視覚プログラミングと対話的操作の融合

2G－4

小池 雄一　前田 康行　古関 義幸 *
NEC C&C研究所 †
e-mail: koike@swl.cl.nec.co.jp

1 はじめに

本稿では、視覚プログラミングと対話的操作の融合方式を提案する。Serius[1], Synchro Works[2]等、従来の視覚プログラミングシステムの多くはGUI部品等の対話部品の配置と視覚プログラミングを別エディタで行なう。このため、GUI構築と視覚プログラミングを頻繁に交互に行なうようなプロトタイピング的なアプリケーション(AP)構築手法が取りづらいという問題がある。対話部品とプログラミングを同一エディタで行なえるシステムもあるが、逆にAP構築終了後、対話部品だけをユーザに提示する段階で困難が生ずる。

本稿ではこれらの問題に対し、対話部品と視覚プログラミングの構成要素であるアイコンを混在させ、同一操作で扱うことが可能な手法を提案する。この手法は

- 視覚プログラミングのアイコンと対話部品を、同一エディタ上で混在可能にした。また、それらの間での連携機能を実現した。
- 視覚プログラミングにおけるアイコンの配置・関係付けと対話部品の配置の双方を、作図ツールでの図形操作と同様の操作で行なえるようにした。このため、対話部品とプログラム部品の混在、分離操作が統一された

ものである。また、この手法を用いた視覚プログラミング環境をVIP(Visual object Integration Platform)と名付ける。以下では、対話部品と視覚プログラミングの融合方式および、その方式に基づいて作成したビジュアルプログラミングシステムについて述べる。

2 Viewを用いた融合方式

GUI部品のような対話部品と視覚プログラミングとを融合するために、VIPではViewという概念を用いる。Viewは対象オブジェクトの視覚化、すなわちオブジェクトの特徴的な属性の視覚的表示及びオブジェクトを視覚的に操作する手段の提供を行なうものである。視覚プログラミングの構成要素となるアイコンや対話部品を含む全てのオブジェクトはViewの一形態として実現される。VIPでは、全てのView共通の上位概念である「View」というクラスを用意する(図1)。

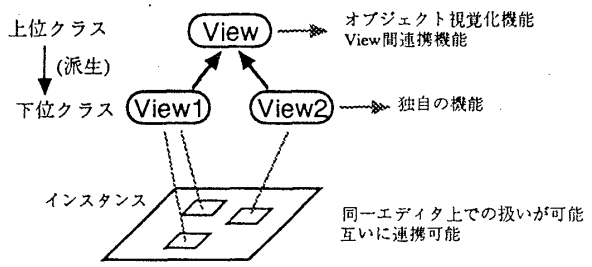

図1: Viewの共通機能

2.1 Viewの共通機能

Viewはオブジェクトを視覚化する上での基本的な以下の機能を提供する。

1. SmalltalkのMVCモデル[3]を基にした、View、オブジェクト間の依存関係の管理及び複数View間の一貫性の管理機構
2. Viewをエディタ上に図形として表示する機能
3. 表示されたViewに対し、作図ツールで図形を扱うようにカット・ペースト操作を可能とする機能

View間連携機能とはView同士がデータをやり取りする機能であり、ポートという概念で実現される。Viewはポートを構成要素とすることで、VIPのエディタとデータをやり取りすることが出来る。ポートは以下の二つの方法で互いに連携する。

1. 結線された二ポート間でのデータの受け渡し。ポート間の結線操作はVIPのエディタが提供
2. ポート上へのドラッグ&ドロップ操作。ドラッグ&ドロップされたViewが視覚化しているオブジェクトがポートへ入力される

2.2 VIPの提供するView

対話部品、視覚プログラムのアイコン等は、以上に述べた機能を持つViewの下位クラスとして実現されるため、視覚化基本機能、View間連携機能を継承し

*Yuichi KOIKE, Yasuyuki MAEDA, Yoshiyuki KOSEKI
†C&C Research Laboratories, NEC Corporation

て有する。VIP ではビジュアルプログラミング機能をプログラム View と呼ばれる View が提供する。対話部品はデータ View と呼ばれる View が提供する。データ View はオブジェクトの特徴的な属性を表示・編集するための View である。例えば文字列のデータ View はテキストフィールド、リスト構造はリストボックス、二次元配列は表となる。データ View は利用者の対話的な操作で内部状態を変更可能なため対話部品の機能を実現する。また、オブジェクト自身を入出力するためのポートを備えている。

プログラム View は図 2に示すように、タイトル、要素、入出力ポートから構成される。タイトルはオブジェクトのクラス名＋オブジェクト ID となっている。要素の一行一行はオブジェクトのメソッド名である。メソッド名の左にある矢印 (▷) は入力ポートであり、右は出力ポートである。プログラム View の入力ポートに

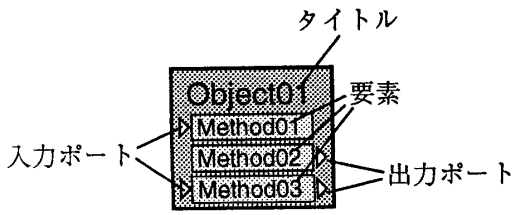

図 2: プログラム View の構造

データ (オブジェクト) が入力されると、それを引数として、オブジェクトの対応するメソッドが起動される。メソッド名に出力ポートがついている場合、メソッドの戻り値が出力ポートより出力される。ある出力ポートと入力ポートを結線した場合、View 間連携機能によって、出力されたオブジェクトは結線された入力ポートに入力される。このように、プログラム View を線で結ぶことで一連の処理を行なうアイコン & ストリング方式の視覚プログラムを記述することが可能である。

対話部品とプログラム View のポート間を線で結ぶことで、対話部品と視覚プログラミングの融合、すなわちユーザの操作からのプログラムの起動、プログラムの結果の対話部品への出力、が行なえる。

3 インプリメンテーション

以上に述べた方式に基づいた視覚プログラミングシステムを試作した。開発言語は CLOS(Common Lisp Object System)[4] であり、Lisp 上に構築された X-Window, Motif 関数の操作ライブラリを利用した。

また、この視覚プログラミングシステムを用いて、ビジネス向けアプリケーション「銀行業務融資支援システム」(図 3) を構築した結果、プロトタイピング的なアプリケーション構築に有効な以下の特長を VIP 方式が持つことが確認された。

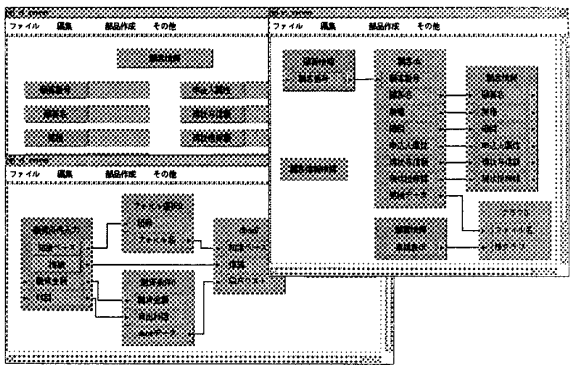

図 3: VIP で作成した融資支援システム

- システムの構築途中では GUI 部品とプログラムを同一画面に配置できるため、構築・実行・修正のサイクルが早い
- プログラミング、対話部品配置、プログラム部品と対話部品の分離を全て同一の操作で行なうことが可能である

4 まとめ

本稿では、対話部品と視覚プログラミング部品を混在して利用可能な環境 VIP を提案し、その実現方式について述べた。また、このような環境がアプリケーションのプロトタイピング的な構築に適することを、視覚プログラミングシステムの試作、及び例題アプリケーションの構築を通して確認した。

参考文献

[1] James A. Landay. Tools Review: Serius - a Visual Programming Environment. *Journal of Visual Languages and Computing*, 2(3):297–303, 1991.

[2] 東洋情報システム. *Synchro Works* 概説書. 東洋情報システム, 1993.

[3] A. Goldberg and D. Robson. *Smalltalk-80 : the language and its implementation.* Addison-Wesley, 1983.

[4] Guy L. Steele Jr. *Common Lisp, The Language, Second Edition.* Digital Press, 1989.