5J−6 ビジュアル言語VOCのソフトウエア部品カストマイズ機構

田代 秀一
電子技術総合研究所

1 はじめに

グラフィカル・ユーザ・インタフェース(GUI)の効率的開発を目的の一つとしてプログラミングシステムVOC[1][2]を開発中である。VOCは,オブジェクトの参照関係,大局的な制御の流れ,GUIウインドウのデザイン等を視覚的に記述し,プログラムの詳細な仕様はテキストで記述できるよう,これらの記述法を混在させてプログラムを構築できることを特徴としている。

特にGUIの作成においては,既製部品の仕様の一部,例えば大きさ,色等,を変更し,それをアプリケーションに組み込むという作業が頻繁に発生する。VOCは部品の仕様を変更するための手続き(カストマイザ)を,部品自体に内蔵させることを特徴としている。これにより,既製ツールの仕様にしばられずに,任意のカストマイズ可能な部品を作成することを可能とした。

本稿ではVOCのカストマイザ機構について報告する。

2 カストマイザ

VOCはオブジェクト指向言語の一種であり,プログラムは,オブジェクトの集合として記述する。

クラスにはアイコンを与えることができ(クラスアイコン),クラスアイコンをアイコニックプログラミング用のウインドウ上に貼りつける操作を行なうと,インスタンスアイコン(通常はクラスアイコンと同一形状)が生成される。Objective-Cに上位互換のテキストプログラムによるほか,インスタンスアイコンをウインドウ上で視覚的に接続することでメッセージ経路を指定し,プログラムを構築できる。アイコンの与えられたオブジェクトを視覚オブジェクトと呼んでいる。

プログラミング時に用いられたアイコンをその実行時に,直接操作の可能なアクティブなアイコンとして表示させ,利用者との対話を担わせることができる。アイコンの持つこの2面性により,プログラムの論理構造とウインドウの視覚的表現とを,同様の操作で,混在させながら記述することができる。

Software parts customizing mechanizm on VOC
Shuichi TASHIRO
Electrotechnical Laboratory

カストマイザは視覚オブジェクトに対応したクラスの内部に定義され,インスタンスアイコンとペアのオブジェクトとなって実行される。アイコンを持たないクラスは必ずしもカストマイザを持つ必要はないが,持つことは可能である。

VOCには標準のカストマイザが用意されており,各クラスのカストマイザはこれを継承したものである。これらを拡張することで,各クラスに対し固有のカストマイズ機能を与えることができる。

2.1 カストマイザの起動

VOCではクラスから視覚オブジェクトを生成するために,2つの段階をふむ。

第一段階ではクラスからインスタンスアイコンが生成される。これは,クラスライブラリからクラスアイコンをドラックし,アイコニックプログラミングウインドウ中に貼り込む操作によって起きる。この時,そのクラスのカストマイザが起動され,仕様の変更等,3で述べるような処理が行なわれる(図1)。

第2段階はアプリケーションの実行時におきるアクティブなユーザインタフェース用アイコンの生成である。

カストマイザはプログラミング時だけに存在し,通常は完成されたアプリケーションの内部へは持込まれない。したがって,アプリケーションの実行効率を落とすことなく,柔軟な適応性をソフトウエア部品に与えることができる。

3 カストマイザの役割

プログラミングの過程で既に能動的であることを利用し,仕様変更の他,プログラミングの支援のために様々な役割を演じることができる。

3.1 仕様の変更

プログラマとの対話を通して,ソフトウエア部品の仕様変更を行なう。

これは,視覚オブジェクトの形状変更など,様々なパラメータの設定を目的としている。デフォルトのカスト

マイザにはマウスポインタおよびマウスボタンの動作を追跡し，その操作に応じて形状の変更等を行なう手続が用意されている。変更結果はパーシステントな変数に記録され，アプリケーション実行時に引継がれる。

カストマイズによって決めるパラメータの値に，何等かの制限を与える必要のある場合が良く発生する。例えば，あるビューの内部に配置したサブビューの大きさを変更する場合，サブビューの大きさが外部のビューの大きさを越えないように制御するなどがその例である。

VOCはこの様に依存関係をもった部品のカストマイザの間にメッセージパスを設定する。

カストマイザ内部で使える *env* という仮のターゲットが用意されており，依存関係の下位にあるカストマイザの *env* は上位のカストマイザにバインドされる。最上位の部品の *env* は，VOCシステムにバインドしている。

ビューの大きさの問い合せに答える等幾つかの標準メソドが *env* による参照のために用意される。

3.2 自動適応

2つのインスタンスアイコン間に少なくとも一つ以上の接続(メッセージパス)が定義された場合，それぞれに対応したカストマイザ間にもメッセージ経路が作られる。

あるアイコンのカストマイザ内部から，そのアイコンに接続されたアイコンのカストマイザへ対しては，

[*neighbor*[*n*] *method*...];

のようにしてメッセージを送ることができる。*n* は接続の順に0から振られて行く番号である。

この交信により，接続された2つの部品の間で様々なデータを交換することが可能である。これによって，接続相手のソフトウエア部品のバージョン，メッセージ仕様などを問い合せ，その結果に基づいて自己の仕様を修正する自動適応機能を実現することができる。

3.3 ナビゲーション

カストマイザには，仕様変更以外に，プログラマを援助するための様々な機能を与えることができる。

デフォルトカストマイザに用意された主要な機能に，アイコンの接続に際して，相手アイコンに定義された複数メソドと，接続元アイコンに定義された複数のメッセージソースから，適切なものを選択して接続する作業を援助する機能がある。型の一致等から接続先と接続元の候補をリストアップし，ポップアップテーブルとして提示して，選択を促す。

オンラインマニュアルに類した機能を内蔵させることも可能である。プログラマの要求によってドキュメントを表示する他，接続の誤り等の発生により自動的に警告を表示させる等様々な動的機能を持つよう拡張できる。

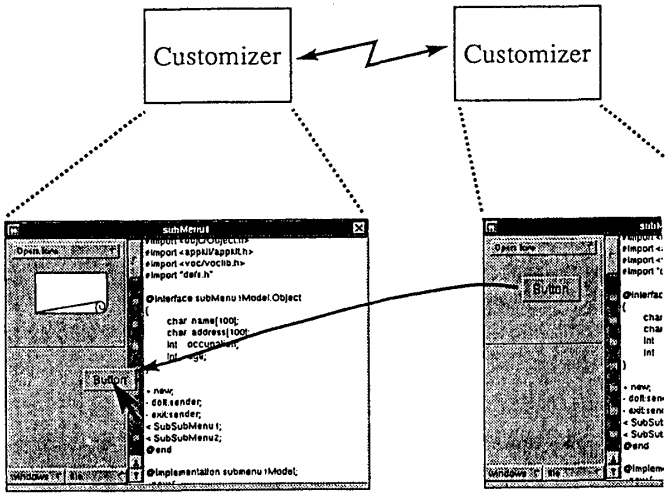

図1　カストマイザの起動

カストマイザはインスタンスアイコン生成時に起動され、他のアイコンのカストマイザ、プログラマ、VOC処理系と交信しながら、仕様変更、自動適応等の処理を行う

4 おわりに

以上,VOCのカストマイザについて述べた。ソフトウエア部品のカストマイズを行なう機能を専用のツールに持たせるのではなく，ソフトウエア部品自体に内蔵させることで，既製ツールの仕様にとらわれず，自由にソフトウエア部品を作り，流通させることができる。又，プログラミング時に既に能動的であることを利用し，プログラマに対し，ナビゲーション，部品の自動適応などの様々な支援機能を提供できる。

これらの特性は，ソフトウエア部品の流通，再利用促進のために有利であると考えている。

カストマイザ間の通信仕様の標準化，ソフトウエアバスを用いた部品合成について現在検討を進めている。

VOC処理系のプロトタイプは現在NeXTコンピュータ上で稼働しており，ライブラリ等の整備を進めている。

参考文献

[1] 田代, 岡田: “VOCにおけるマン-マシンインタフェース教示機構 - MVCTモデルの提案 -”, 第41回情処全大会,'90年9月

[2] 田代: “GUI構築を指向したビジュアル言語VOC”, 日本ソフトウエア科学会第9回ソフトウエア研究会,'91年10月