

エンドユーザ言語への利用者定義機能の導入法

山本 修一郎 西山 茂 小田 英雄 田中 清
(日本電信電話公社 横須賀電気通信研究所)

1. はじめに

計算機の普及にともない、非専門家(エンドユーザ)レベルでの計算機利用を促進するために、これまでにAPL, SPEAKEASY, QBE 等多くのエンドユーザ言語が開発されている。

使い易いエンドユーザ言語を実現するためには、利用者の必要とする機能が利用者の使用目的に合った仕様で利用できるようにすることが望ましい。しかし、利用者ごとに要求が種々異なるため、はじめからこれらの要求すべてを満たす機能を持った汎用のエンドユーザ言語を開発するのは難しい。また、個々の利用者ごとに個別なエンドユーザ言語を開発するのは利用者数が増大するについて困難になる。

このため、エンドユーザ言語に自己拡張性を持たせ、基本機能に加えて、各利用者ごとに自分の必要とする機能を必要とする仕様で追加定義できるようにしておけば、一つの言語で種々の要求に応えることができるようになる。このような自己拡張機能をここでは利用者定義機能と呼ぶことにする。

利用者と計算機とのインタフェースとして従来は文を用いていたが、高度な画面制御機能を持つディスプレイ端末の普及により、図表によるインタフェースを持つエンドユーザ言語が開発されている。たとえば、データベース検索言語としてQBE^[2]があり、習熟性・操作性の良さが指摘されている。また、事務処理用の各種テーブルをディスプレイ上に表示される2次元テーブルで表現したものとしてVisicalc^[3]やPIPS^[4]が注目されている。このようなディスプレイ上に表示される図表についても、個々の利用者の使用目的に合った仕様で利用できるようにする必要がある。したがって、利用者定義機能についても文だけではなく、図表によるインタフェースを実現するものが必要である。

本資料では、以上の観点から、現在検討を進めている図表による対話機能を持つエンドユーザ言語GUIDE^[1]における利用者定義機能について述べる。

2. エンドユーザ言語の役割と従来言語の問題点

最近のパーソナル・コンピュータや日本語情報処理技術等の発展を契機として事務処理業務の担当者を中心とするエンドユーザ自身が計算機を直接使いたいという要求が高まってきている。ここでは事務処理分野に限って考察を進める。事務処理分野のエンドユーザ言語のねらいは、以下のような業務に計算機を適用し、エンドユーザ自身で処理を行なうことにある。

- (1) 簡易統計資料・文書作成など、コスト・効率などの点でこれまでプログラム化されなかった業務
- (2) 経営計画の策定や業務分析などのうちで、段階的な問題解決を必要とし、あらかじめプログラム化されなかった業務

ところが、従来の事務処理分野における計算機の利用形態では、高度なプログラミング知識を持つ専門のプログラマが目的に応じて開発した事務処理用ソフトウェアを一般利用者が利用していた。このため、従来の事務処理用ソフトウェアは以下のような問題点があり、エンドユーザにとって使い易いとはいえない状況にあった。

- (1) 事務処理用ソフトウェアのおもな対象は定型業務であり、以下の問題点を持ったため、エンドユーザの日常業務で必要となる非定型的な情報処理に適用しにくい。
 - ① プログラムを変更しようとしても、事務処理用ソフトウェアの内部処理を理解するのが難しい。
 - ② 事務処理用ソフトウェアの開発者と利用者が異なるため、外部環境の変化に即応できない。
- (2) エンドユーザが計算機を用いて業務処理を行なう場合にも、従来業務に近い形態が望ましい。しかし、事務処理用ソフトウェアは以下の問題点を持つため、従来業務に即したイメージで計算機を利用することができない。
 - ① 業務処理で必要となる各種テーブルを表現するために記憶媒体上の性質を反映したファイルを用いている。したがって、エンドユーザがテーブルを処理する場合、ファイルイメージで処理する必要があり、机上の紙に書かれたテーブルを処理するのと同じイメージで直接アクセスできない。
 - ② コマンド形式の利用者インタフェースを持つことが多いため、コマンドパラメータの名称や位置など、特別なコマンド仕様を覚える必要がある。したがって、エンドユーザにも容易にプログラムの内容が理解でき、高度なイメージで計算機と対話可能な操作性の良いエンドユーザ言語の開発が必要である。

3. エンドユーザ言語への要求条件

エンドユーザが計算機を用いて仕事をこなす場合、コマンドなどの文によって計算機を利用するのではなく、対話的に図や表に直接データを入力することができれば、①文法を覚えなくてよい ②入力データに誤りがないかどうか一目でわかる ③日常業務に近いイメージで計算機に指示を与えることができる、など多くの利点が考えられる。

このため、ディスプレイ上の2次元図表を利用者インタフェースとして事務処理分野の情報処理を行なうエンドユーザ言語が1970年代の半ばから相次いで発表されてきている。CODASYL エンドユーザ機能委員会でも2次元の図表を用いてオフィスの作業環境をモデル化し、オフィス業務を電子的に実現しようとしている^[3]。

3.1 エンドユーザ言語の問題点

このようなディスプレイ上の2次元図表を用いたエンドユーザ言語では、できるだけ適用業務に近いイメージで計算機と対話できるように基本機能ごとに各種図表形式の仕様をシステム側であらかじめ規定している。しかし、これらのエンドユーザ言語には以下のような問題点がある。

- (1) ディスプレイ画面ごとに1つの図表が定義されているため、多くの図表を同時に表示することができない。したがって、作業環境が見渡れにくい。

(2) 図表形式が固定的に定義されているため、自分なりの最適な図表形式が利用できない。

(3) 固定的なパッケージ型の言語として提供されているため、使用過程でシステム提供機能を言語の外側にある既存のソフトウェアで補ぎなかったり、既存のデータを流用することができない。

3.2 エンドユーザ言語に要求される機能

3.1 で述べた問題点を解決し、柔軟で拡張性の高い、エンドユーザ固有のシステムを構築するためには、エンドユーザ言語に以下の機能を導入する必要がある。

(1) ディスプレイ画面上に複数の図表が表示できるマルチウィンドウ機能

(2) 複数の図表が持つ機能を組合せて利用するため、図表間で共用されるデータをリンクする機能

(3) 任意の図表形式が登録できる利用者定義機能

(4) 利用者定義の図表形式や複数の図表間で共用されるデータを自由に追加・修正できる形でエンドユーザごとに保存しておく作業空間機能

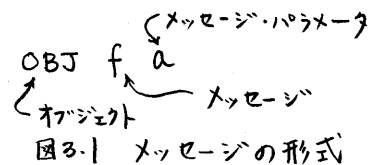
(5) 既存のソフトウェアおよびデータとのインタフェースを仮想的化するフィルタ機能
我々が現在検討を進めている GUI DE では、オブジェクト指向型言語^[6,7,8,9]を用いることによってこれらの機能の導入を図っている。Smalltalk に代表されるオブジェクト指向型言語の概念をのべて述べる。

3.3 オブジェクト指向型言語の概念

オブジェクト指向型言語では、あるデータとそれに関する操作の集まりによってオブジェクトを定義し、すべての処理はオブジェクトの操作で表現される。たとえば、2次元のテーブルはその大きさや各欄の内容などの内部データと挿入、削除、修正などのテーブルに固有な操作によって定義されるオブジェクトである。

オブジェクトとして定義されるテーブルは、大きさや各欄の内容によって様々のものが存在するがテーブルに適用できる操作は共通である。そこで、オブジェクトを生成する機能としてクラスを用意し、クラスによって生成される各オブジェクトの内部データの構造とその操作を一括して定義する。オブジェクトの内部データはオブジェクトが生成されるとそのオブジェクトが消滅されるまで保存される。

オブジェクトに関する操作はメッセージと呼ばれる。あるオブジェクト OBJ に操作 f を適用する場合、OBJ にメッセージを送るという形で図3.1のように記述される。 f がパラメータ a を持つときはメッセージの右側に a を指定することができる。



このように、オブジェクト指向型言語は次のような特徴を持ち、オブジェクトの内部データやその操作の詳細を利用者は知る必要がないため操作性が優れており、エンドユーザ言語に適している。

(1) オブジェクトの内部データをアクセスする手段をその操作によって明確に規定できるので内部データが不当に変更されない。

(2) オブジェクトへのメッセージの適用という形ですべての処理が統一的に表現される。

4. エンドユーガ言語 GUIDE

GUIDE はディスプレイ端末を介して対話的に処理を行なうことができるエンドユーガ言語である。すなわち、GUIDEでは、図4.1に示す基本機能をできるだけ業務に即した仕様で実現するため、データベース検索はテーブル形式の図表を用いる等、提供機能ごとに各種の図表形式を導入している。これらの図表を**カイダンス**と呼ぶ。

カイダンスで表わされる各種機能を総合し、複合サービスを可能とするため、以下の機能を導入している。

- (1) 複数カイダンスをマルチウィンドウ形式で同一画面上に表示する画面管理機能
- (2) 複数カイダンスで共通して用いられるデータを結合するための変数のリンク機能

また、エンドユーガごとの要求に対応できる柔軟なシステムを構築するため、以下の機能を導入している。

- (3) カイダンスを定義する利用者定義機能
- (4) 利用者定義のカイダンスやカイダンス間で共用されるデータを記憶しておく作業空間機能

利用者定義機能はエンドユーガが必要とするカイダンスをオブジェクト指向型言語を用いて追加・拡張する機能である。このため、GUIDEは図4.2に示すように2レベルの言語から構成される。すなわち、上位レベルは図表形式言語であり、下位レベルはオブジェクト指向型言語である。

作業空間はエンドユーガごとに提供される。このため、カイダンスを自分の作業空間に段階的に登録することにより、エンドユーガは拡張性の高い図表形式言語を作業空間上に容易に構築できる。

さらに、既存のプログラムやデータとのインタフェースとして次の機能を導入している。

- (5) フィルタ機能

4.1 GUIDEシステム構成

GUIDEシステムの構成を図4.3に示す。図4.3 GUIDEシステム構成

モニタ部では、GUIDEシステム全体の動作を管理する。オブジェクト処理部ではオブジェクト指向型言語で記述されたカイダンスの処理を行なう。画面管理部では複数のカイダンスをマルチウィンドウ形式で同一画面上に表示するための管理を行なう。リンク変数管理部では複数カイダンス間でリンクする変数を管理する。作業空間管理部では利用者定義のカイダンスやカイダンス間で共用されるデータ等実行に必要なすべての情報を記憶・管理する。フィルタは他言語で記述されたプログラムを呼び出したり、ファイル上のデータとのインタフェースを仮想的にするために用意されたシステム組込みのオブジェクトである。

基本機能

- データベース検索
- グラフ作成
- 帳票作成
- メール転送

拡張機能

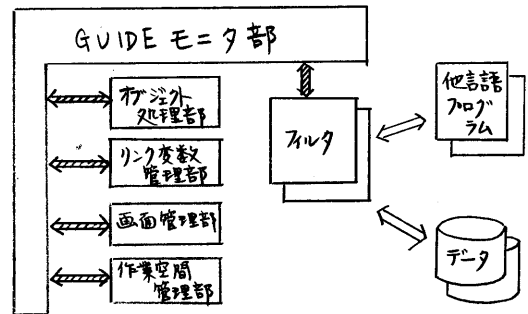
- 利用者定義機能

図4.1 GUIDEの基本的な機能

図表形式言語

— — — ↑ 構築 — — —
オブジェクト指向型言語

図4.2 GUIDEの言語構成



⇔ オブジェクトによるインタフェース
⇔ 従来インタフェース

4.2 カイダンスによる対話

GUIDE では、ディスプレイ画面の最下段にエンドユーザがシステムに指示を与えるための指令カイダンスを設けている。以下では具体的な対話例について述べる。

指令カイダンスに KENSAKU と入力する。どのような指示ができるかわからないときはあらかじめ指令カイダンスに ? を入力して利用可能な指令をマニュアルカイダンスに表示しておく(図4.4)。

次に、指令に対応したカイダンスがディスプレイに表示されるので、そのカイダンスを介して必要な情報を画面に直接入力する。別のカイダンスとデータをリンクする場合にはディスプレイ上にそのカイダンスを表示し、下線付き英字列で示されるリンク変数(図4.5では A および B)を各カイダンスに入力する。

最後に、カイダンスへの入力を終えた段階で指令カイダンスに RUN を入力することにより、結果がディスプレイ上に表示される。

このように、カイダンスは、①システムに対して指令を与える②処理に必要な入力を促進する③結果を表示する④マニュアルやシステムの状態に関する情報を表示する等の機能を持つので、エンドユーザは自分の業務環境に即したイメージで目的とする仕事を容易に行なうことができる。

4.3 画面管理機能

カイダンスが持つ利点を向上させるためには、ディスプレイ画面の物理的大きさの制限を意識することなく、各種の機能を利用できる画面管理機能を実現する必要がある。要求される画面管理機能を表4.1に示す。

表4.1 要求される画面管理機能

	画面管理機能	内 容
1	複数カイダンスの表示・管理	1つの物理画面上で複数カイダンスを任意の位置に表示する
2	個々のカイダンスの管理	任意の時点で任意のカイダンスにアクセスする
3	動的な画面管理	カイダンスの表示・消去・定義を自由に行なう
4	画面の世代管理	カイダンスの集まりからなる画面を格納・再表示する

4.4 物理画面とのインタフェース

カイダンスは以下のような物理画面とのインタフェース機能を用いてディスプレイ画面上に定義される。

- (1) フィールド定義機能 ディスプレイ画面はフィールドと呼ばれる矩形領域の集まりとみなされる。各フィールドは、①画面上の位置およびその範囲②入出力の可否およびそのデータ種別等の属性によって定義される。
- (2) データ入出力機能 キーボードから入力されたフィールド内のデータを読み込んだり、フィールドにデータを書き込む。
- (3) カーソル設定機能 フィールド内にカーソルを設定し、入力を促す。

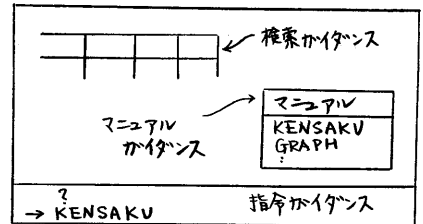


図4.4 対話例(その1)

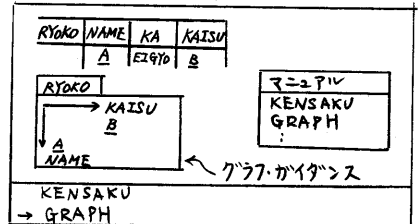


図4.5 対話例(その2)

5. GUIDEにおける利用者定義機能

4で述べたように、利用者定義機能はカイダンスを定義するための言語機能である。カイダンスをいくつかの基本的なフィールドに分割し、それらを組合せて定義することができれば、複雑なカイダンスも基本的なフィールドを用いて容易に定義することができる。ところが、基本となるフィールドごとに適用される操作を通常のプログラミング言語における手続き形式で定義する場合、次の2つの方法が考えられるが、いずれも問題点がある。

〔方法1〕 各カイダンスに閉じた形で使用し、複数のカイダンス間で共用しない。

〔方法2〕 複数のカイダンス間で共用する。

方法1では、同一の手続きを複数のカイダンスのフィールドごとに個別に作成する必要がある。また、方法2では、カイダンスに固有な情報を手続きの内部データとして保存できないため、以下のような問題点がある。

(1) 以下の情報をテーブルで個別に管理する必要がある。

- ① カイダンス種別
- ② カイダンスの表示位置およびその範囲
- ③ カイダンス内のフィールド属性およびその内容

(2) カイダンス種別等の情報を手続きのパラメータとする必要がある。

(3) カイダンスに固有なデータが途中で破壊される可能性がある。

以下では、これらの問題点を解決し、基本的なフィールドを用いてカイダンスを階層的に定義できるオブジェクト指向型の利用者定義機能について述べる。

5.1 システムで提供されるクラス FIELD

4.4で述べたフィールドは次に示すようなシステム提供のクラス FIELD のオブジェクトとして生成される。フィールドの範囲は画面上のある領域の左上隅を原点とする座標値 X 、 Y と高さ H および幅 W によって表現され、フィールドの内容および属性はそれぞれ $CONT$ および $ATTR$ で表現される。

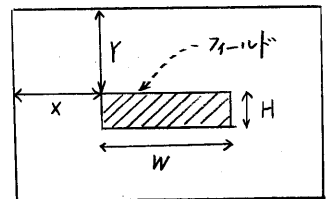


図5.1 フィールドの範囲

```

CLASS FIELD X Y W H CONT ATTR (注)
( ISNEW: X ← (:), Y ← (:), W ← (:), H ← (:), ATTR ← (S). {オブジェクトが生成されると、自動的にこの操作が実行され、フィールドの範囲、属性を初期化する}
IN: { CONT にフィールドの内容を読み込む }
OUT: { CONT の内容をフィールドに表示する }
ATTR: { フィールドの属性を返す }
CONT: { CONT の内容を返す }
X: { X の値を返す }
Y: { Y の値を返す }
W: { W の値を返す }
H: { H の値を返す }
CUR: { フィールドにカーソルを設定する }
MOVE: X ← X + (:), Y ← Y + (:). { フィールドを移動する }
ERASE: { フィールドの内容を消去する } )

```

(注)

- ① 操作の内容を $\{ \}$ に示す。
- ② $(:)$ はメッセージ・パラメータを詳細に入力することを示す。
- ③ (S) はメッセージ・パラメータを文字列として入力することを示す。

ISNEWはFIELDのオブジェクトを生成したときに1度だけ実行される初期化操作を表すためにシステムで用意された操作名である。これ以外のIN,OUT等はクラスごとに定義される操作名である。また、クラスを定義する場合、生成されたオブジェクト自身の参照が必要になることがある。このため、特別なオブジェクト名としてSELFが用意されている。

システムクラスFIELDを用いて利用者がフィールドを定義する場合、利用者は個々の操作の詳細を知る必要はなく、操作の仕方だけを意識すればよい。

5.2 クラスによるガイダンスの定義

以下では、図5.2に示すグラフ・ガイダンスのクラスによる定義法について述べる。

グラフ・ガイダンス内のフィールドは、初期化操作ISNEWでFIELDのオブジェクトとして生成される。各フィールドの位置はグラフ・ガイダンスのオブジェクト自身の位置とグラフ内のフィールドの相対位置によって初期化される。グラフ・ガイダンスのオブジェクト自身の位置はシステム提供のオブジェクトSELFを用いてSELF XおよびSELF Yの形で得ることができる。これを用いてグラフ・ガイダンス上のX軸データ名に対するオブジェクトXAXISは以下のようにしてISNEW操作で生成される。

$$XAXIS \leftarrow FIELD (SELF X) + X_1$$

$$(SELF Y) + Y_1, W_1, H_1, IN$$

ただし、 X_1, Y_1 はXAXISのガイダンス内の相対位置、 W_1, H_1 はその幅および高さ、INは属性を示す。XAXISがFIELDのオブジェクトとして生成されると、FIELDの初期化操作が実行される。たとえば、XAXISのX座標値は $(SELF X) + X_1$ となる。

このように、フィールドを1つのオブジェクトと考えることにより、ガイダンスの階層的な定義ができる。すなわち、図5.2のガイダンスを図5.3のような階層構造としてとらえ、グラフ名NAME、軸名XNAMEおよびYNAME、軸データ名XAXISおよびYAXIS、グラフをプロットする点POINT等のオブジェクトを用いて定義することができる。これらのオブジェクトはシステム・クラスFIELDのオブジェクトとして生成される。この場合、FIELDのオブジェクトの操作はすでに定義されているので、上位のオブジェクトの内部手続きとしてこれらの操作をあらためて定義する必要はない。

5.3 ガイダンスの管理および操作

利用者が定義したガイダンスは利用者ごとに用意される作業空間に登録され、システム組み込みのガイダンスと同様に利用することができる。たとえば、4.2で述べたように指令ガイダンスで利用者定義のガイダンス名を指定すれば、そのガイダンスがディスプレイ上に表示される。また、マニュアル・ガイダンスによってどのようなガイダンスが作業空間に登録されているかを知ることができる。

4.3で述べたような複数ガイダンスを管理する場合、個々のガイダンスに関する

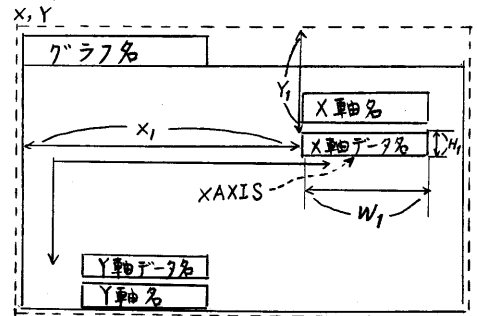


図5.2 グラフガイダンス

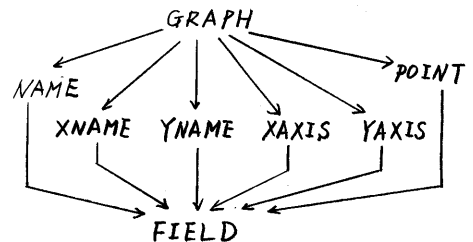


図5.3 階層構造

るフィールドの定義情報はガイダンスのオブジェクトに固有なデータとしてオブジェクト内に保存されるため、画面管理部ではこれらの情報を別テーブルで管理する必要がない。また、ガイダンスの動的な管理では、ガイダンスのオブジェクトに対して表示・移動・消去などの操作がシステムで提供されるので、容易に管理できる。たとえば、座標値X,Yで表わされる位置だけグラフ・ガイダンスを移動する場合、

GRAPH MOVE X Y とすればよい。このような操作は、指令ガイダンスで利用者が直接指定することもでき、システムと利用者が統一的に利用できる。

6. おわりに

本報告では、エンドユーザ言語GUIDEにおけるわかりやすいインタフェースとしての2次元図表(ガイダンス)を定義するための利用者定義機能について述べた。本報告で提案した、オブジェクト指向型言語による利用者定義機能は以下の点で有効である。

- (1) 複雑なガイダンスを基本的なクラスを用いて階層的に設計することができ、
- (2) 必要に応じて上位のクラスだけ意識すればよい。
- (3) 下位のクラスで定義した操作は上位のクラスで定義する必要はない。
- (4) 複数のガイダンスに対して統一的な仕様で操作できる。

今後は、GUIDE処理系の具体的な実現法の検討ならびにその評価を行なう。

最後に、本検討を通じて適切な助言を賜った細谷処理プログラム研究室長ならびにエンドユーザ言語検討グループの諸氏に深謝致します。

〔参考文献〕

- [1] Zloof, M.M.: QBE/OBE: A Language for Office and Business Automation, IEEE Computer, May (1981)
- [2] Thomas, J.C. et al: A Psychological Study of Query by Example, AFIPS (1975)
- [3] A status report on the activities of the CODASYL end user facilities committee, ACM SIGMOD RECORD, 10, 2 & 3, Aug. (1979)
- [4] SORD 電算システム: PIPS電プロ入門 (1981)
- [5] Tandy Radio Shack: VISICALC Manual
- [6] Robson, D.: Object-Oriented Software Systems, BYTE, Aug. (1981)
- [7] Ingalls, D.H.H.: The Smalltalk-76 Programming System. Design and Implementation, 5th ACM symp. on P.O.P.L (1978)
- [8] 山本他: エンドユーザ言語への利用者定義機能導入法, 情報処理学会, 第23回全国大会 (1981)
- [9] 松本他: エンドユーザ言語におけるデータ抽象化導入法の一考察, 電子通信学会, 情報・システム部門全国大会 (1979)
- [10] 西山他: 個別属性情報を用いたマルチウィンドの動的な管理方式, 電子通信学会, 総合全国大会 発表予定
- [11] 田中他: 図表形式エンドユーザ言語GUIDE, 電子通信学会, 情報・システム部門全国大会 (1981)