(12) UK Patent Application (19) GB (11) 2 360 921 (13) A

(43) Date of A Publication 03.10.2001

(21) Application No 0102630.1

(22) Date of Filing 02.02.2001

(30) Priority Data
(31) 09497629 (32) 03.02.2000 (33) US

(71) Applicant(s)
International Business Machines Corporation
(Incorporated in USA - New York)
Armonk, New York 10504, United States of America

(72) Inventor(s)
Daniel O Becker

(74) Agent and/or Address for Service
M J Jennings
IBM United Kingdom Limited, Intellectual Property
Department, Hursley Park, WINCHESTER, Hampshire,
SO21 2JN, United Kingdom

(51) INT CL^7
G06F 3/037

(52) UK CL (Edition S)
H4T TBLA

(56) Documents Cited
US 5668964 A

(58) Field of Search
UK CL (Edition S) H4T TBLA
INT CL^7 G06F 3/00 3/033 3/037
ONLINE: EPOQUE. INTERNET

(54) Abstract Title
Tabbed notebook having a common registry

(57) A method for manipulating a tabbed notebook window in which the pages from an existing tabbed notebook window may be moved to drop points in a graphical user interface outside a display space allocated to the existing tabbed notebook window. If the drop point to which a tabbed page is not within the display space allocate to an existing tabbed notebook window, a new tabbed notebook window is created to contain the moved tabbed page. A shared common container is provided to register the existing tabbed notebook windows. When a new tabbed notebook window is created, it registers itself in the shared common container. Each time a tabbed page is received outside of the tabbed notebook window in which it is currently located, the list of registrations in the shared common container is queried to determine whether the page has been moved to another existing tabbed notebook window. If not, a new tabbed notebook window is created to receive the tabbed page.

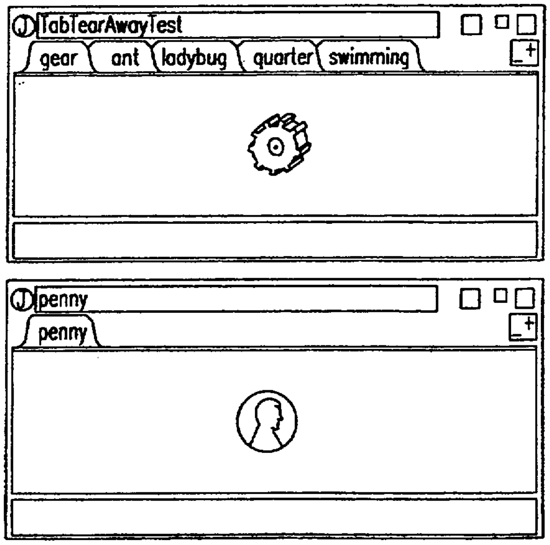


FIG. 4

GB 2 360 921 A

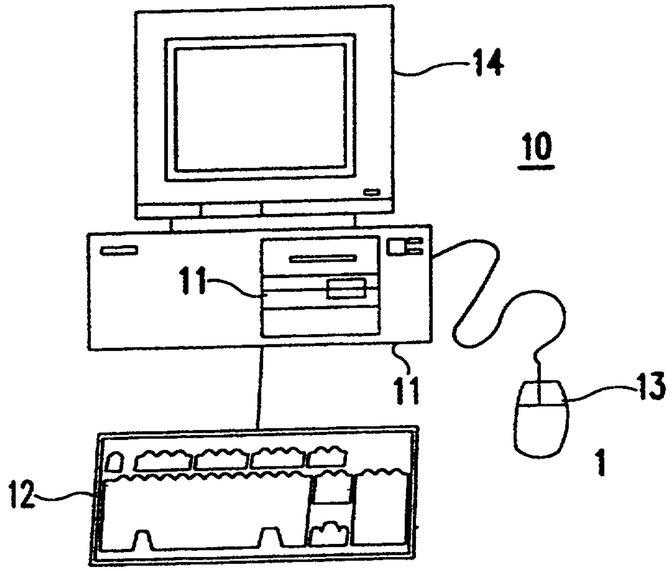
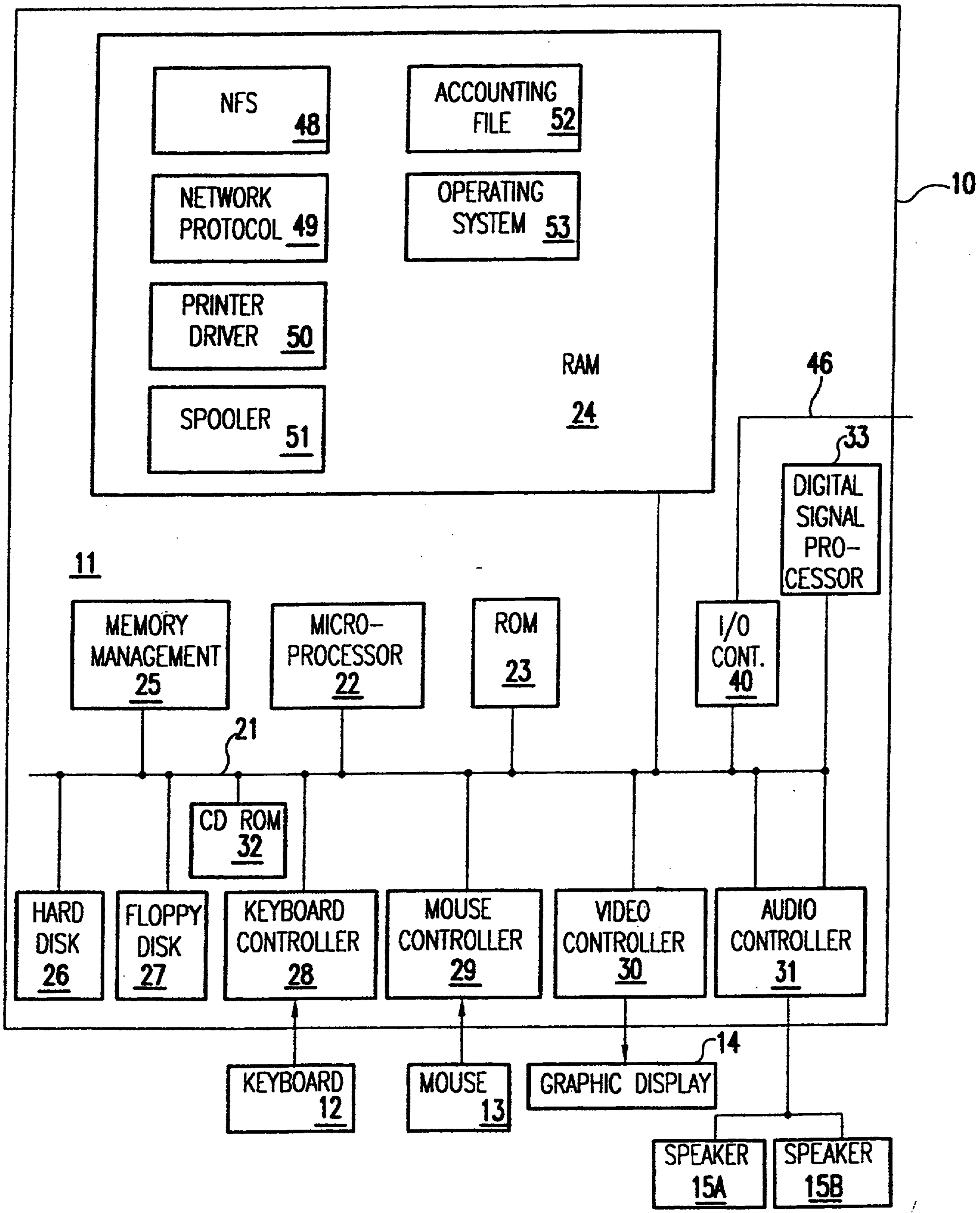


FIG. 1



NFS 48
ACCOUNTING FILE 52
NETWORK PROTOCOL 49
OPERATING SYSTEM 53
PRINTER DRIVER 50
SPOOLER 51
RAM 24
10
46
33
DIGITAL SIGNAL PROCESSOR
11
MEMORY MANAGEMENT 25
MICRO-PROCESSOR 22
ROM 23
I/O CONT. 40
21
CD ROM 32
HARD DISK 26
FLOPPY DISK 27
KEYBOARD CONTROLLER 28
MOUSE CONTROLLER 29
VIDEO CONTROLLER 30
AUDIO CONTROLLER 31
14
KEYBOARD 12
MOUSE 13
GRAPHIC DISPLAY
SPEAKER 15A
SPEAKER 15B

FIG. 2

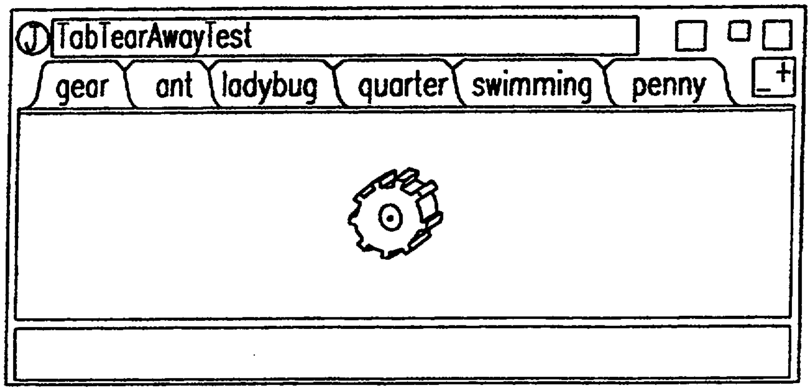

TabTearAwayTest
gear
ant
ladybug
quarter
swimming
penny

FIG. 3

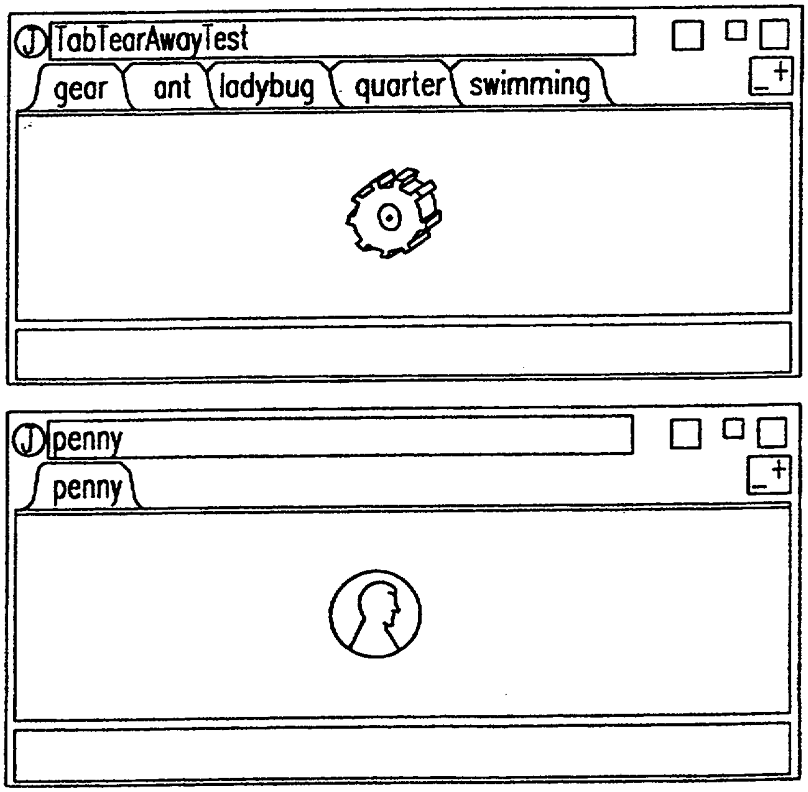
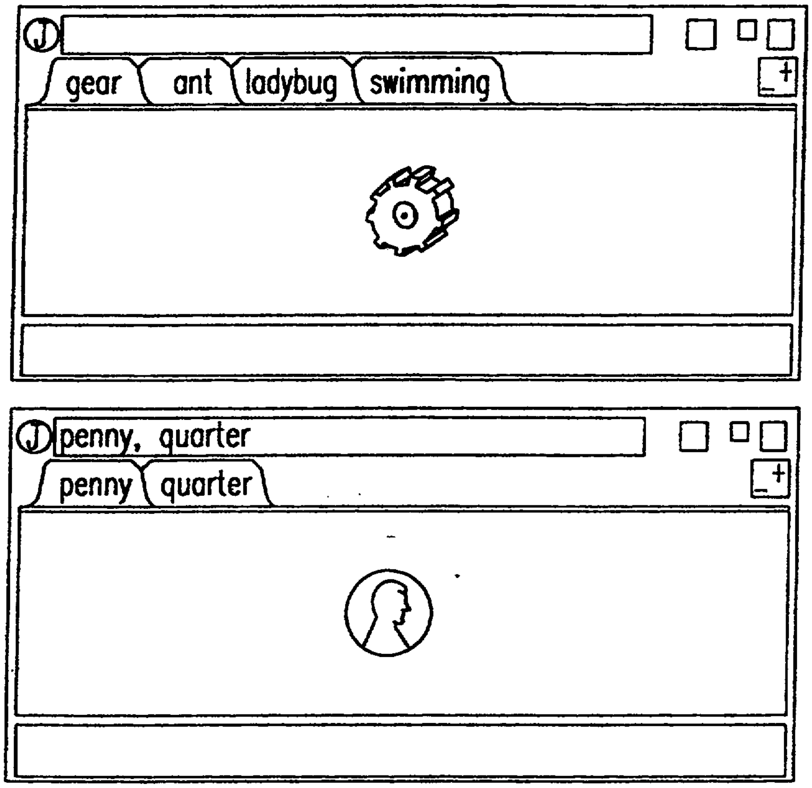

TabTearAwayTest
gear
ant
ladybug
quarter
swimming
penny
penny

FIG. 4



penny, quarter

FIG. 5

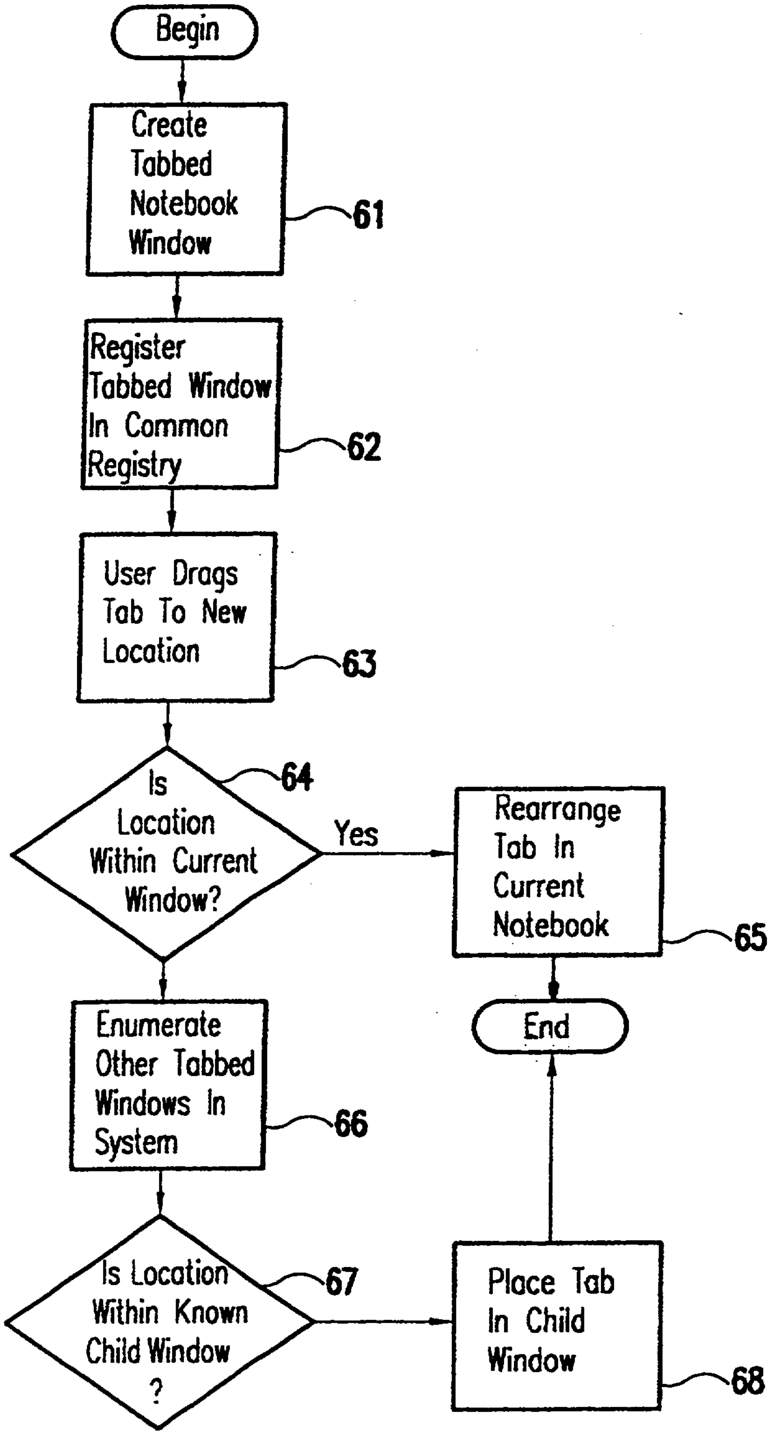

Begin
Create Tabbed Notebook Window
61
Register Tabbed Window In Common Registry
62
User Drags Tab To New Location
63
Is Location Within Current Window?
64
Yes
Rearrange Tab In Current Notebook
65
End
Enumerate Other Tabbed Windows In System
66
Is Location Within Known Child Window ?
67
Place Tab In Child Window
68

FIG. 6

TABBED NOTEBOOK HAVING A COMMON REGISTRY

BACKGROUND OF THE INVENTION

The present invention is directed to a graphical user interface employing tabbed notebook metaphor and more particularly a tabbed notebook metaphor with tear-away pages using a common registry.

Many windowed computer user interfaces employ a tabbed notebook metaphor. The tabbed notebook interface resembles a paper folder commonly used in filing cabinets, presentations and administrative assistance. Computer user interfaces commonly employ tabbed notebooks to present multiple pages of information to a user, allowing the user to select and flip from page to page using the tabs at the edge of the page. However, as many tabbed pages fill the notebook, the user interface becomes difficult to use. Often the multitude of tabs fills the computer window, preventing a user from seeing all the tabs at the same time. Also, certain repetitive tasks may require actions on a small sub-set of tabbed pages. This often leads to clumsy flipping of tabbed pages when many tabs are present. A user may wish to view multiple pages at once which is nearly impossible to do with the typical current tabbed notebook metaphor. What is needed is a method for manipulating and repositioning tabbed pages in a tabbed notebook in a computer user interface.

SUMMARY OF THE INVENTION

The present invention allows a user to grab and drag a tabbed page and move it to a new window outside the original tabbed notebook window. Optionally, a new window may be resized and repositioned by the user allowing a high degree of flexibility and usefulness in the computer user interface. Multiple tabbed pages may be moved from the original notebook to the new window. The computer user interface allows creation of multiple new windows to receive and display tabbed pages taken from the original tabbed notebook window. When a new window is created to receive a tabbed page from an original tabbed notebook, the new window is called a child window of the parent window originally containing the tabbed notebook page. Tabbed pages may be moved and shared among any of the tabbed notebook windows. The new child windows may return tabbed pages to the original parent notebook when closed. These operations are accomplished by using a common shared container or registry for tabbed notebook windows. When a tabbed notebook window is created, it registers itself in the common shared container. The notebook windows de-register themselves from the common shared container when they are closed or destroyed. The common shared container lists all known tabbed notebook windows in existence in the system.

BRIEF DESCRIPTION OF THE DRAWINGS

Preferred embodiments of the invention will now be described in more detail, by way of example, with reference to the accompanying drawings in which:

Fig. 1 illustrates a computer system in which the system of the invention is implemented;

Fig. 2 is an architectural block diagram of the computer illustrated in Fig. 1;

Fig. 3 illustrates a display of a tabbed notebook window displayed by the system of the present invention;

Fig. 4 illustrates a creation of a new tabbed notebook window according to the teachings of the present invention;

Fig. 5 illustrates the operation of moving another tabbed page to the new tabbed notebook window; and

Fig. 6 is a flow diagram of the process used in the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

The invention may be run on a computer or on a collection of computers under a number of different operating systems. The computer could be, for example, a personal digital assistant (PDA), a personal computer, a mini computer, mainframe computer or a computer running in a distributed network of other computers. Although the specific choice of computer is limited only by disk and disk storage requirements, computers in the IBM series of personal computers could be used in the present invention. In the alternative, the computer system might be in the IBM RISC System/600 line of computers which run on the AIX operating system. The various models of the IBM personal computers, RISC System/600™ computers and the AIX™ operating system are described in many publications of the IBM Corporation.

In Fig. 1, a computer 1 0, comprising a system unit 1 1, a keyboard 12, a mouse 13 and a display 14 are depicted. The screen 16 of display device 14 is used to present the graphical user interface (GUI). The graphical user interface supported by the operating system allows the user to use a point and shoot method of input, i.e., by moving the mouse pointer to an icon representing a data object at a particular location on the screen 16 and pressing one of the mouse buttons to perform a user command or selection.

Fig. 2 shows a block diagram of the components of the personal computer shown in Fig. 1. The system unit 1 1 includes a system bus or plurality of system buses 21 to which various components are coupled and by which communication between the various components is accomplished. The microprocessor 22 is connected to the system bus 21 and is supported by read only memory (ROM) 23 and random access memory (RAM) 24 also connected to system bus 21. A microprocessor in the IBM multimedia PS/2 series of computers is one of the Intel family of microprocessors including the 386 or 486 microprocessors. However, other microprocessors including, but not limited to, Motorola's family of microprocessors such as the 68000, 68020 or the 68030 microprocessors and various Reduced Instruction Set Computer (RISC) microprocessors manufactured by IBM, Hewlett Packard, Sun, Intel, Motorola and others may be used in the specific computer.

The ROM 23 contains among other code the Basic Input-Output system (BIOS) which controls basic hardware operations such as the interaction and the disk drives and the keyboard. The RAM 24 is the main memory into which the operating system and application program are loaded. The memory management chip 25 is connected to the system bus 21 and controls direct memory access operations including passing data between the RAM 24 and hard disk drive 26 and floppy disk drive 27. The CD ROM 32, also coupled to the system bus 21, is used to store a large amount of data, e.g., a multimedia program or large database.

Also connected to this system bus 21 are various 1/0 controllers: The keyboard controller 28, the mouse controller 29, the video controller 30, and the audio controller 31. As might be expected, the keyboard controller 28 provides the hardware interface for the keyboard 12, the mouse controller 29 provides the hardware interface for mouse 13, the video controller 30 is the hardware interface for the display 14, and the audio controller 31 is the hardware interface for the speakers 15a and 15b. The speakers 15a and 15b may be used to present audio to the user. An 1/0 controller 40 such as a Token Ring Adapter enables communication over a network 46 to other similarly configured data processing systems.

One of the preferred implementations of the present invention is as a set of instructions in a code module resident in the random access memory 24. Until required by the computer system, the set of instructions may be stored in another computer memory, for example, in the hard disk drive 26, in an optical disk for eventual use in the CD ROM 32 or in a floppy disk for eventual use in the floppy disk drive 27. As shown in the figure, the operating system 50 and presentation manager 52 are resident in RAM 24. In this example, the invention is embodied as an adjunct module onto the operating system. Alternatively, the graphical user interface could be incorporated into a standalone application 54, e.g., in a monitor program. The monitor program 54 may monitor all the nodes in the network or only the subset of nodes which are part of the parallel database 56.

Currently, the IBM OS/2 Operating System Version 2.1, provides a control device called a "notebook" which is useful when multiple panels or pages of data are displayed for review.

Fig. 3 shows an example of a typical tabbed notebook window in a computer user interface. Six pages of information are presented to the user in the window. At the top of the notebook is a set of six tabs marked with alphanumeric data in the form of a title for the information on that page (here marked "gear", "ant", "ladybug", "quarter", "penny", and "swimming"). In this example, the page of information is a simple picture corresponding to the tab title. A user may select and navigate tabbed pages in the notebook by clicking on the tab with the mouse or using a keyboard key sequence. Alternatively or in addition to the alphanumeric identification data on the tabs, the tabs may be color coded. The tabbed pages can be manipulated to display any one of the tabbed pages on the screen of the display as the page with the tab entitled "gear" is displayed in Fig. 3.

The system of the invention permits manipulation and repositioning of tabbed pages from a tabbed notebook like that shown in Fig. 3 by grabbing and dragging a tabbed page and moving it to a new window or to an existing window outside of the original notebook window. Multiple tabbed pages may be moved from the original notebook to a new or existing window. A computer user may create multiple new windows outside of the original tabbed notebook window. Tabbed pages may be moved and shared among the new windows and the original window. Tabbed pages are returned to the original tabbed notebook when a child window containing the tabbed page is closed. When a tabbed page is moved outside the window from which it is selected, and it is not moved to an existing tabbed notebook window, a new tabbed notebook window is created to receive the tabbed page.

New tabbed notebook windows register themselves in a shared common container when they are created. The tabbed notebook windows also de-register themselves from the shared common container when they are destroyed or closed. The shared common container lists all known tabbed notebook windows in the system. The information listed for each notebook window in the shared common container comprises references to the corresponding tabbed notebook windows. From each registration, the assembly of the data comprising the corresponding shared notebook window can be located and the bounds of the corresponding tabbed notebook window can be obtained. In this manner, the list registered in the shared common container constitutes a reference providing an indication of all of the tabbed notebook windows in existence and their boundaries. The shared common container of tabbed notebook window references facilitates searches for tabbed notebook windows and enables tabbed pages that are moved in user interface to be placed in an existing notebook or new notebook window quickly. In the Java programming language, the shared container can be implemented as a static Vector with each element for the Vector representing one tabbed notebook window. When moving a notebook tab, the static Vector is enumerated and the tabbed page is placed in an existing element of the Vector, or in the case where the notebook tabbed page is dropped on an area of the screen outside on the existing window, a new tabbed notebook window is created and added to the shared common container.

Each tabbed page is considered to be owned by the tabbed notebook window in which it was originally created and each tabbed page contains an owner reference identifying the tabbed notebook window by which it is owned. When a tabbed notebook window is closed or destroyed, the tabbed pages are returned to their owners, as indicated by the owner references contained by the page. Alternatively, the tabbed notebook pages will be destroyed when the tabbed notebook window which owned the tabbed pages no longer exists. Fig. 4 illustrates an example of an operation of the system of the invention. In this example, the user has dragged one of the tabbed pages into a new window. This action is analogous to "tearing-away" or removing a tabbed page from a physical notebook of file folder. One way of dragging and moving a tabbed page to a new window is: 1) moving the mouse pointer over one of the tabs, depress and hold a mouse button, move the mouse pointer to an area outside the original tabbed notebook window, and release the mouse button to drop the tabbed page.

When the user drops the tabbed notebook page, the shared common container of registrations of tabbed notebook windows is accessed and each registration in the shared container is enumerated. In this action, each registration in the shared common container is processed and the bounds of each registered tabbed notebook window is queried. If no existing tabbed notebook window contains the drop point for the tabbed notebook page, a new tabbed notebook window is created and registered with shared common container. The tabbed page is then placed in the new tabbed notebook window. Like the original window, the new window may be resized and positioned anywhere on the screen. If the drop point is in an existing tabbed notebook window, the tabbed page is placed in the existing tabbed notebook window.

The user may repeat the process and drag a second tabbed page from the original tabbed notebook window to the new window. An example of this operation is shown in Fig. 5. When the user drops the tabbed notebook page, the shared common container is accessed. Each tabbed notebook window registration in the shared common container is enumerated and the window bounds are queried. The tabbed notebook window containing the drop point receives the tabbed notebook page. If the drop point were outside of any existing tabbed notebook window, another new tabbed notebook window would be created. In the example of Fig. 5, no new tabbed notebook window is created or registered with the shared common container.

The displayed tabbed notebook windows may be viewed and operated independently. Tabbed pages may be moved from one window to any other window. Closing a child window returns the tabbed pages to their original tabbed notebook. When a tabbed notebook window is closed, it removes its registration from the shared common container. The tabbed notebook pages of the closed window are returned to the "owning window" which is the window in which the pages were originally created as tabbed pages and made available for display as tabbed notebook pages.

Fig. 6 is a flowchart illustrating the operation of the system when being used to manipulate tabbed pages in tabbed notebook windows in accordance with the invention. As indicated in Fig. 6, the program starts with an instruction sequence 61 in which a tabbed notebook window is created in response to a user command. The creation of the tabbed notebook window causes a notebook window display like that shown in Fig. 3 to be generated on the screen 60 of the display device 14. Then in instruction sequence 62, the tabbed notebook window created in instruction sequence 61 registers itself in the shared common container. In the example illustrated, the user selects a tab, drags the selected tabbed page to a drop point on the display screen 60 and releases the tabbed page at the drop point. The program responds to this input control by the user in instruction sequence 63 to show the motion of the tabbed page on the display screen to the drop point. Following instruction sequence 63, the program enters a decision sequence 64 in which the program determines whether or not the drop point is within the window created in instruction sequence 61. If the tab is still within this window, then the program branches to instruction sequence 65 wherein the tabbed page is rearranged in the notebook so that the tabbed page is displayed. In this example, this completes the program operation in response to the user input commands. The user, instead of actually dragging a tab, may merely select the tab by clicking on it with the mouse in which case, in instruction sequence 63, the program will detect that the tab has been selected and then in decision sequence 64, the program will determine that the drop point is within the current window and will branch to instruction sequence 65 to display the selected tabbed page.

If, in decision sequence 64, the program determines that the drop point to which the tab has been moved is not within the current window, the program proceeds to instruction sequence 66 in which the shared common container is accessed and the registrations in the shared common container are enumerated. The bounds of the tabbed notebooks windows registered in the shared common container are queried to determine whether the drop point is within one of the tabbed notebook windows registered in the shared common container. If the drop point is not within the boundaries of a tabbed notebook window, the program returns to instruction sequence 61 and a new child tabbed notebook window is created to contain and display the tabbed page that has been moved.

This new window with the tabbed page will then be displayed on the screen 60 of the display 14. The program will then proceed to instruction sequence 62 wherein the newly created tabbed notebook window registers itself in the shared common container. If the user then drags another tabbed page in one of the notebook windows to a new location, the process will proceed into the decision sequence 64 and then into the instruction sequence 65 or instruction sequence 66 and decision sequence 67 and the process will reiterate as described above.

If it is determined in decision sequence 67, that the drop point to which a tabbed page has been moved is within the boundaries of a tabbed notebook window registered in the shared common container, the program will branch to instruction sequence 68 in which the tabbed page is placed and displayed in the tabbed notebook window containing the drop point.

If a child notebook window is closed or destroyed, the program receives the owner reference stored on each tabbed page in the closed or destroyed notebook window and returns the pages to the tabbed notebook window or windows, which are designated in the owner references. If the owner's tabbed notebook window no longer exists, the tabbed pages from the closed or destroyed window are destroyed.

CLAIMS

1. A method of displaying and manipulating data on a computer terminal display screen comprising displaying data on said display screen in the format of a first tabbed notebook window containing tabbed pages, at least some of said tabbed pages having tabs which are visible on said display screen, providing a shared common container for registering tabbed notebook windows which are in existence and capable of being displayed on said display screen, registering said first tabbed notebook window in said shared common container, selecting a first tabbed page from said first tabbed notebook window and moving said first tabbed page to a drop point outside of said first tabbed notebook window, accessing said shared common container to determine whether the drop point is within the bounds of a tabbed notebook window registered in said common container, if said drop point is within the bounds of a tabbed notebook window registered in said shared common container, then placing the first tabbed page in such tabbed notebook window, and if said drop point is not within the bounds of a tabbed notebook window registered in said shared common container, then creating a second tabbed notebook window, placing said first tabbed page in said second tabbed notebook window, and registering said second tabbed notebook window in said shared common container.

2. A method as recited in claim 1, further comprising providing in each of the pages contained by said second tabbed notebook window an owner reference to the tabbed notebook window in which such tabbed page was originally created as a tabbed page, and returning each tabbed page contained by said second tabbed notebook window to the tabbed notebook window indicated by the owner reference of such tabbed page when said second tabbed notebook window is closed or destroyed.

3. A method as recited in claim 1 or claim 2, further comprising selecting a second tabbed page from the tabbed pages in said first notebook container and moving said second tabbed page to a second drop point, accessing said shared common container to determine whether said second drop point is within the bounds of a tabbed notebook window registered in said shared common container, and if said second drop point is within the bounds of a tabbed notebook window registered in said shared common container, then placing said second tabbed page in such tabbed, notebook window, if said second drop point is not within the bounds of a tabbed notebook window registered in said shared common container, creating a third tabbed notebook window, placing said second tabbed page in said third tabbed notebook window, and registering said third tabbed notebook window in said shared common container.

4. A method as recited in claim 3, further comprising providing on each of said tabbed pages an owner reference indicating the tabbed notebook window in which such tabbed page was originally created as a tabbed page, and when a tabbed notebook window is closed or destroyed, returning the tabbed page or pages contained by such closed or destroyed tabbed notebook window to the tabbed notebook window or windows indicated by the corresponding owner references of said tabbed pages.

5. A system for displaying data comprising a processor, a graphic display having a display screen controlled by said processor, a computer memory storage containing a computer program controlling the operation of said data processor to display data on said display screen in the form of tabbed notebook windows, said computer program defining a shared common container for registering references to tabbed notebook windows displayed on said screen, said computer program controlling said data processor to respond to user commands to move said tabbed pages to drop points and to determine from the registrations in said shared common container whether said drop points are within the bounds of tabbed notebook windows registered in said shared common container, said computer program controlling said data processor to place a selected tabbed page in a selected tabbed notebook window when it is determined that the drop point to which the selected tabbed page has been moved is within the bounds of the selected tabbed notebook window, said computer program controlling said data processor to create a new tabbed notebook window when the drop point to which the selected tabbed page has been moved is not within the bounds of a tabbed notebook window registered in said shared common container and registering said new tabbed notebook window in said shared common container.

6. A system as recited in claim 5, wherein said tabbed pages include owner references indicating the tabbed notebook window or windows in which said tabbed notebook pages were originally created as tabbed pages, said computer program controlling said data processor to return the tabbed page or pages in a tabbed notebook window to the tabbed notebook window or windows indicated by said owner references when the tabbed notebook window currently containing the tabbed notebook page or pages is closed or destroyed.

7. A system as recited in claim 5 or claim 6, wherein said data processor is operable to adjust the sizes and positions of the tabbed notebook windows displayed on said screen.

8. A computer program for controlling the operation of a data processing apparatus on which it runs to present data on a display, comprising:

means to display said data in the form of tabbed notebook windows, said tabbed notebook windows each containing one or more tabbed pages;

means defining a shared common container registering references to said tabbed notebook windows;

means responsive to user input commands to move tabbed pages from said tabbed notebook windows to drop points on said screen;

means to determine from the registrations in said shared common container whether each of said drop points is within the bounds of a tabbed notebook window registered in said shared common container, and to place a tabbed page in a selected tabbed notebook window when the drop point to which a tabbed page has been moved is within the bounds of said selected tabbed notebook window; and

means to create a new tabbed notebook window when the drop point to which a tabbed page has been moved is outside the bounds of the tabbed notebook windows registered in said shared common container and to place such tabbed page in said new tabbed notebook window.

9. A computer program as recited in claim 8, further comprising means responsive to the closing or destruction of a tabbed notebook window to return the tabbed page or pages contained in such closed or destroyed tabbed window to the tabbed notebook window or windows in which such tabbed page or pages were originally created as tabbed pages.

10. A computer program as recited in claim 9, wherein said tabbed pages contain owner references identifying the tabbed notebook windows in which said tabbed notebook pages were originally created as tabbed pages, said means to return said tabbed page or pages to the tabbed notebook window or windows in which such tabbed page or pages were originally created being responsive to said owner references to return the tabbed page or pages to the tabbed notebook window or windows in accordance with said owner references.

CHARTER MARK

PATENTS • DESIGNS • The Patent Office • COPYRIGHT • TRADE MARKS

INVESTOR IN PEOPLE

Application No:	GB 0102630.1	**Examiner:**	R. F. King
Claims searched:	All	**Date of search:**	30 July 2001

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.S): H4T[TBLA]

Int Cl (Ed.7): G06F3/00,3/033,3/037

Other: ONLINE: EPOQUE, INTERNET

Documents considered to be relevant:

Category	Identity of document and relevant passage	Relevant to claims
X	US 5,668,964 A [Wall Data Inc] See whole doc.	1, 5, 8

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
		E	Patent document published on or after, but with priority date earlier than, the filing date of this application.
&	Member of the same patent family		