2J−5 第4世代言語のビジュアルプログラミング化

上田　穣　日本Wavefront㈱
藤田　喜彦　㈱ＣＳＫ
白井　靖人　静岡大学教育学部
国井　利泰　東京大学理学部

1　はじめに

ビジュアルプログラミング研究の現状
アイコンとマウス、マルチ・ウインドウを主要素とする各種グラフィック・ユーザー・インターフェイス（ＧＵＩ）ツールの急速な発展につれ、ビジュアル・プログラミング研究も進展してきた。文字データの入力以外は、キイボード入力が不要な事が共通の特徴である。個別処理を示す関数アイコンをマウスで選択移動配置する。１つのアイコンにそれぞれ入出力が付属している。一種の絵合わせパズル的発想で、ユーザが複数のアイコンを組合せる。システムがそれを自動解釈し、文字表現として評価する事で、１つの適用業務が処理される。繰り返しループが必要な場合は、図同士を入れ子構造にして対処するのが通常である。
ウインドウは、多目的ウインドウと編集用ウインドウの最低２種を使い分けている。
現在の問題点
過去１０年余の経過を見ると、Ｃのような高級言語を目指しながら、極少数の変数しか使用できず、適用範囲は極めて制限されている。
また、ビットマップ機能で様々の新奇なアイコンが提案されているが、上流ＣＡＳＥのデータフローダイアグラムのレベルを越えていない。
結果として、現実の業務への適用は難しく、研究教育用として存在しているのが実状である。

新しい方針
「よいユーザーインターフェイスとは何か」の原点に戻り、成功している実例を検討する事により、方針を求めていくことにする。よいＧＵＩは、
(1)　初心者の学習時間が短い
用語が馴染み深く、命令体系が簡潔かつ論理的一貫性を持ち、入力エラーが少ない。（視覚的なオブジェクト指向の操作で実現）
(2)　熟練したユーザがより早く実行できる
（バッチ処理的な近道が可能）
上記(1)(2)は相反する内容を持つが、全てをアイコンとマウスで処理するのでは無く、洗練されたデザインにより、視線と手の移動距離を最小にするというコスト・パフォーマンスの高さを第一目的とするべきである。
　成功しているビジュアル・プログラミングの例は、ＣＡＤとお絵描きソフトであるが、両者とも適用業務の内容が明確であり、まずグラフィックサブルーチンとして、順次開発蓄積されたものがサブルーチン・パッケージとなった。次にマクロ言語が開発され、その後初めてビジュアル・プログラミング化が進められたのである。

Visual Programming of 4GL
Minoru Ueda (Wavefront)　Yosihiko Fijita (CSK)
Yasuto Sirai (Fac. of Education Sizuoka Univ.)
Tosiyasu Kunii (The Univ. of Tokyo)

2　第四世代言語（４ＧＬ）の現状

ソフトウエアの７０％は、事務計算用であり、１９７０年代半ばから、ソフトウエア生産性向上の手段として、ＣＡＳＥ，データベースシステム（ＤＢＭＳ），そして第四世代言語が登場した。当初それぞれの得意範囲は別々であったが、ＤＢＭＳが最も発展するにつれ、そのデータベース操作言語（ＤＢＭＬ）の機能が４ＧＬの機能を順次浸食してきている。しかし、パソコンに搭載されている４ＧＬに対応する、ＤＢＭＬの機能は、キイによるデータ検索、単項式計算、統計計算を含むむ計算処理であり、高級言語が持つ条件判別文と繰り返しループ文を欠く場合が多い。ＤＢＭＬに比べ、４ＧＬの特徴は、個別命令を数十連結したマクロ処理が可能なことである。

また高級言語に比べた時の、４ＧＬの特徴は
特定ハードウエア、ソフトウエアにより異なる
(1)データベース記述に伴うファイル、フィールドデータなどの定義
(2)ワーク領域の確保
(3)入出力に関する諸指定
について考慮する必要が無く、
結果として、ユーザーは
(4)処理のロジック記述だけを考えればよいので、第三世代言語に比べ、生産性が高いとされる。
これを可能にしたのは、表形式データで事務処理をすると、適用業務の内容が明確だからである。

本研究では第四世代言語としてＳＴＹＬＥ*を用た。その体系は、３０余のコマンドと２０余のスイッチの組合せの約８０の命令群からなる。（使用頻度が多いのは、その内約３０命令である）基本的にはインタープリター方式であるが、バッチ様式で使用できる。独自のデータベースを持つが、他のＤＢＭＳとのネットワークが可能で、かつマルチトランザクション可能である。同時に呼び出せるデータファイルは２つ、繰り返しループは２階層で、この機能で全ての適用業務に対処できる事が経験的に分かっている。
　このＡＳＣＩＩ端末使用を前提にした第四世代言語を、ビジュアルプログラミング化するのに、ＩＢＭ　ＡＩＸプラットフォーム上のＸ−ｗｉｎｄｏｗ／Ｍｏｔｉｆをツールとして使用した。

＊　㈱コンピュータソフトの登録商標

3　研究用プロトタイプの仕様

現在ＳＴＹＬＥは、数千人の従業員をもつ組織で管理業務に使用されているが、ＢＡＳＩＣでプログラミング開発するようにして使用されている。その作業過程を以下に述べる３段階でビジュアライズ化を試みた。

第一段階　約８０余の個別命令のクラス化
基本方針は、データのオブジェクト指向化では無く、４ＧＬの命令が表すアルゴリズムのオブジェクト化である。すなわち、アルゴリズムのクラスに具体的なファイル名、フィールド名、変数名、データ値、行列値を与える事により、アルゴリズムのインスタンスを生成する。８０余の命令は機能として下記の９種類になる。

(1)　システム　　PASSWORD,HELP 等
(2)　制御　　IF,LOOP,SUBROUTINE等
(3)　ファイル　　新規、書込、呼出、削除等
(4)　フィールド　新規、書込、呼出、削除等
　　キイ指定　　新規、書込、呼出、削除等
(5)　変数　　新規、書込、呼出、削除等
(6)　データ　　新規、書込、呼出、削除等
(7)　レイアウト　コメント文、変数とデータ
　　行列指定、左中右指定
(8)　入力（画面）INPUT
(9)　出力（画面　印刷）PRINT

このインスタンス化されたアルゴリズムは、単独で単一の処理を実行する。

第二段階　プロシージャー
上記のインスタンスを複数（数十）並べて、バッチ様式で処理を行なう。通常、複数のデータファイルを、同時２つメモリ上に呼び出して、順次処理する。

(1)入出力に関するもの
　データファイルの呼出し
　出力用レイアウトの指定
　入力、出力
(2)同一ファイル内のデータ処理
　欲しいデータの検索方法
(3)複数ファイル間のデータ処理
　キイ同士のマッチング

ここで繰り返し命令LOOPが使用されるが、ファイルを最初からEOF まで読むか、または、開始と終了レコードを指定してLOOPするかである。
また条件判定IFは、指定フィールドに指定データを持つレコードを検索するのみである。

故に、プロシージャー内の命令の順序は基本的に非手続き型でよいことが経験的に判明している。

第三段階　プロジェクト
この段階で初めて手続き型処理が必要になる。すなわち、プロシージャ＃１を起動して、複数のデータファイルを順次呼び出し、その処理結果を１つのファイルとして出力する。
次に、出力ファイルを、入力ファイルの１つとして、プロシージャー＃２を作動する。前項と同じようににして、１つの出力ファイルを得る。
　この操作を数回（数十回では無い）繰り返すことにより、第三世代言語で数十万ステップから、百万ステップを要した事務処理がビジュアルプログラミングにより実行可能となる。
即ち、各プロシージャーが必要とする入出データは予め判明しており、最終結果として必要なデータが何か決まれば、プロシージャ・アイコンを手続き的に配列するにより、解が得られる。

4　おわりに
現在使用可能なＧＵＩ構築ツールを駆使することにより、既発表のビジュアル言語との対比、従来の文字端末使用の４ＧＬに比べて、生産性の違いを実例を通じてより明確にしてゆきたい。

参考文献

T.Kunii et al, *Visual database System*, north-holland, Tokyo, 1989
S.Chang et al, *Visual Languages*, Pleneum Press, NY, 1986
「STYLE ｲﾝﾄﾛﾀﾞｸﾄﾘ ﾏﾆｭｱﾙ」1987、ｺﾝﾋﾟｭｰﾀｿﾘｭｰｼｮﾝｽﾞ株式会社