ビジュアル言語ｖｏｃプロトタイプの試作

4S－7

田代　秀一　　岡田　義邦

電子技術総合研究所

1．はじめに

計算機の普及に伴い，より対話性に優れたビジュアルなユーザインタフェースが求められている．それに伴いソフトウエア開発コストに占めるユーザインタフェース部分のウエイトは増加の一途をたどっている．

このため近年ビジュアルユーザインタフェースの開発支援システムが各所で研究されるようになり，NeXT社のInterface-Builder，Sun Microsystems社のGUIDE等，一部市販されるものも出てきた．

これらのシステムは，①ボタン，スライダー，メニュー等，マンマシンインタフェースの実現に多く使用される基本部品を予め豊富に用意しておく．②これら基本部品の配置，一部の変形(ｶｽﾀﾏｲｽﾞ)等をビジュアルな操作で行なえる．といった点をソフト開発者に対する支援の中心として設計された物が多い．

こういったシステムは，①マンマシンインタフェース用の部品（以後ビジュアルオブジェクトと呼ぶ）を新規に作成し，それを既製部品と同様に扱うことが困難である．②ビジュアルオブジェクトとそれを用いるプログラムテキストとの間の関係の記述力が弱い．といった問題点を持つ．

又，ビジュアルオブジェクトの配置の変更，オブジェクト間の結合状態の変更等のための支援機能がプログラム開発時にしか発揮されず，プログラムの実行時にこれらを動的に変更できない．プログラムの動的変更は，エンドユーザによるカスタマイズ等のためにぜひ必要とされる機能と言えよう．

これらの問題点を解決するため，現在ｖｏｃ[1]の開発を行なっおり，プロトタイプ版の試作を，ＮｅＸＴコンピュータ上で行なっている．

2．ｖｏｃの概要

ｖｏｃは図的定義と手続定義を融合させてクラスを定義することのできるオブジェクト指向言語である．ビジュアルオブジェクトの機能の定義，ビジュアルオブジェクト間の結合関係（メッセージルーティング）の定義を，テキストによる記述と，図的な記述を混在させて記述できる．

ｖｏｃでは，オブジェクトに対して‘名前’に加えて‘形’を与えることができる．クラスの定義は，

①Objective-C言語[2]に上位互換な言語によるメソッドの記述，

②表現図形（このクラスがインスタンシエートされた時に表現される図形），

③図的プログラム（定義済のビジュアルオブジェクトのポート（後述）間を線で結んだもの）

の３種の記法によって行なえる．これら３種は全てが同時に含まれる必要はなく，少なくとも１種があればクラス定義は成立し得る．

表現図形中にはポートと呼ぶ座標点あるいは領域を任意個定義でき，そこに与えた名前をテキストプログラム中で参照できる．

ポートに対してメッセージをやり取りする手続を書くと，実際のメッセージの送り先は，そのポートが図的に接続された先のオブジェクトとなる．

ビジュアルオブジェクトの結合によって構成される図的プログラミング部分は，アプリケーションの開発課程においてビジュアルなだけでなく，実行時にはそのままビジュアルユーザインタフェースとして機能させることができる(結合線等，一部を非表示とすることができる)．又，アプリケーションの実行時にオブジェクトの結合状態等を動的に変更可能である為，イージープログラミング可能なアプリケーションの開発にも適している．

図１にｖｏｃの位置付けを示す．

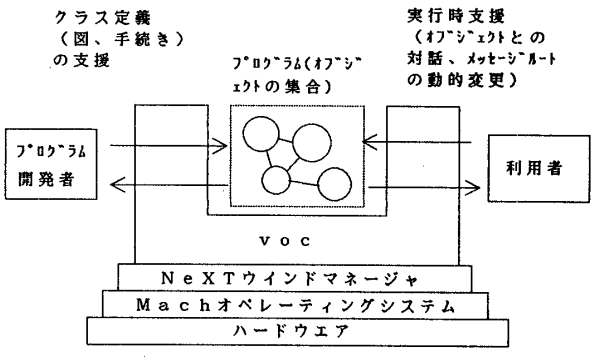

図１　ｖｏｃの位置付け

Implementation of the visual language VOC prototype
Shuichi TASHIRO, Yoshikuni Okada
ElectroTechnical Laboratory

3．クラス定義ツール

クラス定義はc-windowと呼ぶウインド内部で行なう．c-window内にはv-window及びt-windowと呼ぶサブウインドがある．

v-window内にはこのクラスから作られるオブジェクトが形を持つ場合（ビジュアルオブジェクト）にその図的表現を，t-window内には文字による手続を定義する．

c-window, v-window, t-windowはNeXT application kitのwindowクラス, scrollviewクラス, textクラスを用いることにより，容易に実現できた．

4．クラスの図的定義

v-window内部にマウスポインタを置き，ダブルクリックした後ドラッギングするとv-window内部の図形のコピーが現れマウスに追従する．これを他のc-windowへ移動させ，そこでボタンを離すことによりその場所に図形が貼り込まれる．

この操作は，t-window中に

```
id object_name;
object_name = [class_name new];
```

と書くことと等価であり，上記のようなコードが自動生成され，内部的にクラス定義に追加される．

あるビジュアルオブジェクトのポートを，他のビジュアルオブジェクトのポートに接触させる，あるいは線で結合することにより，メッセージのルーティングを定義できる．

5．実行時のサポート

プログラムは，１つ以上のワークウインド(w-window)を持つ．

図２に示すように，w-windowはｖｏｃによるプログラム記述の最上位階層に位置し，プログラム実行時には，w-windowだけが画面に表示されてマンーマシンインタフェースの役割を果す．

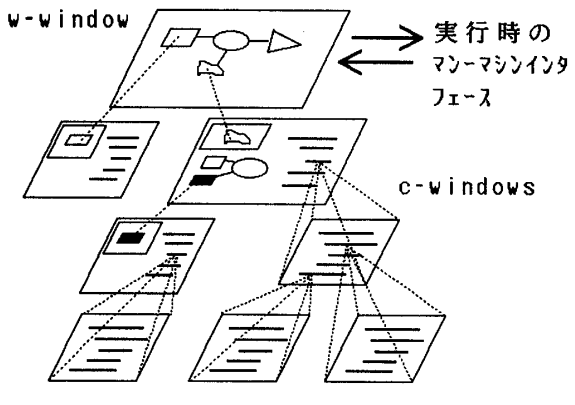

図２　w-window と c-window

プログラム作成時にw-windowに貼り込まれたビジュアルオブジェクトは，実行時に利用者と対話することができる．

又，w-window内部のビジュアルオブジェクトの結合状態は，実行時に利用者が変更することが可能である．

6．教示的プログラミングの実現

幾つかのパラメータの値が未定であるようなライブラリを用意し，実行時に図的な対話を通してその値を決定するという教示的プログラミングは，図的マンマシンインタフェースを記述する際の労力を軽減する際に欠くことのできない機能である．

ところが，教示機能をプログラミングツール側に持たせてしまうと，新規に作成したクラスを部品の一員に加え，既製の部品と同様の操作／編修対象とすることが困難になる．

そこで，ｖｏｃでは教示のされかたの定義もクラスの定義の一部として持たせることが可能な方式をとった．

このために用意したのがpermanent記憶クラスと部分実行機能である．

教示機能の定義されたクラスのオブジェクトをc-windowに貼り込んで利用する場合，まずそのオブジェクトだけを部分実行し，そのクラスに定義された方法で，対話的，教示的にパラメータを決定する．これらのパラメータがpermanent記憶クラスとなっていれば，ここで一旦決めた値は，プログラムを再度実行する際にもリストアされて利用されることになる．

7．おわりに

NeXTのInterface-Builderを用いることにより，ｖｏｃのマンマシンインタフェース部分の開発は極めて容易に行なえた．

現在の版ではメッセージルートの動的変更はw-windowに対してしか行なえない．w-windowという特別なものを設けず，全ての部分を動的変更可能にすることを検討している．

オブジェクトの並行動作化，分散システムへの対応が将来の課題である．

謝辞　本研究の機会を与えていただいた棟上情報ｱｰｷﾃｸﾁｬ部長，有益な助言をしていただいた分散ｼｽﾃﾑ研究室の諸氏に感謝の意を表する．

文献

[1] 田代，岡田："図的記述機能を持つｵﾌﾞｼﾞｪｸﾄ指向言語ｖｏｃ"，情報処理学会ﾏｲｸﾛｺﾝﾋﾟｭｰﾀとﾘｰｸｽﾃｰｼｮﾝ研究会，1989-5

[2] "Objective-C Compiler v4.0 User Reference Manual", Stepstone corp., 1989