U.D.C. 800.92:681.322-181.48:681.32.072.06

マイクロコンピュータによる会話形簡易データベース言語“Hitachi Micro MUMPS”

"Hitachi Micro MUMPS" - A Conversational Data Base Language for Microcomputers

渡辺　坦*　*Tan Watanabe*
大沢恒春*　*Tsuneharu Oosawa*
塚田湧長**　*Wakunaga Tsukada*

マイクロコンピュータの出現により，計算装置は安価になったが，これを使いこなすのは必ずしも容易でない。“Hitachi Micro MUMPS”は，マイクロコンピュータによるデータベースの会話形利用が，コンピュータに不慣れな人にも，容易に行なえるようにするプログラミング言語である。これは，従来主としてミニコンピュータで使われてきたMUMPS言語をマイクロコンピュータ向きに修正し，更に，他の言語と結合する機能や誤操作対策の機能を充実した言語であり，現在，医療事務システム“HIMEC-10”で使える。本言語を使った例によると，アセンブリ言語に比べ，記述量は$\frac{1}{30}$に減り，工数は$\frac{1}{8}$に減った。この論文では，その機能と特徴，データベース処理の高速化の手法及び適用例について述べる。

1 緒　　言

データをデータベースに蓄えておき，必要なときにそれを会話形で利用できれば，事務処理やデータ整理の効率は格段に向上する。コンピュータのこの高度な利用形態をミニコンピュータでも可能としたのがマサチューセッツ総合病院で開発されたプログラミング言語MUMPS[1](Massachusetts General Hospital Utility Multi-Programming System)である。“Hitachi Micro MUMPS”[2](以下,“Micro MUMPS”と略す。)は，これを更にその数分の一の価格であるマイクロコンピュータで簡易に利用可能にすることにより，高度の専門家に限定されがちであったデータベースの作成を，広い層にわたるユーザー自身にできるようにする言語である。これは，マイクロコンピュータによる医療事務システム“HIMEC-10”[3]で使える。以下，その概要と実現方式，及び適用例について述べるが，そこでは，実用性の高い形でマイクロコンピュータ化するために考慮した点とその結果に重点をおく。

2 システム概要

2.1 特　　徴

マイクロコンピュータのプログラムをしやすくする高級言語は，幾種類かある。BASICは，コンピュータの初心者でも使える会話形の簡易言語であり，数値計算の能力は高いが，データベースは備え付けでない。日立マイクロコンピュータ用高級言語PL/H[4]は，システムプログラムの記述能力は高いが，会話形でなく，データベース機能ももたない。“Micro MUMPS”は，データベースを会話形で即時処理でき，数日で修得できる簡易言語である。

ディスクファイルを使うとき，他の多くの言語では，データの形や量を指定し，その格納や検索の手続きを書く必要があるが，“Micro MUMPS”では，これらは一切不要であり，実行させたい命令だけを書けばよい。ディスク上のデータも，主記憶のデータと同じく，添字付き又は添字なし変数として表わし，書き込み，読み出しを，それぞれ，変数に対する代入，参照として表わす。

上記特徴は，医療分野を中心として世界的に普及しているミニコンピュータ用MUMPSと共通のものである。MUMPSに対しては，1977年にANSI(米国の国家標準規格協会)の言語規格が標準MUMPSとして制定されている。

この“Micro MUMPS”の言語仕様は，標準MUMPSに準拠

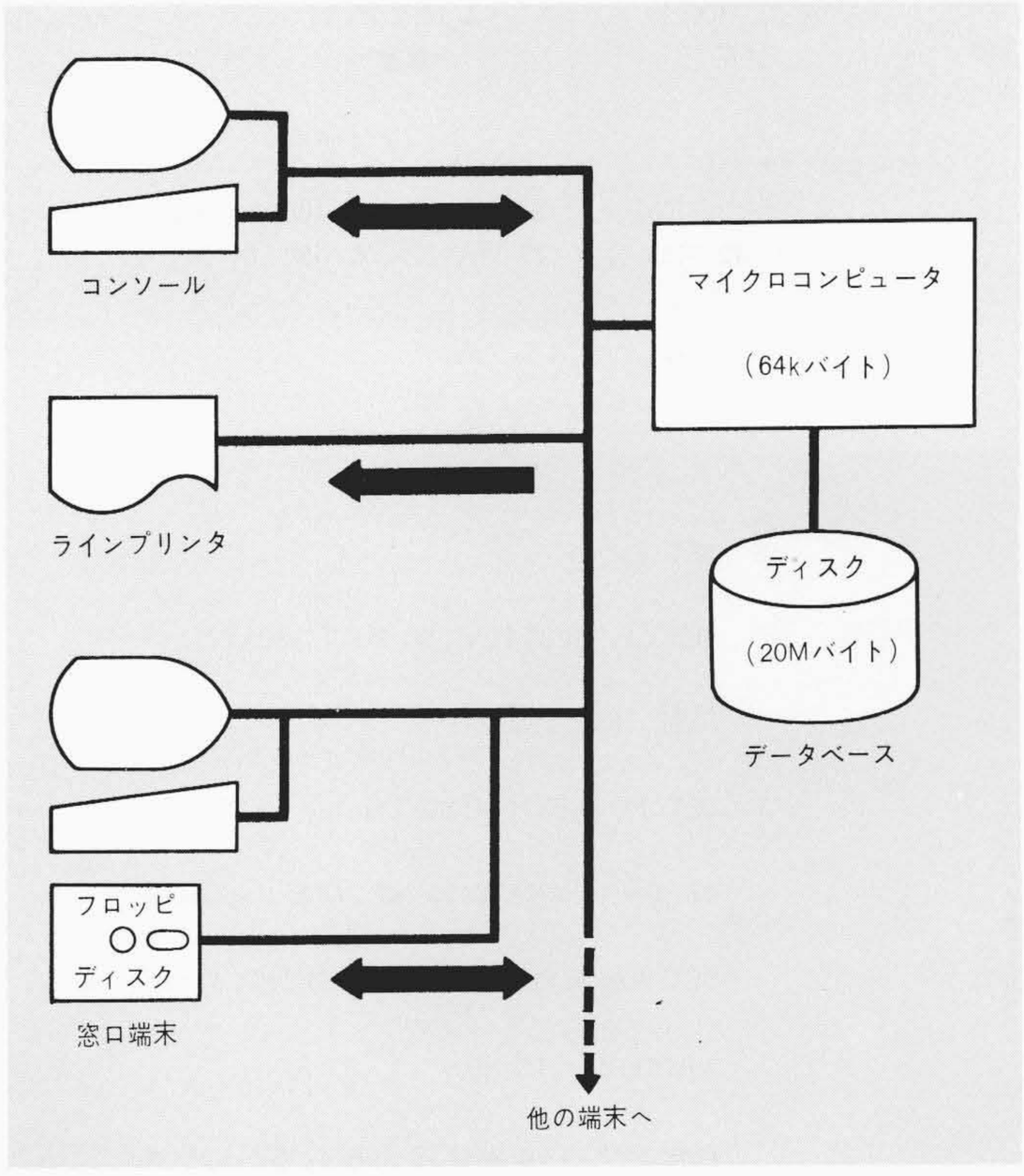

図1　医療事務システム“HIMEC-10”における“Micro MUMPS”
データベースへの入力や検索，演算命令はコンソールから入れる。フロッピディスクはデータベース内容の退避，回復に使う。

* 日立製作所システム開発研究所　** 株式会社日立メディコ柏工場

しているが，マイクロコンピュータでは必ずしも要求されない機能の使い方を制限する一方，実用性の向上と用途拡大のために，以下の機能を追加している。

(1) アセンブリ言語など，MUMPS以外の言語で書かれたルーチンと結合する機能を設け，これにより，他のシステムのファイルにアクセスできるようにした。

(2) MUMPS以外の言語で書かれたプログラムと“Micro MUMPS”を並行して実行できるよう設計した。

(3) 誤操作対策として，入力プログラムの文法誤りの検査は，実行時だけでなく入力時にも行なう。また，誤ってデータベースの内容を破壊した場合も復元が容易なように，データベースの全部，又は一部をフロッピディスクに退避しそれを回復する機能を設けた。

(4) プログラムを会話形で修正する機能や，データベースの内容とディスク上のプログラムの一部，又は全部を出力する機能，及び割り込みキーによる実行の中断など，実用性向上のための各種補助機能を付けた。

2.2 システム構成

“Micro MUMPS”を搭載したときの医療事務システム“HIMEC-10”の構成を**図1**に示す。主記憶容量は64kバイトあり，データベース用のディスクは20Mバイトある。“Micro MUMPS”の命令やデータはコンソールから入力し，結果をCRT(Cathode Ray Tube)やラインプリンタに出力する。データベースの内容の退避，回復のためには，“HIMEC-10”に複数個付く窓口端末のフロッピディスクを使う。

2.3 データ構造

MUMPSでは，主記憶だけに作られる一時的データはローカル変数と呼ぶ変数で表わすが，データベースに保存してあるデータもグローバル変数と呼ぶ変数で表わし，主記憶のデータと同様の扱いをする。いずれも添字付き，又は添字なしの変数として書く。グローバル変数には，変数名の前に^印を付ける。

ローカル変数もグローバル変数も，**図2**に示すように，添字リストによるツリー構造をとる。そこでは，^A(12, 7105)

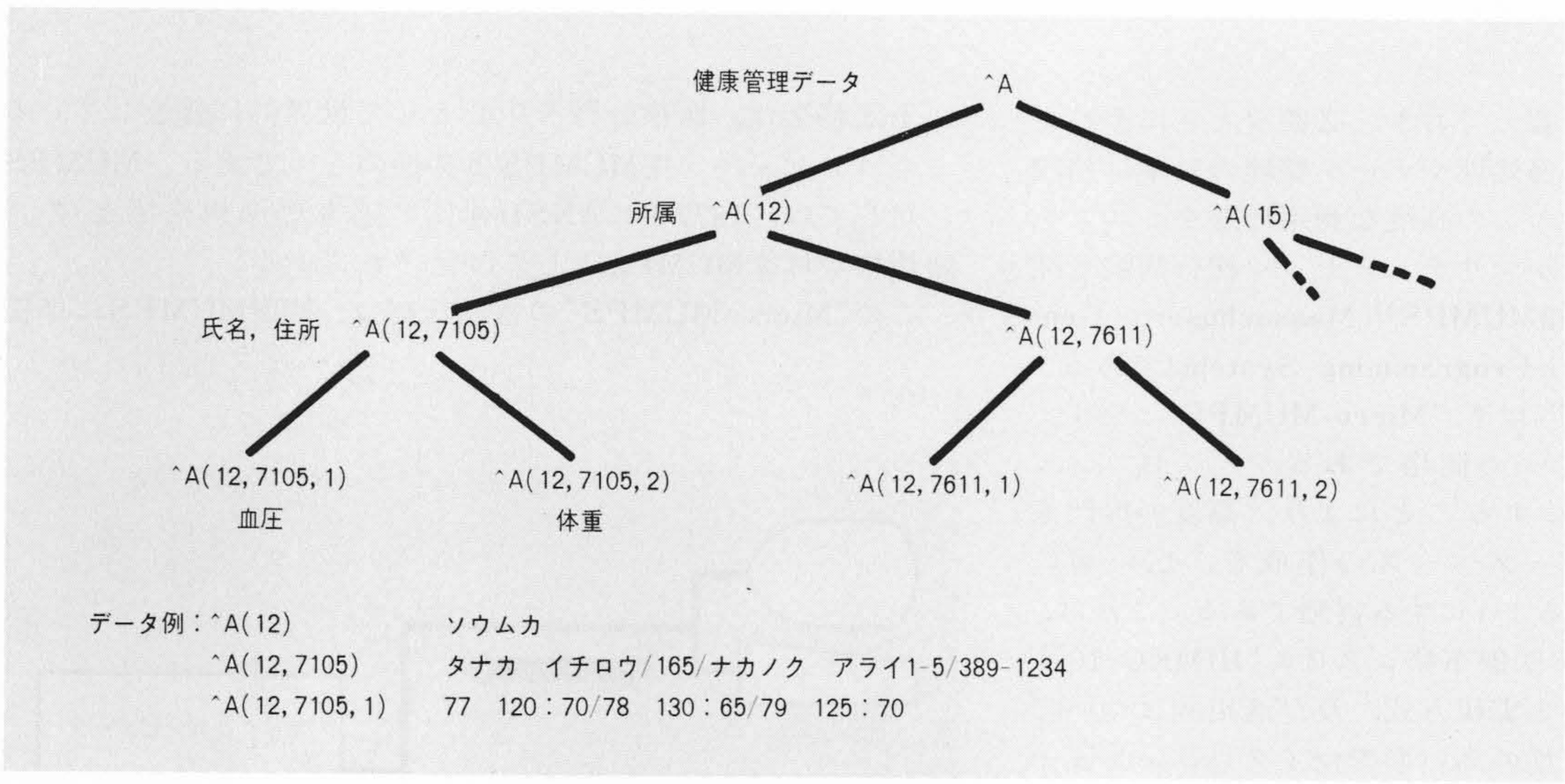

図2 データ構造の例
データベースのデータは，変数で表わし，所属コードや人名コードをそのまま添字として使い，添字リストによるツリー構造に編成する。

```
S   SET I=-1                          該当所属内の先頭の人を探すように添字値を設定。

    READ !,"SECTION",SC QUIT:SC=""    SECTIONと表示して所属コードをSCに読む。
                                      所属コードが与えられなければ終了。
    READ "KEY",KEY                    KEYと表示して検索キーをKEYに読む。
                                      (検索キーは，名前や電話番号など，データ中の任意の単語。)
T   SET I=$NEXT(^A(SC,I))             次の人を探し，その人名コードをIに入れる。
                                      (次の人がいなければI=-1となる。)
    IF I=-1 WRITE !,"NO DATA",! GOTO S  次の人がもういなければ，NO DATAと表示してSへ行く。

    SET W=^A(SC,I) GOTO T:$FIND(W,KEY)=0  その人の情報をWに取り出し，KEYで指定された語を
                                      含むか否か調べる。含んでいなければ次の人を調べる。
    WRITE !!,"ID NO.=",I              含んでいれば，ID NO.=と表示して人名コードを出力する。

    FOR J=1:1:4 WRITE ^AH(J) $PIECE(W,"/",J),!  その人の情報をスラッシュを区切りとして四つに分け，
                                      各行に見出し^AH(J)の内容をつないで出力する。
    GOTO S                            先頭から繰り返す。
```

図3 プログラム例 指定されたデータをデータベースから探し出し，形を整えて表示するプログラムで，COBOLでは54行の記述となる。

の親は^A(12)であるというように，添字の個数の小さいほうが上位に位置する。名前の同じ変数であっても，添字の個数は異なっていてよい。また，添字の値も，1，2，3，…，という連続した値でなく，飛び飛びの値でよいので，人名コードや郵便番号などをそのまま使うことができる。**図2**では，第1添字は所属コードを，第2添字は人名コードを，第3添字は血圧と体重の区別を示す番号としている。

2.4 プログラム例

図3は，**図2**の構造のデータを，所属コードと名前や電話番号など，データに含まれる単語をキーワードとして検索し，形を整えて表示するプログラムの例である。このプログラムを起動するとSECTIONと表示されるので，所属コードとして12を入力し，次にKEYと表示されたときにタナカという名前を与えたとする。すると，**図2**の^A(12, i)のデータを，人名コードiを次々と変えながら調べてゆき，タナカという名前を含むデータを探し出して表示する。該当データがなければNO DATAと表示する。“Micro MUMPS”ではこのプログラムは9行で書けるが，現在最も広く使われている高級言語COBOLで同じ処理を書くと54行にもなる。

表1 “Hitachi Micro MUMPS”の言語機能　本表に示したものの使い方を知れば，データベースの利用や演算，作表ができる簡易言語である。

標準命令	
SET	変数への値の設定
KILL	変数の消去
IF	条件式が真なら実行
ELSE	前の式が偽なら実行
FOR	後続命令を繰返し実行
GOTO	飛び越し
DO	ルーチン呼び出し
QUIT	ルーチン又は繰返しの終了
HANG	時間待ち
BREAK	処理を中断，命令入力待ち
READ	端末からの入力
WRITE	端末への出力
USE	現用装置の指定
HALT	ジョブ終了

拡張命令	
ZFILE	ルーチンの登録
ZLOAD	ルーチンの取出し
ZERASE	ルーチンの一部又は全部の消去
ZMODIFY	ルーチンの修正
ZWRITE	全ローカル変数の出力
ZCALL	他言語ルーチン呼び出し
ZGO	中断した処理の再開
割込みキー	処理を中断，命令入力待ち
ZP	ルーチン名一覧の出力
ZG	グローバル変数一覧の出力
ZW	グローバル変数の内容出力
ZS	データベース内容の退避
ZR	データベース内容の回復

定　数
数値(小数点以上7桁，以下2桁)
文字列(0～200文字)

変　数
ローカル変数(一時的変数)
グローバル変数(データベース変数)

演算子	
+，−，＊，／	(加減乗除)
＼，#	(整除，剰余)
▼，&，！	(論理の否定，積，和)
=，>，<，[	(等，大，小，包含)
]，?，_	(順序，照合，連結)

関　数	
$ASCII	文字のASCIIコード
$CHAR	コードに対する文字
$DATA	変数の定義の有無
$EXTRACT	部分文字列の取出し
$FIND	部分文字列の探索
$JUSTIFY	右づめ
$LENGTH	文字列の長さ
$NEXT	次の添字値
$PIECE	指定記号で区切られた部分文字列
$ZC，$ZN	パラメータ取出し

特殊変数	
$IO	現用装置番号
$STORAGE	可用メモリ量
$TEST	条件式の真偽値
$X，$Y	現桁位置，行位置
$ZAREA	ZCALL パラメータ領域

3 言語仕様

3.1 標準MUMPSから受け継いだ機能

表1に，“Micro MUMPS”の言語機能の概要を示す。

データベース又は主記憶の変数にデータを入れるには，SET命令を使う。データの入れ場所の割付けやデータの長さに応じた入れ場所の拡張，縮小は自動的に行なう。不用になった変数はKILL命令で消せばよい。変数の添字の値は連続しているとは限らないので，添字値の大きさの順でみて次に大きい添字値が何かを求めるには，$NEXT関数を使う。

処理の流れは，分岐，繰返し，ルーチン呼び出しなどの命令で制御する。命令の実行モードには，1行入力するとそれを直ちに実行する直接モードと，何行か入力して一つの「ルーチン」を構成し，それをディスクに保存しておいてから実行する間接モードがある。いずれのモードでも，指定された名前のルーチンは実行時にディスクから探し出してロードするので，実行前にリンクエディトして結合する必要はない。

データを入力するREAD命令には，何を入力すべきかを表示する出力機能がある。WRITE命令で出力形式を整えるには，桁位置や行位置を示す特殊変数$X，$Y及び表示幅を示す$JUSTIFY関数などを使う。

演算機能には，算術演算や大小比較，論理演算のほかに，文字データの分割や連結，探索，パターン照合などの機能がある。これらを使うと，文字データを任意の区切り記号で分割することや，幾つかの文字データを組み合わせて一つの文字データを合成すること，ある単語が文字データに含まれているか否か，含まれていればそれがどこにあるかを調べること，文字データが数字列か英字列か片仮名文字列か，あるいはある一定の型に適合した文字列となっているかを調べること，などが簡単にできる。

3.2 標準MUMPSから拡張した機能

標準MUMPSに比べ追加した言語機能は，**表1**で頭文字がZの命令，及び$Zで始まる関数と特殊変数である。

アセンブリ言語など，MUMPS以外の言語で書かれたルーチンを呼び出すのはZCALL命令である。呼び出されるルーチンは，“Micro MUMPS”の言語プロセッサに手を加えることなく，独立のユーティリティプログラムで追加してゆける。

標準MUMPSには，プログラムの入力や修正，出力の機能及びデータベースに何が入っているかを調べたり，その内容をまとめて出力する機能はない。“Micro MUMPS”では，それらのために，10個の命令を追加した。また，プログラムの文法上の誤りは，実行時だけでなく，入力したプログラムをディスクに保存するときにも行なうようにした。データベースは，個人又は仕事の種類別にクラス分けをし，保全のため，異なるクラスのデータに直接にアクセスすることは禁止した。他のクラスのデータを利用するには，対応するグローバル変数をいったんフロッピディスクに移し，それを自分のクラスのデータベースに組み入れればよい。

4 データ管理手法

データベースのデータを入れるディスクは固定長の「ブロック」に分割し，各グローバル変数がどこに入っているかを容易に探し出せるようインデックスを作る。データとインデックスは別ブロックに入れ，プレフィックスBトリー[5]の手法を応用して各ブロック中の有効データ量のばらつきを少な

くし，どのデータをアクセスするにしても，調べるインデックスブロックの数はわずかで済むようにした。

グローバル変数の間には，添字リストの先頭部分の一致する長さが長いほど互いに強い関係があり，相次いでアクセスされる確率は高いと思われるので，それらをできる限り同じブロックに入れるようにした。また，更新の多いデータに対してはデータ量の増大に伴うブロックのあふれを少なくし，更新の少ないデータに対しては空きスペースを少なくしてブロック中の有効データ量を多くすると，アクセス速度とスペース効率が良くなる。“Micro MUMPS”では，あふれブロックの分割点を制御するパラメータでこの操作を可能にした。

ディスクのアクセス回数を少なくするためには，バッファの有効利用が効果的である。本システムでは，使用頻度の高いブロックほど主記憶にとどまる率が高くなるようにした。また，主記憶に既に入っているデータを参照するときは，インデックスブロックを見ずに直接に参照する。

5 医療システムへの適用

“Micro MUMPS”を用いて，病院での給食管理システムを開発した。病院では，患者の病状に合わせて多種類の食事を用意する必要があり，食事の種類別に栄養価を算出しなければならない。栄養士はこれに忙殺され，患者への栄養指導という本来の任務や，安価で良質な食品の選択がおろそかになり勝ちである。病院給食管理システムは，栄養士の業務の省力化による栄養指導の強化と，材料購入費の節減を目的とするものである(**図4**)。

このシステムでは，栄養価計算や材料の管理と手配，品目ごとの調理件数の算出，配膳計画の作成などを行なう。そのために，患者情報や食品別栄養価，食種情報(糖尿食，腎臓食など特殊な患者の給食ごとに適合する主食，副食や制約条件に関する情報)，献立内容，食糧在庫などの情報を入れたデータベースをもつ。

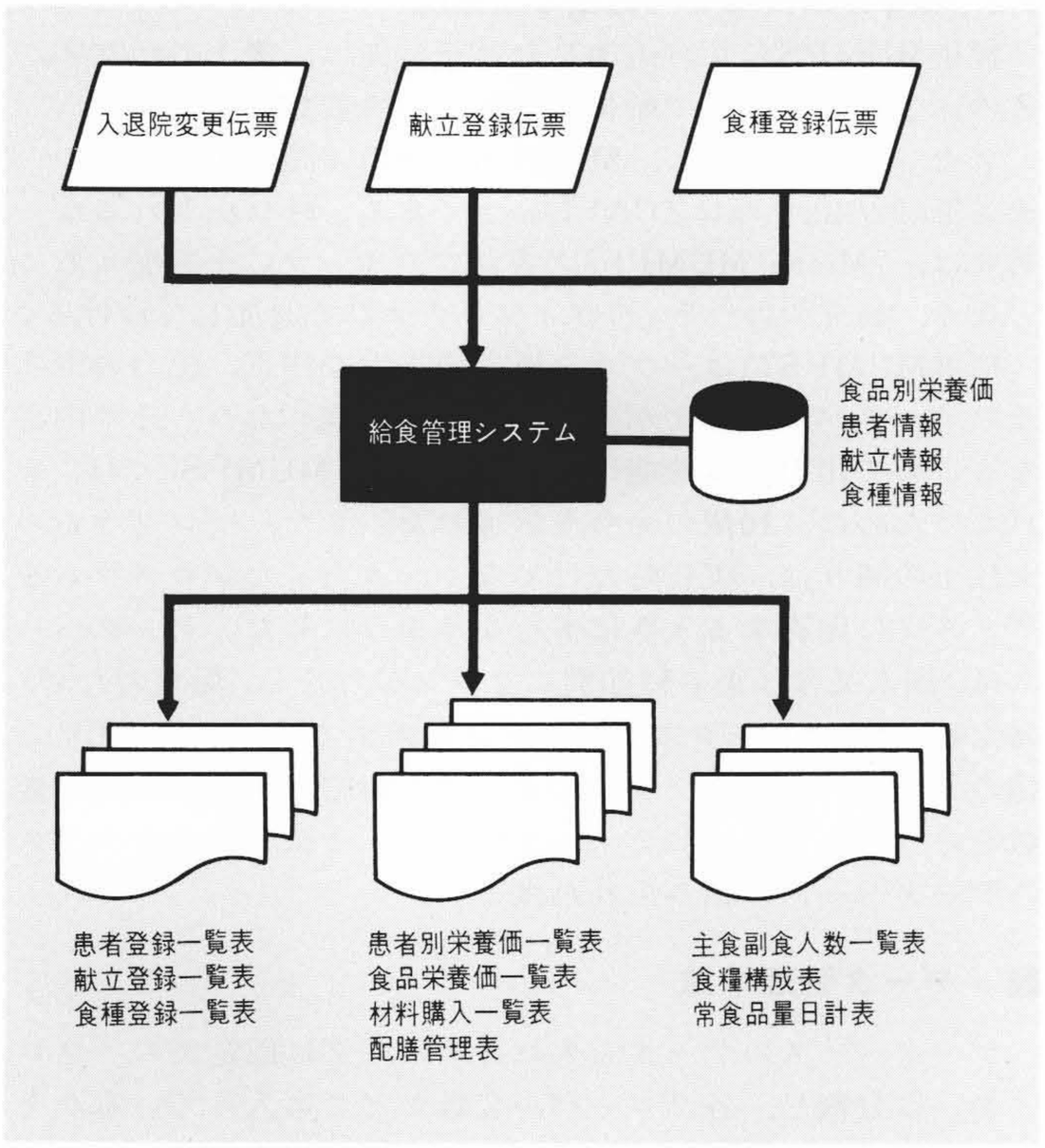

図4 病院給食管理システム “Micro MUMPS”を用いて，病院での患者給食の栄養価計算，配膳管理，材料管理などを行なう給食管理システムを開発した。

表2 “Hitachi Micro MUMPS”の適用例 Micro MUMPSでは，アセンブリ言語に比べ，ステップ数は約$\frac{1}{30}$，工数は$\frac{1}{8}$となった。COBOLに比べてもステップ数は$\frac{1}{6}$となる。

プログラム	言語	ステップ数	作成工数	処理時間
給食管理システム	アセンブリ言語	16,500	8人・月	15.3秒/表 (フロッピディスク使用)
	“Micro MUMPS”	520	1人・月	19.8秒/表
個人情報検索	COBOL	54	7時間	—
	“Micro MUMPS”	9	1時間	—

このプログラムを“Micro MUMPS”と従来どおりのアセンブリ言語の二つの方法で開発し，比較した。**表2**に示すように，“Micro MUMPS”では，ステップ数が約$\frac{1}{30}$に減り，作成工数が$\frac{1}{8}$に減った。処理時間については，磁気ディスクを使った“Micro MUMPS”のプログラムは，フロッピディスクを使ったアセンブリ言語のプログラムの約30%増しである。このことは，“Micro MUMPS”を使うと，アセンブリ言語に比べ，同等の性能を維持しながらプログラミング工数を一桁近く下げるための代償が，フロッピディスクの代わりに磁気ディスクを使うことで済むことを示している。

6 結 言

データベースを，大形コンピュータとコンピュータ専門家の助けを借りなくても，マイクロコンピュータを使ってそのユーザー自身が容易に作成し，利用できるようにする会話形簡易言語“Hitachi Micro MUMPS”を開発し，医療事務システム“HIMEC-10”にのせた。その用途は，医療事務や病歴管理などの医療分野に限らず，一般に，中小規模の事務処理システムや生産管理システム，あるいは個人又は小グループで保有するデータの整理，検索などに有効に使用できるものである。

最後に，本言語につき貴重な御意見をいただいた京都大学医学部附属病院助教授・平川顯名医学博士，マンプスシステム研究所所長・若井一朗医学博士，同所嶋 芳成研究員及び立教大学理学部助教授・島内剛一理学博士，並びにMUMPSについて御教示をいただいた自衛隊中央病院小児科部長・春名英彦医学博士，同病院電計準備室野方 覚3等陸佐及び同室坂本巧二郎氏，また本システムの開発に御協力いただいた関係各位に対し，それぞれ深く感謝する。

参考文献

1) J. O'Neil, Editor：MUMPS Language Standard, NBS Hand Book 118(1976/1)
2) 渡辺，外：マイクロコンピュータ用MUMPSの開発，第6回日本MUG学術大会抄録（1979-9）
3) 隈，外：分散処理システム構成による病院向け医事システム“HIMEC-10”，日立評論，**61**，299～300（昭54-4）
4) 吉村，外：マイクロコンピュータ用高級言語PL/Hシステム，日立評論，**61**，305～310（昭54-4）
5) R. Bayer and K. Unterauer：Prefix B-trees, ACM Transactions on Data Base Systems, **2**，1，11-26 (Mar. 1977)