Trip Report

Chicago User Interface Design Preview

July 8th and 9th, 1993

Rich Hume and Grant Skousen

Note: The information obtained at this preview was obtained under non-disclosure. Tim Satalich, the Windows Systems User Interface Evangelist, took the time at the beginning of the first day to warn everyone that if this information (especially the visuals) appears in PCWeek or any other magazine then Microsoft will attempt to find out who leaked the information and the guilty company will not be invited to future previews like this.

Summary

Chicago is definitely real, but it is at least a year away from release. The Group Program Manager for Chicago said that they are planning three betas of three months each. The first beta is planned to begin late this month[1]. That is when we should receive the PDK (preliminary development kit) and runtime that will include the new shell. When the PDK is received the authors would like to give a demo that will overview the UI changes in Chicago.

There are still a number of areas where there are disagreements within Microsoft on how certain features should be implemented. The presenters seemed most interested in input in those areas, presumably to gather ammunition for their own opinions on how the feature(s) should be implemented. Other presenters gave the impression that they had already decided how things are going to be and that they were not very open to input.

When released, Chicago will boot directly into the Windows GUI interface. All functionality provided by DOS 6.0 utilities today will be available from Windows. Most (if not all) of these utilities will have new graphical interfaces. There will be significant improvements to winoldap (i.e., the DOS box) including a toolbar and the use of TrueType fonts that will auto scale as window size is changed.

Brad Silverberg (VP over system development) emphasized that Microsoft is attempting to grow the market. Today, most machines sold are either upgrades or second machines. To help get more new customers, MS is working with a number of other vendors to get plug and play capabilities in the open PC hardware world much like Apple already enjoys in their proprietary hardware. Based on the things said, it appears that Chicago has two overriding goals. This first is ease of use. Brad Silverberg stated that Microsoft intends to make the Windows platform the easiest computer to use, bar none. Toward that end, when alternative designs are considered, the approach that appears to benefit novice users is given priority. The second goal is to make the upgrade from Windows 3.1 to Chicago a no-brainer.

---

[1] It is interesting to note that the first beta is also the second release, the first release (which we have) was an alpha release. The first beta (or second release) was also referred to as M5, or milestone 5. One would presume that M4 was the alpha release. No mention was made as to what the first three milestones were.

The following quote from the Preliminary Chicago User Interface Design Guide points out some of the challenges WordPerfect Corp. will face as we prepare our applications for the next release of Windows:

"Windows (Chicago) is the advent of the evolution of GUI to OOUI; that is, a more data-centric/object-centric, rather than application-centric, interface; a continuation of the design direction portended by OLE. As a result, application developers/ designers may need to rethink their application in terms of what are their basic components and the respective operations and properties that apply to those objects. This is particularly important since long term, from the user's perspective, the application is destined to become as transparent to the user as the code which manages menus. The user's data/information (and tasks associated with that data) will become the major focus on the user interface."

On the second day of this conference, Joe Belfiore, the program manager for the Chicago Shell/UI gave these 14 items as the UI Designer's "Points Of Light" (Note OLE 2.0 capability including Doc Files was assumed throughout the conference):

1)Use 32 bit APIs. (Note: this is the one non-UI item that was mentioned several times)
2)Support long filenames
3)Support Universal Naming Convention (UNC) path names
4)Make sure that documents/data files are accurately displayed and used in the shell
5)Support drag and drop and other transfers consistently and extensively
6)Use the common dialogs, especially File Open
7)Be careful about multiple instances of application being started too easily
8)Maintain a consistent user interface and object paradigm between the application and the shell.
9)Extend the shell's ability to provide general information about application files (more OLE 2.0)
10)Support pen input for pen notebooks and desktop tablets
11)Support Chicago-Style help
12)Size / color scalability
13)Avoid drawing custom title bars
14)Don't draw into minimized windows

These points are expanded upon in the detailed notes that follow, but they suggest that creating great Chicago applications may require significant new design. The change to an Object Oriented UI (OOUI) may require a paradigm shift greater than that we experienced when going from a character based UI to a Graphical UI. We feel that it is of critical importance that we get the foundation (primarily 32bit and OLE 2.0) in place soon. If we do not, it is unlikely that we'll have a prayer of releasing great Chicago applications in a timely manner after the system is available. You can be sure that Microsoft will expend some effort in making Chicago successfull, especially if NT sales are lackluster over the next year.

Overview
Tim Satalich, Microsoft Windows System User Interface Evangelist
Day 1 Agenda
Introductions
Chicago Product Overview
Chicago Shell Overview
New Controls, Dialogs, and Shell integration

Mobile UI
Breakout Sessions (Transfer model, Explorer / browser, menu behavior)

Day 2 Agenda
Visual Style Guidelines
Application Style Guidelines
User Assistance
How to be a Great Chicago App
Cairo User Interface Directions
Breakout Sessions (User Assistance, UI and Visuals, MDI)

Microsoft will be setting up a private CompuServe forum to discuss Chicago UI issues.

Microsoft is planning more UI design reviews in other system areas (no dates set yet). It was interesting to note that the Cairo Program Manager stated that he felt they were ready for a Cairo design review right now.

The following sections attempt to present information of interest that came out of each presentation. The slides and the information contained on them are not presented verbatim. The authors each have a bound copy of the slides that is available if you are interested. Tim was unwilling to give us an electronic copy of the slides.

Chicago Product Overview
Dennis Adler, Group Program Manager
Chicago delivers features tuned for personal needs while Cairo delivers Chicago features plus features tuned for corporate needs. The typical migration path for Windows 3.1 and Windows for Workgroups 3.1 is to Chicago. The move to Chicago is intended to be a no-brainer for users with a 386 and 4 meg. The typical migration path for Windows NT 3.1 is to Cairo. Cairo will "really shine" on the network.

On a 4 meg 386, Chicago should meet or beat Windows 3.1 in performance. According to Brad Silverberg, with 8 meg Chicago will blow away Windows 3.1.

Chicago components will form the core of MS-DOS 7.

Microsoft is working with many hardware vendors as well as BIOS vendors to establish standards that will provide a "plug and play" environment for consumers. The goal is a "no-compromise standard for PCs".

With Chicago, Microsoft will be providing a new setup program. The system as well as Microsoft applications will be providing an optional 'slim' installation to reduce disk requirements.

The new Chicago shell will provide a unified manager and desktop. There will be no more separate program manager and file manager. Drag and drop will be ubiquitous within the shell (and within the system if apps cooperate). The shell is OLE 2.0 enabled. Long file

names will be supported[2]. The long file name support is being added to the FAT file system using some undocumented bits. No details were given because a patent is being applied for. The new shell will also likely include file viewers (others may be added in an extensible way). The pen gestures have been simplified and additional support is being provided to make pen support easier to add to applications. There is an improved printing interface that offers much better background printing and the DOS application interface has been improved.

Chicago provides an extensible file system architecture and will ship with a 32 bit version of the FAT file system. The file system includes a last accessed date field. Chicago also includes a 32 bit CD-ROM file system. A protected mode implementation of double space will be provided.

Chicago is supposed to provide significantly improved communications capabilities. It was stated that most of today's products do NOT use the Windows communications driver.

Chicago provides a preemptive multi-tasking kernel. The system boots in real mode to maintain compatibility. After autoexec.bat has run, the system switches to protected mode operation. Each MS-DOS application and Win32c application executes in its own address space. All Win16 applications operate within a single virtual machine.

Chicago will provide a 32 bit multimedia subsystem including audio, video, and image compression services. The multimedia utilities will be OLE 2.0 enabled. The default system palette in Chicago contains 256 colors.

Chicago was presented as the enterprise client

According to Brad Silverberg, Chicago is "on track for a mid-'94 release". An update to Chicago is planned to coincide with the release of Cairo to ensure that Chicago operates as a good client for Cairo servers.

It was stated that it will likely be difficult to replace components of the shell.

A couple of the participants expressed strong concerns that Chicago perform better than Windows 3.1. One even said that their product was losing accounts because of the slowness of Windows itself.

Chicago Shell Overview
Joe Belfiore, Program Manager, Chicago Shell/UI
The work on the shell to date has been targeted primarily at existing Windows 3.1 users. However, the team's focus is shifting to server novice users as well. The stated goals for the Chicago shell include: providing a single object oriented paradigm (no more Program manager / File manager duality); begin moving towards document centric view; lower the entry point for novices (as compared to Win 3.1); and share a UI with Cairo (Chicago provides a subset of Cairo's functionality).

---

[2]It is interesting to note that long file names in Chicago will be limited to 254 characters while NT supports 255. The Group Program Manager for Chicago said that they estimated a 15 day development time in Chicago to get 255 characters so he decided to go with 254.

Most of this presentation consisted of a demonstration of the Chicago shell. Since we should have a copy of Chicago with the shell towards the end of the month, no attempt is made here to fully describe it. Only the most notable highlights are mentioned.

- The desktop contains icons representing active tasks as well as icons that do not represent active tasks (e.g., documents and directories can be dragged and dropped on the desktop)
- The desktop contains a tray that can be positioned at any edge of the screen. The tray can be thought of as the system toolbar. It contains three buttons, one for shutting down the system, one for searching for objects, and one for accessing help. The tray also contains an area much like the shelf in Office 4.0. Anything that can be dragged to the desktop can be dragged to the tray for easy access. This area of the tray uses 16x16 icons which will be created if the application does not define any. The tray also contains the current time of day. They realize that the tray will need to be moved out of the way at times. One implementation they have looked at is a tray that pops up when the mouse is moved to the edge of the screen. Dennis Adler said that he would veto this implementation though. The other possibility that was discussed was that you could grab the edge of the tray opposite the edge of the screen and drag it off the screen (leaving probably a three pixel handle for grabbing and dragging back onto the screen).
- The look of open windows has changed quite a bit. The caption will likely have a 16x16 document icon (as defined by the app) at the left edge. This icon will provide access to the system menu on the left button up. The icon will also provide a quick menu on right button up. The text in the caption is left justified rather than centered. The text will consist of the document name followed by the application name. The minimize and maximize/restore button graphics have changed. There is still discussion going on as to whether separate close and/or restore buttons are needed.
- Minimized windows are simply truncated captions. At least this is the most likely implementation (and it is definitely the implementation currently planned for Cairo). This was an issue that was brought up many times as an issue that they were seeking input on. The current alpha copy that we have shows a 32x32 icon with text to its right. The more appealing implementation (and the one preferred by most if not all participants) has a small icon with part of the text from the caption to its right. (Note: the width of the minimized window is under user control. Also, this implementation will break applications that currently write into the minimized window). All minimized windows are of the same size because they make better bricks.
- By default, the desktop includes three icons: the filing cabinet, the world, and a programs folder. The programs folder contains links that pretty much correspond to program manager icons today. The filing cabinet opens the explorer in a manner very similar to today's file manager showing all local drives including mapped drives. The world icon views the network using UNC pathnames.
- New menu behavior. Menus will track with the mouse button up. Cascaded menus automatically unfold when mouse moved over it. (Note: this felt very natural and should not cause any problems for existing users because the press drag release model also still works.) The menus are now gray with a 3D etched look for disabled menu items. There is no support planned for menu items with an indeterminate state (e.g., Bold when text that contains some bold and some not bold is selected).
- Links. There were many participants who disliked the word link. Links are very much like Program Manager icons in Windows 3.1 in that they have command line arguments, open window state settings, hot keys, and working directories associated with them.

Links are implemented as entries in a per-directory database stored in a DESKTOP.INI hidden file in each directory. (Note: support for displaying these links is built into the open and save as common dialogs. This will be an issue for us if we choose not to use the common dialogs because the file system API calls won't return or handle the links stored in a directory). Links can have a different icon and parameters that the original object—this is the difference between links on Windows and aliases on the Mac.

- Transfer model. This is an area that we need to look at very closely. It was interesting to note that the Excel guy who was participating was quite vocal about his disagreement with the currently planned transfer model. The interesting part is that the planned model is almost exactly what Excel already does, with a slight name change. A complete description of the transfer model is beyond the scope of this report (the style guide has about seven pages of description) however, I will attempt to provide a simplistic overview so that you can get the flavor[3]. 'Cut' is no longer a required or recommended action, and 'Paste' is replaced by 'Put'. At the source you make a selection and then pick either 'Move', 'Copy', or 'Link'. None of these options make the selection go away. At the destination, you choose 'Put <object> Here' or possibly 'Put <object> Here As...'. If you had chosen 'Copy' then the menu would likely read 'Put Copy of <object> Here'. The same is true for link. If 'Move' had been chosen, then selecting 'Put <object> Here' would cause the original selection to be removed and placed at the destination (much like cut and paste). The new transfer model still uses the clipboard so that you can say 'Move', 'Put', 'Put', ... 'Cut' is an optional command that may still be provided. The proposed accelerator for 'Move' is Ctrl+M with Ctrl+X as a suggested alias. A right click drag and drop displays a menu at the destination that includes 'Move Here', 'Copy Here', and 'Link Here'. Because the right mouse button is used in this way, quick menus are not displayed until the mouse up. This is a tradeoff because you can no longer press, drag, release to make a quick menu selection. However, Microsoft's usability testing has shown that many (if not most) users click the right mouse button anyway and do not use the press, drag, release method of making selections. Doing a left mouse drag and drop performs the default operation (which is most often a 'Move').
- Scraps. Applications will be expected to support the user selecting a chunk of data and dropping it on the desktop (or tray). It appears there as a scrap and may be later picked up and dropped into a document or container. (supported via OLE)
- The shell is currently using a chevron '»' to denote a link. There was universal distaste amongst the participants regarding this. The design team will likely try to come up with an alternative, possibly graphic, representation for a link.
- The user will be given significantly greater control over the visual look of the environment. Specifically sizes (like scroll bar and caption sizes) and colors can be set.
- Scroll bars now have a proportionally sized thumb.
- The control panel dialogs that were demonstrated had large graphical previews.

The shell properties are not extensible. It will likely be difficult to replace shell components (this is not something that is being designed for).

The shell ties into and used the OLE registration database extensively.

---

[3] As soon as we get an electronic version of the style guide it will be easy to share this information. In the meantime, you can contact either of the authors for a hardcopy.

There is a strong move to eliminate user INI file settings. Brad Silverberg stated that every entry left in the system.ini file has to be justified to him and the group program manager.

Brad noted that a number of the changes coming in Chicago are already available in SnowBall (the latest Windows for Workgroups beta).

New Controls, Dialogs, and Shell Integration
Joe Belfiore, Program Manager, Chicago Shell/UI
The following new controls will be provided in Chicago:
Toolbar
Will support button wrap so it can be made to float or dock
Status Bar
The menu font should be used in the status bar. The status bar appears to have an optional(?) visual at the right end that is used as a larger handle for sizing a window (much as is done on the Mac).
Column Heading
Supports resizing columns by dragging as well as sorting by clicking (exactly as is done in MS Mail today). A request was made that the control be made to support reordering of columns via dragging as well as giving the ability to hook the right mouse button for quick menus.
Slider
Spin Buttons
Much like our counter control
Progress indicator
Can be used in status bar for long processes
Tabs
Supports multiple banks of tabs as well as horizontal scrolling. Bitmaps can be placed on tabs. A request was made that the tabs allow visually showing a 'dirty' state for use in property sheets when one page has been changed but the changes have not yet been committed.
Property Sheet
Pages are added as dialog templates. Much like property sheets in OS/2.
List View
Supports large icon, small icon (these are both positional), list, and details views. Uses an image list icon cache that may be made available for use elsewhere.
Tree View
Used to display hierarchies - containers, outlines, etc. The application can add its own icons as well as hiding or showing lines and expand/collapse visuals.
Rich Text Edit
Supports more that 64k of text. Provides OLE 2.0 client support. Provides left, right, and center align on a per paragraph basis. Supports definable left tabs. Multiple font support as well as simple bulleting, find and replace, and superscript, subscript, and strike through.

Common Dialogs
The 'File Open' and 'Save As' dialogs are changing substantially from their current form and functionality. They will provide most of the functionality of the explorer. It will also be possible to right click on items (files, directories, etc.) in the dialog to get a context menu for

the item. These dialogs will display and handle links properly. These dialogs will also likely provide a file preview using OLE 2.0 thumbnails as well as included file viewers.

The Print dialogs generated a lot of discussion. MS is currently suggesting three dialogs: a 'Print' dialog (much like today's); a 'Choose Printer' dialog (much like today's Print Setup dialog); and a new 'Page Setup' dialog. (Note: in one of the breakout sessions, the Excel person was rather vocal about this not being a very good approach, while the Word person seemed to like it a lot)

The 'Find' and 'Replace' dialogs as well as the 'Fonts' dialog are very similar to the Windows 3.1 versions.

Once again they are planning a color dialog. The design is "still underway".

The OLE 2.0 dialogs will be 'Chicagoized' (e.g., Put Here As instead of Paste Special).

Once again in this presentation the support of OLE 2.0 by applications was emphasized. In particular, the use of compound file summary properties was noted as something that will be displayed by the shell for all files that have it. Also, in the support of 'scraps'.

Joe solicited input as to whether and which common dialogs should be made available for NT as well as Windows 3.1 (where there are differences).

The Pen in Chicago
Michael Van Kleeck, Group Program Manager, Mobile Services
MS appears to have not given up on the pen yet. The Chicago shell will be pen aware. There are several new types of edit control specifically for pen support. The gesture set has been simplified and reduced to five basic gestures. Standard controls like lists and check boxes directly support the pen.

A couple of things that will be provided in Chicago to make pen support easier are a transparent control that supports gestures and ink and a system API - DoDefaultPenInput (not discussed in detail).

Chicago Remote Network Access (RNA)
Wassef Haroun, Microsoft Windows Systems Program Manager
The RNA component of Chicago is intended to make remote access to the network very easy. A Chicago desktop can act as a private host or as a network gateway. Chicago will support merging of a remote file system with a host at a large granularity level (i.e., newer files will overwrite older ones).

From usability testing, MS found that it is important to minimize the number of steps required to establish a connection. Connection information resides in objects that appear in a special 'Remote Networks' folder. Opening one of these objects initiates a connection sequence.

The presenter solicited input as to what kind of information we believe would be useful to display in the RNA Connection Window (e.g., time connected, bytes transferred, packets transferred, errors, etc.).

Break Out Session - Transfer Model

Grant

A new name for the term link was discussed. Suggestions included shadow, pointer, reference, substitute, stand-in, view, and picture. There was also a discussion of how to represent a link better than the double chevron. Ideas mainly centered around using a different attribute for a link title. The Microsoft representatives didn't like this idea because the user can select the font and attributes of the title.

Rich

The Excel guy had strong opinions about transfer model. He was pretty vocal about the planned change being the wrong thing to do. The proposal is actually very close to what Excel does today. Apparently Excel is getting beaten up because 'Cut' doesn't behave the way most evey other app works. It behaves more like the new 'Move'. Of course the simple change of name might be enough to eliminate the problem their customers are currently having.

Eveyone in the group (but me) expressed opposition to the name link. Personally, I have not heard any suggestions that I like better. Besides, Office already uses the term link. :-)

Break Out Session - Menu Behavior

Grant

Everyone liked the new tracking. There was discussion on whether or not clicking on a popout menu should execute a default action. I felt that it should not execute a default action but that it should require a double click to execute a default action like the main menu.

Rich

Everyone liked the new tracking. There was quite a bit of skepticism regarding executing default menu items on cascaded menus. Input was solictited on the desirability of tear off and or scrollable menus. There were a few people who expressed an interest in each, but no overwhelming desire for either.

Break Out Session - Explorer/Browser

Grant

Our group was a little unorganized on this session. We talked about Cairo and SDI vs MDI.

Rich

During this session, we ended up talking mostly about MDI. The Excel guy felt pretty strongly that it would be a mistake to get rid of MDI. Steve Madigan (program manager for Cairo) disagreed very strongly. He is concerned that there is no model of containment today that a user can relate to. His example was two browser windows on the screen, each with a Word and an Excel document in it. Openning the four documents in any order always produces two additional windows, one containing both Word documents and the other containing the Excel documents. That really makes no sense at all. He indicated that there is work underway for Cairo that will likely allow association based on project or task, not based on the application that happened to create the object.

Chicago UI Design from a Visual Perspective

Renee Marceau, Microsoft Windows Systems Visual Interface Designer
Microsoft's visual interface designers now say that the black outlining of icons (that they had previously strongly recommended) cause visual clutter and unnecessary complexity. In one of the breakout sessions the designers were taken to task on this point. Their response was that "they discovered that the icons work without the black outline". They now recommend using black on lower and right edges while using a dark color on the upper and left edges. This helps provide a 3D effect for the icons.

In Chicago they intend to address the areas of the environment that are not visually consistent and intuitive (e.g., the system menu visual vs. the other caption visuals). They also intend to allow users to set all colors used by the system. It will also be possible to alter the scaling between physical and logical pixels. All system visuals are system drawn and scalable.

When released, Chicago should appear more refined and softer that today's Windows interface. They are removing a lot of unnecessary clutter in areas like resizable frames.

The difference between combo boxes where the user can enter text and those where the user can't will no longer be distinguished by a separation between the edit control and the button. Instead, if the user cannot type into the edit control, its background color will be gray (by default) while controls that can be typed into will have a white background (by default).

In Chicago, applications will want to provide both document and application icons in sizes of 32x32 and 16x16.

During the presentation, one of the Lotus representatives related a usability study they did in which they were addressing a confusion their users expressed over not knowing which buttons would dismiss a dialog and which wouldn't. They found that placing the buttons that would dismiss the dialog in a separate raised area at the bottom of the dialog produced dramatically improved user understanding.

Chicago UI Design Guidelines
Tandy Trower, Acting Program Manager, UI Design Guide
Tandy emphasized that the new style guide is still preliminary. We need to review it and get out input back to them. He also emphasized that the style guide presents guidelines, not rules. I suspect that he is trying to head off some of the debate that took place during the discussions on the first style guide.

The UI is changing to treat everything as objects. These objects are, or have, components (object are often composed of other objects), properties, and relationships.

The following types of relationships between objects were presented:
- •Collection - a set of objects
- •Constraints - a set where a change to an object affects another (e.g., layout)
- •Composite - a set where the aggregation is a unique object (e.g., grouped objects)
- •Containment - a set where one of the members is related, but independent, a place where a set "lives" (e.g., folder, document)

Mouse selection will change somewhat. You can use the right mouse button to do a drag select. On the mouse up, a context menu will appear for the selection. A Shift+click always

extends the last selection. It is no longer necessary to use both the Shift and Ctrl keys to extend a disjoint selection. There is currently no definition for what a double right mouse click should do. They are discussing possibilities including help and properties. Right clicking on a selection should not collapse the selection (this allows for a context menu on the mouse up). They are evaluating having this same behavior on the left mouse button. They are also thinking about swallowing a right click that is used to dismiss a popup menu so that the selection does not change.

Tandy said that they have observed a problem where many users will open the same document multiple times because the previous window will become obscured. They are recommending that if the user attempts to open a document that is already open, the previously opened document should be surfaced rather than opening it a second time. There would obviously need to be some non-standard way of opening the same document twice for those rare cases where that is actually what you want to do. If a user attempts to open an application that is already running, they should be given the option to surface the running app or to open a second instance.

Property sheets were felt by Tandy to be a significant new aspect of the UI. Property sheets provide pages of information and tabs as the common form of navigation. In the presentation, Tandy was proposing that 'Apply Now' be a standard/common button on property sheets. There was a strong vocal reaction that the button should be 'Apply' and not 'Apply Now'. Tandy said that they are evaluating (and soliciting input on) whether there should be recommendation made as to when changes made to a page are committed. From the discussions in the breakout session, it seemed apparent that no recommendation could realistically be made. Some apps will commit changes on a page change and others will commit on the dialog's dismissal. Given this, there was a request made that the tab control be enhanced to support a visual clue that a page is 'dirty' (i.e., changes have been made that have not yet been committed).

It was stated during this discussion that Cairo will support extensible properties on objects and that Cairo will address per user views of objects. This information will be stored externally and will be accessed via a new OLE protocol.

Chicago will likely not provide property inspectors, but an application could do this on their own and Cairo will likely address the capability in the system. A property inspector differs from a property sheet in that it is like a modeless dialog within which you could display the properties of various objects. The Lotus Notes guy made some comments that gave the impression that they are planning this type of capability.

A number of window design changes are still being debated within Microsoft. Their questionnaire specifically requested the participants to choose amongst several possible title bar alternatives. They are trying to decide whether to include close and or restore buttons on the window title bar. The title bar will include an icon that represents the object (much like Office 4.0 does today). They were also soliciting input on what a maximized MDI child window should look like. The title bar caption is being reversed. It should contain the name of the document optionally followed by the name of the application. A minimized window will most probably look just like the left portion of the title bar (the width is under user control, but all minimized windows will have the same width).

MDI was a very popular topic. I got the impression that most participants want to see MDI go away. Several (including Lotus) specifically asked that Microsoft make a strong statement encouraging vendors to move away from MDI. Based on what Tandy and other Microsoft people said, no such statement will be made in the Chicago time frame. Chicago will definitely support MDI. The Cairo program manager talked quite a bit in the break out session about his desire to have Cairo support an evolution of MDI where windows from multiple applications will either be contained in some minimal workspace window, or will simply be related to one another in some visual way with common window management functions (e.g., put this project away, arrange the windows of this project). He even made the statement that he was surprised no ISVs have done something like this with their own application suite already.

Windows User Assistance
Marion Hogan, Chicago User Education
Microsoft's usability studies and focus groups have shown that people have trouble finding the help they need. To attempt to address this, Chicago will make help more visible by adding a persistent access point to help on the desktop. There was some discussion in a break out session about providing a mechanism for applications to tie into that access point so that help for the active app/object could be found there. They intend to emphasize search and build much better keyword lists. They may even ship a way to fully index the help files, but the system help won't include this because of disk space limitations.

Microsoft has also found that the search dialog was hard to figure out. Users do not typically learn the most efficient way to find topics. In Chicago, the search dialog will be simplified and will be modeled after the standard Chicago property sheet.

The system help for Windows 3.1 has been found to be too lengthy and frustrating to read through. In Chicago, each control on dialogs will have a quick menu that includes a help menu item (they mentioned that it might be called 'Quick Info' or 'Quick Help', since we already call our context menus quick menus, we should change help to quick help or quick info). Selecting the help menu item displays a popup help window describing the control. The help writers at Microsoft are also taking a much more minimalistic approach to writing help. There will not be a lot to read through, and there will often be shortcut buttons in the help that take the user directly into the application dialog being described. The help will also focus more on tasks.

Microsoft has also observed problems when a user is reading help and they switch back to the application. This typically obscures the help window and causes frustration. To address this, they plan to use more secondary windows and give them the 'always on top' style. These windows will either be typically small, or dynamically sized so as not to obscure too much of the application window.

The performance of help has been a problem. They have reduced the startup time for help by 50%.

They have also found that users often have a hard time getting our of help. To address this, they have enabled the escape key to dismiss a help window and they have added a close button to the button bar.

The new help system will support interactive procedures by allowing help to establish a one to one link with an application so that help authors can write trackable procedures. This will be much like que cards (not like wizards or coaches).

Chicago will include a tour that teaches Chicago skills, a tips and tricks help for advanced users (they have had a lot of feedback that says users like this kind of stuff), and a User's Guide that includes conceptual information and troubleshooting information.

The new help compiler will have its dependency on MS-DOS conventional memory removed so that larger help files can be compiled without resorting to using OS/2. A problem with code page problems for non-english versions has been fixed. Support for PCX and other graphic formats has been added. Compression for graphics will be better.

How to be a great app in the Chicago Shell
Joe Belfiore, Program Manager, Chicago Shell/UI
The items presented here relate primarily to user interface issues and the shell. Other system related Chicago issues were not explicitly discussed.

1)Use 32 bit APIs. (Note: this is the one non-UI item that was mentioned several times)

2)Support long filenames
- •Open and save files with long names
- •Display them in title bars

3)Support UNC pathnames
- •Open and save files with UNC paths to enable direct network browsing
- •Display the "friendly form" in the title bar (My File on SERVER1)
- •Use correct sharing modes so multi-user scenarios work well

4)Make sure that documents/data files are accurately displayed and used in the shell
- •Put icons for document types in EXE and register them in the OLE database so they are displayed correctly in the shell
- •Add data-specific commands to the database for document types (e.g., "Play" for a sound). These will then be displayed in menus throughout the shell.
- •Make sure a document-window title bar displays the object a user double clicks (SDI parent or MDI child, as appropriate)
- •Support "print-to" in the registration database to enable drag/drop printing to specific printers in the shell.
- •Long extensions. (This sounded like a half-baked idea of registering long extensions for use in file type identification, and then only showing users the first three characters of the extension.)

5)Support drag and drop and other transfers consistently and extensively
- •Support OLE 2.0 drag and drop to enable users to move data to and from the desktop, tray or explorer and other applications. Shell transfers are interesting and important! (mail someone a link to a document, etc.)
- •Support non-default drag with mouse button 2, displaying a menu
- •Revise menu-based transfer model to work well with the one in the shell. Change "Paste" to "Put <object> Here", and consider replacing "Cut" with "Move". (Note:

there was much discussion on this point. Further elaboration can be found below. This is an area that we need to think about and provide input to Microsoft soon.)

6)Use the common dialogs, especially File Open
- •To be consistent with other applications and with the explorer itself, making the system easier to learn overall. (in breakout session, many said they would likely not use the common dialogs although they seemed concerned for same reasons that I am)
- •Get lots of functionality for free if File Open and Save As common dialogs are used
  - ❍links (both enumeration and resolution)
  - ❍direct browsing of the network
  - ❍shell operations like drag and drop of files, creation of new folders, etc. via quick menus

7)Be careful about multiple instances of application being started too easily
- •Restore windows that are open rather than creating additional windows on the same document by default
- •Put up a list of currently open application instances if a user double-clicks the application icon a second time -- MS has seen people run out of memory by accidentally starting the same app over and over
- •Follow the Style Guide for details (Note: we have a hard copy of the new style guide in its current form, we do not have an on-line copy)

8)Maintain a consistent user interface and object paradigm between the application and the shell.
- •Use quick menus on mouse button 2 like the shell does (Note: the menu appears on button up to allow right button drag and drop)
- •Replace complicated "settings" dialogs in applications with property sheets where appropriate (a new control is provided to assist in this)
- •Take advantage of new system-supplied controls like the Toolbar, Status Bar, Spin Buttons, etc. (Note: 'Toolbar' is no longer trade marked by Microsoft)
- •If a delete function is provided by the application, the app should call the new Wastebasket API so that deletions to the user's Wastebasket instead of disappearing completely.

9)Extend the shell's ability to provide general information about application files (more OLE 2.0)
- •Use compound files as datatype and store the OLE 2.0 general properties like author, thumbnail, etc. in them so they will be displayed by the shell

10)Support pen input for pen notebooks and desktop tablets
- •Use pen APIs to activate the pen so that users can edit documents using gestures and input text using handwriting recognition
- •Use ink-edit controls to allow users to enter scribbled notes, drawings, and signatures
- •Enable "ink" annotation of documents using the OLE 2.0 annotation server
- •Add other natural pen-oriented features and gestures

(Note: Chicago will make it possible to add a lot of pen functionality with little investment, or so it appears, and so they say)

11)Support Chicago-Style help
- •Use context-sensitive help popups in dialogs

- •Add "Help" items to the quick menus
- •Revise Help menu according to the Style Guide

12)Size / color scalability
- •Custom controls need to get 3D colors and size metrics from system
- •Test application when logical/physical scaling is altered. Using a TrueType font for dialogs can help a lot here (Arial 8 is the preliminary suggestion)

13)Avoid drawing custom title bars

14)Don't draw into minimized windows

Cairo User Interface Directions
Steve Madigan, Group Program Manager, Advanced Systems User Interface
Steve stated that Cairo is being designed as a framework for extensibility. He emphasized that Cairo is intended to address cooperative computing issues at the system level.

Today, most personal computers are used by information producers. Cairo is being designed for information consumers. Cairo will incorporate technology advances that won't initially be seen in Chicago.

In Cairo, OLE is the model for extensible services, period. OLE is even more extensively used than it is in Chicago. Improvements to OLE include moving links with an object automatically, distributing OLE across the network, making programming interfaces extensible, and storing OLE naively in the object file system. The desktop is an OLE container.

Some things added by Cairo that Chicago will not have are, query support (everything in the file system is indexed), distributed processing, and tools.

In Cairo, a property set can have an associated behavior. The example of e-mail was used. An e-mail message is any object that has the e-mail property set. That property set might be added to an object by dropping the object in an Out Box.

Break Out Session - User Assistance
Grant
I ask that they consider including an application with the PDK release the shows the help style they are recommending.

Rich
There was discussion, but no real answers, regarding how online material relates to printed material. Both the Aldus representative and the Symantec representative expressed a lot of interest in this.

Given their proposed model for the Help window using property sheets, I asked if they had thought of compressing the help menuitems to one, and if so, then is there any need for a help menu at all or could active applications tie into the persistant help icon on the tray. They seemed interested in this idea, but still uncertain as to what there help menu recommendations are going to be.

Break Out Session - UI and Visuals

Grant

Tandy emphasized that Microsoft would like feedback on the style guide. There was general agreement that the new visuals are an improvement. Dialog button placement will still follow the guidelines in the 3.1 style guide except that for tabbed dialogs bottom button placement seems to work the best.

Rich

The graphics designers denied my accusation that they must have received a NeXT system for Christmas. They definately seem to be struggling with the idea of putting a close button on the caption. Tandy mentioned that they are the only significant GUI that does not do this.

Break Out Session - MDI

Grant

SDI is the model for Cairo. The Microsoft representatives were asking what obstacles we saw in getting our applications to SDI. We also talked a little about the Visual Basic model - this does not appear to be a model they are pursuing for general use.

Rich

This was my last session on Friday and it was after 5:00. There were only three of us left and there were three of them. We mostly just talked about Cairo and the Cairo demo they gave that day.

Further Information

Rich Hume and Grant Skousen have the following documents and software. If you need more information than is provided in this document something in this list may be of help to you.

- *Slides from Chicago User Interface Design Preview*, July 8-9. This was the basis for this document. It contains visuals of most of the new UI.
- *User Interface Design Guide for Microsoft Windows (Chicago)—Preliminary Draft*. We need to get comments on this back to Microsoft as soon as possible
- *Slides from Chicago Overview* presented at WordPerfect by Glenn Thompson of Microsoft, July 2. This is a high level overview of Chicago.
- *Application Compatibility in Future Versions of Windows*, dated March 24, 1993. This document points out what must be done to insure current applications are compatible with Chicago. This document was passed out at the Chicago Overview.
- *List of Win32c APIs*. This is a soft copy of a Excel spreadsheet listing all of the Win32c APIs. This was also passed out at the Chicago Overview
- Alpha Chicago CD. This copy is missing the new shell and many other features of Chicago. Installation on at least one machine has caused the loss of the FAT. It may be best to wait till the PDK release to spend time looking at Chicago.