APPENDIX M

Working with Motif

Since neither OPEN LOOK nor Motif has a complete corner on the X market, there will always be some applications that you want to run that aren't available with the interface you want. Since Motif is dominant, though, a gradual transition to Motif seems probable. This appendix deals with some specifics of using applications written using the Motif toolkit in an OPEN LOOK environment, and the window manager and file considerations.

M.1 Working with Motif Applications

The Athena widget set provides X Toolkit applications with certain common features, many of which were described in Chapter 8. As explained in Chapter 1, an application coded using the Motif widget set has a different look and feel.

In this Appendix, we'll look at some of the features you're liable to encounter in a Motif application and learn how to use them. Most of these features are provided in a slightly different flavor by the OPEN LOOK GUI.

Most of the sample components we're using are taken from the *mre* demo program described in The Motif resource editor assists you in editing your own resource specification file, but it is primarily intended to demonstrate many of the Motif widgets.

The following sections mention the comparable Athena widgets where appropriate. Some of the Athena widgets are illustrated using the standard MIT clients in Chapter 8, *Other Standard Clients*, as well as in Appendix A, *The xterm/olterm Terminal Emulator*, and Chapter 9, *Graphics Clients*.

M.1.1 Dialog Boxes and Push Buttons

If you ask an X client for any drastic action, it is supposed to prompt you for confirmation. For example, the OPEN LOOK Window Manager will prompt you with a Notice window when you try to exit. Motif applications behave similarly. For example, if you were running the Motif window manager *mwm*, and try to "restart" it from the window manager's *Root* menu, you would see the dialog box pictured in Figure M-1.

Figure M-1. Typical Motif dialog box with two push buttons

A dialog box generally displays a message relevant to the application and requires a response from the user. In this case, the dialog box queries whether you really want to `Restart mwm?`.

A Motif dialog box contains one or more *push buttons* that allow you to respond to the message. (Many applications use push buttons; they are not confined to dialog boxes.) When a dialog is displayed and the default click-to-type focus is in effect, the input focus is usually switched to the dialog window. Until you respond to the dialog box, the application cannot continue. Once you respond to the dialog, the focus should switch back to the main application window.

Whether the dialog box contains one push button or multiple buttons, one button is always highlighted, generally by outlining. You can push the highlighted push button simply by pressing the Return key on your keyboard. To push another button, you must place the pointer on it and click the first pointer button.

A response might be a simple acknowledgment that you've seen the message: some dialogs feature only one button that reads **OK**. For instance, when you start the *mre* resource editor demo without a filename argument, the program looks for a file called *.Xdefaults* in your home directory. If *mre* cannot find the file, it displays a dialog box containing a message similar to:

```
Couldn't open /home/pat/.Xdefaults.
```

with an **OK** button. When a dialog has only one button, the button is always highlighted. Pressing Return or clicking the first pointer button on the **OK** button informs the client that you've seen the message and removes the dialog window.

Some responses request an action, such as proceeding with a previously invoked process, cancelling the process, or even exiting the program. The dialog box in Figure M-1 contains two push buttons labeled **OK** and **Cancel**. Pushing the **OK** button tells *mwm* to proceed with the restart process. The **Cancel** button gives you a chance to avert the restart process in case you invoked the command by mistake or have changed your mind. Since **Cancel** is highlighted, you can push it either by pressing Return or by using the pointer.

Whatever the message or potential responses, you react to a dialog box either by pressing Return (to push the highlighted push button) or by placing the pointer on one of the push buttons and clicking the first pointer button. Action will be taken if requested and the dialog box will be removed.

As we'll see, some Motif applications support another kind of push button called a *drawn button*. A drawn button is basically a push button decorated with a bitmap rather than text.

The Athena widget set provides comparable widgets to the Motif dialog box and push button. An Athena dialog box provides virtually the same functionality as a Motif dialog. The most obvious difference is that, in an Athena dialog, you must click on a command button to invoke it. The Return key shortcut only works with a Motif push button. See "Dialog Boxes and Command Buttons" in Chapter 9, *Graphics Clients*, for more information about Athena dialogs.

M.1.2 Menu Bars and Pull-down Menus

Figure M-2 illustrates the *menu bar* on a Motifwindow.

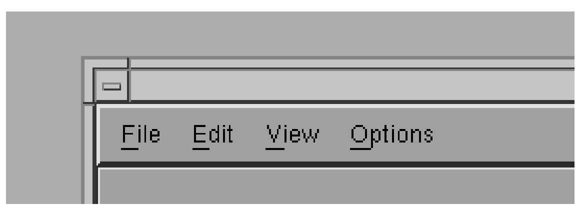

Figure M-2. Motif menu bar

A menu bar is a horizontal bar from which pull-down menus can be displayed. Each word on the bar is a menu title; you display the menu by placing the pointer on its title and clicking the first pointer button. The title becomes raised and highlighted by a box, the menu is displayed and the first selectable item is also raised and boxed. OPEN LOOK applications do not have a menu bar, but have separate main menus, usually in about the same position as a menu bar. One advantage of a menubar is that you can slide the pointer left or right and see all the available menus one after another.

Figure M-3 shows a Motif application's **File** pull-down menu.

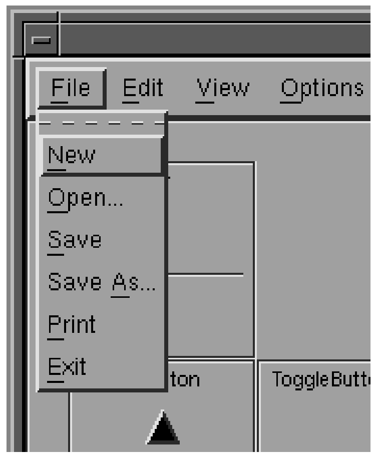

Figure M-3. Motif File menu

Notice that one letter of each menu item is underlined. That letter represents a unique abbreviation, or mnemonic, for the menu item, which can be used in a "mouseless" mode to select the item.

Notice also that each menu item has a keyboard shortcut, or *accelerator*, that appears in the right hand column of the menu. An accelerator can be used to invoke the action without displaying the menu at all (though they also work while the menu is displayed).

When you've displayed a menu by placing the pointer on the title and clicking the first button, you can select an item by:

- Placing the pointer on the item and clicking the first button.
- typing the mnemonic abbreviation for the menu item.
- Typing the accelerator key combination. (Though these are intended to save you the trouble of displaying the menu, they also work when it is displayed.)
- To select the boxed item (the first available for selection), you can alternatively press either the Return key or the space bar.

You can also display a menu from a menu bar by placing the pointer on the title and pressing the first pointer button. The menu is displayed as long as you continue to hold the pointer button down. To select an item, drag the pointer down the menu (each item is highlighted by a box in turn), and release the button on the item you want.

M.1.3 File Selection Box

Several Motif applications feature a widget called a *file selection box*, which allows you to select a filename from a list. Most modern OPEN LOOK applications provide similar dialogs; others provide this level of functionality by use of the familiar File Manager (see Chapter 4, *Using the OPEN LOOK File Manager*).

Using a file selection box is not exactly difficult, but it's not particularly obvious either. Let's consider the file selection box that is displayed when you select **Open...** from *mre*'s **File** menu illustrated in the preceding section. The **Open...** menu item is used to read a file into *mre*'s main edit window. When you select **Open...**, a second window is displayed—the file selection box illustrated in Figure M-4.

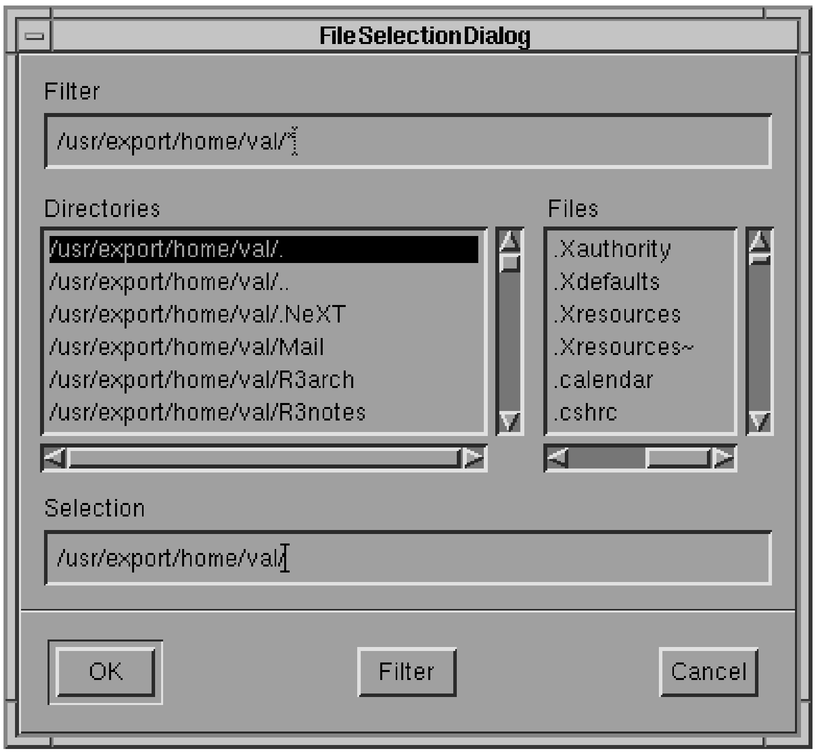

Figure M-4. A file selection box

Notice the window labeled **Selection** near the bottom of the box. You want to place the name of the file to select in this window. Initially this window contains an incomplete pathname—a directory is specified but no file. You can specify a file in a variety of ways.

Notice the two areas labeled **Directories** and **Files**. These are *list boxes* that are contained within the larger window. The **Directories** box lists the directories from which you can choose a file; the first directory is usually highlighted. The **Files** box lists the files within the highlighted directory.

Notice that the list boxes are bordered by horizontal and vertical scrollbars, which allow you to view text that is currently outside the box. (The Scrollbar widget is discussed in the next section.) A list box and its accompanying scrollbars form what is known as a Scrolled-

Window. The Motif ScrolledWindow is comparable to the Athena Viewport widget, discussed in Section 8.1.5, "Browsing Reference Pages: xman" on Page 194.

The file selection box allows you to select a file from any directory on the system, using various procedures. You can select a file from the list currently in the **Files** box; you can list the files in another directory currently displayed in the **Directories** box and select one of those files; or you can list the contents of an entirely different directory and select a file from that directory.

M.1.3.1 Selecting a File from the Files Box

To select a file currently in the **Files** box:

1. Place the pointer on the filename.
2. Click the first pointer button. The filename is highlighted by a dark bar; the letters appear in reverse video.
3. Notice also that the **Selection** window will be updated to reflect the filename; and the push button to confirm the selection (**OK** in many applications) will be highlighted, indicating that you can select the file by pressing Return.
4. Select the filename either by pressing Return or by placing the pointer on the **OK** push button and clicking the first pointer button.

When you select a file in *mre*'s file selection box, the file is read in to the initial *mre* editing window and the selection box disappears.

M.1.3.2 Choosing a File from another Directory in the Directories Box

To list the files in another directory in the **Directories** box and select one of those files:

1. Place the pointer on the directory name and click the first button. The directory name is highlighted. Notice that the box labeled s-1fHFilterfRs0 is updated to reflect the new pathname and the s-1fHFilterfRs0 push button at the bottom of the box is highlighted for selection.
2. Then, to display the contents of the highlighted directory in the **Files** box either:

- Press Return; or
- Click on the **Filter** push button.

To select a file from the updated **Files** box, follow the steps outlined previously in "Selecting a File from the Files Box."

M.1.3.3 Choosing a File from Another Directory on the System

You can specify an alternative directory from which a file can be selected by changing the filter, that is, the path in the **Filter** window (near the top of the file selection box). Initially the **Filter** window reflects the current working directory. In Figure M-4, the filter is */work/motif/demos/mre/** and the **Directories** box lists two directories:

```
/work/motif/demos/mre/.\"the current directory
/work/motif/demos/mre/..\"previous directory in the tree
```

To specify another filter, place the pointer within the **Filter** window and double click the first pointer button. The window becomes highlighted with a black bar (the text is visible in reverse video); now whatever you type will replace the current text.

When you type a pathname and hit Return (or click on the **Filter** push button at the bottom of the file selection box), the **Directories** box will be updated to reflect the filter you've specified. For example, if you enter the following pathname in the **Filter** window:

```
/home/pat/*
```

and hit Return or click on the **Filter** push button, the **Directories** box will be updated to reflect the directory */home/pat*, its subdirectories, and the directory above it in the tree. The first directory in the Directories box, */home/pat/.*, is highlighted and the files in that directory are listed in the **Files** box.

You can then choose any of the files in the **Files** box using the steps outlined previously in "Selecting a File from the Files Box."

M.1.4 The Motif Scrollbar

Each of the list boxes in the File Selection Box features both a horizontal and a vertical scrollbar. A vertical scrollbar is commonly used review text that has scrolled off the top of a window or extends past the bottom. In the case of *mre*'s **Files** box, the vertical scrollbar is used to scan a list of files too long to fit in the window at one time. A horizontal scrollbar is commonly used to view text or graphics that are too wide to fit in the viewing area. You'll probably encounter vertical scrollbars most often, as in Figure M-5.

Both the Motif and Athena widget sets provide scrollbar widgets. A Motif scrollbar operates differently than an Athena scrollbar, such as the one used by *xterm*. As you know, an Athena scrollbar is simple in design—just a rectangular thumb within a rectangular scroll region. Both parts are flat; the thumb is distinguished from the scroll region only by its (generally) darker color. The Athena scrollbar also operates differently than the Motif scrollbar. While a Motif scrollbar has separate parts to invoke different types of scrolling, the Athena scrollbar moves text according to which pointer button you use and how you use it. (See Appendix A, *The xterm/olterm Terminal Emulator*, for instructions on how to use *xterm*'s scrollbar.)

Now let's take another look at the **Files** box from *mre*'s file selection box, which is bordered by two scrollbars. A Motif scrollbar is comprised of four parts, labeled in Figure M-5: two *arrows* (one at either end of the bar), the *scroll region* between the arrows, and the *thumb*, the raised area which moves within the scroll region. The thumb displays the position and amount of text currently showing in the window relative to the amount saved. If text does not extend beyond the window, the thumb fills the entire scroll region. In Figure M-5, the thumbs in both scrollbars indicate that text extends beyond the bounds of the window.

Let's consider the pointer commands used to operate a vertical scrollbar. (You'll probably use a vertical scrollbar most often.) To scroll the text forward one window, place the pointer below the thumb and click the first button. To scroll the text back one window, place the pointer above the thumb and click the first button. Clicking on one of the arrows scrolls the text one line at a time: each click on a down arrow lets you view one more line

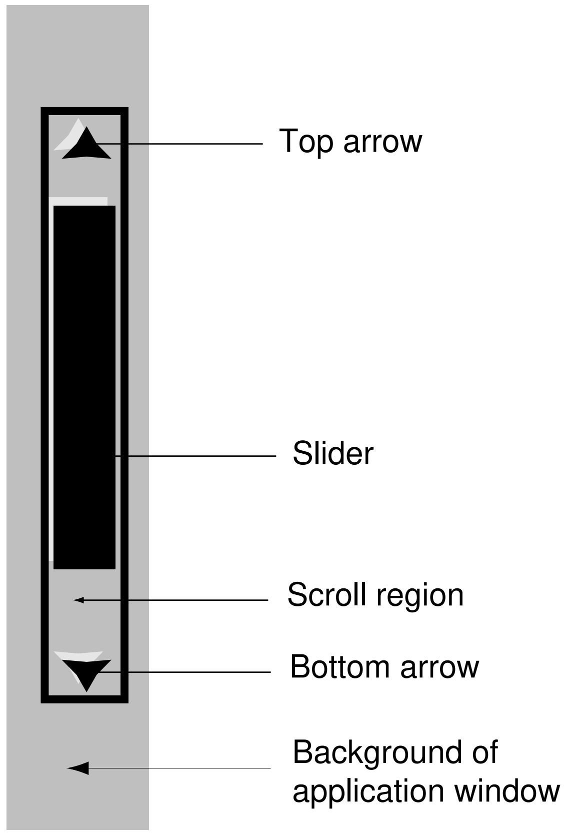

Figure M-5. Motif Scrollbar

of text at the bottom of the window; each click on an up arrow lets you view one more line of text at the top of the window.

A horizontal scrollbar lets you view the remaining part of lines that are too wide to fit in a single window. You use the same pointer commands to use a horizontal scrollbar as you do a vertical scrollbar; obviously the orientation of text and directions of movement are different. Clicking to the right of the thumb scrolls the text horizontally to the right. Clicking to the left of the thumb scrolls the text horizontally to the left. In Figure M-4, the **Files** box is displaying filenames only—the earlier parts of the pathnames are not in view. Notice that the horizontal scrollbar's thumb is all the way to the right of the scroll region. If you place the pointer to the left of the thumb and click the first button, the text is scrolled to the left to reveal the earlier parts of the pathname. Clicking on either arrow of the horizontal scrollbar moves the text one character to the left or right, depending on the direction of the arrow.

The unit scrolled when you click on an arrow depends on the application. Scrollbars are also sometimes featured on application windows that contain graphic elements rather than text. Obviously, such a window cannot be scrolled by text characters or lines. The *mwm* icon box, described in Chapter 13, *Customizing olwm*, can be scrolled the height or width of one icon.

M.1.5 Drawn Buttons

A *drawn button* is a push button decorated with a pixmap rather than a text label. Figure M-6 shows four drawn buttons from *mre*'s main window.

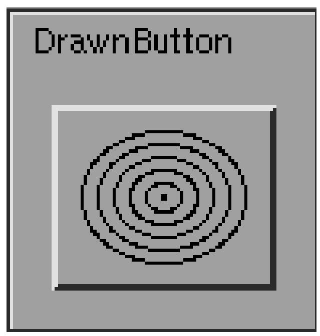

Figure M-6. Motif drawn buttons

The image on a drawn button should signal its function. If a program uses drawn buttons effectively, they can enhance an application's aesthetics.

mre uses drawn buttons well. The button decorated with an artist's palette tells *mre* to place resource specifications relating to color in the application's editing window. The button showing the letter "A" places resources specifying fonts in the editing window. The button featuring the mirror image of the arrows in reverse colors is a rather clever graphical representation of a difficult concept. The image is a sort of technical yin-yang symbol: it tells *mre* to place resources that can be toggled (turned on or off; set to be true or false, yes or no, etc.) in the editing window. Finally, the button featuring the eternity symbol tells *mre* to put all of the resources in the user's resource file in the editing window.

M.1.6 Radio Boxes and Toggle Buttons

A *radio box* is made up of a column of toggles (mutually exclusive choices). Figure M-7 shows four Motif radio boxes in *mre*'s Font Selection window.

Each column is one radio box; each box contains several diamond shaped *toggle buttons*. You push a toggle button by placing the pointer on the diamond symbol and clicking the first pointer button. The toggle button becomes darker (appearing as if it's been pressed). Actually, if you examine the button closely, the highlighting has just switched from the bottom edge to the top edge of the button. When you first make a selection from a column, the button and the accompanying text label are highlighted by a box. When you make a selection in another column, the highlighting box appears in that column (and disappears from the previous one).

Figure M-7. Motif radio boxes

Toggles in the same column are mutually exclusive. If you select one and then select another from the same column, the first one is toggled off. (The button appears to pop up—i.e., the highlighting switches back to the bottom edge of the button; also the highlighting box appears around the latest selection.)

M.2 Motif Applications under OLWM

Applications written for the Motif toolkit conform to the vendor-neutral ICCCM document (see Volume Zero, *X Protocol Reference Manual*) so they will operate with any ICCCM-conferment window manager, including *olwm*. It is thus possible to have a "hybrid" environment.

The most common problems we have seen have been related to "OSFkeysyms." If you get a batch of messages about keysyms beginning with the string "OSF" when starting a Motif client, ensure that your Keysyms file (*/usr/openwin/lib/XKeysymDB* or */usr/lib/X11/XKeysymDB*) contains a number of lines beginning with the string "OSF". If not, get a later copy from an X11R5 site or from OpenWindows 3.1.

If you are migrating to Motif, or sometimes use one interface and sometimes the other, you can simplify your life by using X Resources to make one behave more like another. For example, Motif normally uses Alt-F4 to kill an application. You can easily make olwm behave the same way by setting the resource

```
olwm.MenuAccelerator.Quit: F4+Meta
```

in your X defaults file. Similar customizations can be had by perusing Appendix K, *OPEN LOOK Mouseless Operations*

M.3 OPEN LOOK Applications under mwm

M.3.1 X-based clients

Applications written for the X-based OPEN LOOK toolkits generally conform to the vendor-neutral ICCCM document (see Volume Zero, *X Protocol Reference Manual*) so they will operate with any ICCCM-conformant window manager, including *twm* and the Motif window manager *mwm*. It is thus possible to have a “hybrid” environment

However certain features such as pushpins will not always operate. You may need to use the “f.delete” item in your *twm* or *mwm* menu in order to close an OPEN LOOK dialog window. Under *mwm*, you may be able to dismiss pop-up windows by double-clicking the left pointer button on the “menu” icon at the left of the titlebar.

As well, you may find that some Motif-based X servers are deficient in their supply of OPEN LOOK-specific fonts. If you get messages about missing fonts, see Appendix B, *OpenWindows and X11 Fonts*, which talks about adding fonts to the server.

M.3.2 NeWS-based clients

Applications written using the NeWS toolkit, of which there are very few[1], will only operate with an X server that has the “X/News” features; these applications are **not** compatible with a modern “Display PostScript” X server. As well, such applications will have OPEN LOOK-style titlebars and window menus, regardless of what window manager is in operation. It is recommended that you not run these programs under *twm* or *mwm*. Figure M-8, for example, shows the effect of running the NeWS-based OPEN LOOK FrameMaker under *twm* using the OpenWindows 3.0 server. Notice that FrameMaker and the window with this text in it have titlebars and a menu from the OPEN LOOK GUI, while everything else has *twm* titlebars. The same would happen with *mwm* or any other X-based window manager..

1. The only known applications that this applies to are the 3.1 OPEN LOOK version of FrameMaker, the Hyperlook and SimCity programs (which don’t use OPEN LOOK conventions anyway), and the Hexsweeper game mentioned in Appendix N, OPEN LOOK Programs.

M

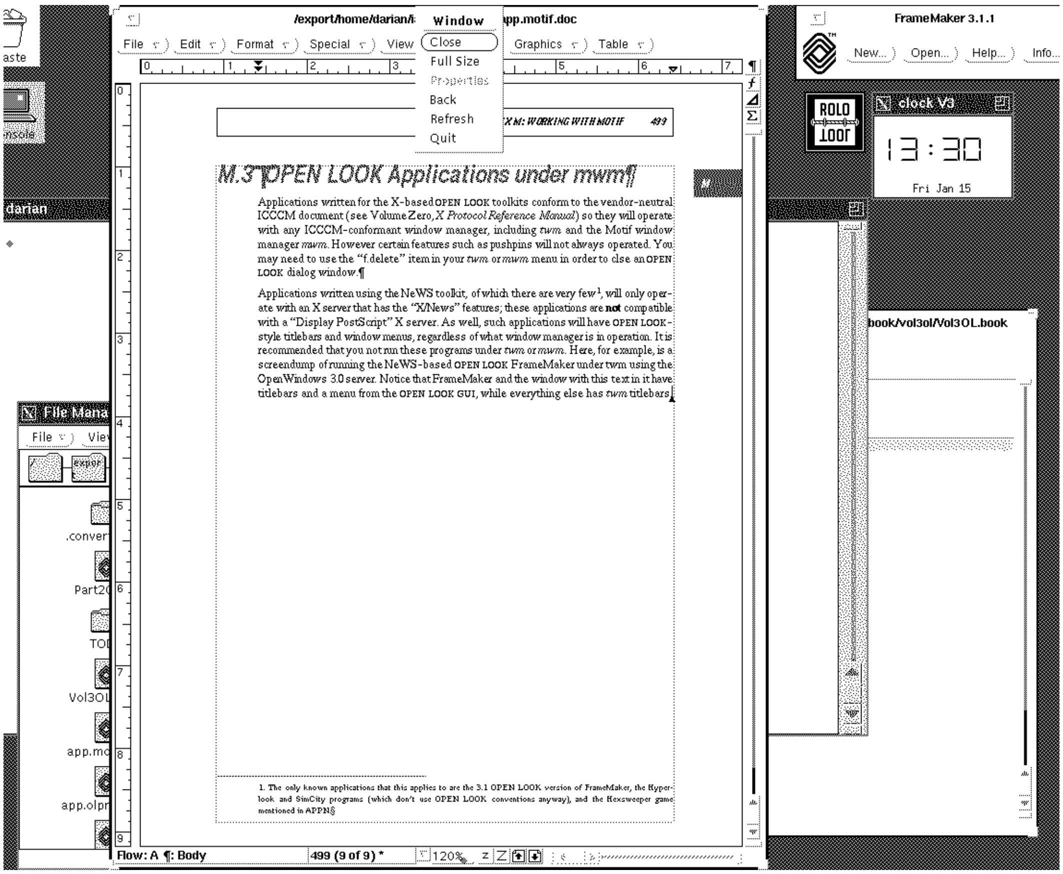

Figure M-8. NeWS Applications running under a non-OPEN LOOK window manager