PX 400

Chicago Explorer Superset and Replacement

Program Mgr VinodA [1]

Summary

Office Explorer will leverage the property promotion work done in Office 95, providing users rich views on documents based on the properties that they expose. Office Explorer will superset and replace the Chicago Explorer to become the single place where users can find and manipulate all their information irrespective of its type, including all documents and files, in addition to personal information such as appointments, task lists and mail. By allowing Office users to browse rich views on documents without requiring them to be connected to a groupware store, Office 96 undercuts Lotus Notes, giving away a large part of the Notes functionality for free.

[1] Portions of this spec derived from *Office Explorer Goals*, by Richard Wolf

Justification

Office '95 documents already publish their standard and custom properties in docfile streams. Users can create custom document properties, and even bind custom properties to native document content, such as bookmarks in Word or range names in Excel, enabling many important user scenarios.

The drawback of property promotion today is that it is exposed only when the user is connected to a MAPI store. On the standard Chicago Explorer, however, these properties are visible through property sheets on a per-document basis only, rather than in more useful aggregated views. By providing rich views on FAT (OFS as well if this is available on NT in the Office 96 time frame), Office Explorer makes rich views available to the average user without requiring them to be connected to a groupware store.

By implementing a feature superset of Chicago's Explorer, Office Explorer will replace the Chicago Explorer across the board and be the single place where users can find, view and manipulate, in a consistent way, all their documents, appointments, tasks and mail. This makes Office Explorer inherently more information-centric, it presents users with the all the information they need without them having to worry about its type.

Description

There are 3 main parts to making Office Explorer the browser of choice for rich views on documents:

1. Implementing rich views on FAT and OFS.
2. Implementing a "document module" in Ren, which allows the Ren app window to be the single locus for locating, viewing and manipulating all documents, in addition to personal information.
3. Implementing a feature superset of Chicago's Explorer, to be able to replace the latter across the board when Office 96 installs on Chicago.

This spec is mostly concerned with 3 above, i.e., superset and replace the Chicago Explorer. There are 3 sub parts to this:

I. Implement a feature superset of Chicago's Explorer

By doing rich views and providing a single place for documents and personal information, Office Explorer adds valuable new functionality to that provided by the Chicago Explorer. However, Office Explorer has to provide all of the functionality of the native Chicago Explorer, otherwise it forces users to choose between the two Explorers.

This functionality includes:

1. Browsing Special Folders, such as Control Panel, Fonts and Printers
2. Network browsing capabilities of Chicago's Explorer
3. Supporting Small, Large and List view modes that are in Chicago today
4. Correct UI objects for both file system and special (i.e., non file system) objects, including context menus, icons, and property sheets, as well as any registered shell extensions for these.
5. Shell compatible cut, copy, paste, delete and drag and drop operations, all of which need to work in the framework of the shell's Undo mechanism. Deleted files need to go to the special Recycle Bin folder.
6. Supporting the base menu and toolbar items created by the Explorer frame
7. Supporting the additional menu and toolbar items added by the Explorer's content pane (IShellView) and those added by the selected object in that pane; providing keyboard accelerators for these.

8. Updating views on file system change notifications

Our basic implementation approach is to leverage the Chicago shell team's work as much as possible. Chicago provides the crucial interfaces that we need to simplify our work in implementing a document browser, these include:

IShellFolder	This interface, derived from IOleContainer, is the key to doing document views in the Office Explorer. It provides an enumeration of all the objects visible to the Chicago Explorer, including the file system and network as well as all Special Folders. It also provides name parsing and all the UI elements (e.g., icons, property pages, context menus) for these objects, handles any registered shell extensions, as well as shell compatible cut, copy, paste, delete and drag/drop functionality. IShellFolder also provides support for handling the menu items added to Explorer by the content pane and by the selected object in that pane, through the IContextMenu interface.
IShellView	This interface handles the display of IShellFolder objects in the right hand side contents pane of Chicago's Explorer. IShellView will be used by Office Explorer to display all the Special Folders such as Control Panel, Fonts, Printers, Recycle Bin, etc., since Chicago has code that is specific to these special folders in IShellView. For its FAT views, Office Explorer cannot use this interface since IShellView does not support rich views.

Our implementation strategy, in a nutshell, is;

1. Leverage IShellFolder to get the list of all the Chicago Explorer objects (both FAT and non-file system), and their basic functionality, as discussed above. Use this list to build the Office Explorer scope pane, as well as for constructing the Office specialized view (which provides rich views, in addition to the default shell Explorer views) on FAT. Leverage Chicago's tree and list view controls for the scope pane tree and for Small, Large and List views.
2. Use IShellView only for the Special Folders in Chicago.
3. Implement the base menu and toolbar items created by Chicago's Explorer frame. These include Find Files/Folders/Computers, View Options and Map/Disconnect Network Drive menu items, and a couple of simple toolbar buttons.
4. Have Chicago export and publish certain APIs that will make our life easier. These APIs include shell Undo, SHChangeNotifyRegister, SHChangeNotifyDeregister, and SHFindComputer. All of these are already implemented in Chicago today, only these are internal and need to be published. Also, Chicago needs to test these APIs more rigorously, if published.

NOTE: The table in Appendix A has a detailed list of all the work items, it includes an implementation approach for each item, as well as any API support required of the Chicago shell team.

II. Replace Explorer

The Explorer is invoked from the Chicago shell in a couple of ways. It is an item on the Start menu located on the tray. Second, the right mouse button context menu on a folder includes an Explore verb that runs Explorer with that folder opened.

If the Office Explorer simply supersets the Chicago Explorer but does not replace it, then the user will invoke the standard Chicago Explorer from the Start menu or the

context menu, making it extremely easy and likely for the user to invoke the wrong Explorer.

To replace the Chicago Explorer, a few entries under Chicago's registry HKEY_CLASSES_ROOT\Folder\shell\explore branch need to be modified - the "command"entry should be replaced with the path of the Office Explorer executable, and the "ddeexec" sub branch entries need to be suitably modified as well.

III. Replace Folders

Many users of Windows 95 will rarely, if ever, encounter the Explorer *per se*, rather they will only use it in the guise of a Folder, a simplified Explorer that lacks a scope pane. These are typically invoked from the Chicago shell by double clicking a folder, which is essentially the same as running the default "Open" verb from the right mouse button context menu on that folder.

If the Office Explorer replaces only the full Explorer but not Folders, several users will never encounter it to realize its benefits.

To replace these Folders on Chicago, some entries under Chicago's registry HKEY_CLASSES_ROOT\Folder\shell\open branch need modification - here again, the "command"entry should be replaced with the path of the Office Explorer executable, and the "ddeexec" sub branch entries need to be modified as appropriate.

Platform Explain how this feature will differ on different platforms (Mac, NT, etc.)

International -

Visual Basic Need the same object model for the Office Explorer as whatever's being planned for the Chicago Explorer. The current Chicago plan seems to be to support IDispatch and publish their object model so the Explorer can be automated using VB or VBA.

User Ed If Office Explorer replaces the Chicago Explorer, do we need to ship with the Chicago Explorer docs as well? This will increase our COGS in a big way.

Q & A -

Changes

14-Nov-94	The table in Appendix A	vinoda
17-Nov-94	Description, updated table	vinoda
21-Nov-94	Updated table	vinoda

APPENDIX A

Features Office Explorer must implement to superset and replace the Chicago Explorer.

	WORK ITEM	OFFICE EXPLORER	CHICAGO
1	Network browsing	Use IShellFolder.	-
2	Special Folders - Control Panel Fonts Printers Recycle Bin Dial-Up Networking Remote Access	Use IShellFolder & IShellView. There is code in the IShellView implementation that is specific to special folders (including all the special properties seen in Details view), so Office has to use IShellView for special folders to get all this functionality. Issue: Is the Office Explorer architecture extensible to the extent it will allow IShellView to plug in and provide the view for special folders?	-
3	Shell-compatible property sheets for file system and special (non-file system) objects. This includes property sheet page extensions created by ISVs on a per class basis	Use IShellFolder.	-
4	Drag and drop. Besides FAT, this includes the network, printers, and other special objects. Has to work in the framework of the shell's Undo mechanism.	Using IShellFolder allows us to add to the shell undo stack.. Office needs API support from Chicago to invoke the Undo.	Provide shell **Undo** APIs. This is currently in the Chicago shell, but is internal and not exported.
5	Shell-compatible cut, copy, paste and delete operations. Deleted files should go to the Recycle Bin. Should work with the shell Undo mechanism.	Works via IContextMenu.	Expose **Undo** APIs
6	Generate correct icons for each object. Chicago has support in the shell for 48x48, 256 color icons.	Use IShellFolder. Chicago's has a system imagelist (i.e., the icon cache) but this cannot be exposed by APIs because it cannot function the same way on porting to NT. Office explorer has to implement its own icon caching mechanism, has to use the public Imagelist APIs in Chicago and duplicate Chicago's icon caching.	Chicago cannot expose this through APIs, it uses a global imagelist accessed by multiple processes, which needs to be per-process on NT.

	WORK ITEM	OFFICE EXPLORER	CHICAGO
7	Support all registered shell extensions: Context Menu Handler Property Sheet Handler Icon Handler Drag/Drop Handler Copy Hook Handler	Use IShellFolder::GetUIObject to get the menu and property sheet (via IContextMenu), icon (IExtractIcon), and drag/drop (IDropTarget). Copy hook handler is taken care of through the shell copy mechanism.	-
8	Display correct names for all objects shown in the explorer. Filenames may or may not include extensions, show labels for disks, prettified names for network objects.	Taken care of by IShellFolder	-
9	Views on FAT have to update on change notifications from the underlying container.	The preferred solution is for Chicago to publish their shell notification APIs so we can use them. The alternative is to use standard Win32 APIs, possibly borrow Chicago's scheme that translates FS notifications into internal messages.	Publish the existing **SHChangeNotifyRegister** & **SHChangeNotifyDeregister** APIs in Chicago.
10	Implement the Explorer frame toolbar buttons, there are just a couple of these - the dropdown with the folder hierarchy, and the "One Level Up" button.	To be added in Office Explorer.	
11	Provide Large Icons, Small Icons and List views on FAT, including the toolbar buttons added by these.	Office explorer will implement these. Office can leverage Chicago's ListView control to implement these views.	-
12	File menu - Create Shortcut Properties	Handled by IContextMenu	-
13	Edit Menu - Select All Invert Selection	Code for these exists in Ren today.	-
14	View Menu - Arrange Icons (by Name, Type, Size, and Date, also AutoArrange for Small and Large modes) Line Up Icons	Office Explorer needs to implement these.	-
15	View Options dialog Office Explorer needs to honor the options the user has set here, automatically filtering file classes that the user doesn't wish to see.	In the absence of a Chicago API to call this, Office Explorer has to create this dialog, update and keep in sync with the registry settings for the filters and other options the user has set.	View Options is in explorer.exe. Chicago does not plan to move this to the shell DLL to expose this as an API.

	WORK ITEM	OFFICE EXPLORER	CHICAGO
16	Tools menu: Find (Files or Folders/Computer) Map Network Drive & Disconnect Network Drive (including the toolbar buttons added for these) Goto	We will use our own Find File, need to get Chicago to publish the FindComputer API. Issue: Will we change Find Computer to look like our Find File? The two network commands are relatively easy to implement in Office Explorer using Win32 network APIs. Goto is easy to do in Ren.	Publish the **SHFindComputer** API that exists in Chicago today.
17	Menu items that get added to the Explorer from verbs from the selected item in the right pane, e.g.: New (Folder or Shortcut) Open Open With: when there's no registry association for a file; brings up a dialog Send To Print QuickView Explore, Sharing & Find (for folders only)	These are handled via IContextMenu.	
18	Keyboard accelerators for merged menu items	This needs to be implemented in Ren's views. Chicago does this in their IShellView implementation and IContextMenu does not expose accelerators.	
19	Open in place - Explorer in folder mode has option to open in place not a new window	New functionality to be implemented in Ren.	-
20	Load time needs to be at least as good as Chicago's explorer	Performance implication for Ren.	-
21	Support formatting disks	Format is just another verb, appears for all drives. This again gets handled via IContextMenu.	-
22	Views onto floppies compressed using Diamond compression	-	This is likely handled the same way as views into special folders. This feature may not ship on Chicago at all.
23	Replacing the default explorer through registry settings	In the registry's HKEY_CLASSES_ROOT\Folder\shell\<exp lorelopen> branches, change the "command" and the "ddeexec" entries to point to the Office Explorer executable.	Start Menu's Settings folders (Control Panel, Printers, etc.) will run explorer.exe even after this registry change, this might be a bug. This behavior will need to be tied to these registry settings as well to allow Ren to replace Explorer across the board.