ユーザインタフェースモデル記述主導型の実行可能モデル記述支援環境の開発

4K－11

大津 信之　辻 新太郎　上田 賀一

茨城大学工学部情報工学科

1 はじめに

GUIを持つソフトウェアアプリケーションの開発に適した方法論の一つとして、Smalltalkに端を発するMVCモデルが挙げられる。MVCモデルはアプリケーション本体とユーザインタフェースを分離できるという利点を持ち、そのためMVCモデルを基底とした、ユーザインタフェース開発支援環境・ツールが数多く構築されている。

ここでは、まずGUIを構築する際のMVCモデルについて述べ、つぎに、そのMVCモデルを発展させた我々のユーザインタフェースモデルとその開発支援環境について述べる。

2 MVCモデル

MVCモデルは、Smalltalkで対話型のGUIを構築するために考案された枠組で、ユーザインタフェースを以下の三つのオブジェクトの組を単位として構成していくものである。

モデル	問題対象としてのデータとそのデータに対する操作をカプセル化したもの。
ビュー	モデルを画面上に表示する手段を提供するもの。
コントローラ	ユーザからの入力を解釈して、モデルやビューに適切な調整を行うもの。

MVCモデルはこの三つのオブジェクトのうち、ビュー・コントローラにアプリケーションの表示・対話手段がカプセル化され、モデルはユーザインタフェースへの関連を極力排除した形で実現される。よってMVCモデルでは、アプリケーション本体をユーザインタフェース部から分離してモデリングできるという利点を持つ[1][2]。

しかし、MVCモデルにおけるビュー・コントローラは、実装時にはモデルに深く依存するために、アプリケーションの開発は、モデル・ビュー・コントローラそれぞれを連動させて行う必要がある。よって、ユーザインタフェース部のみの動作確認が困難となっている。

本研究では、アプリケーションの本質的なものを表すモデルとしてMVCモデルの「モデル」オブジェクトとそれを操作する「コントローラ」オブジェクトからなるモデル、それとはある程度独立して動作するユーザインタフェース部として、「ビュー」オブジェクトに「コントローラ」オブジェクトの機能を付随させたオブジェクトから構成されるモデルを考え、アプリケーションをMVCモデルのMC・VCの組からなるモデルに分けて考える。これにより、アプリケーション本体とユーザインタフェース部を極力独立させた開発・実行が可能となる。

本研究ではこのVCの組からなるモデルをユーザインタフェースモデルと呼び、その概要を次に示す。さらに、このユーザインタフェースモデルの開発支援環境を設計した。これは、ユーザインタフェース部のみの開発・実行を可能にし、ユーザインタフェース記述主導型のモデル開発を支援するものである。

3 ユーザインタフェースモデル

我々が扱う、ユーザインタフェースモデルの基底となるものは、MVCモデルの「ビュー」オブジェクトに、「コントローラ」オブジェクトの一部を付随させたオブジェクトからなるモデルである。

MVCモデルでは、「コントローラ」オブジェクトがユーザの操作によって発生したイベントを受け取ると、自分の属する「モデル」オブジェクトに対して参照・操作などの動作を行い、「ビュー」オブジェクトは「モデル」オブジェクトの状態が変化したことを伝えられて自分自身の再描画を行う。しかし、我々のユーザインタフェースモデルでは、そのイベントの発生によってアプリケーションの本体にアクセスする必要性を生じさせないようなイベントに対しては、そのオブジェクト自身が他の「ユーザインタフェースモデル」オブジェクトの属性を参照する、もしくはそれ

Description supporting system for UIM driven executable model

Nobuyuki Otsu, Shintaro Tsuji, Yoshikazu Ueda
Department of Computer and Information Sciences, Faculty of Engineering, Ibaraki University

に対して操作を行うといった局所反射的な動作をする。これは、「ビュー」オブジェクトに「コントローラ」オブジェクトの一部を付随させることで、「コントローラ」オブジェクトや「モデル」オブジェクトを必要としないことによる。

このユーザインタフェースモデルには、直線・円といった図形を描画するオブジェクト、ボタン・スクロールバーといったビューとコントローラをカプセル化したオブジェクトがあり、それぞれのオブジェクトがウィンドウの階層にあたる階層構造と他のオブジェクトへのメッセージ送受信関係を持つことができる。これらオブジェクトでモデルを構成することで、「モデル」オブジェクト、すなわちアプリケーション本体とは独立した動作を持つユーザインタフェースのモデル化が行える。

4 開発支援環境

ユーザインタフェースモデルの作成、および実行可能モデルの生成のための開発支援環境を構築した。この構成概念図を図1に示す。

この環境は、画面レイアウトやオブジェクトのメッセージ送受信関係を視覚的かつ対話的に定義するためのインタフェースビルダと、モデルの構成要素となるオブジェクト生成のためのクラスライブラリから構成される。

4.1 インタフェースビルダ

ユーザインタフェースモデルを構成するオブジェクトの実画面上の大きさ・位置の決定や、それぞれのオブジェクトのメッセージ送受信関係を、視覚的かつ対話的に定義できる環境として、インタフェースビルダを用意する。実際のウィンドウ階層と対応するオブジェクトの階層構造はオブジェクトのビューを構築する上で定義される。また、オブジェクトのメッセージの送受信関係の記述には、ユーザが関係付けるオブジェクト、条件・送信メッセージを選択することで行う。

さらに、編集されたオブジェクトは位置・大きさ・関係を持ったインスタンスを生成するC++コードに落され、それをコンパイル、実行することでユーザインタフェースのみの動作を確認することができる。

4.2 クラスライブラリ

前述のインタフェースビルダで記述されたオブジェクトを生成するためのC++ クラスライブラリを独自に用意する。ビューを表示する手段としてX-Windowを利用している。このクラスライブラリの各ウィジェットはMVCモデルの「ビュー」オブジェクトと同様の機能を持つものと、「ビュー」オブジェクトの機能に、「コントローラ」オブジェクトの機能の一部をカプセル化したものがある。

また、これらウィジェット上で生じたイベントを管理し、それぞれのウィジェットにコントロールを渡すために、「ビューマネージャ」というクラスを用意している。

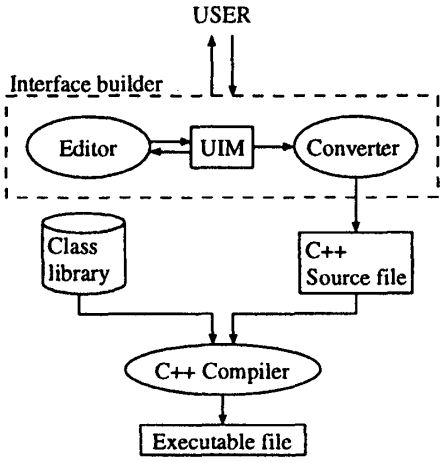

図 1: 開発支援環境の構成概念図

5 おわりに

以上、ユーザインタフェース記述主導型のアプリケーション開発を行うための、MVCモデルを改良したユーザインタフェースモデルについて述べてきた。これからの課題としては、インタフェースビルダの機能拡張とC++クラスライブラリの充実を行うことが挙げられる。さらに、アプリケーション本体を表す「モデル」との結合に関する方法を考察しなければならない。

参考文献

[1] 所, 松岡, 垂水:オブジェクト指向コンピューティング, 岩波書店, (1993.11)

[2] 増田, 笠原:Smalltalk-80 における拡張 MVC モデルとその応用, 情報処理学会論文誌, Vol.31, No.2, pp.259-267, (1990.2)