2K−7　応用プログラム群を統合するウィンドウ型ユーザインタフェースシステム：UAI／X

高濱徹行*　小倉久和*　中谷洋一**　中村正郎*

*福井大学工学部　**(株)永和システムマネジメント

1．はじめに

ウィンドウ環境は、①操作対象を直接画面上で指示できるため直感的に分かりやすく、②対象毎にどの様な操作があるかを覚える必要がなく、③イメージを活用するため視覚的に情報を捉えることができる等の長所があり、初心者にも使いやすい望ましい環境である。しかし逆に、①ウィンドウの機能を利用した応用プログラムを作成するのはかなり困難である②既存の応用プログラム、特に日常的に使用されるコマンド等がウィンドウ環境に対応しきれていない③ウィンドウシステムの機能を十分に利用した応用プログラムは通常の端末からは使用できない等の問題点がある。

これらの問題点を解決する一つの方法として、ウィンドウ環境に対応していない応用プログラム群を統合してウィンドウ環境に適応させるインタフェースシステムを構築することが考えられる。これによりウィンドウを意識した応用プログラムを作成しなくてもよい上に、通常の端末からも利用でき、ウィンドウ環境を生かした動作を実現する応用プログラムも得られることになる。

本研究では、応用プログラム群を統合してウィンドウ環境に適応させようとする開発者がなるべくプログラミングを行わないで済むように、ウィンドウ定義用の簡易言語を提供し、その中に応用プログラムの呼出しを埋め込むだけで上記のような環境を実現できるユーザインタフェースシステムUAI／Xについて述べる。

2．ウィンドウ型インタフェースシステムの問題点

ウィンドウ型インタフェース（WUIと呼ぶ）はユーザにとっては好ましいが、応用プログラムの開発者に取ってはかなり負担が大きい。このため、①標準のツールキットと規約を定め、応用プログラムの開発者がその規約に準じてツールキットを使用することでインタフェースを作成する方法②ユーザインタフェース管理システム（UIMS）を用意する方法がある。

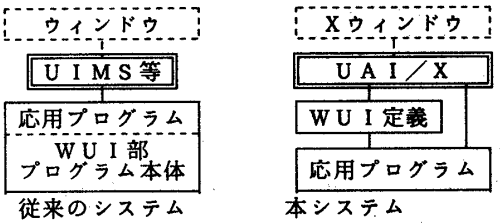


図1　従来のシステムとの比較

いずれの場合でも、既存の応用プログラムをウィンドウ環境に統合するためには応用プログラム中にWUIのための処理を記述しなければならない。本システムでは、図1に示すようにWUI部を応用プログラムから切り離すことによって応用プログラムとユーザインタフェースとの独立性をより一層高め、既存の応用プログラムを修正することなくWUIを達成することを目指している。

3．システム構成

本システムの構成を図2示す。

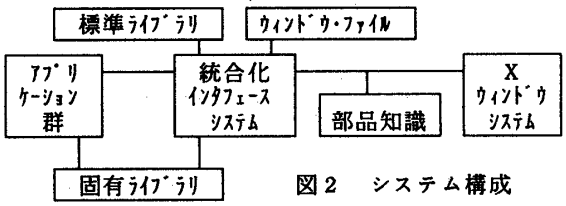


図2　システム構成

統合化インタフェースシステム本体は、ウィンドウ・ファイルに記述された内容に従って、マウスの移動やクリック等各種のイベントに対処するための準備を行ない、実際にウィンドウを開く。ウィンドウ・ファイルには、統合しようとする各応用プログラムに関係づけられるウィンドウ、通常は応用プログラムの実行結果等を表示するウィンドウについて、そのウィンドウの機能や表示位置・形状等の情報を記述する。また、マウスに関連して発生するイベントに対する処理や、応用プログラムの呼出しもここで指定する。

4．ウィンドウ・ファイルの記述

ウィンドウ・ファイルには、本システムが提供するウィンドウ記述用の簡易言語を用いてウィンドウに関する情報を記述する。記述は、①内部で使用する変数を指定する変数定義部②ウィンドウの機能・属性・親子関係，応用プログラム呼出しを指定するウィンドウ定義部③マウス・キーボード等の動作（アクションと呼ぶ）によるウィンドウの状態変化を指定するアクション定義部に分かれる。

ウィンドウ・ファイルの記述だけでテスト的にウィンドウを表示させることが可能であるため、画面のレイアウト確認と応用プログラムの統合の作業を独立して進めることもできる。

ウィンドウ定義部

ウィンドウの種類はクラス指定によって選択し、属性は属性名に対する属性値を指定することによって決定する。

実際の表記方法は、

```
window: ウィンドウ名
  class: ウィンドウのクラス
  属性名: 属性値
winend
```

Window Oriented Shell for Application Integration on X Window

Tetsuyuki TAKAHAMA*, Hisakazu OGURA*, Youichi NAKAYA**, Masao NAKAMURA*

*Fukui Univ., **EIWA System Management Corp.

である。クラスとしては、メニューを定義するmenuクラス、ボタンを定義するboxクラス等がある。属性名としては、ウィンドウの大きさを指定するwidth,height、表示位置を指定するx,y等がある。ウィンドウはさらに子ウィンドウを持つことがあるが、window:〜winendまでを入れ子にすることによって記述する。この時には省略された属性に対する属性値は親ウィンドウの属性が継承される。

応用プログラムの呼出しは、

属性名: <応用プログラム

により、応用プログラムの出力が属性値として扱われる。ウィンドウのデータを引数として用いるためには、

callback: 応用プログラム <ウィンドウ属性名

出力を他のウィンドウに表示するには、

callback: 応用プログラム >ウィンドウ属性名

と指定すればよい。以下に例を挙げる。

```
window:          ¦  window:
  class: list    ¦    class: command
  list: <ls      ¦    callback: echo Good >message
winend           ¦  winend
```

アクション記述部

主にマウスの操作によるウィンドウの変化を指定する。

```
action: アクション名
  キーボード・マウスの操作列: ウィンドウの動作
actend
```

のように記述する。なお、この記述はウィンドウ定義の外側で行わなければならない。マウスの操作列としては、マウスがウィンドウへ移動したことを表現するEnter、右ボタンの押下を表現するBtn1Down等がある。ウィンドウの動作としては、マウスでのショートカットを容易に行うためにウィンドウの再実行を行なうActivateを用意している。

どのウィンドウでどのアクションを採用するのかは、ウィンドウ定義部においてaction属性の属性値を指定し、

action: アクション名

で記述する。

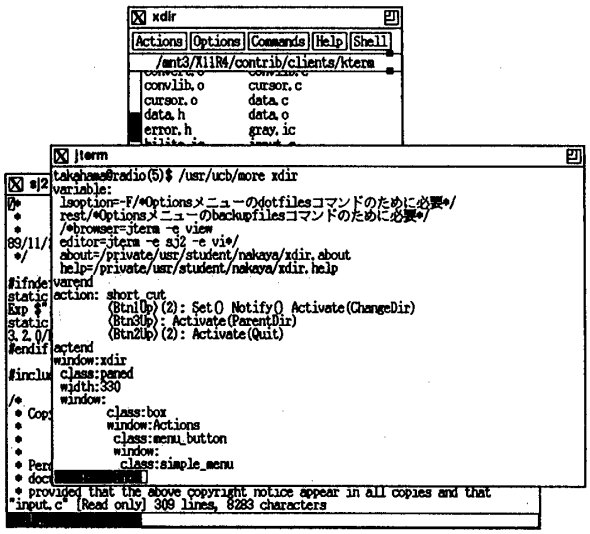
5．応用例

本インタフェースシステムを利用して、X11R4 のコントリビューションに含まれるファイル管理用の応用プログラムであるxdirと同等の機能を持つシステム（xdir/UAI）を構築した。ウィンドウ・ファイルの抜粋および表示例を右に示す。xdir/UAIでは、起動するエディタを簡単に変更でき、同時に複数のファイルを開くことができ、さらにシェル用のウィンドウを開くことができるという機能の拡張も行なっている。

簡単な評価のためにコーディング量を比較してみると、xdirがＣ言語で1703行に対してxdir/UAIは固有ライブラリのためにＣ言語で51行、ウィンドウ・ファイルが197行であり、1/6以下になっている。

6．今後の課題

本システムにより、比較的単純な応用プログラムの統合が実現できた。今後は、①対話型の応用プログラムを統合する機構②簡易言語を拡張して、ウィンドウをファイルのように扱う機能（すなわちウィンドウに対するリダイレクションやパイプの機能）を提供するシェル③ビジュアルな指定による簡易言語の自動生成④簡易言語のコンパイル等の実現を進めて行く予定である。




```
window:
 class:sme_bsb
 label:Copy
 callback:popup
          window:where
           class:dialog
           x:0
           y:0
           label:Where?
           value:
           grab_kind:nonexclusive
           window:Ok
            class:command
            callback:
             cp -r <list,selected <where >error.label
             update list
             popdown
            cbend
         window:Help
          class:command
          callback: jterm -e view $help
         winend
         window:Shell
          class:command
          callback: jterm &
         winend
winend
window:working_directory
         class:label
         label: <pwd
winend
window:
         class:viewport
         allow_vert:true
         window:list
          class:list
          list: < ls $lsoption $rest
          action:short_cut
         winend
         winend
winend
```

参考文献

1) 兜木昭男，木下凌一他：X-Window OSF/Motif プログラミング，日刊工業新聞社（1990）

2) 暦本純一，垂水浩幸他：エディタを部品としたユーザインタフェース構築基盤：鼎，情報処理，Vol.31，No.5，pp.602-611（1990）

3) 横山孝典，谷正之他：対話の階層モデルに基づくユーザインタフェース管理システム，情報処理学会論文誌，Vol.30，No.4，pp.507-517（1989）

4) 久野靖，角田博保：流れて行かないＵＮＩＸ環境，情報処理学会論文誌，Vol.29，No.9，pp.854-861（1988）