

Retrovirus: 新しいウィンドウ操作メカニズム

神 田 陽 治†

ウィンドウを通常のテキスト処理によって操作できる, 新しいウィンドウ操作インタフェースを提案する. これは, 直接操作インタフェースであるウィンドウマネージャや, コンパイルを必要とするアプリケーションプログラミングインタフェースと並ぶ, 第三のウィンドウ操作インタフェースとして使用者に提供される. このウィンドウ操作インタフェースのプロトタイプ (Retrovirus) を, X11 ウィンドウシステム上に作成した. Retrovirus は画面上のどのウィンドウとも結び付くことができ, 相手のウィンドウはいつでも変更できる. また, 結び付いた先のウィンドウのウィンドウ制御データを読み書きできる. 読み出し時にウィンドウ制御データを複製して記憶する. 複製されたウィンドウ制御データは, 現実のウィンドウの状態とは切り離されているので, 複製後のウィンドウ制御データに自由に編集を加えることができる. 編集後のウィンドウ制御データを書き戻すことで, 編集結果を, 実際のウィンドウの状態に反映できる. さらに Retrovirus は, 複製したウィンドウ制御データをテキスト形式で公開するので, ユーザあるいは他のプログラムは, 通常のテキスト操作によりウィンドウ制御データを編集できる.

Retrovirus: A New Window Manipulation Mechanism

YOUJI KOHDA†

We propose a new window manipulation interface through which we can handle windows on the screen with ordinary text handling capabilities. It is used together with a window manager that is a direct manipulation interface and Application Programming Interface that needs compilation. A prototype named Retrovirus is made on the X11 Window System. The Retrovirus can hang on to any window on the screen and exchange the associated window any time. It can read and write the management data of the window. The window management data are read and duplicated. The duplicated data could be freely modified because they are isolated from the actual state of windows. The modification is reflected onto the actual state of the associated window, whenever writing back the data. Moreover, the Retrovirus offers the duplicated window management data to users and programs in textual format, and they can modify the data with ordinary text operations.

1. はじめに

ウィンドウ技術は, 現在のワークステーションにおけるユーザインタフェース技術の中核をしめ, 不可欠の技術とさえ言える. そこで, より良いユーザインタフェース技術の追求のなかで, ウィンドウ技術の改良は大きな価値を持っている. 本論文で我々は, 新しいウィンドウ操作インタフェースを提案する. 話を具体的にするために, Xウィンドウシステム上に, Retrovirus と名付けたプロトタイプを作成した. 本論文では新しいウィンドウ操作インタフェースが持つ基本的な機能を, プロトタイプの動作例によって説明した後, Retrovirus を用いて作成したウィンドウ操作インタフェースの実用例を紹介する.

2. 新しいウィンドウ操作インタフェースとしての Retrovirus

図1に, 既存のウィンドウ操作インタフェース—ウィンドウマネージャおよびAPI (Application Programming Interface)—と, Retrovirus が提供するインタフェースとの比較を示す.

ウィンドウマネージャは, 操作が容易なウィンドウ操作インタフェースを提供する. ウィンドウマネージャは直接操作の原理に基づいて作られており, ちょうど“机の上の紙”を扱うように, マウスを用いて画面上のウィンドウを操作できる. しかしマウスでは操作の正確性がしばしば犠牲になる. 例えば, ウィンドウの大きさや位置をマウスだけで正確に決めることは難しい. 具体的な数値表示による視覚フィードバックがあれば正確に位置付けたり, 大きさを制御できるが, あくまでマウスによる操作を行うのは人間であり, 作

† 富士通研究所国際情報社会科学研究所
FUJITSU LABORATORIES, IIAS

業には試行錯誤を要する。

一方、API はアプリケーションプログラマ向けに、プログラミング用に標準化されたウィンドウ操作インタフェースを提供する。API では正確性を得るのは容易だが、標準操作のレベルの低さをプログラミングで補う必要があり、使いこなすには技量が必要である。

従来のウィンドウ操作インタフェースに対して、Retrovirus は正確で扱いやすいインタフェースを提供することを狙っている。といっても、Retrovirus はウィンドウマネージャを置き換えるものではなく、ウィンドウマネージャとともに用い、ウィンドウマネージャの機能を補う位置付けにある。ウィンドウマネージャがワークステーションで一つ起動され、画面上のすべてのウィンドウを管理するのに対し、Retrovirus はいくつでも起動でき、一つの Retrovirus は一時に一つのウィンドウを管理する。Retrovirus とウィンドウの結び付きは固定されておらず、いつでも相手となるウィンドウを変更できることを特徴とする。さらに、一つの Retrovirus は、一つ分のウィンドウ制御データ（ウィンドウの大きさや位置などを直接制御しているデータ）を保存できる。そして、結び付けられているウィンドウからウィンドウ制御データを読み出

して保存したり、逆に、保存しているウィンドウ制御データを書き戻す能力を持っている。

ウィンドウマネージャは、ユーザからのウィンドウ操作要求や、API を通してのプログラムからのウィンドウ操作要求を、受けとった時点で即座に実行に移す。しかし、直接操作インタフェースであるウィンドウマネージャは、実行を終えた時点で受けとった要求を忘れてしまう。Retrovirus を使えば、ある時点のあるウィンドウに関するウィンドウ制御データを読み出して保存し、それを後で別のウィンドウに書き出し、過去のウィンドウの状態を再生することが容易にできる。しかもウィンドウ制御データはテキスト形式で保存されるので、ユーザは通常のテキスト操作により、編集を加えることが可能である。適当なインタフェースを準備することで、プログラムからもテキスト操作によって編集を行うことが可能である。これらの可能性は、新しい能力をウィンドウ操作環境にもたらす。例えば、過去のいくつかの時点での複数のウィンドウの状態に基づいて、新しい理想状態を計算によって求め、それを現在のウィンドウの状態に反映する、といったことができる。

同等な機能を、ウィンドウマネージャを作り変えることによって実現することは可能であろうが、ウィンドウマネージャの機能を肥大させることは好ましいことではない。ウィンドウ制御データの保存記録を利用する種々のサービスの提供は、さまざまなバリエーションの Retrovirus の提供という形で実現するほうが好ましい。また、ウィンドウマネージャの管理が及ぶ範囲は、一般には同一のワークステーション上にあるウィンドウに限られるが、Retrovirus にはその制約はない。ネットワークを越えて別のワークステーションにあるウィンドウと結び付くことにより、複数のワークステーションにあるウィンドウの状態を互いに関連させて扱うことが可能となる。さらにウィンドウマネージャとは異なり、一台のワークステーションで動かせる Retrovirus の数に制限はないので、Retrovirus 自らが別のワークステーションに移動したり、自分の分身を増やしたりが自由に行える。

本論文では、Retrovirus が持つ基本機能を、プロトタイプの動作例により説明し、その後、Retrovirus の基本機能を応用した、簡単な実用例を示す。Retrovirus のプロトタイプは、人間である利用者に向けたビジュアルインタフェースを持つ。ビジュアルインタフェースは、ウィンドウ制御データをテキストの形式

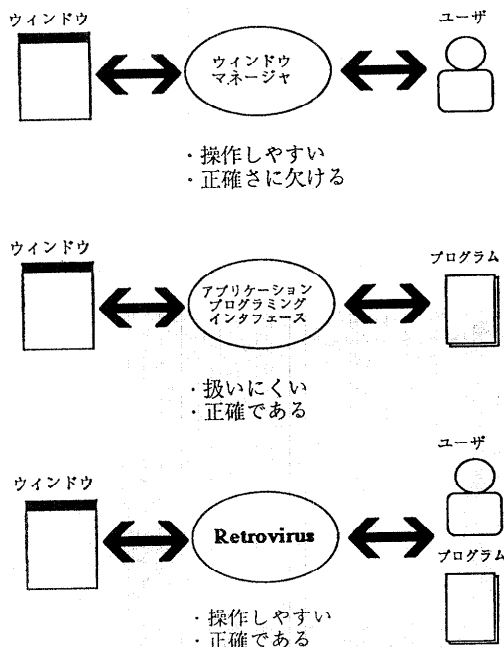


図 1 三つのウィンドウ操作インタフェースの比較
Fig. 1 Comparison between the three window manipulation interfaces.

で公開するので、キーボードなどを用いて正確かつ容易に編集でき、その結果、利用者は正確かつ容易にウィンドウを操作できる。実用例では、ウィンドウの過去の状態を記憶しておいて、後で再利用する手続きを簡単化する、実用的な Retrovirus を紹介する。

3. Retrovirus の基本機能

本論文では、ウィンドウ制御データという言葉を用いて、ウィンドウシステムがウィンドウの状態を制御するために保持しているデータ表現を指す言葉として用いている。これまでは、ウィンドウマネージャがウィンドウ制御データの唯一の管理者であり、ウィンドウ制御データは利用者に直接公開されておらず、ウィンドウマネージャにあらかじめ組み込まれたやり方でのみ操作できた。そして、利用者は好みのウィンドウマネージャを選択するという形で、ウィンドウ制御データの管理の方法を選択できたに過ぎなかった。

Retrovirus が提供するウィンドウ操作インタフェースの目的は、ウィンドウ制御データの管理を利用者に開放することにある。ところで、ウィンドウ制御データそれ自体は、個々のウィンドウの画面上の状態に直結しているから、ウィンドウ制御データを直接公開することはできない。直接公開すれば、利用者がウィンドウ制御データを編集している途中経過が、逐一、実際のウィンドウの状態変化として画面上に現れてしまう。また、ウィンドウマネージャが扱うウィンドウ制御データの表現は内部表現となっているために、そのままの表現で利用者に公開しても扱いにくい。

Retrovirus は、これら二つの問題に対処するために、ウィンドウ制御データを利用者に直接に操作させるのではなく、いったんウィンドウ制御データの複製をとり、利用者は複製を編集し、最後に編集結果を実際のウィンドウ制御データとして復帰させるという、三段階に分かれた方式を採用した (図 2)。複製を編集している間は、実際のウィンドウとは切り離されているから、編集経過が実際のウィンドウの状態変化として目に見えてしまうという問題を避けられる。また、複製を作るときには利用者のデータ表現へと、復帰させるときには元のデータ表現へと、ウィンドウ制御データの表現を適宜変換することで、利用者に扱いやすい表現でウィンドウ制御データを公開できる。

ただし、複製をとる操作は、不可分 (indivisible) に行われねばならない。同様に、復帰させる操作につ

いても、不可分に行われねばならない。これまではウィンドウマネージャがウィンドウ制御データの唯一の操作者であったから、ウィンドウ制御データを排他制御する必要はなかった。しかし、ウィンドウマネージャ

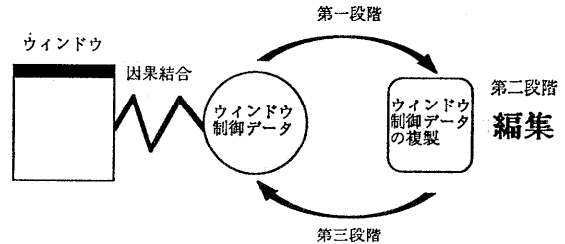


図 2 Retrovirus のウィンドウ制御データ編集方式
Fig. 2 The method to modify window management data.

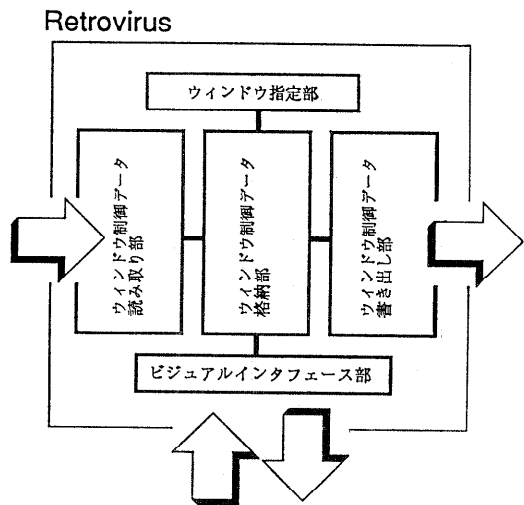


図 3 Retrovirus の基本構成
Fig. 3 The basic organization of the Retrovirus.

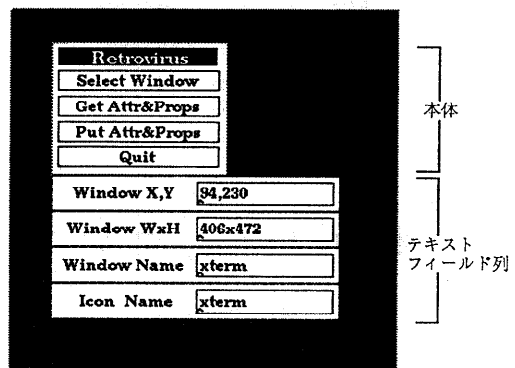


図 4 Retrovirus のビジュアルインタフェース
Fig. 4 The visual interface of the Retrovirus.

のほかに複数の Retrovirus が、並行にウィンドウ制御データを読み書きする状況では、ウィンドウ単位でウィンドウ制御データを排他制御する必要がある。例えば、ある Retrovirus がウィンドウ制御データを読み出している最中に、別の Retrovirus が同じウィンドウへウィンドウ制御データの書き出しを行うと、読み出したデータは前後で別の版のデータを読んだことになってしまう。それゆえに複製をとる操作は、不可分に行われねばならない。また、二つの Retrovirus が同じウィンドウにほとんど同時に書き戻し始めたとき、片方が他方を追い抜けば、書き込んだデータは前後で別の版のデータとなってしまう。それゆえに復帰する操作も、不可分に行われねばならない。

しかしながら、ここで述べたウィンドウ制御データの排他制御は、ウィンドウ制御データの一貫性制御に、不十分である（一貫性制御に関しては、例えば、文献 1）を参照）。例えば、ある Retrovirus があるウィンドウのウィンドウ制御データを複製し編集している間に、別の Retrovirus が同じウィンドウの状態を変更してしまえる。それゆえ、先の Retrovirus が編集結果を復帰したときに、最新のウィンドウの状態からすると一貫性がない状態になってしまうことがありうる。このような事態を防ぐためには、ウィンドウ制御データを共有資源として、ウィンドウマネージャと Retrovirus からのアクセスを、並行制御する必要がある。Retrovirus に並行制御の仕組みを組み込む価値はあるが、並行制御を実現するコストは高い

ので、実装には工夫が必要であろう。なお、Retrovirus の三段階によるウィンドウ制御データの編集方式は、並行制御の分野でシャドウページ (shadow page) と呼ばれる、アトミックな操作を実装する技法に対応し

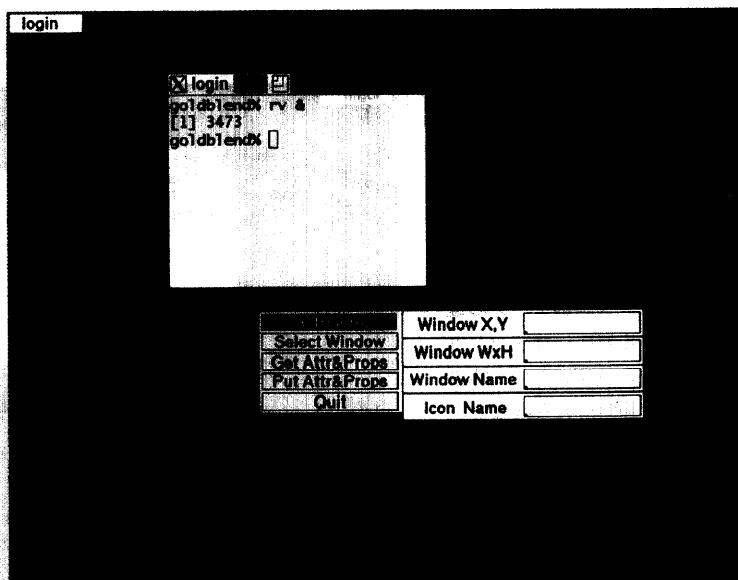


図 5 Retrovirus 起動直後
Fig. 5 Just after the start of a Retrovirus.

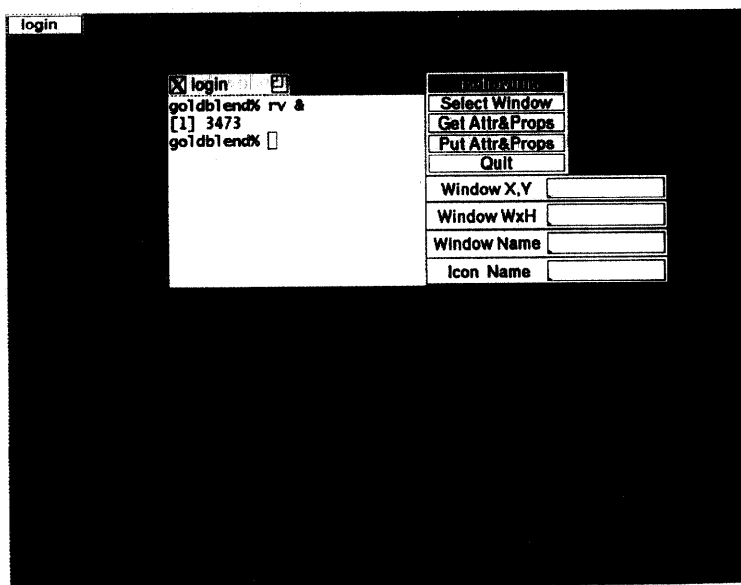


図 6 Select Window ボタンを使って、Retrovirus を画面内の一つのウィンドウに結び付けた
Fig. 6 Associating a Retrovirus with a window on the screen by Select Window button.

ている。また別の例では、ウィンドウ制御データの複製をデータラメな状態に変更して復帰させれば、書き込み後のウィンドウの状態は予想できない状態になりうる。これを防ぐには、書き込み時に、ウィンドウ制御データが書き戻して安全かどうかの検査をすればよいが、そのためには、ウィンドウ制御データの一貫性や安全性に関する意味付けを、はっきりさせておく必要がある。Retrovirus の場合には、ウィンドウ制御データの複製を復帰する際に、ウィンドウ制御データの項目ごとに、簡単な型検査と範囲検査を行っている。例えば、ウィンドウの大きさを表す項目に対しては、幅と高さが非負の整数であることを検査しており、ウィンドウの名称を表す項目に対しては、印刷可能な文字列であることを検査している。一つの項目でも検査が失敗した場合には、ウィンドウ制御データの復帰は、すべての項目について中止される。

4. Retrovirus の基本構成

Retrovirus の基本構成は、ウィンドウ指定部、ウィンドウ制御データ読み取り部、ウィンドウ制御データ書き出し部、ウィンドウ制御データ格納部、ビジュアルインタフェース部の五つの部分からなる (図 3)。ビジュアルインタフェース部の外観を図 4 に示す。基本構成の Retrovirus は一つの本体パネルと一組のテキストフィールドからできている。本体パネルには *Select Window*, *Get Attr&Props*, *Put Attr&Props* と *Quit* の各ボタンがある。*Quit* ボタンは Retrovirus を終了する。

その他のボタンの働きと、各構成部の働きは次の通りである。

ウィンドウ指定部

ウィンドウ制御データを読み書きする相手のウィンドウを指定する。ウィンドウの指定は何回でも行

表 1 Retrovirus が読み書きするウィンドウ制御データ
Table 1 Window management data read and written by the Retrovirus.

名 称	内 容	備 考
<i>Window X,Y</i>	ウィンドウの座標	左上角の <i>X,Y</i> 座標
<i>Window W×H</i>	ウィンドウの大きさ	幅×高さ
<i>Window Name</i>	ウィンドウの名称	タイトルバー等に付けられる文字例
<i>Icon Name</i>	アイコンの名称	アイコンに付けられる文字例

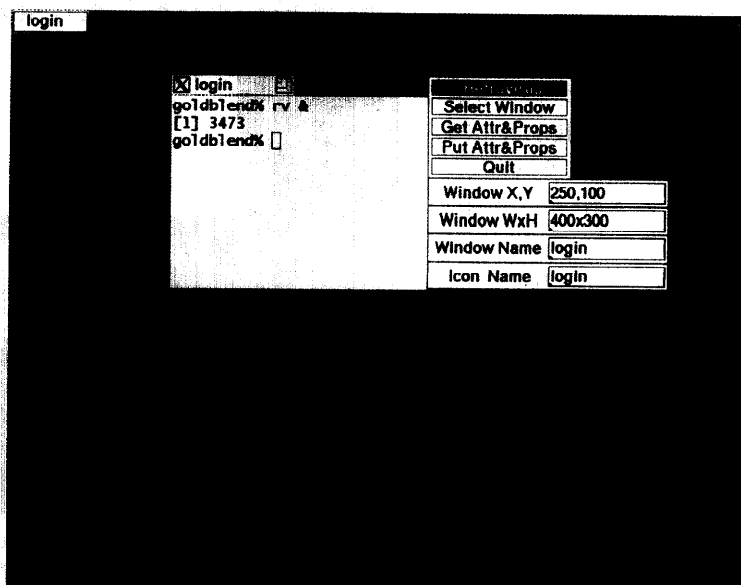


図 7 *Get Attr&Props* ボタンを使って、結び付けられたウィンドウからウィンドウ制御データを読み出した

Fig. 7 Reading out the window management data from the associated window by *Get Attr&Props* button.

表 2 ウィンドウ制御データ例
Table 2 Examples of window management data.

名 称	例 1	例 2
<i>Window X,Y</i>	250, 100	350, 200
<i>Window W×H</i>	400×300	500×600
<i>Window Name</i>	login	BIG LOGIN
<i>Icon Name</i>	login	LOGIN

ってよく、相手ウィンドウをいつでも変更できる。指定されたウィンドウは、ウィンドウ制御データ読み取り部とウィンドウ制御データ書き出し部で、操作の対象となる。本体パネルの *Select Window* ボタンによって、新しいウィンドウに選択し直せる。

ウィンドウ制御データ読み取り部

ウィンドウ指定部が指定するウィンドウから、ウィンドウ制御データを不可分に読み出し、テキスト表現に変換したのち、ウィンドウ制御データ格納部

へ送る。本体パネルの *Get Attr&Props* ボタンによって、データ読み取りが開始される。

ウィンドウ制御データ書き出し部

ウィンドウ制御データ格納部から、ウィンドウ制御データの複製を受け取り、それを内部表現形式に戻したうえで、ウィンドウ指定部が指定するウィンドウへ、ウィンドウ制御データとして不可分に書き出す。本体パネルの *Put Attr&Props* ボタンによって、データ書き出しが開始される。

ウィンドウ制御データ格納部

ウィンドウ制御データの複製をテキスト表現で記憶している。どのような情報がウィンドウ制御データとして扱えるか、または扱って意味があるかは、ウィンドウシステムやウィンドウマネージャに依存して決まる。あるアプリケーションが独自にウィンドウ制御データを設けて運用している場合には、そのウィンドウ制御データを読み書きすることにも意味が出てくる。

ビジュアルインタフェース部

人間を利用者として想定したインタフェースであり、ウィンドウ制御データ格納部に記憶されているウィンドウ制御データが、テキストフィールド列に表示される。各テキストフィールドにテキスト表示されたデータは、マウスやキーボードによる通常のテキスト操作で自由に編集できる。ビジュアルインタフェースは人間の利用者のためのものであり、使う場面を想定して、応用ごとに使い勝手の良いビジュアルインタフェースが工夫されて提供されてよ

い。

Retrovirus の構成は、以下の三つの特徴を持つ。

- Retrovirus とウィンドウの結び付きは固定していない

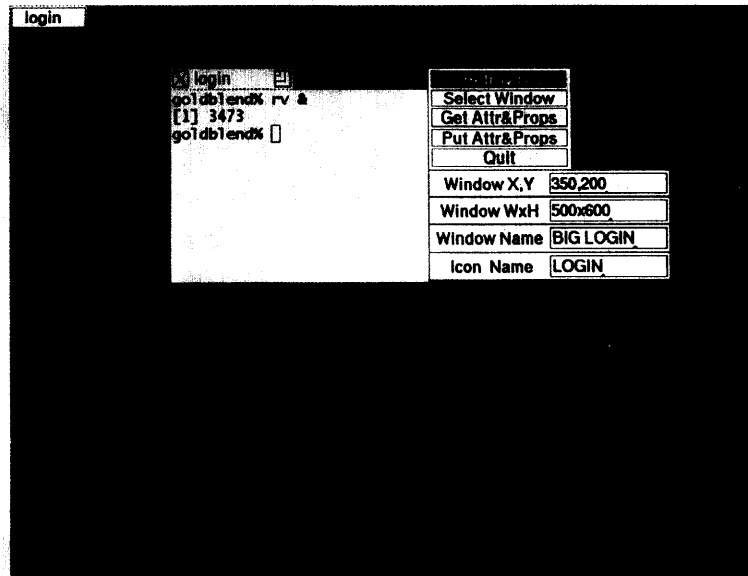


図 8 Retrovirus が保持しているウィンドウ制御データを編集した
Fig. 8 Editing the window management data kept in the Retrovirus.

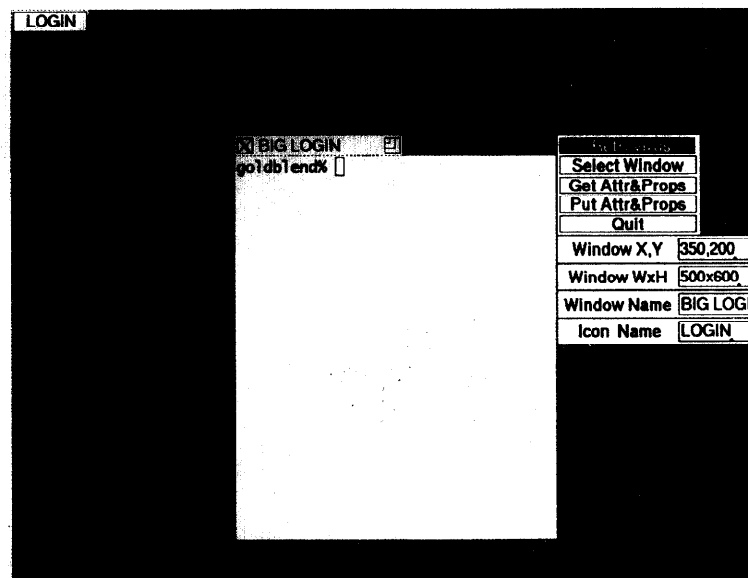


図 9 *Put Attr&Props* ボタンを使って、結び付けられたウィンドウへウィンドウ制御データを書き戻した
Fig. 9 Writing back the window management data into the associated window by *Put Attr&Props* button.

Retrovirus はいつでも相手のウィンドウを変更でき、ウィンドウと Retrovirus の組み合わせは固定されない。特に、複数の Retrovirus が、同一のウィンドウを同時に指定してもかまわない。

- Retrovirus はウィンドウ制御データを不可分に読み書きする
ウィンドウ制御データの読み取りと書き出しを不可分に行うことは、読み込み最中の、または書き込み最中の、ウィンドウ制御データへの排他的アクセスを保証する。
- Retrovirus は利用者ごとに適切な表現形式で、ウィンドウ制御データを提供する
ビジュアルインタフェースは人間の利用者のために、テキスト表現でウィンドウ制御データを提供する。その他、利用者がプログラムの場合にも、それぞれに適した表現形式で、ウィンドウ制御データが提供されてよい。

5. Retrovirus によるウィンドウ操作

Retrovirus の基本機能を説明するために、Xウィンドウシステム⁴⁾上にプロトタイプを作成した³⁾。ここでは、プロトタイプの動作例により、Retrovirus の基本機能を説明する。

プロトタイプの実装にあたって、考慮した点を述べる。第一に、試作したプロトタイプでは、操作の不可分性を Retrovirus の間どうしでのみ実現している(付録参照)。第二に、プロトタイプでは、X11ウィンドウシステムで標準的と考えられ

るウィンドウ制御データのみを扱う(表1)。すなわち、ウィンドウの位置 (*Window X, Y*), ウィンドウの大きさ (*Window WxH*), ウィンドウの名称 (*Window Name*), アイコンの名称 (*Icon Name*)

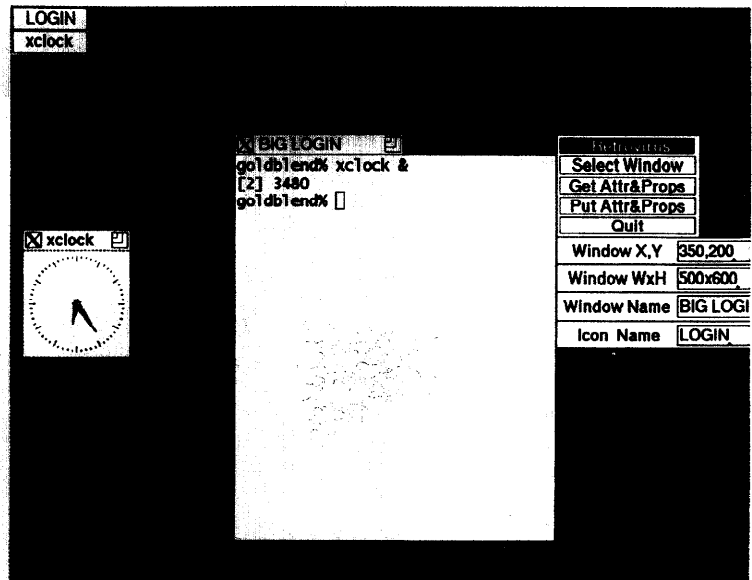


図 10 新しいアプリケーションを一つ起動して、ウィンドウを一つ生成した
Fig. 10 Starting a new application and creating a new window.

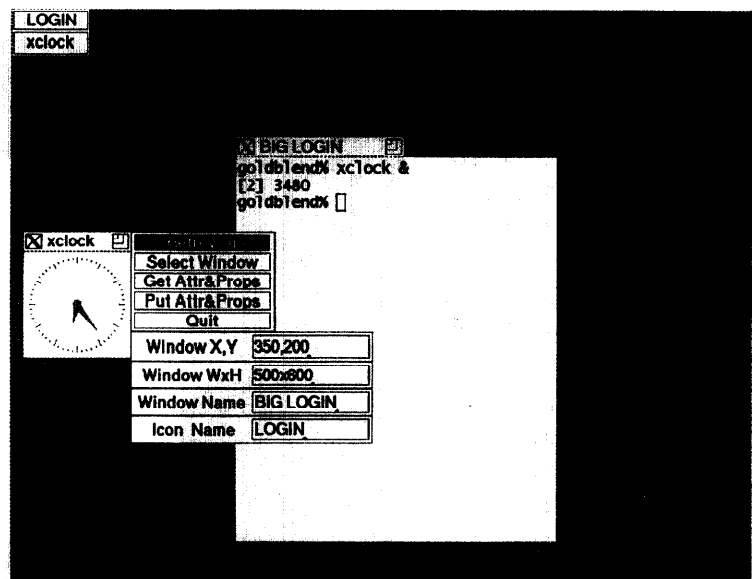


図 11 *Select Window* ボタンを使って、Retrovirus を新しいウィンドウに結び付け直した
Fig. 11 Re-associating the Retrovirus with the new window by *Select Window* button.

を読み書きする。

図 5 に起動直後の Retrovirus を示す。Retrovirus のプロトタイプは基本構成に忠実に、一つの本体パネルと一組のテキストフィールドからできている。なお説明の都合上、大きいフォントを使用しているために、図では Retrovirus 等が必要以上に大きい表示となっている。実際には、もっと小さいフォントでよく、Retrovirus ももっと小型にできる。これは、以後の図でも同様である。

次に *Select Window* ボタンにより、ウィンドウを選択する。Retrovirus は選択されたウィンドウの右端に“とりつく”(図 6)。すなわち、選択されたウィンドウの右隣に接する位置に自ら移動する。*Select Window* ボタンは随時使え、利用者は相手のウィンドウを好きな時に好きなウィンドウに変更できる。

Get Attr&Props ボタンにより、ウィンドウ制御データを解読して、テキストフィールドに読み出すことができる(図 7, 具体値を表 2 例 1 に示す)。同様に *Put Attr&Props* ボタンにより、テキストフィールドの値を内部表現形式に戻して、実際のウィンドウ制御データを更新することができる。例えば、テキストフィールドの値を、表 2 の例 1 から例 2 へ編集する(図 8)。この変更はキーボードを使って行った。次いで *Put Attr&Props* ボタンを押す。すると、ウィンドウが即座に反応する(図 9)。ウィンドウの位置、大きさ、ウィンドウの名称、アイコンの名称が一斉に変更される。

さらにもう一つ別のアプリケーションを起動し(図 10)、*Select Window* ボタンで相手ウィンドウを、いま起動したアプリケーションのウィンドウに変更し(図 11)、*Put Attr&Props* ボタンを押す。ウィンドウ制御データの内容が同一となるために、新しいほうのアプリケーションのウィンドウはもともとあ

ったウィンドウに完全に重なる(図 12)。同時に、ウィンドウの名称や、アイコンの名称も一致する。このとき Retrovirus は、ウィンドウ制御データを中身とする、コピーアンドペーストバッファの役割を果たしたといえる。

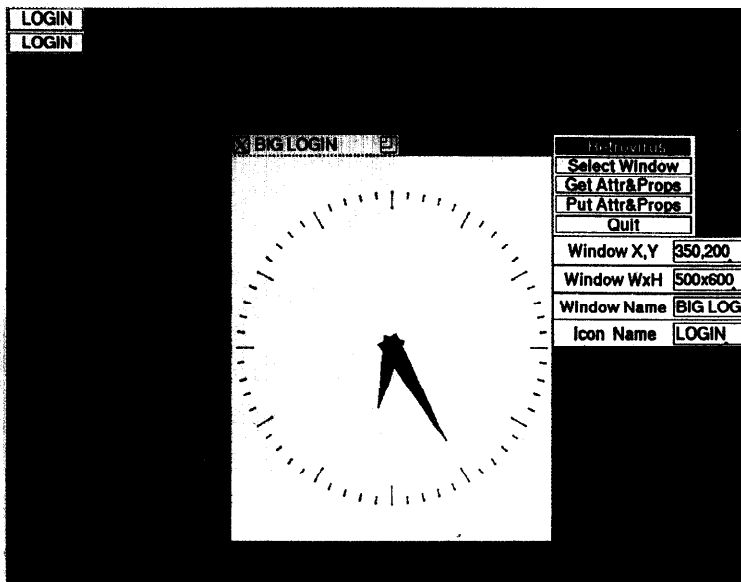


図 12 *Put Attr&Props* ボタンを使って、もともとあるウィンドウのウィンドウ制御情報を、新しいウィンドウに複写した

Fig. 12 Copying the window management data from the old window to the new window by *Put Attr&Props* button.

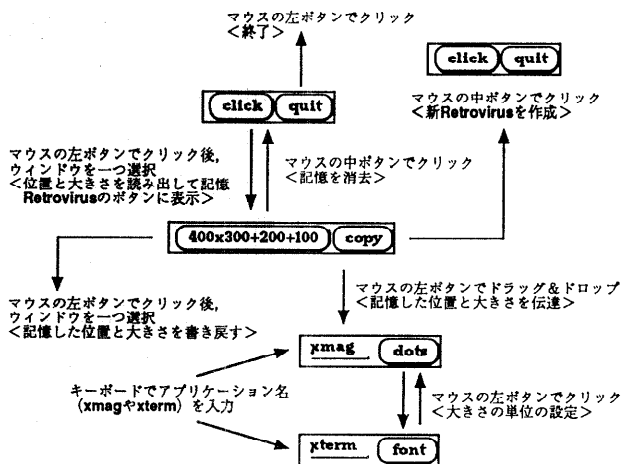


図 13 Retrovirus とアプリケーションバインダの操作一覧
Fig. 13 The operations map of the Retrovirus and the application binder.

6. Retrovirus を用いた実用例

多くのウィンドウシステムでは、アイコンと呼ばれる、図案的な表示を持つ小さいウィンドウが使われる。普通のウィンドウがアプリケーションの内容を表示するのに使われるのに対して、アイコンはアプリケーションプログラムやアプリケーションデータ、あるいは起動後のアプリケーションプロセスを指し示すために使われる。ユーザはアイコンを操作することで、アプリケーションの起動や、アプリケーションプロセスの中断、再開を指示できる。

ところで、Retrovirus は画面上に常時表示されているから、一種のアイコンのようなものと見なす。ただし、通常のアプリケーションを代理するアイコンとは違って、ウィンドウの状態を記憶するアイコンである。我々は、ウィンドウを代理する Retrovirus と、アプリケーションを代理するアイコンを併用することで、ウィンドウ利用の新しい可能性が開けると期待している。従来、アプリケーションを起動したときに開かれるウィンドウは、毎回マウス等で指定するか、アプリケーションごとに暗黙に決められたものが使われるのが普通だった。本論文では、Retrovirus が指定するウィンドウ情報を利用して、アイコンが指定するアプリケーションをマウス操作一つで素早く起動し、望むアプリケーションウィンドウをただちに開くことができる例を示す。

それには、プロトタイプの Retrovirus から必要な機能を残して、新しい Retrovirus の

変種を作り出す。プロトタイプは Retrovirus の基本機能をよく表すように作成されたもので、実際の応用では、使う場面を想定した使い勝手の良いビジュアル

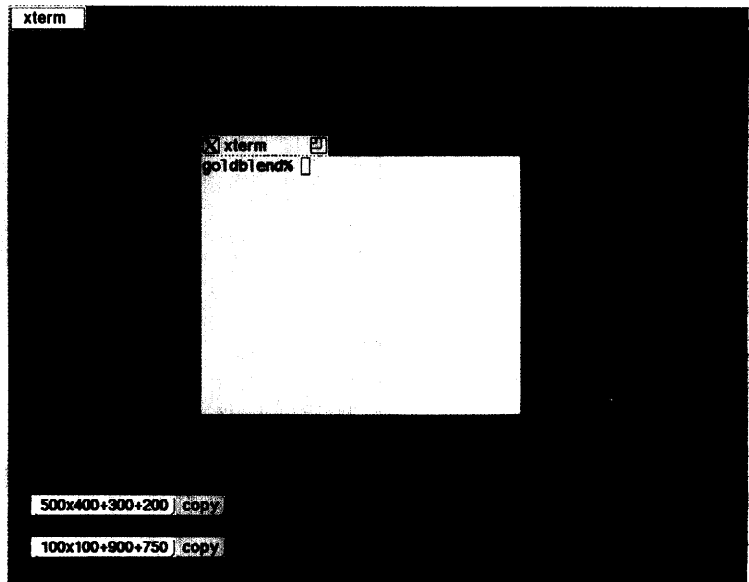


図 14 上側の Retrovirus が保存しているウィンドウ制御データで、ウィンドウの大きさと位置を設定した

Fig. 14 Adjusting the window size and location with the window management data kept in the upper Retrovirus.

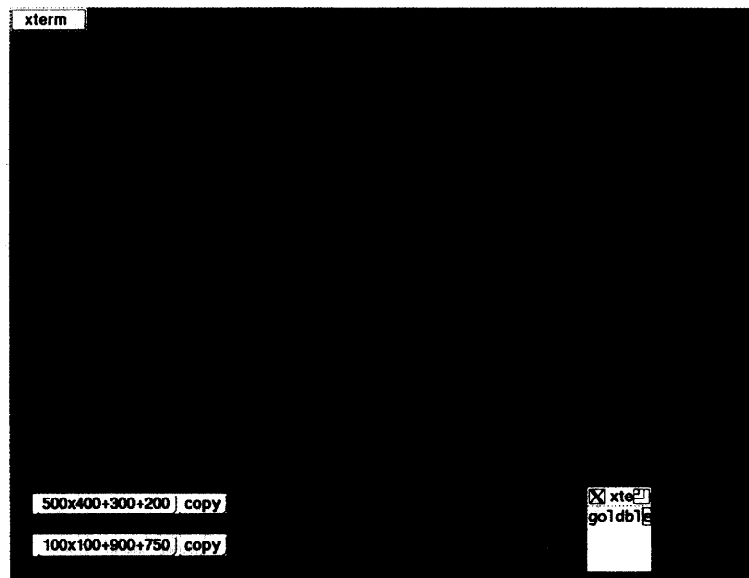


図 15 下側の Retrovirus が保存しているウィンドウ制御データで、ウィンドウの大きさと位置を設定した

Fig. 15 Adjusting the window size and location with the window management data kept in the lower Retrovirus.

インタフェースを工夫して用いる。新しい Retrovirus は編集可能なテキストフィールドを持たず、相手のウィンドウにまで移動しない。また、読み書きするのはウィンドウの大きさと位置だけである。さらに、Retrovirus からウィンドウ制御データを受け取り、アプリケーションを起動するプログラム（以後アプリケーションバインダと呼ぶ）も作成した。アプリケーションバインダは、アプリケーションプログラムを指すアイコンの役目をモデル化したものである。図 13 は、新しい Retrovirus とアプリケーションバインダの、操作を仲立ちとした関係を示している。

click と *quit* のボタンを持った小さいウィンドウが、起動直後の Retrovirus である。*quit* ボタンを押すと、Retrovirus は終了する。*click* ボタンをマウスの左ボタンでクリックすると、マウスカーソルの形状が変わり、利用者にウィンドウを選択するように促す。利用者がウィンドウを選択すると、選択されたウィンドウの大きさと位置を読み込んで記憶し、*click* ボタンの代わりに記憶したウィンドウの大きさと位置を刻印したボタンを表示する。このとき、*quit* ボタンは *copy* ボタンに代わる。続けて、ウィンドウの大きさと位置が刻印されたボタンをマウスの左ボタンでクリックすると、マウスカーソルの形状がまた変わり、利用者に再びウィンドウを選択するように促す。利用者がウィンドウを選択すると、選択されたウィンドウの大きさと位置は、ボタンに刻印された大きさと位置に一致させられる。画面上では、選択されたウィンドウの大きさと位置が、*click* ボタンによって保存されていたウィンドウの大きさと位置に、即時に再生されたかのように見える。Retrovirus 内の記憶が消去されるまでは、保存しているウィンドウの大きさと位置を、何度でも画面上のウィンドウに再生できる。記憶の消去は、ウィンドウの大きさと位置を表示したボタンをマウスの中ボタンでクリックすることで行え、再び、新たなウィンドウの大きさと位置を保存できる最初の状態

に戻る。

図 14 と図 15 には左下隅に、二つの Retrovirus が置かれている。二つともすでに大きさと位置が記憶されている状態になっていて、記憶値がボタンの上に刻印されている。この二つの Retrovirus を使うと、図 14 と図 15 を交互に、マウスの操作一回で行き来ができる。図 14 の状態で、下側の Retrovirus の、 $100 \times 100 + 900 + 750$ と刻印されているボタンをマウスでクリックした後、真中の大きなウィンドウを選択すると、ウィンドウの大きさは 100×100 に位置は $900, 750$ に変更されて、図 15 になる。図 15 の状態で、上側の Retrovirus の、 $500 \times 400 + 300 + 200$ と刻印されているボタンをマウスでクリックした後、右下の小さいウィンドウを選択すると、ウィンドウの大きさは 500×400 に位置は $300, 200$ に変更されて、図 14 になる。

もし、途中で第三のウィンドウ状態を記憶させたいと思ったときには、もう一つ Retrovirus を起動すればよい。これには *copy* ボタンを使う。*copy* ボタンをマウスの中ボタンでクリックすると、現在動いている Retrovirus とは別に、もう一つ新しい Retrovirus が起動される。

copy ボタンは、アプリケーションバインダとの連携にも用いる。*copy* ボタンをマウスの左ボタンでク

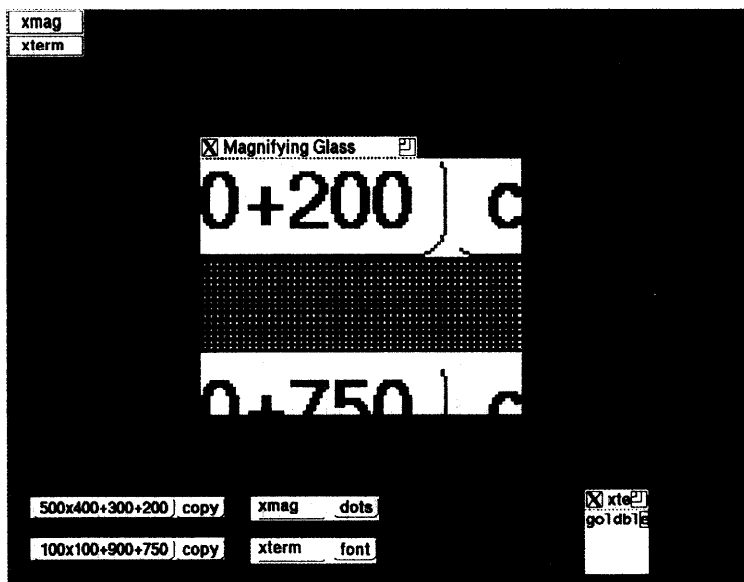


図 16 Retrovirus が保存しているウィンドウ制御データで、アプリケーションを起動した

Fig. 16 Starting new applications with the window management data kept in the Retrovirus.

リックすると、マウスカーソルの形状が変わり、ドラッグアンドドロップができる状態になる。ドロップされる内容は、記憶しているウィンドウの大きさと位置である。*dots* あるいは *font* と表示されたボタンを持つ小さいウィンドウがアプリケーションバイндаで、ドラッグアンドドロップの標的となっており、伝えられたウィンドウの大きさと位置をジオメトリ引数として、アプリケーションを起動する。起動されるアプリケーションは、ボタン左側にある、テキスト入力フィールドで指定する。また、X11 ウィンドウシステムでは、ウィンドウの大きさはアプリケーション自身によって、ドット単位あるいは文字単位に解釈される。アプリケーションバイндаは、ドット単位か文字単位かを知らないと、適切なジオメトリ引数をアプリケーションに引き渡せない。*dots* あるいは *font* と表示されたボタンは、どちらの型のアプリケーションかをアプリケーションバイндаに教えるボタンである。ボタンのクリックで、*dots* と *font* の表示は交互に入れ代わる。

図 16 には、先の二つの Retrovirus に加えて、*xmag* を起動するアプリケーションバイндаと、*xterm* を起動するアプリケーションバイндаが左下隅に置かれている。*xmag* は、ウィンドウの大きさをドット単位で解釈するアプリケーションであり、*xterm* は、文字単位で解釈するアプリケーションである。上側の Retrovirus の *copy* ボタンをマウスの左ボタンでクリックして、上側のアプリケーションバイндаにドラッグアンドドロップすると、*xmag* アプリケーションが起動され、図 16 の真中に示されるウィンドウが出現する。下側の Retrovirus の *copy* ボタンをマウスの左ボタンでクリックして、下側のアプリケーションバイндаにドラッグアンドドロップすると、*xterm* アプリケーションが起動され、図 16 の右下に示されるウィンドウが出現する。Retrovirus とアプリケーションバイндаの組み合わせは自由であり、Retrovirus に保存しておいたウィンドウの大きさと位置で、指定のアプリケーションをマウス操作一回で、即時に起動できる。ここでの Retrovirus は、ウィンドウ制御データを保存記憶するディポジトリの役目を果たしているといえる。

7. おわりに

ウィンドウを操作するための、新しいウィンドウ操作インタフェースとして Retrovirus を提案した。

Retrovirus は、ウィンドウを指定して、そのウィンドウ制御データを読み書きできる。相手のウィンドウはいつでも変更できる。読み出し時にウィンドウ制御データを複製して記憶する。複製後のウィンドウ制御データは、実際のウィンドウの状態と切り離されているので、複製されたウィンドウ制御データに自由に編集を加えることができる。編集後のウィンドウ制御データを書き戻すことで、編集結果を、実際のウィンドウの状態に反映できる。

また、Retrovirus の基本機能を持つプロトタイプを作成した。ビジュアルインタフェースを介して、利用者はウィンドウを、通常のテキスト編集操作で正確かつ容易に操作できる。また、このプロトタイプのビジュアルインタフェースを工夫して、ウィンドウの状態を保存して後で再利用できる、実際的な応用例を作成した。

Retrovirus の拡張をいろいろ考えることができる。ここでは三つの拡張を述べる。どれも、試作版がすでに稼働している。最初の拡張は、プログラムを利用者とする Retrovirus への拡張である。UNIX オペレーティングシステムの特徴の一つは、テキストフィルタを組み合わせるパイプ機構である。ウィンドウ制御データを標準出力に出力するインタフェースと、標準入力からウィンドウ制御データを入力できるインタフェースを持つ Retrovirus を作成した。この Retrovirus を介せば、UNIX オペレーティングシステムのコマンドレベルで、ウィンドウを制御できる。第二の拡張は、他のワークステーションの上にあるウィンドウを指定する能力を持ち、自由にワークステーションを切り渡れる Retrovirus への拡張である。この Retrovirus は、とりつき先のウィンドウを探索する機能を持っていることが特徴で、他のワークステーションの上にあるウィンドウまでも探索範囲に含めることができる。ウィンドウマネージャは一つのワークステーションの上のウィンドウのみを管理するのが一般的であるから、この新しい Retrovirus は、ウィンドウマネージャにはない可能性を持っている。第三の拡張は、他のプログラムから制御できる Retrovirus への拡張である。他のプログラムから Retrovirus をサブプロセスとして使うことができるように、アプリケーション間通信で命令を受けとれる Retrovirus を作成した。ウィンドウマネージャとは独立に、複数のウィンドウを制御するプログラムが、簡単に作れるようになると期待できる。例題として、スプレッドシートに似たビ

ュアルインタフェースを持ち、複数のウィンドウを関連付けて制御できるプログラムを試作してみた。

謝辞 Retrovirus の実装に関して、木下 彰氏の助力を得、また渡部 勇氏からは多数の有益な助言を受けました。

参 考 文 献

- 1) Bernstein, P. A. and Goodman, N.: Concurrency Control in Distributed Database Systems, *ACM Comput. Surv.*, Vol. 13, No. 2, pp. 185-221 (1981).
- 2) Dijkstra, E. W.: Cooperating Sequential Processes, In *Programming Languages*, F. Geunys (Ed.), Academic Press, N. Y., pp. 43-112 (1968).
- 3) Kohda, Y.: Retrovirus: A New Window Manipulation Mechanism, *FUJITSU IIAS Research Memorandum* IIAS-RM-91-9 E (October, 1991).
- 4) Scheifler, R. W., Gettys, J. and Newman, R.: *X Window Systems*, p. 701, Digital Press, Bedford, Massachusetts (1988).

付録 *Get Attr&Props* と *Put Attr&Props* 操作の不可分性の実装法

Retrovirus のプロトタイプは、X 11 ウィンドウシステム上で動くアプリケーションとして作成された。*Get Attr&Props*, および *Put Attr&Props* 操作の不可分性は、X 11 ウィンドウシステムの機能の枠内でできる範囲で実装されている。実装は、オペレーティングシステムの核でしばしば用いられる *lock* 命令と *unlock* 命令の手法による (*lock* 命令と *unlock*

命令については、例えば文献 2) を参照)。ウィンドウ制御データを読み出す部分と書き出す部分のプログラムコードをそれぞれ、*lock* 命令と *unlock* 命令に相当するプログラムコードで囲む。

X 11 ウィンドウシステムは、ウィンドウにプロパティと呼ばれる情報を自由に付与できるが、これを排他制御のフラグとして使う。プロパティを読み込むと同時に削除するリクエストを用いて、*lock* 命令を実装している。フラグのプロパティがウィンドウに付いているときに限り、*lock* 命令が成功する。付いていないときにはビジーウェイトする。*unlock* 命令は、フラグのプロパティを再びウィンドウに付け直すことで実装している。さらに、新しくウィンドウが生成されたときに、フラグのプロパティをウィンドウに最初に付ける初期化の作業が必要である。これは、特別な監視プロセスを常時一つ走らせておき、ウィンドウの生成時点でプロパティを付けることで実装している。

(平成 4 年 5 月 22 日受付)

(平成 5 年 5 月 12 日採録)



神田 陽治 (正会員)

1959 年生。1981 年東京大学理学部情報科学科卒業。1986 年同大学工学系研究科情報工学専門課程修了。工学博士。同年より富士通 (現富士通研究所) 国際情報社会科学研究所入社。ユーザインタフェース、グループウェア等の研究に従事。電子情報通信学会、日本ソフトウェア科学会各会員。