US 20040261039A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2004/0261039 A1**

Pagan (43) **Pub. Date: Dec. 23, 2004**

(54) **METHOD AND SYSTEM FOR ORDERING ON-SCREEN WINDOWS FOR DISPLAY**

(75) Inventor: **William G. Pagan**, Durham, NC (US)

Correspondence Address:
SAWYER LAW GROUP LLP
PO BOX 51418
PALO ALTO, CA 94303 (US)

(73) Assignee: **International Business Machines Corporation**, Armonk, NY (US)

(21) Appl. No.: **10/464,900**

(22) Filed: **Jun. 19, 2003**

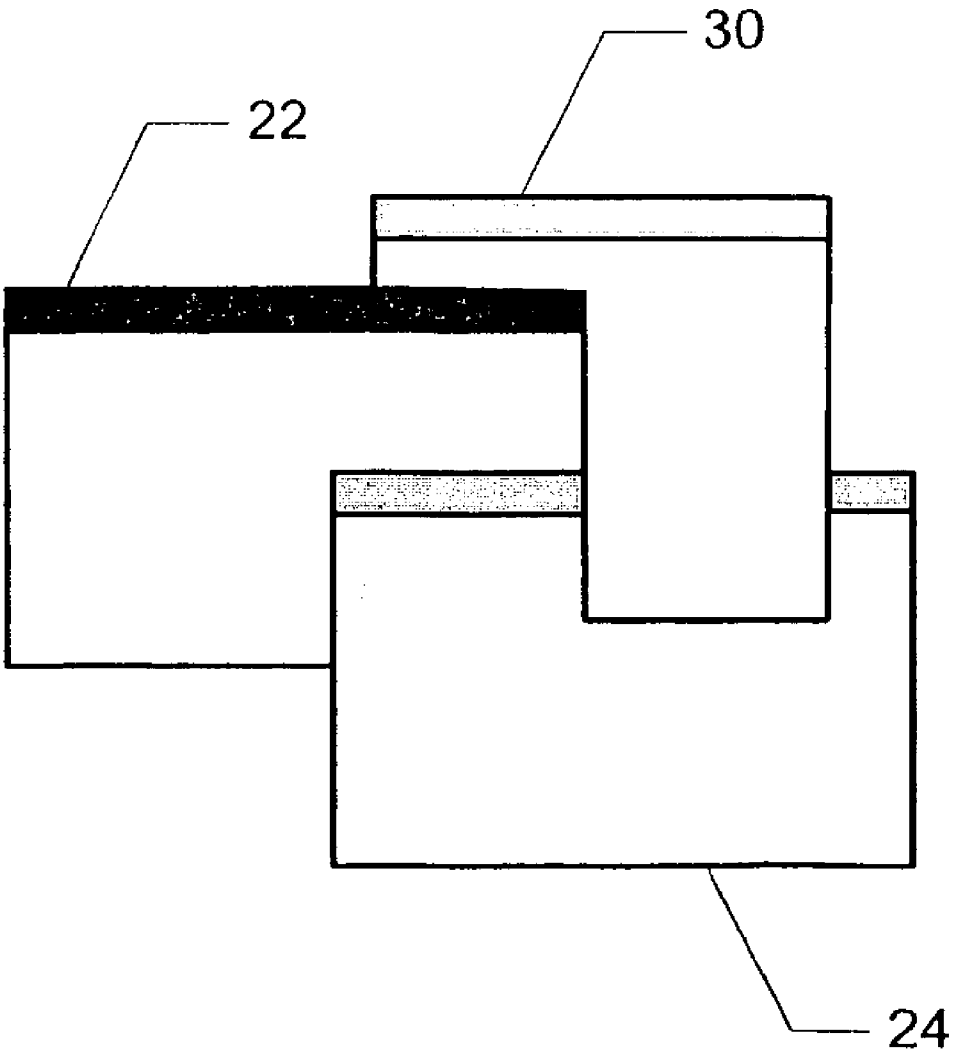
Publication Classification

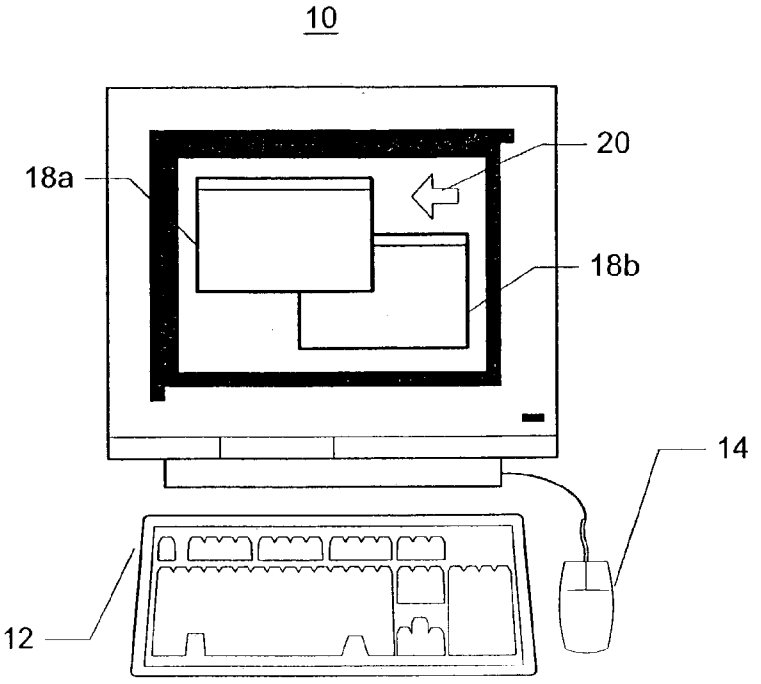
(51) **Int. Cl.**[7] **G09G 5/00**

(52) **U.S. Cl.** **715/797**; 715/794; 715/790

(57) **ABSTRACT**

A method and system for ordering on-screen windows for display is disclosed in which an active window partially overlaps an inactive window and hides a portion of the inactive window. The movement of a mouse pointer is monitored to determine its location on the screen. In response to the inactive window being selected with the mouse pointer, the inactive window is given focus to enable the inactive window to receive event signals even though a portion of the inactive window remains hidden, whereby original display depths of the windows is maintained.





Prior Art

FIG. 1

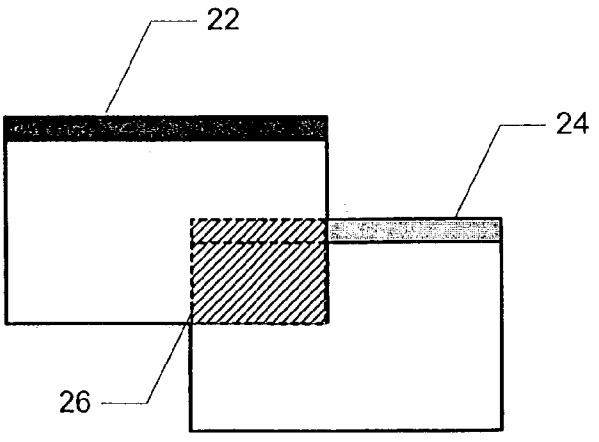


FIG. 3

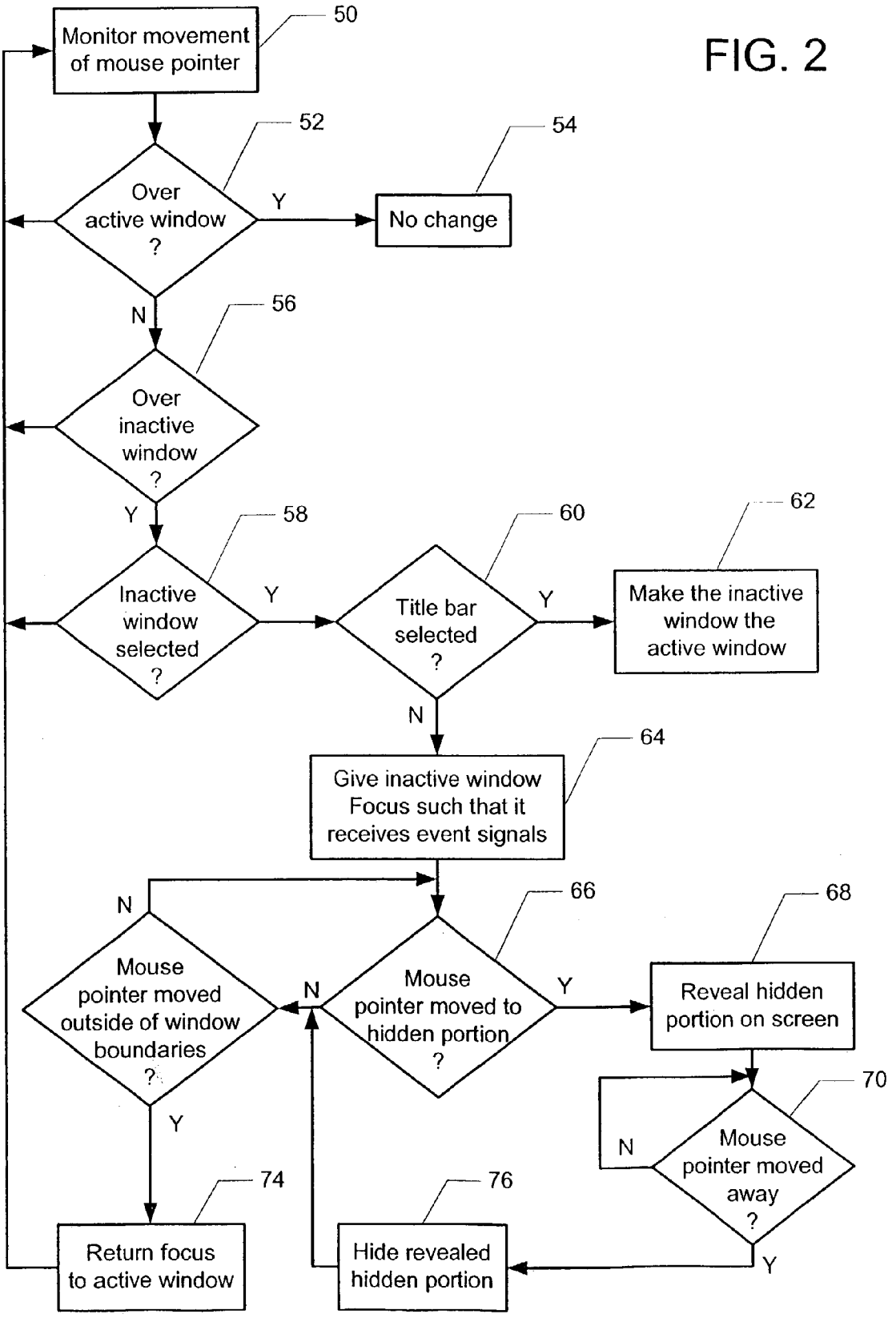

Monitor movement of mouse pointer
50
Over active window ?
52
Y
No change
54
N
Over inactive window ?
56
Y
Inactive window selected ?
58
Y
Title bar selected ?
60
Y
Make the inactive window the active window
62
N
Give inactive window Focus such that it receives event signals
64
N
Mouse pointer moved outside of window boundaries ?
N
Mouse pointer moved to hidden portion ?
66
Y
Reveal hidden portion on screen
68
N
Mouse pointer moved away ?
70
Y
Y
Return focus to active window
74
Hide revealed hidden portion
76

FIG. 2

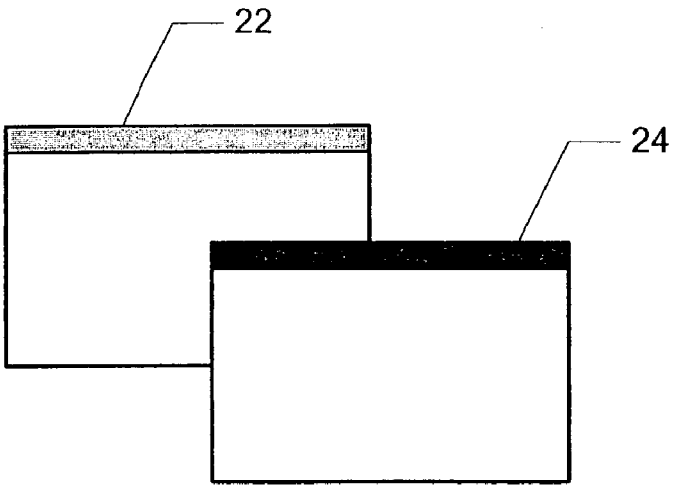


FIG. 4

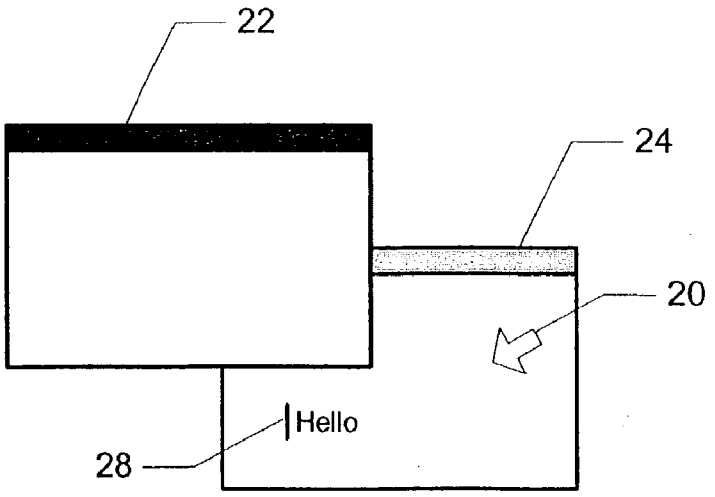


FIG. 5

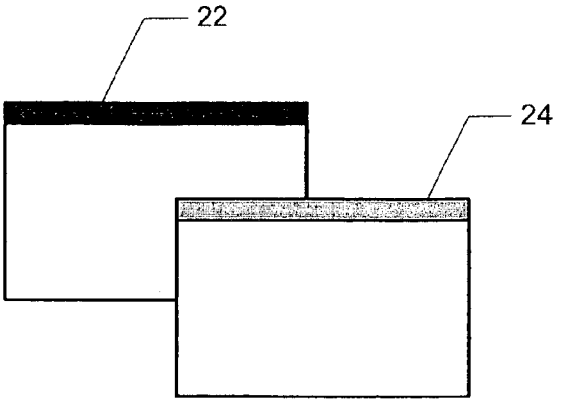


FIG. 6

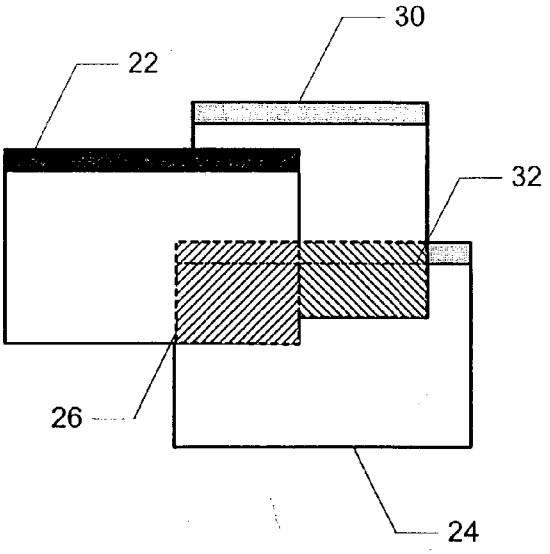


FIG. 7A

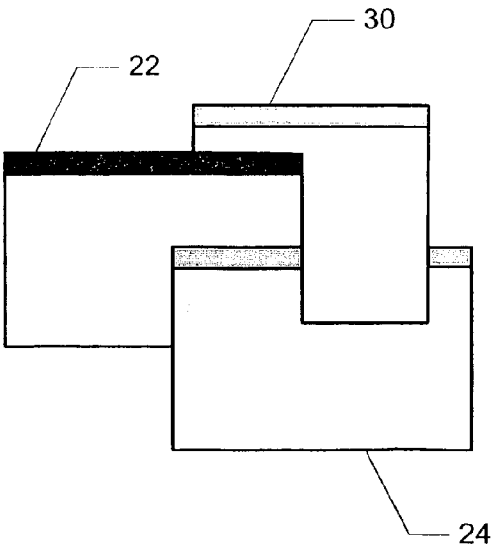


FIG. 7B

METHOD AND SYSTEM FOR ORDERING ON-SCREEN WINDOWS FOR DISPLAY

FIELD OF THE INVENTION

[0001] The present invention relates to window-based user interfaces, and more particularly to a method and system for providing improved on-screen window ordering.

BACKGROUND OF THE INVENTION

[0002] Almost all of today's personal computers are controlled by operating systems that have a window-based user interface. Most of these operating systems are capable of multitasking in which multiple programs execute simultaneously, and each program displays a window on a display screen for user interaction. The ability to resize application windows and change style and size of fonts are the significant advantages of a GUI versus a character-based interface. Window-based GUIs have become the standard way users interact with computers, and the major GUIs include Windows™ and Mac™ along with Motif for UNIX and the GNOME and KDE interfaces for Linux.

[0003] **FIG. 1** is a block diagram illustrating a conventional computer system with a window-based user interface. As is well known, a typical personal-computer (PC) system **10** includes a microprocessor and memory (not shown) for executing computer programs, various input devices, e.g., keyboard **12** and mouse **14**, and a display screen **16**.

[0004] Depending on the number of computer programs running, the display screen **16** may display any number of open windows **18***a* and **18***b* (collectively referred to as windows **18**), each awaiting user input. Example types of computer programs include spreadsheets, word processors, and Web browsers. Depending on the arrangement and number of windows **18** displayed, it is common for the windows **18** to overlap so that some windows **18** partially or completely obscure other windows **18**. To manage the display of windows **18**, the operating system typically views the windows **18** as a stack and maintains an ordering of the windows **18** with respect to display depth.

[0005] The mouse **14**, or other type of input device, may be used to move a mouse pointer **20** among the various windows **18**. The user typically activates a particular window **18** by clicking the mouse pointer **20** anywhere on the window **18**. When a window **18** is activated, i.e., becomes the active window **18***a*, it is displayed in its entirety on top of all other windows **18***b* and is the only window that can receive user input. The user typically interacts with the active window **18***a* by selecting or entering data in the window **18**, or by selecting menu functions associated with the window **18**.

[0006] Although windows **18** can be moved to open areas of the screen and/or resized, the current window management schema makes it difficult for user to juggle more than a very few windows **18** at a time without obscuring other windows **18** that are being used. When one window **18***a* that the user wishes to interact with is obscured by another, the user must take extra steps to return the window **18***b* to the forefront. This can be terribly annoying at times, particularly as the number of windows **18** being juggled increases.

[0007] One attempted solution to this problem is disclosed in U.S. Pat. No. 5,046,001, which describes a method for accessing selected windows in a multitasking system. In this method, user inputs are monitored to determine the position of a cursor, and as the cursor passes over each window, the window is automatically promoted to the top display position and made active. The problem this method, however, is that the user can quickly lose control of which windows are promoted to the top of the display. Consider for example, the situation where there are many tiled windows displayed and the user needs to move the cursor to the side of the display screen **16** to select a program icon. Assuming that the cursor passes over each of the tiled windows on its way to the icon, each of the windows would pop to the top of the display in quick succession whether the user wanted them to or not. Rearranging the depth order of the windows in this fashion without user consent could be distracting and annoying to the user in certain situations.

[0008] Accordingly, what is needed is an improved method for ordering on-screen windows. The method should retain the original depth ordering of the windows, but also allow inactive windows to be used as though they were active. The present invention addresses such a need.

SUMMARY OF THE INVENTION

[0009] The present invention provides a method and system for ordering on-screen windows for display in which an active window partially overlaps an inactive window and hides a portion of the inactive window. The movement of a mouse pointer is monitored to determine its location on the screen. In response to the inactive window being selected with the mouse pointer, the inactive window is given focus to enable the inactive window to receive event signals even though a portion of the inactive window remains hidden.

[0010] According to the method and system disclosed herein, the present invention enables a user to interact with inactive windows as though they were active without upsetting the original display depths of the windows. In addition, the user must physically select an inactive window to provide it focus before hidden parts of the inactive window can be made to appear, thereby overcoming the problems inherent in automatically bringing to the forefront any window that the mouse pointer passes over.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] **FIG. 1** is a block diagram illustrating a conventional computers system having a window-based user interface.

[0012] **FIG. 2** is a flow chart illustrating a process for intelligently ordering on-screen windows for display in accordance with a preferred embodiment of the present invention.

[0013] **FIG. 3** is a block diagram showing two open windows displayed on the screen.

[0014] **FIG. 4** is a block diagram showing an inactive window being made the active window and brought to the forefront of the screen.

[0015] **FIG. 5** is a block diagram illustrating the screen once the inactive window is given focus.

[0016] **FIG. 6** is a block diagram illustrating the screen showing the once hidden portion of the inactive window

being displayed in response to movement of the mouse pointer when the window has focus.

[0017] **FIGS. 7A and 7B** are block diagrams illustrating the process of displaying hidden portions of the inactive window when one or more intervening windows are displayed between the active window and inactive window.

DETAILED DESCRIPTION OF THE INVENTION

[0018] The present invention relates to a method for managing the display of windows in a window-based computer system. The following description is presented to enable one of ordinary skill in the art to make and use the invention and is provided in the context of a patent application and its requirements. Various modifications to the preferred embodiments and the generic principles and features described herein will be readily apparent to those skilled in the art. Thus, the present invention is not intended to be limited to the embodiments shown, but is to be accorded the widest scope consistent with the principles and features described herein.

[0019] The present invention provides an improved system for managing window-based environments that reduces the need for a user to mandate the position, dimensions, layered depth (i.e., the number of windows that a particular window is from being in the forefront, or the number of windows away that the particular window is from being the active window), and maximized/minimized status. The present invention accomplishes its objectives by changing what occurs when an inactive window is clicked, and how the window then behaves when the window is not in the forefront.

[0020] The present invention provides a mechanism that allows the user to interact with an inactive window as though the window was at the forefront without changing the relative display depths the windows. This mechanism is referred to as giving an inactive window focus and operates as follows. When the user moves the mouse pointer over an inactive window and clicks on a visible portion of the inactive window, the inactive window is given focus, meaning that the window receives even signals, even though the window is not the "active" window and displayed in the forefront. Once the window is given focus and the user passes the cursor over a hidden portion of the inactive window, the hidden portion of the inactive window is temporarily revealed on the display for user interaction. Once the user moves the cursor away this revealed portion of the window, that portion of the window returns to its hidden state. If the cursor is removed from within the boundaries of the inactive window, focus is automatically returned to the active window.

[0021] Thus, because the present invention requires the user to physically select an inactive window before focus is given to that window, the present invention does not suffer the problem of the method that automatically brings a window to the forefront when the mouse pointer is passed over the window. And because the window with focus is not automatically displayed in the forefront and its hidden portions are only temporarily revealed during a mouse pointer pass over, the present invention does not permanently change the relative display depths of the windows.

[0022] **FIG. 2** is a flow chart illustrating a process for intelligently ordering on-screen windows for display in accordance with a preferred embodiment of the present invention. The process will be explained at a base level where there is at least two open windows displayed on the screen **16**, as shown in **FIG. 3**, which shows an active window **22** partially overlapping an inactive window **24** so that a portion **26** of the inactive window **24** is hidden by the active window **22**.

[0023] Referring to both **FIGS. 2 and 3**, the process begins in step **50** by monitoring the movement of the mouse pointer **20** across the screen **16** until the mouse pointer **20** is positioned over any one of the open windows. It is then determined in step **52** if the mouse pointer **20** is over the active window **22**. If so, no change is made in step **54**.

[0024] In step **56**, it is determined if the cursor is positioned over any inactive window **24**. If yes, then it is determined in step **58** whether the user has selected the inactive window **24**. In a preferred embodiment, the user selects an inactive window **24** by clicking with the mouse **14** when the pointer **20** is over the window **24**.

[0025] According to the present invention, when the user selects an inactive window **24** it must be determined whether the user wishes to make the inactive window **24** the active window, or rather give the inactive window focus so that it can receive user input, while the original depth ordering is maintained. In a preferred embodiment, the distinction is made in step **60** by determining whether the user selects the title bar of the inactive window **24**, or instead selects any visible portion of the inactive window **24**.

[0026] If the user has selected the title bar, typically by clicking on it, then the inactive window **24** is made the active window in step **62** and brought to the forefront of the screen, as shown in **FIG. 4**.

[0027] Referring again to **FIG. 2**, if the user has clicked on any visible portion of the inactive window **24**, then in step **64**, the inactive window **24** is given focus. **FIG. 5** is a block diagram illustrating the screen **16** once the inactive window **24** is given focus. When the window **24** is given focus, the mouse pointer **20** and/or a cursor **28** are displayed in the window **24** and the window **24** receives all events signals from the user input devices, but the depth ordering of the windows remain unchanged so that some portions **26** of the inactive window **24** remain hidden.

[0028] Referring again to **FIG. 2**, once the window **24** is given focus, it is determined in step **66** whether the mouse pointer **20** is moved over a hidden portion **26** of the window **24**. If the mouse pointer **20** has been moved over the hidden portion **26**, then in step **68** the hidden portion **26** is revealed on the screen.

[0029] **FIG. 6** is a block diagram illustrating the screen **16** showing the once hidden portion **26** of window **24** being displayed in response to movement of the mouse pointer **20** when the window **24** has focus. Once displayed, the once hidden portion **26** of window **24** is also able to receive even signals because all displayed areas of window **24** are given focus. Note, although the example shows all of window **24** being displayed, window **22** remains the active window, which is typically indicated by displaying a highlighted title bar in the active window **22** and displaying a grayed-out title bar in the inactive window **24**.

[0030] Referring again to **FIG. 2**, if it is determined that the mouse pointer **20** is moved away from the portion **26** of the window **24** that would normally be hidden in step **70**, then that portion **26** of the window **24** is returned to its hidden state in step **76**, as shown in **FIG. 5**. If the mouse pointer **20** is not moved away, then step **70** is repeated to keep checking.

[0031] If the mouse pointer **20** is not moved over a hidden portion **26** in step **66**, then in step **72** it is determined if the mouse pointer **20** is from within the boundaries of the inactive window **24**. If yes, then in step **74**, focus is automatically returned to the active window **22** and the movement of the mouse pointer **20** continues to be monitored. If the mouse pointer is not moved outside the boundaries of the inactive window **24**, then the process continues by determining if the cursor passes over a hidden portion **26** of the inactive window **24** in step **66**.

[0032] **FIGS. 7A and 7B** are block diagrams illustrating the process of displaying hidden portions of the inactive window **24** when one or more intervening windows are displayed between the active window **22** and inactive window **26**. **FIG. 7A** shows an intervening window **30** between active window **22** and window **24**, which currently has focus. The windows are positioned such that window **30** hides a portion **32** of window **24** that the active window **22** does not. **FIG. 7B** shows that if the mouse pointer **20** is moved to the portion **26** of the window **24** hidden by the active window **22**, then only that hidden portion **26** of window **24** is displayed. The portion **32** hidden by window **30** remains hidden. Should the mouse pointer **20** pass from portion **26** to portion **32**, then portion **26** would return to its hidden state, and portion **32** would be displayed.

[0033] A method and system for ordering on-screen windows has been disclosed. The present invention has been described in accordance with the embodiments shown, and one of ordinary skill in the art will readily recognize that there could be variations to the embodiments, and any variations would be within the spirit and scope of the present invention. Accordingly, many modifications may be made by one of ordinary skill in the art without departing from the spirit and scope of the appended claims.

We claim:

1. A computer-implemented method for ordering on-screen windows for display in which an active window partially overlaps an inactive window and hides a portion of the inactive window, the method comprising the steps of:

(a) monitoring movement of a mouse pointer around the screen; and

(b) in response to the inactive window being selected with the mouse pointer, causing the inactive window to gain focus to enable the inactive window to receive event signals even though a portion of the inactive window remains hidden, whereby original display depths of the windows is maintained.

2. The method of claim 1 wherein step (b) further includes the step of: in response to the mouse pointer being moved over the hidden portion of the window, revealing the hidden portion of the window on the screen and enabling that portion of the window to also receive event signals.

3. The method of claim 2 wherein step (b) further includes the step of: in response to the mouse pointer being moved away from the revealed hidden portion of the inactive window, hiding the previously revealed portion of the inactive window.

4. The method of claim 2 wherein step (b) further includes the step of: in response to the mouse pointer being removed from boundaries of the focused window, returning the focus to the active window and hiding any previously revealed portions of the inactive window.

5. The method of claim 1 wherein the inactive window is selected by a user when the mouse pointer is clicked on a visible portion of the inactive window.

6. The method of claim 5 wherein if the mouse pointer is clicked on a title bar of the inactive window, the inactive window is made the active window, wherein the display depths of the windows are reordered such that the active window is moved to a forefront of the screen.

7. A computer-readable medium containing program instructions for ordering on-screen windows for display in which an active window partially overlaps an inactive window and hides a portion of the inactive window, the program instructions for:

(a) monitoring movement of a mouse pointer around the screen; and

(b) in response to the inactive window being selected with the mouse pointer, causing the inactive window to gain focus to enable the inactive window to receive event signals even though a portion of the inactive window remains hidden, whereby original display depths of the windows is maintained.

8. The computer-readable medium of claim 7 wherein instruction (b) further includes the instruction of: in response to the mouse pointer being moved over the hidden portion of the window, revealing the hidden portion of the window on the screen and enabling that portion of the window to also receive event signals.

9. The computer-readable medium of claim 8 wherein instruction (b) further includes the instruction of: in response to the mouse pointer being moved away from the revealed hidden portion of the inactive window, hiding the previously revealed portion of the inactive window.

10. The computer-readable medium of claim 8 wherein instruction (b) further includes the instruction of: in response to the mouse pointer being removed from within boundaries of the focused window, returning the focus to the active window and hiding any previously revealed portions of the inactive window.

11. The computer-readable medium of claim 7 wherein the inactive window is selected by a user when the mouse pointer is clicked on a visible portion of the inactive window.

12. The computer-readable medium of claim 11 wherein if the mouse pointer is clicked on a title bar of the inactive window, the inactive window is made the active window, wherein the display depths of the windows are reordered such that the active window is moved to a forefront of the screen.

13. A computer system for ordering on-screen windows for display, comprising:

a display screen;

a computer coupled to the display screen; and

a computer program for displaying multiple windows on the display screen in which an active window partially

overlaps an inactive window and hides a portion of the inactive window, the computer program having instructions for:

(a) monitoring movement of the mouse pointer around the screen; and

(b) in response to the inactive window being selected with the mouse pointer, causing the inactive window to gain focus to enable the inactive window to receive event signals even though a portion of the inactive window remains hidden, whereby original display depths of the windows is maintained.

14. The system of claim 13 wherein instruction (b) further includes the instruction of: in response to the mouse pointer being moved over the hidden portion of the window, revealing the hidden portion of the window on the screen and enabling that portion of the window to also receive event signals.

15. The system of claim 14 wherein instruction (b) further includes the instruction of: in response to the mouse pointer being moved away from the revealed hidden portion of the inactive window, hiding the previously revealed portion of the inactive window.

16. The system of claim 14 wherein instruction (b) further includes the instruction of: in response to the mouse pointer being removed from within boundaries of the focused window, returning the focus to the active window and hiding any previously revealed portions of the inactive window.

17. The system of claim 13 wherein the inactive window is selected by a user when the mouse pointer is clicked on a visible portion of the inactive window.

18. The system of claim 17 wherein if the mouse pointer is clicked on a title bar of the inactive window, the inactive window is made the active window, wherein the display depths of the windows are reordered such that the active window is moved to a forefront of the screen.

* * * * *