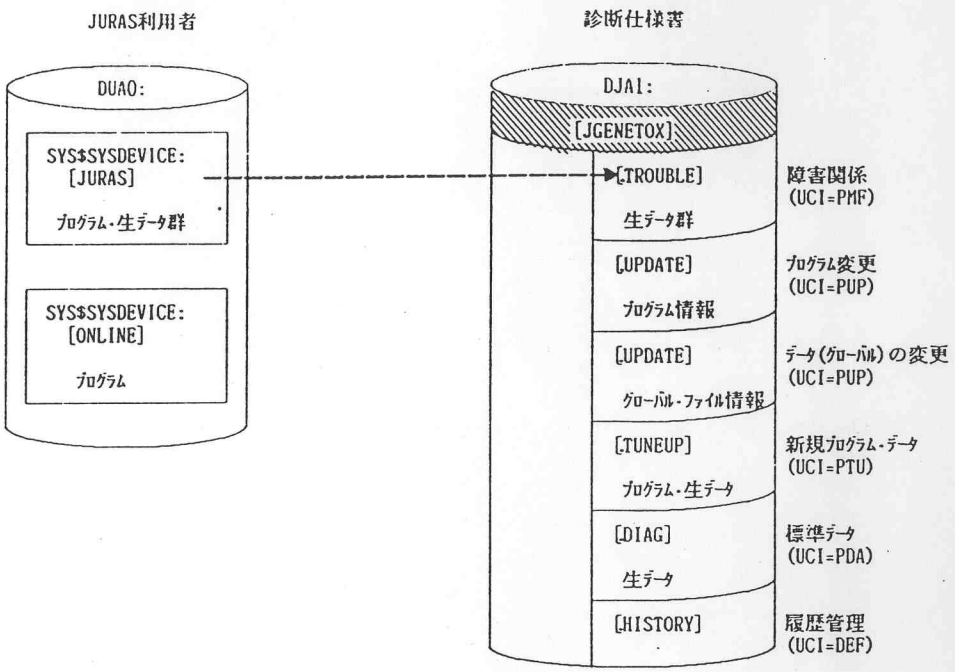
アップジョン ファーマシュウティカルズ リミテッド総合研究所





診断仕様書の概念図

アップジョン ファーマシューティカルズ リミテッド総合研究所

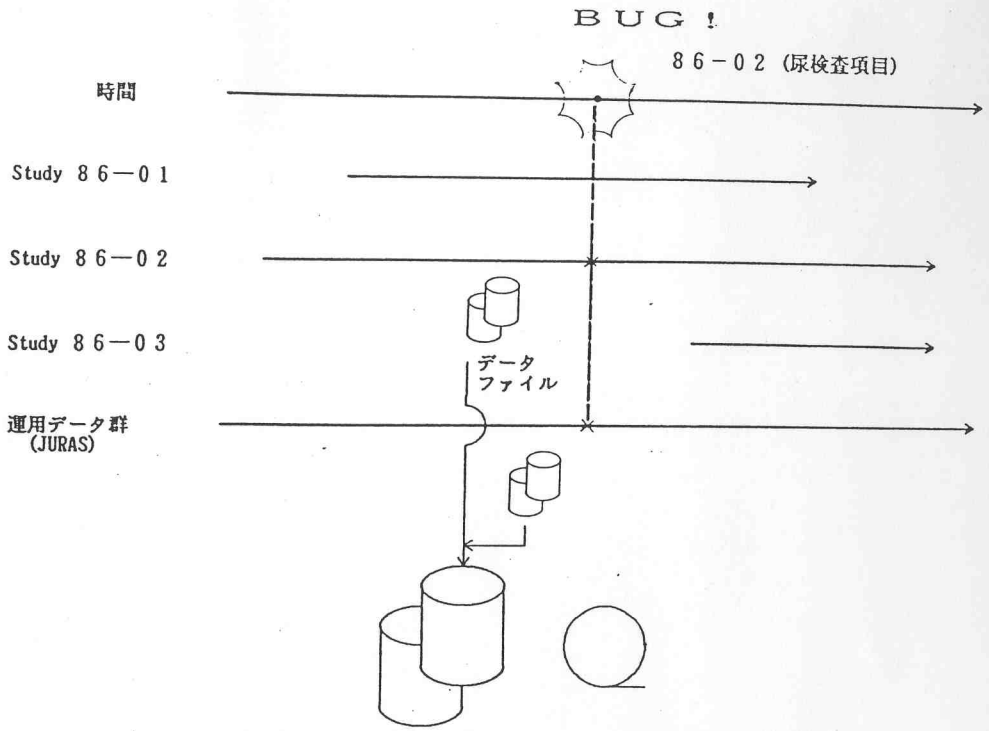


[JGENETOX] ⇨ 障害関係、標準データなどを実行する空間である。
従って未使用時は空である。

[JGENETOX.TROUBLE] ⇨ JURAS利用者が発見したbug の状態(生データ群)を
保存しておく。順編成ファイル(*.DAT形式)とする。

JURASにおける診断仕様書

アップジョン ファーマシュウティカルズ リミテッド総合研究所



発生した時点の当Study-no.(生データ・ファイル群) 及び運用データ群を
他の媒体あるいは、違うdirectory にSAVEしておく。

障害発生時処理

アップジョン ファーマシューティカルズ リミテッド総合研究所

障害票 (BUG REPORT)

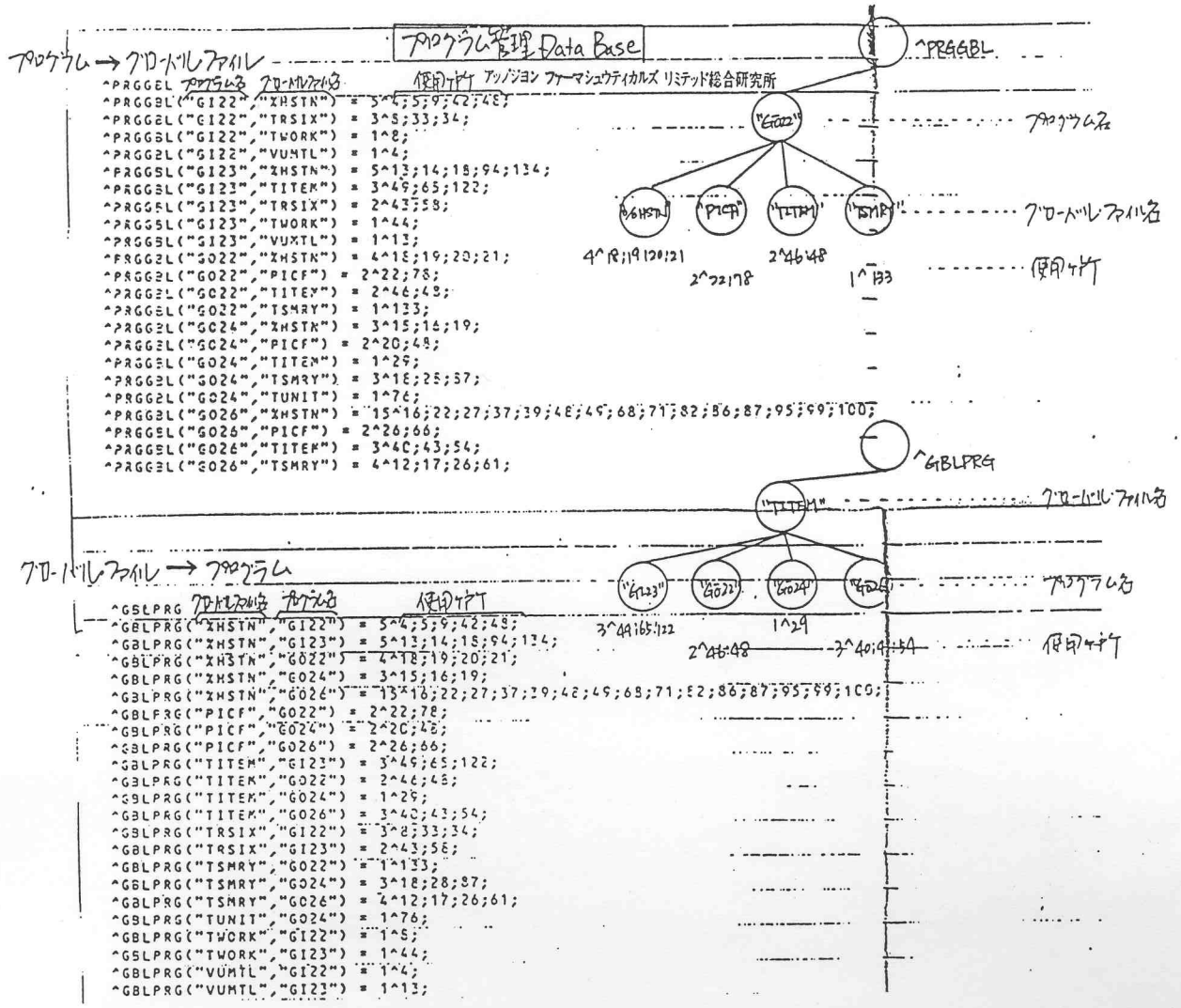
type I (状況編)

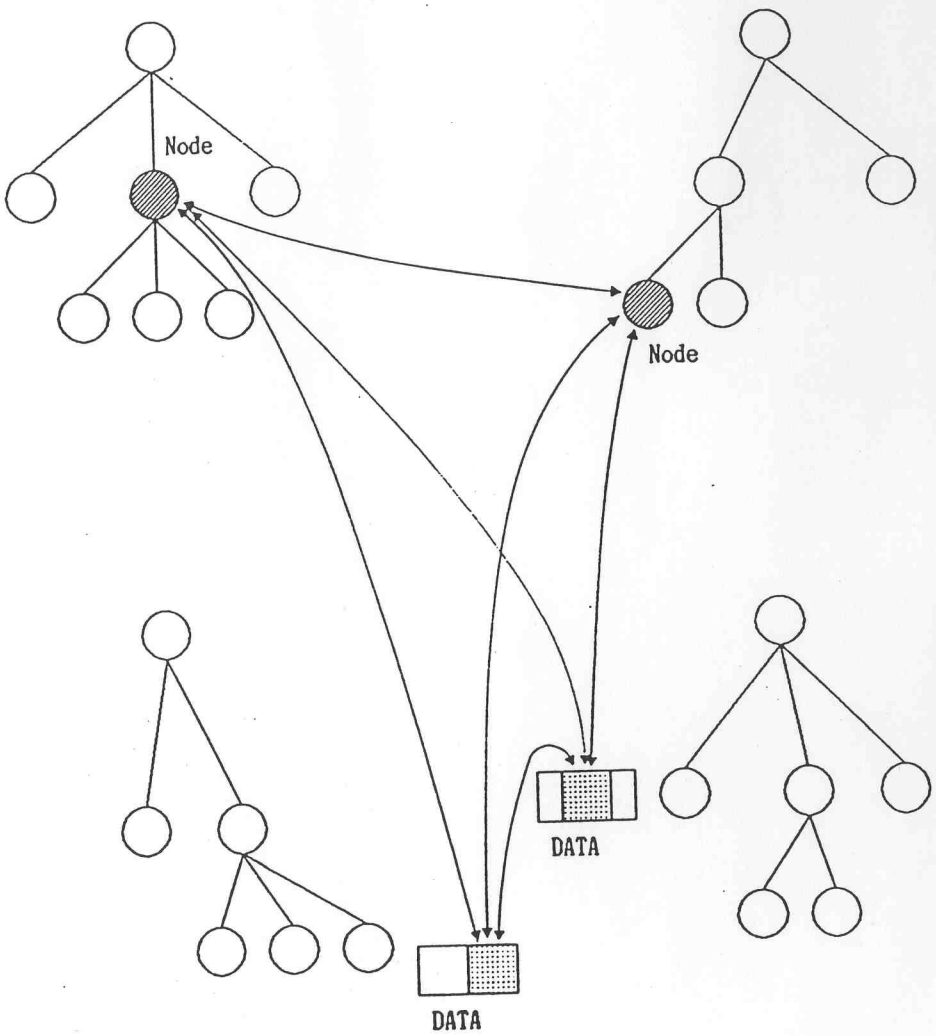
障害票番号 (bug report NO)	BR — —			1
発生時概略s	発生日付3A1	— —	発生時間3A2 :	担当者3A3
	システム名3B	1 PART-I (一般毒性) 2 PART-I (Utility) 3 PART-II (生殖) 4 PART-II (Utility) 5 Others ( )		
	メニューNO3B1	. メニュー名3B2:		
	検出方法3C	1 計算機試験 2 机上チェック 3 Walkthrough 4 Other( )		
	計算機試験 使用機種3C1	1 VAX11/750 2 T-1000 3 PR0350 4 NECPC9801F 5 Other( )		
	計算機試験 使用端末3C2	1 CIT-101 2 T-1000 3 PR0350 4 NECPC9801F 5 LA120 6 WX4636R(XYplotter) 7 CGT560(Graphic) 8 Other( )		
	計算機試験 発生環境3C3	1 全User数( ) 2 JURAS User数( )		
	計算機試験 使用重症度3C4	1 ハードエラー 2 基本ソフトエラー 3 言語レベルエラー(V/L: 略称:) 4 OSレベルエラー(V/L: 略称:) 5 システムDump有り 6 Others( )		
	計算機試験 影響度3C5	1 システム全体 2 サブシステム 3 回避手段有り 4 Others( )		
	計算機試験 TestCase3C6	1 Test NO( ) 2 Study NO(-) 3 Others( )		
	計算機試験 障害控え3C7	1 無 2 有 (Printer,HardCopy,XYplotter,)		
	提出日付3B1	— —	相手先名3B2:	
	再提出日付3B3	— —	相手先名3B4:	
発生時詳細s	状況詳細3Z : ----第3者にわかる様に詳細に記述すること。(不足時はtypeIII用紙)			

typeII (処置編)

障害票 (BUG REPORT)

障害票番号 (bug report NO)		BR — . —	2
原因c	原因分類c1	1 再現性なし　2 障害でない(　) 3 設計書不良　4 製造不良　5 修正不良 (BR- -) 6 test仕様書不良　7 Others(　)	
	不良分類c2	1 入力処理もれ　2 処理もれ　3 出力処理もれ　4 ロジック・ミス 5 入力形式・属性ミス　6 処理データ形式・属性ミス　7 出力形式・属性ミス 8 共通ルーチン使用ミス　9 ルーチン・インタフェイス・エラー 10 OSインタフェイス・エラー　11 WalkFile・インタフェイス・エラー 12 静的File・インタフェイス・エラー　13 動的File・インタフェイス・エラー 14 テーブル処理ミス　20 仕様・設計書ミス　21 データ・ミス 22 共通ルーチン仕様ミス　23 共通ルーチン処理ミス 24 すでにあるファイルの形式・属性ミス 0 文法エラー　30 基本ソフトエラー　31 ハードエラー	
処置T	処置T2: ----第3者にわかる様に詳細に記述すること。 (不足時はtypeIII用紙)		
修正明細M	修正者MA1		所属MA2:
	修正開始期日MA3	- -	修正終了期日MA4 - -
	修正対象MB1	1 program　2 Data-Base　3ハード　4 基本ソフト　5 Others(　)	
	修正対象物MB2	VAX:directory(　) 1 RA80　2 RA60　3 RL02	
	媒体MB3	T-1000: 1 Floppy　2 Disk	
	修正対象 プログラム名 データ名MB4	1 原始プログラム(　) 2 実行プログラム(　) 3 Data-Base　(　)	
	実行結果MC1	1 無　2 有(資料NO:　)	
	document変更MD	1 外部仕様書　2 基本設計書　3 詳細設計書　4 テスト仕様書　5 なし	
試行G	受理日付GA1	- -	受理者GA2
	再受理日付GA3	- -	再受理者GA4
	試行日付GB1	- -	試行時間GB2 : 試行者GB3
	comment GBV:		


プログラム管理 Data Base
^PRGGBL
プログラム → グローバルファイル
^PRGGBL("GI22","XHSTN") = 5^4;5;9;42;48;
^PRGGBL("GI22","TRSIX") = 3^8;33;34;
^PRGGBL("GI22","TWORK") = 1^8;
^PRGGBL("GI22","VUMTL") = 1^4;
^PRGGBL("GI23","XHSTN") = 5^13;14;18;94;134;
^PRGGBL("GI23","TITEM") = 3^49;65;122;
^PRGGBL("GI23","TRSIX") = 2^43;56;
^PRGGBL("GI23","TWORK") = 1^44;
^PRGGBL("GI23","VUMTL") = 1^13;
^PRGGBL("GO22","XHSTN") = 4^18;19;20;21;
^PRGGBL("GO22","PICF") = 2^22;78;
^PRGGBL("GO22","TITEM") = 2^46;48;
^PRGGBL("GO22","TSMRY") = 1^133;
^PRGGBL("GO24","XHSTN") = 3^15;16;19;
^PRGGBL("GO24","PICF") = 2^20;48;
^PRGGBL("GO24","TITEM") = 1^29;
^PRGGBL("GO24","TSMRY") = 3^18;25;87;
^PRGGBL("GO24","TUNIT") = 1^76;
^PRGGBL("GO26","XHSTN") = 15^16;22;27;37;39;48;49;68;71;82;86;87;95;99;100;
^PRGGBL("GO26","PICF") = 2^26;66;
^PRGGBL("GO26","TITEM") = 3^40;43;54;
^PRGGBL("GO26","TSMRY") = 4^12;17;26;61;
プログラム名
グローバルファイル名
使用ナンバー
アッソシエーション フォーマシュウティカルズ リミテッド総合研究所
"GI22"
"XHSTN"
"PICF"
"TITEM"
"TSMRY"
4^8;19;20;21
2^22;78
2^46;48
1^133
^GBLPRG
グローバルファイル → プログラム
^GBLPRG
^GBLPRG("XHSTN","GI22") = 5^4;5;9;42;48;
^GBLPRG("XHSTN","GI23") = 5^13;14;18;94;134;
^GBLPRG("XHSTN","GO22") = 4^18;19;20;21;
^GBLPRG("XHSTN","GO24") = 3^15;16;19;
^GBLPRG("XHSTN","GO26") = 15^16;22;27;37;39;48;49;68;71;82;86;87;95;99;100;
^GBLPRG("PICF","GO22") = 2^22;78;
^GBLPRG("PICF","GO24") = 2^20;48;
^GBLPRG("PICF","GO26") = 2^26;66;
^GBLPRG("TITEM","GI23") = 3^49;65;122;
^GBLPRG("TITEM","GO22") = 2^46;48;
^GBLPRG("TITEM","GO24") = 1^29;
^GBLPRG("TITEM","GO26") = 3^40;43;54;
^GBLPRG("TRSIX","GI22") = 3^8;33;34;
^GBLPRG("TRSIX","GI23") = 2^43;56;
^GBLPRG("TSMRY","GO22") = 1^133;
^GBLPRG("TSMRY","GO24") = 3^18;28;87;
^GBLPRG("TSMRY","GO26") = 4^12;17;26;61;
^GBLPRG("TUNIT","GO24") = 1^76;
^GBLPRG("TWORK","GI22") = 1^8;
^GBLPRG("TWORK","GI23") = 1^44;
^GBLPRG("VUMTL","GI22") = 1^4;
^GBLPRG("VUMTL","GI23") = 1^13;
3^49;65;122
2^46;48
1^29
3^40;43;54
"GI23"
"GO22"
"GO24"
"GO26"



事実DataBaseの考え方

アップジョン ファーマシュウティカルズ リミテッド総合研究所

事実DataBaseの構造

アップジョン ファーマシュウティカルズ リミテッド総合研究所

<例>

↑TWORK(WCD) = 作業項目総称;作業項目略名;検疫用プロトコールコード;本試験用プロトコールコード;レコード種別コード

↑TRSIX(RSC) = ...

↑FACTDB("TWORK",1,"NODE",4) = 1;WCD;VA(3);work item code@

↑FACTDB("TWORK",1,"DATA",4) = 1;WCD↑... @RSC;D2;V9(2);record sheet code@

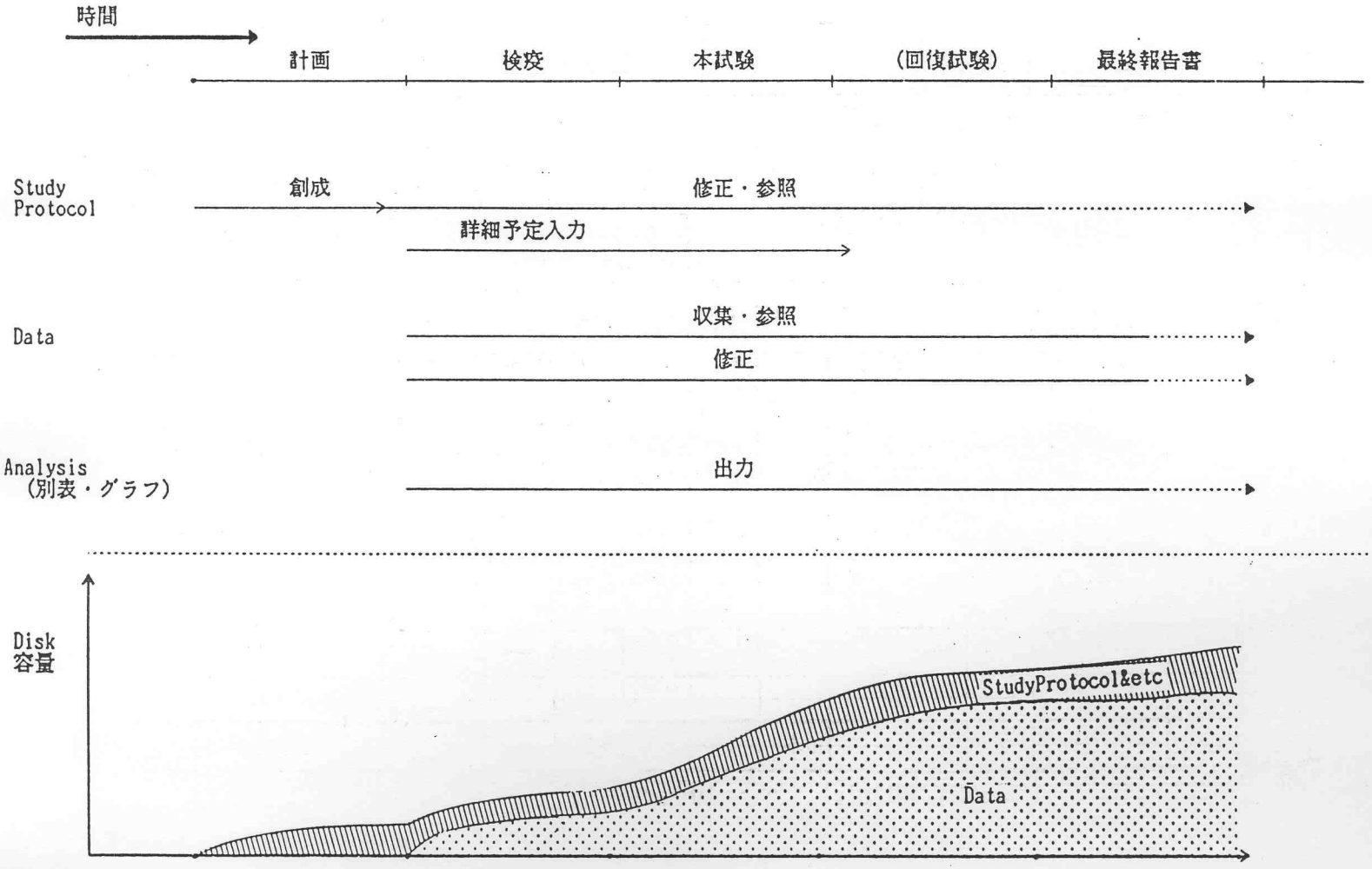
↑FACTDB("TWORK",1,"KBDA",4) = "NODE";TRSIX;;1;RSC@ ... @

JURASのフローバルファイル

```
^TRSIX =
^TRSIX(1) = Clinical observation;1;CSGN;3;3;C;8;
^TRSIX(2) = Histopathology;2;HSTP;20;2;1;9;
^TRSIX(3) = Grosspathology;3;GRSP;27;2;1;10
^TRSIX(4) = Body weight;C;WGHT;4;1;0;4;WGHT
^TRSIX(5) = Water intake;C;WITK;5;1;C;5;WITK
^TRSIX(6) = Food consumption;C;FITK;6;1;C;6;FITK


^TWORK =
^TWORK(1) = Randomization;CR;^;
^TWORK(2) = Administration;CA;^;71
^TWORK(3) = Clinical Observation;C0;13.0101^16.0101;01
^TWORK(4) = Body Weight;C4;13.0401^16.0401;04
^TWORK(5) = Water intake;CW;13.0301^16.0301;C5
^TWORK(6) = Food consumption;CF;13.0201^16.0201;C6
```

薬案 Data.Base

```
^FACTDB
o^FACTDB('TRSIX',1,'NODE',5) = 1;RSC;V9(2);record sheet code@
 ^FACTDB('TWORK',1,'DATA',4) = 1;WCD^WRKFNAM;D2;VA(30);work item full name@WRKBNAM;D2;VA(30);work item brief name@QID;D4;VA(3.2);
 ocol item code in quarantine@TPID;D2;VA(3.2);protocol item code in test@ZX1@RSC;D4;V9(2);record sheet code@ZX1
o^FACTDB('TWORK',1,'KBDA',4) = 'NODE';TRSIX;;1;RSC@'NODE';SZHSTNC;;1;REC@'NODE';SZHSTND;;1;REC@
o^FACTDB('TWORK',1,'NODE',4) = 1;WCD;VA(3);work item code@
```


時間
計画
検疫
本試験
(回復試験)
最終報告書
Study Protocol
創成
修正・参照
詳細予定入力
Data
収集・参照
修正
Analysis (別表・グラフ)
出力
Disk 容量
StudyProtocol&etc
Data

毒性試験・チェック表 (A)

T30-001

Study No.		99-Q1	99-Q2	99-Q3	99-Q4	備考
試験タイプ		Acute	Acute	Sub acute	Sub acute	
動物種		Mouse	Rat	Rat	Rat	
動物数	投与時匹数	65	65	130	130	
	試験時匹数	60	58	120	120	
	回復試験匹数	x	x	40	40	
検疫期間	雄 総日数	14	14	14	14	
	雄 (開始日/終了日)	[illegible] / 19-MAY-85	7-MAY-85 / 20-MAY-85	14-MAY-85 / 27-MAY-85	[illegible] / 22-[illegible]	
	雌 総日数	14	14	14	14	
	雌 (開始日/終了日)	6-MAY-85 / 19-MAY-85	14-MAY-85 / 27-MAY-85	7-MAY-85 / 20-MAY-85	[illegible] / 22-[illegible]	
群分け	群分けの状況	[illegible]	連続的に作業	連続的に作業	[illegible]	
	雄 群分け日	20-MAY-85	21-MAY-85	28-MAY-85	24-MAY-85	
	雌 群分け日	20-MAY-85	28-MAY-85	21-MAY-85	24-MAY-85	
投与期間	雄 総日数	14	14	90	28	
	雄 (開始日/終了日)	20-MAY-85 / 2-JUN-85	21-MAY-85 / 3-JUN-85	28-MAY-85 / 25-AUG-85	24-MAY-85 / 20-JUN-85	
	雌 総日数	14	14	90	28	
	雌 (開始日/終了日)	20-MAY-85 / 2-JUN-85	28-MAY-85 / 10-JUN-85	21-MAY-85 / 18-AUG-85	24-MAY-85 / 24-JUN-85	
解剖日	雄	[illegible]	[illegible]	25-AUG-85	21-JUN-85	
	雌	[illegible]	[illegible]	18-AUG-85	21-JUN-85	
回復期間	雄 総日数	/	/	10	/	
	雄 (開始日/終了日)			21-AUG-85 / 4-SEP-85		
	雌 総日数	/	/	10	/	
	雌 (開始日/終了日)			19-AUG-85 / 29-AUG-85		
剖検日	雄			4-SEP-85		
	雌			28-AUG-85		
最終報告日		13-JUN-85	30-JUN-85	30-SEP-85	21-JUL-85	
LD50 算出方法						
			99-01 8 [illegible] 97-4[illegible]			
			Female 7日おくれ	Male 7日おくれ		
		[illegible] control [illegible] かえる。				

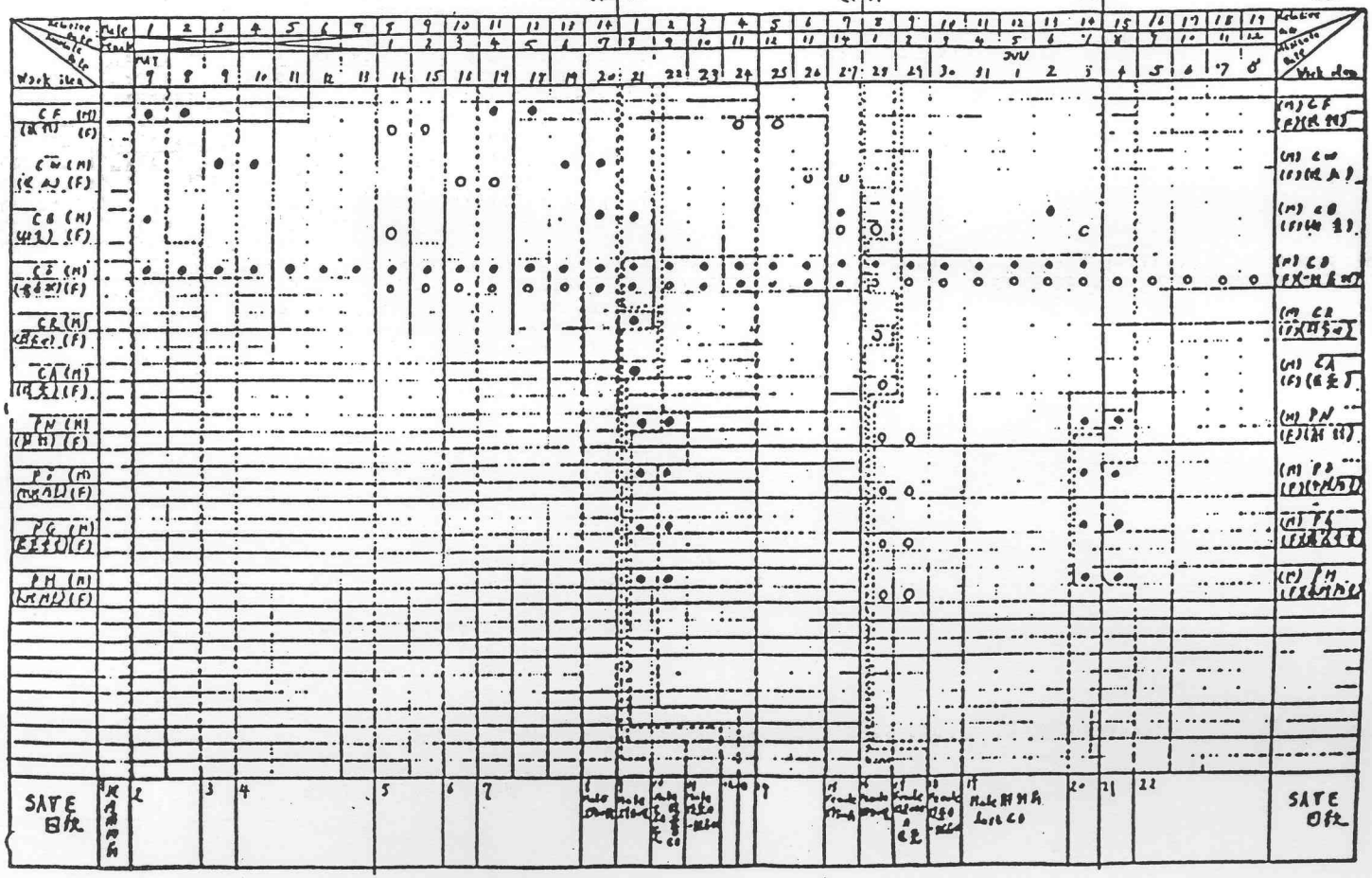
コクヨ シヨ-16

Time Table

Ⓠ Quarantine
Ⓣ Treatment

アップジョン ファーマシュウティカルズ リミテッド総合研究所

Study No. 99-02 Study Type. Acute



履歴管理 Data Base

アップジョン ファーマシューティカルズ リミテッド総合研究所

新規プログラム

```
^DHIST
^DHIST("NEWPRG",1) = ECR-001;VAX;Logout;Final_character;LDC;Y.IMAIZUMI15-FEB-85;850215;52641@05-MAR-85;850305;526590
^DHIST("NEWPRG",1,"DOCUM") = ;;@;131;;@;@
^DHIST("NEWPRG",1,"GLOBAL") = NEW;TM2CX;24@
^DHIST("NEWPRG",1,"GLOBAL",1) = TM2OX(TM2CLS,TM2DEV,TM2LNE)
^DHIST("NEWPRG",1,"GLOBAL",2) =  TM5CLS is class name
^DHIST("NEWPRG",1,"GLOBAL",3) =  TM8DEV is device information
^DHIST("NEWPRG",1,"GLOBAL",4) =   DATA are general_name;length;colum;
^DHIST("NEWPRG",1,"GLOBAL",5) =  TM2LNE is output character information
^DHIST("NEWPRG",1,"GLOBAL",6) =   DATA are character;charater type;
^DHIST("NEWPRG",1,"MEDIA") = RA6O;DJA1:[JGENETOX.TROUFLE];UCI=PMF
^DHIST("NEWPRG",1,"REQUEST") = Disolay "JURAS" when JURAS is end@
^DHIST("NEWPRG",1,"REQUEST",1) = Display "JURAS" what are expanded in full screen
^DHIST("NEWPRG",1,"REQUEST",2) =  for instace , J constructed J characters.
^DHIST("NEWPRG",1,"ROUTINE") = NEW;CXDSP;IO@CLD;CX01;2@OLD;CXC1M;2
^DHIST("NEWPRG",1,"ROUTINE",1) = CXDSP; new program
^DHIST("NEWPRG",1,"ROUTINE",2) = CXDSP; display VT_terminal from ^TMFOX
^DHIST("NEWPRG",1,"ROUTINE",3) = CXDSP; using %CRT routine
^DHIST("NEWPRG",1,"ROUTINE",4) = HAL^CX01;to add;D ^CXDSP:SSD(^VTOP($IO))=0
^DHIST("NEWPRG",1,"ROUTINE",5) = HAL+1^CXC1;to acd; D ^CXDSP
^DHIST("NEWPRG",1,"ROUTINE",6) = HAL^CXC1M; similary CX01
^DHIST("NEWPRG",1,"ROUTINE",7) = HAL+1^CXC1M; similary CXC1
^DHIST("NEWPRG",1,"WORKINF") = Y.IMAIZUZMI@25-FEB-85;850225;52651@27-FEB-85;850227;52653@135.OH^K.FUJIMOTO@26-FEB-85;
```

プログラムの変更

```
^DHIST("OLDPRG",1) = SC-001;VAX;11.01;Signal_change;Pathology;R.NAKAMURA@15-FEB-85;850215;52641@05-MAR-85;850305;526590
^DHIST("OLDPRG",1,"DOCUM") = ;;@;131;;@;@
^DHIST("OLDPRG",1,"GLOBAL") = OLD;TREPO;4@
^DHIST("OLDPRG",1,"GLOBAL",1) = TREPO(record_sheet_ccce,line)
^DHIST("OLDPRG",1,"GLOBAL",2) =  line is 54,55,56,57
^DHIST("OLDPRG",1,"MEDIA") = RA6O;DJA1:[JGENETOX.UPDATE];UCI=PUP
^DHIST("OLDPRG",1,"REQUEST") = Signal of Individual_report@
^DHIST("OLDPRG",1,"REQUEST",1) = Signal change of life and death information
^DHIST("OLDPRG",1,"REQUEST",2) =  !--->D , %--->K , @--->M , #--->E
^DHIST("OLDPRG",1,"ROUTINE") = OLD;GC01;1@OLD;G0C2;1@CLD;G004;1@CLD;G009;2
^DHIST("OLDPRG",1,"ROUTINE",1) = G001+10;S FCDT(1)="!",FCDT(2)="%",FCDT(3)="@",FCDT(4)="#"
^DHIST("OLDPRG",1,"ROUTINE",2) = G002+11; similarly GC01
^DHIST("OLDPRG",1,"ROUTINE",3) = G004+13; similarly G001
^DHIST("OLDPRG",1,"ROUTINE",4) = G009+16; similarly G001
^DHIST("OLDPRG",1,"ROUTINE",5) = G009+155; S DT=$S(SP...))
^DHIST("OLDPRG",1,"ROUTINE",6) = ;S DT=$S(SP...,FC=1:FCDT($P(GMAN,"^",3))
^DHIST("OLDPRG",1,"WORKINF") = Y.IMAIZUZMI@25-FEB-85;850225;52651@27-FEB-85;850227;52653@135.OH;6.OH^K.KOIKE@26-FEB-85;
```

850226;526520

850226;526

ＭＤ－１　　第１２回日本マンプスユーザーズグループ学術大会（１９８５年８月）

汎用性を重視した臨床検査コンピュータシステムの設計・導入について
－第３報　検査前処理業務について－

○長原　三輝雄　　長谷川　俊雄　　黒田　満彦

福井医科大学　検査部
福井県吉田郡松岡町下合月２３

〔要　約〕

汎用性と能率性を重視して臨床検査コンピュータシステムを設計し、導入した。検査受付後から検査実施までの、いわゆる前処理業務において、ワークシート、検体ラベル、分取分注リストなどの印字情報やフォーマット条件をテーブル設定することにより、新設や変更に対する汎用化を図った。また、作成帳票のグループＮｏ．や出力装置Ｎｏ．をテーブル化し、ワンタッチ指定が行えるなどの能率化を図った。さらに各種帳票の印字情報として、帳票の種類に応じて検体区別、検体コメント、患者ＩＤの印字とその位置を吟味し、作業の能率化を図るとともに検体取り違えなどの防止を図った。

〔はじめに〕

血液などの検体を取り扱う検査は、通常、検体受付、検査前処理、検査の実施、結果の報告の順に作業が進められる。このうち検査前処理は作業が煩雑であり、能率性が要求される。また、検体の流れとシステムの流れが別々に進む部分を生ずるため、システム設計にあたっては、照合しやすいよう、また検体の取り違えなどが生じないように留意する必要がある。

ところで、福井医科大学検査部の臨床検査コンピュータシステムは、汎用性と能率性を重視して設計を行い、１９８３年１０月、附属病院の診療開始と同時に稼働を開始し、その概要については既に報告した[1)-3)]。１９８４年４月からは第２期として、端末の増設と血清分取分注装置のオンライン化を行い、検査前処理業務の充実を図った。そこで今回は、検査受付後の操作である、ワークシート発行、ラベル発行、血清分取分注装置の運用など検査前処理業務について報告する。

〔システム構成〕

本システムのハードウェア構成を図１に示した。ＤＥＣ社ＰＤＰ１１／４４（主記憶容量１ＭＢ）を中心に磁気ディスク記憶装置ＲＡ６０（記憶容量２０５ＭＢ）２台および磁気テープ記憶装置ＴＲ１１（記録密度１６００ＢＰＩ）１台より構成されている。端末としてカナＣＲＴ　１３台、高速シリアルプリンタ　６台、シリアルプリンタ　５台、ラインプリンタ　１台が設置され、６台の分析装置をコントロールボックス　ＣＢＸ－３６０を介してオンライン処理している。また、高速通信回線を介し、病歴システム、入院システムおよび外来システムとコミュニケーションを行っており、院内総合医療情報システムの一翼を担っている[4)]。

血清分取分注装置は立石電機（株）製ＨＥＵ－５０１を使用した[5)]。本装置にはコントローラとしてソード社製マイクロコンピュータＭ３４３が接続されており、このＭ３４３より信号変換装置ＲＣＣ　ＭＯＤＥＬ６５を介してホストコンピュータとコミュニケーションを行っている（図２）。

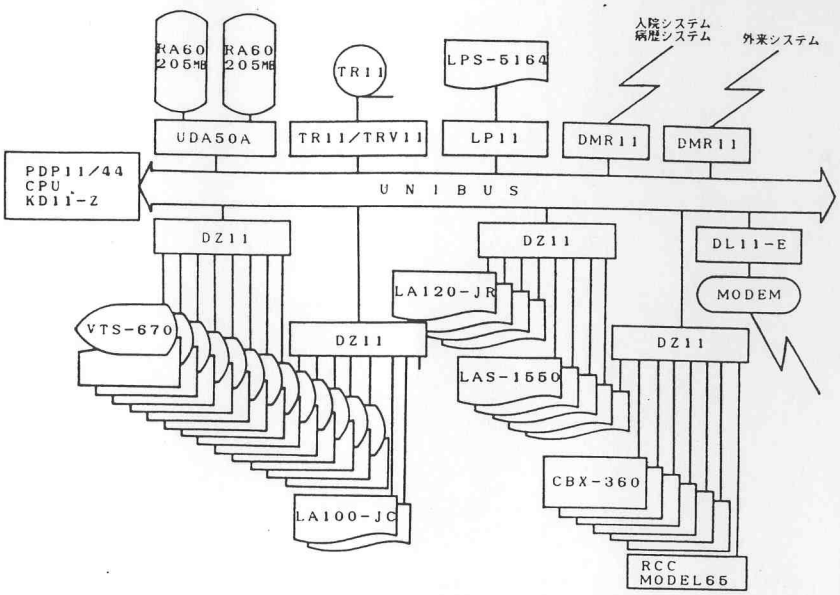


図1　福井医科大学　臨床検査コンピュータシステム　ハードウェア構成図

なお、ソフトウェアは、われわれの基本設計をもとに住友電工が製作した。

〔機　能〕

1．ワークシートの発行

ワークシートは、部門、検査実施日、ワークシートＮｏ．、および発行の対象となる受付期間を指示入力して発行する（図３）。再発行や発行取消も行える。いずれの場合も、ワークシートＮｏ．を複数指定出来、一回のオペレーションで複数のワークシートを連続発行することが出来る。

図４には、大型自動分析装置のワークシートを示した。このワークシートのように取り扱う項目数が多いものは、１項目あたり縦一行で処理出来るよう設計した。また、糖負荷試験は１依頼に対し複数の検体が存在する検査であり、システム上の取り扱いが複雑となることから、従来のシステムでは別処理を余儀されていたが、本システムではプログラムのモジュール化などの工夫を行い、通常の検体と同じ扱いで同一のワークシートへの打ち出しが可能となった。一方、尿蛋白検査など用手法による検査のワークシートは、検査を

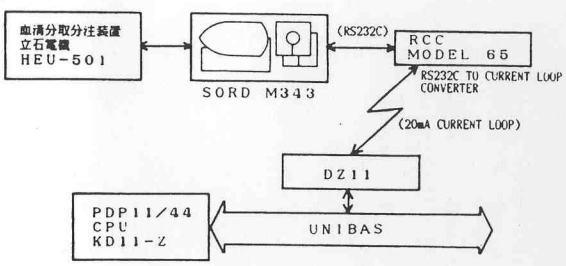


図2　血清分取分注装置オンライン構成図

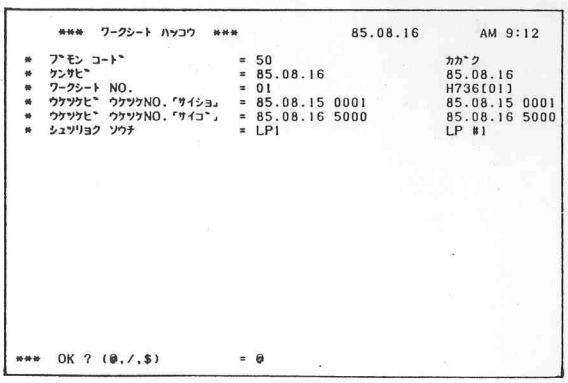
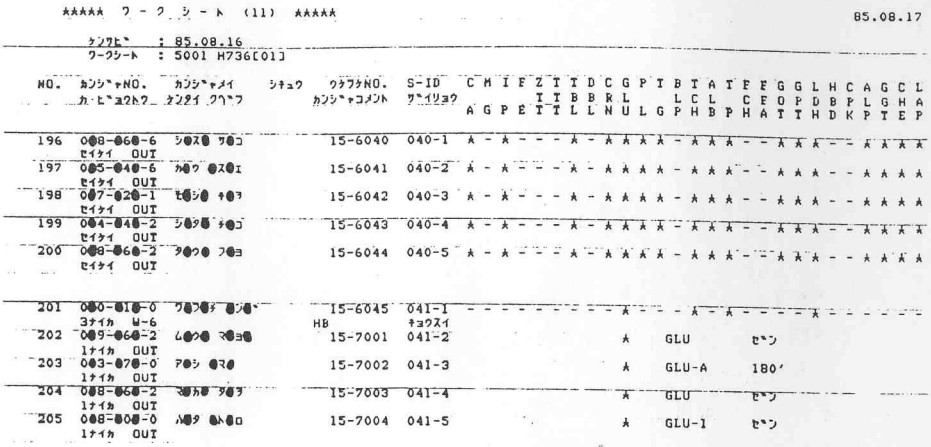


図3　ワークシート発行指示画面

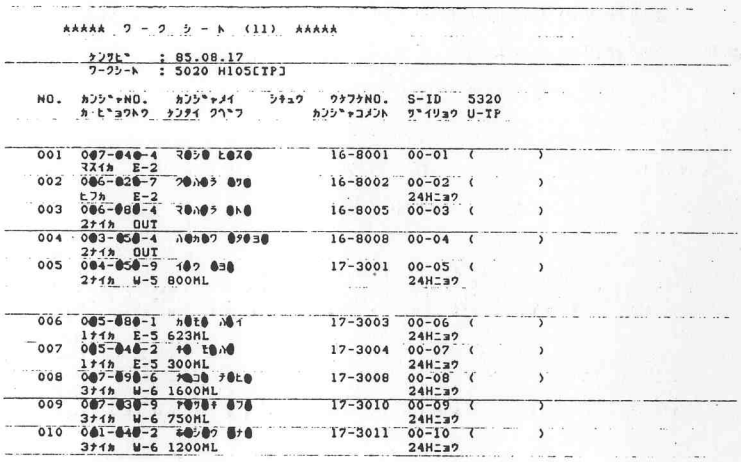


AAAAA ワ ー ク シ ー ト (11) AAAAA 85.08.17

ｹﾝｻﾋﾞ : 85.08.16
ﾜｰｸｼｰﾄ : 5001 H736[01]

NO.	ｶﾝｼﾞｬNO. ｶ･ﾋﾞｮｳﾄｳ	ｶﾝｼﾞｬﾒｲ ﾅﾝﾀｲ	ｼｭﾋｭｳ	ｳｹﾌﾀNO. ｶﾝｼﾞｬｺﾒﾝﾄ	S-ID ｻﾝｲﾘｮｳ	A G	C P	H E	I T	F E	Z T	I T	T L	D B	C R	G L	P N	I U	B L	A C	T L	A B	F P	E H	G C	G E	G O	L P	H D	C B	A P	G K	C P	L T	A E	C P
196	008-060-6 ｾｲｹｲ OUT	ｼ●ｽ● ﾜ●ｺ		15-6040	040-1	A	-	A	-	-	-	A	-	A	A	A	A	-	A	A	A	-	-	A	A	A	-	-	A	A	A	A	A	A
197	005-040-6 ｾｲｹｲ OUT	ｶ●ｳ ●ｽ●ｴ		15-6041	040-2	A	-	A	-	-	-	A	-	A	A	A	A	-	A	A	A	-	-	A	A	A	-	-	A	A	A	A	A	A
198	007-020-1 ｾｲｹｲ OUT	ﾋ●ｼ● ﾅ●ﾗ		15-6042	040-3	A	-	A	-	-	-	A	-	A	A	A	A	-	A	A	A	-	-	A	A	A	-	-	A	A	A	A	A	A
199	004-040-2 ｾｲｹｲ OUT	ｼ●ﾌ● ﾁ●ｺ		15-6043	040-4	A	-	A	-	-	-	A	-	A	A	A	A	-	A	A	A	-	-	A	A	A	-	-	A	A	A	A	A	A
200	008-060-2 ｾｲｹｲ OUT	ﾀ●ｳ● ﾌ●ﾖ		15-6044	040-5	A	-	A	-	-	-	A	-	A	A	A	A	-	A	A	A	-	-	A	A	A	-	-	A	A	A	A	A	A
201	000-010-0 ｻﾝﾌﾟﾙ W-6	ﾌ●ﾝ●ﾃﾞ ●ﾝ●ﾞ		15-6045 HB	041-1 ｷｮｳｾｲ	-	-	-	-	-	-	-	-	-	-	A	-	-	A	-	A	-	-	-	-	A	-	-	-	-	-	-	-	-	-
202	009-060-2 1ﾅｲｶ OUT	ﾑ●ﾝ● ﾏ●ﾖ●		15-7001	041-2												A		GLU		ｾﾝ														
203	003-070-0 1ﾅｲｶ OUT	ｱ●ｼ ●ﾏ●		15-7002	041-3												A		GLU-A		1B0'														
204	008-050-2 1ﾅｲｶ OUT	ﾏ●ｶ● ﾀ●ﾗ		15-7003	041-4												A		GLU		ｾﾝ														
205	000-000-0 1ﾅｲｶ OUT	ﾊ●ﾀ ●ﾊ●ﾛ		15-7004	041-5												A		GLU-1		ｾﾝ														

図4 ワークシート（大型自動分析装置用）



AAAAA ワ ー ク シ ー ト (11) AAAAA

ｹﾝｻﾋﾞ : 85.08.17
ﾜｰｸｼｰﾄ : 5020 H105[IP]

NO.	ｶﾝｼﾞｬNO. ｶ･ﾋﾞｮｳﾄｳ	ｶﾝｼﾞｬﾒｲ ﾅﾝﾀｲ	ｼｭﾋｭｳ	ｳｹﾌﾀNO. ｶﾝｼﾞｬｺﾒﾝﾄ	S-ID ｻﾝｲﾘｮｳ	5320 U-IP
001	007-040-4 ﾏｽｲｶ E-2	ﾏ●ｼ● ﾋ●ｽ●		16-8001	00-01	( )
002	006-020-7 ﾋﾌｶ E-2	ﾌ●ﾊ●ﾗ ●ﾂ●		16-8002	00-02 24Hﾆｭｳ	( )
003	006-080-4 2ﾅｲｶ OUT	ﾏ●ﾊ●ﾗ ●ﾄ●		16-8005	00-03	( )
004	003-050-4 2ﾅｲｶ OUT	ﾊ●ｶ●ｸ ●ﾂ●ﾖ●		16-8008	00-04	( )
005	004-050-9 2ﾅｲｶ W-5 800ML	ｲ●ｸ ●ﾖ●		17-3001	00-05 24Hﾆｭｳ	( )
006	005-080-1 1ﾅｲｶ E-5 623ML	ｶ●ﾋ● ﾊ●ｲ		17-3003	00-06 24Hﾆｭｳ	( )
007	005-040-2 1ﾅｲｶ E-5 300ML	ﾅ● ﾋ●ﾊ●		17-3004	00-07 24Hﾆｭｳ	( )
008	007-090-6 3ﾅｲｶ W-6 1600ML	ｱ●ｺ● ﾅ●ﾋ●		17-3008	00-08 24Hﾆｭｳ	( )
009	007-030-9 3ﾅｲｶ W-6 750ML	ﾏ●ﾜ●ｷ ●ﾂ●		17-3010	00-09 24Hﾆｭｳ	( )
010	001-040-2 3ﾅｲｶ W-6 1200ML	ﾙ●ｼ●ｸ ●ﾅ●		17-3011	00-10 24Hﾆｭｳ	( )

図5 ワークシート（用手法による尿蛋白検査用）

行いながら結果の記入が出来るよう検査依頼マークに”（ ）”を用いた（図５）。このように項目によって、あるいは運用によって変化し得るパラメータは、汎用性を持たせるためテーブル設定した。テーブル設定してあるパラメータには、各ワークシートに対応する検査項目コード、ワークシート印字用の項目略号、検査依頼マークおよびマークの打ち出し位置、精度管理用サンプルの挿入位置、さらに印字のための行コントロール情報などがあり、ワークシートの新規設定や変更がテーブルメンテナンスにより行える。いずれのワークシートも、特殊な材料の場合にはその材料名を印字する。また、ＨＢｓ抗原陽性や梅毒血清反応陽性など取り扱いに注意を要する検体ではその旨のコメントを印字する。

現在、血液検査部門１０種類、血清検査部門１８種類、生化学検査部門２２種類、緊急検査部門１種類の計５１種類のワークシートを設定し、運用している。

２．検体ラベルの発行

検体ラベルは、部門、受付日、ラベルＮｏ．および発行の対象となる受付Ｎｏ．の範囲を指示して発行する。ラベルＮｏ．は複数指定することが出来、さらに同一の画面で再発行が行える（図６）。

検体ラベルには２３ｍｍ×２３ｍｍのシールタイプのものを用い、１シートが横１０枚、縦５枚の連続タイプのものを設計した。印字は水平ピッチが１インチ１２文字、垂直ピッチが１インチ８行となるようにソフトウェアで制御し、１枚のラベルに横１１文字、縦７行が打ち出せるようにした。ラベルには受付日、受付Ｎｏ．、診療科名、患者名のほかに、ワークシートと同様に、取り扱いに注意を要する検体にはその旨のコメントを印字し、血清鉄やＣＰＫなどコンタミネーションや失活に注意を要する検査の依頼がある場合には、その項目名を印字する。また、糖負荷試験など１依頼に対して複数の検体が存在する場合には、ラベルに採血時間を印字する（図７）。さらに、血液検査の末梢血液像やペルオキシダーゼ染色などの検査では、臨床への返却用、検査部での保存用など各検査ごとに同一検体の血液塗抹標本を複数作成するため、このプレパラートに貼付するラベルは同じものを３枚ずつ発行する。また、最下段に染色法の略号を印字する（図８）。このように検体ラベルのフォーマットには一般検体に使用するもの、負荷検体に使用するもの、およびプレパラートに使用するものの３タイプを設定し、汎用性を持たせるため各ラベルの印字情報や発行枚数などのパラメータはテーブル設定とした。

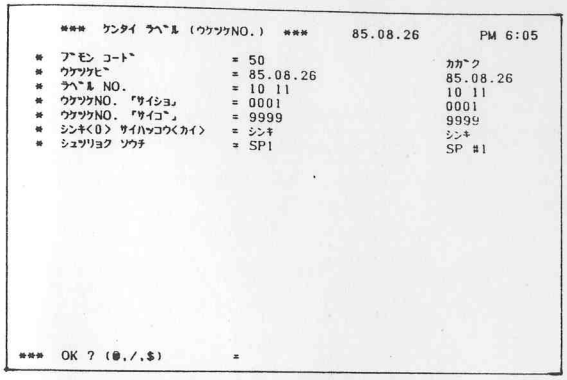


図６　検体ラベル発行指示画面

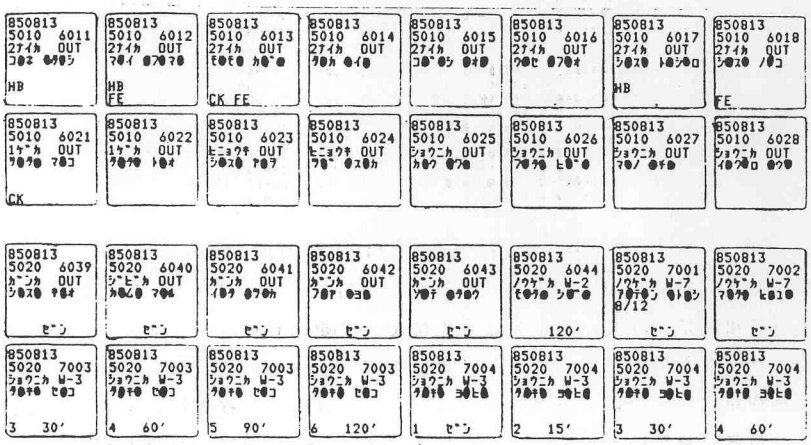


図７　検体ラベル（一般検体用；上段２列，負荷検体用；下段２列）

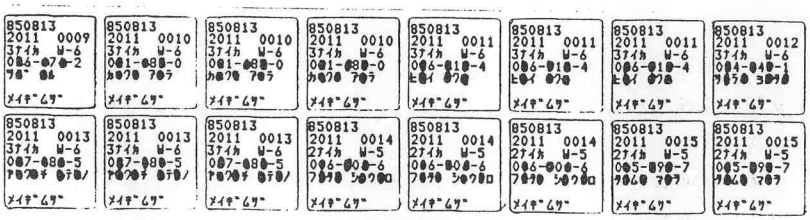
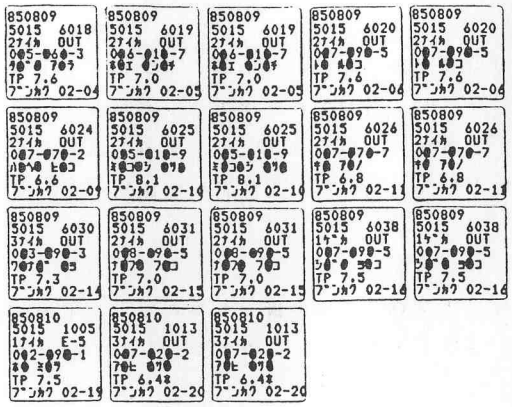


図８　検体ラベル（プレパラートラベル）



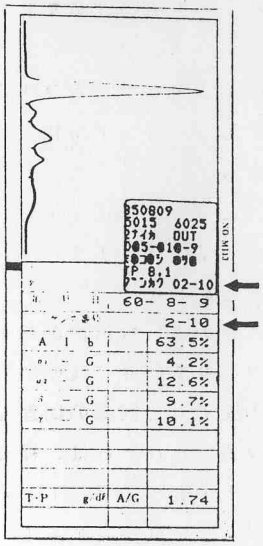


図9　検体ラベル（血清蛋白分画泳動像貼付用）

←：サンプルＩＤ

★★★★★　ﾌﾞﾝｼｭ　ﾌﾞﾝﾁｭｳ　ﾘｽﾄ（02）　★★★★★

ﾌﾞﾝﾁｭｳﾋﾞ　：85.08.17
ﾌﾞﾝﾁｭｳﾘｽﾄｺｰﾄﾞ　：5001　ｶｶﾞｸ 1

NO.	ｶﾝｼﾞｬNO. / ｶ･ﾋﾞｮｳﾄｳ	ｶﾝｼﾞｬﾒｲ / ｹﾝﾀｲ	ｼｷｭｳ / ｸﾍﾞﾂ	ｳｹﾂｹNO. / ｶﾝｼﾞｬｺﾒﾝﾄ	ﾗｯｸNO.-ﾎﾟｼﾞｼｮﾝ / ｻﾞｲﾘｮｳ	1ﾚﾂ	2ﾚﾂ	3ﾚﾂ	4ﾚﾂ
001	0●7-●4●-4	ﾏ●ﾝ● ﾋ●ｽ●		16-6001	105-A1	---	---	0.3	0.9
	ﾏｽｲｶ E-2								
002	0●2-2●5-●	ﾅ● ﾊ●ｴ		16-6002	105-A2	---	---	---	1.0
	ｼｶ E-2								
003	0●4-●9●-8	ｶ●ﾂ● ﾏ●ｺ●		16-6003	105-A3	---	---	---	0.9
	3ﾅｲｶ W-6								
004	0●5-●8●-2	ｱ●ｱ● ﾃ●ﾖ		16-6004	105-A4	---	---	---	0.8
	ｼﾞﾋﾞｶ OUT								
005	0●5-●7●-9	ﾋ●ｲ●ﾞ● ｲ●ﾒ		16-6005	105-A5	---	---	0.3	0.8
	1ｹﾞｶ W-4								
006	0●3-●1●-7	ﾏ●ﾀ● ﾚ●ｺ		16-6006	105-A6	---	---	---	0.8
	2ｹﾞｶ OUT								
007	0●3-●8●-6	ｳ●ﾀ● ｽ●ｺ		16-6007	105-A7	---	---	---	0.9
	2ｹﾞｶ OUT								
008	0●2-●4●-6	ｲ●ｳ ●ﾁ●ｴ●ﾝ		16-6008	105-A8	---	---	---	0.9
	2ｹﾞｶ OUT								
009	0●5-●2●-5	ｱ●ﾅ●ｼ ●ｮ●ｲ●		16-6009	105-A9	---	---	---	0.9
	2ｹﾞｶ OUT								

図10　分取分注リスト

さらに、ラベルの発行には、受付Ｎｏ．をインデツクスにする上記のタイプ（図７、図８）のほかに、ワークシートの連番をインデツクスにするタイプを設定した。その１例として図９に血清蛋白分画の泳動像に貼付するラベルを示す。ラベルにはサンプルＩＤを印字し、泳動像に印字されているサンプルＩＤとの照合を行いやすくしたほか、血清蛋白など関連する項目の検査結果を印字し、泳動像の解釈の参考となるよう配慮した。

現在、血液検査部門４種類、血清検査部門２種類、生化学検査部門１２種類の計１８種類のラベルを設定し、使用している。

３．分取分注リストの発行および分取分注指示情報の転送

分取分注リストは、一旦分離した血清をさらに分析装置や担当者別に分注するためのもので、部門、分注日、リストＮｏ．、対象とする受付期間を入力して発行する。分取分注

リストには、装置や担当者に対応する各分注列とそれぞれの分注量が打ち出されている（図１０）。分注量を計算するために必要な検査項目毎の血清量およびデッドボリュームはテーブル設定されている。

一方、ＨＥＵ－５０１で自動分取分注を行うため、ホストコンピュータのＣＲＴから指示し分取分注指示情報（分取分注リストに出力された検体ラックＮｏ．、検体セット位置、受付Ｎｏ．、分注列、分注量などの情報）をＭ３４３に転送する。ＨＥＵ－５０１では送られた分取分注指示情報により分取分注が行われ、同時に検体識別のため受付Ｎｏ．を分注カップに印字する。

〔まとめ〕

本学のような新設医科大学では、年次的に機器の購入や検査項目の増加が行われるため種々の作業形態に流動的な部分が多い。また、従来の検査室においても、機器の更新や運用の変更など同様の問題をかかえており、コンピュータ化を行う場合この点の工夫が必要となる。われわれは、検査前処理の設計にあたり、ワークシートや検体ラベル、分取分注リストの発行に関して必要な情報をテーブル設定することで、この問題の解決を図った。テーブルには該当する項目の情報のほかにリスト内の文字の印字位置や行間隔など、リストのフォーマッティングに関する情報も設定されており、新設や変更に対して柔軟な対応が行えた。これら帳票の複数指定やプリンタの指定もテーブル化し、ワンタッチ指定が行えるなどの能率化も図られた。また、ワークシート、検体ラベル、分取分注リストには受付日、受付Ｎｏ．と同時に患者名が打ち出されているため、検体の識別が明確になり、特に検体ラベルの発行は検体取り違えの防止に役立っている。さらに”ＨＢ”などのコメントの印字は、それぞれの担当者に取り扱いの注意を促し、ＨＢ（肝炎ウイルス）感染事故の防止にも役立っていると思われる。

〔文　献〕

１）長谷川　俊雄，他：日本臨床検査自動化学会誌，９（補）：８２，１９８４．
２）長原　三輝雄，他：日本臨床検査自動化学会誌，９（補）：８３，１９８４．
３）長谷川　俊雄，他：第４回医療情報学連合大会論文集，４５０～４５３，１９８４．
４）長谷川　俊雄，他：臨床病理，３２（補）：３６７，１９８４．
５）井村　敏雄，他：日本臨床検査自動化学会誌，１０（補）：２４１，１９８５．

ＭＤ — ２　　第１２回日本マンプスユーザーズグループ学術大会（１９８５年８月）

福井医大病院における栄養管理システムについて

○本間　富士子、西　秋人、関山　明彦、佐藤　裕保

福井医科大学附属病院　医事課栄養係
福井県吉田郡松岡町下合月２３

当院においては病院開設時より、医学面・看護面と共に栄養管理の充実がかかげられて、これらを三大支柱として疾病治療が行われている。
栄養管理業務は、適時適温給食を基本に患者の病状に応じた適切な栄養給与のため、ＣＰＵを用いてシステム管理を行っている。また、より効果的に行うための基礎調査にも利用している。
一方、栄養指導を効果的に実践するためにも、積極的利用を行っている。

Ⅰ．はじめに

当院における栄養管理システムは、食数管理、献立作成，食材料の発注、在庫管理、調理配膳など、単に患者に適正な食事を給与するだけではなく、栄養指導や治療及び診療研究の協力などさまざまな方面に渡っている。その幅広い業務内容を、少ない人員で効果的に行うためコンピュータシステムを導入している。

Ⅱ．システムの概要

1)．食数管理

病棟から送られてくる　入院・退院・外出・外泊・転室・食事変更などに関する伝票類を、患者ＩＤ番号を用いて入力し個人別に行う。

2)．献立作成

約束食事として３５８種類あるが、そのすべての献立を入力・作成することは、時間及び容量の関係で不可能となるため、基本となる２４食種のみ入力・作成する。方法としては、基本食種をＡ～Ｆまでの６ブロックに分け、各ブロック毎に持つ献立カードの組み合せにより行う。献立カードには、季節がもたせてあるので献立に季節感が出るようになる。

3)．食材料の発注及び在庫管理

発注は国の会計法に基づき行い、月々の業務としては翌月の予定献立と予定人数から使用予定食材料の見積り合せを行い、納品業者の決定・契約を行う。また日々の業務として２日先の予定献立にあがってくる純使用量に、廃棄率をかけた１人当

りの食材料と予定人数から発注予定数量を出力し、それを基にして発注業務を行い。発注書も専用用紙にて出力を行っている。

在庫品は、入庫・出庫共に品目・数量などを入力し管理している。

4). 調理・配膳

以下の様な帳票類を出力する。

①、主食配食票

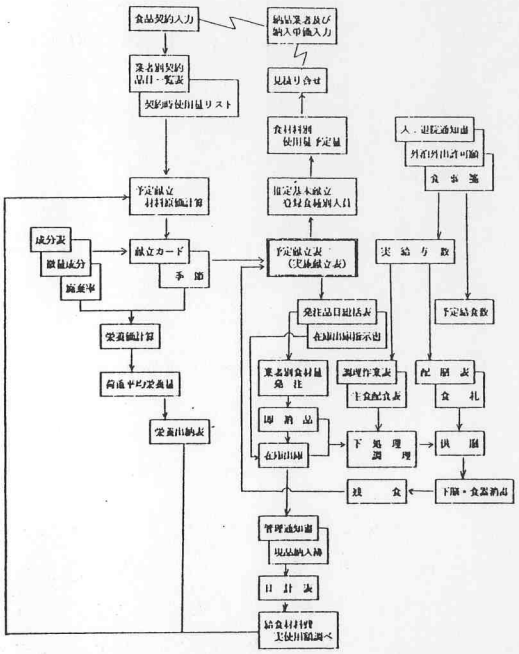
1日1回3食分を出力し主食の種類や数量ごとにそれぞれの人数が把握可能となっているもので、翌日の主食のセットに用いる。

②、調理作業表

毎食出力するもので、全食種のそれぞれの人数が、付加食塩量ごとに把握可能となっているもの。

③、食札

出力後の取扱い時間及び処理の軽減や無駄を省く目的で、配膳車に固定する形をとっているので、通常形態よりもかなり大きなものとすることが出来、ID番号や氏名から食種・主食・副食・飲料・間食の種類・形態及び量まで一目で把握可能となっている。



④、配膳表

部屋番号の小さい順にならべてある患者名簿で、各病棟一枚となっておりナースが配膳を行う時に用いる。

5). 病院給食運営上の必要関係帳票

様々な日報，月報，年報類を出力する。

6). 栄養指導

オ1・オ2・オ3内科，産科及び婦人科に入院中の患者と、その他の科に入院中及び外来にて栄養指導を受けられる患者に、栄養生活状況調査表の聞きとりを行う。これは、患者の社会環境、特に栄養学的環境をふまえて患者の特性を知ることを目的としているもので、この調査表により、患者の栄養指導以前における栄養摂取量、嗜好品摂取量，塩分摂取量、食事状況、運動状況及び休養状況の動向や、理想体重肥満度，基礎代謝量より、適正栄養所要量が把握可能となる。

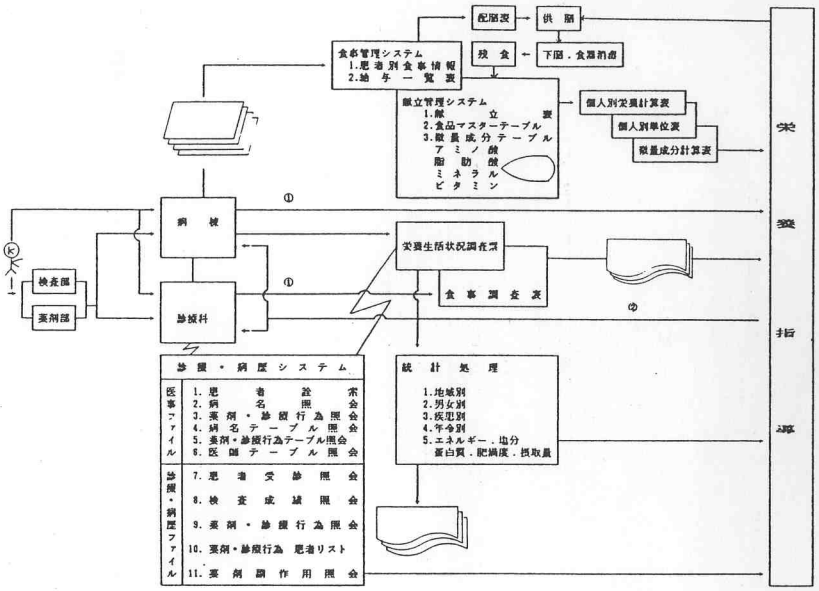
また、診療・病歴システムより、患者検索、病名照会、薬剤・診療行為照会、病名テーブル照会、薬剤・診療行為テーブル照会、医師テーブル照会、患者受診照会、

検査成績照会，薬剤・診療行為照会など各種データの参照を積極的に行っている。

さらに、実際に患者が摂取した食事量を入力して、個人別栄養計算表、個人別糖尿単位表、各種アミノ酸・脂肪酸・ミネラル・ビタミンなどの微量成分計算表などを出力させて、きめ細かな栄養指導を行っている。

7). 調査・研究

栄養生活状況調査表より入力したデータを用いて統計処理を行い、地域・性・疾患・年令・肥満度及び栄養摂取量の相互関係をつかみ、業務上の種々のデータとして活用している。



Ⅲ. 将来への展望

今後はプログラムの自己開発能力を身につけて、より現在のシステムのレベルアップを計り、医療スタッフの一員として、よりよい患者のための栄養管理システムの充実を目指していきたいと思う。

85ネン 0ガツ 1ニチ チョウサ　　*** セイカツシュウカンチョウサヒョウ ***　　〈〈 ガイライ 〉〉

カンジャ NO 02 8 2　　カンジャ シメイ ミ ワタヲ　　セイベツ F　　セイネンガッピ T 08ネン 0 ガツ 05ニチ ( )サイ

シュショク ヨシダークン　　ショクセイカツ A　　カゾクスウ 2ニン　　シンチョウ 148.5cm　　タイジュウ 66.0kg

シンリョウカ ダイ 3 ナイカ　　シッカン D コウシケッショウ　　リソウタイジュウ 48.5kg

タイヒョウメンセキ 1.37M**2　　キソタイシャリョウ 1036Kcal　　ヒシコウ +　　タイシボウリツ　　ヒマントン 36.1%

	エネルギー Kcal	エネルギー KJ	タンパクシツ g	シシツ g	トウシツ g	ナトリウム mg	カリウム mg	Na/K	エンブン g
* エイヨウショクリョウ	1554	6502	55.3	38.9	245.7				
* ニンシンジショクリョウ									
* シシエイヨウリョウ	1200	5021							
* セッシュエイヨウリョウ	1998	8361	59.7	42.1	345.7	4922	4356	1.1	12.5

* ショクヒンヘンセッシュリョウ	22.8							* シコウヒンセッシュジュウキョウ
サカナ	0.4							〈サトウ〉 25g
ニク	0.8		ミトリドクサイ	0.3	コクルイ	11.2		
タマゴ	0.7		ソノタヤサイ	0.7	イモルイ	3.0		〈アルコール〉
ダイズセイヒン	1.4	ミルクルイ 1.5	クダモノ	0.3	カシルイ	1.0	ユシルイ 1.5	ニホンシュ 0.0コップ
コウソウ	3.3	1.5		1.3		15.2	1.5	ビール 0.0ホン
								ヨウシュ 0.0ハイ
								ワイン 0.0ハイ
								ショウチュウ 0.0コップ
								ソノタ 1.0ハイ

タンパクシツエネルギーヒ	12.0%	* ウンドウシュウキョウ			*ウンドウリョウ	0.6
シシツエネルギーヒ	19.0%	ゲートボール	1カイ	60フン	16Kcal	〈タバコ〉
トウシツエネルギーヒ	69.2%					
トウブツセイタンパクシツヒ	33.7%					
トウブツセイシシツヒ	36.6%	* キュウヨウシュウキョウ			*キュウヨウテイド	
コクルイカロリーヒ	44.8%	[15-13テン キュウヨウシュウケツアツ]				
		[12-10テン フツウ]			14テン	
* ショクシセッシュジュウキョウ		[9- 7テン ヤヤフソク]				
		[6テンイカ フソク]				
ショクジカイスウ 3カイ						
アサ マイニチ						
ヒル シタク						
ユウ シタク						
カンショクカイスウ 1カイ						
ショクシジカン ハヤイ						
チョウリタントウ ホンニン						
ケンショクウイケン ケンサイ ナシ	カコ ナシ					

ＭＤ－３　　第１２回日本マンプスユーザーズグループ学術大会（１９８５年８月）

ＭＵＭＰＳの健診支援への人工知能的応用の検討

林恭平○、六反奈利子、森田益次、山口希、青池晟、川井啓市

京都府立医科大学　公衆衛生学教室
京都市上京区河原町広小路上る

［要約］

集団検診や人間ドックのいわゆる『健診』における計算機の利用として、人工知能・知識工学的な方法を用いて、受診検査データから問題のある受診者をスクリーニングする健診支援システムについて検討した。このシステムを作成するプログラム言語としてＭＵＭＰＳを用い、人工知能言語として注目されているＰｒｏｌｏｇを念頭にＭＵＭＰＳの人工知能への応用の可能性を検討し、その実用的応用が期待できることを確かめた。

［はじめに］

今後の医療の中で、集団検診や人間ドック等のいわゆる『健診』は予防医学の一つの方策として重要な位置を占めているが、その方法や評価などの検討されるべき事柄は多い。　その問題の一つとして、健診においてはその健診の目的に応じた各種の検査データを多数の受診者について処理せねばならず、そのために多くの医師及びその知識を必要とすることがある。今回はこの健診における多量のデータを処理することに関して、計算機の人工知能的な応用としての試みを行ない、医師の機械的な作業の軽減、処理の迅速化などに関しての計算機の健診支援の可能性を検討するものである。

［方法］

多くの健診において、大半の受診者の検査データは全ての検査項目で正常値範囲であり、その診断・指導内容は『健常』であり『問題なし』として処理され、また問題が考えられる受診者についても、施行された検査から判定する限りに於いては比較的簡単な医学的論理に基づいて診断されるのが殆どである。今回我々は、この様に比較的簡単な論理で処理が可能な受診者は計算機で人工知能的な手法を用いてスクリーニングし、残りのより複雑な推論から診断を行なう必要のある検査データを持った受診者についてのみ、医師が診断することを可能にするシステムを検討するものである。

そこで医師が健診結果を診断する過程を分析し、上記の目的の為のこのシステムに必要な機能を考えて見ると

１）各健診における検査項目に関する情報・知識の確保
２）その検査項目から診断を下す論理・知識の確保
３）各受診者の検査データの入手
４）以上の情報から推論を行ない診断する

のようなものが上げられ、このような機能を計算機に持たすためには従来の方法では困難であり知識工学・人工知能的な手法を用いる必要があると思われた。

そのためにはシステムを作成するプログラム言語が問題になるが今回はＭＵＭＰＳを使用した。

＜検査項目情報＞

検査項目は、実施される健診の目的を始め経済学的な観点等色々な理由で種々多様で、健診の評価の問題としては重要な問題であるが、実際に健診の診断を行なうためにはとりあえずａ）検査項目名、ｂ）その測定単位、ｃ）いわゆる正常値、ｄ）健診目的に応じた異常判定値、に関する情報・知識が必要となるので、各健診毎にこの情報を検査項目ファイル（グローバルファイル）として計算機に入力する機能を持たせた。（図－１）

```
Global name: ^KENSA
^KENSA(1)       ＧＯＴ ｜ K.U. ｜ 8-40 ｜ X>65 ｜
^KENSA(2)       ＧＰＴ ｜ K.U. ｜ 5-35 ｜ X>50
^KENSA(3)       ＡＬＰ ｜ K.-A.U. ｜ 3-13 ｜ X>20
^KENSA(4)       ＺＴＴ ｜ ZTT-U. ｜ 4.0-12.0 ｜ X>20
^KENSA(5)       γ－ＧＴＰ ｜ U/L ｜ 0-50 ｜ X>60
^KENSA(6)       総蛋白 ｜ g/dl ｜ 6.5-8.5 ｜ X<6.0
^KENSA(7)       コレステロール ｜ mg/dl ｜ 130-230 ｜ X>250 ｜ X<110
^KENSA(8)       中性脂肪 ｜ mg/dl ｜ 35-180 ｜ X>180
^KENSA(9)       βリポ蛋白 ｜ mg/dl ｜ 200-550 ｜ X>600
^KENSA(10)      血糖（無負荷） ｜ mg/dl ｜ 65-110 ｜ X>110
^KENSA(11)      Ｈｂ ｜ g/dl ｜ 14-18:12-16 ｜ X<14:X<12
^KENSA(12)      Ｈｔ ｜ % ｜ 40-48:34-42 ｜ X<40:X<34
^KENSA(13)      尿蛋白 ｜  ｜ - ｜ X'="-"
^KENSA(14)      尿潜血 ｜  ｜ - ｜ (X'="-")&(X'="+-")
```

図－１　検査項目グローバルファイルの内容例

＜診断論理情報＞

診断論理を計算機に与え、それによって推論・診断をどのように行なわせるかと言う点が重要な問題であるが、今回のような健診の場合、その健診で行なわれた検査の中で異常値と判定された検査項目の組合せによって各種の疾患を推論・診断するというのがその基本と考えられるので、異常判定検査項目を添字とした配列（グローバルファイル）にその組合せで推論される診断名を入力し、ＭＵＭＰＳ独特の木構造ファイルを作成し、このファイルを利用して推論を行なうことにした。（図－２）

```
^SINDAN1(ＡＬＰ1) ＡＬＰ高値
^SINDAN1(ＧＯＴ1) 肝炎
^SINDAN1(ＧＯＴ1,ＡＬＰ1) 肝障害
^SINDAN1(ＧＯＴ1,ＧＰＴ1) 肝炎
^SINDAN1(ＧＯＴ1,ＧＰＴ1,ＡＬＰ1) 肝障害
^SINDAN1(ＧＯＴ1,ＧＰＴ1,ＺＴＴ1) 慢性肝炎疑い
^SINDAN1(ＧＯＴ1,ＧＰＴ1,γ－ＧＴＰ1) アルコール性肝炎
^SINDAN1(ＧＯＴ1,ＺＴＴ1) 慢性肝炎疑い
^SINDAN1(ＧＯＴ1,γ－ＧＴＰ1) アルコール性肝炎
^SINDAN1(ＧＰＴ1) ＧＰＴ高値
^SINDAN1(ＧＰＴ1,ＡＬＰ1) 肝障害
^SINDAN1(ＧＰＴ1,ＺＴＴ1) 慢性肝炎疑い
^SINDAN1(ＧＰＴ1,γ－ＧＴＰ1) アルコール性肝炎
^SINDAN1(ＺＴＴ1) ＺＴＴ高値
^SINDAN1(コレステロール1) コレステロール高値
^SINDAN1(コレステロール1,βリポ蛋白1) 高脂血症
^SINDAN1(コレステロール1,中性脂肪1) 高脂血症
^SINDAN1(コレステロール1,中性脂肪1,βリポ蛋白1) 高脂血症
^SINDAN1(γ－ＧＴＰ1) γ－ＧＴＰ高値
^SINDAN1(中性脂肪1) 中性脂肪高値
^SINDAN1(中性脂肪1,βリポ蛋白1) 高脂血症
```

図－２　診断論理グローバルファイルの内容例

＜診断・推論＞

前でも述べたように基本の診断論理は木構造をしたファイルとして計算機に入力されているが、診断は受診者のデータとマッチする診断名をこの木構造を辿って捜していくことによって行なう。この条件に合うものをバックトラックしながら捜して診断に行き当たる流れはＰｒｏｌｏｇでは得意とする動作であるがＭＵＭＰＳでもＭＵＭＰＳの関数＄Ｎ（＄Ｏ）、＄Ｆ、＄Ｄ、及び間接指定Ｘと間接指定演算子@を利用して比較的簡単に作成出来た。

［結果］

図－３はこのシステムの基本メニューで、各健診毎に検査項目ファイル、診断論理ファイル、及びデータファイルを作ること及び診断を行ない診断名を出力するメニューから選択することになっている。

1．検査項目情報入力
2．診断論理入力
3．検査データ入力
4．診断出力
何番ですか ？

図－３　メニュー画面

図－４は検査項目の情報を入力する例で、検査名、単位、「正常値」、及び異常判定条件を順に入れていく。検査項目ファイルで異常判定は、多くの場合「正常値」より高値が問題になるが、検査項目によって低値も高値も問題になるような場合、両方の判定基準ををいれる。また判定基準を適時変えることによって診断の強弱も変えることが出来る。

1　ＧＯＴ	2　ＧＰＴ	3　ＡＬＰ
4　ＺＴＴ	5　γ－ＧＴＰ	6　総蛋白
7　コレステロール	8　中性脂肪	9　βリポ蛋白
10　血糖（無負荷）	11　Ｈｂ	12　Ｈｔ
13　尿蛋白	14　尿潜血	

検査番号と条件番号入力

```
検査項目番号　7
1.検査名，2.単位，3.正常値，4.判定条件1，5.判定条件2 .....

更に入力
 NO. 1 - コレステロール
 NO. 2 - mg/dl
 NO. 3 - 130-230
 NO. 4 - X>250
 NO. 5 - X<110
 NO. 6 -

コレステロール mg/dl          130-230          X>250          X<110
 OK
```

図－４　検査項目情報入力例

図－５は診断論理を入力している例で、その検査項目で異常と判定される検査の中で単独で診断が想定されるものから関連検査との組合せによるもの全て入れることが出来る。

図－６は診断結果を出力した例であるが、今回は診断とその根拠を出力したが、この出力内容は要求に応じて変化すべきと考える。

［考察］

医師の診断過程のような推論を知識工学・人工知能的手法を適応して計算機で行なわせる場合、その知識・推論をどのように表現するか及びどのようなプログラム言語を用いて処理するかと言うことが問題になる。

今回我々は、健診における医師の診断論理は図－２のグローバルファイルような枝分れ（木構造）でほとんど表現できると考えてその論理を計算機に構築した。この正当性は更に

```
診断論理ファイルは  ? SINDAN1

検査項目ファイルは  ?  KENSA

1  ＧＯＴ          2  ＧＰＴ          3  ＡＬＰ
4  ＺＴＴ          5  γ－ＧＴＰ       6  総蛋白
7  コレステロール

          検査番号と条件番号入力

検査番号  1 条件(高値- 1,低値 - 2 ) 1
検査番号  2 条件(高値- 1,低値 - 2 ) 1
検査番号  5 条件(高値- 1,低値 - 2 ) 1
検査番号
        診断  アルコール性肝炎

          検査番号と条件番号入力

検査番号  1 条件(高値- 1,低値 - 2 ) 1
検査番号  2 条件(高値- 1,低値 - 2 ) 1
検査番号  4 条件(高値- 1,低値 - 2 ) 1
検査番号
        診断  慢性肝炎疑い
```

図－５　診断論理入力例

検討する必要性は残されているが、このシステムを利用して、その結果を蓄積・フィードバックさせることにより、これまでの医学知識が逆に検討・整理されることが期待され、このシステムのもう一つの利点・効果と思われる。

プログラム言語としてこのような木構造の論理を扱う言語としてＰｒｏｌｏｇが注目され、本研究もこの言語を念頭に検討を行なったが、次の理由で今回はＭＵＭＰＳを採用した。

１）ＭＵＭＰＳが人工知能言語としてどの程度可能性が有るかを検討するため。

２）ファイルの処理や細かい制御においてＭＵＭＰＳの方が容易であると思われたため。

１）については、Ｐｒｏｌｏｇ特有の機能とされているａ）パターンマッチング、ｂ）バックトラッキングのような動作がＭＵＭＰＳ特有の関数＄Ｄや＄Ｆなどを用いてＭＵＭＰＳでも可能であり、Ｐｒｏｌｏｇの得意とされるａ，ｂによる三段論法処理も間接指定Ｘや間接指定演算子@を用いＭＵＭＰＳでも推論的な処理が可能であると考えられる。

２）については、従来の言語に慣れているものにとってカットの制御等Ｐｒｏｌｏｇ独特の表現に不慣れで、それに切替える等に困難な点があった。

```
性(男-1,女-2) 1   年齢 50
ＧＯＴ  : 70
ＧＰＴ  : 60
ＡＬＰ  : 10
ＺＴＴ  : 30
γ－ＧＴＰ : 20
総蛋白  : 7
コレステロール  : 260
中性脂肪 : 200
βリポ蛋白 : 500
血糖（無負荷）  : 100
Ｈｂ  : 15
Ｈｔ  : 45
尿蛋白 : -
尿潜血 : -

慢性肝炎疑い :      ＧＯＴ : 高い
                    ＧＰＴ : 高い
                    ＺＴＴ : 高い

高脂血症 :          コレステロール : 高い
                    中性脂肪 : 高い
```

図－６　診断例　　左：入力データ
右：その診断出力

［結語］

ＭＵＭＰＳを用いて健診を人工知能的な手法で支援するシステムを作成・検討すると同時に、ＭＵＭＰＳの人工知能への応用の可能性を検討し、その実用的な応用が期待できることが確かめられた。

マイクロマンプスによる眼科救急外来患者の統計的観察処理

木村一元，西村雅晴＊，小暮文雄＊＊，関　亮＊＊
獨協医科大学　総研ＭＥ室，＊)公衆衛生学，＊＊)眼科
栃木県下都賀郡壬生町北小林 880

要　　約

眼科救急外来の診療記録を資料として受診患者の状況分析を行なった。入力データの内、症状・診断名，状況，原因，処置・経過は、その内容状況のより詳細な分析を行なうため、コード化せずそのままの形（日本語）で入力した。その事により、データ構造のより深い解析が可能となった。

はじめに

救急医療の需要は近年急激に増し、それに伴って救急医療に対する関心が急速な高まりを見せている。眼は耳と並んで我々が外界と交流する重要な器官であり、日常生活におけるその外傷原因と疾患状態の関係を究明することは眼科外傷防止対策の上で必要不可欠である。しかしながら、その原因、疾患の状況は様々であり、さらに診断記録の記載も担当医師により異なりなかなかコンピュータを用いての統一的な処理が困難である。そこで、我々は当院眼科救急患者の診療記録（S.56～S.58）を資料として、記載された診断名あるいは症状名と発生の原因並びに発生時の状況との関連を検討した。

今回は、この検討・処理において発生した病名（診断名・症状名）の入力時の問題点ならびに状況、原因、処置の取り扱いについて報告する。

対象　及び　方法

昭和 56 年 1月から昭和 58 年 12 月までの３年間の当病院眼科救急外来を受診した 1,124名（男 810名，女 314名）の診療記録を資料として、13項目（性、年令、来院年、来院月日、来院時間、来院手段、地域区分、県内住所、職業、症状・診断名、状況、原因、処置・経過）をマイクロコンピュータに入力した。なお、年令，来院年月日，時間はそのままの値を、性，来院手段，地域区分，県内住所，職業はコード化し、症状・診断名，状況，原因，処置・経過はコード化せずそのままの形（日本語）で入力した。図１にデータ入力画面、図２に入力データのグローバルファイルを示す。

入力、解析の実際

コンピュータへの入力は、診療記録から直接ではなく一度、計算機入力用紙に必要項目を転記した後に行なった。コード化は、それぞれ、性（1:男，2:女）、来院手段（1:本人のみ，2:家族同伴，3:

救急車，4:その他，9:記入なし）、地域区分（1:県内，2:茨城県，3:埼玉県，4:千葉県，5:東京・神奈川，6:その他，9:記入なし）、県内住所（49の市町村，99: 記入なし）、職業（1:幼児，2:学生，3:会社員，4:自営業，5:公務員，6:農林，7:主婦，9:不明）とした。また、コード化を行なわなかった項目については、その入力文字の出現頻度を調べ改めて概念のまとめを行なった。

1．眼科救急患者の性・年度別頻度、性・年令別頻度、年令・来院手段別頻度、曜日・受診時間別頻度：　これらの頻度表はその項目を添字とする事で比較的容易に得られる。ここで、問題となるのは年令区分をどの様にするかと曜日別（日曜・祭日，土曜日，平日）頻度を来院年月日から如何に得るかであるが、年令は生のデータそのままでなく 10 才毎で区分する様にプログラムし、また曜日別については万年暦のプログラム(2) を改良し、年月日を入力すれば曜日ならびに祝祭日を返すプログラムを作製した（図３）。

曜日・受診時間別頻度：救急外来は、平日，土曜は昼間12時から翌朝 9時まで、日曜，祝祭日及び年末，年始は終日、当直医制を取り行なっている。その中でも、日曜，祭日の患者数は全体の 58.1% を占め多い。また来院時間別で見ると準夜間診療が 42.6%で多い。また、夜間についての曜日別頻度を見ると平日が一番多いが、これをそれぞれの平均日数で除ると日曜，祭日0.81人／日、土曜日0.61人／日、平日0.36人／日となる。

2．症状・診断名，状況，原因，処置・経過：　これらの項目は単純には集計できない。データ入力時にカテゴライズ（コード化）を行なっていれば簡単に集計が可能であるが、この様な手法を取るとある一方向からだけの解析となり実際のデータ構造（症状，状況，原因，経過）のより深い解析が困難である。そこで、我々は出来る限り診断記録に忠実に記載内容を入力する事を試みた。しかし、記載内容には種々様々な表現（例えば、同一症状の別な表現、状況と原因が一つにまとまっているもの、同一状況の名詞的表現と動詞的表現、等）があり、これらをコード化せず入力する事は人数分の異なったデータを扱う事となり、集計の目的から外れるのではないかと云う危惧はあった。

図４は症状・診断名の出現頻度順のグローバルリストの一部である。第１ノードが出現頻度で、そのデータはその様な症状・診断名が何種あるかを示す。第２ノードは種のシーケンシャル番号である。この図からは、１回のみのデータが 260件、２回現われたデータが 31 件あり、複数の症状・診断名の記載がある事が解る。

図５は受傷原因の出現頻度順のグローバルリストの一部である。第１ノードが出現頻度、データが異なった原因が何種類あるかを示す。第２ノードは種のシーケンシャル番号である。この図から、１回だけ使われた受傷原因の表現が 237件、２回使われた表現が 34 件ある事が解ると同時に、受身的表現や状況を示す言葉まで種々様々な記載表現がある事が解る。

3．疾患別順位、受傷原因別疾患順位：　これらは、２．で述べた各項目の記載表現の出現頻度順リストを基に改めて概念の分類、グループ化を行ない、得たものである。グループ化は、順次、入力データを画面に表示し対話形式により行ない、新しい分類によるデータをグローバルに追加した。図６に追加データのグローバルファイルを示す。

疾患別順位（表２）：　救急患者を疾患順に分けさらに男女別に示した。１位は角膜剝離及び角膜びらんで以下、角膜・結膜異物、角結膜炎、結膜浮腫となっていた。これら疾患の上位三つは、山城・関(1) の報告と同じであった。また、例数は少ないが急激な視力低下をきたす球後視神経炎や原田氏病なども見られた。

受傷原因別疾患順位（表３）：　表２の疾患について、受傷原因又は受診時の状況との関連を示す。

この場合の分類も発生場所（家庭，仕事場，等）で分けるか状況（ＣＬ装着中，けんか，又は打撲，裂傷，穿孔，等）で分けるか立場により異なるが、今回は原因、状況の様子を大まかに把握する事とし表の如くの分類とした。これによると、工業などの作業中に受傷した者が 181名(28.5%) と最も多く、続いて遊戯中(21.5%)、運動中(13.5%)、ハードコンタクトレンズ（ＨＣＬ）又はソフトコンタクトレンズ（ＳＣＬ）装着中(10.5%) の順であった。

ま　と　め

１．当大学病院眼科救急外来受診患者（S.56.1～S.58.12,３年間，1,124 名）の受診記録を基にマイクＭＵＭＰＳを用い、その状況分析を行なった。

２．入力データ 13 項目の内、症状・診断名，状況，原因，処置・経過はコード化せず、出来るだけそのままの形（日本語）で入力した。

３．これらの生のデータの出現頻度を調べ、どの様な概念で分類するかを決定した。

４．順次、入力データを画面に表示し対話形式により新しい分類によるデータをグローバルに追加した。

５．その後に、改めて種々の頻度分布を得た。

参　考　文　献

1)山城敏嗣・関　亮：独協医科大学眼科救急外来報告，眼科臨床医報，76:3，51，1982

2)大槻陽一：万年カレンダー，第13回マンブスソフトウェア研究会資料，13-3，1979.7

```
1.性　別 [    ]        2.3.年　齢 [     才     月]

4.来院年 [    ]  5.来院月 [    ]   6.来院日 [    ] 7.来院時間 [    ]

8.手　段 [    ]  9.地　域 [    ]  10.市町村 [    ] 11.職　業 [      ]

12.症　状 [                                        ]

13.原　因 [                                        ]

14.状　況 [                                        ]

15.処　置 [                                        ]

Case Number 1759                  確　認 [ ] OK->1.CANCEL->2
```

図１．　データ入力画面

順位	疾患	計 実数	%	男 順位	実数	女 順位	実数
1	角膜剝離・びらん	284	25.3	1	180	1	104
2	異物（角膜・結膜）	176	15.7	2	164	5	12
3	角結膜炎	124	11.0	3	68	2	56
4	結膜浮腫	53	4.7	5	37	3	16
5	結膜下出血	52	4.6	4	38	4	14
6	外傷性虹彩炎	45	4.0	6	35	7	10
7	眼球打撲	41	3.6	7	32	8	9
8	眼瞼裂傷・擦過傷	39	3.5	8	31	9	8
9	外傷性前房出血	37	3.3	7	32	11	5
10	火傷・熱傷	34	3.0	10	23	6	11
11	穿孔性外傷	32	2.8	9	27	11	5
12	電気性眼炎	31	2.7	8	31	-	0
13	霰粒腫	25	2.2	12	15	7	10
14	アレルギー性結膜炎	23	2.0	11	16	10	7
15	角膜潰瘍	22	2.0	13	12	7	10
16	緑内障	12	1.1	15	7	11	5
17	網膜振盪症	8	0.7	16	6	13	2
17	眼球破裂	8	0.7	14	8	-	0
17	角膜裂傷	8	0.7	16	6	13	2
18	硝子体出血	7	0.6	17	5	13	2
19	網膜剝離	6	0.5	17	5	14	1
19	麦粒腫	6	0.5	19	3	12	3
20	球後視神経炎	5	0.4	18	4	14	1
20	ブドウ膜炎	5	0.4	20	2	12	3
21	網膜動脈閉塞症	3	0.3	20	2	14	1
22	原田氏病	2	0.2	-	0	13	2
23	網脈絡膜炎	1	0.1	21	1	-	0
24	眼窩骨折	1	0.1	21	1	-	0
30	その他	34	3.0	-	19	-	15
	合　計	1,124	100.0		810		314

表２．　疾患別順位

順位	疾患名	実数	運動	家庭	工業	遊戯	CL	交通	けんか	紹介	手術後	その他	不明
1	角膜剝離・びらん	284	7	29	25	50	49	6	1	1	8	1	107
2	異物（角膜・結膜）	176	2	9	72	12	1	-	-	1	11	-	68
3	角結膜炎	124	1	8	6	13	8	-	-	-	-	-	88
4	結膜浮腫	53	-	3	-	12	-	-	-	1	-	-	37
5	結膜下出血	52	7	5	3	7	-	-	1	-	-	-	29
6	外傷性虹彩炎	45	21	-	6	3	2	-	2	-	-	-	11
7	眼球打撲	41	13	3	2	9	-	4	5	-	-	-	5
8	眼瞼裂傷・擦過傷	39	4	4	2	12	-	6	-	1	-	-	10
9	外傷性前房出血	37	20	2	4	5	-	1	-	-	-	-	5
10	火傷・熱傷	34	-	7	14	5	-	-	-	-	-	1	7
11	穿孔性外傷	32	2	2	8	6	-	7	-	3	-	-	4
12	電気性眼炎	31	-	-	31	-	-	-	-	-	-	-	-
13	霰粒腫	25	-	-	-	-	-	-	-	-	7	-	18
14	アレルギー性結膜炎	23	1	1	-	-	-	-	-	-	-	-	21
15	角膜潰瘍	22	-	-	2	-	3	-	-	-	1	-	16
16	緑内障	12	-	2	-	-	-	-	-	1	2	-	7
17	網膜振盪症	8	4	-	1	-	-	-	1	-	-	-	2
17	眼球破裂	8	1	1	1	-	4	-	1	-	-	-	-
17	角膜裂傷	8	-	2	2	1	-	1	-	-	-	-	2
18	硝子体出血	7	1	-	-	-	-	1	-	-	-	-	5
19	網膜剝離	6	2	-	1	-	-	-	-	-	-	-	3
19	麦粒腫	6	-	-	-	-	-	-	-	-	2	-	4
20	球後視神経炎	5	-	-	-	-	-	-	-	-	-	-	5
20	ブドウ膜炎	5	-	1	-	-	-	-	-	-	-	-	4
21	網膜動脈閉塞症	3	-	-	-	-	-	-	-	1	-	-	2
22	原田氏病	2	-	1	-	-	-	-	-	-	-	-	1
23	網脈絡膜炎	1	-	-	-	-	-	-	-	-	-	-	1
24	眼窩骨折	1	-	-	-	-	-	1	-	-	-	-	-
30	その他	34	-	-	1	3	-	-	-	-	4	-	26
	合　計	1,124	86	80	181	138	67	27	11	9	35	2	488

表３．　受傷原因別疾患順位

```
1021
-     1   = 1/4.0/5/9/14/12/9/1/6/1
-     12  = 眼球打撲
-     13  = 転倒
-     14  =
-     15  =
-     22  = 眼球打撲
-     23  = 遊戯中
-     24  = 打撲　その他
1023
-     1   = 1/17.0/5/9/9/21/9/1/15/2
-     12  = 角膜裂傷
-     13  = 異物飛入
-     14  = 自転車を運転中
-     15  =
-     22  = 角膜裂傷
-     23  = 交通　自転車
-     24  = 飛入　異物
1026
-     1   = 1/23.0/5/9/6/16/9/1/12/2
-     12  = 眼球打撲
-     13  = ボールが目に当る
-     14  = 野球中
-     15  =
-     22  = 眼球打撲
-     23  = 運動中　野球
-     24  = 打撲　ボール
```

図６．　追加データのグローバルファイル

時刻	昼間 9～12	昼間 12～17	夜間 17～24	夜間 24～ 9	合計　％
日曜・祭日	241	237	133	42	653 (58.1)
	(36.9)	(36.3)	(20.4)	(6.4)	(100.0)
土曜日		44	74	17	135 (12.0)
		(32.6)	(54.8)	(12.6)	(100.0)
平日		22	270	44	336 (29.9)
		(6.5)	(80.4)	(13.1)	(100.0)
合　計	241	303	477	103	1,124 (100.0)
％	(21.4)	(27.1)	(42.6)	(8.9)	(100.0)

表１．　曜日別・受診時間

```
-     253   =  網膜裂孔　硝子体出血
-     254   =  薬物性角膜腐蝕
-     255   =  薬物性眼炎
-     256   =  乱視　角膜障害
-     257   =  流行性角膜炎
-     258   =  緑内障　角膜白斑
-     259   =  緑内内障発作
-     260   =  老人環
2   = 31
-     1   =  異物除去後
-     2   =  角結膜炎
-     3   =  角膜びらん　球結膜下出血
-     4   =  角膜炎
-     5   =  角膜化学熱症
-     6   =  角膜擦過症
-     7   =  角膜上皮剝離　前房出血
-     8   =  角膜深層異
-     9   =  角膜深層異物除去後
```

図４．　症状・診断名の出現頻度順のグローバルリスト

```
-     229   =  薬剤混入
-     230   =  薬物
-     231   =  油がはねて入る
-     232   =  油飛入
-     233   =  溶網が目に入る
-     234   =  溶溶網混入
-     235   =  溶溶網剤を目に入れる
-     236   =  溶接中
-     237   =  輪ゴムが目に入る
2   = 34
-     1   =  ＳＣＬ長時間装用
-     2   =  ごみ飛入
-     3   =  オーツコンタクトレンズが入る
-     4   =  カガミデナフロートで切った
-     5   =  ガラス片が目に入る
-     6   =  コンタクトレンズ片が入る
-     7   =  ソフトコンタクトレンズが当たる
-     8   =  バッテリー破片が入る
-     9   =  ペンキマーカーが入る
```

図５．　受傷原因の出現頻度順のグローバルリスト

```
1074
-     1  = 2/4.0/3/5/20/21/2/1/12/1
-     12  = 薬物性角膜腐蝕
-     13  = コロンを目にスプレー
-     14  =
-     15  = 外来へ
1075
-     1  = 1/3.0/3/5/17/19/2/1/2/1
-     12  = 角膜火傷
-     13  = 燃えくずが入る
-     14  =
-     15  = 外来へ
1076
-     1  = 1/38.0/3/5/17/10/2/1/12/3
-     12  = 角膜エロージョン
-     13  = 作業中
-     14  = ごみ飛入
-     15  = 外来へ
1077
-     1  = 1/16.0/3/5/12/17/2/1/1/2
-     12  = 網膜剝離
-     13  = 視力低下
-     14  =
-     15  = 入院
```

図２．　入力データのグローバルファイル

```
KAL2        ; Calender In:YMD="YYYY/MM/DD", Out:YMD="X,yyy"
            D ISET
STR         I YMD'?.N1"/".N1"/".N S YMD="9,NON" QUIT
            S Y=+$P(YMD,"/",1),M=+$P(YMD,"/",2),D=+$P(YMD,"/",3),YY=Y,MM=M
            I M<3 S YY=Y-1,MM=M+12
GREG        S Z=YY+(YY¥4)-(YY¥100)+(YY¥400)+(26*(MM+1)¥10)+1,W=Z#7
            S DS=W-1 S:W=0 DS=6
            I (M=1)!(M=3)!(M=5)!(M=7)!(M=8)!(M=10)!(M=12) S MAX=31 G SET
            I (M=4)!(M=6)!(M=9)!(M=11) S MAX=30 G SET
            I M'=2 S YMD="9,NON" QUIT
URUU        S MAX=29 S:'(Y#400=0) MAX=28
            S:'(Y#100=0) MAX=29 S:'(Y#4=0) MAX=28
SET         I (D<1)!(D>MAX) S YMD="9,NON" QUIT
            S DX=(D+DS)#7 S:DX=0 DX=7 D HOL1,HOL2
            S YMD=DX_","_WEK(DX)
            QUIT
            ;
ISET        S HOL=$P($T(HD1)," ",2),HOL2=$P($T(HD2)," ",2)
            F I=1:1:8 S WEK(I)=$P("SUN/MON/TUE/WED/THU/FRI/SAT/HOL","/",I)
            Q
HOL1        S MD="/"_M_":"_D_"/"
            S:HOL[MD DX=8
            I DX=2 S MD="/"_M_":"_(D-1)_"/" S:HOL[MD DX=8
            Q
HOL2        S MD="/"_M_":"_D_"/"
            S:HOL2[MD DX=8
            Q
            ;
HD1         /1:1/1:15/2:11/3:21/4:29/5:3/5:5/9:15/9:23/10:10/11:3/11:23/
HD2         /1:2/1:3/4:23/12:29/12:30/12:31/
            ;
```

図３．　年月日から曜日を得るプログラム

マイクロコンピュータによる未熟児部データベースの作製

○土屋喬義、田中吾朗＊、木村一元＊＊、本間道

独協医科大学　　第一小児科，　＊）第二小児科
＊＊）総研ＭＥ室

要約

マイクロＭＵＭＰＳ　を使用して未熟児部用データベースシステムを開発した。このデータベースは拡張可能な辞書を持ち医学の早い進歩に対応できるような柔軟な構造を持つように工夫した。

1　はじめに

医療用の電子技術の進歩にともない、近年未熟児センターにもＩＣＵ的な治療法が持ち込まれ、治療成績が大幅に向上してきた。それと同時に仕事量が爆発的に増加し、医師およびスタッフは事務処理のために非常に多くの時間を割かなければならなくなってきている。

当院未熟児部でも1978年の開棟以来、毎年約 300名の入院があり、入院患者数は計 2000 名に近くなっている。そのため入院患者台帳よりの情報の検索や統計（例えば疾患毎、年度毎の死亡率など）の算出も限界を越えてきている。また対象となる患者は、体重が 500～5000ｇ、日齢が１ヶ月以内の新生児であり、問題となる疾病も極めて特殊であり、その種類も限られている。以上のごとき、数の多さと、質の均一性は、未熟児部の病歴整理をコンピュータ化しようと考える源となった。

しかし、BASIC や市販のデータベースを使用してプログラムを作成するのでは、この業務を十分満足させることは困難であると思われた。治療法や薬剤などは日進月歩であり、それらを次々と追加してもデータ処理が可能となるような柔軟なデータベースがどうしても必要である。このためデータはコード化せず自然語で入力するのが望ましいが、この際の入力ミス防止、分類法の相違などによる混乱の予防ならびに入力の省力化のために、種々の辞書を作成し、これにより入力することとした。診断名、治療法などは樹枝状構造をとっているため、これに合わせ辞書も樹枝状のデータ構造が必要である。また、これらの辞書は常にアップデートされる必要がある。

これらを満たすためには、プログラムをＭＵＭＰＳで開発するのが適当と思われた。そこで、我々はマイクロＭＵＭＰＳを使用して未熟児用の病歴データベースシステムの開発を試み、初期の目的をかなり達成できたと思われたのでここに報告する。

2　システム構成

ＣＰＵにはＮＥＣ－ＰＣ 9801 M2　384KB　RAM SYSTEM、　ＣＲＴにＰＣ－8853Ｎ、　プリンターにＰＣ－ＰＲ－201　を使用した。

言語はＭＳ－ＤＯＳ版ＵＣＤマイクロＭＵＭＰＳを使用した。また、日本語入力にはＶＪＥ－αを用いた。

3　プログラムの構成

現在は未熟児・新生児の病歴のみを扱っているが、将来は小児科患者全体、さらには病院の全入院患者を扱えるようにするための拡張性を考慮してある。このためプログラムは、1）患者のＩＤ番号を登録するＩＤ入力画面、2）母体および周産期の情報を扱っている周産期情報入力画面、3）診断、検査、治療などの入院情報ならびに退院後の追跡結果を入れた入院情報入力画面の３画面に分けられる。以下プログラムの流れおよび特長について説明する。

a　ＩＤ入力画面（図１）

我々の病院では患者は初診の順にＩＤカードを発行しているので、このＩＤ番号を第１ノードとした。

ＩＤ入力画面は、ＩＤカードに記載されている患者番号、性、生年月日と患者の住所を入院時に登録する。

b　周産期情報入力画面（図２）

この画面は、1）出生場所、2）母親の既往歴、3）妊娠分娩経過、4）出生時のデータと大きく４項目に分類され、さらにその下に 21 項目が存在する。

c　入院情報入力画面（図３）

入院は未熟児部では通常一回のみであるが複数の入院登録が可能なようにしてある。この画面では、1）未熟児部での入退院番号、2）入院時の情報（重症度、主訴など）、3）診断、4）治療、5）検査、6）予後、7）経過の追跡の７項目に分類され、その下に 27 項目が存在する。

4　辞書機能

以上の入力項目に対し、 20 の辞書を作製し入力操作の軽減と正確さを計った。これらの辞書は検索の効率を上げ、またデータの追加、削除を容易に行なうために、論理的な配列を持たせなければならず、作成に非常な努力を払った。以下作製した辞書のリストを示し、代表的な辞書のグローバルを示す。

1）	紹介産科医院名	2）	地名
3）	母親の既往歴（病名）	4）	妊娠合併症
5）	出産合併症	6）	蘇生法
7）	大学病院の臨床科名	8）	未熟児部までの搬送担当者
9）	未熟児部への搬送手段	10）	重症度
11）	血液型	12）	主訴
13）	診断名（病名）	14）	特殊検査項目
15）	治療法	16）	転帰
17）	転出先	18）	転出後の他病棟での転帰
19）	症状追跡（１才まで）	20）	症状追跡（１才以上）

これらの辞書は１～３階層のデータ構造を持ち、それぞれのカテゴリーを指定して行くと正しいデータを得ることが出来るようにした。またそのカテゴリーに目的のデータが存在しない時には新たなデータをその場で付け加えられるようになっている。

図４に診断辞書のグローバルの一部を示す。この辞書は１～３階層のデータ構造を持つ、第１階層は臓器別に、第２階層は疾患のカテゴリーを、第３階層には病名となっている。これによりユーザは上の階層より順に選んで行くことで正しい診断名に到達し、この診断名を病歴のグローバルにセットする。これにより診断名の混乱している病名についても重複を避ける事ができると考えている。

図５に主訴の辞書を示す。今回作成した辞書の大部分はこのように１階層の辞書である。

５　未熟児部データベース

このプログラムにて作成した患者データのグローバルを図６に示す。数字以外は全て辞書を通して作成した自然語にてこのグローバルは作られており、この自然語にて検索を行なうこととした。これにより将来より進んだ診断、治療、新しい病名などが出現しても対応できる自由度が確保されたと思われる。

６　退院サマリー

このプログラムのもう一つの目的は退院サマリーを自動作成することである。これにより現場での書類作成時間の短縮と、退院サマリーの脱落が防げるものと思われる。図７に出力図を示す。

７　結語

我々の作成したデータベースならびにシステムの概要と使用辞書について述べた。現在このシステムを用いて患者データを入力中であるが、ほぼ満足すべき実行結果が得られている。これは、第一に、本システムにおいて入力データの辞書化が容易であったからであると思われる。即ち対象患者が比較的均一でありまた、病名も限定されており、主訴についても成人に見られるような不定愁訴などなくカテゴライズし易かったためである。また第二には、本プログラムでは必要な事態が生じる度に、その場で辞書の拡張が行なえるようにしたこともその要因であったと思われる。

今後入力データの増加を待って、検索、統計処理ならびに種々の書類作成のプログラムを開発して行く予定である。

未熟児部 病歴台帳

```
-----------------------------------
I    独協医科大学病院 受診券      I
I                                 I
I    患者番号 18-7526-4  1) 性 M  I
I                                 I
I  2) 氏名   岩○一○              I
I                                 I
I  3) 年月日 S60-07-26.03:18      I
-----------------------------------
```

4) 患者住所　茨木県¥結城市¥

5) ＩＤ情報　　6) 入院情報

7) 次の患者　　8) 終了

何番を選びますか ？　番号を入力しなさい ！

図1　ＩＤ　入力画面

病歴整理 ID 情報
新入院患者の情報

ID No. #: 18-7526-4　　氏名 : 岩○一○　　生年月日:S60-07-26.03:18

1) 出生の場所　: 結城病院、大木昌衛/茨木県¥結城市¥結城¥633-1.TEL02963-3-416

1

2) 母親の既往歴 : /TOMOKO/25/O.Rh(+)/no//0¥0¥0¥0

3) 妊娠分娩経過 : 6/20¥oxygen over face only/10

4) 新生児情報　: 39/2960/50/31.5/O.Rh(+)/

5) 入院情報

6) 終了

どれを選びますか ？

図2　周産期情報入力画面

病歴整理　入院情報
入院時の Data

ID No. #: 18-7526-4　　氏名 : 岩○一○　　生年月日:S60-07-26.03:18
入院日　: 60-07-29.14:15　　退院日　: 60-08-12

1) 入退院（未熟児部）

2) 入院時の情報

3) 診断

4) 治療

5) 検査

6) 予後

7) 追跡

8) 終わり

どれを選びますか ？　Ｎｏ．：

図3　入院情報入力画面

```
0  = GENERAL INFORMATION
-    0  = Preterm (<37weeks)
-    1  = Post-term (>=42weeks)
-    2  = SFD (<-3/2SD)
-    3  = AFD
-    4  = LFD (>3/2SD)
-    5  = LBWI (<2500g)
-    6  = Giant baby (>=4000g)
-    7  = Observed for high risk prenatal / perinatal Hx.
-    8  = Normal baby
1  = RESPIRATORY disorders
-    0  = Nasal obstructions and stridors
-    -     0  = Choanal atresia
-    -     1  = Simple congenital laryngeal stridor
-    -     2  = Congenital laryngeal and tracheal stenosis
-    -     3  = Vascular ring
-    -     4  = Cysts and neoplasms of the larynx
-    -     5  = Neurogenic stridor
-    -     6  = Paralysis of vocal cords
-    1  = Aspiration syndromes
-    -     0  = MAS
-    -     1  = Aspiration during respirator care
-    -     2  = Aspiration after birth
-    -     3  = Atelectasis
-    2  = RDS
-    3  = Chronic lung diseases
-    -     0  = BPD
-    -     1  = Wilson-Mikity syndrome
-    -     2  = Chronic pulmonary insufficiency of prematurity (CPIP)
-    4  = Air leak and air trapping
-    -     0  = Pneumothorax
-    -     1  = Pneumomediastinum
-    -     2  = Pneumopericardium
-    -     3  = Pulmonary interstitial emphysema (PIE)
-    -     4  = Lobar emphysema
-    -     5  = Pneumoperitoneum
-    5  = PFC
-    6  = Apnea
-          0  = Apnea of prematurity
-          1  = Apnea of full term infant
-          2  = Sudden infant death syndrome (SIDS)
```

図4　診断名辞書グローバル

```
1  = low birth weight;prematurity;preterm
2  = jaundice
3  = respiratory distress
4  = cyanosis
5  = heart murmur
6  = apnea
7  = birth asphyxia
8  = convulsion;irritability;twitching
9  = fever
10  = hypothermia
11  = poor activity;not doing well
12  = vomiting
13  = poor sucking
14  = abdominal distension
15  = bleeding(hematoemesis;melena;petechiae)
16  = poor weight gain
17  = anomaly(head;extremity;abdominal wall)
18  = for observation due to high risk
19  = other reason
```

図5　主訴の辞書グローバル

```
18-7526-4  = M/岩○一○/S60-07-26,03:18
-    1  = 茨木県¥結城市¥
-    -    1  = 結城病院、大木昌衛/茨木県¥結城市¥結城¥633-1,TEL02963-3-4161
-    -    2  = /TOMOKO/25/O,Rh(+)/no//0¥0¥0¥0
-    -    3  = 6/20¥oxygen over face only/10
-    -    -    1
-    -    -    -    1  = toxemia;eclampsia
-    -    -    -    2  = severe anemia
-    -    -    2
-    -    -    -    1  = induced delivery
-    -    -    -    2  = use of forceps
-    -    4  = 39/2960/50/31.5/O,Rh(+)/
-    2
-    -    1  = 60-07-29,14:15/60-08-12
-    -    -    0  = 85-102/85-108/3,10/17
-    -    -    1  = 第二小児科/大原
-    -    -    -    1  = 結城病院、大木昌衛/茨木県¥結城市¥結城¥633-1,TEL02963-3-
4161
-    -    -    -    2  = ¥/mild
-    -    -    -    3
-    -    -    -    -    1  = vomiting
-    -    -    -    -    2  = fever
-    -    -    -    -    3  = cyanosis
-    -    -    2  = DIAGNOSIS
-    -    -    -    1  = Idiopathic vomiting
-    -    -    -    2  = Jaundice due to increased enterohepatic circulation
-    -    -    -    3  = Apnea of full term infant
-    -    -    3  = TREATMENT
-    -    -    -    1  = MEDICATION-2¥Antibiotics¥
-    -    -    -    2  = MISCELLANEOUS¥phototherapy¥
-    -    -    4  = 36.7/16.4/CT¥EEG¥G-I study/
-    -    -    5  = 感染はなかった/alive¥without any sequelae/home
```

図6　患者データのグローバル

```
DISCHARGE SUMMARY
Patient's name    : 岩○一○  M          ID No.                : 18-7526-4
Center No.        : 85-102              Discharge No.         : 85-108
                                        Gestational age       : 39 weeks
Date of birth     : S60-07-26,03:18     Birth weight          : 2960 gm
Date of admission : 60-07-29,14:15      Birth length          : 50 cm
Age on admission  : 3 day 10 hours      Head circumference    : 31.5 cm
Date of discharge : 60-08-12            Blood type of infant  : O,Rh(+)
Age on discharge  : 17 days             Blood type of mother  : O,Rh(+)
Maternal positive Hx. : no
Parity of mother      : 0-0-0-0
Complication of pregnancy : toxemia;eclampsia
                            severe anemia
Complication of delivery  : induced delivery
                            use of forceps
Apgar score : 6
Mode of resuscitation : 20¥oxygen over face only
Main reasons of admission : vomiting
                            fever
                            cyanosis
Onset and course of present illness :


Positive physical findings on admission :

Positive laboratory findings on admission :

Peak value of T.Bilirubin : 16.4 mg/dl
Special examination : CT
                      EEG
                      G-I study
Treatment : Antibiotics
            phototherapy
Diagnosis : Idiopathic vomiting
            Jaundice due to increased enterohepatic circulation
            Apnea of full term infant
Outcome on discharge : alive, without any sequelae
Transfered place     : home
Comment : 感染はなかった
Department of admission : 第二小児科
Physician : 大原
At discharge : weight      gm , head      cm
               Hb       gm/dl , T.Bilirubin      mg/dl
Medication on discharge :


Appointment on clinic visit :   /   /
```

図7　退院サマリーの出力

入院台帳プログラム・パッケージ

○林寺　忠　　和田行文　　西角　淳　　上坂邦夫
国立京都病院　小児科
京都市伏見区深草向畑町１－１（〒620）
℡　075-641-9161

パソコンを用いて、日本語マンプスによる入院台帳プログラムパッケージを作成した。
１．＾ＭＥＮＵというメニュールーチンを作成した。
２．誤入力を避けるために強力なサブルーチン（＾ＲＥＡＤ汎用入力サブルーチン）を用意。
３．カラー表示と漢字入力を可能にした。
４．項目の数はあまり多過ぎることがない。（入力に要する時間の節約）
５．患者に任意のインデックスをつけ、それをもとに出力可能にした。
６．記憶された患者のグローバルの構造はカタカナによるあいうえお順とした。
７．退院総括、入院時刻、紹介医、担当医、入院・退院の年月日の把握を主な目的とした。

昭和６０年１月より実用化することができたこのプログラムはマンプス言語を用いて作成されている。このプログラム・パッケージは他の目的に変更したり、入力項目の変更ができるが、その変更の幅を出来るだけ小さくするために記憶・改変の部分のみを３本のルーチン（＾ADMGET 、＾ADMFILE、＾ADMINP ）にまとめた。こうして、この部分のみを改変すればよいようにした。入院台帳プログラム・パッケージは未熟児室入院台帳と外来でのＢ型肝炎ワクチンフォローアップ台帳などにも応用出来るようにした。

１．メニュープログラム　＾ＭＥＮＵについて
これはプログラムの選択を容易にするためのものであります。スクリーンの上段に全てのキーの意味が表示されるようにしました。しかし、スペースがなかったためにテン・キーの意味だけ表示出来ませんでした。
必要なサブルーチンは＾READです。（これについては後で詳述します。）
その操作法はつぎの通りであります。
①ROLL UP ②ROLL DOWN は表示されているライン（行）をそれぞれスクロールアップまたはスクロールダウンさせます。１画面に１６行しか表示できない不便さを解消することが出来ます。１６行以内の場合にはこのROLL UP 、ROLL DOWN のキーは押してもビープするだけでその用をなしません。それはスクロールの必要がないからです。
③INS ④DEL は改頁に使用します。それぞれ次頁、前頁の意味です。ここでも最初と最後の頁では更に前または次の頁を改頁しようとするとビープが鳴ります。　このキーもROLL UP・ROLL DOWNの時に１６行という制限項目を意識して作られています。１６行以内の画面の場合はビープ音が鳴るだけで、改頁はされません（出来ません）。
⑤HOME⑥CLR はSHIFT+HOME、HOMEに相当します。それぞれカーソルが最上段、最下段に移動します。
⑦←⑧→はそれぞれ⑤と⑥のキーと全く同じ意味で使用されます。
⑨↑⑩↓はそれぞれカーソルを一段上、一段下に移動させます。カーソルが最上段または最下段に達した時にはスクロールが働きます。この場合も１６行の制限項目を意識して作られています。ビープ音で確かめて下さい。１６行以内の場合には、それぞれ最上段と最下段にてビープ音が鳴ります。
⑪テンキー（０－９）は押した数字の行までカーソルが移動します。
⑫ESC は２桁の数字を指定したい時に使用します。ESC を押すとプロンプトが表示されます。
注意すべきことは、この部分のみ先程のREADサブルーチンを使用している。必要であればこのメニュープログラムにREADサブルーチンを準備しておくと困らないと思います。

更に、このメニュープログラムを使用するためにはＣＰ／Ｍ８６でＫＥＹ．ＣＭＤを実行してＫＥＹ．ＴＢＬを次のように変更しておく必要があります。

```
RLUP  F21:<DC3>          RLDN  F22:<DC4>
 INS  F23:<DC2>           DEL  F24:<CAN>
HOME  F29:FD16<CR>       HELP  F30:A016<CR>
 CLR  F31:<RS>
```

ＣＰ／Ｍ８６ファンクションキーを以下にしめします。

```
          ******  ファンクションキー定義一覧表  ******

F01:V 1<CR>                              / F11:dir a:<CR>
F02:V 2<CR>                              / F12:dir b:<CR>
F03:V 3<CR>                              / F13:stat a:<CR>
F04:ZP<CR>                               / F14:stat b:<CR>
F05:D @$ZN<CR>                           / F15:stat a:*.*<CR>
F06:ZCLEAR<CR>                           / F16:stat b:*.*<CR>
F07:X ^%X<CR>                            / F17:pip a:=b:*.*[v]
F08:MUMPS<CR>                            / F18:pip b:=a:*.*[v]
F09:HALT<CR>                             / F19:pip<CR>
F10:D ^MENU<CR>                          / F20:asm86
 RLUP F21:<DC3>      /   DEL F24:<CAN>          /  DOWN F28:<LF>
 RLDN F22:<DC4>      /    UP F25:<VT>           /  HOME F29:FD16<CR>
  INS F23:<DC2>      /   <-- F26:<BS>           /  HELP F30:A016<CR>
                     /   --> F27:<FF>           /   CLR F31:<RS>
キーの番号 ？：
```

この^ＭＥＮＵは他の人がほかの目的に使用できるようにするため、カスタマイズ（ユーザーのための変更）の方法を記述します。

先ず、ＭＥＮＵのラベル　Ａに注目して下さい。$P($T(MENU),";",2)は最初のラベルMENUの部分を使っていますので、そのためこの部分を変更して下さい。

次に、プログラムの最後の部分にあるラベル PROG 以後から PEND の部分を注目して下さい。

イ）ラベルはルーチン名をいれる方が便利ですが、なくてもよろしい。

ロ）R(1),R(2),...,R(IX)にはルーチン名をいれる。

ハ）U(1),U(2),...,U(IX)には８文字（２バイト文字では４文字）以内の文字をいれて、ルーチンの作業項目とします。

ニ）T(1),T(2),..,T(IX)にはルーチンの作業内容を示す適当な言葉をいれて下さい。

ホ）ここで IX はルーチンの個数を意味し、PEND にそれを定義するわけです。

例えば、入院台帳のメニュープログラムには１４個のルーチンを示したかったので

```
PEND     S IX=14 Q  ;....
```

としました。

ヘ）最後に、PROG、PROG+1、PENDはその位置（順番）を変更しないで下さい。ましてそれらの行を削除しないで下さい。正常に働くかどうかの保証がなくなります。

２．ＲＥＡＤサブルーチンの改定版について

これは、ＣＯＳＴＡＲ　ＶのＲＥＡＤサブルーチンを日本語およびスクリーンを使うマイコン向けに改変・作成したものです。

【使用方法】

サブルーチンに入る前に次のような変数を定義します。

それぞれの変数を説明することにします。

①Ｘはスクリーンでの Ｘ座標

②Ｙはスクリーンでの Ｙ座標

③Ｚはプロンプトの後に表示されるヴィデオ（ＶＩＤＥＯ）の幅

をそれぞれ表わします。従って、Ｚは入力される（期待される）文字の数（CHRACTER NUMBER ）です。文字の数は１行８０文字未満として下さい。また、この数は２バイト文字は２文字として計算しています。

④ＰＲはプロンプト（応答）
ですが、スクリーンでは黄色い文字で表示され、キーボードからの入力を待ちます。ＰＲの後は１文字のスペースを入れてカーソルは止まります。カーソルの後にはＺの幅を持った白いヴィデオが現われます。
⑤Ｓは予めプログラムで用意された答え（サンプルとしての反応）です。
⑥ＰＭは入力された文字をプログラムが期待したものであるか（条件を満たしているか、あるいは条件にあっているか）をチェックします。
⑦ＱＳはＨＥＬＰキーを押した時に表示される文字行です。ＱＳは必ず８０文字未満の文字数である必要があります。８０文字以上のＱＳは正常に作動するかどうかの保証はできませんのでプログラム作成時に注意することが要求されます。また、ＱＳが表示される先頭の座標は（０，２２）［それぞれＸ、Ｙに相当］の位置です。これもここにＰＲなどを表示させる時に注意が要求されます。
以上が変数の説明です。

ところで、Ｄ＾READ　にてこのサブルーチンをコール（CALL）する前にＸとＹは必ず定義されていなければなりませんが、他のＺ、ＰＲ、Ｓ、ＰＭ、ＱＳは必ずしも定義されている必要はありません。Ｚは未定義の場合４、Ｓは””（ヌルストリング:NULL STRING）と自動的にサブルーチンの中で定義されます。
ＰＲ、ＰＭ、ＱＳが未定義の時は、サブルーチンの中では定義されず、そのまま実行されます。この部分がまたREADサブルーチンの魅力となっています。
ＰＭは必ずしもＩＦ文で始まる必要はありませんが、サブルーチン＾READではX PM　といった使われ方をしていて、必ずＳＴを変化させる文（STATEMENT ）である必要があります。従って、ＩＦ文をどこかで使用していることが条件となります。
サンプルプログラムをつくってありますが、TESTR を実行してください。短いプログラムですので、理解は容易だと思います。

注意：
このREADサブルーチンを使用するためにはＣＰ／Ｍ８６でＫＥＹ．ＣＭＤを実行してＫＥＹ．ＴＢＬを変更しておく必要があります。その部分はＨＯＭＥとＨＥＬＰキーです。ＨＯＭＥはＦＤ(HEXADECIMAL) +<CR> 、ＨＥＬＰはＡＯ(HEXADECIMAL)+<CR>です。これは＾ＭＥＮＵで使ったＫＥＹ．ＴＢＬと全く同じです。

３．入院台帳プログラムパッケージについて
入院台帳プログラムパッケージは以下のようなルーチンがセットになっています。

ADMI	;入院台帳プログラム	１	［登録］	;昭和60年 1月 3日;林寺
ADME	;入院台帳プログラム	２	［修正］	;昭和60年 1月 3日;林寺
ADMD	;入院台帳プログラム	３	［削除］	;昭和60年 1月 3日;林寺
ADMREP	;入院台帳プログラム	４	［修正］ｶﾅ氏名	;昭和60年 5月 2日;林寺
ADMNAM	;入院台帳プログラム	５	［名前検索］	;昭和59年 8月11日;林寺
ADMSM	;入院台帳プログラム	６	［記録］	;昭和60年 1月 3日;林寺
ADMR	;入院台帳プログラム	８	［表示］	;昭和60年 1月 9日;林寺
ADMO	;入院台帳プログラム	９	［表示］	;昭和60年 1月 9日;林寺
ADMN	;入院台帳プログラム	１０	［目次メニュー］	;昭和60年 1月10日;林寺
ADMN1	;入院台帳プログラム	１１	［インデックス作成］	;昭和60年 1月10日;林寺
ADMP	;入院台帳プログラム	１４	［印刷メニュー］	;昭和60年 1月10日;林寺
ADMP1	;入院台帳プログラム	１５	［印刷　その１］	;昭和60年 1月10日;林寺
ADMP2	;入院台帳プログラム	１６	［印刷　その２］	;昭和60年 1月10日;林寺
ADMDX1	;入院台帳プログラム	１７	［印刷　ﾌｫｰﾏｯﾄ］	;昭和60年 1月11日;林寺
ADMGET	;入院台帳プログラム	１８	［印刷　その４］	;昭和60年 1月11日;林寺
ADMP0	;入院台帳プログラム	１９	［条件設定］	;昭和60年 1月12日;林寺
ADMSHOW	;入院台帳プログラム	２０	［データ表示］	;昭和60年 1月12日;林寺
ADMFILE	;入院台帳プログラム	２１	［データ記憶］	;昭和60年 1月12日;林寺
ADMINP	;入院台帳プログラム	２２	［データ入力］	;昭和60年 1月12日;林寺
ADMN2	;入院台帳プログラム	２３	［インデックス削除］	;昭和60年 1月13日;林寺
ADMN3	;入院台帳プログラム	２４	［インデックス表示］	;昭和60年 1月13日;林寺
ADME1	;入院台帳プログラム	２５	［日付変更］	;昭和60年 1月13日;林寺
ADMD1	;入院台帳プログラム	２６	［日付削除］	;昭和60年 1月13日;林寺
ADMP3	;入院台帳プログラム	２７	［印刷　その３］	;昭和60年 1月13日;林寺
ADMP4	;入院台帳プログラム	２８	［印刷　その４］	;昭和60年 1月13日;林寺
ADMIND	;入院台帳プログラム	２９	［入力日付］	;昭和60年 2月 7日;林寺
ADMSM1	;入院台帳プログラム	３０	［記録　その２］	;昭和60年 1月 3日;林寺
ADMP5	;入院台帳プログラム	３１	［印刷　その５］	;昭和60年 1月13日;林寺
ADMADM	;入院台帳プログラム	３２	［入院日ｲﾝﾃﾞｯｸｽ］	;昭和60年 1月10日;林寺

```
MENU      ;入院台帳メニュー・プログラム                    ;昭和59年12月30日;林寺
READ      ;汎用ＲＥＡＤサブルーチン                        ;昭和60年 1月 3日;林寺
```

そのグローバルの構造は^ＡＤＭＦＩＬＥと^ＡＤＭＧＥＴにあります。
その一部分を以下に表示しました。

```
ADMFILE     ;入院台帳プログラム　２１［データ記憶］

X1          ;AID;NAME;SEX;BD;PARENT
X2          ;ZIP;ADR;TEL;RTEL

S1          ;DX1;DX2;DX3
S2          ;DX4;DX5
S3          ;NID;AGE;MD;RMD;ADTIME;DISCH;SUMMARY
S4          ;FRSLT;MEM1;MEM2;MEM3
```

そのグローバルを^％ＧＬ（これはまだ未発表で）で表示しますと以下のようになります。

```
^ADM = 221

ｱｻ[redacted]ｴ 1
- 1
- - 1 00-0103-2^浅[redacted]恵^F^S54[redacted]^[redacted]
- - 2 ^[redacted]市[redacted]区[redacted]の[redacted]２８－２４^6[redacted]-[redacted]58^
- - S591001
- - - 1 川崎病（冠動脈りゅう無し）^左冠動脈肺動脈ろう^
- - - 2 ^
- - - 3 59-4776^4Y^上坂^^10:40 AM^S591008^Y
- - - 4 検査終了^S570102-0203 MCLS にて入院 一過性に左冠動脈の拡大を
        認めた^^
```

特徴は患者の氏名を ID 番号順にではなく、アイウエオ順に格納したことです。これは、患者さんの検索を氏名で取り出すことが容易であるという単純な理由です。 ID 番号は国立京都病院ではシリアルな番号（来院順に割り当てるという方式）を採用しているため、 ID 番号には大きな意味をもたせていないためであり、あまりそれにこだわる必要がなかった。（本音は、国立京都病院の ID 番号は全くくだらないシロモノであると考えているからです。）

実際の運用の例を示します。
これは最初のメニュー画面です。

```
                入院台帳メニュー・プログラム
      Copyright (C) 1984 All rights reserved by PERSONAL GANG CORPORATION

コマンド :  ROLL UP, ROLL DOWN, INS(次頁), DEL(前頁), HOME(最上段), CLR(最上段)
            ↑(上), ↓(下), ←(最上段), HELP(中段), →(最下段), ESC. CR(選択)

No. ﾌﾟﾛｸﾞﾗﾑ  ---------------------------- 内      容 ----------------------------
 0  終　　了  ＥＸＩＴ（プログラム・モード）
 1  入　　力  入院時のデータをいれる。
 2  新規登録  患者さんの基礎データを登録する。
 3  変　　更  患者さんの基礎データを変更する。
 4  カナ氏名  患者さんのカナ氏名のみを変更する。
 5  削　　除  患者さんのデータをすべて抹消する。
 6  表　　示  氏名を指定してその人の基礎データを表示
 7  入院目次  入院日順のインデックス作成
 8  目　　次  インデックス・ファイルの作成・削除および目次によるｶﾀｶﾅ氏名表示
 9  印　　刷  印　　刷
10  日付変更  入院年月日の日付変更をする。
11  日付削除  入院年月日の日付削除をする。
12  入力日付  入力した最終の日付を入れる。
13  一覧表    ID#、姓名、性、生年月日、住所をすべての患者さんについて表示する
```

次にデータの入力では、「ア」とヒントを与えると「ア」で始まる名前の患者さんを全て表示してくれます。

```
次の人たちが登録されています。

 1) アサ ■エ          2) アサ■ フ■          3) アサ■ ヒ■          4) ア■ ト■
 5) アラ■ ユウ■        6) アン■ ヨシ■

何番目の人ですか？ 1
```

ここで1番を選びますと、その人のデータを取り出して、入力するかどうかをきいてきます。

```
                    データの入力
カタカナ シメイ: アサ■エ
外来ＩＤ ： 00-0103-2        氏名(漢字) ： 浅■恵                    性： F
生年月日 ： S54■                                  親の名前 ： ■
住所 ： ■市■西■                                ２８－２４
郵便番号 ：          電話番号 ： 6■-■58      連絡先の電話番号 ：

上の人のデータを入力しますか？ (Y/N) Y
```

入院の日付をきいてきますが、もし過去に入院してくるとそれを用意してくれます。

```
氏名(カタカナ) ： アサ■エ          氏名(漢字) ： 浅■恵          SEX ： F
                                                    誕生日 ： S54■

既に登録されている入院の月日は:
S591001

入院年日 (例えば SYYMMDD) ： S591001
```

```
氏名(カタカナ) ： アサ ■エ          氏名(漢字) ： 浅■恵          SEX ： F
                                    入院年月日 ： S591001          誕生日 ： S54■
入院 ID# ： 59-■76                  年令(例:12Y,3M) ： 4Y
診断名 1: 川崎病（冠動脈りゅう無し）
診断名 2: 左冠動脈肺動脈ろう
診断名 3:
診断名 4:
診断名 5:
主治医 ： 上坂                                  紹介医の氏名 ：
入院時刻(HH:MM AM/PM) ： 10:40 AM          退院日(SYYMMDD) ： S591008   総括 ： Y
転帰 ： 検査終了
ｺﾒﾝﾄ 1 ： S570102-0203 MCLS にて入院 一過性に左冠動脈の拡大を認めた
ｺﾒﾝﾄ 2 ：
ｺﾒﾝﾄ 3 ：
```

この画面の中でカーソルを動かせて、その部分のデータをいれます。

これは作業を中断した時の画面を示したものです。このようにいつでも作業を中断することができます。ＲＥＡＤサブルーチンを使っています。

```
氏名(カタカナ) ： アサ■エ          氏名(漢字) ： 浅■恵          SEX ： F
                                    入院年月日 ： S591001          誕生日 ： S54■

入院 ID# ： 59-■76                  年令(例:12Y,3M) ： 4Y
診断名 1: 川崎病（冠動脈りゅう無し）
診断名 2: 左冠動脈肺動脈ろう
診断名 3: 中断
診断名 4:
診断名 5:
主治医 ： 上坂                                  紹介医の氏名 ：
入院時刻(HH:MM AM/PM) ： 10:40 AM          退院日(SYYMMDD) ： S591008   総括 ： Y
転帰 ： 検査終了
ｺﾒﾝﾄ 1 ： S570102-0203 MCLS にて入院 一過性に左冠動脈の拡大を認めた
ｺﾒﾝﾄ 2 ：
ｺﾒﾝﾄ 3 ：
```

これはインデックスファイルを作成する過程を示したものです。ある患者さんのある入院日のデータに目印をつけることが出来ます。

```
                入院台帳プログラム　１０［目次メニュー］

機能を選んでください：1

1 >  インデックス作成

2 >  インデックス削除

3 >  インデックスに従って名前を表示

4 >  入院日をもとにインデックス・ファイル ^ADMADM を作成する
```

この患者さんは川崎病でしたので、その項目に印をつけたことになります。

```
                          インデックス作成
氏名(ｶﾀｶﾅ) : ｱｻ■ｴ              氏名(漢字) : 浅■恵          SEX : F
入院年月日 : S591001                                        誕生日 : S54■
入院 ID# : 59-■76                年令(例:12Y,3M) : 4Y
診断名 1: 川崎病（冠動脈りゅう無し）
診断名 2: 左冠動脈肺動脈ろう
診断名 3:
診断名 4:
診断名 5:
主治医 : 上坂                              紹介医の氏名 :
入院時刻(HH:MM AM/PM) : 10:40 AM          退院日(SYYMMDD) : S591008      総括 : Y
転帰 : 検査終了
ｺﾒﾝﾄ 1 : S570102-0203 MCLS にて入院 一過性に左冠動脈の拡大を認めた
ｺﾒﾝﾄ 2 :
ｺﾒﾝﾄ 3 :
登録されているインデックスは
  1)てんかん  2)ネフローゼ  3)化膿性リンパ節炎  4)気管支ぜんそく  5)救急輪番
  6)血管性紫斑病  7)血小板減少性紫斑病  8)甲状腺疾患  9)死亡
 10)若年性関節リウマチ 11)心疾患 12)心身症 13)新生児頭蓋内出血 14)腎炎
 15)髄膜炎 16)川崎病 17)脱水症 18)登校拒否症 19)熱性けいれん 20)肺炎 21)白血病
 22)白痴症 23)肥満症 24)不明熱 25)羊水吸引症候群 26)流行性筋痛
インデックスにする文字（例えば＜１＞）を入れて下さい 16
```

その後で、川崎病の患者さんをひろってくる作業を示します。

```
                    インデックスによるデータ表示

登録されているインデックスは
  1)てんかん  2)ネフローゼ  3)化膿性リンパ節炎  4)気管支ぜんそく  5)救急輪番
  6)血管性紫斑病  7)血小板減少性紫斑病  8)甲状腺疾患  9)死亡
 10)若年性関節リウマチ 11)心疾患 12)心身症 13)新生児頭蓋内出血 14)腎炎
 15)髄膜炎 16)川崎病 17)脱水症 18)登校拒否症 19)熱性けいれん 20)肺炎 21)白血病
 22)白痴症 23)肥満症 24)不明熱 25)羊水吸引症候群 26)流行性筋痛


インデックスにする文字（例えば＜１＞）を入れて下さい 16
```

選択の条件をマンプス言語で作成し、（ここでは I SEX="M" つまり男児だけを指定しました）検索を始めます。

```
マンプス言語でその症例選択の条件を書けますか？ Y
AID       NAME      SEX       BD        PARENT    ZIP       ADR       TEL
RTEL      DX1       DX2       DX3       DX4       DX5       NID       AGE
MD        RMD       ADTIME    DISCH     SUMMARY   FRSLT     MEM1      MEM2
MEM3


症例選択の条件 : I SEX="M"
```

```
インデックス : 川崎病

No. 氏    名        性 年 齢  入院日   入院時刻 退院日   主治医  紹介医
  1 江[■]淳[■]      M  9M    S590216 11:00 AM S590310 上坂   清益 英雄
  2 沢[■]美[■]      F  2Y    S590225 11:55 AM S590325 上坂
  3 内[■]み[■]      F  3Y    S590723 10:30 AM S590729 上坂
  4 杉[■]晋[■]      M  5Y    S590820 10:00 AM S590827 上坂
  5 渡[■]麻[■]子    F  4Y    S590903 10:30 AM S591002 上坂   仁科 周子
  6 浅[■]恵[■]      F  4Y    S591001 10:40 AM S591008 上坂
  7 平[■]亘[■]      M  1Y    S591008 10:15 AM S591015 上坂
  8 寺[■]未[■]      F  1Y    S591126 11:55 AM S591227 上坂   黒田 大典
  9 堀[■]敏[■]      M  1Y    S591222 11:30 AM S600129 上坂   河野 忠義
 10 谷[■]香[■]      F  3Y    S600116 11:00 AM S600204 上坂   若泉 悟
 11 海[■]沢[■]江    F  5Y    S600304 10:30 AM S600310 上坂
 12 平[■]圭[■]      M  4Y    S600311 10:30 AM S600317 上坂
 13 小[■]有[■]      F  6M    S600325 11:30 AM S600418 上坂   若泉 悟
 14 笠[■]麻[■]      F  4Y    S600325 09:30 AM S600331 上坂
 15 新[■]智[■]      F  9M    S600326 10:30 AM S600418 上坂
 16 森[■]徳[■]      M  4Y    S600520 10:15 AM S600527 上坂
 17 清[■]広[■]      M  5Y    S600527 09:50 AM S600603 上坂
 18 瀬[■]口[■]朗    M  4Y    S600603 10:30 AM         上坂
```

カナの氏名で登録されて（ディスクにしまいこまれて）いますので、その氏名を変更しました。

```
基礎データ: 浅[■]恵                              F S54[■]生  親の氏名:[■]
住所:[■]市[■]西[■]の[■]２８－２４                                TEL:6[■]-[■]58

入院日  入院ID  入院時刻 退院日  主治医   紹介医

                        カナ氏名のみの変更

ｶﾀｶﾅ ｼﾒｲ: ｱｻ[■]ｴ
外来ＩＤ : 00-0103-2          氏名(漢字) : 浅[■]恵                    性 : F
生年月日 : S541016                                  親の名前 : [■]
住所 : [■]の[■]２８－２４
郵便番号 :          電話番号 : 6[■]-[■]58     連絡先の電話番号 :


[ｱｻ[■]ｴ] を [ｱｻ[■]ﾖ] に変更します。

それでよろしいか (Y/N) ? Y
```

これは、入院時刻のところで、ＨＥＬＰキーを押したところです。このようにいつでも例えばの入力方法を示してくれます。また、もし答えが間違っていた（入力去れた文字が期待どおりでなかった）ときもこの例えばの入力方法を教えてくれます。これも＾ＲＥＡＤサブルーチンを使用しています。

```
氏名(ｶﾀｶﾅ) : ｱｻ[■]ｴ          氏名(漢字) : 浅[■]恵          SEX : F
                              入院年月日 : S591001          誕生日 : S54[■]

入院 ID# : 59-[■]76           年令(例:12Y,3M) : 4Y
診断名 1: 川崎病（冠動脈りゅう無し）
診断名 2: 左冠動脈肺動脈ろう
診断名 3:
診断名 4:
診断名 5:
主治医 : 上坂                 紹介医の氏名 :
入院時刻(HH:MM AM/PM) : 10:40 AM   退院日(SYYMMDD) : S591008   総括 : Y
転帰 : 検査終了
ｺﾒﾝﾄ 1 : S570102-0203 MCLS にて入院 一過性に左冠動脈の拡大を認めた
ｺﾒﾝﾄ 2 :
ｺﾒﾝﾄ 3 :


  例えば、10:15 AM や 20:38 PM のように入れる。
```

実際に実用化してみて、問題点がないわけではありません。
1．漢字入力に大変時間がかかることが最大の問題でした。
2．検索時にマイコンのスピードが遅いこと。
3．症例数が少ないためまずまず退院日から1週間以内に入力できた。
でも、このプログラムパッケージはそのほかにも応用ができますので、以下のような作業に使用できました。
①B型肝炎フォローアップ　　②心臓外来患者ノート
③長期入院患者の検査値整理　　④住所録管理
さらに、最近になって国立京都病院の病歴管理室ではＮＥＣ９８１０ｍ３が導入され、ハードディスクを使うことが可能になりましたので、病歴管理に応用しようと考えています。

最後に全てのルーチンのリストを示します。皆様のお役にたてば幸いと考えます。

```
READ        ;汎用ＲＥＡＤサブルーチン                    ;昭和60年 1月
            3日;林寺 忠
  + 1       S Q=$C(27) W Q,"[",Y,";",X,"H" W:$D(PR) Q,"[33m",PR," "
  + 2       S:'$D(Z) Z=4 S:'$D(S) S="" W Q,"[37m",Q,"[s",Q,"[47m",$J("
            ",Z)
  + 3       W Q,"[u" S $ZB=1,R=S,R=$ZE(R,Z) W Q,"[37m"
  + 4       F Q=1:1 Q:R'["  "  S R=$P(R,"  ",1)_" "_$P(R,"  ",2,999)
  + 5       F Q=1:1 Q:$E(R)'=" "!'$L(R)  S R=$E(R,2,$L(R))
  + 6       F Q=1:1 Q:$E(R,$L(R))'=" "!'$L(R)  S R=$E(R,1,$L(R)-1)
  + 7       S Q=$S($F(R,$C(253)):3,$ZB>0:2,$ZB<0:1,1:0) G READQ:R=""
  + 8       S:Q=3 R="" S:$F(R,"^") R=S G READQ:Q>1
  + 9       I '$F(R,$C(160)) X:$D(PM) PM G READQ:$T
  +10       W *7,*27,"[u?" G READ:'$D(QS) W *27,"[21;0H",*27,"[33m",!,Q
            S G READ
READQ       I $D(QS) W *27,"[21;0H",!,*27,"[0J"
  + 1       W *27,"[",Y,";",X,"H" W:$D(PR) PR," " S ZPL=$ZPL,$ZPL=1
  + 2       I Q=3 W *7,*27,"[31m中断",$J(" ",Z-4),*27,"[37m" H 2 S $ZPL
            =ZPL Q
  + 3       W:Q=2 S,$J(" ",Z-$L(S)) W:Q'=2 R,$J(" ",Z-$L(R)) S $ZPL=ZPL
             Q

TESTR       ;ＲＥＡＤ　サブルーチンのテストプログラム;昭和５９年１２月２
            ６日
A           W #,?8,*27,"[32m",$P($T(TESTR),";",2),"  ",$P($T(TESTR),";"
            ,3)
B           S X=30,Y=6,Z=14,S="適当な答え",PR="答え１は",PM="I R?,T "
  + 1       S QS="例えば: 漢字で入力してください。"
  + 2       S:$D(A) S=A D ^READ W:Q=2 *7 G B:Q=2,EXIT:Q=3 S A=R
C           K QS S X=30,Y=8,Z=14,S="適当",PR="答え２は",PM="I R?,漢" D
            ^READ
  + 1       G B:Q=2,EXIT:Q=3 S B=R G B
KIL         K PM,PR,Q,R,S,X,Y,Z,ZPL Q
EXIT        D KIL Q


MENU        ;入院台帳メニュー・プログラム                ;昭和59年12月3
            0日;林寺 忠
A           W #,?20,*27,"[32m",$P($T(MENU),";",2),!,?7
  + 1       W *27,"[33m","Copyright (C) 1984 All rights reserved by PER
            SONAL "
  + 2       W *27,"[5;7;31m","GANG" W *27,"[33m"," CORPORATION"
INSTRCTN    D BTMOFF,CSROFF W *27,"[34m",!! ;W *27,"[22;0H"
  + 1       W "コマンド :  ROLL UP, ROLL DOWN, INS(次頁), DEL(前頁)
            , HOME(最上段), CLR(最上段)"
  + 2       W !,"              ↑(上), ↓(下), ←(最上段), HELP(中
            段), →(最下段), ESC, CR(選択)"
  + 3       W !!,*27,"[36mNo. ﾌﾟﾛｸﾞﾗﾑ    ",*27,"[35m-------------
            -------------- 内       容 ---------------------------"
B           W *27,"[37m" S ZPL=$ZPL,$ZPL=1 S X=0,L=1,I=0 D PROG S $ZPL=
            ZPL
C           S I=I+1 I I'>IX W !,$J(I-1,2),"  ",U(I),?14,T(I) G C
  + 1       S LEND=IX+1,LX=LEND-1,L=1,LS=0 S:LX>16 LX=16
```

```
GET         S A="",B=0 W *27,"[",L+6,";0H",*27,"[37m",!
  + 1       W $J(L+LS-1,2),"  " W:L+LS'=1 *27,"[5;36m" W:L+LS=1 *27,"[7
            :5;32m" W U(L+LS)
  + 2       W *27,"[37m",?24,"  ",*27,"[32m",T(L+LS)
D           R *A G:A>47&(A<58) E S B=$F("/8/10/11/12/13/18/19/20/24/30"
            ,"/"_A) I B G E
ASK         I A=27 S X=0,Y=24,Z=2,S=LEND-2,PR="ﾌﾟﾛｸﾞﾗﾑ番号 (1-"
            _(LEND-2)_")は？",PM="I R?.N,R<(LEND-1)" D ^READ.CSROFF,
            ERAS G EXIT:Q=3,GET:Q=2 S IX=R G E
  + 1       I A=253!(A=160) R *B G E
  + 2       I A'=13 F I=0:1 R *B:1 Q:B=13  E  Q
  + 3       G GET
E           W *27,"[37m",*27,"[",L+6,";0H",!,$J(L+LS-1,2),"  ",U(L+LS),
            ?14,T(L+LS)
DO          I A=13 S L=L+LS,X=R(L) D EXIT K:L=1 X,L Q:'$D(L)  K L D @("
            ^"_X) G A
UP          I A=11 W:L=1&'LS *7 D:L=1&LS ROLLDOWN S L=L-1 S:L<1 L=1 G G
            ET
DOWN        I A=10 W:L=LX&(LX+LS=LEND) *7 D:L=LX&(LX+LS<LEND) ROLLUP S
            L=L+1 S:L>LX L=LX G GET
TEN         I A>47,A<58 S IX=A-48 G TENKEY
ESC         I A=27 G TENKEY
HOME        I A=8!(A=30) S L=1 G GET
MIDDLE      I A=160 S L=LX¥2+1 G GET
BOTOM       I A=12!(A=253) S L=LX G GET
  + 1       D ROLLUP:A=19,ROLLDOWN:A=20,PAGEUP:A=18,PAGEDOWN:A=24
F           G GET
TENKEY      :I A'=27 S L=IX-LS+1 G GET
  + 1       I 'LS,L>IX S L=IX+1 G GET
  + 2       I IX+1'<LEND S IX=LEND-1
  + 3       I L+LS>IX S IX=LS-IX F I=1:1:IX W:L=1&'LS *7 D ROLLDOWN S L
            =L-1 S:L<1 L=1 I I=IX S L=1 G GET
  + 4       I LX+LS<IX S IX=IX-LX-LS+1 F I=1:1:IX W:L=LX&(LX+LS=LEND) *
            7 D ROLLUP I I=IX S L=LX G GET
  + 5       I IX+1=LEND W *7 G GET
  + 6       I LX+LS'<IX!(L+LS'>IX) S L=IX-LS+1 I L<1 S L=1
  + 7       G GET
ROLLUP      S LS=LS+1 I LX+LS'<LEND S LS=LS-1 W *7 Q
  + 1       W *27,"[8;0H",*27,"[1M",*27,"[22;0H",!,$J(LX+LS-1,2),"  ",U
            (LX+LS),?14,T(LX+LS) Q

ROLLDOWN    S LS=LS-1 I LS<0 S LS=0 W *7 Q
  + 1       W *27,"[23;0H",*27,"[1M",*27,"[8;0H",*27,"[1L"
  + 2       W *27,"[7;0H",!,$J(1+LS-1,2),"  ",U(1+LS),?14,T(1+LS) Q
PAGEUP      I LS+LX+1'<LEND W *7 Q
  + 1       S I=LS,LS=$S(2*LX+LS+1'>LEND:LX+LS+1,1:LEND-LX)-1
  + 2       S IX=LS-I,LS=I,A=0 G SHFT
  + 3       :I LS-I<9 S IX=LS-I,LS=I,A=0 G SHFT
PAGECLR     W *27,"[7;0H",!,*27,"[0J"
  + 1       W *27,"[7;0H" F I=1:1:LX W !,$J(I-1,2),"  ",U(I+LS),?14,T(I
            +LS)
  + 2       Q
PAGEDOWN    I LS=0 W *7 Q
  + 1       S I=LS,LS=LS-$S(LS-LX>0:LX,1:LS)
  + 2       S IX=I-LS,LS=I,A=1 G SHFT
  + 3       :I I-LS<9 S IX=I-LS,LS=I,A=1 G SHFT
  + 4       G PAGECLR
SHFT        F I=1:1:IX D ROLLUP:'A,ROLLDOWN:A
  + 1       Q
CSRON       W *27,"[>5l" Q
CSROFF      W *27,"[>5h" Q
ERAS        W *27,"[24;0H",$J(" ",79) Q
BTMON       W *27,"[>1l" Q
BTMOFF      W *27,"[>1h" Q
PROG        ;
  + 1       S R(1)="",U(1)="終　　了",T(1)="ＥＸＩＴ（プログラム・モード
            )"
ADMSM       S R(2)="ADMSM",U(2)="入　　力",T(2)="入院時のデータをいれる
            。"
ADMI        S R(3)="ADMI",U(3)="新規登録",T(3)="患者さんの基礎データを登
            録する。"
ADME        S R(4)="ADME",U(4)="変　　更",T(4)="患者さんの基礎データを変
            更する。"
ADMREP      S R(5)="ADMREP",U(5)="カナ氏名",T(5)="患者さんのカナ氏名のみ
            を変更する。"
```

```
ADMD        S R(6)="ADMD",U(6)="削　　除",T(6)="患者さんのデータをすべて
            抹消する。"
ADMO        S R(7)="ADMO",U(7)="表　　示",T(7)="氏名を指定してその人の基
            礎データを表示"
ADMADM      S R(8)="ADMADM",U(8)="入院目次",T(8)="入院日順のインデックス
            作成"
ADMN        S R(9)="ADMN",U(9)="目　　次",T(9)="インデックス・ファイルの
            作成・削除および目次によるｶﾀｶﾅ氏名表示"
ADMP        S R(10)="ADMP",U(10)="印　　刷",T(10)="印　　刷"
ADME1       S R(11)="ADME1",U(11)="日付変更",T(11)="入院年月日の日付変更
            をする。"
ADMD1       S R(12)="ADMD1",U(12)="日付削除",T(12)="入院年月日の日付削除
            をする。"
ADMIND      S R(13)="ADMIND",U(13)="入力日付",T(13)="入力した最終の日付
            を入れる。"
ADMR        S R(14)="ADMR",U(14)="一覧表　",T(14)="ID#、姓名、性、生
            年月日、住所をすべての患者さんについて表示する"
PEND        S IX=14 Q  ;プログラムの数は無制限ですが、PEND の行は削除し
            てはいけません。
KIL         K A,B,I,IX,LEND,LS,LX,PM,PR,Q,R,S,T,U,Y,Z,ZPL Q
EXIT        S $ZPL=ZPL D BTMON,CSRON W:L=1 *27,"[6;34H",*27,"[5;33m"
  + 1       W:L=1 *7,*27,"[5;33mお疲れさまでした。" S Y=$S(LEND>16:22,1
            :LEND+6)
  + 2       W *27,"[",Y,";0H",*27,"[37m" D KIL W ! Q


ADMI        ;入院台帳プログラム　　1 [登録]　　;昭和60年 1月
            3日;林寺 忠
A           K QS
  + 1       W *26,# S X=0,Y=22,Z=20,PR="登録する人の氏名は(ｶﾀｶﾅ)
            ? ",PM="I R?,ｶﾄ1"" "".ｶﾄ",S="" S:$D(NAMEK) S=NAMEK D ^REA
            D G EXIT:Q>1!(R="") S NAMEK=R
B           S FADD=0 I '($D(^ADM(NAMEK))#10) G ADD
C           D ERAS S X=0,Y=22,Z=1,S="N",PM="I R=""Y""!(R=""N"")"
  + 1       S PR="同名の人がいますが追加しますか ? (Y/N) "
  + 2       D ^READ I Q=3 G EXIT
  + 3       G:R'="Y" EXIT
CONF        D ERAS W *7 S X=0,Y=22,Z=1,S="N",PM="I R=""Y""!(R=""N"")"
  + 1       S PR="同名の人を本当に追加しますか ? (Y/N) "
  + 2       D ^READ I Q=3 G EXIT
  + 3       G:R'="Y" EXIT S FADD=1
ADD         D WHITE W,*26,#,*27,"[32m",?34,"データの追加",!,"ｶﾀｶﾅ ｼﾒｲ:
            ",NAMEK
  + 1       D WHITE,BSHOW^ADMINP,BINP^ADMINP
  + 2       G ASKSAV:Q=3,FILE
ASKSAV      D ERAS S PR="中断しましたが、それまでのデータをしまいこみま
            すか ? (Y/N)",S="N",X=0,Y=22,Z=1,PM="I R=""Y""!(R=""N"")"

  + 1       D ^READ G B:Q=2,ASK:Q=3 I R'="Y" G ASK
FILE        D FILE^ADMFILE
ASK         D ERAS S S="N",X=0,Y=22,Z=1,PM="I R=""Y""!(R=""N"")"
  + 1       S PR="更に追加（登録）を続けますか ? (Y/N)"
  + 2       D ^READ G B:Q=2,EXIT:Q=3 I R="Y" W *26,# D KIL K NAMEK G AD
            MI
  + 3       G EXIT
WHITE       W *27,"[37m" Q
MOVE        W *27,"[",Y,";",X,"H" Q
ERAS        W *27,"[22;0H",$J(" ",79) Q
KIL         D KIL^ADMINP Q
EXIT        D KIL,WHITE Q


ADME        ;入院台帳プログラム　　2 [修正]　　;昭和60年 1月
            3日;林寺 忠
A           K QS
  + 1       W *26,# S X=0,Y=22,Z=20,PR="氏名は(ｶﾀｶﾅ) ? ",PM="S:R?
            1ｶ R=R_"" -"" I R?,ｶﾄ1"" "".ｶﾄ",S="" S:$D(NAMEK) S=NAMEK
            D ^READ G EXIT:Q>1!(R="") S NAMEK=R
B           D ^ADMNAM I 'N G A
  + 1       D WHITE W *26,#,*27,"[32m",?34,"データの修正",!,"ｶﾀｶﾅ ｼﾒｲ:
            ",NAMEK D SRCH
  + 2       D WHITE,BSHOW^ADMINP,BINP^ADMINP
  + 3       G ASKSAV:Q=3,FILE
ASKSAV      D ERAS S PR="中断しましたが、それまでのデータをしまいこみま
            すか ? (Y/N)",S="N",X=0,Y=22,Z=1,PM="I R=""Y""!(R=""N"")"
```

```
  + 1       D ^READ G B:Q=2,ASK:Q=3 I R'="Y" G ASK
FILE        D EDIT^ADMFILE
ASK         D ERAS S PR="更に修正を続けますか ? (Y/N)",S="N",X=0,Y=22,Z
            =1,PM="I R=""Y""!(R=""N"")"
  + 1       D ^READ G B:Q=2,EXIT:Q=3 I R="Y" W *26,# D KIL K NAMEK G AD
            ME
  + 2       G EXIT
SRCH        G:N=1 GET W ! F I=1:1:N S A=^ADM(NAMEK,I,1) W !,$J(I,2),")
            ",$P(A,"^",1),?15,$P(A,"^",2),?28,$P(A,"^",3),?32,$P(A,"^
            ",4)
  + 1       W !!,*27,"[33m" R "上のうちどの人を修正しますか? ",A
  + 2       D WHITE W *26,#,*27,"[32m",?34,"データの修正",!,"ｶﾀｶﾅ ｼﾒｲ:
            ",NAMEK
  + 3       I A<1!(A>N) W *7 G SRCH
  + 4       S N=A
GET         D BASE^ADMGET Q
WHITE       W *27,"[37m" Q
MOVE        W *27,"[",Y,";",X,"H" Q
ERAS        W *27,"[22;0H",$J(" ",79) Q
KIL         D KIL^ADMINP Q
EXIT        D KIL,WHITE Q


ADMD        ;入院台帳プログラム　　３［削除］　　;昭和60年 1月
            3日;林寺 忠
A           K QS
  + 1       W *26,# S X=0,Y=22,Z=20,PR="氏名は(ｶﾀｶﾅ) ? ",PM="S:R?
            1ｶ R=R_"" -"" I R?.ｶﾄ1"" "".ﾄ",S="" S:$D(NAMEK) S=NAMEK
            D ^READ G EXIT:Q>1!(R="") S NAMEK=R
B           D ^ADMNAM I 'N W *7 S NAMEK="" G ADMD
  + 1       D WHITE W *26,#,*27,"[32m",?34,"データの削除",!,"ｶﾀｶﾅ ｼﾒｲ:
            ",NAMEK D SRCH
  + 2       D WHITE,BSHOW^ADMINP
E           W *27,"[19;0H",*27,"[33m[",NAMEK,"] ",*27,"[37mを削除します
            。"
CONF        D ERAS S X=0,Y=22,Z=1,PR="それでよろしいか (Y/N) ? ",S="N"
  + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3,A:Q=2,ASK:R="
            N"
  + 2       I '$D(^ADM(NAMEK,N)) W *7 D ERAS W *27,"[22;0H[",A,"] は記録
            にありませんので出来ません。" H 4 D ERAS G A
DEL         S NAMOLD=NAMEK,NOLD=N D DEL^ADMFILE
ASK         D ERAS S X=0,Y=22,Z=1,S="N",PR="更に削除を続けますか ? (Y/N
            )",PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3,A:Q=2
  + 1       I R="Y" D KIL K NAMEK W *26,# G A
  + 2       G EXIT
SRCH        G:N=1 GET W ! F I=1:1:N S A=^ADM(NAMEK,I,1) W !,$J(I,2),")
            ",$P(A,"^",1),?15,$P(A,"^",2),?28,$P(A,"^",3),?32,$P(A,"^
            ",4)
  + 1       W !!,*27,"[33m" R "上のうちどの人を削除しますか? ",A
  + 2       D WHITE W *26,#,*27,"[32m",?34,"データの削除",!,"ｶﾀｶﾅ ｼﾒｲ:
            ",NAMEK
  + 3       I A<1!(A>N) W *7 G SRCH
  + 4       S N=A
GET         D BASE^ADMGET Q
WHITE       W *27,"[37m" Q
ERAS        W *27,"[22;0H",$J(" ",79) Q
MOVE        W *27,"=",*Y+32,*X+32 Q
KIL         K NAMOLD,NAMNEW,NNEW,NOLD,NX D KIL^ADMINP Q
EXIT        D KIL,WHITE Q

ADMREP      ;入院台帳プログラム　　４［修正］ｶﾅ氏名　　:昭和60年 5月
            2日;林寺 忠
A           K QS
  + 1       W *26,# S X=0,Y=22,Z=20,PR="氏名は(ｶﾀｶﾅ) ? ",PM="S:R?
            1ｶ R=R_"" -"" I R?.ｶﾄ1"" "".ﾄ",S="" S:$D(NAMEK) S=NAMEK
            D ^READ G EXIT:Q>1!(R="") S NAMEK=R
B           D ^ADMNAM I 'N G A
  + 1       D WHITE W *26,#,*27,"[32m",?34,"カナ氏名のみの変更",!,"ｶﾀｶﾅ
             ｼﾒｲ: ",NAMEK
  + 2       D SRCH S NOLD=N,NAMOLD=NAMEK D WHITE,BSHOW^ADMINP
ASKX        S X=0,Y=22,Z=25,PR="登録氏名: "_NAMOLD_" を変更し、新しい氏
            名は (ｶﾀｶﾅ)? ",PM="S:R?1ｶ R="" -"" I R?.ｶﾄ1"" "".ｶ
            ﾄ",S=NAMOLD
```

```
   + 1          K QS D ^READ G ASK:Q=3,A:Q=2
   + 2          W *27,"[19;0H","[",NAMOLD,"] を ",*27,"[33m[",R,"] ",*27,"[
                37mに変更"
   + 3          I R=NAMOLD W *27,"[31mは出来ません",*7 G ASKX
   + 4          W "します。        " S NAMNEW=R
CONF            D ERAS S X=0,Y=22,Z=1,PR="それでよろしいか (Y/N) ? ",S="N"
   + 1          S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3,ASKX:Q=2!(R="
                N")
   + 2          ;I $D(^ADM(NAMEK,N)) W *7 D ERAS W *27,"[22;0H[",A,"] が既に
                登録されていますので、そのような変更は出来ません。" H 4 D
                 ERAS G ASKX
FILE            S NAMEK=NAMNEW D BASE^ADMFILE G EXIT:NAMEK="" S NNEW=N
TRNSF           S KD="S",N=NOLD D TRNSF^ADMFILE
DEL             S NAMEK=NAMOLD D DEL^ADMFILE
ASK             D ERAS S PR="更にカナ氏名の変更を続けますか ? (Y/N)",S="N",
                PM="I R=""Y""!(R=""N"")"
   + 1          S X=0,Y=22,Z=1 D ^READ G B:Q=2,EXIT:Q=3 I R="Y" W *26,# D K
                IL K NAMEK G ADMREP
   + 2          G EXIT
SRCH            G:N=1 GET W ! F I=1:1:N S A=^ADM(NAMEK,I,1) W !,$J(I,2),")
                ",$P(A,"^",1),?15,$P(A,"^",2),?28,$P(A,"^",3),?32,$P(A,"^
                ",4)
   + 1          W !!,*27,"[33m" R "上のうちどの人のカナ氏名を変更しますか?
                ",A
   + 2          D WHITE W *26,#,*27,"[32m",?34,"カナ氏名の変更",!,"ｶﾀｶﾅ ｼﾒｲ
                : ",NAMEK
   + 3          I A<1!(A>N) W *7 G SRCH
   + 4          S N=A
GET             D BASE^ADMGET Q
WHITE           W *27,"[37m" Q
MOVE            W *27,"[",Y,";",X,"H" Q
ERAS            W *27,"[22;0H",$J(" ",79) Q
KIL             K M,NAMNEW,NAMOLD,NNEW,NOLD,NX D KIL^ADMINP Q
EXIT            D KIL,WHITE Q


ADMNAM          ;入院台帳プログラム     5 [名前検索]          ;昭和59年 8月1
                1日;林寺 忠
A               ;
B               ;R !,"名前は? ",NAMEK
   + 1          I $D(^ADM(NAMEK)) S N=^ADM(NAMEK) Q
   + 2          S B=NAMEK,A=$E(B,2,$L(B)),B=$E(B,1),C=$A(B)+1,B=$C(C),B=B_A
                ,C=0,D=NAMEK
C               S D=$N(^ADM(D)) I D<0!(D'?.ｶ1" ".ｶ) G E
   + 1          S E=^(D) I D]B G E
   + 2          S C=C+1,C(C)=D G C
E               I 'C W *7,*7," は登録されていません。" S N=0 H 2 G EXIT
   + 1          I C=1 S NAMEK=C(1),N=^ADM(NAMEK) G EXIT
   + 2          W *26,#,"次の人たちが登録されています。",!
   + 3          F I=1:1:C W:'(I-1#4) ! W ?I-1#4*18,$J(I,2),") ",C(I)
G               S X=0,Y=22,Z=3,PR="何番目の人ですか? ",S=1,PM="I R?.N.R>0,R
                <(C+1)" D ^READ I Q>1 S N=0 G EXIT
   + 1          S NAMEK=C(R) K C S N=^ADM(NAMEK) G EXIT
MOVE            W *27,"=",*Y+32,*X+32 Q
KIL             K A,B,C,D,E,FUNC,I,J,PM,PR,Q,R,S,SEL,X,Y,Z,ZPL Q
EXIT            D KIL Q


ADMSM           ;入院台帳プログラム     6 [記録]              ;昭和60年 1月
                3日;林寺 忠
   + 1          ;W *27,"[22;0H",*27,"[33mパスワード(三文字)は? " R *X,*Y
                ,*Z
   + 2          ;I X'=24,Y'=24,Z'=24 W *7,*27,"[31m 使用禁止です。" H 2 G E
                XIT
A               ;
   + 1          W *26,# S X=0,Y=22,Z=20,PR="氏名は (ｶﾀｶﾅ) で? ",S="",P
                M="S:R?1ｶ R=R_"" -"" I R?.ｶ1"" "".ｶ" S:$D(NAMEK) S=NAMEK
                D ^READ G EXIT:'$L(R)!(Q>1) S NAMEK=R
B               D ^ADMNAM I 'N W *7 S NAMEK="" G A
   + 1          D WHITE W *26,#,*27,"[32m",?34,"データの入力",!,"ｶﾀｶﾅ ｼﾒｲ:
                ",NAMEK D SRCH
   + 2          D WHITE,BSHOW^ADMINP
ASKREG          S X=0,Y=22,Z=1,S="Y",PR="上の人のデータを入力しますか? (Y/
                N) "
   + 1          S PM="I R=""Y""!(R=""N"")" D ^READ G A:Q=2,ASK:R'="Y"!Q
```

```
REG             D ^ADMSM1
ASK             S X=0,Y=22,Z=1,S="N",PR="更に入力を続けますか ? (Y/N)",PM="
                I R=""Y""!(R=""N"")" D ERAS,^READ
  + 1           G EXIT:R'="Y"!Q I R="Y" D KIL W *26,# G A
SRCH            G:N=1 GET W ! F I=1:1:N S A=^ADM(NAMEK,I,1) W !,$J(I,2),")
                ",$P(A,"^",1),?15,$P(A,"^",2),?28,$P(A,"^",3),?32,$P(A,"^
                ",4)
  + 1           W !!,*27,"[33m" R "上のうちどの人のデータを入力しますか? ",
                A
  + 2           D WHITE W *26,#,*27,"[32m",?34,"データの入力",!,"ｶﾀｶﾅ ｼﾒｲ:
                ",NAMEK
  + 3           I A<1!(A>N) W *7 G SRCH
  + 4           S N=A
GET             D BASE^ADMGET Q
WHITE           W *27,"[37m" Q
ERAS            W *27,"[22;0H",$J(" ",79) Q
MOVE            W *27,"[",Y,";",X,"H" Q
KIL             D KIL^ADMINP Q
EXIT            D KIL,WHITE Q


ADMR            ;入院台帳プログラム     8 [表示]                ;昭和60年 1月
                9日;林寺 忠
A               W *26,#
  + 1           S %QTY=2,%DTY="CRT",%IOD=0,%DEF=0 D ^%MIOS :  GET OUTPUT DE
                VICE
  + 2           I '$D(%IOD) G EXIT
  + 3           I "CRT^LP"'[%DTY W !,?5,"出力デヴァイス ???" G A
  + 4           I "CRT,LP"[%DTY U %IOD U 0
  + 5           S NAMEK="ｱ",%LIN=0
TITLE           U %IOD W:'%IOD *26,# W "ＩＤ#",?10,"氏     名",?27,"性",?31
                ,"誕生日",?38,"住     所"
B               S NAMEK=$N(^ADM(NAMEK)) I NAMEK<0 G EXIT
  + 1           S N=^(NAMEK)
  + 2           S Q=0 D SRCH G EXIT:Q=3,B
SRCH            G:N=1 GET S M=N F N=1:1:M D GET
  + 1           Q
GET             D BASE^ADMGET
PRINT           W !,AID,?10,NAME,?28,SEX,?30,BD,?38,ADR
  + 1           Q
WHITE           W *27,"[37m" Q
MOVE            W *27,"[",Y,";",X,"H" Q
ERAS            W *27,"[22;0H",$J(" ",79) Q
KIL             K %DTY,%IOD,%LIN D KIL^ADMINP Q
EXIT            I %DTY="CRT" W !,?22,*27,"[33m" R "リターンキーを押して下さ
                い。",*%LIN D WHITE
  + 1           U 0 D KIL Q


ADMO            ;入院台帳プログラム     9 [表示]                ;昭和60年 1月
                9日;林寺 忠
A               K QS
  + 1           W *26,# S X=0,Y=22,Z=20,PR="氏名は(ｶﾀｶﾅ) ? ",PM="S:R?
                1ｶ R=R_"" -"" I R?.ｶﾄ1"" "".ｶﾄ".S="" S:$D(NAMEK) S=NAMEK
                D ^READ G EXIT:Q>1!(R="") S NAMEK=R
B               D ^ADMNAM I 'N G A
  + 1           D WHITE W *26,#,*27,"[32m",?34,"データの表示",!,"ｶﾀｶﾅ ｼﾒｲ:
                ",NAMEK D SRCH
  + 2           D WHITE,BSHOW^ADMINP
ASK             D ERAS S PR="更に表示を続けますか ? (Y/N)",S="N",X=0,Y=22,Z
                =1,PM="I R=""Y""!(R=""N"")"
  + 1           D ^READ G B:Q=2,EXIT:Q=3 I R="Y" W *26,# D KIL K NAMEK G AD
                MO
  + 2           G EXIT
SRCH            G:N=1 GET W ! F I=1:1:N S A=^ADM(NAMEK,I,1) W !,$J(I,2),")
                ",$P(A,"^",1),?15,$P(A,"^",2),?28,$P(A,"^",3),?32,$P(A,"^
                ",4)
  + 1           W !!,*27,"[33m" R "上のうちどの人を表示しますか? ",A
  + 2           D WHITE W *26,#,*27,"[32m",?34,"データの修正",!,"ｶﾀｶﾅ ｼﾒｲ:
                ",NAMEK
  + 3           I A<1!(A>N) W *7 G SRCH
  + 4           S N=A
GET             D BASE^ADMGET Q
WHITE           W *27,"[37m" Q
MOVE            W *27,"[",Y,";",X,"H" Q
```

```
 ERAS          W *27,"[22;0H",$J(" ",79) Q
 KIL           D KIL^ADMINP Q
 EXIT          D KIL,WHITE Q


ADMN           ;入院台帳プログラム　１０［目次メニュー］　　;昭和60年 1月1
               0日;林寺 忠
A              W *26,# S X=20,Y=0 D MOVE W *27,"[33m",$P($T(+1),":",2) D W
               HITE
  + 1          F I=1:1 S X=$T(FUNC+I) Q:X=""  S FUNC(I)=X
  + 2          S X=0,Y=4 D MOVE G LIST
READ           S X=0,Y=4 D MOVE R "機能を選んでください: ",SEL G LIST:SEL=
               "?" G EXIT:SEL=""
  + 1          F J=1:1:I-1 I SEL=$P(FUNC(J),":",2) G ROUT
  + 2          S X=22,Y=4 D MOVE W *7," ? 正しく入れて下さい。"
LIST           W !!,*27,"[32m" F J=1:1:I-1 W $P(FUNC(J),":",2)," >  ",$P(F
               UNC(J),":",3),!!
  + 1          D WHITE S X=22,Y=4 D MOVE W " " G READ
ROUT           S ROUT=$P(FUNC(J),":",4) D @("^"_ROUT) K ROUT G A
  + 1          Q
KIL            K FUNC,I,J,SEL Q
EXIT           D KIL Q  ;S X=0,Y=20 D MOVE W "お疲れさま。" K X,Y Q
WHITE          W *27,"[37m" Q
MOVE           W *27,"=",*Y+32,*X+32 Q
FUNC           ; FUNCTION LIST FOLLOWS
  + 1          ;1;インデックス作成;ADMN1
  + 2          ;2;インデックス削除;ADMN2
  + 3          ;3;インデックスに従って名前を表示;ADMN3
  + 4          ;4;入院日をもとにインデックス・ファイル ^ADMADM を作成する;
               ADMADM


 ADMN1         ;入院台帳プログラム　１１［インデックス作成］;昭和60年 1月1
               0日;林寺 忠
 A             K QS
   + 1         W *26,# S X=0,Y=22,Z=20,PR="氏名は(ｶﾀｶﾅ) ? ",PM="S:R?
               1ｶ R=R_"" -"" I R?.ｶﾄ1"" "".ｶﾄ",S="" S:$D(NAMEK) S=NAMEK
               D ^READ G EXIT:Q>1!(R="") S NAMEK=R
 B             D ^ADMNAM I 'N G A
   + 1         W # D SRCH,BASE^ADMGET,SB^ADMINP,SKD^ADMINP
 ASKD          S X=0,Y=22,Z=7,S="",PM="I R?1""S""6N!(R?1""*S""6N),$E($P(R,
               ""S"",2),1,2)<80,$E($P(R,""S"",2),3,4)<13,$E($P(R,""S"",2
               ),5,6)<32,$D(^ADM(NAMEK,N,R))"


   + 1         S VAR="KD" S:$D(@VAR) S=@VAR S PR="入院年日 (例えば SYYMMD
               D) :"
   + 2         D ^READ G A:Q=2,EXIT:Q=3
   + 3         D CLRLINE5 S KD=R
 C             W # S Z=1 D WHITE,GET^ADMGET,SB1^ADMINP,SD^ADMINP
 D             W *27,"[0;30H",*27,"[32mインデックス作成",!,"氏名(ｶﾀｶﾅ
               ) : ",NAMEK,!,"入院年月日 : ",KD


 SHOW          W *27,"[16;0H",*27,"[33m登録されているインデックスは",*27,"
               [32m",!
   + 1         K INDX S N=-1 F I=0:1 S N=$N(^ADMINDX(N)) Q:N<0  W:$L(N)*2+
               $X>74 ! W $J(I+1,3),")",N S INDX(I+1)=N
   + 2         S N=I W !
 ASKINDX       S X=0,Y=22,Z=20,PR="インデックスにする文字（例えば＜１＞）を
               入れて下さい",S=""
   + 1         S PM="I R?.TｶA!(R?.N&(R>0)&(R'>N))" D ^READ G EXIT:Q=3,A:Q
               =2
   + 2         W:R="" *7 G ASKINDX:R="" S INDX=R I R?.N.R'>N S INDX=INDX(R
               )


 CONF          D ERAS S X=0,Y=22,Z=1,PR="ｲﾝﾃﾞｯｸｽ はく"_INDX_">でよろ
               しいか (Y/N) ? ",S="Y"
   + 1         S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3,ASKINDX:Q=2!(
               R="N")
   + 2         S DAT=NAMEK_"^"_NX I $D(^ADMINDX(INDX,KD)) D CHECK I  W *7
               D ERAS W *27,"[22;0H既に登録済みです。" H 2 D ERAS G ASKC
               ON


 SAV           S:'$D(^ADMINDX(INDX)) ^ADMINDX(INDX)=0 S N=^ADMINDX(INDX),N
               =N+1
   + 1         S ^ADMINDX(INDX)=N I '$D(^ADMINDX(INDX,KD)) S ^ADMINDX(INDX
               ,KD)=0
   + 2         S N=^ADMINDX(INDX,KD),N=N+1,^(KD)=N,^ADMINDX(INDX,KD,N)=DAT
                D ERAS
   + 3         D ^ZOPTION
```

```
ASKCON      S X=0,Y=22,Z=1,S="N",PR="更にインデックス作成を続けますか ?
             (Y/N)"
  + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3
  + 2       I R="Y"!(Q=2) W *26,# D KIL K NAMEK G A
  + 3       G EXIT
CHECK       S X=^ADMINDX(INDX,KD),Y=0 F I=1:1:X I ^ADMINDX(INDX,KD,I)=D
            AT S Y=1 Q
  + 1       I Y Q
  + 2       Q
SRCH        G:N=1 GET W ! F I=1:1:N S A=^ADM(NAMEK,I,1) W !,I,") ",$P(A
            ,"^",1),?11,$P(A,"^",2),?31,$P(A,"^",3),?35,$P(A,"^",4)
  + 1       W !!,*27,"[33m","上のうちどの人のインデックスを作成しますか
            ? " D WHITE R A I A<1!(A>N) W *7,*27,"[22;0H" G SRCH
  + 2       S N=A W *26,# D DISP

GET         S NX=N Q
CLRLINE5    W *27,"[5;0H",*27,"[0J" Q
WHITE       W *27,"[37m" Q
MOVE        W *27,"[",Y,";",X,"H" Q
ERAS        W *27,"[22;0H",$J(" ",79) Q
KIL         K DAT,INDX,NO D KIL^ADMINP Q
EXIT        D KIL,WHITE Q


ADMP        ;入院台帳プログラム　１４［印刷メニュー］　　;昭和60年 1月1
            0日;林寺 忠
A           W *26,# S X=20,Y=0 D MOVE W *27,"[33m",$P($T(+1),":",2) D W
            HITE
  + 1       F I=1:1 S X=$T(FUNC+I) Q:X=""  S FUNC(I)=X
  + 2       S X=0,Y=4 D MOVE G LIST
READ        S X=0,Y=4 D MOVE R "機能を選んでください: ",SEL G LIST:SEL=
            "?" G EXIT:SEL=""
  + 1       F J=1:1:I-1 I SEL=$P(FUNC(J),";",2) G ROUT
  + 2       S X=22,Y=4 D MOVE W *7," ? 正しく入れて下さい。"
LIST        W !!,*27,"[32m" F J=1:1:I-1 W $P(FUNC(J),";",2)," > ",$P(F
            UNC(J),";",3),!!
  + 1       D WHITE S X=22,Y=4 D MOVE W " " G READ
ROUT        W # S ROUT=$P(FUNC(J),";",4) D @("^"_ROUT) K ROUT G A
  + 1       Q
KIL         K FUNC,I,J,SEL,X,Y D KIL^ADMINP Q
EXIT        D KIL Q  ;S X=0,Y=20 D MOVE W "お疲れさま。" K X,Y Q
WHITE       W *27,"[37m" Q
MOVE        W *27,"=".*Y+32,*X+32 Q
FUNC        ; FUNCTION LIST FOLLOWS
  + 1       ;1;すべてを印刷する;ADMP1
  + 2       ;2;名前・性・生年月日・入院日・入院時刻・退院日・主治医・紹
            介医を印刷する;ADMP2
  + 3       ;3;インデックス・ファイルをもとにデータを印刷する;ADMP3
  + 4       ;4;名前を指定してその児のデータを印刷する;ADMP4
  + 5       ;5;昭和６０年１月１日以後入院の氏名・入院日・診断名を印刷す
            る;ADMP5


ADMP1       ;入院台帳プログラム　１５［印刷　その１］　　;昭和60年 1月1
            0日;林寺 忠
A           ;
  + 1       S %QTY=2,%DTY="CRT",%IOD=0,%DEF=0 D ^%MIOS ;  GET OUTPUT DE
            VICE
  + 2       I '$D(%IOD) G EXIT
  + 3       I "CRT^LP"'[%DTY W !,?5,"出力デヴァイス ???" G A
  + 4       I "CRT,LP"[%DTY U %IOD U 0
  + 5       S NAMEK="ア",NO=0,N=-1 I %IOD=0 W *26,# G B
  + 6       W !,"プリンターにて出力実行中です。" U %IOD
B           D LIN0^ADMDX1,LIN2^ADMDX1
C           S N=-1 D NAM G EXIT:NAMEK="" G C:NF
  + 1       S NO=NO+1 D BASE^ADMGET,L0^ADMDX1 S KD="A"
D           D GET G C:DF D GET^ADMGET,L2^ADMDX1 G D
NAM         S NAMEK=$N(^ADM(NAMEK)) I NAMEK<0!(NAMEK'?.ｶ1" ".ｶ) S NAMEK
            ="" Q
  + 1       S NF=0,N=$N(^ADM(NAMEK,N)) S:N<0!(N'?.N) NF=1 Q
GET         S DF=0,KD=$N(^ADM(NAMEK,N,KD)) S:KD<0 DF=1 Q
KIL         K %IOD,%DTY,AID,DF,NF,NO D KIL^ADMINP Q
EXIT        I %IOD U %IOD W ! C %IOD
  + 1       I '%IOD W !,*27,"[33m" R "リターン・キーを押して下さい。",A
             W *27,"[37m"
  + 2       U 0 D KIL Q
```

```
ADMP2       ;入院台帳プログラム　１６［印刷　その２］　　;昭和60年 1月1
            0日;林寺 忠
A           D ^ADMP0 I Q=3 S %IOD=0 G EXIT
  + 1       S %QTY=2,%DTY="CRT",%IOD=0,%DEF=0 D ^%MIOS ;  GET OUTPUT DE
            VICE
  + 2       I '$D(%IOD) G EXIT
  + 3       I "CRT^LP"'[%DTY W !,?5,"出力デヴァイス ???" G A
  + 4       I "CRT,LP"[%DTY U %IOD U 0
  + 5       S NAMEK="ｱ",NO=0 I %IOD=0 W *26,# G ORDER
  + 6       W !,"プリンターにて出力実行中です。" U %IOD I $D(PM) W !,"選
            択条件 : ",PM,!
ORDER       S L=0 D LIN1 I ORDER="N" G G
C           S N=-1 D NAM G ASKCON:NAMEK="" G C:NF S KD="A" D BASE^ADMGE
            T
GA          S KD=$N(^ADM(NAMEK,N,KD)) I KD<0 G C
  + 1       D GET^ADMGET,PRINT
  + 2       G GA
NAM         S NAMEK=$N(^ADM(NAMEK)) I NAMEK<0!(NAMEK'?.ｶ1" ".ｶ) S NAMEK
            ="" Q
  + 1       S NF=0,N=$N(^ADM(NAMEK,N)) S:N<0!(N'?.N) NF=1 Q
G           S KD=$N(^ADMADM(KD)) I KD<0 G ASKCON
  + 1       S X=^(KD) F Y=1:1:X S DAT=^ADMADM(KD,Y),NAMEK=$P(DAT,"^",1)
            ,N=$P(DAT,"^",2) I $D(^ADM(NAMEK,N,KD,1))#10 D BASE^ADMGE
            T,GET^ADMGET,PRINT
  + 2       G G
ASKCON      W:%IOD # U 0 S X=0,Y=22,Z=1,S="N",PR="更に"_$S(ORDER="N":"入
            院日",1:"アイウエオ")_"順印刷を続けますか ? (Y/N)"
  + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3
  + 2       I R="Y"!(Q=2) W *26,# D KIL K NAMEK G A
  + 3       G EXIT
PRINT       I '$D(PM) S L=L+1,NO=NO+1 D L1^ADMDX1 G LCHK
PM          X PM I  S L=L+1,NO=NO+1 D L1^ADMDX1
LCHK        I '(L#60),%IOD W # D LIN1^ADMDX1 Q
  + 1       E  Q
LIN1        S KD="S60",NO=0 W:'%IOD *27,"[3;0H",*27,"[0J",*27,"[33m" W:
            %IOD ! W $S(ORDER="N":"来院日",1:"アイウエオ"),"順印刷 :
            " W:$D(PM) PM W ! D:'%IOD WHITE D LIN1^ADMDX1 Q
WHITE       W *27,"[37m" Q
KIL         K %IOD,%DTY,AID,DF,L,NF,NO,ORDER D KIL^ADMINP Q
EXIT        I %IOD U %IOD W # C %IOD
  + 1       U 0 D KIL Q


ADMDX1      ;入院台帳プログラム　１７［印刷　ﾌｫｰﾏｯﾄ］　　;昭和60年 1月1
            1日;林寺 忠
A           ;
LIN0        W !,"No.",?4,"氏　　名",?17,"性",?20,"誕生日",?29,"住
            所" Q
L0          W:%IOD ! W !,$J(NO,3),?4,NAME,?18,SEX,?20,BD,?29,ADR Q
LIN1        W !,"No.",?4,"氏　　名",?17,"性",?20,"年　齢",?28,"入院日",
            ?36,"入院時刻",?45,"退院日",?53,"主治医",?61,"紹介医" Q
L1          W !,$J(NO,3),?4,NAME,?18,SEX,?20,$J(AGE,3),?28,KD,?36,ADTIM
            E,?45,DISCH,?53,MD,?61,RMD Q
LIN2        I '%IOD W !,?4,"入院日",?12,"入院ID",?22,"入院時刻",?31,"
            退院日",?39,"主治医",?48,"紹介医" Q
  + 1       E  W !?4,"入院日",?12,"入院ID",?22,"入院時刻",?31,"退院日
            ",?39,"主治医",?48,"紹介医",?68,"診断名　１",?88,"診断名
             ２" Q
L2          I '%IOD W !?4,KD,?12,NID,?20,ADTIME,?29,DISCH,?37,MD,?46,RM
            D Q
  + 1       E  W !?4,KD,?12,NID,?20,ADTIME,?29,DISCH,?37,MD,?46,RMD,?66
            ,DX1,?86,DX2 Q
LIN3        W !,"No.",?4,"氏　　名",?17,"性",?20,"誕生日",?28,"入院日",
            ?36,"入院時刻",?45,"退院日",?53,"主治医",?61,"紹介医" Q
L3          W !,$J(NO,3),?4,"****　****",?18,SEX,?20,BD,?28,KD,?36,ADTI
            ME,?45,DISCH,?53,MD,?61,RMD Q
LIN4        W !,"入院日",?8,"入院ID",?18,"入院時刻",?27,"退院日",?35,
            "主治医",?44,"紹介医" Q
L4          W !,KD,?8,NID,?16,ADTIME,?25,DISCH,?35,MD,?42,RMD Q
LIN5        W !,"No.",?4,"氏　　名",?17,"入院日",?26,"診断名１",?45,"診
            断名２" Q
L5          W !,$J(NO,3),?4,NAME,?17,KD,?26,DX1,?45,DX2 Q
```

```
ADMGET      ;入院台帳プログラム　１８［印刷　その４］　　;昭和60年 1月1
            1日;林寺 忠
A           ;
BASE        D X S S=$T(X1),X=X1 D S S S=$T(X2),X=X2 D S Q
X           S X1=^ADM(NAMEK,N,1),X2=^(2) Q
GET         D GETD S S=$T(S1),X=S1 D S S S=$T(S2),X=S2 D S S S=$T(S3),X
            =S3 D S S S=$T(S4),X=S4 D S Q
GETD        S S1=^ADM(NAMEK,N,KD,1),S2=^(2),S3=^(3),S4=^(4) Q
VSHOW       S C=0 F I=0:1 S A=$T(X1+I) Q:$E(A,1,4)="EXIT"  D V1
VKIL        K A,B,C,I,J Q
V1          F J=2:1 S B=$P(A,";",J) Q:'$L(B)  S C=C+1 W:'(C-1#8) ! W ?C
            -1#8*10,B
  + 1       Q
S           F I=2:1 S A=$P(S,";",I) Q:A=""  S @A=$P(X,"^",I-1)
  + 1       Q
X1          ;AID;NAME;SEX;BD;PARENT
X2          ;ZIP;ADR;TEL;RTEL
S1          ;DX1;DX2;DX3
S2          ;DX4;DX5
S3          ;NID;AGE;MD;RMD;ADTIME;DISCH;SUMMARY
S4          ;FRSLT;MEM1;MEM2;MEM3
EXIT        Q

ADMP0       ;入院台帳プログラム　１９［条件設定］　　;昭和60年 1月1
            2日;林寺 忠
A           W *26,#
ORDER       S X=0,Y=3,Z=1,PR="印刷はアイウエオ順（Ａ）、入院日順（Ｎ）？
            "
  + 1       S S="N",QS="A または N で答えて下さい。",PM="I R=""A""!(R="
            "N"")"
  + 2       D ^READ G EXIT:Q>1 S ORDER=R
B           S X=0,Y=4,Z=1,PR="マンブス言語でその症例選択の条件を書けます
            か？"
  + 1       S S="N",QS="Y または N で答えてください。",PM="I R=""Y""!(R
            =""N"")"
  + 2       D ^READ G EXIT:Q>2,ORDER:Q=2,EXIT:R'="Y"
  + 3       D VSHOW^ADMGET
C           S X=0,Y=20,Z=60,PR="症例選択の条件 :",S="I 1",PM="I R?1""I
            "".E",$ZB=3
  + 1       K QS D ^RD G EXIT:Q=3,B:Q=2,GO
KIL         K PM,PR,QS,S,X,Y,Z,ZPL Q
GO          D KIL S PM=R K R Q
EXIT        D KIL K R Q

ADMSHOW     ;入院台帳プログラム　２０［データ表示］　　;昭和60年 1月1
            2日;林寺 忠
A           ;
SB          F I=0:1 S FM=$T(FM+I) Q:FM="FEND ;"  D DISP
  + 1       Q
SB1         F I=0:1 S FM=$T(FM+I) Q:FM="FEND ;"  S Y=$P(FM,";",2)+Z,X=$
            P(FM,";",3),VAR=$P(FM,";",5),PR=$P(FM,";",6) D MOVE W PR,
            " " I $D(@VAR),$L(@VAR) W @VAR
  + 1       Q
SD          F I=0:1 S FM=$T(DAT+I) Q:FM="DEND ;"  D DISP
  + 1       Q
SKD         D CLRLINE5 W *27,"[5;0H",*27,"[33m既に登録されている入院の月
            日は:" D WHITE
  + 1       S A="A" F I=1:1 S A=$N(^ADM(NAMEK,N,A)) Q:A<0  W:'(I-1#8) !
             W ?I-1#8*10,A S KD=A
  + 2       Q
DISP        S Y=$P(FM,";",2),X=$P(FM,";",3),VAR=$P(FM,";",5),PR=$P(FM,"
            ;",6)
  + 1       S:$D(@VAR) S=@VAR D MOVE W PR," " W:$L(VAR)&$D(@VAR) @VAR Q

CLRLINE5    W *27,"[5;0H",*27,"[0J" Q
WHITE       W *27,"[37m" Q
BTMOFF      W *27,"[>1h" Q
ERAS        W *27,"[22;0H",$J(" ",79) Q
MOVE        W *27,"[",Y,";",X,"H" Q
FM          ;1;0;;NAMEK;氏名(ｶﾀｶﾅ) :;;
  + 1       ;1;30;;NAME;氏名(漢字) :;;
  + 2       ;1;64;;SEX;SEX :;;;
  + 3       ;2;64;;BD;誕生日 :;;
FEND        ;
```

```
DAT         ;4;0;7;NID;入院 ID# :;I R?2N1"-"4N;例えば 60-0001 のようにいれる。
  + 1       ;4;30;3;AGE;年令(例:12Y,3M) :;I R?.N!(R?.N1"M")!(R?.N1"Y")!(R?.N1"D");
  + 2       ;5;0;68;DX1;診断名 1:;I R?.TP;漢字で診断名を入れて下さい。
  + 3       ;6;0;68;DX2;診断名 2:;I R?.TP;漢字で診断名を入れて下さい。
  + 4       ;7;0;68;DX3;診断名 3:;I R?.TP;漢字で診断名を入れて下さい。
  + 5       ;8;0;68;DX4;診断名 4:;I R?.TP;漢字で診断名を入れて下さい。
  + 6       ;9;0;68;DX5;診断名 5:;I R?.TP;漢字で診断名を入れて下さい。
  + 7       ;10;0;28;MD;主治医 :;S R=$S(R=1:"林寺",R=2:"上坂",R=3:"西角",R=4:"和田",1:R) I R?.TP;漢字で指定するか 1=林寺、2=上坂、3=西角、4=和田 と答えて下さい。
  + 8       ;10;40;26;RMD;紹介医の氏名 :;I R?.TP;漢字で答えて下さい。
  + 9       ;11;0;8;ADTIME;入院時刻(HH:MM AM/PM) :;I R?2N1":"2N1" AM"!(R?2N1":"2N1" PM");例えば、10:15 AM や 20:38 PM のように入れる。
  +10       ;11;40;7;DISCH;退院日(SYYMMDD) :;I R?1"S"6N
  +11       ;11;71;1;SUMMARY;総括 :;I R="Y"!(R="N");
  +12       ;12;0;68;FRSLT;転帰 :;I R?.TP;
MEM         ;13;0;70;MEM1;ｺﾒﾝﾄ 1 :;;
  + 1       ;14;0;70;MEM2;ｺﾒﾝﾄ 2 :;;
  + 2       ;15;0;70;MEM3;ｺﾒﾝﾄ 3 :;;
DEND        ;

ADMFILE     ;入院台帳プログラム　２１［データ記憶］　　;昭和60年 1月12日;林寺 忠
A           ;
BASE        S FADD=0 I '($D(^ADM(NAMEK))#10) G FILE
C           D ERAS S X=0,Y=22,Z=1,S="N",PM="I R=""Y""!(R=""N"")"
  + 1       S PR="同名の人がいますが追加しますか ? (Y/N) " D ^READ I Q=3 G EXIT
  + 2       S:R'="Y" NAMEK="" G:R'="Y" EXIT
CONF        D ERAS W *7 S X=0,Y=22,Z=1,S="N",PM="I R=""Y""!(R=""N"")"
  + 1       S PR="同名の人を本当に追加しますか ? (Y/N) " D ^READ I Q=3 G EXIT
  + 2       G:R'="Y" EXIT S FADD=1
FILE        D X I 'FADD S ^ADM(NAMEK)=0 S:'($D(^ADM)#10) ^ADM=0 S I=^ADM,I=I+1,^ADM=I
  + 1       S N=^ADM(NAMEK),N=N+1,^ADM(NAMEK)=N
SAV         S ^ADM(NAMEK,N,1)=X1,^(2)=X2 D ^ZOPTION Q
EDIT        D X G SAV
SAVD        S S=$T(S1) D T S S1=X,S=$T(S2) D T S S2=X,S=$T(S3) D T S S3=X,S=$T(S4) D T S S4=X
SVD         S ^ADM(NAMEK,N,KD,1)=S1,^(2)=S2,^(3)=S3,^(4)=S4 D ^ZOPTION Q
TRNSF       S ND=-1,KD=$N(^ADM(NAMOLD,NOLD,KD)) I KD<0 K ND Q
TR1         S ND=$N(^ADM(NAMOLD,N,KD,ND)) I ND<0 G TRNSF
  + 1       S S=^ADM(NAMOLD,NOLD,KD,ND),^ADM(NAMNEW,NNEW,KD,ND)=S G TR1
DEL         S NX=NOLD,M=^ADM(NAMEK),^(NAMEK)=M-1 I M<2 S I=^ADM-1,^ADM=I
  + 1       I M'=NX S NAMNEW=NAMOLD F I=NX:1:M-1 S N=I+1 D X^ADMGET S N=I K ^ADM(NAMEK,N) D SAV S KD="S",NOLD=I+1,NNEW=I,N=I+1 D TRNSF
  + 2       I NX'=M K ^ADM(NAMEK,M)
  + 3       S M=M-1 I 'M K ^ADM(NAMEK)
  + 4       Q
X           S S=$T(X1) D T S X1=X,S=$T(X2) D T S X2=X Q
T           F I=2:1 S A=$P(S,";",I) Q:A=""  S:'$D(@A) @A="" S:I=2 X=@A S:I'=2 X=X_"^"_@A
  + 1       Q
X1          ;AID;NAME;SEX;BD;PARENT
X2          ;ZIP;ADR;TEL;RTEL
S1          ;DX1;DX2;DX3
S2          ;DX4;DX5
S3          ;NID;AGE;MD;RMD;ADTIME;DISCH;SUMMARY
S4          ;FRSLT;MEM1;MEM2;MEM3
ERAS        W *27,"[22;0H",$J(" ",79) Q
EXIT        Q
```

```
ADMINP      ;入院台帳プログラム　２２［データ入力］　　;昭和60年 1月1
            2日;林寺 忠
A           W #;
IND         F I=0:1 S FM=$T(DAT+I) Q:FM="DEND ;"  D:I=1 ^ADMAGE D ITEM,
            ^READ W:Q=2&(I=0) *7 S:Q=2 I=I-1-(I>0) Q:Q=3  D MOVE S:Q'
            =2 @VAR=R
  + 1       Q
SB          F I=0:1 S FM=$T(FM+I) Q:FM="FEND ;"  D DISP
  + 1       Q
SB1         F I=0:1 S FM=$T(FM+I) Q:FM="FEND ;"  S Y=$P(FM,";",2)+Z,X=$
            P(FM,";",3),VAR=$P(FM,";",5),PR=$P(FM,";",6) D MOVE W PR,
            " " I $D(@VAR),$L(@VAR) W @VAR
  + 1       Q
SD          F I=0:1 S FM=$T(DAT+I) Q:FM="DEND ;"  D DISP
  + 1       Q
SD1         F I=0:1 S FM=$T(DAT+I) Q:$E(FM,1,5)="MEM ;"  D DISP
  + 1       Q
SKD         D CLRLINE5 W *27,"[5;0H",*27,"[33m既に登録されている入院の月
            日は:" D WHITE
  + 1       S A="A" F I=1:1 S A=$N(^ADM(NAMEK,N,A)) Q:A<0  W:'(I-1#8) !
             W ?I-1#8*10,A S KD=A
  + 2       Q
B           W #
BSHOW       F I=0:1 S FM=$T(FMT+I) Q:FM="FMEND ;"  D ITEM,DISP
  + 1       Q
BINP        F I=0:1 S FM=$T(FMT+I) Q:FM="FMEND ;"  D ITEM,^READ W:Q=2&(
            I=0) *7 S:Q=2 I=I-1-(I>0) Q:Q=3  D MOVE S:Q'=2 @VAR=R
  + 1       Q
DISP        S Y=$P(FM,";",2),X=$P(FM,";",3),VAR=$P(FM,";",5),PR=$P(FM,"
            ;",6)
  + 1       S:$D(@VAR) S=@VAR D MOVE W PR," " W:$L(VAR)&$D(@VAR) @VAR Q

ITEM        S Y=$P(FM,";",2),X=$P(FM,";",3),Z=$P(FM,";",4),VAR=$P(FM,";
            ",5)
  + 1       S PR=$P(FM,";",6),PM=$P(FM,";",7),QS=$P(FM,";",8),S=""
  + 2       S:$D(@VAR) S=@VAR K:'$L(PM) PM K:'$L(QS) QS Q
MOVE        W *27,"[",Y,";",X,"H" Q
CLRLINE5    W *27,"[5;0H",*27,"[0J" Q
WHITE       W *27,"[37m" Q
FM          ;1;0;;NAMEK;氏名(ｶﾀｶﾅ) :;;
  + 1       ;1;30;;NAME;氏名(漢字) :;;
  + 2       ;1;64;;SEX;SEX :;;;
  + 3       ;2;64;;BD;誕生日 :;;
FEND        ;
FMT         ;3;0;9;AID;外来ＩＤ :;I R?2N1"-"4N1"-"1N;例えば :12-3456-7
            のように答えて下さい。
  + 1       ;3;28;26;NAME;氏名(漢字) :;I R?.T1" ".T;名字と名前の間にス
            ペースを入れて下さい。
  + 2       ;3;70;1;SEX;性 :;I R="M"!(R="F");M(男児) または F(女児
            ) を入れて下さい。
  + 3       ;4;0;7;BD;生年月日 :;I R?1"S"6N;例えば S581027 のように入れ
            て下さい。
  + 4       ;4;52;16;PARENT;親の名前 :;I R?.TP;漢字で名前を入れて下さい
            。
  + 5       ;5;0;72;ADR;住所 :;I R?.TP;漢字で答えて下さい。
  + 6       ;6;0;6;ZIP;郵便番号 :;I R?3N!(R?3N1"-"2N);例えば 612 のよう
            に入れて下さい。
  + 7       ;6;19;12;TEL;電話番号 :;I R?4N1"-"2N1"-"4N!(R?3N1"-"4N)!(R?
            3N1"-"3N1"-"4N)!(R?2N1"-"3N1"-"4N);例えば 075-640-2850 ま
            たは 640-2850 のように入れて下さい。
  + 8       ;6;43;12;RTEL;連絡先の電話番号 :;I R?4N1"-"2N1"-"4N!(R?3N1"
            -"4N)!(R?3N1"-"3N1"-"4N)!(R?2N1"-"3N1"-"4N);例えば 075-64
            0-2850 または 640-2850 のように入れて下さい。
FMEND       ;
DAT         ;4;0;7;NID;入院 ID# :;I R?2N1"-"4N;例えば 60-0001 のようにい
            れる。
  + 1       ;4;30;3;AGE;年令(例:12Y,3M) :;I R?.N!(R?.N1"M")!(R?
            .N1"Y")!(R?.N1"D");
  + 2       ;5;0;68;DX1;診断名 1:;I R?.TP;漢字で診断名を入れて下さい。

  + 3       ;6;0;68;DX2;診断名 2:;I R?.TP;漢字で診断名を入れて下さい。

  + 4       ;7;0;68;DX3;診断名 3:;I R?.TP;漢字で診断名を入れて下さい。
```

```
  + 5           ;8;0;68;DX4;診断名 4:;I R?.TP;漢字で診断名を入れて下さい。

  + 6           ;9;0;68;DX5;診断名 5:;I R?.TP;漢字で診断名を入れて下さい。

  + 7           ;10;0;28;MD;主治医 :;S R=$S(R=1:"林寺",R=2:"上坂",R=3:"西角
                ",R=4:"和田",1:R) I R?.TP;漢字で指定するか 1=林寺、2=上坂
                、3=西角、4=和田 と答えて下さい。
  + 8           ;10;40;26;RMD;紹介医の氏名 :;I R?.TP;漢字で答えて下さい。
  + 9           ;11;0;8;ADTIME;入院時刻(HH:MM AM/PM) :;I R?2N1"
                :"2N1" AM"!(R?2N1":"2N1" PM");例えば、10:15 AM や
                20:38 PM のように入れる。
  +10           ;11;40;7;DISCH;退院日(SYYMMDD) :;I R?1"S"6N
SUMMARY         ;11;71;1;SUMMARY;総括 :;I R="Y"!(R="N");
FRSLT           ;12;0;68;FRSLT;転帰 :;S R=$S(R=1:"治癒",R=2:"軽快",R=3:"不変
                ",R=4:"増悪",R=5:"死亡",R=6:"続発症による死亡",R=7:"検査終
                了",R=8:"その他",1:R) I R?.TP;1=治癒、2=軽快、3=不変、4=
                増悪、5=死亡、6=続発症による死亡、7=検査終了、8=その他
MEM             ;13;0;70;MEM1;ｺﾒﾝﾄ 1 :;;適当な文字で答えて下さい。
MEM2            ;14;0;70;MEM2;ｺﾒﾝﾄ 2 :;;
MEM3            ;15;0;70;MEM3;ｺﾒﾝﾄ 3 :;;
DEND            ;
KIL             K A,ADR,ADTIME,AGE,AID,BD,DISCH,DX1,DX2,DX3,DX4,DX5,FADD,FM
                ,FRSLT,I,KD,MEM1,MEM2,MEM3,MD,N,NAME,NID,PARENT,PM,PO2,PR
                ,Q,QS,R,RMD,RTEL,S,S1,S2,S3,S4,SEX,SUMMARY,TEL,VAR,X,X1,X
                2,Y,Z,ZIP,ZPL Q
EXIT            Q

ADMN2           ;入院台帳プログラム ２３［インデックス削除］;昭和60年 1月1
                3日;林寺 忠
A               K QS
  + 1           W *26,# S X=0,Y=22,Z=20,PR="氏名は(ｶﾀｶﾅ) ? ",PM="S:R?
                1ｶ R=R_"" -"" I R?.ｶﾄ1"" "".ｶﾄ",S="" S:$D(NAMEK) S=NAMEK
                D ^READ G EXIT:Q>1!(R="") S NAMEK=R
B               D ^ADMNAM I 'N G A
  + 1           W # D SRCH,BASE^ADMGET,SB^ADMINP,SKD^ADMINP
ASKD            S X=0,Y=22,Z=7,S="",PM="I R?1""S""6N!(R?1""*S""6N),$E($P(R,
                ""S"",2),1,2)<80,$E($P(R,""S"",2),3,4)<13,$E($P(R,""S"",2
                ),5,6)<32,$D(^ADM(NAMEK,N,R))"
  + 1           S VAR="KD" S:$D(@VAR) S=@VAR S PR="入院年日 (例えば SYYMMD
                D) :"
  + 2           D ^READ G A:Q=2,EXIT:Q=3
  + 3           D CLRLINE5 S KD=R
C               W # S Z=1 D WHITE,GET^ADMGET,SB1^ADMINP,SD^ADMINP
D               W *27,"[0;30H",*27,"[32mインデックス削除",!,"氏名(ｶﾀｶﾅ
                ) : ",NAMEK,!,"入院年月日 : ",KD
SHOW            W *27,"[16;0H",*27,"[33m登録されているインデックスは",*27,"
                [32m",!
  + 1           K INDX S N=-1 F I=0:1 S N=$N(^ADMINDX(N)) Q:N<0  W:$X>65 !
                W $J(I+1,3),")",N S INDX(I+1)=N
  + 2           S N=I W !
ASKINDX         S X=0,Y=22,Z=20,PR="インデックスにする文字（例えば＜１＞）を
                入れて下さい",S=""
  + 1           S PM="I R?.TｶA!(R?.N&(R>0)&(R'>N))" D ^READ G EXIT:Q=3,A:Q
                =2
  + 2           W:R="" *7 G ASKINDX:R="" S INDX=R I R?.N,R'>N S INDX=INDX(R
                )
CONF            D ERAS S X=0,Y=22,Z=1,PR="ｲﾝﾃﾞｯｸｽ はく"_INDX_">でよろ
                しいか (Y/N) ? ",S="Y"
  + 1           S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3,ASKINDX:Q=2!(
                R="N")
  + 2           S DAT=NAMEK_"^"_NX G NOREG:'($D(^ADMINDX(INDX,KD))#10) D CH
                ECK I G DEL
NOREG           D ERAS W *7,*27,"[22;0H",*27,"[33m","[",INDX,"]",*27,"[31m
                は登録されていません。" H 2 D ERAS G ASKCON
DEL             S N=^ADMINDX(INDX),N=N-1,^(INDX)=N I N'>0 K ^ADMINDX(INDX)
                G DFIN
  + 1           S N=^ADMINDX(INDX,KD),N=N-1,^(KD)=N I N'>0 K ^ADMINDX(INDX,
                KD) G DFIN
  + 2           S X=0,Y=-1 F I=1:1:N+1 S Y=$N(^ADMINDX(INDX,KD,Y)) Q:Y<0!(Y
                '?.N) I DAT=^(Y) S X=I Q
  + 3           S (Y,Z)=X I 'X G NOREG
  + 4           F I=X:1:N S Z=$N(^ADMINDX(INDX,KD,Y)),^(Y)=^(Z),Y=Z
  + 5           K ^ADMINDX(INDX,KD,Z)
DFIN            D ERAS,^ZOPTION
```

```
ASKCON      S X=0,Y=22,Z=1,S="N",PR="更にインデックス削除を続けますか ?
             (Y/N)"
  + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3
  + 2       I R="Y"!(Q=2) W *26,# D KIL K NAMEK G A
  + 3       G EXIT
CHECK       S X=^ADMINDX(INDX,KD),Y=0 F I=1:1:X I ^ADMINDX(INDX,KD,I)=D
            AT S Y=1 Q
  + 1       I Y Q
  + 2       Q

SRCH        G:N=1 GET W ! F I=1:1:N S A=^ADM(NAMEK,I,1) W !,I,") ",$P(A
            ,"^",1),?11,$P(A,"^",2),?31,$P(A,"^",3),?35,$P(A,"^",4)
  + 1       W !!,*27,"[33m","上のうちどの人のインデックスを削除しますか
            ? " D WHITE R A I A<1!(A>N) W *7,*27,"[22;0H" G SRCH
  + 2       S N=A W *26,# D DISP
GET         S NX=N Q
CLRLINE5    W *27,"[5;0H",*27,"[0J" Q
WHITE       W *27,"[37m" Q
MOVE        W *27,"[",Y,";",X,"H" Q
ERAS        W *27,"[22;0H",$J(" ",79) Q
KIL         K DAT,INDX,NO D KIL^ADMINP Q
EXIT        D KIL,WHITE Q


ADMN3       ;入院台帳プログラム　２４［インデックス表示］;昭和60年 1月1
            3日;林寺 忠
A           K QS
  + 1       W *26,# W *27,"[0;26H",*27,"[32mインデックスによるデータ表示
            "
SHOW        W *27,"[2;0H",!,*27,"[33m登録されているインデックスは",*27,
            "[32m",!
  + 1       K INDX S N=-1 F I=0:1 S N=$N(^ADMINDX(N)) Q:N<0  W:2*$L(N)+
            $X>74 ! W $J(I+1,3),")",N S INDX(I+1)=N
  + 2       S N=I W !
ASKINDX     S X=0,Y=22,Z=20,PR="インデックスにする文字（例えば＜１＞）を
            入れて下さい",S=""
  + 1       S PM="I R?.TカA!(R?.N&(R>0)&(R'>N))" D ^READ G EXIT:Q=3,A:Q
            =2
  + 2       W:R="" *7 G ASKINDX:R="" S INDX=R I R?.N,R'>N S INDX=INDX(R
            )
CONF        D ERAS S X=0,Y=22,Z=1,PR="ｲﾝﾃﾞｯｸｽ はく"_INDX_">でよろ
            しいか (Y/N) ? ",S="Y"
  + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3,ASKINDX:Q=2!(
            R="N")
  + 2       I '($D(^ADMINDX(INDX))#10) G NOREG
GETKD       S KD=-1,NO=1 W *27,"[3;0H",*27,"[0J",*27,"[33mインデックス
            : ",INDX,! D WHITE,LIN1^ADMDX1
G           S KD=$N(^ADMINDX(INDX,KD)) I KD<0 G ASKCON
  + 1       S X=^(KD) F Y=1:1:X S DAT=^ADMINDX(INDX,KD,Y),NAMEK=$P(DAT,
            "^",1),N=$P(DAT,"^",2) I $D(^ADM(NAMEK,N,KD,1))#10 D BASE
            ^ADMGET,GET^ADMGET,L1^ADMDX1 S NO=NO+1
  + 2       G G
NOREG       D ERAS W *7,*27,"[22;0H",*27,"[33m","[",INDX,"]",*27,"[31m
            は登録されていません。" H 2 D ERAS G ASKCON
ASKCON      S X=0,Y=22,Z=1,S="N",PR="更にインデックス表示を続けますか ?
             (Y/N)"
  + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3
  + 2       I R="Y"!(Q=2) W *26,# D KIL K NAMEK G A
  + 3       G EXIT
CLRLINE5    W *27,"[5;0H",*27,"[0J" Q
WHITE       W *27,"[37m" Q
MOVE        W *27,"[",Y,";",X,"H" Q
ERAS        W *27,"[22;0H",$J(" ",79) Q
KIL         K DAT,INDX,NO D KIL^ADMINP Q
EXIT        D KIL,WHITE Q


ADME1       ;入院台帳プログラム　２５［日付変更］　　;昭和60年 1月1
            3日;林寺 忠
A           K QS
  + 1       W *26,# S X=0,Y=22,Z=20,PR="氏名は(ｶﾀｶﾅ) ? ",PM="S:R?
            1カ R=R_"" -"" I R?.ｶﾄ1"" "".ｶﾄ",S="" S:$D(NAMEK) S=NAMEK
            D ^READ G EXIT:Q>1!(R="") S NAMEK=R
B           D ^ADMNAM I 'N G A
  + 1       W # D SRCH,BASE^ADMGET,SB^ADMINP,SKD^ADMINP
```

```
ASKD        S X=0,Y=22,Z=7,S="",PM="I R?1""S""6N!(R?1""*S""6N),$E($P(R,
            ""S"",2),1,2)<80,$E($P(R,""S"",2),3,4)<13,$E($P(R,""S"",2
            ),5,6)<32,$D(^ADM(NAMEK,N,R))"
  + 1       S VAR="KD" S:$D(@VAR) S=@VAR S PR="入院年日 (例えば SYYMMD
            D) :"
  + 2       D ^READ G A:Q=2,EXIT:Q=3
  + 3       D CLRLINE5 S KD=R
C           W # S Z=1 D WHITE,GET^ADMGET,SB1^ADMINP,SD^ADMINP
D           W *27,"[0;30H",*27,"[32m入院年月日の変更",!,"氏名(ｶﾀｶﾅ
            ) : ",NAMEK,!,"入院年月日 : ",KD
ASKX        S X=0,Y=22,Z=7,PR="新しい入院年月日は？ ",S=KD,PM="I R?1""S
            ""6N"
  + 1       S QS="例えば、S601011 のように答えてください。" D ^READ G E
            XIT:Q=3,A:Q=2
  + 2       W *27,"[19;0H","[",KD,"] を ",*27,"[33m[",R,"] ",*27,"[37mに
            変更"
  + 3       I R=KD W *27,"[31mは出来ません",*7 G ASKX
  + 4       W "します。      " S A=R
CONF        D ERAS S X=0,Y=22,Z=1,PR="それでよろしいか (Y/N) ? ",S="Y"
  + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3,ASKX:Q=2!(R="
            N")
  + 2       I $D(^ADM(NAMEK,NX,A)) W *7 D ERAS W *27,"[22;0H[",A,"] が既
            に登録されていますので、そのような変更は出来ません。" H 4
             D ERAS G ASKX
CHNG        S KDOLD=KD,N=NX,KD=A D SVD^ADMFILE
  + 1       K ^ADM(NAMEK,N,KDOLD) D ^ZOPTION
ASKCON      S X=0,Y=22,Z=1,S="N",PR="更に日付変更を続けますか ? (Y/N)"
  + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3
  + 2       I R="Y"!(Q=2) W *26,# D KIL K NAMEK G A
  + 3       G EXIT
CHECK       S X=^ADMINDX(INDX,KD),Y=0 F I=1:1:X I ^ADMINDX(INDX,KD,I)=D
            AT S Y=1 Q
  + 1       I Y Q
  + 2       Q
SRCH        G:N=1 GET W ! F I=1:1:N S A=^ADM(NAMEK,I,1) W !,$J(I,2),")
            ",$P(A,"^",1),?15,$P(A,"^",2),?28,$P(A,"^",3),?32,$P(A,"^
            ",4)
  + 1       W !!,*27,"[33m","上のうちどの人の日付変更をしますか？ " D WH
            ITE R A I A<1!(A>N) W *7,*27,"[22;0H" W # G SRCH
  + 2       S N=A W #
GET         S NX=N Q
CLRLINE5    W *27,"[5;0H",*27,"[0J" Q
WHITE       W *27,"[37m" Q
MOVE        W *27,"[",Y,";",X,"H" Q
ERAS        W *27,"[22;0H",$J(" ",79) Q
KIL         K FM,KDOLD,NX,QS D KIL^ADMINP Q
EXIT        D KIL,WHITE Q


ADMD1       ;入院台帳プログラム　２６［日付削除］          ;昭和60年 1月1
            3日;林寺 忠
A           K QS
  + 1       W *26,# S X=0,Y=22,Z=20,PR="氏名は(ｶﾀｶﾅ) ? ",PM="S:R?
            1ｶ R=R_"" -"" I R?.ｶﾄ1"" "".ｶﾄ",S="" S:$D(NAMEK) S=NAMEK
            D ^READ G EXIT:Q>1!(R="") S NAMEK=R
B           D ^ADMNAM I 'N G A
  + 1       W # D SRCH,BASE^ADMGET,SB^ADMINP,SKD^ADMINP
ASKD        S X=0,Y=22,Z=7,S="",PM="I R?1""S""6N!(R?1""*S""6N),$E($P(R,
            ""S"",2),1,2)<80,$E($P(R,""S"",2),3,4)<13,$E($P(R,""S"",2
            ),5,6)<32,$D(^ADM(NAMEK,N,R))"
  + 1       S VAR="KD" S:$D(@VAR) S=@VAR S PR="入院年日 (例えば SYYMMD
            D) :"
  + 2       D ^READ G A:Q=2,EXIT:Q=3
  + 3       D CLRLINE5 S KD=R
C           W # S Z=1 D WHITE,GET^ADMGET,SB1^ADMINP,SD^ADMINP
D           W *27,"[0;30H",*27,"[32m入院年月日の削除",!,"氏名(ｶﾀｶﾅ
            ) : ",NAMEK,!,"入院年月日 : ",KD
E           W *27,"[19;0H",*27,"[33m[",KD,"] ",*27,"[37mを削除します。"

CONF        D ERAS S X=0,Y=22,Z=1,PR="それでよろしいか (Y/N) ? ",S="N"
  + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3,B:Q=2,ASKCON:
            R="N"
  + 2       I '$D(^ADM(NAMEK,NX,KD)) W *7 D ERAS W *27,"[22;0H[",A,"] は
            記録にありませんので出来ません。" H 4 D ERAS G ASKD
```

```
 DEL          K ^ADM(NAMEK,NX,KD) D ^ZOPTION
 ASKCON       S X=0,Y=22,Z=1,S="N",PR="更に日付削除を続けますか ? (Y/N)"
   + 1        S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3
   + 2        I R="Y"!(Q=2) W *26,# D KIL K NAMEK G A
   + 3        G EXIT
 CHECK        S X=^ADMINDX(INDX,KD),Y=0 F I=1:1:X I ^ADMINDX(INDX,KD,I)=I
              AT S Y=1 Q
   + 1        I Y Q
   + 2        Q
 SRCH         G:N=1 GET W ! F I=1:1:N S A=^ADM(NAMEK,I,1) W !,$J(I,2),")
              ",$P(A,"^",1),?15,$P(A,"^",2),?28,$P(A,"^",3),?32,$P(A,"^
              ",4)
   + 1        W !!,*27,"[33m","上のうちどの人の日付を削除しますか? " D WH
              ITE R A I A<1!(A>N) W *7,*27,"[22;0H",# G SRCH
   + 2        S N=A W *26,#
 GET          S NX=N Q
 CLRLINE5     W *27,"[5;0H",*27,"[0J" Q
 WHITE        W *27,"[37m" Q
 MOVE         W *27,"[",Y,";",X,"H" Q
 ERAS         W *27,"[22;0H",$J(" ",79) Q
 KIL          K FM,NX,QS D KIL^ADMINP Q
 EXIT         D KIL,WHITE Q

ADMP3        ;入院台帳プログラム　２７［印刷　その３］　　;昭和60年 1月1
             3日;林寺 忠
A            D B^ADMP0 G EXIT:Q=3
   + 1       W *26,# W *27,"[0;26H",*27,"[32mインデックスによるデータ印刷
             "
SHOW         W *27,"[2;0H",!,*27,"[33m登録されているインデックスは",*27,
             "[32m",!
   + 1       K INDX S N=-1 F I=0:1 S N=$N(^ADMINDX(N)) Q:N<0  W:$L(N)*2+
             $X>76 ! W $J(I+1,3),")",N S INDX(I+1)=N
   + 2       S N=I W !
 ASKINDX     S X=0,Y=22,Z=20,PR="インデックスにする文字（例えば＜１＞）を
             入れて下さい",S=""
   + 1       S PM="I R?.TカA!(R?.N&(R>0)&(R'>N))" D ^READ G EXIT:Q=3,A:Q
             =2
   + 2       W:R="" *7 G ASKINDX:R="" S INDX=R I R?.N,R'>N S INDX=INDX(R
             )
 CONF        D ERAS S X=0,Y=22,Z=1,PR="ｲﾝﾃﾞｯｸｽ はく"_INDX_">でよろ
             しいか (Y/N) ? ",S="Y"
   + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3,ASKINDX:Q=2!(
             R="N")
   + 2       I '($D(^ADMINDX(INDX))#10) G NOREG
 H           S %QTY=2,%DTY="CRT",%IOD=0,%DEF=0 D ^%MIOS ;  GET OUTPUT DE
             VICE
   + 1       I '$D(%IOD) G EXIT
   + 2       I "CRT^LP"'[%DTY W !,?5,"出力デヴァイス ???" G A
   + 3       I "CRT,LP"[%DTY U %IOD U 0
   + 4       S NAMEK="ｱ",NO=0 I %IOD=0 W *26,# G GETKD
   + 5       W !,"プリンターにて出力実行中です。" U %IOD
 GETKD       S KD=-1,NO=0 W:'%IOD *27,"[3;0H",*27,"[0J",*27,"[33m" W:%IO
             D ! W "インデックス : ",INDX,! D:'%IOD WHITE D LIN1^ADMDX
             1
 G           S KD=$N(^ADMINDX(INDX,KD)) I KD<0 W:%IOD # G ASKCON
   + 1       S X=^(KD) F Y=1:1:X S DAT=^ADMINDX(INDX,KD,Y),NAMEK=$P(DAT,
             "^",1),N=$P(DAT,"^",2) I $D(^ADM(NAMEK,N,KD,1))#10 D BASE
             ^ADMGET,GET^ADMGET,PRINT
   + 2       G G
 NOREG       U 0 D ERAS W *7,*27,"[22;0H",*27,"[33m","[",INDX,"]",*27,"[
             31m は登録されていません。" H 2 D ERAS G ASKCON
 ASKCON      U 0 S X=0,Y=22,Z=1,S="N",PR="更にインデックス印刷を続けます
             か ? (Y/N)"
   + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3
   + 2       I R="Y"!(Q=2) W *26,# D KIL K NAMEK G A
   + 3       G EXIT
 PRINT       I '$D(PM) S NO=NO+1 D L1^ADMDX1 Q
 PM          X PM I  S NO=NO+1 D L1^ADMDX1
   + 1       Q
 CLRLINE5    W *27,"[5;0H",*27,"[0J" Q
 ERAS        W *27,"[22;0H",$J(" ",79) Q
 WHITE       W *27,"[37m" Q
 KIL         K %IOD,%DTY,AID,DF,NF,NO D KIL^ADMINP Q
 EXIT        I $D(%IOD),%IOD C %IOD
   + 1       U 0 D KIL Q
```

```
ADMP4       ;入院台帳プログラム　２８［印刷　その４］　　;昭和60年 1月1
            3日;林寺 忠
A           D B^ADMP0 I $D(PM) S PMX=PM
  + 1       W *26,# S X=0,Y=22,Z=20,PR="氏名は(ｶﾀｶﾅ) ? ",PM="S:R?
            1ｶ R=R_"" -"" I R?.ｶﾄ1"" "".ｶﾄ",S="" S:$D(NAMEK) S=NAMEK
            D ^READ G EXIT:Q>1!(R="") S NAMEK=R
B           D ^ADMNAM I 'N G A
C           S %QTY=2,%DTY="CRT",%IOD=0,%DEF=0 D ^%MIOS ;  GET OUTPUT DE
            VICE
  + 1       I '$D(%IOD) G EXIT
  + 2       I "CRT^LP"'[%DTY W !,?5,"出力デヴァイス ???" G A
  + 3       I "CRT,LP"[%DTY U %IOD U 0
  + 4       I %IOD=0 W *26,# G D
  + 5       W !,"プリンターにて出力実行中です。" U %IOD
D           S KD="S60",NO=0 W:'%IOD *27,"[3;0H",*27,"[0J",*27,"[33m" W:
            %IOD !
  + 1       K PM I $D(PMX) S PM=PMX K PMX
  + 2       W:$D(PM) "条件は : ",PM,! D:'%IOD WHITE D BASE^ADMGET,L1,LI
            N4^ADMDX1
G           S KD=$N(^ADM(NAMEK,N,KD)) I KD<0 G ASKCON
  + 1       D GET^ADMGET,PRINT
  + 2       G G
ASKCON      U 0 S X=0,Y=22,Z=1,S="N",PR="更に患者個人の印刷を続けますか
            ?(Y/N)"
  + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3
  + 2       I R="Y"!(Q=2) W *26,# D KIL K NAMEK G A
  + 3       G EXIT
PRINT       I '$D(PM) S NO=NO+1 D L4^ADMDX1 Q
PM          X PM I  S NO=NO+1 D L4^ADMDX1
  + 1       Q
L1          W !,"基礎データ: ",NAME,?40,SEX,?42,BD,"生　親の氏名:",PARE
            NT,!,"住所:",ADR,?60,"TEL:",TEL,! Q
CLRLINE5    W *27,"[5;0H",*27,"[0J" Q
ERAS        W *27,"[22;0H",$J(" ",79) Q
WHITE       W *27,"[37m" Q
KIL         K %IOD,%DTY,A,ADTIME,ADR,AGE,AID,BD,DF,DISCH,DX1,DX2,DX3,DX
            4,DX5,FRSLT,I,KD,MD,MEM1,MEM2,MEM3,N,NAME,NF,NID,NO,PAREN
            T,PM,PR,Q,R,RMD,RTEL,S,S1,S2,S3,S4,SEX,SUMMARY,TEL,X,X1,X
            2,Y,Z,ZIP,ZPL Q
EXIT        I $D(%IOD),%IOD U %IOD W ! C %IOD
  + 1       U 0 D KIL Q


ADMIND      ;入院台帳プログラム　２９［入力日付］　　:昭和60年 2月
            7日;林寺 忠
A           W # S S="" I $D(^ADMD)#10 S S=^ADMD
B           W !!,"最終の入力の日付は: ",$S($L(S):S_" です。",1:"記録にあ
            りません。")
  + 1       S QS="例えば、S600115 ののように最終の日付を入れて下さい。"

  + 2       S X=0,Y=5,Z=7,PR="入力の最終の日付は？",PM="I R?1""S""6N" D
            ^READ G EXIT:Q>1
  + 3       I $L(R) S ^ADMD=R
KIL         K PM,PR,Q,QS,R,S,X,Y,Z,ZPL Q
EXIT        D KIL Q


ADMSM1      ;入院台帳プログラム　３０［記録 その２］　　:昭和60年 1月
            3日;林寺 忠
A           K QS
  + 1       W *26,# D SB^ADMINP
  + 2       S KD=$N(^ADM(NAMEK,N,"A")) I KD<0 K KD
  + 3       I $D(KD) D SKD^ADMINP
ASKD        S X=0,Y=22,Z=7,S="",PM="I R?1""S""6N!(R?1""*S""6N),$E($P(R,
            ""S"",2),1,2)<80,$E($P(R,""S"",2),3,4)<13,$E($P(R,""S"",2
            ),5,6)<32"
  + 1       S VAR="KD" S:$D(@VAR) S=@VAR S PR="入院年日（例えば SYYMMD
            D) :"
  + 2       D ^READ I Q>1 G EXIT
  + 3       D CLRLINE5 S KD=R
GET         I '($D(^ADM(NAMEK,N,KD,1))#10) G ST
  + 1       E  S X=0,Y=22 D MOVE W "入院月日 : ",KD," は既に記録にありま
            した。",*7 H 1 D ERAS
  + 2       D GET^ADMGET
```

```
ST          S Y=2,X=30 D MOVE W *27,"[32m入院年月日 : ",KD
  + 1       D WHITE,SD^ADMINP,IND^ADMINP
B           G ASKSAV:Q=3,FILE
ASKSAV      D ERAS S PR="中断しましたが、それまでのデータをしまいこみま
            すか ? (Y/N)",S="N",X=0,Y=22,Z=1,PM="I R=""Y""!(R=""N"")"

  + 1       D ^READ G A:Q=2,EXIT:Q=3 I R'="Y" G EXIT
FILE        D SAVD^ADMFILE G EXIT
CLRLINE5    W *27,"[5;0H",*27,"[0J" Q
WHITE       W *27,"[37m" Q
BTMOFF      W *27,"[>1h" Q
ERAS        W *27,"[22;0H",$J(" ",79) Q
MOVE        W *27,"[",Y,";",X,"H" Q
EXIT        D KIL^ADMINP Q


ADMP5       ;入院台帳プログラム  ３１［印刷  その５］     ;昭和60年 1月1
            3日;林寺 忠
A           D ^ADMP0 G EXIT:Q=3
H           S %QTY=2,%DTY="CRT",%IOD=0,%DEF=0 D ^%MIOS ;  GET OUTPUT DE
            VICE
  + 1       I '$D(%IOD) G EXIT
  + 2       I "CRT^LP"'[%DTY W !,?5,"出力デヴァイス ???" G A
  + 3       I "CRT,LP"[%DTY U %IOD U 0
  + 4       S NAMEK="ア",NO=0 I %IOD=0 W *26,# G ORDER
  + 5       W !,"プリンターにて出力実行中です。" U %IOD
ORDER       D LIN1 I ORDER="N" G G
C           S N=-1 D NAM G ASKCON:NAMEK="" G C:NF S KD="A" D BASE^ADMGE
            T
GA          S KD=$N(^ADM(NAMEK,N,KD)) I KD<0 G C
  + 1       D GET^ADMGET,PRINT
  + 2       G GA
NAM         S NAMEK=$N(^ADM(NAMEK)) I NAMEK<0!(NAMEK'?.ｶ1" ".ｶ) S NAMEK
            ="" Q
  + 1       S NF=0,N=$N(^ADM(NAMEK,N)) S:N<0!(N'?.N) NF=1 Q
G           S KD=$N(^ADMADM(KD)) I KD<0 G ASKCON
  + 1       S X=^(KD) F Y=1:1:X S DAT=^ADMADM(KD,Y),NAMEK=$P(DAT,"^",1)
            ,N=$P(DAT,"^",2) I $D(^ADM(NAMEK,N,KD,1))#10 D BASE^ADMGE
            T,GET^ADMGET,PRINT
  + 2       G G
ASKCON      W:%IOD # U 0 S X=0,Y=22,Z=1,S="N",PR="更に"_$S(ORDER="N":"入
            院日",1:"アイウエオ")_"順印刷を続けますか ? (Y/N)"
  + 1       S PM="I R=""Y""!(R=""N"")" D ^READ G EXIT:Q=3
  + 2       I R="Y"!(Q=2) W *26,# D KIL K NAMEK G A
  + 3       G EXIT
PRINT       I '$D(PM) S NO=NO+1 D L5^ADMDX1 Q
PM          X PM I  S NO=NO+1 D L5^ADMDX1
  + 1       Q
LIN1        S KD="S60",NO=0 W:'%IOD *27,"[3;0H",*27,"[0J",*27,"[33m" W:
            %IOD ! W $S(ORDER="N":"入院日",1:"アイウエオ"),"順印刷 :
            " W:$D(PM) PM W ! D:'%IOD WHITE D LIN5^ADMDX1 Q
CLRLINE5    W *27,"[5;0H",*27,"[0J" Q
ERAS        W *27,"[22;0H",$J(" ",79) Q
WHITE       W *27,"[37m" Q
KIL         K %IOD,%DTY,AID,DF,NF,NO D KIL^ADMINP Q
EXIT        I $D(%IOD),%IOD U %IOD W ! C %IOD
  + 1       U 0 D KIL Q
```

```
ADMADM      ;入院台帳プログラム　３２［入院日ｲﾝﾃﾞｯｸｽ］　:昭和60年 1月1
            0日:林寺 忠
A           ;
  + 1       S B=0,NAMEK="ア",NO=0,N=-1 W *26,#
  + 2       W !,"入院日によるインデックス ^ADMADM 作成中です。"
B           K ^ADMADM
C           S N=-1 D NAM G EXIT:NAMEK="" G C:NF
  + 1       S NO=NO+1,KD="A"
D           D GET G C:DF D INDEX G D
NAM         S NAMEK=$N(^ADM(NAMEK)) I NAMEK<0!(NAMEK'?.カ1" ".カ) S NAMEK
            ="" Q
  + 1       S NF=0,N=$N(^ADM(NAMEK,N)) S:N<0!(N'?.N) NF=1 Q
GET         S DF=0,KD=$N(^ADM(NAMEK,N,KD)) S:KD<0 DF=1 Q
INDEX       S:'$D(^ADMADM) ^ADMADM=0 S ^ADMADM=^ADMADM+1
  + 1       S:'$D(^ADMADM(KD)) ^ADMADM(KD)=0 S A=^ADMADM(KD)+1,^(KD)=A
  + 2       S ^ADMADM(KD,A)=NAMEK_"^"_N,B=B+1 W:'(B-1#3) ! W ?B-1#3*25,
            $J(B,3),")",KD," ",NAMEK Q
KIL         K A,B,DF,N,NAMEK,NF,NO Q
EXIT        D KIL Q


%MIOS       ; GEF ; DSM UTILITIES ; I/O DEVICE SELECTOR
  + 1       ;  INPUT VARS ARE  :    '%DEF' - DEFAULT FOR DISPLAY: CAN BE
            UNDEFINED
  + 2       ;                       '%QTY' - TYPE OF QUERY SELECTOR
  + 3       ;  OUTPUT VARS ARE :    '%IOD' - THE DEVICE NUMBER SELECTED:
            CAN BE UNDEF.
  + 4       ;                       '%DTY' - THE DEVICE TYPE SELECTED
%INIT       ;S %ST=$V(44),%DT=$V(%ST+8)
  + 1       S %QRY="装置は ? " I '$D(%QTY) G %ASK
  + 2       I %QTY=1!(%QTY=2) S %QRY=$P("入力^出力","^",%QTY)_%QRY G %A
            SK
  + 3       K %QTY
  + 4       ;
%ASK        W !,%QRY W:$D(%DEF) " < ",%DEF
  + 1       R " > ",%X I %X'="",%X'="^" G %CK
  + 2       I '$D(%DEF)!(%X="^") K %IOD G %KL
  + 3       S (%IOD,%X)=%DEF G %CK
  + 4       ;
%CK         I %X="?" D %Q1 G %ASK
  + 1       I %X="0" S %X=$I G %OPEN
  + 2       I %X=1 G %OPEN
  + 3       I %X="CRT" S %X=0 G %OPEN
  + 4       I %X="LP" S %X=1 G %OPEN
  + 5       E  D %IV G %ASK
%OPEN       S %IOD=+%X C %IOD O %IOD E  W "   その装置は使用出来ません。
            " G %ASK
  + 1       I %IOD=0 S %DTY="CRT"
  + 2       I %IOD=1 S %DTY="LP"
  + 3       ;
%KL         K %QRY,%DEF,%QTY,%X,%ZA Q
%Q1         W !!,?5," 装置の番号または、その略称を入れて下さい。",!
  + 1       W !,?16,"装置の番号は次のようなものがあります。"
  + 2       W !,?16,"0       ＣＲＴディスプレイ"
  + 3       W !,?16,"1       プリンター"
  + 4       W !,?16,"CRT     ＣＲＴディスプレイ"
  + 5       W !,?16,"LP      プリンター"
  + 6       W !,?16,"<CR>    Ｄｅｆａｕｌｔ（予め用意された装置）"
  + 7       W !,?16,"^       装置の選択はしないで実行を終了する。",! Q
```

Set GNOSIS=MUMPS+Prolog

○Donald A. Smith, Toshiaki Suzuki,
Tatsuhiro Uchida, and Ichiro Wakai

MUMPS System Laboratory
MUG Bldg. 39 Daikan-cho, Higashi-ku, Nagoya 461

ABSTRACT

At the MUMPS System Laboratory, we have developed and are at the stage of proving versatility of, and standardization of GNOSIS language which is Prolog in MUMPS. Originally two versions were implemented, and the standard specification of the Language is strongly desired. Presently the documentation of the Language would be the one being compiled by Uchida, who authored a paper on the implementation of the Language in this proceedings. It has been proven that a Prolog database of rules and facts can be stored in a MUMPS global database so that the latter language can have access to the inferential reasoning ability of the former. A MUMPS-Prolog marriage would further provide a friendly knowledge management for Prolog, as well as providing in MUMPS capabilities which a pure, logic only Prolog lacks --data preparation, setting and killing of data, powerful and easy to use input/output facilities and iterative execution.

INTRODUCTION

Prolog is fundamentally different from common programming languages like MUMPS, BASIC, COBOL, FORTRAN, PASCAL, and even LISP which is another popular artificial intelligence programming language. While MUMPS and others are "procedural" languages, Prolog is a "declarative" language. Learning MUMPS involves mastering the workings of the MUMPS commands, functions, and operators. Understanding and debugging a large MUMPS package is as we all know, a big chore. It becomes increasingly difficult to picture in our minds the workings and dependencies of the whole package. Even if we divide the program into simpler self-contained modules, conventional programming languages suffer from this inherent weakness. Just as humans have troubles by convincing themselves that their programs are correct, so would automatic programming systems have difficulty in verifying the semantics of conventional programs. There is the hope that Prolog will remedy these weaknesses.

A Prolog program in contrast is more like a mathematical proof: one defines facts (assumptions) and relations on the one hand, and rules on the other hand. Although most technical writing on Prolog is quite difficult, the basic ideas of Prolog are really quite simple. Indeed, someone with no knowledge of computers may very well find Prolog to be simpler than MUMPS and the other languages.

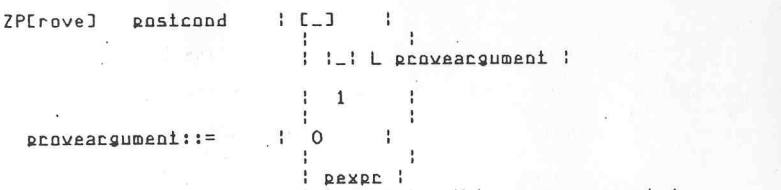
BACKGROUND

T. Uchida, one of the MUMPS development collaborators at the MUMPS System Laboratory, implemented some functionalities of Prolog on his U1 pre-compiled MUMPS during the summer of 1984 and found that the power of MUMPS would become more than doubled by embedding the Prolog syntax. Students at the Meijo University have been exposed to this experimental marriage of languages to excercise programming in logic.

Japanese extended standard MUMPS implementation on Pascal compiler, with assembler subroutines of frequent use, was in use at the MUMPS System Laboratory by late 1984, which was easy to add new functions, such as every READ acting in editor mode. Prolog syntax and semantics were added on this version of MUMPS taking some six months as extension of the Japanese standard MUMPS. Since extensions of MUMPS needed to have "Z's", specifications of Prolog was rather complex, i.g. PROVE had to be specified as ZP(rove) (fig. 1). The syntax extension for Prolog integration was specified on a 12 paged document. When GNOSIS was suggested to imply MUMPS which integrates Prolog, the specification was modified; Uchida has completed its Japanese document.

The main objective of the GNOSIS specifications has been 1) to reduce programming efforts and program spaces to a few tenth of the current MUMPS environment, 2) to speed up execution by bringing into MUMPS the logic programming t ools, 3) to secure transferability of the Prolog knowledge database in the MUMPS global files as if in DNA transmission, as has MUMPS standardization maintained the transferability of MUMPS and its database.

fig. 1. Example of Prolog syntax description to include in MUMPS



1. The empty argument list form attempts another match of a previously entered pexpr proof. If there is no valid proof in progress the empty argument list form has no effect. Otherwise, it forces backtracking of the resolution proof procedure and a search for another match. If no further match is available, $TEST is set to 0 and the command ends. If on the other hand, a further match is available, then $TEST is set to 1 and any Prolog variables appearing in the pexpr will be re-SET to reflect the new variable bindings resulting from unification.

2. If the value of proveargument is the single character "1", then exhaustive Prolog trace mode is activated: variable bindings are displayed and success and failure of unifications are displayed.

3. If the value of proveargument is the single character "0", then exhastive Prolog trace mode is turned off.

4. When the argument to ZPROVE is an expression other than "1" or "0", a call is made to the Prolog interpreter, which expects the pexpr to conform to the snytax described below in the section on Prolog Language Syntax. If the proof fails, $TEST is set to "0". If the proof succeeds, $TEST is set to "1" and MUMPS local symbol table variables corresponding to Prolog variables, if any, in the pexpr are SET to reflect the new variable bindings resulting from unifcation.

DESIGN CONSIDERATIONS AND SYSTEM DESCRIPTION

Our original documentation for the syntax and semantics of Prolog implementation as extension of Japanese standard MUMPS is obtainable from the MUMPS System Laboratory. We would, however, suggest potential GNOSIS users to read the Japanese documentation by Uchida ("GNOSIS=MUMPS+Prolog") as purest possible Prolog functionalities are embedded in ANS standard MUMPS, without "Z's". In the current paper of verification of GNOSIS ideas, we still use the MUMPS extension syntax. Since it is out of place to describe Prolog language here, just those features special to GNOSIS will be described.

1) ZC(onsult) initializes Prolog when written in MUMPS without argument, and clears it of any definitions of facts and rules. If GNOSIS is entered from the beginning Prolog is automatically initialized. When the argument to ZC(onsult) is a (list of) global variable name(s), each global variable should be defined, else an error is flagged. Every global node containing data (i.e. every node for which $DATA(node) euals 11 or 1), is pre-compiled and made known to the Prolog knowledge base of facts and rules. An error message and the offending string is displayed in case of a syntax error, e.g., a missing ' (' .

2) In global references of the form: ^global (s1,s2...,sn)=value ,
the first subscript (s1) is NOT pre-compiled or otherwise used by Prolog. Its only effect is to determine the sorting sequence of Prolog rules and facts, since Prolog searches its database in the same order as the nodes appear in the MUMPS database. The rest of the subscripts (s2...,sn) in combination with the global variable name comprise the head of a Prolog rule. The value to the right can be null to indicate truth (a fact or a rule true by default), or the value can be a list of Prolog predicates that must be true for the rule to secceed. The scope of Prolog predicate variables extends through one head and tail pair. Comments can appear in the value by preceding them with a semi-colon in place of where a Prolog predicate would normally appear.

3) pexpr ::= expr
From MUMPS' standpoint, the syntax of pexpr is merely any expression. However, the embedded Prolog language command ZCONSULT expects expressions to conform to the format of Prolog.

4) Predicates and functors appearing in structures (lists, etc.) and in rules are introduced by the character "#". Argument lists appear in parentheses.

5) Structures can be embedded to an arbitrary depth. For example:
#PRED((1,(2,0),#PRED(2)),*VARIABLE)

However, ",", "(", ")", "(", "*", "#" and ")" are reserved characters and hence cannot generally appear in Prolog atoms. Prolog predicate (logical) variables begin with the character "*". Lists begin with the character "(" and end with the character ")".

6) Indirect Predicates:

The syntax "#*X(*Y)" exemplifies the use of logical variables to generate indirect predicates. The predicate character "#" is followed immediately by a logical variable: the Prolog call generated gets the predicate name from the current value (if any) of the logical variable. Only the predicate name is gotten indirectly from the logical variable; the argument list is taken from the line on which the indirect predicate appears. If the logical variable is undefined or is not bound to a predicatre, the indirect predicate fails. Indirect predicates are useful to speed searches (e.g. path finding through graphs). Indirect predicates have the advantage of letting the pre-compiler (called by ZCONSULT) calculate addresses, instead of using the slower search algorithm used during unification.

7) The tail of a list can be referenced in unification by the character ":". The Prolog expression coming after the tail character ":" must be a predicate variable (starting with "*").

8) There is a 255 character limit to Prolog predicate (logical) variables names. However, common standard MUMPS implementations, such as UCD MUMPS, limit identifiers to 8 characters, so variables appearing in pexpr are truncated if greater than 8 characters. Predicate names are limited to 8 characters throughout GNOSIS.

9) MUMPS limits strings to 255 byte length. Therefore, values output in predicate variables cannot exceed this length. But internally the length of lists and structures is unlimited. When using the built-in predicate #$Display, "infinite" lists can be displayed: one exits with cntl-c.

10) Built-in Prolog Predicates:

The arithmetic operators can be used in two ways, depending on whether the third argument is a free variable or not. The first two arguments must always be defined, else the predicates fail. If the third argument is defined, the predicates succeed only if the result of the operation when applied to the first two arguments yields the third argument. If the third argument is undefined, then the operation is applied to the first two arguments and the result becomes the value of the third, newly bound argument variable.

#$+(a,b,c) Addition

#$-(a,b,c) Subtraction

#$*(a,b,c) Integer multiplication

#$/(a,b,c) Integer division

#$#(a,b,c) Integer modulus

#$CALL(a) True if argument can be proven; the argument can be passed as a variable. Unlike indirect predicates (above), #$CALL uses the argument list provided by the variable or other structure in 'a'.

#$VAR(a) Returns true only if argument is an unbound variable.

#$=(a,b) String or integer comparison: true if first argument equals second argument. Both arguments must be defined (literal or bound variable); otherwise the predicate fails.

#$'=(a,b) String or integer comparison: true if first argument does not equals second argument. Both arguments must be defined (literal or bound variable); otherwise the predicate fails.

#$<(a,b) String or integer comparison: true if first argument sorts before second in ASCII order (if string argument), or if first argument is numerically less than second argument (if integer argument). Both argument must be defined (literal or bound variable); otherwise the predicate fails.

#$>(a,b) String or integer comparison: true if first argument sorts after second in ASCII order (if string argument), or if first argument is numerically greater than second argument (if integer argument). Both arguments must be defined (literal or bound variable); otherwise the predicate fails.

#$False Always forces failure. Useful to force looping.

#$Display(a,b,.....)

Displays value of arguments. Writes CR LF when argumentless. During display control-s stops output, control-c aborts the current proof.

#$! Implements Prolog cut operation to "prune" the search tree.

STATUS REPORT

The current prototype GNOSIS was implemented on PC-9801 with 256Kb RAM. Because of memory constraints the global buffers were few and disk access was needed

whenever the global buffers failed to accommodate all the globals to be ZConsulted.

This means that the more the global buffers are available the better GNOSIS performs, but at the same time GNOSIS permits automatic disk access whenever necessary. GNOSIS may be ran on personal computers with some sacrifice of time as the amount of intelligence in the global increases.

1) TEST: Simple globals (^MEMBER, ^APPEND) containing Prolog rules were made, and a sample program was written for proving and writing the list and element while proof succeeded (fig. 2).

```
D ^%MG
Enter global name: ^MEMBER

^MEMBER(10,"*X","[*X:*Y]")
^MEMBER(20,"*X","[*Y:*Z]")    #MEMBER(*X,*Z)

Enter global name: ^APPEND

^APPEND(10,"[]","*X","*X")
^APPEND(20,"[*V:*Y1]","*Y2","[*V:*Y]")    #APPEND(*Y1,*Y2,*Y)


>P
SAMPLE      ;DAS,,;6/15/85;Sample Prolog programs
            ZC  ;Initialize Prolog
            ZC ^MEMBER,^APPEND ;pre-compile Prolog rules contained in globals
            ZP "#APPEND([A,B,C],[D,E],*LIST),#MEMBER(*ELEMENT,*LIST)"
LOOP        I  W LIST,?30,ELEMENT,! ZP  G LOOP ;WHILE proof succeeds, show bind
ngs
END         ;

>D SAMPLE
[A,B,C,D,E]                 A
[A,B,C,D,E]                 B
[A,B,C,D,E]                 C
[A,B,C,D,E]                 D
[A,B,C,D,E]                 E
```

fig. 2. Initializing Prolog, pre-compiling globals, proving, and writing bindings.

2) FAMILY TREE: A family tree example of Tokugawa Shogun was implemented to global data usinga simple PARENT-CHILD relationship. Using the Prolog interaction, such conversations as

-GRANDPARENT(*X,*Y):-PARENT(*X,*Z),PARENT(*Z,*Y)

-?GRANDPARENT(６代将軍家宣,*X)

*X= ３代将軍家光

was possible. In reality, the fact global lacked the essential knowledge of the mothers, sisters, and even brothers to test the relations as "who are uncles of 9th Shogun Ieshige?" or "who are cousins of 6th Shogun Ienobu?".

3) FLEIGHT CONNECTIONS BETWEEN TWO CITIES: An artificial fleight time table was implemented in global database. Questions were given in the form of two airports (e.g. Kagoshima, Narita). Non-stop fleights were first written with fleight # and time, then one-stop, two-stop, three-stop connections were written, most of them

being geographical detours. Fleight time tables from prulal air-lines, or updating the time tables showed no difficulty. However, the minute by minute updating of the reservations, the most dynamic part of the passenger scheduling, was left unsolved.

4) SENTENCE CHECKING AND GENERATION: 21 globals containing the rules of sentence elements were created for prototype of sentence checking and generation. The GNOSIS program used was as following;

```
SENT    ;DAS,,SENTENCE;6/26/85:SENTENCES
        I 0
        ; Pre-compilation
        ZC  ZC ^SENTENCE,^NOUNPHRA,^PREDICAT,^IVERBPHRA,^TVERBPHRA,^ADV,^DETERMIN
        ZC ^NOUN,^TVERB,^IVERB,^RELCLAUS,^RW,^ADJVPHRA,^ADV2,^ADJ,^PREP,^PREPOS
        ZC ^COMPOUND,^INDPRONOUN,^PRONOUN,^COPULATIVE
        Q
GEN     ; Sentence Generation
        ZP "#SENTENCE(*SENT,0)"     F I=1:1 W I,?10,SENT,! ZP
TEST    ; Sentence Checking
        S SENT=""
TL      R !,"Enter sentence to test: " S SENT=$ZE(SENT) Q:SENT=""
        ZP "#SENTENCE("_SENT_",())" W !
        I  W "Yes, it's a sentence." G TL
        W "No, it's not a sentence." G TL
```

The compilation time for the SENT subroutine of the above program, owing to the 21 globals' relatively small size, was 10 seconds, as shown below.

```
>W $H,! D SENT W $H,!
52778,51516
52778,51526
```

The GEN subroutine of the above program ran by "D GEN" generated series of sentence like lines until cntl-c terminated the generation.

```
>D GEN
1       (a,man,smiled)
2       (a,man,smiled,and,smiled)
3       (a,man,smiled,and,smiled,on,a,man)
```

The TEST subroutine of the above program worked as shown below.

```
Enter sentence to test: [the,cat,that,sat,on,the,mat,ate,the,rat]
Yes, it's a sentence.
Enter sentence to test: [the,cat,that,sat,on,the,mat,ate,rat]
No, it's not a sentence.
Enter sentence to test: [the,extremely,fat,man,ate,the,cat,and,sang,a,song]
Yes, it's a sentence.
Enter sentence to test: [the,grasshopper,on,the,computer,smiled,at,me]
Yes, it's a sentence.
Enter sentence to test: [the,grasshopper,on,the,computer,smiled,at,I]
No, it's not a sentence.
Enter sentence to test:
```

Updating of new words (adj, adv, noun, tr.v., itr.v., prep., pron. ind. pron) was successfully done using MUMPS routine "SEADD".

```
>D ^SEADD

Enter type of new word (Adj, aDv, Noun, Tverb, Intverb, Preposition,pRonoun,indi
reCt pronoun) I
Enter new IVERB: stood

stood is already registered.
Enter new IVERB: slept
Enter new IVERB:
Enter type of new word (Adj, aDv, Noun, Tverb, Intverb, Preposition,pRonoun,indi
reCt pronoun)
```

Some of the structures of the 21 globals to be pre-compiled at "D ^SENT" are shown as the space allows.

```
Enter global name: ^SENTENCE

^SENTENCE(10,"*S0","*S")    #NOUNPHRASE(*S0,*S1),#PREDICATE(*S1,*S2),#COMPOUND(*
S2,*S)
^SENTENCE(30,"*S0","*S")    #NOUNPHRASE(*S0,*S1),#COPULATIVE(*S1,*S2),#NOUNPHRASE
(*S2,*S)
^SENTENCE(40,"*S0","*S")    #NOUNPHRASE(*S0,*S1),#COPULATIVE(*S1,*S2),#ADJVPHRA(*
S2,*S)

Enter global name: ^PREDICATE

^PREDICAT(10,"*S0","*S")    #IVERBPHRASE(*S0,*S1),#PREPP(*S1,*S)
^PREDICAT(20,"*S0","*S")    #TVERBPHRASE(*S0,*S1),#NOUNPHRASE(*S1,*S)

Enter global name: ^NOUNPHRASE

^NOUNPHRA(10,"*S0","*S")    #DETERMINER(*S0,*S1),#ADJVPHRASE(*S1,*S2),#NOUN(*S2,*
S3),#PREPP(*S3,*S)
^NOUNPHRA(20,"*S0","*S")    #DETERMINER(*S0,*S1),#ADJVPHRASE(*S1,*S2),#NOUN(*S2,*
S3),#RELCLAUSE(*S3,*S)
^NOUNPHRA(30,"*S0","*S")    #PRONOUN(*S0,*S)

Enter global name: ^IVERBPHRASE

^IVERBPHR(10,"*S0","*S")    #ADV(*S0,*S1),#IVERB(*S1,*S2),#ADV(*S2,*S)

Enter global name: ^TVERBPHRASE

^TVERBPHR(10,"*S0","*S")    #ADV(*S0,*S1),#TVERB(*S1,*S2),#ADV(*S2,*S)

Enter global name: ^ADJVPHRASE

^ADJVPHRA(10,"*S","*S")
^ADJVPHRA(20,"*S0","*S")    #ADV2(*S0,*S1),#ADJ(*S1,*S)

Enter global name: ^NOUN

^NOUN      250
^NOUN(10,"[man:*S]","*S")
^NOUN(20,"[apple:*S]","*S")
^NOUN(30,"[computer:*S]","*S")
^NOUN(40,"[keyboard:*S]","*S")
^NOUN(50,"[raisin:*S]","*S")
^NOUN(60,"[diskette:*S]","*S")
^NOUN(70,"[house:*S]","*S")
^NOUN(80,"[car:*S]","*S")
^NOUN(90,"[moon:*S]","*S")
^NOUN(100,"[peacock:*S]","*S")
^NOUN(110,"[frog:*S]","*S")
^NOUN(120,"[rattlesnake:*S]","*S")
^NOUN(130,"[printer:*S]","*S")
^NOUN(140,"[book:*S]","*S")
^NOUN(150,"[nose:*S]","*S")
^NOUN(160,"[ear wax:*S]","*S")
^NOUN(170,"[glasses:*S]","*S")
^NOUN(180,"[song:*S]","*S")
^NOUN(190,"[gas:*S]","*S")
^NOUN(200,"[cat:*S]","*S")
^NOUN(210,"[grasshopper:*S]","*S")
^NOUN(220,"[rat:*S]","*S")
^NOUN(230,"[mat:*S]","*S")
^NOUN(240,"[hat:*S]","*S")
^NOUN(250,"[bat:*S]","*S")
```

LESSONS LEARNED

During prototype implementation of MUMPS+Prolog, various degrees of integration were possible. At the minimum, the addition of a ZPROVE command to MUMPS, then the ability to pass variables between Prolog and MUMPS: the values returned by Prolog's unification in response to one's queries should come out as MUMPS variables. Further provision for the addition of lists as a MUMPS datatype and list functions as extension of standard MUMPS made us decide calling this marriage GNOSIS, since MUMPS codes would include non-Z tagged new functions. It went too far to the stage that one could allow the use of MUMPS expressions, variables, operators and functions in Prolog rules. While quite powerful, many such extensions of Prolog beyond its pure form are open to the criticism of being too impure, thus destroying the logical rigor and abstract beauty of logic. This was the stage when "standardization" of GNOSIS was needed between the two prototype versions. The convergent result of the specifications of GNOSIS after repeated discussions would be written in Japanese by Uchida (1).

It has been our hope that by limiting programming to pure logic, programmer productivity would be increased and, more importantly, computers would be able to do the programming themselves. In the early 70's, computer scientists advised against the (over-) use of the GOTO statement. Now some advise against the use of even the SET, FOR and DO statements!

Since Prolog is based so closely on logic and since logic is so well understood and easily verified, most of the clumsy works as implied in the advices for the procedural language programmers would be able to be avoided by providing the Prolog capabilities in MUMPS itself.

Further hope for Prolog as a fifth generation language relies on the fact that the matching operations Prolog uses can often be done in parallel, which is one component of the ICOT Fifth Generation Project in Tokyo. By doing the matching in hardware in parallel, one moves beyond present Prologs, which do unification in software, and one move beyond the von Neumann architectural limitation of a single instruction stream.

K. O'Kane (2) of the University of Tennessee has developed a rather extended "impure" version of Prolog in MUMPS, which, however, lacks lists and functors. GNOSIS has lists and functors, in contrast, and is in the stage to allow MUMPS expressions to be freely used in Prolog rules, currently in this prototype version letting all data be shared through top level variables between MUMPS and Prolog.

Our belief at the MUMPS System Laboratory is that a Prolog database could be stored in a MUMPS global: a MUMPS-Prolog marriage would provide a friendly knowledge management environment for Prolog, as well as providing in MUMPS capabilities which a pure Prolog lacks; e.g. input-output, data preparation, setting and killing of data, iterative execution. The Prolog rules and facts can reside neatly in MUMPS global or local variables. The values returned by unification's binding of Prolog

logical variables can (at the top level of the proof) enter MUMPS variables. GNOSIS would furthermore provide many opportunities for extending Prolog beyond its pure form, thus needing standardization effort among the GNOSIS users.

Prolog database can be static as in the case of the family tree of Tokugawa Shoguns, and once it is completed there would be no need for updating. There are many applications where a Prolog program would be cumbersome, slow or inscrutable, as in the case of month after month updated fleight time tables, and minute by minute changing reservations. MUMPS global database has its dynamic real time multi-terminal updating facilities, providing the way to transform the data after standard checking into meaningful Prolog database.

Declarative languages like Prolog have their rightful place in the scheme of things, but they will never replace procedural languages altogether. Furthermore, in a well-integrated marriage of MUMPS and Prolog into GNOSIS, division of labor between declarative and procedural languages will allow both partners to benefit from the other: each will remedy the other's weakness.

Lastly, the fully demonstrated transferability of global database of MUMPS would provide the facility for accessibility and portability of updated knowledge databases, which otherwise would suffer from aging and fossilization. Such maintainability of knowledge database should be assured for the users of GNOSIS through standardization.

MUMPS users have dreamed what their future programming style would be. MDC has some 30 wish tables with priorities beside many back logs, each of which has to go through precise parliamentary precessing. MUMPS should not again ramify into dialects. GNOSIS, a branch of MUMPS evolution, is hoped to have the way to join the main stem of evolution.

ACKNOWLEDGMENT

The authors' grateful thanks are due to Henry G. Heffernan, S.J., Past President of SCAMC and Past Chairman of MUMPS Users' Group of North America, for his continuous guidance and encouragement.

REFERENCES

(1) Uchida, T.: Implementation of MUMPS in which Prolog is integrated, Proceedings of the 12th Meeting of MUG-Japan, Aug. 1985

(2) O'Kane, K.: An Expert Systems and Relational Data Management Facility for MUMPS, Univ. of Tennessee, Dec. 14, 1984

SP−MUMPS

小林　勝、○久江　正、　米田　研、　阪倉　明

住友電気工業株式会社ME開発室
〒554 大阪市此花区島屋1丁目1番3号

SP−MUMPSは、パソコン用の一般的なオペレーティングシステムであるMS−DOSの環境下で稼動し、MUMPSをパソコンの世界に持ち込んだものである。SP−MUMPSは、ANSI標準に準じ、MS−DOSとのインタフェース、日本語フルサポートなど機能拡充がなされている。しかも、プリコンパイル方式を始め処理速度向上のための種々の技術により、高パフォーマンスのシステムを提供する。

1. 動作環境

SP−MUMPSは、パソコン用のシングルユーザ／シングルタスクのMUMPSである。現在、PC9800，N−5200，IBM5550シリーズで使用可能である。以下に、これらのパソコンでの動作環境を示す。

OS	：	MS−DOS（Version　2.××）
メモリ容量	：	384KB以上
外部記憶装置	：	フロッピーディスク2台 又は、フロッピーディスク＋固定ディスク
端末装置	：	CRT キーボード プリンタ（セントロニクスインタフェース） RS−232Cインタフェース

2. SP−MUMPSの構成と機能

SP−MUMPSは、図1に示す4つの基本構成から成り、MS−DOSの環境下で稼動する。以下この構成要素に従って、SP−MUMPSの機能を説明する。

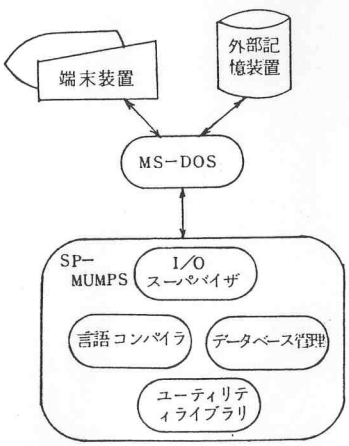


図1.　SP−MUMPSの構成

(1) 言語コンパイラ

言語処理系は言語コンパイラと中間言語（P−code）インタプリタより構成される。プログラムはセーブコマンド実行時にMUMPSコードのシンタックスチェック後、実行形式である中間コードに翻訳してストアされる。
これにより、実行速度の高速化を図るとと

もに、インタプリタ言語の利点であるデバッグの容易性はそのまま生かされている。

ＳＰ－ＭＵＭＰＳの言語仕様は、１９８４年ＡＮＳＩ標準（Ｘ１１．１）に準拠するとともに、後述する日本語処理機能などの拡張を行なっている。表１に、SP-MUMPSの言語仕様を示す。

表1.　ＳＰ－ＭＵＭＰＳ言語仕様

項　　目	仕　　様
拡張コマンドおよび関数	ZGO, ZINSERT, ZLOAD, ZPRINT, ZREMOVE, ZSAVE, ZWRITE $ZBOOLEAN, $ZCRC, $ZDATE, $ZHEX, $ZNEXT $ZORDER, $ZUCI, $ZVERIFY, $ZWIDTH
ファイル構造	Multiway B-tree（キーコンプレッション）
文字列型添字	可（英数・記号・漢字・カナ）最大：２５５バイト
パフォーマンスエンハンスメント	プリコンパイル ダイナミックディスクキャッシング
通信方式	RS-232Cインターフェース
ＯＳ	MS-DOS（Version2.××）
その他	日本語プログラミングフルサポート upper/lower case characterフルサポート ローカル変数は２０４４バイトまで可 グローバル添字は２５５バイトまで可 負のグローバル添字フルサポート 包括的なユーティリティパッケージ （基本ユーテリティ約２０本，拡張ユーテリティ約５０本） セキュリティ　Log-on エラートラップ EGR-９８パッケージ使用によるグラフィック処理可 ＶＪＥ-α使用による連分節日本語入力可

(2)　データベース管理

ＳＰ－ＭＵＭＰＳのデータベースは、ＭＳ－ＤＯＳ上の最大４つの独立したファイルから構成される。このうち１つは、基本データベースと呼ばれ常に存在しなければならない。残り３つは、拡張データベースと呼ばれ、必要に応じて基本データベースにマウント／デマウントして使用することができる。このため、フロッピーディスク２台の構成でも、片方を拡張データベースとして業務単位のフロッピーディスケットを交換して使用することにより、実用的なシステムを作れる。

ファイル構造は、Ｍｕｌｔｉｗａｙ－Ｂｔｒｅｅ構造をとるとともにキー圧縮を行なっている。また、グローバルデータのデータ長は、変数名・添字を含めて２５５バイトであり、添字には負の数値を指定することもできる。

(3) Ｉ／Ｏスーパバイザ

Ｉ／Ｏスーパバイザは、システム周辺装置・端末装置の入出力およびエラー検出等を行なう。特に、ディスクとの入出力については、キャッシュバッファを用いることによりスループットを向上させている。

また、ＭＳ－ＤＯＳのファイルとのインターフェースのために、ＨＦＳ（ホストファイルサーバ）と呼ばれる仮想デバイスが用意されている。これによりＭＳ－ＤＯＳのファイルを介して、ＳＰ－ＭＵＭＰＳで作成されたデータを他言語によるパッケージルーチン（たとえば、グラフィックパッケージ等）で処理したり、逆に他言語で作成されたデータをＳＰ－ＭＵＭＰＳに読込んだりすることが可能である。

さらにグラフィックユーティリティであるＥＧＲ－９８のコマンドを出力し、ＳＰ－ＭＵＭＰＳでグラフィック表示することもできる。

(4) ユーティリティライブラリ

ＳＰ－ＭＵＭＰＳは、プログラムの開発支援（エディタ，ドキュメンテーションツール等）およびシステム運用支援（システムジェネレーション，データベースの整合性チェック等）のための豊富なユーティリティを持つ。なお、これらのユーティリティメッセージは全て日本語化されている。

3. ＳＰ－ＭＵＭＰＳの日本語処理

ＳＰ－ＭＵＭＰＳは、漢字を含む日本語処理を以下のようにサポートする。

(1) コード体系

ＳＰ－ＭＵＭＰＳの内部処理コードとしては、パソコン等で日本語処理コードの標準になりつつあるシフトＪＩＳコードを採用している。外部端末からの入出力には、シフトＪＩＳコードはもちろんＪＩＳコード，ＤＥＣ漢字コードが使用でき、それぞれの端末について漢字シフトイン／シフトアウトコードが設定できる。

(2) 日本語の使用範囲

グローバル／ローカル変数のデータのみならず、プログラムの名称、ラベル名、コメント文およびグローバル／ローカル変数の変数名・添字に漢字を含む日本語を使用することができる。また日本語入力は、ＭＳ－ＤＯＳの日本語入力機能を使用する他、ＶＪＥ－αを使用した連分節変換入力も可能である。

(3) 言語拡張

ＳＰ－ＭＵＭＰＳでサポートする全ての関数・演算子を日本語処理が可能なように拡張した。また、新たに漢字を含むデータの印字幅を計算する関数（＄ＺＷＩＤＴＨ）およびカナ・漢字を識別するパターンマッチ演算子（“カ”および全角文字“Ｔ”）を追加した。また、日本語を含む文字列のタブレーションも端末毎に全角／半角文字ピッチ比を定義することにより、これを制御できる。

＄ＺＨ（＄Ａ（“漢”））→８ＡＢＦ　　＄Ｃ（＃８ＡＢＦ）→“漢”

＄ＺＨ（＄Ａ（“Ａ”））→４１　　＄Ｃ（＃４１）→“Ａ”

＄ＺＷ（“漢Ａ”，４）→３　　＄ＺＷ（“漢Ａ”，３）→２．５

4. ＳＰ－ＭＵＭＰＳの処理速度

ＰＣ－９８０１ＦによるＳＰ－ＭＵＭＰＳとＮ８８－ＢＡＳＩＣ，ＤＥＣ社製ＰＤＰ１１／３４によるＤＳＭ－１１（Ｖｅｒｓｉｏｎ　２．０）のシングルユーザにおけるベンチマークテストを社内で実施した。テスト項目は、ＭＵＭＰＳの代表的なコマンドとし、またＢＡＳＩＣについてはこれと同等のコマンドを使用した。この結果を表２に示す。表の各数値は、ＳＰ－ＭＵＭＰＳのデータベースを固定ディスクに作成したときの処理時間を１とした場合の相対処理時間である。

各テスト項目で差はあるが、ＤＳＭ－１１と比して、メモリオペレーションでは平均的に４倍、ディスクオペレーションでは同等の性能を示している。また、Ｎ８８－ＢＡＳＩＣに比しても同等かそれ以上の処理速度となっている。

表２．　ＳＰ－ＭＵＭＰＳベンチマークテスト結果

分類	テスト項目	DSM-11 (V2.0)	N88-BASIC	SP-MUMPS	
		PDP 11／34	PC-9801F	フロッピーベース	ボンベース
ローカルデータ処理	'F I=1:1:50000	5.1	2.9	1.0	1
	F I=1:1:5000 S A=1	4.0	1.9	1.0	1
	F I=1:1:5000 S A="1234567890"	2.0	1.5	1.0	1
	F I=1:1:100 F J=1:1:50 S A(J)=J	1.5	0.6	1.0	1
	(F I=1:1:50 S A(1)=I) F I=1:1:100 F J=1:1:50 S A=$N(A(J))	6.6	-	1.0	1
	(F I=1:1:50 S A(1)=I) F I=1:1:100 F J=1:1:50 S A=$D(A(J))	1.3	-	1.0	1
	F I=1:1:100 F J=1:1:50 S A(J)="1234567890"	1.4	0.8	1.0	1
演算子処理	F I=1:1:5000 S A=I+1-1	4.8	1.7	1.0	1
	F I=1:1:5000 S A=I*1/1	2.0	0.7	1.0	1
	F I=1:1:5000 S A=I#100\10	4.8	-	1.0	1
	F I=1:1:5000 S A="12345"_"67890"	3.4	1.2	1.0	1
	F I=1:1:50 F J=1:1:100 S A=1)J	5.3	-	1.0	1
	(S X=1234567890,Y=X) F I=1:1:5000 S A=X=Y	6.4	2.0	1.0	1
	(S X=1234567890,Y="A") F I=1:1:5000 S A=X[Y	7.0	-	1.0	1
	(S X=01234567B9,Y="0123456789A") F I=1:1:5000 S A=X]Y	10.7	-	1.0	1
	(S X="858,08,05") F I=1:1:5000 S A=X?1A2N1","2N1","2N	8.9	-	1.0	1
関数処理	F I=1:1:5000 S A=$A("A")	4.1	2.1	1.0	1
	F I=1:1:5000 S A=$C("65")	2.8	1.7	1.0	1
	F I=1:1:5000 S A=$E("12345",2,3)	4.7	2.1	1.0	1
	F I=1:1:5000 S A=$F("1234567890","6")	4.4	1.8	1.0	1
	F I=1:1:5000 S A=$J("12345,7890",10,1)	0.6	-	1.0	1
	F I=1:1:5000 S A=$L("12345")	4.7	2.2	1.0	1
	F I=1:1:5000 S A=$P("1/2/3/4","/",2,3)	3.3	-	1.0	1
	F I=1:1:5000 S A=$T(A)	0.5	-	1.0	1
グローバルデータ処理	(S X="1234567890",L1=1000,Y=5 K 1A) F I=1:1:Y F J=1:1:L1 S ^A(J)=X	1.0	-	1.1	1
	(S X="1234567890",L1=1000,Y=5) F I=1:1:Y F J=1:1:L1 S A=^A(J)	1.0	-	1.4	1
	(S X="1234567890",L1=1000,Y=5) F I=1:1:Y F J=1:1:L1 S A=$N(^A(J))	0.8	-	1.5	1
	(S X="1234567890",L1=1000,Y=5) F I=1:1:Y F J=1:1:L1 S A=$D(^A(J))	0.9	-	1.8	1
	(S X="1234567890",L1=100,L2=10,Y=1 K 1A) F I=1:1:Y F J=1:1:L1 F K=1:1:L2 S ^A(J,K)=X	1.0	-	1.0	1
	(S X="1234567890",L1=100,L2=10,Y=1) F I=1:1:Y F J=1:1:L1 F K=1:1:L2 S A=^A(J,K)	1.0	-	1.3	1
	(S X="1234567890",L1=100,L2=10,Y=1) F I=1:1:Y F J=1:1:L1 F K=1:1:L2 S A=$N(^A(J,K))	1.0	-	1.4	1
	(S X="1234567890",L1=100,L2=10,Y=1) F I=1:1:Y F J=1:1:L1 F K=1:1:L2 S A=$D(^A(J,K))	1.0	-	1.2	1

SP－MUMPSディスベースによる処理時間を1とする相対処理時間で表示

5. おわりに

ＳＰ－ＭＵＭＰＳは、処理速度向上を目ざした種々の技術により、扱い易いデータベースシステムとして実用性の高いＭＵＭＰＳシステムをパソコン上で実用化することができた。著者等は、ＳＰ－ＭＵＭＰＳが多くのパソコンユーザの強い要望である簡易なデータベースの構築に貢献することを期待するとともに、今後さらに、ＬＡＮとの結合による分散データベースのサポート強化、ＺＣＡＬＬファンクション等による拡張性・発展性のあるシステムの開発を進めていく考えである。

＊ＭＳ－ＤＯＳは米国Ｍｉｃｒｏｓｏｆｔ社の登録商標である。
＊ＥＧＲ－９８はカノープス電子株式会社の製品である。
＊ＶＪＥ－αは、㈱バックス社の製品である。

ＮＥＣ９８０１とＳＨＡＲＰ　Ｉｎｋ　Ｊｅｔ　プリンターをつないで カラーグラフィックスをハードコピーする

○林寺　忠、和田行文、西角　淳、上坂邦夫
国立京都病院　小児科
京都市伏見区深草向畑町１－１（〒612）
℡　075-641-9161

ＵＣＤ日本語マンプスはカラーグラフィックを作成することが容易であるが、今回このグラフィックスをハードコピーするユーティリティを開発した。ソースプログラムはＣＰ／Ｍ８６アセンブラーにて作成し、NEC 9801のGRAMに表示されているグラフィックスをそのままハードコピーするのである。インクジェット・プリンターはSHARP MZ1P04を使用した。NEC とSHARPをつないだこともあって、640X400 のすべてをカラープリンターにハードコピーは出来ないという制限がある。画面の一部を除いて複写可能である。

カラーハードコピーの仕方に工夫を加え、次のような仕事を可能とした。
１．グラフィックスを上下左右にスクロールする。
２．画面に黒く表示されている部分を白く、白く表示されている部分を黒くコピー（白黒反転）する。
３．濃いコピーと薄いコピーの２つのオプションを用意する。
４．コピーの幅に制限があるためにコピーする範囲を移動させて選択する。

これとは別に、Ｂ型肝炎ワクチンフォローアップ台帳プログラムを日本語マンプスで作成した。これは今回別のセッションで発表する「入院台帳プログラム・パッケージ」と基本的に同じ構造をしたものである。Ｂ型肝炎フォローアップ台帳プログラムは、臨床検査データをファイルするが、その出力に検査値をグラフィック表示させるようにルーチンを拡張した。よく「統合化ソフト」とかいって、市販のソフトはワープロとグラフィックスとデータベースを「統合化」することが流行しているのであるが、日本語マンプスにはその概念は始めからない。つまり最初から「統合化」しているからである。項目数に制限があったり、グラフが出来ないという制限がない。従って、「統合化」の強調をする必要もない。

作成されたグラフィックスをインクジェットプリンターにそのままコピーするルーチンで患者さんの検査値を供覧します。

以下にはそのルーチンの使用法を詳述します。

ＵＣＤ　ｍｉｃｒｏＭＵＭＰＳ　Ｕｔｉｌｉｔｙ　Ｃｏｍｍａｎｄ　Ｃ．ＣＭＤ

Ｃ．ＣＭＤ

【内容】
画面に表示されたグラフィックスを
①縦横に自由にスクロールします。
②カラープリンターにハードコピーをします。

【前提条件】
６４０×４００の高解像度のＣＲＴが必要であります。
プリンターはＳＨＡＲＰ株式会社製のカラーインクジェットプリンター形名ＭＺ－１Ｐ０４が必要です。セントロニクスの端子でＰＣ９８０１Ｅ／Ｆ／Ｍと簡単につながりますので、特別なインターフェースを準備する必要はありません。またＮＥＣ９８０１の標準ＲＡＭのみで走りますので、ＲＡＭの増設は全く必要はありません。

【使用方法】
A＞C．CMD　としますと、
　黒地を白く［反転］複写？［Y／N］　Y

①YまたはyまたはＣＲを押しますと、カラーコピーは画面の黒く表示されている部分を白く、白い表示を黒く複写（コピー）します。
②Ｎまたはｎを押しますと、画面の表示と全く同じコピーを作成します。白い表示は白く、黒い表示は黒く複写されます。
③ＳＴＯＰキーを押しますと、ＣＰ／Ｍ－８６コマンドモードに戻ります。

次いで、
　複写は濃く［H］？　淡く［L］？　［H／L］　H
①ＨまたはｈまたはＣＲを押しますと、カラーコピーは濃いものが作られます。
②Ｌまたはｌを押しますと、カラーコピーは淡い（薄い）ものが作られます。
③ＳＴＯＰキーを押しますと、ＣＰ／Ｍ－８６コマンドモードに戻ります。

最後に、
　複写範囲の指定［←・→・ＣＲ］
と表示されます。
①←を押しますと、画面の可視範囲が左に移動します。このみえている範囲がカラーコピーされます。
②→を押しますと、画面の可視範囲が右に移動します。このみえている範囲がカラーコピーされます。
　＊注　プリンターの制限があるために、画面全体を同時にコピー出来ませんから、御承知下さい。
③５を押しますと、テキスト画面がクリアされます。もう一度５を押しますと、テキスト画面は回復して表示されます。
④４を押しますと、グラフィックス画面が左へスクロールされます。
⑤６を押しますと、グラフィックス画面が右へスクロールされます。
⑥８を押しますと、グラフィックス画面が上へスクロールされます。
⑦２を押しますと、グラフィックス画面が下へスクロールされます。
⑧↑を押しますと、グラフィックス画面が上へ細かくスクロールされます。
⑨↓を押しますと、グラフィックス画面が下へ細かくスクロールされます。
⑩ＳＴＯＰキーを押しますと、ＣＰ／Ｍ－８６コマンドモードに戻ります。
⑪最後に、ＣＲを押しますと、
　　複写実行中
と表示されてＳＨＡＲＰインク・ジェット・プリンターに複写を開始します。

　＊注　カラー・インクジェット・プリンターに複写（コピー）を開始した後ではＳＴＯＰキーを押しても複写を中断することなく最後まで複写を実行し続けますので、ご注意下さい。

研究用患者データの管理システムのマイクロマンプスへの変換

○本多正幸[*]　里村洋一[*]　北村嘉章[**]

[*] 千葉大病院医療情報部　千葉市亥鼻１－８－１
[**] 神戸大学医学部第二内科　神戸市中央区楠町７－５－２

ＤＥＣ社の標準マンプスにより開発された、医学研究用データの管理および解析を目的とした研究用患者データ管理システムをＣＰ／Ｍ－８６の下の蝶理ＭＵＭＰＳ（ＵＣＤ－ＭＵＭＰＳ　Ｖ３．０）によって書き換え、マイクロ版のシステムとして利用できるようにした。本報告では、その際の手順、問題および特徴などを述べ、さらに両システムの実行時間（処理時間）について若干の考察をする。

§１．はじめに

医療統計や診療データの解析の要求に伴い、千葉大病院においては８２年の秋より、医療データの編集、解析を目的とし、研究者別にデータ管理を行う、研究用患者データ管理システム（「ＣＵＩＰＤＳ（Chiba　University Integrated Patient　Data System）」を設計開発し、運用中である。このシステムの概要および特徴については、ＭＵＧ学術大会およびその他の関連学会において報告ずみである。このシステムは病棟および外来の端末より利用することができるが、業務（データ検索および検査データ報告等）用の端末を共用しているため、ＣＵＩＰＤＳ利用者は十分な時間をこのシステムに費すことができないでいる。また端末からのデータ入力を主体にしたシステムであるので、ファイル定義（データ項目の定義）およびデータ入力に少なからず時間を必要とするので、利用者から夜間の利用を切望する声が増えてきた。ＣＵＩＰＤＳは病院全体のシステムの中の一部のシステムであるので、コンピュータの稼動時間内で利用せざるをえない。そこで、最近個人レベルで普及しているマイコンへ移植して利用できるよう変換し、利用者の便を増すとともに、院外への普及を企てたいと考えている。本報告では、マイコン上でシステムを再構築するためのルーチンの転送および編集について述べ、さらにマイコン上で構築されたシステムの構成および特徴を述べるとともに処理スピードの変化について若干の考察を加える。

特にマイクロ版のシステムをキューピッド：ＣＵＰＩＤＳ（Chiba　Univ.Personal Investigation　Data System）と命名した。（図１．参照）

【　図　１　】

```
******  ＣＵＰＩＤＳ　管理メニュー  ******

 1)= ファイル定義と修正          ( FILE DEFINITION & CORRECTION)
 2)= データの入力                ( DATA ENTRY)
 3)= 欠測データの補充            ( DATA SUPPLEMENT)
 4)= 症例の削除                  ( CASE DELETION)
 5)= データの修正，削除          ( DATA CORRECTION & DELETION)
 6)= データの変換，編集          ( DATA EDIT. & CONVERSION)
 7)= 症例のリスト，計数          ( CASE LIST AND COUNT)
 8)= データの解析，集計          ( DATA ANALYSIS & COUNTING UP)
 9)= ***************
10)= ファイルの消去              ( FILE ERASURE)
11)= ファイルの編集，コピー      ( FILE EDIT. & COPY)
12)= ファイルの管理表            ( FILE MANAGEMENT TABLE)

どの業務をしますか (JOB)?>
```

§２．マイクロ版への変換

［ルーチン転送について］

ＤＥＣ社の標準マンプス（ＰＤＰ11／70、ＤＳＭ－11）により開発された、約６０本のルーチンより構成されているＣＵＩＰＤＳを、ＣＰ／Ｍ－８６下の蝶理ＭＵＭＰＳ（ＵＣＤ－ＭＵＭＰＳ　Ｖ３．０）のＴＡＬＫ－through Mode を利用し転送した。

［画面制御部分の変換について］

カーソル位置指定、画面消去、表示色指定等の方法はシステムにより異なってくるので、キューピッドを稼動する時、いちばん最初のルーチンで画面制御命令をローカル変数として作成しておき、Ｘecute 命令で利用するようにした。システムにより違いがあれば、この画面制御命令を作成するルーチンのみを変更すればよい。

例）％ＨＯＭ＝“＄Ｃ（２７，９１，１，５９，１，７２）”　：ホームポジション
　　％ＥＯＳ＝“＄Ｃ（２７，９１，４８，７４）”　：カーソル以降消去
　　％ＥＯＬ＝“＄Ｃ（２７，９１，４８，７５）　：カーソル以降その行のみ消去
　　％ＸＹ＝“＄Ｃ（２７，９１，＄Ｓ（＄Ｌ（％Ｙ）＝１：０……　：カーソル移動
　　％ＢＥＬ＝“＄Ｃ（７）”　　：ベル

このような画面制御命令の変更はほとんどすべてのＣＵＩＰＤＳのルーチンにおいて必要なため自動変換システムを作り、従来のルーチン名を指定するだけで必要な部分を次々に書き直した。

Ｗ　＄Ｃ（１０８）⟶　Ｗ@％ＨＯＭ　：ホームポジション
Ｗ　＄Ｃ（１０９）⟶　Ｗ@％ＥＯＳ　：カーソル以降消去
Ｗ　＄Ｃ（１１１）⟶　Ｗ@％ＥＯＬ　：カーソル以降その行のみ消去
Ｗ　＄Ｃ（７）　⟶　Ｗ@％ＢＥＬ　：ベル
Ｗ　＄Ｃ（１６，１１，３１＋yy，３１＋xx）
　　⟶　Ｓ　％Ｘ＝xx、％Ｙ＝yy　Ｗ　@％ＸＹ

［日本語表示について］

ＤＳＭでは漢字を用いていないがキューピッドでは必要な表現をすべて漢字に書き直すこととした。ここでも自動変換システムを作り、ルーチンの中より”　”で区切られた部分および；で始まるコメント行を順次ＣＲＴに表示し、それぞれ適当な日本語表示におきかえた。この時同時に辞書を作成し、一度変換した表現は、自動的におきかえることとした。

§３．キューピッドについて

①マイコン版システム、キューピッドはルーチンおよびシステムを含んだシステム用フロッピーとグロバールデータを書き込むためのデータ用フロッピーの２枚より構成されている。データ用フロッピーには、ユーザーより入力されるデータと統計関係のテーブル（ｔ－分布表とＦ－分布表）が含まれるが、初回のシステム起動時に統計関係テーブルを作成する方式を採っているので、予めデータ用フロッピーにテーブルを準備する必要がない。（（テーブル作成にあたっては、１０分足らずで作成できる。（メモリーは２５６ＫＢ））これによってグローバルディスクを新らしく追加作成したい時にも、この統計テーブルのグロバールの転送を必要としない。

②移植前のシステム（ＣＵＩＰＤＳ）では、病名関係テーブル（ＩＣＤコード、ＩＣＤ術式コード、ＳＮＯＰコード）をサポートしているが、キューピッドでは容量の関係で割愛している。

③その他移植前のシステムと異なる点はプロテクト機能である。現在１５名ほどの医師がモニターとして登録しているが、発行する際にパスワードを与え、パスワードを知らなければシステム内に入れないよう設計してある。また第三者が契約者に無断でコピーし使用するのを防ぐために、使用者名をシステム内に組みこみ、使用者名を変更するとシステムエラーとなるよう設計した。（図２．参照）

④またＣＵＩＰＤＳではデータベースより研究者の個人ファイルにデータをとり込む機能を有しているが、その部分に関してはキューピッドでは削除した。

【 図 ２ 】

```
****  ＣＵＰＩＤＳ  管理メニュー  ******

      These programs for controlling and analysis of the personal data file
is developed at Division of Medical Informatics Chiba University.
When you publish the results of your scientific studies ,which are assisted
by the system, Please make reference of our contribution and, if possible,
make a small comment on your paper, Like as follows

"The results of the studies were achieved by the assistance of

[ CUPIDS ] -- Chiba University Personal Investigation Data System --"

                製作    千葉大学医学部附属病院医療情報部
                        （〒２８０  千葉市亥鼻  １－８－１）

                里村  洋一    本多  正幸    北村  嘉章

      このプログラムは  千葉大学  の  医療情報部  さんとの契約により
    提供したものです。他の人が使用する時は上記の所にユーザー登録をして下さい。

 ＰＡＳＳ  ＷＯＲＤ ? >
```

§４．処理スピードについて

キューピッドではフロッピーディスクを使用する事としており、ルーチンのスワップおよびデータアクセス等にかなりの時間を要する。

入力、編集処理は移植前のＣＵＩＰＤＳとほとんど差がないが、統計解析のような計算ルーチンの実行では、ディスクへのアクセスタイムも加わって、結果が出力されるまでの待時間はかなり増えている。何種類かの統計ルーチンの処理時間（計算を始めてから結果が出力されるまでの時間）を比較すると次の様になる。

Ａ：　処理内容	Ｂ：　ＣＵＩＰＤＳ	Ｃ：　キューピッド	Ｃ／Ｂ
（１）	８秒	２分	１５
（２）	３秒	１分３０秒	３０
（３）	７秒	１分４０秒	１４
（４）	３秒	１分２０秒	２７
（５）	２５秒	３分５０秒	９
（６）	３７秒	６分２０秒	１０
（７）	１０秒	３分３０秒	２１

（Ｂ：ＣＵＩＰＤＳ…ＤＳＭ－１１，ＰＤＰ１１／７０）
（Ｃ：キューピド…ＰＣ９８０１，２５６ＫＢ）

（１）：１変量基本統計量の算出，計算に用いたデータ数＝１５５例
（２）：　〃　，　〃　＝７例
（３）：　〃　，　〃　＝１１６例
（４）：２変量基本統計量の算出，欠測値のため「計算に使えるデータなし」の表示までのデータサーチに要した時間
（５）：２変量基本統計量の算出，計算に用いたデータ数＝１１６例
（６）：単回帰分析，計算に用いたデータ数＝１１６例
（７）：等平均の検定（ｔ－検定，ウェルチの検定），計算に用いたデータ数＝（第一群＝１１６例，第二群＝３９例）（図３．参照…出力例）

……ただし、計算については同一ファイル上のデータを用い、そのファイルに登録されているデータ数は１６８例である。

【　図　３　】

```
***** 等平均の検定 ***** (85.08.23)

          FILE= DEMO
          ITEM= SURV TERM

              グループの 条件
グループ１                    グループ２

(A=1)                         (A=2)

  WHERE                         WHERE

  A=SEX                         A=SEX

NO. OF DATA= 116              NO. OF DATA= 39
MEAN=      267.155            MEAN=      326.923
VARIANCE=   60448.358         VARIANCE=  177290.283

            <<<検定結果 >>>

[EQUAL VARIANCE TEST]: 有意水準 .01 で有意
( BY F-TEST , DEGREE OF FREEDOM (38,115) )
                  <WITH ONE SIDED TEST>
< 参考 : 検定 に 用いた 統計量の値 =        2.933 >

[EQUAL MEAN TEST] : 有意水準 0.05で有意でない
( BY WELCH-TEST , DEGREE OF FREEDOM = 47 )
                  <WITH ONE SIDED TEST>
< 参考 : 検定 に 用いた 統計量の値 =       -0.840 >
```

ＤＳＭ－Ｖ３による業務スピード向上評価

○大槻陽一、古林榮次郎、野口　弘、寺村昌文

大阪府立羽曳野病院　情報企画室
〒５８３　羽曳野市はびきの３－７－１

「要約」
ＤＳＭ１１－Ｖ２．０ＪからＤＳＭ１１－Ｖ３Ｊへの移行による業務の処理時間を比較した。１０８の月次業務では、延べ時間が５，６３８分から４，７８８分と短縮した。（８５％）　スピードアップは各業務により、かなりの違いがみられた。これはバージョンアップのディスク・バッファ容量の増加、マップド・ルーチン、ローカル処理の高速化、ＩＯスループットの向上などの恩恵が異なるためと思われる。レセプトのワークファイル作成と出力は大幅に時間短縮された。（３６～７６％）

「本文」

羽曳野病院では、昭和６０年３月１８日にＤＳＭ１１－Ｖ２．０ＡＪからＶ３Ｊへバージョンアップを行った。３月２７日にはマップドルーチン化も行った。この前後での各業務の処理時間を比較出来たので報告する。

ハードウエアの構成を図１に示す。ＰＤＰ１１／７０の３台で構成されており、オンライン１は外来業務、オンライン２は病棟業務と検査を中心とした業務、中央情報は診療から少し離れた業務を載せている。オンライン１または２がダウンすると、中央情報へ切り換えることができるようになっている。各ＣＰＵのメモリ容量は、バージョンアップ前は７６８ＫＢであったが、現在５１２ＫＢを増設して１２８０ＫＢとなっている。

測定対象としたのは、月次処理１０８業務、外来医事窓口会計と外来医師オーダエントリである。月次処理については、Ｖ２．０ＡＪの安定期である昭和５９年１１月とＶ３Ｊへの移行後の４月の処理時間を比較した。（表１）　外来医事窓口会計は、昭和５９年９月から昭和６０年５月までの毎週の平均をグラフにした。（図１）
外来医師オーダエントリについては、処理時間が３秒を越えて問題になっていた前回処方の取込みについて適宜計測を行った。（図２）

月次処理の全体では、延べ５，６３８分が４，７８８分になった。（８５％）　しかし、各処理間ではスピードアップにかなりの差があった。オンライン中に処理したものは、競合するジョブの関係のため、データにバラツキが発生することは避けられない。このため、処理時間がみかけ上増加したものもある。最も効果があったのは、外来レセプト発行業務であった。特に、枚数が最も多く夜間業務となっていた社保・国保単独は、１枚当りの処理時間がワークファイル作成で２．９秒から１．３秒、発行が３．９秒から１．４秒となった。（３６％）　これにより、朝８～９、午後３～７、土曜午後などの時間帯を利用して、夜間業務から開放されることとなった。元々

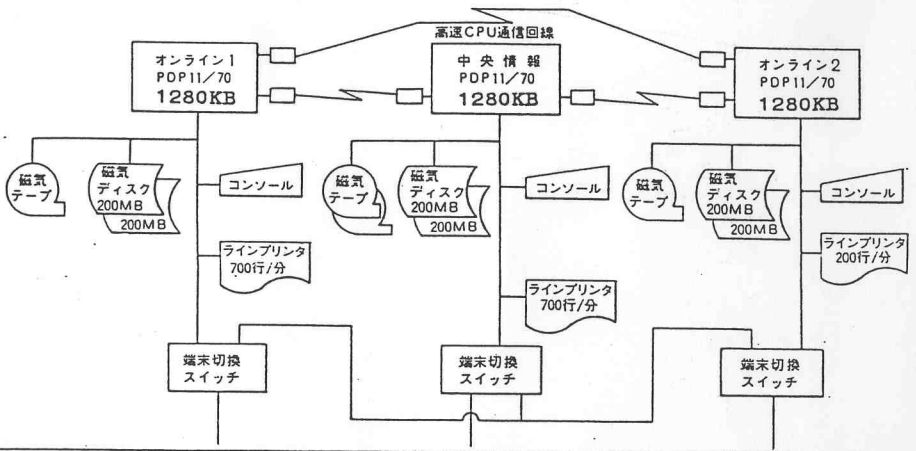
オンライン中に行っていた入院レセプトでも全用紙平均でワークファイル作成が１３．１秒／件から９．９秒／件（７６％）、発行が１０．０秒／件から６．６秒／件（６６％）となった。シングルジョブでも処理時間にほとんど変化の無いものもある。月次不要ファイル消去と病名から患者を検索するためのファイルを造る業務（インバーテッド　ファイル作成）である。これらの業務では、ディスク・アクセスのみであるため、処理時間はディスクのハードウエアにのみ依存しているものと思われる。またシングル・ジョブで明らかに処理時間が増えたものが一つあった。これはテーブル転送をＤＤＰ経由で行う業務である。Ｖ３ではＤＤＰがかなり改善され、高速になったと言われていた。また、ＤＤＰジョブをマルチで走らせる方がスループットが向上すると言われていた。これに合うように変更したところ、実際には遅くなっている。この理由は、ＤＰＰにバグがあったため、旧のＤＤＰハンドラに戻したためと、マルチでディスク・アクセスを行うとヘッドムービィングが大きくなるためと思われる。実際、新アプリケーション・プログラムではヘッドの移動音がかなり大きく聞える。しかたなく、次の月からは元のアプリケーションに戻した。

医事外来窓口会計での一患者当りの処理時間は、Ｖ３になってから明らかに短縮された。平均で７０秒から５０秒以下になっている。（７０％）　これは月次業務より大きく効果があらわれている。複数の端末が同じ処理を行うため、ディスク・バッファの効果が大きいことに起因しているためであろう。

当院のシステムで最も問題になっていたのが外来医師オーダ・エントリでの前回処方の取込み時間であった。この処理では、前回の処方を取込む時に各薬剤が使用中止になっている可能性があるため、薬剤マスターを見にいくなどのチェックも行うためディスク・アクセスの多い処理となっている。昭和６０年１月の初めには平均で５秒薬剤の多い場合で午前１１～１２時頃のピークには１０秒近くかかっていた。このため、まずはアプリケーションの改修をおこなった。グローバル・アクセス回数の少なくなるようなコーディングに変更すると同時に外部ルーチンをコールしていた所を内部モジュールに取込んだコーディングに変更した。これにより、取込み時間は平均で３．５秒となった。しかし、ピークの時間帯には遅いと言われることがあった。昭和６０年３月１８日にＶ３へバージョンアップした。この時にはディスク・バッファを１７０ＫＢから２３０ＫＢに増やしたが、マップドルーチンはまだ使わなかった。９日後の２７日に外来医師端末で使われるすべてのアプリケーションとユーティリティプログラムをマップドルーチンに載せた。（４４０ＫＢ）　これらにより、取込み時間は、平均２秒弱となり、３秒を越えることが無くなった。この段階でほぼ満足という声がほとんどとなった。今後の処理量と対象業務量の増加にも対応するため、余裕をもつために、ユーティリティ類の整理統合を続けた。（４月～７月）　８月２１日に再び計測したところ、平均１秒位で２秒を越えることはなかった。また、この時点で、他の処理でも１秒を越えるものは無い。（患者番号入力、患者情報表示＆確認、指導管理料などの表示、未実施検査チェック、メニュー選択、薬剤検索、検査オーダ病名登録、診察予約、処置・各科実施検査入力、フォローアップ患者登録、各種照会など）　現在では、外来医師のオーダについてのレスポンス上の問題は無くなったも

のと考えられる。

以上のように、システム全体での処理時間は８５％程度と、「ある程度の改善」と言えるものであったが、当院で問題となっていた夜間の業務、外来のピーク処理などは「大幅な改善」となり「ほぼ満足」している。あとＤＤＰのスループットがよくなれば、今回のバージョンアップについては「十分な満足」と言えるようになる。



オンライン1		中央情報		オンライン2	
外来診察室	２０ライン－２０台	医事入院会計	８ライン－　９台	病棟	２８ライン－４９台
外来各科受付	７ライン－１４台	薬局倉庫など	５ライン－１０台	検査室	２５ライン－３６台
採血	２ライン－　４台	会計課	１ライン－　２台	看護科	１ライン－　２台
薬局処方箋用	１ライン－　２台	オープン利用室	３ライン－　６台	給食	２ライン－　４台
医事外来	９ライン－１１台	情報企画室	９ライン－１３台	医事入院受付	１ライン－　２台
マシン室	２ライン－　４台	マシン室	３ライン－　６台	マシン室	２ライン－　４台
		その他	５ライン－　８台		
小計	４１ライン－５５台	小計	３４ライン－５４台	小計	５９ライン－９７台

総計　１３４ライン－２０６台
(内訳) カナＣＲＴ：１２１台，１０インチプリンター：４３台，１５インチプリンター：２５台
マークカードリーダ：２台，バーコードリーダ：１台，漢字ＣＲＴ：４台，漢字プリンター：３台
(検査自動分析器) 直結：２台，ＣＲＴ経由：１台，コントロールＢＯＸ経由：４台

図１　羽曳野病院情報システムのハードウエア構成

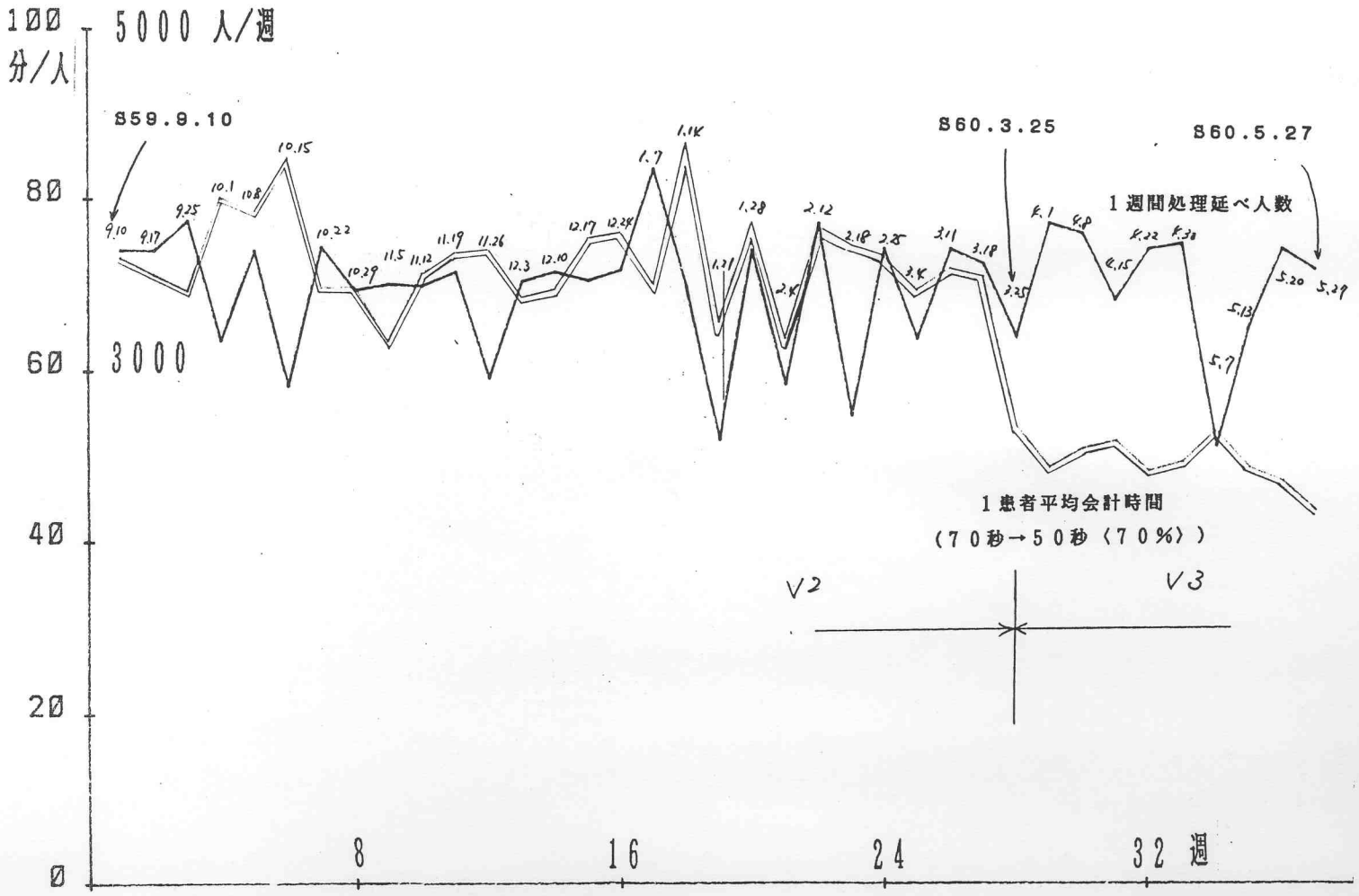


図2 医事外来窓口会計の処理時間の変化(9:30-12:30のピーク時間帯について)

昭和６０年１月　９日　（計測）平均５秒（３～９秒）
昭和６０年１月１０日　（改修）アプリケーションを改修
　　グローバル・アクセス回数削減
　　外部ルーチンを内部モジュール化
昭和６０年１月１６日　（計測）平均３．５秒
昭和６０年３月１８日　（改修）ＤＳＭ１１－Ｖ３へ
　　ディスク・バッファを１７０→２３０ＫＢへ
昭和６０年３月２７日　（改修）マップド・ルーチン化（４４０ＫＢ）
　　（計測）平均２秒弱（３秒を越えることは無かった）
昭和６０年４月～７月　（改修）ユーティリテイ類整理統合
昭和６０年８月２１日　（計測）平均１秒位（２秒を越えることは無かった）

☆外来医師端末の他の処理では１秒を越えるものは無い。
（患者番号入力、患者情報表示＆確認、指導管理料など表示、未実施検査チェック、メニュー選択、薬剤検索、検査オーダ、病名登録、診察予約、処置・各科実施検査入力、フォローアップ患者登録、各種照会など）

図３　外来医師端末での前回処方の取込み時間の変化

表1　バージョンアップ前後の月次処理時間の比較

業務名	V2.0AJ S59.11月度 処理時間(分)	ページ数	V3.0J S60.4月度 処理時間(分)	ページ数	増減時間	備考
外来保留者リスト	25	7	20	6	− 5	
外来レセプトワーク作成	216	/	95	/	−121	1枚当り 2.9秒 → 1.3秒(45%)
外来レセプト枚数	4	1	3	1	− 1	
外来レセプト単独発行	294	4512	192 (10K)	4410 (5K97)	−102	1枚当り (67%) (36%) 3.9秒→2.6秒→1.4秒
〃 病名リスト	331	710	277	705	− 54	
外来レセプト併用発行	50	782	32	852	− 18	15:00より実行
〃 病名リスト	54	129	62	135	+ 8	〃
外来レセプト公単発行	11	138	10	124	− 1	〃
〃 病名リスト	11	26	10	24	− 1	〃
外来レセプト老人発行	53	775	35	763	− 18	〃
〃 病名リスト	58	131	57	132	− 1	〃
外来レセプト退職発行	14	169	11	184	− 3	〃
〃 病名リスト	12	28	13	31	+ 1	〃
老人一部負担金助成者リスト(併用)	2	8	2	8	0	〃
〃 (老人)	7	19	3	19	− 4	〃
科目別入金表	28	2	17	2	− 11	
調定明細書	1	1	3	2	+ 2	S59.11月度には、この業務は実施されておりませんでしたのでS60.3月度と比較しております。
外来個人別請求状況	55	242	38	238	− 17	
未収金整理簿	13	52	7	56	− 6	
薬品出庫カード	69	1514	83	1881	+ 14	
入院レセプト単紙 ワーク作成	124	/	49	/	− 75	
〃 発行	163	701	69	601	− 94	
〃 病名リスト	48	115	40	98	− 8	
調剤件数リスト	18	3	5	3	− 13	

調査期間 S60.4.1 ~ S60.5.8

○EPは、ほぼシングルジョブと言えるもの

業務名	S59.11月度		S60.4月度		増減時間	備考
	処理時間	ページ数	処理時間	ページ数		
入院レセプト併用 ワーク作成	155		62		− 93	
〃 発行	64	354	29	287	− 35	
〃 病名リスト	21	63	25	60	+ 4	
入院レセプト公単 ワーク作成	63		36		− 27	
〃 発行	57	212	18	190	− 39	
〃 病名リスト	17	48	17	43	0	入院レセプト平均
入院レセプト老人 ワーク作成	65		44		− 21	1件当りワーク作成 13.1秒→9.9秒(76%)
〃 発行	55	322	30	253	− 25	1件当り発行 10.0秒→6.6秒(66%)
〃 病名リスト	21	50	27	56	+ 6	
入院レセプト退職 ワーク作成	46		37		− 9	
〃 発行	7	30	6	50	+ 1	
〃 病名リスト	4	13	6	20	+ 2	
老人一部負担金助成者リスト(併用)	1	4	3	3	+ 2	
〃 (老人)	3	13	3	12	0	
レセプト枚数(外来)	(4)	1	(3)	1	− 1	
主保険有効期限切れリスト(外来)	(2)	2	(2)	1	0	
レセプト枚数(入院)	6	1	2	2	− 4	
主保険有効期限切れリスト(入院)	2	6	4	2	+ 2	
外来患者統計	18	62	11	60	− 7	
外来科別売上月報	2	18	3	18	+ 1	
当月新入院統計	22	10	7	10	− 15	
出納.セイサイ 日次ヲアテート	26		23		− 3	
出納月報	17	148	45	104	− 28	
見舞入執行状況 ワーク作成	17		9		− 8	

業務名	S59.11月度		S60.4月度		増減時間	備考
	処理時間	ページ数	処理時間	ページ数		
購入執行状況発行	3	8	1	8	− 2	月報
値引ワースト ワーク作成	17	／	9	／	− 8	〃
〃 発行	6	12	2	12	− 4	〃
薬品入出庫状況 ワーク作成	94	／	18	／	− 76	四半期月報
〃 発行	4	12	1	8	+ 3	〃
購入執行上位100位 ワーク作成	71	／	56	／	− 15	〃
〃 発行	8	12	9	8	+ 1	〃
薬効別購入執行 ワーク作成	31	／	32	／	+ 1	〃
〃 発行	1	6	1	4	0	〃
値引ワースト ワーク作成	34	／	32	／	− 2	〃
〃 発行	4	21	2	12	− 2	〃
業者別購入状況 ワーク作成	17	／	27	／	+ 10	〃
〃 発行	29	20	138	82	+ 109	〃
外来総括集計（ソウカツファイル作成）	102	／	55	／	− 47	
〃 （aマーク付加）	7	／	13	／	+ 6	
外来レセプト不整合チェック	93	1	78	1	− 15	
入院患者科別統計	5	2	3	2	− 2	S53.11月度には、この業務は実施されておりませんでしたので S60.3月度と比較しています
入院患者延人数統計	126	2	32	2	− 94	〃
調剤.出納統計 グラフプロット	41	6	44	6	+ 3	
検査月報	26	30	22	28	− 4	
在院一括請求書						S60.4月度は通常と違う方法で発行した為，比較しておりません
請求NOリスト	(11)	23	(11)	22	0	
薬品出庫カード ワーク作成	13	／	5	／	− 8	
〃 発行	26	1242	27	1468	+ 1	

業務名	S59.11月度		S60.4月度		増減時間	備考
	処理時間	ページ数	処理時間	ページ数		
入院レセプト個人別 ワーク作成	22		5		− 17	
〃 発行	53	117	15	54	− 38	
感染症サーベイランス リスト	(1)	4	(1)	4	0	
退院疾病統計 リスト	(12)	16	(13)	16	+ 1	
退院疾病集計表	(18)	9	(14)	9	− 4	
退院疾病統計 グラフプロット	41	6	39	6	− 2	
患者給食台帳	134	233	86	224	− 48	
ディスク・ブロック・タレイ	12	24	6	14	− 6	オンライン2を除く、ファイルKILL前後のDBT処理
月次不要ファイル消去(センター)	(36)		(42)		+ 6	
〃 (オンライン1)	(63)		(72)		+ 9	
〃 (オンライン2)	(30)		(29)		− 1	
外来病名リスト	(16)	18	(23)	18	+ 7	
入院疾病統計 リスト	(19)	16	(23)	16	+ 4	
入院疾病集計表	(6)	6	(9)	6	+ 3	
外来病名集計	(12)		(12)		0	
入退院疾病統計 外来病名統計 グラフプロット	152	23	148	23	− 4	
テーブル転送MTセーブ(センター)	7		5		− 2	
〃 (オンライン1)	23		22		− 1	
〃 (オンライン2)	4		4		0	
テープル転送DDPリストア(センター)	(21)		(24)		+ 3	プログラムを変更した為、S60.4月度の方が時間がかかっています
〃 (オンライン1)	(64)		(88)		+ 24	〃
〃 (オンライン2)	(8)		(8)		0	
外来項目別集計	147		77		− 70	
貯蔵品出納簿	253	1406	242	1361	− 11	S59.11月度にこの業務を実施しなかった為、S59.12月度と比較しております

業務名	S59.11月度		S60.4月度		増減時間	備考
	処理時間	ページ数	処理時間	ページ数		
入院総括集計(シーケンスファイル作成)	242	/	116	/	−126	
〃 (リラカンファイル作成)	37	/	22	/	− 15	
〃 (Bマーク付加)	3	/	6	/	+ 3	
入院保留者リスト	9	12	12	24	+ 3	
外来レセプト売上日報	2	14	2	16	0	
入院レセプト売上日報	19	128	12	94	− 7	
入院病棟別売上日報	278	48	243	48	− 35	
医事売上統計集計	6	/	4	/	− 2	
医事売上統計 グラフプロット	102	34	98	36	− 4	
項目別入院集計(ファイル作成)	261	/	154	/	−107	
項目別プリントアウト(ファイル作成)	13	/	34	/	+ 21	
項目別集計 発行	84	415	62	412	− 22	
項目別ファイル転送	1	/	1	/	0	
医事薬剤請求効率表(ファイル作成)	79	/	84	/	+ 5	
〃 発行(年報)	26	48	27	50	+ 1	
〃 発行(月報)	52	20	26	20	− 26	
TB薬請求状況表	3	2	3	2	0	
抗結核薬グラフプロット	145	16	178	16	+ 33	
入院レセプト分整合チェック	12	1	14	1	+ 2	
Inverted File作成	(452)	/	(427)	/	− 25	
固定資産減価償却明細書	126	220	173	144	+ 47	S59.11月度にこの業務を実施していなかった為、S60.2月度と比較しております。
計	5,638分	→	4,788分		(85%)	

ＤＳＭ－Ｖ２とＶ３のスピード比較

○山本　和子、須藤　正克

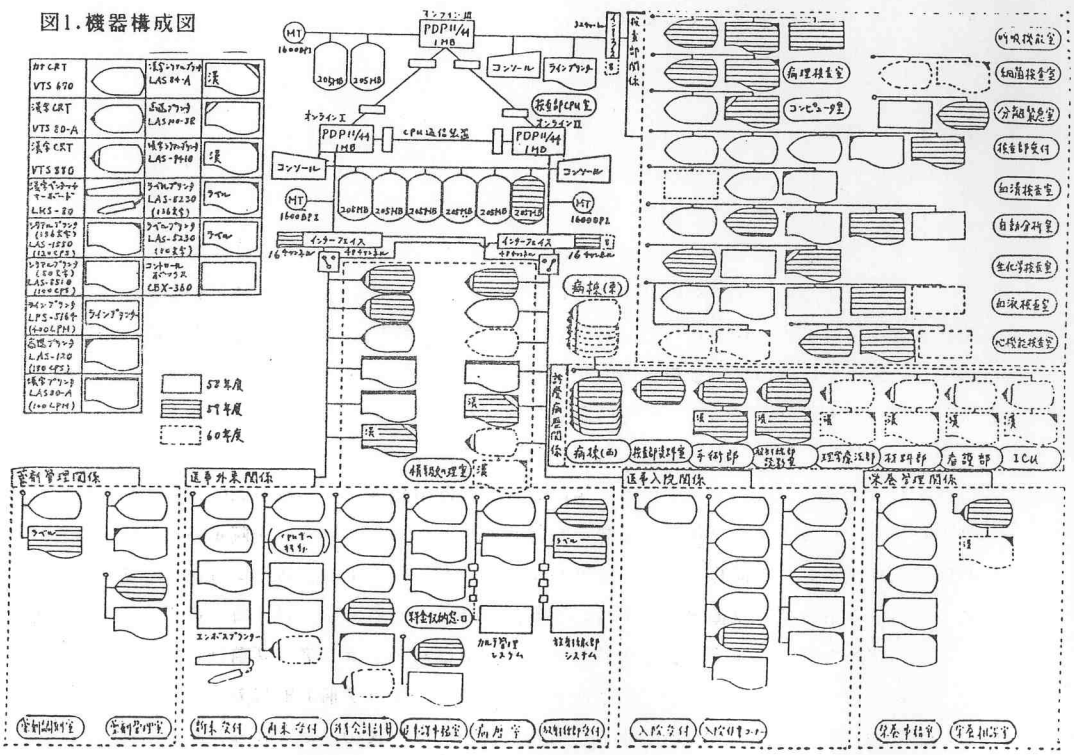
福井医科大学医学部附属病院情報処理部
福井県吉田郡松岡町下合月２３

［要約］
昭和60年 4月中旬に、ＤＳＭ－１１（漢字）－Ｖ２からＶ３へＯＳのバージョンアップを行ない、前後の処理速度を測定し比較した。処理速度として毎日午前11時にメニューから次のメニューへ移る時間を測定したところ、外来系ＣＰＵではバージョンアップにより58％時間短縮され、外来窓口会計での患者待ち時間が減少した。入院系ＣＰＵでは業務量の増大もあって減少していない。帳票類の出力時間も短縮され、レセプト編集時間は、外来系ＣＰＵで37％短縮、入院系ＣＰＵでは51％短縮された。

１．はじめに

福井医大病院では、ＰＤＰ１１／４４を 3台用いて、外来系ＣＰＵで医事外来と薬剤部の業務を、入院系ＣＰＵで医事入院・栄養・診療病歴業務を、検査系ＣＰＵで検査部業務を行なっている。この 3台のＣＰＵは高速通信回線により、他のＣＰＵへのデータ転送とデータ参照が可能である。また、外来系と入院系のＣＰＵは端末切り換え装置により、一方がダウンした時は他方に切り換えて稼働可能である。各ＣＰＵのメモリーは各々１ＭＢで、その機器構成図を図 1に示す。ＯＳはＤＳ

図1.機器構成図



M－１１（漢字）を用いている。

ＤＳＭ－１１Ｖ２からＶ３へのバージョンアップは 4月13日（土）の業務終了後から 4月15日早朝にかけて実施した。その後、 7ビットを 8ビットに変換するために、ファイルの整理（表 1）を行ない、 7月20日（土）と21日（日）に高速通信回線を利用しているファイルを 7ビットから 8ビットに変換した。残りのファイルは 9月に 8ビットに変換予定である。今回、バージョンアップ前後の処理速度を比較検討したので報告する。

表1.ファイル整理作業の内容

5月～ 6月	昭和58年10月より昭和60年 5月までの外来入院両レセプトファイルを病歴システムに転送、整理、８ビット変換。同時に磁気テープにとる。
6月10日	入院レセプトファイルを病歴システムへオンラインで転送開始。
6月～ 7月	病歴の検査ファイルの整理と移動、８ビット変換、磁気テープにもとる。

２．方法

処理速度として、オンライン時の応答速度と毎月定期的に出力される主要帳票類の出力時間を用いた。昭和59年 6月 1日、全端末について１時間毎に応答速度（データ入力後リターンキーを押して次の画面が現われるまでの時間）と、メニュー変更時間（メニュー番号を入力して次のメニュー画面が現われるまでの時間）を測定したところ、業務内容および入力したデータ量により大きなバラツキはあるが平均的にみると応答速度とメニュー変更時間とは大差ないことが判明した（図2)。情報処理部ではデータ入力を行なっていないので、応答速度は測定できないため、メニュー変更時間で代用することとし、昭和60年 1月より昭和60年 7月まで、毎日午前11時のピーク時にその時のジョブ数とその前後のメニュー変更に要する時間（前後の平均値をとる）を測定し、バージョンアップ前と後を比較した。なお、外来系は外来会計入力画面から再診受付メニューへ移る時間を、入院系は入院基本データ入力画面から入院会計メニューへ移る時間を測定している。メニューによって速度は異なるので、外来系と入院系とを単純には比較できない。帳票類については実際の出力時間を測定した。

３．結果並びに考察

１）応答速度について

測定結果を図 3, 4に示す。外来患者数は月、火、水と増加し、以後減少して土曜日に最低になるという周期をくり返しながら、一日の患者数は 1月平均380 名に対し、 7月は531 名と増加している。外来系ＣＰＵのメニュー変更時間は、レセプト出力とか帳票類の出力時と重なると高くなるため変動がはげしいが、 1月から 4月へかけては月平均7.7 秒から13.6秒へと、患者数の増加とともに増加し、医事会計窓口にての患者待ち時間20分と極限状態であったのが、バージョンアップ後は患者の行列もなくなり、メニュー変更時間もバージョンアップ前10.1秒（100 %）に対し、バージョンアップ後は4.2 秒（42%）に減少（58%減）している。

在院患者数は、バージョンアップ前370 名に対し、 4月中旬はベット数が350 床から600 床に増床されたためバージョンアップ後の在院患者数は442 名と増加している。入院系ＣＰＵのメニュー変更時間は、外来系ＣＰＵ同様、非常に変動がはげしいが、平均するとバージョンアップ前の12.1秒に対し、バージョンアップ直後は6.8 秒と一時的に減少したものの 5月以降上昇し、平均12.7秒とバージョンアップ前よりも高い。その時の総ジョブ数はバージョンアップ前18.5に対し、バージョンアップ後23.1と増加し、実ジョブ数も同様にバージョンアップ前3.8 に対し、バージョンアップ後4.4 と増加している。ベット数の増床、バージョンアップ後のファイル整理作業（表１）、 5

月末の病棟端末の設置等による業務量の増大等によるジョブ数の増加が、バージョンアップ後もメニュー変更時間を押し上げたと考えられる。

メニュー変更時間と患者数、ジョブ数との関係をみたのが表 2である。外来系ＣＰＵでは、外来患者数、ジョブ数ともに、メニュー変更時間に大きな影響を与えていないが、少しは関係があるように見受けられる。入院系ＣＰＵでは、実ジョブ数がメニュー変更時間と関係が深い。今後、業務量の分散をはかる必要があると思われる。

２）主要帳票類の出力時間について

毎月定期的に出力される主要帳票類の出力時間は表 3，4の通りである。バージョンアップ前の帳票出力時間を100 %として、バージョンアップ後の帳票出力時間の割合を表 3，4の右端に示す。帳票出力時間は主としてプリンターの出力スピードに依存するので、バージョンアップ前後で大差はないが、入院系，外来系ともにかなり改善されている。当院のレセプト編集は、オンライン中に開始するので、月毎に条件が異なり、比較はできないが、大ざっぱにみれば、外来系ＣＰＵでバージョンアップ前 1～ 4月678 分（100 %）に対してバージョンアップ後 5～ 8月428.3 分（63%）と減少し、入院系ＣＰＵではバージョンアップ前 1～ 4月349 分（100 %）に対してバージョンアップ後 5～ 8月172 分（49%）と減少している。メニュー変更時間でみるかぎり、入院系ＣＰＵではバージョンアップによる影響はないように見受けられるが、レセプト編集時間は、バージョンアップにより49%に減少しているから、外来系ＣＰＵ同様、スピードアップしたことは間違いないようである。

４．おわりに

バージョンアップによって、処理速度はかなり改善された。今後は、ファイルの 7ビットから 8ビット変換への完成とマップドルーチンの採用、業務の整理、検討をし、現在以上にスピードの向上をはかるべく努力する予定である。

図2.外来系ＣＰＵの応答速度とメニュー変更時間

＊は情報処理部の端末で測定

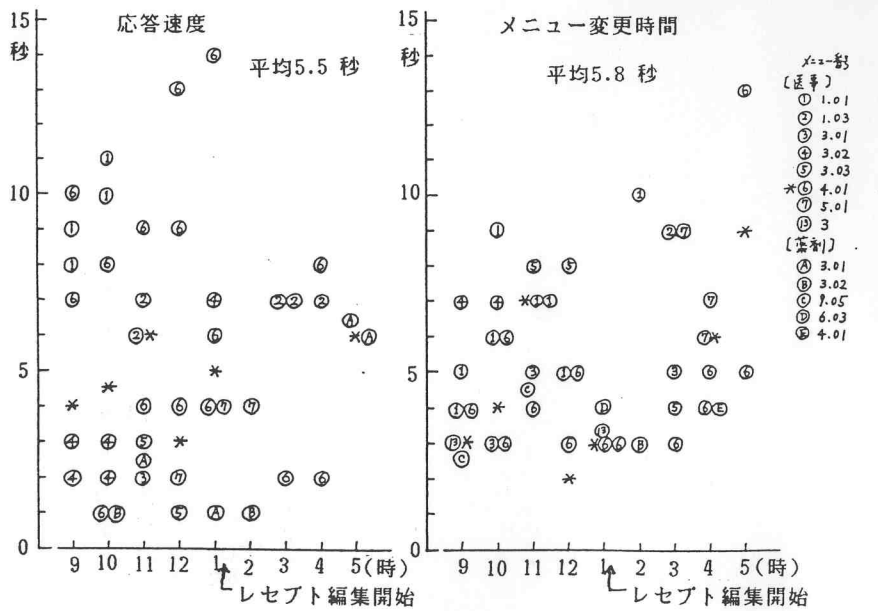
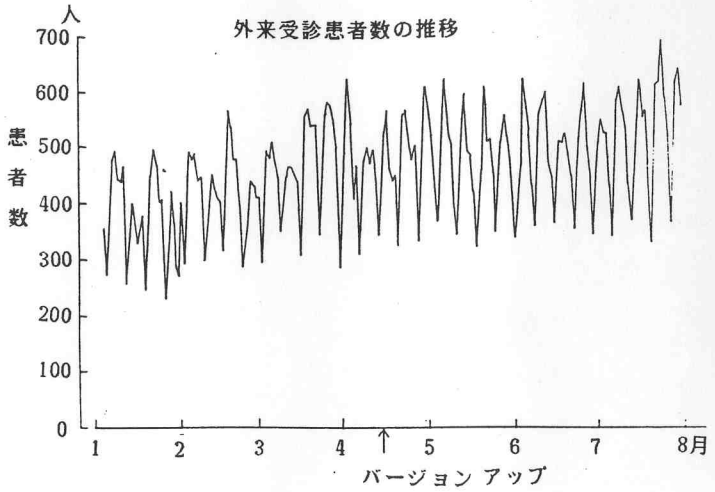
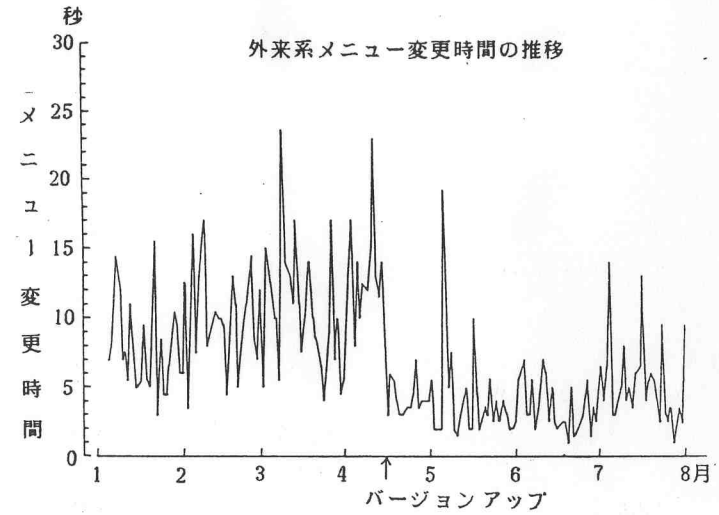


図3.外来患者数と外来系ＣＰＵのメニュー変更時間とその時のジョブ数の推移



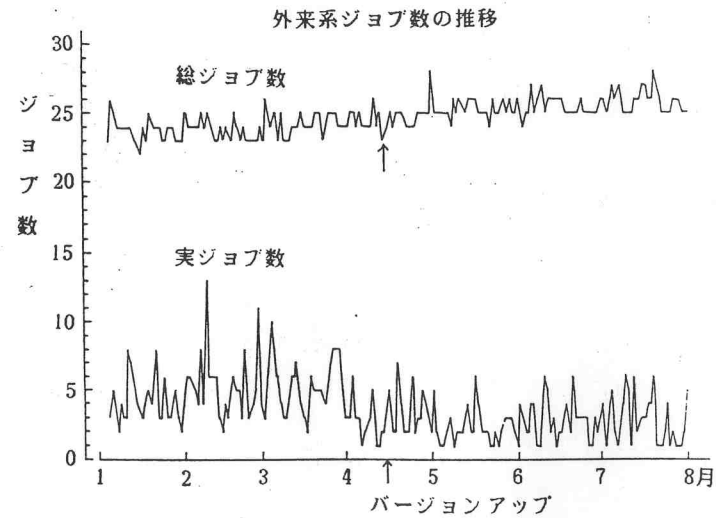
外来受診患者数

	平均値
1　月	３７９．５
2　月	４２１．９
3　月	４６５．５
4月前	４６４．７
1～4月平均	４２９．５
4月後	４８６．８
5　月	４８３．３
6　月	４９２．５
7　月	５３１．４
4～7月平均	５００．８



メニュー変更時間

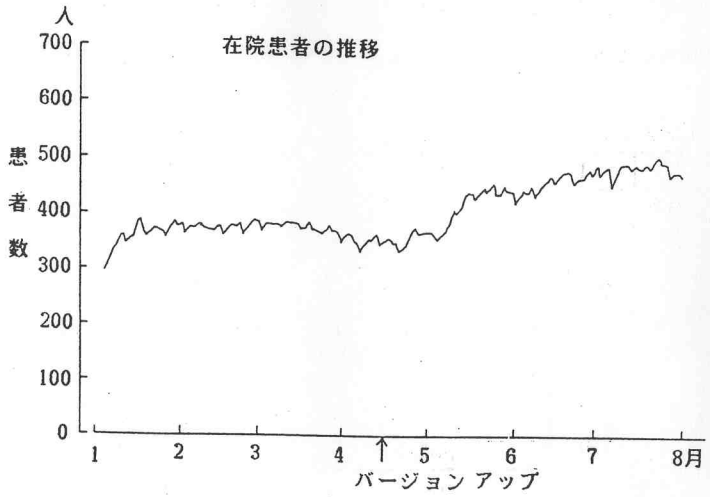
	平均値
1　月	７．７
2　月	１０．３
3　月	１０．４
4月前	１３．６
1～4月平均	１０．１
4月後	４．２
5　月	３．５
6　月	３．６
7　月	５．４
4～7月平均	４．２



実ジョブ数

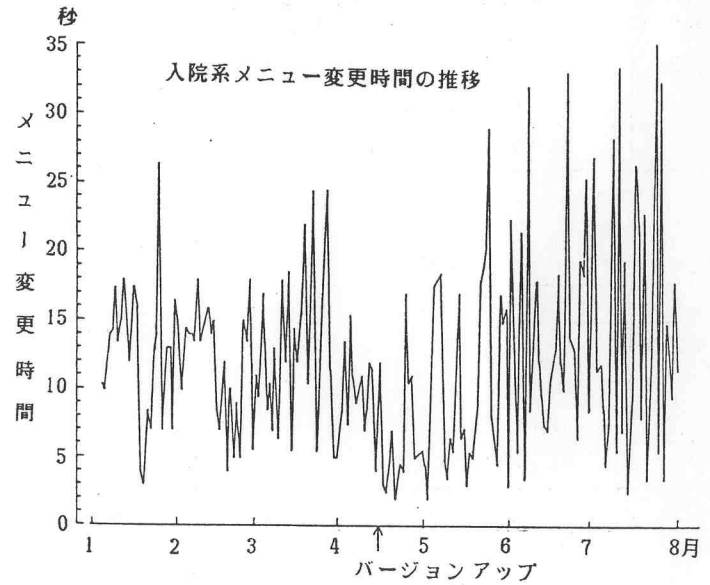
	平均値
1　月	４．１
2　月	５．４
3　月	５．３
4月前	２．７
1～4月平均	４．６
4月後	３．５
5　月	２．４
6　月	２．８
7　月	３．０
4～7月平均	２．８

図4.在院患者数と入院系ＣＰＵのメニュー変更時間とその時のジョブ数の推移



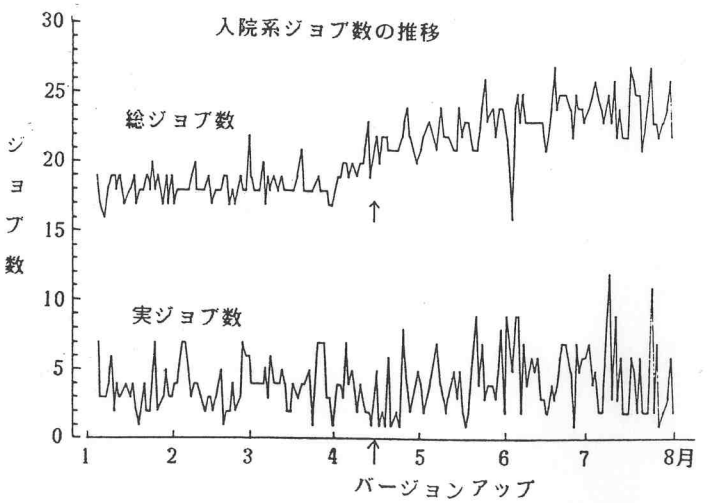
在院患者数

	平均値
1　月	362.0
2　月	377.0
3　月	377.9
4月前	355.7
1～4月平均	370.0
4月後	358.2
5　月	422.9
6　月	459.3
7　月	484.6
4～7月平均	442.2



メニュー変更時間

	平均値
1　月	12.7
2　月	11.7
3　月	13.0
4月前	10.0
1～4月平均	12.1
4月後	6.8
5　月	10.6
6　月	14.6
7　月	15.6
4～7月平均	12.7



実ジョブ数

	平均値
1　月	3.6
2　月	3.9
3　月	4.0
4月前	3.4
1～4月平均	3.8
4月後	3.4
5　月	4.1
6　月	5.2
7　月	4.4
4～7月平均	4.4

表2.メニュー変更時間と患者数、ジョブ数との関係

外来系バージョンアップ前1～4月の相関係数

	1	2	3
1.メニュー変更時間			
2.外来受診患者数	0．280		
3.総ジョブ数	0．332	0．259	
4.実ジョブ数	0．097	0．263	0．092

外来系バージョンアップ後4～7月の相関係数

	1	2	3
1.メニュー変更時間			
2.外来受診患者数	0．243		
3.総ジョブ数	0．048	－0．092	
4.実ジョブ数	0．387	0．146	0．020

入院系バージョンアップ前1～4月の相関係数

	1	2	3
1.メニュー変更時間			
2.在院患者数	0．161		
3.総ジョブ数	0．030	0．014	
4.実ジョブ数	0．540	0．050	0．071

入院系バージョンアップ後4～7月の相関係数

	1	2	3
1.メニュー変更時間			
2.在院患者数	0．376		
3.総ジョブ数	0．555	0．529	
4.実ジヨブ数	0．730	0．173	0．341

表3.外来系ＣＰＵの帳票出力時間

（単位：分）

出力帳票名	1月	2月	3月	4月	5月	6月	7月	%
レセプトソートファイル作成	21	18	16	120	22	15	19	42.7
医科レセプト編集	518	615	660	780	392	304	493	61.6
歯科レセプト編集	21	22	36	60	16	20	19	52.8
医科レセプト出力	550	690	660	840	535	640	708	91.6
歯科レセプト出力	30	40	50	70	50	120	60	161.4
歯科モニターリスト出力	70	90	130	71	74	76	104	93.8
生活保護患者名簿			10	20	20	15	12	104.4
妊婦・乳児健康審査リスト		3	5	5	5	3	10	139.5
外来患者月報（診療科別）	3	1	4	1	1	2	1	59.3
〃　（保険別）	3	1	4	1	1	2	1	59.3
簡易モニターリスト（編集）		~~180~~	~~240~~	~~420~~	~~116~~	~~270~~	~~469~~	~~101.8~~
〃　（出力）								
月次ファイル収集	360	480	492	420	331	300	255	67.4
レセプト総括表－1	21	20	15	16	14	13		75.0
〃　2	58	42	31	25	22	18		51.3
科別・事項別集計表	3	4	2	1	1	1		40.0
保険別・事項別稼働額	3	5	3	2	3	2		76.9
当月分審査請求ＭＯＶＥ	40	42	24	15	16	9		41.3
審査請求額内訳表	14	10	7	6	6	4		54.1
診療料金証明書（医科）	75	72	75	90	80	73	65	93.2
〃　（歯科）	5	5	5	5	5	4	5	93.3
未収金一覧	5	5	2	5	3	4	4	86.3
入院請求書発行一覧	10	15	10	10	16	17	18	151.1
入金月報	300	312	255	360	140	133	210	52.5
調定額・収納額対照表	60	90	78	105	30	30	30	36.0
妊婦・乳児健康審査リスト				5	3	5	2	66.7
国費患者集計表	0.25	0.17	1	0.5	－	0.17	0.5	69.8
診療項目コード別件数一覧	4	7	12	8	8	10	7	107.5
放射線統計	3	4	4	3	5	5	5	142.9
薬品使用効率一覧	11	15	10	13	12	15	10	100.7
診療科別新規患者名簿	30	102	210	320	90	135	85	62.4
外来患者月報（診療科別）		2	3	2	1	1	1	43.5
〃　（保険別）		4	3	3	2	1	1	40.4
新登録病名通知書		10	15	15	15	20	20	140.0
生活保護患者名簿	25	30	25	10	10	15	11	43.1
未収整理簿（外来）	19	20	25	25	15	15	10	76.3
〃　（入院）	12	15	15	15	15	15	20	116.1

表4.入院系ＣＰＵの帳票出力時間

（単位：分）

出力帳票名	1月	2月	3月	4月	5月	6月	7月	%
レセプトソートファイル作成	7	4	4	7	3	3	8	84.8
医科レセプト編集	372	300	310	390	139	164	159	44.9
歯科レセプト編集	5	5	4	10	3	2	6	61.1
医科レセプト出力	260	250	340	380	193	515	435	123.9
歯科レセプト出力	30	30	30	30	28	70	37	150.0
実施献立表（Ａブロック）	66	55	60	70	50	40	43	70.7
荷重平均栄養年齢構成表	24	10	20	20	5	5	21	55.9
在庫品受け払い簿	90	60	65	70	35	55	72	75.8
栄養出納表	200	130	115	360	70	149	205	70.2
給食材料消費月報	330	240	230	320	175	215	227	73.5
食種別給食数表	348	260	220	315	117	225	127	54.7
給与一覧表	232	215	180	170	52	110	130	48.8
科別・保険別入院患者数	165	90	150	90	47	45	67	42.8
簡易モニターリスト		60	50	120	54	31	44	56.1
月次統計ファイル収集	210	320	330	235	288	128	153	69.3
実施食数表	318	260	270	280	240	300	105	76.2
栄養計算	410	360	315	380	180	160	265	55.1
診療料金通知書・領収証書	563	410	485	720	275	620	475	83.9
外来請求書一括発行				360	354	360	480	110.6
診療料金証明書（医科）	42	39	25	40	3	32	28	57.5
診療料金証明書（歯科）	1	1	1	1	3	1	3	233.3
入院患者名簿（病名つき）	30	35	15	35	30	74	20	143.8
退院患者名簿（　〃　）	26	34	20	65	25	37	55	107.6
外来受診日別患者名簿（病名つき）	780	720	810	885	640	1073	795	104.7
病棟別病床使用状況表	38	22	30	45	13	56	25	92.8
診療科別病床使用状況表	88	22	47	65	10	54	30	56.5
入院疾病統計	15	7	30	15	4	6	15	49.8
診療項目コード別件数一覧	15	10	14	10	10	10	6	70.7
放射線統計	15	4	8	8	6	6	6	68.6
薬品使用効率一覧	27	12	27	15	13	13	18	72.4
診療科別新規患者名簿	30	40	43	30	24	25	30	73.7
退院理由一覧	20	7	20	15	9	4	12	53.8
年齢階層別患者数	10	5	17	5	4	4	5	46.8
使用量予定リスト	486	330	420	295	240	290	270	66.4
〃	282	380	578	240	260	276	548	80.1
原価計算	180	270	277	240	295	225	135	92.3
実施献立表（全ブロック）	570	480	630	385	505	415	411	76.6
〃　（Ａブロック）	48	45	50	40	35	45	46	87.0

ＤＳＭ－１１　Ｖ３によるパーフォーマンスの向上の評価

○服部敏夫、上田清治、青木浩二、永井利廣、三嶋博昭、北川　護、河村徹郎、野村　裕

大阪府立　成人病センター　情報企画室
大阪市東成区中道１－３－３

『概要』大阪府立成人病センターでは、昭和６０年３月より日本語処理等の新機能を含んだ病院情報第二期システムを稼働させ、３月中にＤＳＭ－１１　Ｖ２からＶ３に移行した。Ｖ３の採用により、新システムのパーフォーマンスは、日本語処理等の新機能を拡充したにもかかわらず、従来のレベルを維持できているが、まだ十分とはいえない。そこで、パーフォーマンス向上の手段として、Ｖ３の新機能であるマップド・ルーチンについて、種々のテストを行ない、その効果、制約等について調査したので報告する。

［１］はじめに

大阪府立成人病センターでは、病院情報第二期システムとして、昭和５９年１０月にハードウェアをＰＤＰ－１１／４４　５台とローカル・エリア・ネットワーク(ＬＡＮ)に、昭和６０年３月には適用業務システムを新システムに更新した。　第二期システムの特徴としては、(1)ＰＤＰ－１１／４４　５台のネットワークによる分散処理、(2)ＰＤＰ－１１／４４とＩＢＭ４３６１とのデータ通信、(3)ＣＰＵと端末間にＬＡＮを採用し、拡張性・障害対策を充実、(4)日本語処理を採用、(5)オンライン業務の拡充(栄養管理、薬品管理、事務管理等)、(6)アプリケーションの改良(医事会計レセプト・イメージ編集、照会機能の充実等)、(7)障害対策の充実、などがあげられる。新システムには、Ｖ２．０Ｊのままで移行したが、機能の拡充、特に日本語処理のため、パーフォーマンスが予想以上に低下した。

そこで、昭和６０年３月上旬に、従来のＤＳＭ－１１　Ｖ２．０ＪからＶ３．０Ｊに変更し、３月下旬には、グローバル・ファイルを８ビット化し、また、ルーチン内で漢字・非漢字モードをこまめに切りかえるなどのアプリケーション上の工夫により、パーフォーマンスを従来のレベルにまで向上させることができた。パーフォマンスが特に問題になる外来窓口会計では、この間の会計処理人数の推移は以下のようになっている。

(1)８５／２／４　：ＤＳＭ－１１　Ｖ２．０Ｊ　旧業務システム　　約３８０人／Ｈ
(2)８５／３／５　：ＤＳＭ－１１　Ｖ２．０Ｊ　新業務システム(漢字化)　約２１０人／Ｈ
(3)８５／４／１　：ＤＳＭ－１１　Ｖ３．０Ｊ　新業務システム(漢字、8ビット)　約３９０人／Ｈ

しかしながら、この数字は最多患者来院日の設定数　４６０人／Ｈと比べるとまだ十分とはいえず、さらにパーフォーマンスの向上が望まれている。

そこで、パーフォーマンス向上の１つの手段として、ＤＳＭ－１１　Ｖ３の新しい機能であるマップド・ルーチンに注目し、当システムへの適合性、効果などを調べるためにさまざまな実験を行なった。また、ＤＳＭ－１１のユーティリティの１つであるPerformance Statistics Utilityについても使用してみたので報告する。

［２］．マップド・ルーチン機能

マップド・ルーチン機能とは、使用頻度の高いルーチンを主記憶上に常駐させ、ルーチンを呼ぶためのディスク・アクセスおよびルーチンのパーティション内へのロードを軽減する機能である。マップド・ルーチン機能の概略を以下に示す。

(1)常駐エリア・サイズは、ＳＹＳＧＥＮ時に指定する。パーティション・プールと同じエリアにとられるので、マップド・ルーチン・エリアの分だけパーティション・プールが少なくなる。
(2)マッピングされるルーチンのサイズは、８ＫＢ以下でなければならない。８ＫＢ以上の場合は

ロード時にはじかれる。

(3)マッピングされるルーチンの数は、エリア・サイズ内ならば、幾つでもよい。ひとかたまりのルーチンでルーチン・セットをつくる。

(4)マップド・ルーチンをコールできるパーティションは、８ＫＢ以下でなければならない。８ＫＢ以上の場合はたとえマップド・ルーチンをコールしてもディスクからロードする。

(5)マップド・ルーチンのロードは、ルーチン・セット単位に行なう。１ＵＣＩにつき１ルーチン・セットのみロード可能。また、最大７ルーチン・セットまでロード可能。ロードは、スタート・アップ時、またはユーティリティにより行なう。

(6)マップド・ルーチンのコールは、まず、マップド・ルーチン・ディレクトリをバイナリ・サーチし、あれば、マップド・ルーチンを使用する。ただし、マップド・ルーチン自体はパーティション内にロードされず、パーティションはすべて変数域として使える。なければ、ディスクからロードされる。

(7)マップド・ルーチンのロード、DISABLE/ENABLE、ロードされたマップド・ルーチンの表示、ルーチン・セットの追加・修正・削除を行なうためのユーティリティ（^ＲＭＡＰ）がある。

［３］Performance Statistics Utility(^ＲＴＨＩＳＴ)

マップド・ルーチンのテストを実施するにあたって、実システムのルーチン、グローバル参照回数などの基礎資料を得るために、システム負荷が大きく、使用が最も望まれている外来会計システムにPerformance Statistics Utilityを適用し、パーフォーマンス測定を行なった。

このユーティリティは、実際に稼働中のシステムのある特定時間内の(1)キュー待ちのジョブの数、(2)データベースに対するイベントの回数(ルーチン、グローバル参照、ディスクへの実アクセス回数、論理的な参照回数、端末入出力回数、ＤＤＰ入出力回数等)、(3)(2)より派生する比率、(4)ディスクの使用率、(5)各ルーチン毎の使用時間比率、(6)各グローバル毎のアクセス回数などを測定するものであり、パーフォーマンスに対してどの部分がネックになっているかの解析の手助けとなる。

以下に昭和６０年８月１９日(月)の外来会計システムのピーク時のそれぞれ１０分間の測定結果の一部を示す。この表より、当センターでは、(1)Disk BoundよりもCPU Boundの待ちジョブが多い、(2)ルーチン参照はグローバル参照の約１／５である、(3)ピーク時間帯では、アイドル時間の比率が遅さを感じはじめるといわれている２５％を割っている、などが読み取れ、定量的にもパーフォーマンス改善が必要であることがわかる。

項　目	'85／08／19		
	10：31AM	11：27AM	12：34AM
平均稼働ジョブ数	17.00	15.67	15.83
Total CPU Bound	2.69	3.46	1.15
Total GLOBAL Bound	1.76	0.93	0.68
In GLOBAL	2.01	1.46	0.90
ルーチン参照(ROUREF)回数	5.15	5.13	3.21
ｸﾞﾛｰﾊﾞﾙ参照(GLOREF)回数	27.31	24.53	17.28
論理ＲＥＡＤ回数(LOGRD)	107.28	106.11	72.57
物理ＲＥＡＤ回数(READS)	11.20	8.80	7.31
ﾌﾞﾛｯｸREAD要求/ｸﾞﾛｰﾊﾞﾙ参照比 (LOGRD/(ROUREF+GLOREF))	3.31	3.58	3.54
実ﾌﾞﾛｯｸREAD/ﾌﾞﾛｯｸREAD要求比 (READS/LOGRD)	0.10	0.08	0.10
ﾄｰﾀﾙ ﾃﾞｨｽｸ ﾄﾞﾗｲﾌﾞ READ/WRITE(%)	72.05	56.94	40.09
アイドル時間比率（%）	23.17	11.81	45.17

［３］マップド・ルーチン　テスト

（テスト１）　外来会計入力（ＰＤＰ１１／４４＋ＲＡ６０）

いくつかの診療行為を組み合わせた外来会計データを１０人分自動入力し、マップド・ルーチンを使用しない場合と使用した場合についてトータルの時間を測定した。測定は、１jobのみ，３job同時，５job同時に実行した場合をそれぞれ３回ずつ行なった。３，５jobの場合の内１回については、同時にPerformance Statistics Utility(＾ＲＴＨＩＳＴ)を実行させた。

入力データは、投薬としてホウリエキ(２薬剤のセット)、検査として乳腺術前検査(４検査のセット)を使用した。マップド・ルーチン・セットは、入力ルーチンのほとんどすべてを含む'ＦＵＬＬ'(３８ルーチン)を使用した。

テスト環境は、ＰＤＰ－１１／４４(1MB)＋ＲＡ６０(205MB)ｘ２を使用し、マップド・ルーチン使用の場合は、Tied Terminal Modeで８KBのパーティションを設定し、未使用の場合は、１６KBのパーティションで行なった。なお、ディスク・バッファとしては、最大の230KBをとっている。

１，３，５Job稼働時のテスト結果を以下に示す。

Ｊｏｂ数		ｎｏＭＡＰ(ａ)(秒)	ＭＡＰ(ｂ)(秒)	ｂ　／　ａ	ａ－ｂ(秒)
１	１回目	１４６	１０９．７	０．７５１	３６．３
３	１回目	２６６	２２２．７	０．８３７	４３．３
	２回目	２６６．７	２２２	０．８３３	４４．６
	３回目	２６９	２２８	０．８４８	４１
	平均	２６７．２	２２４．２	０．８３９	４３．０
５	１回目	４０９．４	３７１．４	０．９０７	３８
	２回目	４１０	３７１．８	０．９０７	３８．２
	３回目	４１７	３７７．８	０．９０６	３９．２
	平均	４１２．１	３７３．７	０．９０７	３８．５

表より、１，３，５Job稼働時ともマップド・ルーチン使用時、未使用時の時間の差がほぼ一定であることがわかる。また、３，５Jobのそれぞれ３回目については、同時にPerformance Statistics Utilityを実行させたので、その結果の一部を示す。この表より、マップド・ルーチンの採用により、実際のDiskアクセス回数、Global Boundジョブ数が減少していることがわかる。

項　　目	３　ＪＯＢＳ		５　ＪＯＢＳ	
	ｎｏＭＡＰ	ＭＡＰ	ｎｏＭＡＰ	ＭＡＰ
Total CPU Bound	１．４８	１．６８	３．３８	３．４２
Total GLOBAL Bound	０．０６	０．０１	０．１１	０．０３
In GLOBAL	０．４８	０．１３	０．４８	０．１２
ルーチン参照(ROUREF)回数	４．８７	０．３６	５．２７	０．３８
ﾏｯﾌﾟﾄﾞ･ﾙｰﾁﾝ参照(MAPROU)回数	０．００	５．３４	０．００	４．８９
ｸﾞﾛｰﾊﾞﾙ参照(GLOREF)回数	１６．７２	２０．１８	１９．８６	１９．８７
論理ＲＥＡＤ回数(LOGRD)	１２１．１７	１０５．１１	１３４．４１	１００．０２
物理ＲＥＡＤ回数(READS)	３．７２	０．３６	２．８７	０．２６
ﾌﾞﾛｯｸREAD要求/ｸﾞﾛｰﾊﾞﾙ参照比 (LOGRD/(ROUREF+GLOREF))	５．６１	５．１２	５．３５	４．９４
実ﾌﾞﾛｯｸREAD/ﾌﾞﾛｯｸREAD要求比 (READS/LOGRD)	０．０３	０．００	０．０２	０．００
ﾃｨｽｸ ﾄﾞﾗｲﾌﾞ ０　READ (%)	１３．８８	１．９５	１０．２８	１．３

(テスト2)

テスト1と同様の実験をマップド・ルーチン・セットをかえて行なった。テスト環境は、ＰＤＰ11/70+ＲＰ06で、マップド・ルーチン使用時は(1)標準パーティションを８KBに設定、(2)Tied Terminal Modeで８KBのパーティションを設定の２つの場合で行なったが差はほとんどなかった。マップド・ルーチン未使用の場合は１６KBのパーティションで実行した。ＣＲＴ端末は、ＶＴ－８０を用いた。

以下に、テストの結果を示す。(単位は　秒)

入力データは、セット・コードを用い、それぞれ、Ｓ０２０(２薬剤),Ｓ７６００(胸部単純レントゲン),Ｓ００５(３薬剤),Ｓ０５１(乳腺術前検査)である。

NO	入力データ	ルーチン・セット		VT－80 1ジョブ	VT－80 5ジョブ
1	Ｓ０２０ Ｓ７６００	'Ｍ' (15ルーチン,42112B) (サブルーチンを除く必要ルーチンの90%をマップ化)	未使用 使用	１０４ ９９	２６６ ２５３
2	Ｓ０２０ Ｓ７６００	'ＳＵＢ' (7ルーチン,11392B) (サブルーチンのみマップ化)	未使用 使用	１０４ １０２	２８７ ２６８
3	Ｓ００５ Ｓ０５１	'ＦＵＬＬ' (38ルーチン,149760B) (全ルーチンをマップ化)	未使用 使用	１６６ １４６	４８６ ４５３

[５] 考察

マップド・ルーチンのテストにより、以下の点が明らかとなった。

(1)マップド・ルーチン化により、処理時間、ディスク・アクセス回数、Global Boundジョブの減少がみられた。

(2)処理時間減少の度合は、稼働ジョブ数による違いはあまりみられず、マップド・ルーチン化したルーチンのアクセス回数、サイズによる。

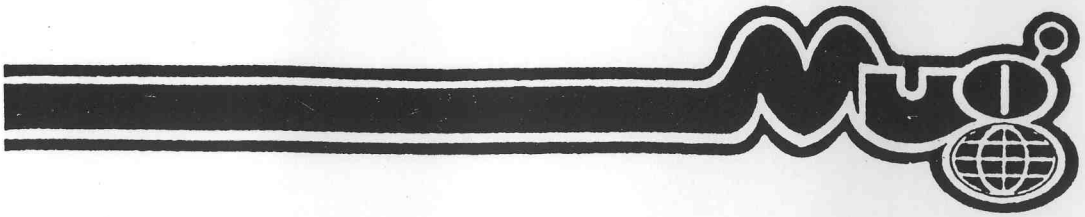
(3)ディスク・バッファ等の効果がきいているため、ルーチン・アクセス、ディスク・アクセスの影響の定量的な測定はむずかしい。

(4)Performance Statistics Utilityを稼働させても、システムに与える影響は少ないと思われる。

以上のテスト結果により、マップド・ルーチン機能が一応の成果をあげられるとの確認はできたが、実際に利用しようとすると、次のような理由で、パーティション・サイズの制約が大きな障害となる。当センターでは、標準のパーティション・サイズは最大の１６KBをとっており、また、ルーチン・サイズの平均は約３KBであるが、外来システムでは最大９KBのものがあり、５～７KBのものも１～２割ある。さらに、外来会計入力では、１患者分の入力データをすべてローカル変数としてもっているため、データによっては８KBを越える可能性がある。

これらの点から、現在のシステムでは、パーティション、ルーチン・サイズとも８KB以下というマップド・ルーチン機能の採用はむずかしい。マップド・ルーチン機能が使えるパーティション・サイズが１６KBになるという希望的観測もあるが、まず、(1)ルーチン、ロジックを改良して８KBのパーティションで実行可能とする、(2)現在の環境でも実施可能な業務に適用する、等を考慮して、今後さらに、パーフォーマンス向上の努力を続けていくつもりである。

なお、Performance Statistics Utilityを実際に使用してみて、システムが実際に稼働している時の状況を把握でき、パーフォーマンスの解析に役立つが、この値がこれぐらいの時はパーフォーマンスにこのような影響を及ぼすという判断基準のようなものがあれば、さらに役立つものと思われる。



第12回日本MUG学術大会

プログラム

大会長　山本和子

1985年8月24日(土)・25日(日)・26日(月)

芦　原

会場案内図

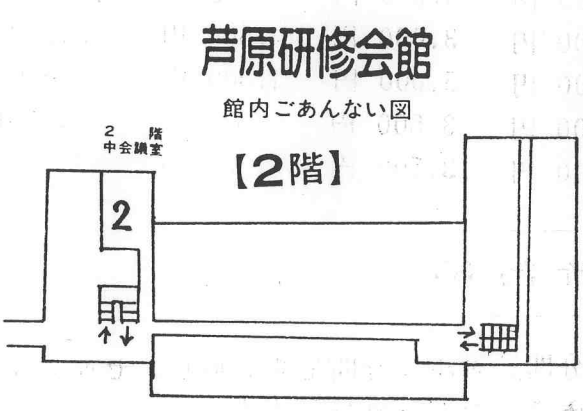
芦原研修会館　　〒910　福井県坂井郡芦原町舟津
TEL（0776）78－5300

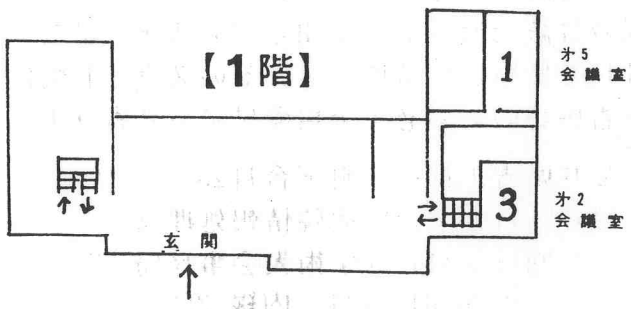
■交通案内

・京福三国線＝芦原湯の町駅より徒歩15分、バス 5分（舟津下車）
・国鉄＝芦原温泉駅よりバス10分
　国鉄バス：芦原湯の町行・湯の町下車
　京福バス：芦原湯の町経由・舟津下車
・航空＝小松空港より車40分

各地よりの所要時間

	国鉄	高速	航空
東京より	４時間	６時間	２時間
名古屋より	２時間30分	３時間	
大阪より	２時間30分	３時間	
	（芦原温泉駅まで）	（丸岡Ｉ．Ｃ経由）	





大会長より

本年は「考えるマンプス」というメインテーマで芦原温泉にて開催します。知識ベース時代に、ＭＵＭＰＳの果たすべき役割も大きくなりつつあります。このＭＵＭＰＳにＰｒｏｌｏｇも包含されるようになりました。また、マイクロＭＵＭＰＳも開発され、新しい各種の端末装置もふえて、用途が広がりつつあります。新知識を吸収し、将来展望について話し合いましょう。　多数の御参加を希望いたします。

会　期：８月24日（土）、25（日）、26（月）
会　場：芦原研修会館（左頁　案内図参照）

参加者各位

参加費：

７月１０日までに登録して下さい、７月１０日以降の登録者は当日参加費をいただきます。

	会　員	一　般	当日登録者 当日会員	当日一般
１．マンプス講習会	5,000 円	8,000 円	7,000 円	10,000円
２．教育セミナー（各々）	2,000 円	3,000 円	2,500 円	3,500円
３．大会参加費	3,000 円	5,000 円	4,000 円	7,000円
４．論文集代（送料込み）	3,500 円	3,500 円	3,500 円	3,500円
５．懇親会費	3,500 円	3,500 円	3,500 円	3,500円

発表者各位

１．発表時間は一題につき発表10分間、討論 5分間です。但し、セッションの最初には、セッション基調を置き、時間を延長します。
各セッションに与えられた時間を参考にして下さい。

２．スライドを使用される場合は該当セッション開始30分前までにスライド受付にご提出下さい。ＯＨＰを使用される場合は、そのスライド受付にご連絡下さい。また発表者到着確認のため必ず会場受付にお立寄り下さい。

連絡先き

〒910-11　福井県吉田郡松岡町下合月23
福井医科大学附属病院情報処理部
第12回日本ＭＵＧ学術大会事務局
TEL 0776-61-3111　内線 3580

第12回日本MUG学術大会プログラム

メインテーマ：「考えるマンプス」
会期：昭和６０年８月２４日（土）～２６日（月）
会場：芦原研修会館（福井県坂井郡芦原町舟津）
＊機器・書籍の展示　　全期間（２４～２６日）［第３会場（第２会議室）］

第１日目　８月２４日（土）

＊マイコンによる日本語マンプス講習会　AM 9:00 ～17:00
　マンプス初級より応用まで　AM9:00～PM12:30
　　講師　蝶理情報システム［第２会場（２階中会議室）］
　マンプス中級より応用まで　PM13:30 ～PM17:00
　　講師　住友電気工業　　［第２会場（２階中会議室）］
＊福井医大病院医療情報システム見学　PM 13:00～15:50
［福井医大病院情報処理部］

＊開会式　PM 16:50～17:00

＊イブニングセッション「マンマシンインターフェイス」　PM 17:00～18:30
　－－人間とシステムとの接点である端末装置の新しい展開－－
　　座長　木村一元（独協医大・総研ＭＥ室）［第１会場（第５会議室）］

＊ディナーセッション「ＭＵＧのあり方」　PM 18:30～20:00
　　座長　大谷元彦（藤田学園・医）　　［第１会場（第５会議室）］

第２日目　８月２５日（日）

第１会場（第５会議室）

＊特別講演Ｉ　AM 9:00 ～9:30
　　座長　吉田欣弥（三井造船システム）
　マンプスによる輸出貨物における書類作成と情報検索システムの
　適用例－－ネットワーク化への対応
　　服部泰夫（日本通運国際輸送事業部）

第12回日本MUG学術大会プログラム

I－I

＊Implementation I　AM 9:30 ～10:20

座長　田久浩志（産業医大）

1　M／VXについて
吉田欣弥（三井造船システム）

2　コンパイラ型MUMPSを用いたシステムの導入例紹介
石丸径美（三井造船システム）

3　U－MUMPSの機能強化
小林　勝、久江　正、上戸　隆、阪倉　明（住友電気・ME）

＊特別講演II　AM 10:20～10:50

座長　里村洋一（千葉大・医）

Frame Work（システム開発支援システム）
藤江　昭（日本SMS・システム開発部）

MD

＊医療へのアプリケーション　AM 10:50～11:40

座長　長谷川俊雄（福井医大）

1　汎用性を重視した臨床検査コンピューターシステムの設計・導入についてーー第三報　特に検査前処理業務についてーー
長原三輝雄、長谷川俊雄、黒田満彦（福井医大検査部）

2　福井医大病院における栄養管理システムについて
本間富士子、西秋人、関山明彦、佐藤裕保（福井医大医事課）

3　MUMPSの健診支援への人工知能的応用の検討
林恭平、六反奈利子、森田益次、山口希、青池晟、川井啓市
（京都府立医大公衆衛生）

第２会場（２階中会議室）

教育セミナー

＊モーニングレクチア　AM9:30～11:30

ルーチンのモジュール化と標準化
講師　嶋　芳成（久美愛病院）

（昼食　PM11:40 ～13:00 ）

運営委員会・評議員会　　大谷元彦

第12回日本MUG学術大会プログラム

第１会場（第５会議室）

＊総会　PM13:00 ～13:30

会長　大谷元彦（藤田学園・医）

＊招待講演　PM13:30 ～14:10

座長　須藤正克（福井医大）

知識工学と医療への応用

開原成允（東大・医・中央医療情報部）

＊特別講演Ⅲ　PM14:10 ～15:20

座長　河村徹郎（大阪成人病センター）

Ｐｒｏｌｏｇとは何ぞや

井手真理子（日本ＤＥＣ）

ＭＵＭＰＳのＰｒｏｌｏｇシンタックスの包含装備

内田達弘（マンプスシステム研究所・名城大学）

第２会場（２階中会議室）

D

＊デモンストレーションⅠ　PM15:30 ～16:15

座長　林寺　忠（国立京都病院）

1　ＳＥＴ　ＧＮＯＳＩＳ　＝　ＭＵＭＰＳ　＋　Ｐｒｏｌｏｇ

Donald A.Smith,William A.Ackerman,若井一朗

（マンプスシステム研究所）

2　ＳＰ－ＭＵＭＰＳ

小林勝、久江正、米田研、阪倉明（住友電気・ＭＥ）

D

＊デモンストレーションⅡ　PM16:15 ～17:00

座長　服部敏夫（大阪成人病センター）

3　ＮＥＣ９８０１とＳＨＡＲＰ　Ink Jet プリンターをつないで

カラーグラフィックスをハードコピーする

林寺　忠、和田行文、西角　淳、上坂邦夫（国立京都病院小児科）

4　研究用患者データ管理システムのマイクロマンプスへの変換

本多正幸、里村洋一（千葉大）、北村嘉章（神戸大）

＊懇親会　PM18:00 ～20:00　　［３階大会議室］

第12回日本MUG学術大会プログラム

第３日目８月２６日（月）

第１会場（第５会議室）

＊特別講演Ⅳ　AM9:00～10:00

座長　若井一朗（マンプスシステム研究所）

パソコンＭＵＭＰＳ開発と今後の展望

David J.Marcus Ph.D （ Micronetic Design Corporation）

新時代のネットワークを迎える漢字ＤＳＭ－１１

小林泰道（日本ＤＥＣ）

ＭＣ

＊マイコン・アプリケーション　AM10:10 ～11:00

座長　八日市谷　隆（東北大・医）

１　マイクロマンプスによる眼科救急外来患者の統計的観察処理

木村一元、西村雅晴＊、小暮文雄＊＊、関　亮＊＊

（独協医大　総研ＭＥ室，＊ 公衆衛生学，＊＊眼科）

２　マイクロコンピューターによる未熟児データーベースの作製

土屋喬義、田中吾朗＊、木村一元＊＊、本間　道

（独協医大　第一小児科，＊ 第二小児科，＊＊総研ＭＥ室）

３　入院台帳プログラム・パッケージ

林寺　忠、和田行文、西角　淳、上坂邦夫

（国立京都病院小児科）

Ⅰ－Ⅱ

＊ImplementationⅡ　AM11:10 ～12:00

座長　小林　勝（住友電気・ＭＥ）

１　マルチウインドー機能を持った日本語ＭＵＭＰＳ

鈴木利明（アレフコンピュータ）

２　ＭＵＭＰＳ下で実現したニューメディアシステムの紹介

山口光大、柿崎賢一（高岳製作所）

３　（株）高岳製作所・タイムリーシステム

重松郁也、佐々木一郎（高岳製作所）

第12回日本MUG学術大会プログラム

第２会場（２階中会議室）

教育セミナー

＊モーニングレクチア　AM9:30～11:30

データベースの汎用化と標準化

講師　嶋　芳成（久美愛病院）

（昼食　PM12:00 ～13:00 ）

第１会場（第５会議室）

S

＊システムデザイン　PM13:00 ～13:40

座長　本多正幸（千葉大・医）

1　スモールコンセプト（第二報）

田久浩志、馬場謙介＊

（産業医大　振動研究室、＊ 第二病理学教室）

2　時系列システムにおける保守時の診断仕様書

今泉幸雄（日本アップジョン）

W

＊ワークショップ「ＤＳＭ－Ｖ３をめぐる諸問題」PM13:40 ～14:50

座長　大櫛陽一（羽曳野病院）

1　ＤＳＭ－Ｖ３による業務スピード向上評価

大櫛陽一、古林栄次郎、野口　弘、寺村昌文（羽曳野病院）

2　ＤＳＭ－Ｖ２とＶ３のスピード比較

山本和子、須藤正克（福井医大情報処理部）

3　ＤＳＭ－Ｖ３によるパーフォーマンスの向上の評価

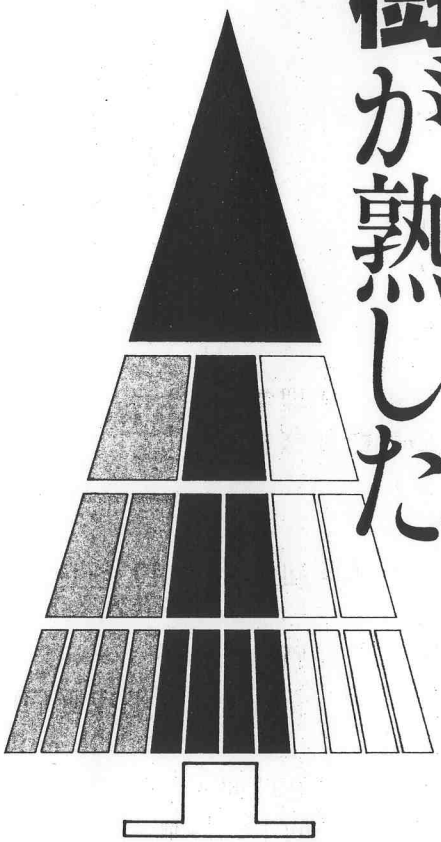
服部敏夫、上田清治、青木浩二、入江真行、三嶋博昭、

河村徹郎、野村　裕

（大阪成人病センター／大阪母子医療センター）

＊閉会式　PM14:50 ～

情報処理の"樹"が熟した。



▶SP-MUMPSは、プログラマー向けのデータベースアプリケーション開発ツール。

▶言語は、ANSI標準です。もちろん日本語プログラミングフルサポート。

▶SP-MUMPSは、コンパイラー(中間言語)方式で、従来のMUMPSの10倍以上の実行スピード。しかもディスクキャッシュ技術でミニコン並みのハイパフォーマンス。

▶SP-MUMPSを特におすすめしたい方は、次のようなシステムハウス、自主開発ユーザです。

- 簡易データベースパッケージではフレキシビリティに欠け、適用業務仕様を充分満足させられない方。
- COBOLでは、開発生産性の効率が悪いと思っている方。
- BASICでは、データベース操作に困っている方。

仕　様	
対象機種	NEC 9800シリーズ、N5200モデル05/05mkII、IBM 5550、メモリ384kB以上
フロッピーディスク	5インチ(2DD/2HD)または8インチ(2D)
オペレーティング システム	日本語MS-DOS
言語	ANSI標準 MUMPS(日本語対応)、プリコンパイル方式
データベース	論理ツリー型(内部B-tree構造) 可変長フィールド、可変長レコード、 スパース構造対応、データ階層 設計(親子・構造体反復)可

MUMPSはM.G.H(マサチューセッツ・ジェネラル・ホスピタル)とハーバード大学医学部で1966年開発されたデータベースシステムです。

SP-MUMPS

待望のパソコンMUMPS・コンパイラー版

価格 98,000円
(マニュアル別売)

新/製/品

ME開発室
大阪市此花区島屋1-1-3 〒554 ☎(06)461-1031(大代表)
東京都港区元赤坂1-3-12 〒107 ☎(03)478-3111(大代表)

住友電工