
Structure

The structure of an interface refers to the windows, dialog boxes, and menus you use to create your interface. This section was developed with information from three sources: Microsoft's Web site (<http://www.microsoft.com/win32dev/uiguide/default.htm>); "The Windows Interface Guidelines for Software Design" published by Microsoft Press; and "GUI Design Essentials" by Weinschenk, Jamar and Yeo. Please refer to these publications for additional information or clarification. This document should be considered the "GUI Look and Feel Standard" to follow for structure issues in all application developed by the DENR ITS department. If an area is not covered here or any of the other three sections (Interaction, Presentation, Re-Use) of the Standards, then refer to one of the three sources mentioned above.

Primary and Secondary Windows

Windows are the backgrounds that the rest of your interface controls sit on.

Use cascading windows

Cascading windows (see Figure 8.1) keep users focused on one task at a time. However, if by cascading you cover up information that needs to be viewed simultaneously, allow for the use of tiling as shown in Figure 8.2.

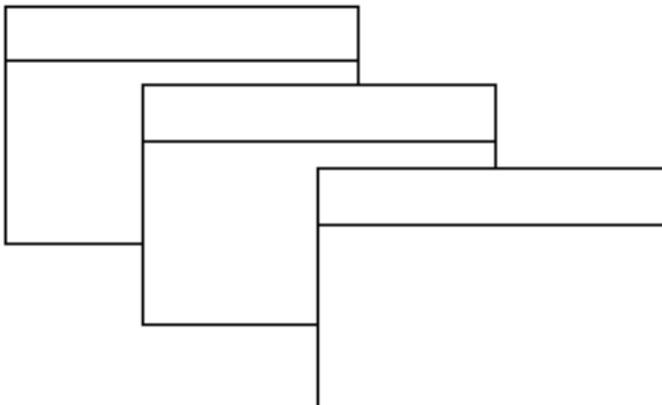


Figure 8.1 Cascading windows

Tiling windows allows users to display multiple windows at one time without covering up other windows. However, the more windows the user opens, the more the windows shrink to accommodate the additional windows. Therefore information can also be hidden when using tiling.

Figure 8.2 Tiled windows.

Avoid horizontal scrolling

Avoid having users scroll horizontally to see information in a window or dialog box. Instead of horizontal scrolling, try one or more of the following:

- A larger window
- Breaking the information up into more than one window or using tabs
- Allow expanding, zooming in, and collapsing to show only some information at a time

Size secondary windows to fit data

Size secondary windows to best fit the information in them. Do not rely on users to resize windows, even if the window allows it. All secondary windows *do not* have to be the same size.

Place pop-up windows in the center of the action

Place pop-up windows and dialogs in the center of the area they relate to in the application window.

Dialog Boxes

Dialog boxes allow users to complete a set of actions for a particular task.

Use modal dialogs for closure

A user must respond to a modal dialog box before they can perform work in any other box or window. Use modal dialog boxes for form filling and small, discrete tasks.

Use modeless dialogs for continuing work

A modeless dialog box is a dialog that can remain open and active even while users perform work in other windows or dialogs. Use a modeless dialog box for tasks that need to be repeated or monitored over time, for displaying different objects at the same time, or when the user needs to have access to menus or toolbars while the dialog box is open.

Tabs

Tab cards are a popular conceptual model in recent graphical user interfaces. Tab cards can be useful, but they also have their drawbacks.

Consider tab cards for discrete categories of information

Tab cards are useful as long as users can tell by a brief title which tab would include a particular piece of information (see Figure 8.3). Avoid using tab cards when you have a lot of general information, where you may end up with a cluttered card or multiple cards with headings like Personal and More Personal. These titles are not discrete enough for the user to know where to look.

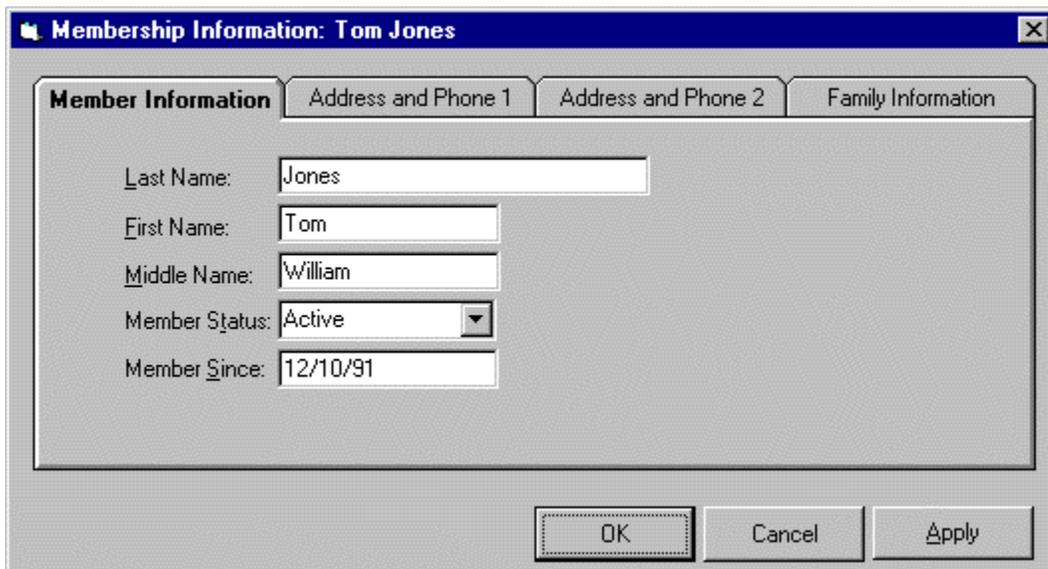


Figure 8.3 Don't split discrete information between two tabs. "Address and Phone 1" and "Address and Phone 2" are not discrete.

Tab card sets should relate to an object

A set of tab cards should relate to a specific object as shown in Figure 8.4. For example, one set of tab cards might contain information for a particular person, and include cards titled Address, Hobbies, and

Training. This set would have a person as its object. You would not have cards in this set titled Printer Setup or Defaults (see Figure 8.5). These cards might appear as a set for the Printer object.

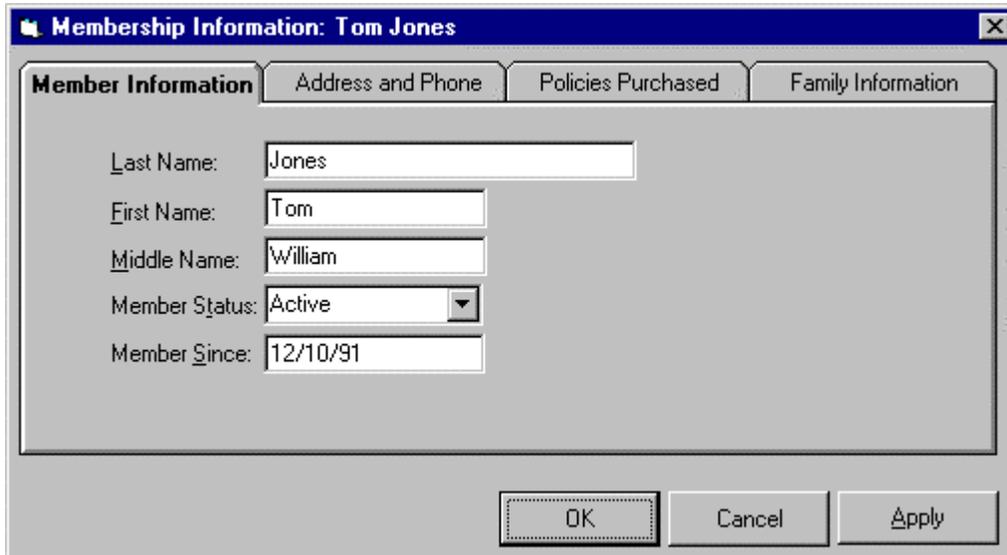


Figure 8.4 All tab cards relate to the object "Member."

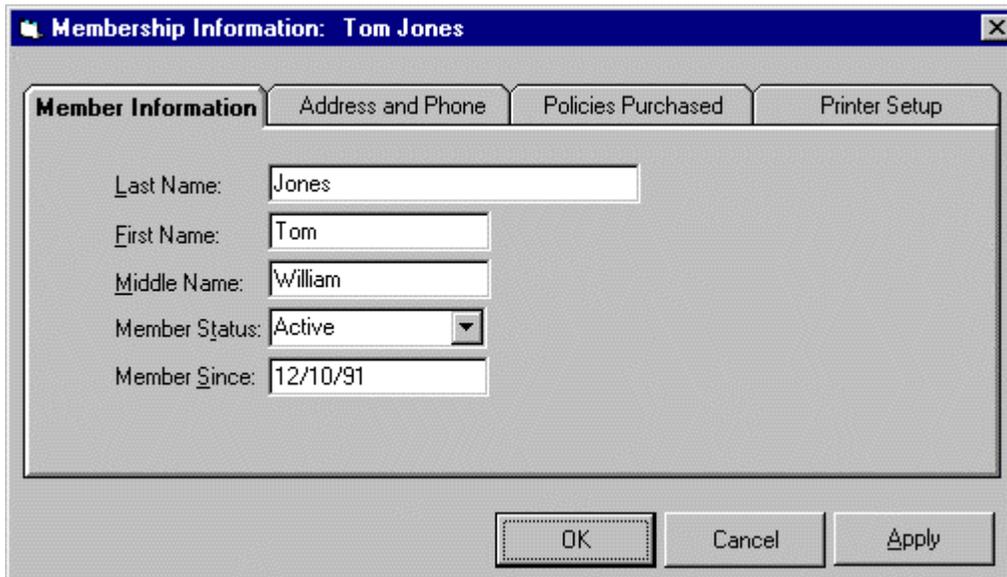


Figure 8.5 Don't mix objects within a tab card set. Tab card "Printer Setup" does not relate to object "Member."

Consider tab cards when the order of information varies

If the order in which information is viewed varies by user or by task, tab cards are a good way to organize information. However, if all users will view or use information in the same order, tab cards may be slower than other methods.

Make sure the information is independent

Avoid tab cards when the information on one card is heavily dependent on the information on another card (see Figure 8.6). Users will either have to keep flipping back and forth, or you will have to change data on the cards without the user knowing it has changed.

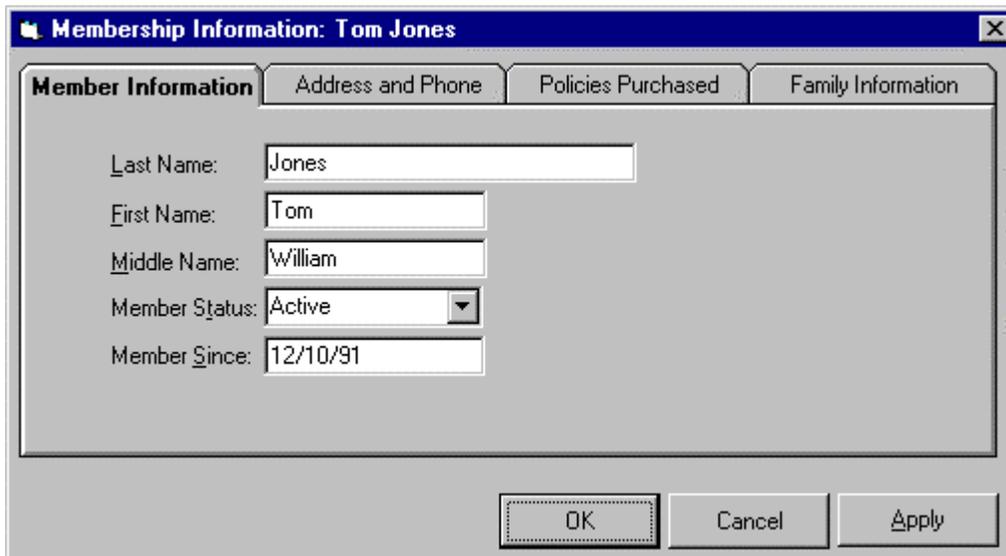


Figure 8.6 Information on each tab card is independent.

Use only one to two rows of tabs

Avoid tab cards if you have more than two rows of tabs—if you have that much information you need to consider a different method, such as menus going to forms or windows going to dialog boxes.

Use a master window or dialog box

A set of tab cards should reside on a window or dialog box (a "master") that also contains any buttons that affect the entire set of cards. Table 8.1 shows buttons that are commonly found on a master window.

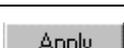
Button	Action
	Saves changes to any tab card in the set, closes the window
	Clears changes on any tab card in the set that were not invoked using Apply and closes the window
	Invokes changes made to any tab card in the set

Table 8.1 Buttons for a tab card master window.

Place buttons appropriately

If a button action pertains only to a specific tab card, place the button on the specific tab card. If a button action applies to the whole tab set, place the button on the master window.

Be consistent

Tab controls should be consistent within and across applications. If you use a master tab window and include OK, Cancel, and Apply buttons in one instance, you should use the same setup in the next instance.

Choose a horizontal or vertical flow

Decide whether you will use a horizontal or vertical flow of information for each tab card. Not every card in a set has to have the same flow—decide separately for each card.

A horizontal flow starts in the upper left and moves to the right. The most common or critical information appears in the top row. Less common or critical information appears in a second row. Buttons to control the window are on the top right. Use white space between rows to show the horizontal flow.

Vertical flow starts in the upper left and moves down. The most common or critical information appears in the left column. Less common or critical information appears in a second column. Buttons to control the window are centered on the bottom. Use white space between columns to show the vertical flow.

Navigation in Secondary Windows

With the mouse and pen, navigation to a particular field or control involves the user pointing to the field and clicking or tapping it. For button controls, this action also activates that button. For example, for check boxes, it toggles the check box setting and for command buttons, it carries out the command associated with that button.

The keyboard interface for navigation in secondary windows uses the TAB and SHIFT+TAB keys to move between controls, to the next and previous control, respectively. Each control has a property that determines its place in the navigation order. Set this property such that the user can move through the secondary window following the usual conventions for reading: in western countries, left-to-right and top-

to-bottom, with the primary control the user interacts with located in the upper left area of the window. Order controls such that the user can progress through the window in a logical sequence, proceeding through groups of related controls. Command buttons for handling overall window transactions are usually at the end of the order sequence.

You need not provide TAB key access to every control in the window. When using static text as a label, set the control you associated with it as the appropriate navigational destination, not the static text field itself. In addition, combination controls such as combo boxes, drop-down combo boxes, and spin boxes are considered single controls for navigational purposes. Because option buttons typically appear as a group, use the TAB key for moving the input focus to the current set choice in that group, but not between individual options -- use arrow keys for this purpose. For a group of check boxes, provide tab navigation to each control because their settings are independent of each other.

Optionally, you can also use arrow keys to support keyboard navigation between controls in addition to the tab navigation technique wherever the interface does not require those keys. For example, you can use the up arrow and down arrow keys to navigate between single-line text boxes or within a group of check boxes or command buttons. Always use arrow keys to navigate between option button choices and within list box controls.

You can also use access keys to provide navigation to controls within a secondary window. This allows the user to access a control by pressing and holding the ALT key and an alphanumeric key that matches the access key character designated in the label of the control.

Unmodified alphanumeric keys also support navigation if the control that currently has the input focus does not use these keys for input. For example, if the input focus is currently on a check box control and the user presses an alphanumeric key, the input focus moves to the control with the matching access key. However, if the input focus is in a text box or list box, an alphanumeric key is used as text input for that control so the user cannot use it for navigation within the window without modifying it with the ALT key.

Access keys not only allow the user to navigate to the matching control, they have the same effect as clicking the control with the mouse. For example, pressing the access key for a command button carries out the action associated with that button. To ensure the user direct access to all controls, select unique access keys within a secondary window.

You can also use access keys to support navigation to a control, but then return the input focus to the control from which the user navigated. For example, when the user presses the access key for a specific command button that modifies the content of a list box, you can return the input focus to the list box after the command has been carried out.

OK and Cancel command buttons are typically not assigned access keys if they are the primary transaction keys for a secondary window. In this case, the ENTER and ESC keys, respectively, provide access to these buttons.

Pressing ENTER always navigates to the DEFAULT command button, if one exists, and invokes the action associated with that button. If there is no current DEFAULT command button, then a control can use the ENTER key for its own use. Likewise, pressing ESC always navigates to the CANCEL command button, if one exists, and invokes the action associated with that button.

Menus

Menus play two critical roles in graphical user interfaces. They are a major method of navigation through the interface and they convey the mental model to the user in a snapshot. Giving attention to the design of usable menus is time well spent.

Word menu titles and items carefully

Pick names and test them to ensure that they make sense to users. It is not easy to pick labels that users will understand.

Change menus as you need

It is okay for menu bars and their drop-down menus to change as users move through an application.

Use initial capitals

Menu bar titles and items should have an initial capital letter with the rest of the word in lower case. For drop-down menu items, follow book title capitalization rules—capitalize the first letter of all major words.

Follow industry standards for menus

Follow industry standards on menu bars and drop-down menus. You do not have to use these menu bar titles or their drop-down menus if you do not have these tasks in your menu, but if you do use them, follow the standards. Figures 8.7 through 8.9 show different menus for Windows 95. If appropriate, provide shortcuts (hot keys) for frequent users. Use of hot keys will be consistent within the system. If you have any questions regarding a menu title or the sequence that they should be arranged, you can refer to any of the Microsoft Office products for examples.

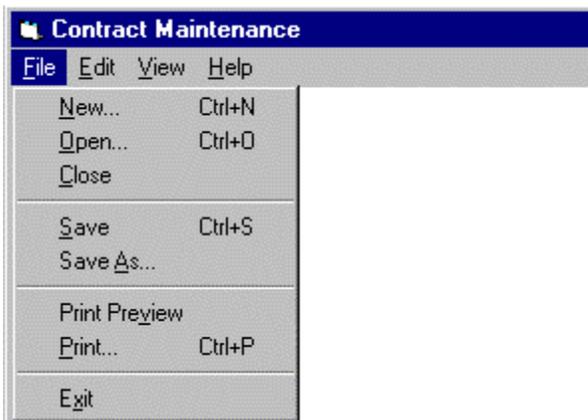


Figure 8.7 Windows 95 File menu.

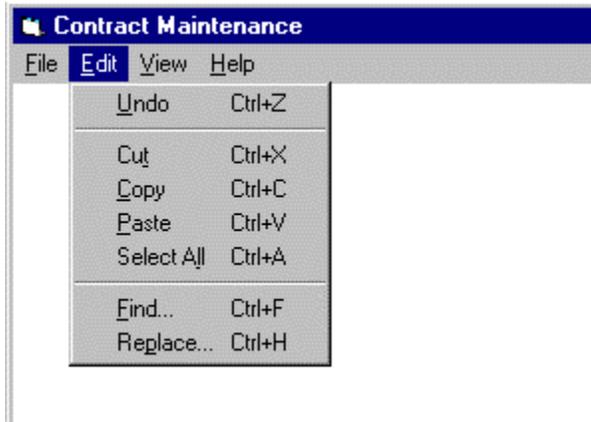


Figure 8.8 Windows 95 Edit menu.

Menu items are the individual choices that appear in a menu. Menu items can be text, graphics -- such as icons -- or graphics and text combinations that represent the actions presented in the menu. The format for a menu item provides the user with visual cues about the nature of the effect it represents.

Always provide the user with a visual indication about which menu items can be applied. If a menu item is not appropriate or applicable in a particular context, then disable or remove it. Leaving the menu item enabled and presenting a message box when the user selects the menu item is a poor method for providing feedback.

In general, it is better to disable a menu item rather than remove it because this provides more stability in the interface. However, if the context is such that the menu item is no longer or never relevant, remove it. For example, if a menu displays a set of open files and one of those files is closed or deleted, it is appropriate to remove the corresponding menu item.

If all items in a menu are disabled, disable its menu title. If you disable a menu item or its title, it does not prevent the user from browsing or choosing it. If you provide status bar messages, you can display a message indicating that the command is unavailable and an explanation why.

The system provides a standard appearance for displaying disabled menu items. If you are supplying your own visuals for a disabled menu item, follow the visual design guidelines for how to display it with an unavailable appearance.



Figure 8.9 Windows 95 Help menu.

Menu Bars

Menu bars point to major functionality in the application. Choose, organize, and name menu bar titles carefully.

Match menu bars to the users' workflow

When users look at a menu bar it should match how they think of their work. Spend time deciding on and testing menu categories to make sure they fit the users' mental model.

Give critical or frequent tasks even weight

Make sure all critical or frequent tasks are represented equally on the menu bar. Avoid grouping all critical or frequent items under one category, and then using the remaining five or six titles on the menu bar for different, but nonessential tasks.

Place application-specific menu titles where they fit

Place menu bar titles that are specific to your application where they best fit, for example, Jobs and Preferences before Help.

Use only one word for menu bar titles

Titles on the menu bar must be one word only. If they are more than one word, or use hyphens or dashes, it is hard to tell whether they are one item or two (Figure 8.16).



Figure 8.16 Don't use two words for menu bar titles.

Use only one line for the menu bar

Menu bars must be only one line long. If you have too many titles on your menu bar for one line, then collapse some of your titles into one. The menu bar in Figure 8.17 is too long.

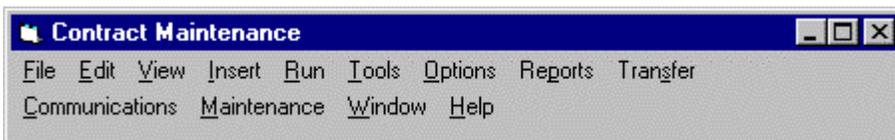


Figure 8.17 Don't use more than one line for the menu bar.

Do not gray out menu bar titles

Do not use graying out to make menu bar titles temporarily unavailable. Instead you should either not show the item at all or place the item on a drop-down menu where it is appropriate to use graying out.

Menu bar titles should always activate a drop-down menu

Menu bar titles should not initiate actions directly. Items on drop-down menus can initiate actions.

Drop-Down Menus

Drop-down menus reveal more detailed information to the user. Word and order them carefully.

Use more than one drop-down menu item

Drop-down menus should have more than one item on them. If you have a menu bar title that has one or no drop-down items, it should not be a separate menu bar category. Combine it with another menu title.

Use unique drop-down menu items

Do not start each drop-down item with the same word that is on the menu bar as shown in Figure 8.18. Drop-down items should be unique.



Figure 8.18 Don't start each drop-down menu item with the same word on the menu bar.

Limit drop-down menus to one screen in length

Drop-down menus can go from the top of the screen to the bottom of the screen. Do not use scrolling. If you have more items than will fit on the screen, you will need to combine some items and use cascading drop-downs, or separate some items into an additional menu bar item.

Put frequent or critical items at the top

Place the most frequent or critical items at the top of the drop-down menu.

Use separator bars

Whenever a menu contains a set of related menu items, you can separate those sets with a grouping line known as a separator. The standard separator is a single line that spans the width of the menu. Avoid using menu items themselves as group separators. Use separator bars in two ways—to group related items (see Figure 8.19) and to separate destructive items

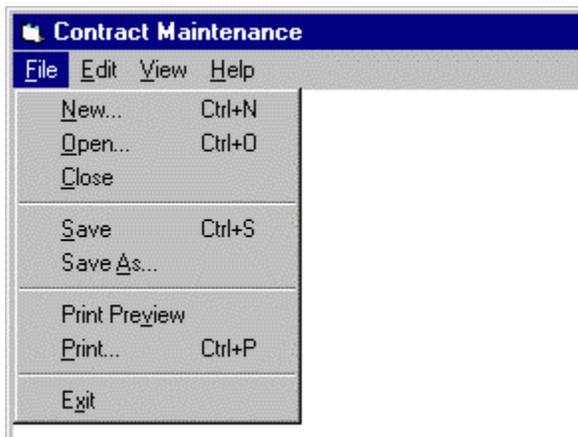


Figure 8.19 Separator bars used to group related items.

Use no more than two levels of cascading

It is okay to use cascading drop-down menus, but do not use more than two levels of cascading. Figure 8.20 illustrates the use of too many levels of cascading. Use the right arrow symbol (>) to the right of the drop-down menu item to denote that the item has a cascading menu.

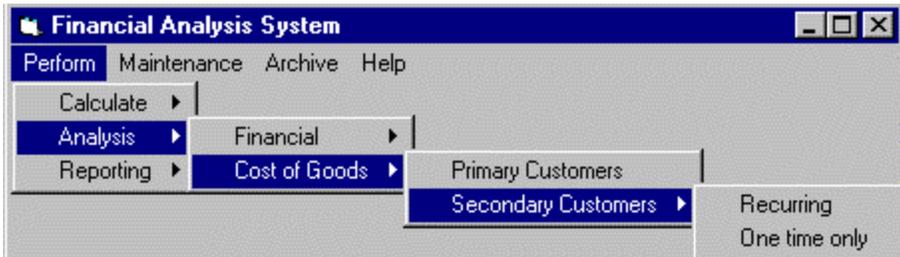


Figure 8.20 Don't use too many levels of cascading.

Use ellipses (...) to denote dialogs

If more input will be required to complete an action, use ellipses (...) after the drop-down item (see Figure 8.21).

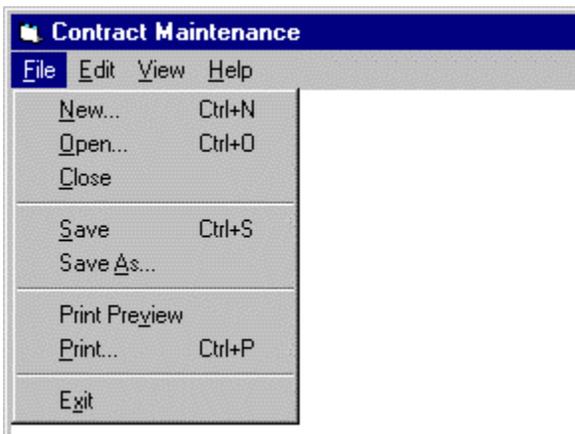


Figure 8.21 Ellipses after the "New" item indicate that more input will be required.

Use industry standard keyboard equivalents

A keyboard equivalent allows users to choose a menu item without using the mouse. A keyboard equivalent requires that the drop-down menu be open when it is used as shown in Figure 8.22. All drop-down menu choices should have keyboard equivalents.

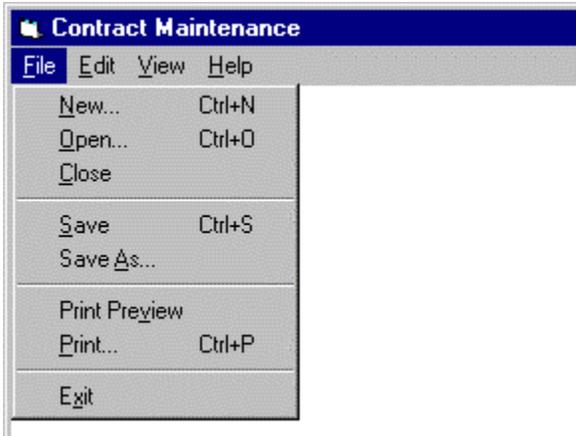


Figure 8.22 Keyboard equivalents show as underlined letters.

Use accelerators sparingly

Accelerators are combinations of keystrokes that allow users to choose a menu item when the drop-down menu is not open, as shown in Figure 8.23. Use accelerators only for those drop-down menu items that you think users will want to use without pulling down a menu, for example, Ctrl+V to paste from the clipboard. Figure 8.24 shows the standard Windows 95 Edit menu accelerators.

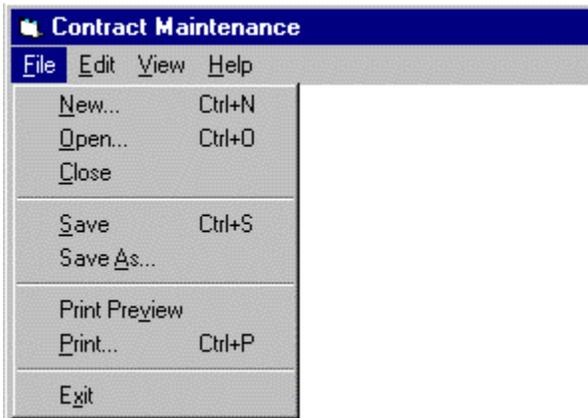


Figure 8.23 Accelerators show to the right of a drop down menu item.

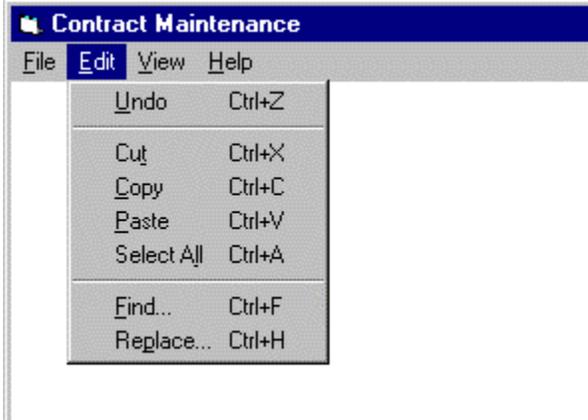


Figure 8.24 Windows 95 standard Edit menu accelerators.

Use consistent accelerators

Use consistent accelerators in your enterprise-wide applications. Place the accelerators in the drop-down menus to the right of the drop-down menu item.

Pop-Up Menus

Pop-up menus provide shortcuts for expert users.

Use pop-up menus for specific options

Pop-up menus appear when users click the right mouse button (see Figure 8.25). Use them for a subset of actions specific to the place or action. For example, if a user clicks the right mouse button on text in a word processing application, the pop-up menu would contain actions that could be taken on that text, such as Cut, Copy, Paste, and Formatting.



Figure 8.25 Pop-up menu.

Use redundant interactions

Don't make a pop-up menu the only place an action appears. Also provide a menu item, command button, or toolbar button for the action.

Roll-Up Menus

Roll-up menus can save space and are useful for expert users. Roll-up menus are floating, or movable, menus. They have a similar feel to toolbars. The user can activate them from a drop-down menu and place them anywhere in the workspace. By clicking the top right corner of the roll-up menu, the user can cause them to roll down (expand) or roll up (contract).

Use roll-up menus for frequent actions

Use roll-up menus for a group of actions that users will go back to frequently until they have the exact result they want, like applying special effects in a graphics application (see Figures 8.26 and 8.27).



Figure 8.26 Roll-up menu contracted.



Figure 8.27 Roll-up menu expanded.

Toolbars

Toolbars serve as a menu shortcut, or as a way to present controls that would be hard to convey in words, like drawing tools.

Make toolbars consistent

If you use a toolbar throughout the application, or between applications, make sure you use the same button graphics for the same functions throughout.

Make only active items available

Only toolbar items that are currently available should display. It is okay for some toolbar items to not show at all and for others to display as users move from one part of the application to another. It is all right for some items on a toolbar to be grayed out if they are only temporarily unavailable.

Allow users to move some toolbars

Allow users to move some toolbars to different locations on the screen to ensure they are out of the way of the work the users are performing.

Allow users to toggle toolbars on and off

Let users turn toolbars on and off through a dialog box or an option on a drop-down menu. This is especially important if you are providing more than one toolbar.

Allow customizing

Consider allowing users to customize their toolbars by deciding what to put on or take off the toolbar. You should, however, make decisions on what should be on the toolbar and provide that as the default. Most of the time, users should not have to customize a toolbar for it to be usable.

Use buttons with a purpose

Pay attention to the number of buttons on a toolbar. Too many buttons create visual and cognitive strain. Users will not see some of the buttons if there are too many.

Use tooltips

If you have the ability to include tooltips (the label that appears as the mouse is delayed over a toolbar graphic) then include them (see Figure 8.28). Don't substitute tooltips for good design.



Figure 8.28 Tooltip on a toolbar item.

Group like items

If you have a lot of buttons on a toolbar, consider grouping them. For example, place all editing graphics together. Use white space to group them.

Relationship between Toolbars, Command Buttons, and Menus

You will need to decide which actions go in the menu system, which go on the toolbar, and/or which actions should be buttons on a window (see Table 8.2).

Use toolbars for frequent actions across screens

Toolbars should contain actions that users need to take frequently and need to access across several screens. Do not use toolbars instead of command buttons.

Action Type	Proper Placement
Most frequent and critical	Command buttons
Fairly frequent and across several screens	Toolbars
All actions: frequent, critical, and infrequent	Menu bar and drop-down menus

Table 8.2 Considerations for proper placement of actions.

Use toolbars to supplement menus

Some toolbar items are used in conjunction with menu bars when users need a shortcut for certain actions. In these cases the toolbar items also appear on the menu bar.

Use toolbars in place of some menu items

Some toolbar items can be used in place of menu items. For instance, some drawing tools cannot be described with words and would be difficult to place on a drop-down menu.