テーブル・ドリブン型データベース・ジェネレータ†

高藤政雄†† 浜田亘曼††
平沢宏太郎†† 奥田健三†††

テーブル・ドリブン型データベース・ジェネレータ (DG) について論じる．DGはソフトウェアの標準化を進めるための道具であり，
1）プラントの各入出力点の現在値，各点の制御ゲイン，上下限値などのプラント情報を格納しているプラント・テーブル
2）シーケンス制御などの手続きプログラムを中間語の形で記憶している中間語テーブル
を自動作成するものである．この DG は入力データカードの処理方法により
（1） 手続き実行型 DG
（2） テーブル・ドリブン型 DG
の2つに分類されるが，近年，ソフトウェアコストに占める保守コストが開発コストをはるかに上回ることから，保守性の点で優れたテーブル・ドリブン型 DG を立案した．
本DGにおいては，データの入力処理，加工処理，出力処理をできるだけ標準化するとともに，ユーザテーブルの修正，データの追加を容易にすることはもちろんのこと，DG そのものの保守性の向上を図っている．そのために，本 DG はユーザテーブルおよび DG システムテーブルをブート・ストラッピング手法により，同一処理で作成している．
本稿では，上記テーブル・ドリブン型 DG の処理概要およびその特徴について述べている．

1. まえがき

近年，ソフトウェア開発における保守性をも含めた生産性の向上が重要な課題となっている．この生産性を向上させるには，システム設計から，プログラム製作，デバッグ，テスト，保守まで一貫した形での生産性向上を図ることが必要であるが，筆者らは，このうち特に保守のしやすさをも考慮したシステム設計，プログラム製作に重点をおいた生産性向上手法について検討した．

この分野での生産性を向上させる手法として，2つの手法が考えられる．ひとつはストラクチャード・プログラミング技法，トップダウン・プログラミング技法等をとり入れた設計法[1),2)]ならびに高級言語[3),4)]による高品質なプログラムの製作手法であり，他のひとつは応用分野のノウハウをもとにして機能の標準化を行い，その機能に対応して作成した問題向き言語[5)~8)] (Problem Oriented Language, POL) によるプログラム製作手法である．

本稿では上記2つの手法のうち，POL によるプログラム製作をサポートする一方法について述べる．一般に POL としては，機能の標準化によりデータと手続きを分離し，適用システムの規模や構成，処理内容の違い等をデータの違いとしてテーブル（プラントテーブルと称する単なるデータの集合）に設定するもの，シーケンス制御等のためのオンサイトでの修正が比較的簡単に行える中間語プログラムをテーブル（中間語テーブルと称する手続きを意味するデータの集合）に設定するもの等が考えられる[9)]．これらのテーブルは POL インタプリタにより解釈実行される．

これらの POL を実現するためにはテーブルを自動作成するデータベース・ジェネレータ（以下 DG と略す）が必要である．この DG は入力データカードの処理方法により下記の2つに分類される．

（1） 手続き実行型 DG
（2） テーブル・ドリブン型 DG

上記(1)は，入力データカードを，手続き向き言語で記述された処理方法に従って処理するもので，(2)はデータとしてテーブルに設定された処理方法に従って処理するものである．

従来の問題向き言語システム[5)~8)]も，DG の機能が必要となるが，その処理方式は明らかにされておらず，特にテーブル・ドリブン型DGについては，発表

† Table Driven Database Generator by MASAO TAKATO, NOBUHIRO HAMADA, KOTARO HIRASAWA (Hitachi Research Laboratory, Hitachi Ltd.) and KENZO OKUDA (College of Technology, Utsunomiya University).
†† (株)日立製作所日立研究所
††† 宇都宮大学工学部

された例はない．これは，従来の問題向き言語システムにおけるDGが，DG内部の処理機能の標準化を考慮していなかったことおよび，汎用DGの場合[10]にも，従来はDGの汎用性あるいは処理性に重点が置かれていたことから，DGは手続き実行型になっていたと考えられる．しかし，近年，ソフトウェアコストにおける保守コストが開発コストをはるかに上回る[11]ことを考えると，DGの汎用性，処理性よりもむしろ，DGが作成するユーザテーブルのデータの修正あるいは追加の容易さはもちろんのこと，各応用分野（電力システム，化学システムなど）用DGそのものの保守性の向上，すなわち作成するユーザテーブルのデータ構造に変更が生じた場合に，DGの処理内容の変更が容易にできることが重要である．

そこで筆者等は，DGのデータの入力処理，加工処理，出力処理を標準化するとともに，その処理方法をテーブルに記憶することにより，DGの保守性の点で優れたテーブル・ドリブン型DG[12]を立案した．以下，その処理概要，特徴について述べる．

2. DGの概要

DGは，**図1**に示すように，エンドユーザが作成したFIF (Fill In the Form：空欄穴埋め) 形式のデータあるいは自然語風なコメント形式のデータを読み込んで，ユーザ固有のプラント・データベース（ユーザテーブルであるプラント・テーブル，中間語テーブルを意味する）を作成する機能を有し，以下の処理を行う．

（1） 入力データカードを認識し，そのデータカードの入力フォーマットに従って，データを抽出する．

（2） 抽出したデータに対して，上下限チェックなどのデータの合理性チェックおよび，スケール変換などの加工処理を行う．

（3） 合理性チェック，加工処理後のデータをユーザ指定のフォーマットでプラント・データベースへ設定する．

ここで，DGのプラント・データベースへのデータの設定方法について考えてみると

（1） データのオブジェクト・モジュールを介する方法

（2） データを直接ユーザ固有のプラント・データベースへ設定する方法

の2つの方法がある．

（1）の方法は，DGの出力として，プラント・データベースにデータを設定するための宣言文 (FORTRAN言語のDATA文，アセンブラ言語の定数定義命令など) を出力し，それをプロセッサ（コンパイラ，アセンブラなど）で処理し，データのオブジェクト・モジュールを作成し，リンケージ・エディタ，ローダを用いて，ユーザ固有のプラント・データベースを作成するものである．この方法は，DGそのものをオンサイトの計算機に持ち込めないような小規模システムのプラント・データベースを作成するために必要である．一方，（2）の方法は，主メモリあるいは二次メモリ上のプラント・データベースへデータを直接書込むことにより，ユーザ固有のデータベースを作成するもので，アセンブラ，リンケージ・エディタ，ローダなどを必要としないので，（1）の方法に比較して，（a）DGの処理性が良い，（b）プラント・データベースの作成および修正が容易である，（c）入力データカードの処理枚数に制限がない，などの点で有利である．

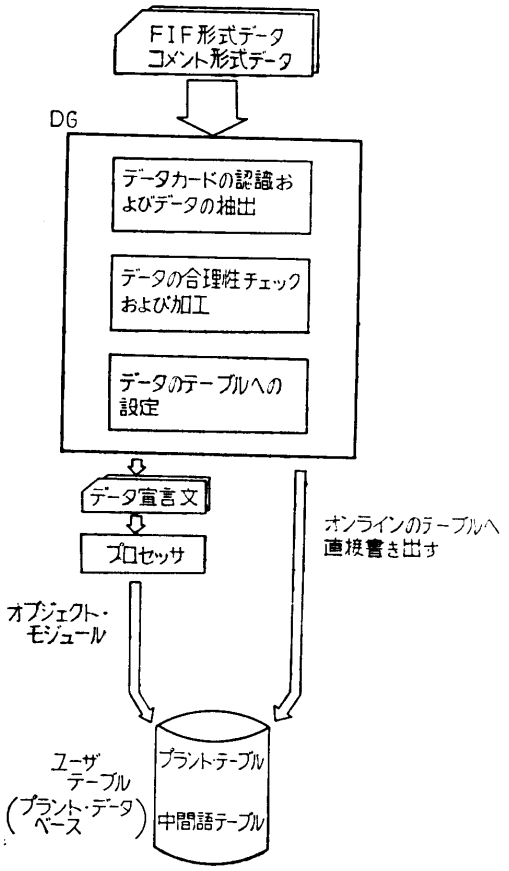


図1 データベース・ジェネレータの処理概要
Fig. 1 Processing of data-base generator.

以上述べたDGを実現するために必要となる機能を

表1 データベース・ジェネレータの機能

Table 1 Function of database generator.

DGで必要とする機能			機能概要および有効性（必要性）
基本機能	入力処理機能	コメント形式データ処理機能	自然語風なステートメントを処理する機能で中間語テーブル作成に有効
		FIF形式データ処理機能	カラム切りの空欄穴埋め式データを処理する機能で，入力データ項目の多いデータ処理に有効
		定数名の登録参照機能	入力データカード中に現われるある意味を持った文字列を定数名としてDG内部のテーブルに登録し，別の入力データカード中で参照する機能で，データの記述性およびドキュメント性を向上するのに有効
	内部処理機能	演算機能	加減算，乗除算，論理演算，シフト，文字列の連結などを行う機能で，入力データの加工に必要
		制御機能	判定，飛び越しを行う機能で，入力データの合理性チェックなどに必要
		手続き参照機能	ほかの処理手続きを参照する機能で，DGの容量削減あるいはDGの標準加工処理の拡張のために必要
	出力処理機能	データ宣言文出力機能	アセンブラの定数定義命令などのデータ宣言文を出力する機能で，データのオブジェクト・モジュールを介してユーザテーブルを作成するために必要
		テーブル直接書込み機能	オンラインのユーザテーブルへデータを直接書込む機能でDGの処理性の向上，テーブルの作成および修正の容易さ，入力データカード処理枚数の増加などを図るのに有効
		ドキュメント出力機能	ドキュメントとしてのリストを出力する機能でDGのデバックあるいは保守のために有効
	補助処理機能	バッファ管理機能	二次メモリ上のユーザテーブルへのアクセスに際して使用する主メモリ上のバッファの使用方法を規定する機能で，DGの処理性の向上のために有効
応用分野別機能		データカードの認識およびデータの抽出機能，入力データの合理性チェック加工処理機能，テーブルへのデータ設定機能	電力，鉄鋼，生産管理，化学などの各応用分野で固有に必要となる機能で，各応用分野に適したプラント・データベースを作成するために必要

表1に示す．本表において基本機能とは，各応用分野別DGを作成するために共通的に使用される機能で，応用分野別機能とは，応用分野固有の処理を実現するための機能である．なお，応用分野別機能のうち，データカードの認識およびデータ抽出機能は，DGの基本機能である入力処理機能の中から1つの機能を，テーブルへのデータ設定機能は，出力機能の中から1つの機能を応用分野別DG作成者が選択することにより実現される．また，入力データの合理性チェック，加工処理機能は，基本機能の内部処理機能である演算機能，制御機能および手続き参照機能を応用分野別DG作成者が必要に応じて組合せることにより実現される．

3. テーブル・ドリブン型 DG の概要

ここでは，テーブル・ドリブン型DGの開発方針およびその処理概要について述べる．

3.1 開発方針

テーブル・ドリブン型DGの開発方針を要約すると次のようになる．

（1） DGの保守性の向上

計算機制御システムの分野では，応用分野によってソフトウェアの標準化の程度に差がある．標準化が徐徐に進行する分野においては，状態監視，制御などで必要とするテーブルのデータ構造に変更が生ずる可能性が大きい．このために，データ構造の変更に対して容易に対処できることが必要である．

（2） テーブル作成の自由度の向上

プラント・データベースの作成にあたっては，システム規模によりDGをオンサイトに持ち込めない場合や作業の進捗状況により，テーブルの先頭から順序よく作成できない場合などがある．またテーブルも単純なレコードの繰返しからなるテーブルのみでなく，インデックスポインタ付のテーブルなど各種のデータ構造を持つテーブルが必要となる．

（3） テーブルのデータの追加，修正の簡単化

プロセス制御などでは，プラントの増設や，仕様変更などが起こる可能性が大きく，プラント・データベースのデータ項目の追加や，データそのものの修正が生じやすい．そのため，これらデータの追加および修正が容易にできることが必要である．

（4） DGの加工処理の標準化

応用分野別機能は，先に述べたように応用分野に固有な処理を実現するものであるが，このうちデータカードの認識およびデータ抽出機能，テーブルへのデータ設定機能は比較的標準化が簡単である．一方，入力データの合理性チェック，加工処理は，応用分野ごとさらには作成するテーブルごとに異なる可能性が大きく，標準化が難しい．しかし，この入力データの合理性チェック，加工処理をできるだけ標準化することが，応用分野ごとさらにはテーブルごとに作成する部分を減少させ，DGの生産性の向上に役立つと考えられる．

（5） DGの処理性の向上およびコンパクト化

応用分野，たとえば電力システムなどでは，プラント・データベースを作成するための入力データのカード枚数が20～30万枚にもなる．そのため，当然のことながら，DGの処理性の向上が必要となる．また，できるだけオンサイトにDGを持ち込むためには，コンパクトなDGが必要となる．

以上の方針に基づき，テーブル・ドリブン型DGを

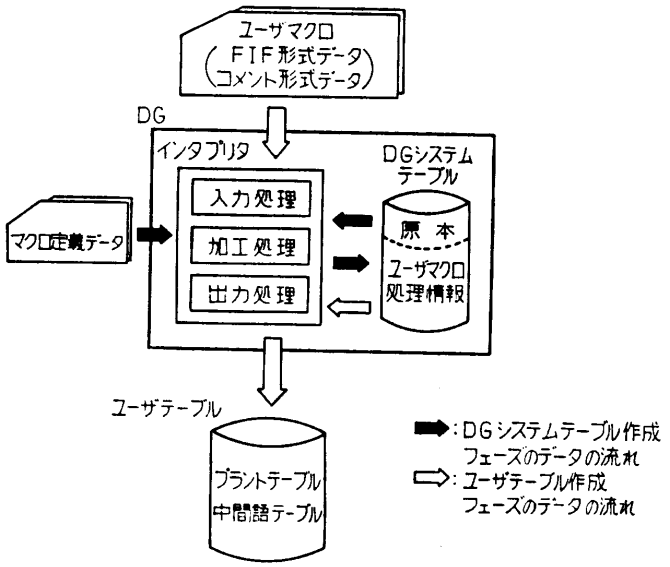


図 2 テーブル・ドリブン型データベース・ジェネレータの処理概要
Fig. 2 Processing of table-driven database generator.

立案した.

3.2 テーブル・ドリブン型 DG の処理概要

テーブル・ドリブン型DGは，**図2**に示すように入力データカードの処理方法を記憶しているDGシステムテーブルとそのテーブルを参照して入力データカードを処理するインタプリタより構成される．このテーブル・ドリブン型DGは，第2章で述べた入力，加工，出力の各処理をDGシステムテーブルを参照して行うもので，その処理は大きく2つのフェーズに分けられる．

(1) DG システムテーブル作成フェーズ

本フェーズは，ユーザテーブルを作成するための準備フェーズであり，図2の➡で示される処理を行う．すなわち，インタプリタはマクロ定義データを読み込み，DG システムテーブルの原本を参照して，ユーザマクロ処理情報を作成する．

ここで，マクロ定義データは，FIF 形式データおよびコメント形式データであるユーザマクロの処理方法を記述したものであり，DG システムテーブル原本は，上記マクロ定義データの処理方法を示す情報で，本 DG 作成時に DG システムの固定データとして DG システムテーブルの先頭にあらかじめ設定される．また，マクロとは，ユーザテーブルを作成するために DG に入力される一群のデータセットに対して付けられる識別子で，ユーザが任意に付けることができる．

(2) ユーザテーブル作成フェーズ

本フェーズは，ユーザテーブルを作成するフェーズで，図2の ⇨ で示される処理を行う．ユーザは，上記DGシステムテーブル作成フェーズで，ユーザマクロ処理情報を DG システムテーブルに設定した後で，ユーザテーブルを作成するためのデータ群であるユーザマクロ群を入力する．インタプリタはこのユーザマクロ群を上記 (1) で設定したユーザマクロ処理情報を参照しながら処理し，ユーザテーブルを作成する．

このユーザマクロ処理情報の参照方法は，(1)のフェーズにおけるDGシステムテーブル原本の参照方法と同じである．すなわち，DG システムテーブル原本とユーザマクロ処理情報のデータ構造は同じであり，本 DG は，以下に示す手順によるブート・ストラッピング手法を用いている．

(a) まず，本DGの一構成要素であるインタプリタはマクロ定義データを読み込み，DG システムテーブルの原本を参照して，ユーザマクロ処理情報を作成する．

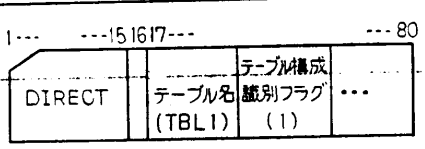
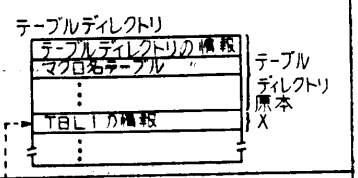
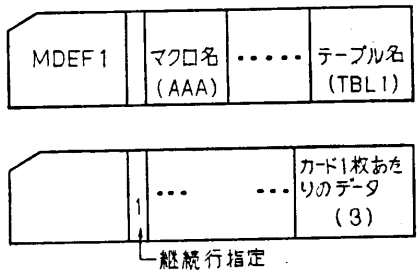
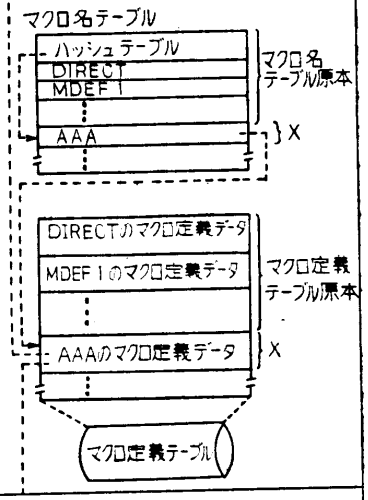
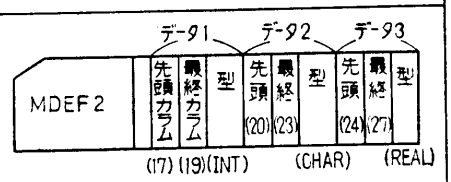
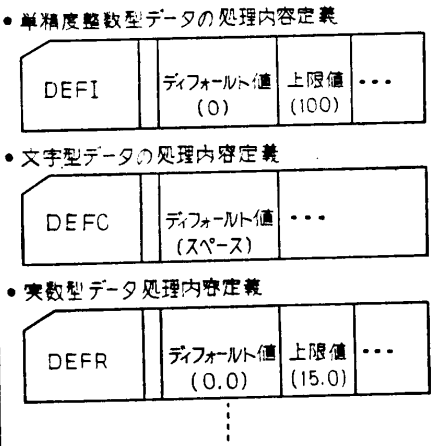
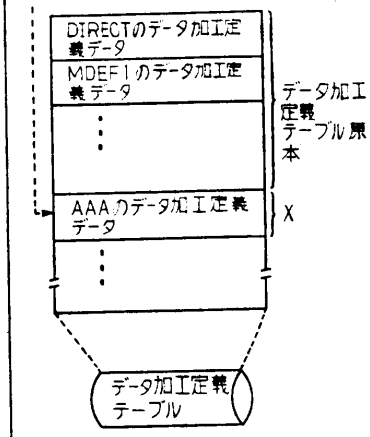
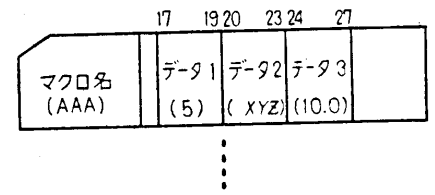
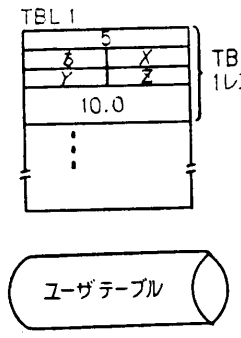
(b) 次に，上記インタプリタはユーザマクロを読み込み，(a)で作成したユーザマクロ処理情報を参照して，ユーザテーブルを作成する．

以上述べたように，DG システムテーブルを作成するための最小限のデータ(ここでは原本と称している)を基に，最終的にDGのオブジェクトであるユーザテーブルを作成することになる．

通常インタプリタを有するシステムは，マクロの処理方法を記述するマクロ定義命令群を中間語に変換するトランスレータと，その中間語を解釈実行するインタプリタとから構成されるのが普通であるが，本 DG では，ブート・ストラッピング手法により上記トランスレータの機能をインタプリタで兼ねることにより，DG のコンパクト化と DG の保守性の向上を図っている．

以上述べたテーブル・ドリブン型DGによるプラント・データベース作成手順を**表2**に示す．同表において，処理内容とは応用分野別DG作成者およびユーザが行うべき処理を，入力とはDGへの入力データを，出力とはDGが作成するテーブルをそれぞれ意味している．表2に示すように，DG システムテーブルは最初テーブルディレクトリ，マクロ名テーブル等の原本のみから成っているが，(1)のDGシステムテーブル

表 2 プラント・データベース作成手順　　Table 2 Plant database generation procedure

処理順序		処理内容	入力（マクロ定義データまたはユーザマクロ）	出力（DGシステムテーブルまたはユーザテーブル）
(1) DGシステムテーブル作成フェーズ	1	DGで作成するテーブルのテーブル情報の登録		
	2	上記テーブルを作成するためのユーザマクロの登録		
	3	上記ユーザマクロのカラム切り情報の登録		
	4	上記ユーザマクロの各データの加工処理情報の登録		
(2) ユーザテーブル作成フェーズ	5	ユーザテーブルの作成		

□：主メモリ上テーブル　　⬭：二次メモリ上テーブル

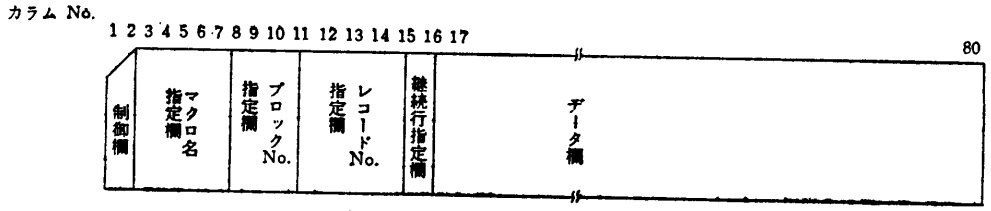


図 3 データカードフォーマット
Fig. 3 Data card format.

作成フェーズにおいてユーザマクロ AAA のマクロ定義データをシステムマクロ群 (DIRECT, MDEF 1, MDEF 2, DEFI 等) を用いて入力することにより，ユーザマクロ AAA に関するユーザマクロ処理情報X が作成される．このユーザマクロ処理情報と原本とのデータ構造が等しいことは前述の通りである．最後に (2)のユーザテーブル作成フェーズにおいてユーザマクロ AAA を入力することにより，ユーザテーブル TBL 1 の1レコードが作成される．なお，入力データカードフォーマットの詳細は **4.2** の図3，表3に示すが，表2の入力カードのフォーマットでは，ブロック No., レコード No. は省略している．

4. テーブル・ドリブン型 DG の入力

ここでは，本 DG の入力となるデータカード，データの入力仕様および入力例について述べる．

4.1 入力データカード

本 DG の入力データカードのカードフォーマットを**図3**に，そのカードフォーマットの各カラムの説明を**表3**に示す．図3に示すカードフォーマットは，FIF 形式データカードのフォーマットを示しており，コメント形式データカードのフォーマットについてはここでは言及しない．図3に示すカードフォーマットの特徴は下記の通りである．

(1) 各データカードに，マクロ名指定欄，ブロック No. 指定欄，レコード No. 指定欄がある．

(2) 従来のアセンブラ言語，FORTRAN 言語などのフォーマットと異なり，通常カードの整理のための記号および通し番号を記入する ID シーケンス欄がない．

上記(1)は，入力データカードの認識および入力順序のチェックあるいは入力データカードにより作成する1レコード分のテーブルの出力アドレスを求めるために必要である．

(2)は，ID シーケンス欄を第2～16カラムのマクロ名，ブロック No., レコード No. および継続行の各指定欄で兼ねることにより，データ欄を広くし，できるだけ多くのデータを1枚のカードに納めることが可能なほかに，カード枚数および FIF フォーマット用紙枚数を削減することができる．

表 3 データカードフォーマットの意味
Table 3 Meaning of data card format.

項番	カラム	名称	内容
1	1	制御欄	本カラムが下記のとき，それぞれの意味を持つ @：コントロールカードである C：コメントカードである b：データカードである
2	2～7	マクロ名指定欄	マクロ名を指定する．
3	8～11	ブロック No. 指定欄	各マクロで作成するテーブルのブロック番号を指定する． 1ブロックからなるテーブルはブランクとする．
4	12～15	レコード No. 指定欄	各マクロで作成するテーブルの各ブロックの先頭からのレコード番号を指定する．
5	16	継続行指定欄	マクロに継続行がある場合，2枚目のカード以降 1, 2, 3, 4 と通し番号で継続行を指定する． 継続行は4行以内とする．
6	17～80	データ欄	マクロで処理するデータを記入する．

4.2 入力データ

入力データとして下記特徴がある．

(1) 入力データの種類およびデータの型は，制御用で用いられること，DG の処理性の向上およびコンパクト化を図るため，(a)単精度整数型，(b)倍精度整数型，(c)ビット型，(d) 16進型，(e)実数型，(f)文字型および(g)リスト型 (単精度整理型) の定数のみに限定する．

(2) 入力データとして(a)単精度整数型の定数名 (定数に対して与えたシンボリックな名称) および (b)アドレス定数 (記憶装置上のテーブルの先頭アドレスを示す定数) を許す．これは，(a)単なる数値データの代りに，データの内容を表わす定数名を用いることにより，データの記入誤りを減少させ，かつドキュメント性の向上が図れる，(b)ユーザテーブルのバッファ管理をユーザが行いたい場合にはテーブルの先頭アドレスなどのデータが必要となる，ことによる．

(3) ブランクのデータカラムに対して，いくつかの処理を指示できる (表5の項番1参照)．

4.3 入 力 例

図4に本DGの入力データカードおよび出力結果の具体例を示す．この例は，表4に示す項番1の単一レコードからなるエリア定義テーブル（ユーザ自身がユーザテーブルを管理するために各テーブルの管理情報を保持するテーブル）を作成するための入力データカードフォーマットおよびその出力結果としてのエリア定義テーブルの1レコードを示したもので，(番号)はカード内データとテーブル内データとの対応関係を示している．また，このデータカードのマクロ名は AREADF で，ブロック No. は無く，レコード No. はⓑのエリア No. であることを示しており，このエリア No. は，エリア定義テーブル中のレコード No, に対応している．

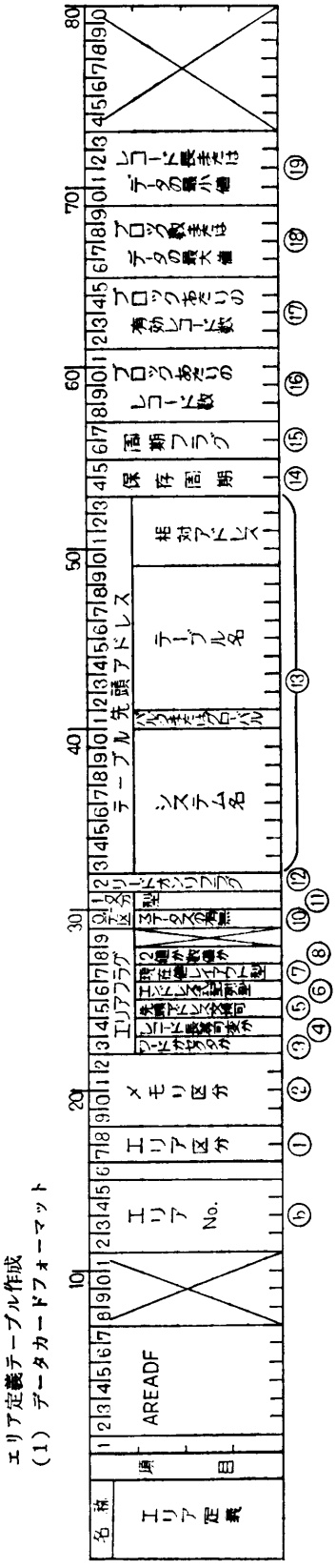


図4 本 DG への入力データカードおよび出力結果の例
Fig. 4 Example of input data card and output of DG

5. テーブル・ドリブン型 DG の特徴

本DGの機能的特徴を3.1の開発方針に対応させて述べる．

（1） ブート・ストラッピング手法を採用しDGの保守性の向上を図る．

作成するプラント・データベースのデータ構造に変更が生じると，一般に手続き実行型DGの場合は，入力データの処理方法を記述した手続きの修正を行った後，再コンパイル，再ローディングが必要になる．本DGの場合は，入力データの処理方法をデータとして記憶しているため，プラント・データベースのデータ構造の変更を単なるDGシステムテーブル作成用データの修正で処理でき，しかも，その修正はプラント・データベースの修正と同一の方法で処理できるので，コンパイラ，ローダ等が不要となり，DG の保守が簡単である．

（2） テーブル作成の自由度を向上させる

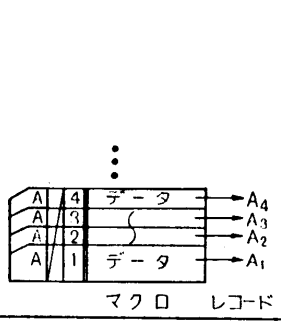
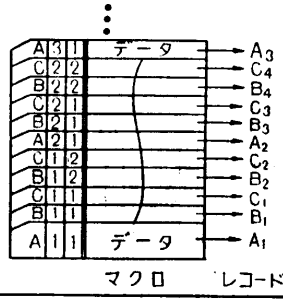
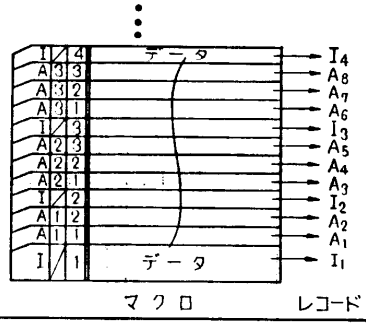
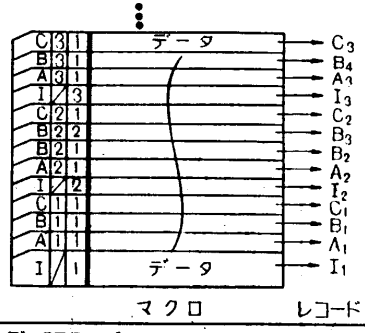
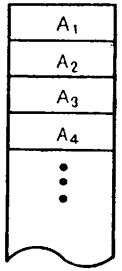
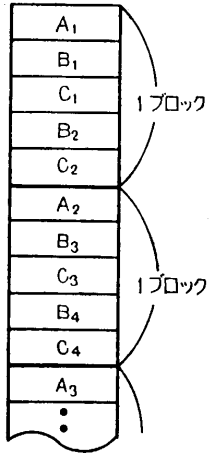
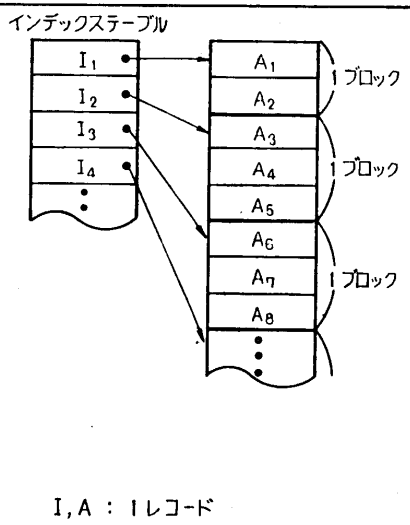
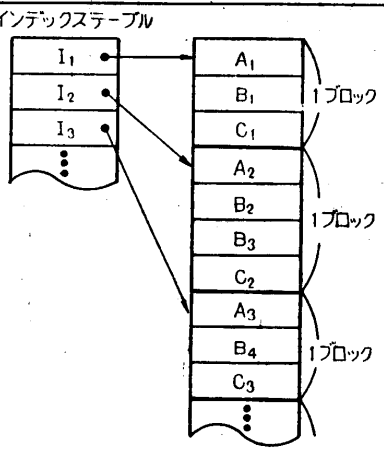
本DGの適用範囲を広げるために，ユーザテーブルの出力方法として

（a） データのオブジェクト・モジュールを介する方法

（b） データを直接オンラインのユーザ

表 4 マクロとそれに対応するテーブルのデータ構造の例

Table 4 Examples of table data structure corresponding with macros.

項番	1	2	3	4
テーブル構成要素	単一レコード	固定長ブロック	単一レコードからなる可変長ブロック	複数レコードからなる可変長ブロック
マクロと対応するレコード				
作成されるテーブルのデータ構造				

テーブルへ書込む方法
をサポートし，一つのテーブルの作成方法として
（c） ランダム作成
（d） ブロック内シーケンシャル作成
（e） シーケンシャル作成
をサポートしている．上記(c)，(d)，(e)はユーザが，テーブル作成において，データカードの入力順序の自由度を重視するか，あるいは処理性を重視するかにより選択できる．また，本DGで作成可能なテーブルのデータ構造として，**表4**に示す4種類のテーブルがある．

項番1の単一レコードからなるテーブルは，Aという1つのマクロ名で認識されるデータ群により作成されていることを示している．このマクロは図3に示すデータカードフォーマットにおいて，ブロックNo.指定欄はブランクで，レコードNo.指定欄が1～mのデータカードとなる．

項番2の固定長ブロックからなるテーブルは，A，B，Cの3種のマクロ名で認識されるデータ群により作成されていることを示している．たとえば，図のレコード B_4 はマクロ名=B，ブロックNo.=2，レコードNo.=2で示されるデータ群により作成される．

項番3の単一レコードからなる可変長ブロックで構成されるテーブルは，AおよびIのマクロ名で認識されるデータ群により作成されていることを示している．たとえば，インデックステーブルの I_1 レコード内のレコード A_1 へのポインタは，第1ブロックの A_1 レコードを作成するデータカードの直前に I_1 レコードを作成するデータカードを入力することにより，自動的にセットされる．

項番4の複数レコードからなる可変長ブロックで構成されるテーブルは，A，B，C，Iの4種のマクロ名により認識されるデータ群により作成されることを示している．インデックステーブルは項番3と同様にして作成される．

（3） テーブルのデータの追加，修正は，追加あるいは修正指示用のコントロールカードと，追加あるいは修正すべきデータのみで処理でき，きわめて簡単である．

（4） DGの加工処理のうち，標準化可能な部分は標準加工処理とし，応用分野のアーキテクチャに依存する部分などの標準化不可能な部分は加工ルーチンによる処理とし，ユーザテーブル作成の容易さの向上およびDGの処理の標準化を進める．

表5 標準加工処理の機能
Table 5 Function of standardized data processing.

項番	処理項目	機能概要
1	ディフォールト処理	指定されたカラムがブランクのとき指示内容により下記処理を行う (a) あらかじめ指定した標準値に対して下記項番2,3,4の処理を行う (b) エラーとみなす (c) そのデータの処理をスキップする (d) そのデータ以降のカラムのデータの処理をスキップする
2	チェック処理	入力データの上下限チェックを行う
3	スケール変換処理	入力データ (x) の一次変換 ($ax+b$) を行う
4	フォーマット変換処理	整数型定数，16進型定数データのビットデータへの変換，文字型定数データのパック，アンパック処理を行う
5	インデックス処理	インデックステーブルへインデックスポインタを自動設定する
6	重複処理	同一カラムの入力データに対して，複数の標準加工処理あるいは加工ルーチン処理を行う
7	加工ルーチンリンケージ処理	上記1～3の処理後のデータを指定されたリンケージエリア（ローカルリンケージエリアあるいはグローバルリンケージエリア）へ設定し，指定された加工ルーチンへリンクする

標準加工処理として，**表5**に示す7種の処理をサポートしているが，これらは，基本的な標準加工処理であり，さらに標準加工処理として追加すべき処理が生じた場合には，それを加工ルーチンとして登録し，本DGの加工ルーチンリンケージ処理機能を用いて，DGの加工処理の標準処理部分を徐々に拡張していくことが本DGの基本的考え方である．

なお，加工ルーチンリンケージ処理は，上記標準加工処理で処理できないデータの合理性チェック，加工処理をリンケージエリアを介して（高級言語でデータの処理内容を記述してある）加工ルーチンにリンクし，処理を行うものである．

（5） DGの1パス処理，DGの入出力処理と内部処理の並列処理，ブート・ストラッピング手法などによりDGの処理性の向上，コンパクト化を図る．その一例として，本DGを用いて50種程度のプラント・データベースを作成するには，制御用計算機（1語=16ビットの計算機と仮定すると主記憶容量32k語以上），磁気ドラム（85k語以上），カードリーダ，ラインプリンタが必要である．また，カード1枚あたり30データ，本DGプログログラム用主メモリ容量を6k語，磁気ドラムのアクセスタイム10 ms，転送時間150k語／秒，ディスクのアクセスタイム55 ms，転送時間75k語／秒，カード・リーダの処理時間600枚／分でDGシステムテーブルが磁気ドラム上，ユーザテーブルがディスク上にある場合，ラインプリンタの処理時間は並列処理により無視できるので50～60枚／分

の処理スピードが期待される.

6. むすび

以上，標準化を推進するための道具として用いられるテーブル・ドリブン型DGの処理概要，特徴について述べた．本テーブル・ドリブン型DGは，手続き実行型DGと比較して汎用性の点で劣るが，単純なデータ設定により作成するテーブルの多いシステムや，テーブルのデータ構造に変更が生ずる可能性の大きなシステムに適している．今後，さらにDGの加工処理部分の標準化を進め，応用分野別DG作成における生産性，保守性の向上を図っていく予定である．

最後に，本研究の遂行にあたり日頃ご指導いただく日立製作所大みか工場北之園英博，平河内良樹の両氏ならびに日立研究所川本幸雄博士をはじめ関係各位に深謝致します.

参考文献

1) IBM: Improved Programming Technologies, IBM Management Overview (1976).
2) Stevens, W. P. et al.: Structured Design, IBM System Journal, Vol. 13, No. 2, pp. 115-139 (1974).
3) 渡辺：表現能力に富む小さな文法について，情報処理，Vol. 14, No. 4, pp. 260-266 (1973).
4) 浜田他3名：制御用ストラクチャードプログラミング言語 SPL の問題向言語への応用，情報処理学会論文誌，Vol. 21, No. 1, pp. 1-7 (1980).
5) 工業用コンピュータソフトウェアの標準化動向（第一報），日本電子工業振興協会 (1972).
6) 山中他3名：TOSBAC-7000 オムニバスシステム，東芝レビュー，Vol. 25, No. 4, pp. 509-515 (1970).
7) AUTRAN-DACS SOFTWARE REFERENCE MANUAL VERSION 1.0 CDC Publication No. 39644500-A (1971).
8) 首藤他3名：プロセス制御用問題向き言語システム MDSS，三菱電機技法，Vol. 47, No. 10, pp. 1092-1099 (1973).
9) 長谷川他2名：化学プラントにおける計算機制御システム，日立評論，Vol. 58, No. 6, pp. 23-28 (1976).
10) AN INTRODUCTION TO PROCESS MANAGEMENT SYSTEMS, FERRANTI, WYTHENSHAWE DIVISION D 74000-1 (1975).
11) ソフトウェア・エンジニアリングの動向（上），日経エレクトロニクス，1979. 1. 22, pp. 146-170 (1979).
12) 高藤他4名：テーブル・ドリブン型データ・ベース・ジェネレータ，昭和53年度情報処理学会第19回全国大会講演論文集，6C-5 (1978).

（昭和54年3月7日受付）
（昭和55年10月23日採録）