ウィンドウ設計における ウィンドウ・パネル間の関係の抽象化

6S－6

費　國寶
hi@asro.csk.co.jp
(株) アドバンスド・システム研究所

1．はじめに

現在、ウィンドウシステム上で応用プログラムのプロトタイプの開発を行っている。　ここでは、ウィンドウ上に一つ以上のイメージやテキストなど各種情報を表示するパネルを張り付けて機能の実現しており、またそれぞれのウィンドウの連携機能を組み合わせて、より複雑な機能を実現している。

本稿では、ウィンドウ及びパネル相互の関係を抽象化する事により、

・ウィンドウ及びパネル相互の関係の変更
・複数ウィンドウでのパネルの共有化などの機能
・独立したウィンドウ開発

を容易に実現させる設計手法について報告する。

2．背景

開発しているプロトタイプ・プログラムでは、一つのウィンドウには一つのパネルが張り付けられているのが基本であるが、場合によっては複数のパネルを張り付ける事が可能になっている。　また、各ウィンドウは単純な親子関係だけでなく、ウィンドウ間の連携機能を実現するため多重に関係づけられている場合も少なくない。

さらに、一つのパネルを複数ウィンドウでの共有や、パネルの多重化などを実現させる要求も出てきている。

従来の開発では、ウィンドウやパネルの指定を直接記述して行っているため、モジュールの独立性が低く、ウィンドウ単体を仕様に依存しない形での開発が出来難くなっている。　また、仕様の変更があった場合に、ウィンドウの組み込み順序やウィンドウとパネルの組み合わせを変える必要が生じた場合の対応が容易ではなかった。

3．設計モデル

まず、ウィンドウとパネルの機能を明確にするため、それぞれをオブジェクトとして管理する。　ウィンドウ間の関係とウィンドウとパネル間の関係はウィンドウ管理オブジェクトによって管理する。

Abstraction of Relation between Window and Panel on Window Design
Kokuhou hi
Advanced System Research Organization Ltd.

（1）ウィンドウ・オブジェクト

一つのウィンドウに対応するオブジェクトで、ウィンドウに関する必要な情報を持たせる。

（2）パネル・オブジェクト

通常は一つのパネルに対応するオブジェクトだが、パネルの共有化や多重化を行う場合は、複数のパネルと対応する事になる。　パネルを制御するために必要な情報を持たせる。

（3）ウィンドウ管理・オブジェクト

プログラム内の全てのウィンドウとパネルの関係を持たせる。

これらのオブジェクトの関係を図1に示す。

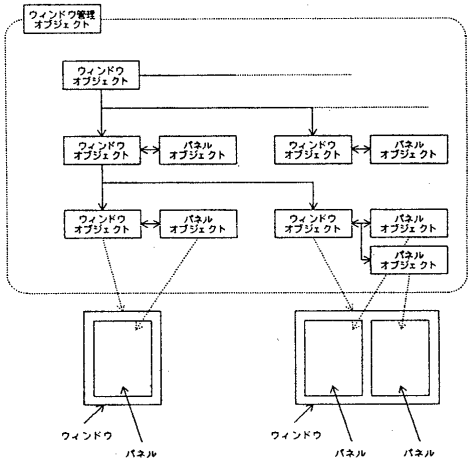

図1　ウィンドウ・パネルとオブジェクトの関係

実際の設計では、それぞれのオブジェクトの定義を行う事になる。

3．1　関係の定義

プログラムの外枠となる、作成すべきウィンドウそれぞれにＩＤを与える。　次にそれらのウィンドウの順序や親子関係を定義する。　さらにそれぞれのウィンドウにパネルを割り当てていく。　ここで定義される内容がそのままウィンドウ管理オブジェクトとして記述する事になる。

3．2　ウィンドウの定義

ウィンドウのデザイン、ウィンドウ自身やパネルに対する動作内容の定義を行う。　パネルの無いメニューの様なウィンドウの場合も、ウィンドウの定義だけは行う。

3．3　パネルの定義

実装すべき機能とパネルの関係の定義を行う。　パネルの動作定義は原則的にウィンドウ管理オブジェクトを通して、親子関係が定義されているウィンドウまたはパネルに対して指定が出来る。　但し例外として、ウィンドウに付けられたＩＤを用いれば直接指定する事も出来る。

4．応用例

これらの定義により、ウィンドウとパネルの機能が明確に分離されるので、それぞれを組み合わせる事で、より複雑な機能を実現する事も容易に可能になる。

一つのウィンドウ上で一つのパネルの複数箇所を同時に参照したい場合には、一つのパネル・オブジェクトに複数のパネルを割り当てる事によりパネルの多重化が実現できる。　同様に一つのパネル・オブジェクトを複数のウィンドウのパネルに割り当てる事により、パネルの共有化を実現できる。（図２）

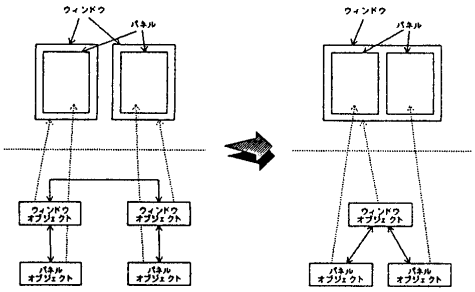

図２　パネルの多重化と共有化例

パネルの共有化は、大きなイメージをパネルに表示している時に、別のパネル上に縮尺して全体イメージを表示する様な場合に用いる事が考えられる。

5．　効果

この様に設計を行う事により、ウィンドウの関係が独立するとともに、パネル自体もウィンドウと独立する事が可能になる。　この事は、ウィンドウ間の関係に依存しないウィンドウ単体での開発やパネル単体での開発も可能になる。　仕様変更に伴うウィンドウ間の関係やウィンドウとパネル間の変更は、ウィンドウ管理オブジェクトの変更だけで行う事が可能になる。　変更例を図３に示す。　また、結果的に再利用が容易になり開発効率の向上にもつながると考えられる。

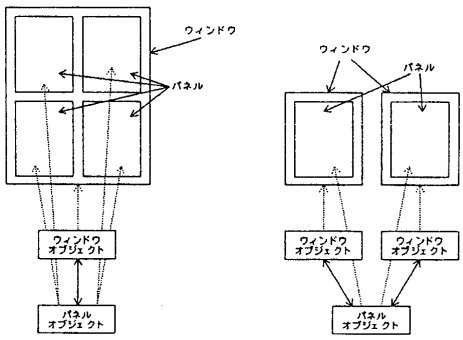

図３　ウィンドウとパネルの関係変更例

6．問題点と課題

実際にこの設計モデルに従って開発を行う上では、これらの機能を容易に実現するためのライブラリーの開発も必須となっている。　ライブラリーの支援がある事によりこの設計モデルの効果が発揮できると思われる。

今後は、設計の段階での関係の定義を支援ツール上で行う事により自動的に設計ドキュメントとウィンドウ管理オブジェクトを記述したソースコードを生成する様にして開発効率をより向上させたい。

7．おわりに

現在はここで述べた設計モデルに基づき、実際にプロトタイプ・プログラムの開発を進めている。　この結果、開発効率の向上とともにウィンドウ間に関する変更が容易になった事が確認された。　また、各種変更が容易になったので様々な実験を簡単に行う事が出きるようになり、プロトタイプ・プログラム開発としての成果がより良くなると期待できる。

【参考文献】

［１］Ｂ．Ｊ．コックス著／前川　守監訳：「オブジェクト指向のプログラミング」，トッパン，１９８８

［２］ＢＥＲＴＲＡＮＤ　ＭＥＹＥＲ著／二木厚吉監訳／酒匂　寛・酒匂順子共訳：「オブジェクト指向入門」，アスキー，１９９０

［３］Ｄ．Ａ．ヤング著／川手恭輔訳：「Ｘｔｏｏｌｋｉｔプログラミング」，トッパン，１９９０