(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 953 599 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
06.08.2008 Bulletin 2008/32

(51) Int Cl.:
G03G 15/00 *(2006.01)*

(21) Application number: **08150752.7**

(22) Date of filing: **29.01.2008**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MT NL NO PL PT RO SE SI SK TR
Designated Extension States:
AL BA MK RS

(30) Priority: **31.01.2007 US 669721**
31.01.2007 US 669746
29.03.2007 US 692957
30.03.2007 US 729850

(71) Applicant: **Ricoh Company, Ltd.**
Tokyo 143-8555 (JP)

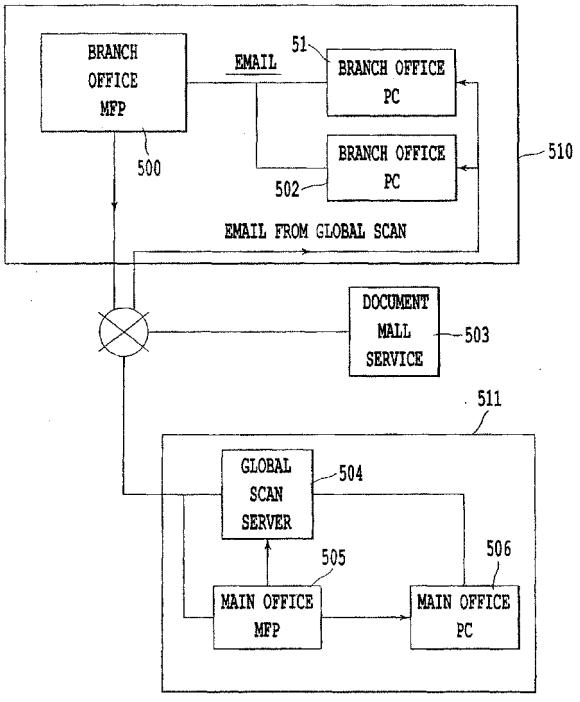
(72) Inventors:
- **Kitada, Hiroshi**
West Caldwell, NJ 07006 (US)
- **Wang, Helen**
West Caldwell, NJ 07006 (US)
- **Tang, Weiyun**
West Caldwell, NJ 07006 (US)
- **Jenning, Andrew**
West Caldwell, NJ 07006 (US)
- **Xu, Shen**
West Caldwell, NJ 07006 (US)
- **Wong, Lana**
West Caldwell, NJ 07006 (US)
- **Vellanki, Revathi**
West Caldwell, NJ 07006 (US)

(74) Representative: **Schwabe - Sandmair - Marx**
Stuntzstrasse 16
81677 München (DE)

(54) System and method of seamlessly switching between embedded and external functions on a multi-function printer

(57) A method and apparatus for launching a host application of an image handling device, determining external functions for the image handling device, storing information regarding available external functions determined by the determining in a configuration file, presenting a graphical interface that includes selectable graphical indicia representing each function accessible on the image handling device including the at least one embedded function and the available external functions and executing the at least one embedded function and the determined external functions based on a selection of the corresponding graphical indicia.

FIG.1



EP 1 953 599 A1

Description

BACKGROUND OF THE INVENTIONS

[0001] The present invention relates to a system and method of seamlessly switching between embedded and external functions on a multi-function printer.

[0002] The present invention relates to a system and method to allow the removal and addition of functions on a multi-function printer.

[0003] The present invention relates to a system and method for image thumbnail/preview on an image processing device.

[0004] The present invention is directed to methods and computer-based systems for authenticating a user of an image processing system.

[0005] Conventionally, Multi-Function Printers (MFPs) could be configured to use embedded functions, functions that were installed in a storage device in the MFP, or alternatively MFPs could be configured to use predetermined external functions, the predetermined external functions being functions that were installed on the MFP but used an external server to accomplish functions other than image scanning/printing. However, once an MFP was configured to use embedded functions, it was not possible for the user to use any external functions. Similarly, once a MFP was configured to use the predetermined external functions the user was unable to use the embedded functions. Thus when the external server became unavailable the user was unable to use the function of the MFP without a service person physically visiting the MFP to change the configuration of the MFP.

[0006] Additionally, when the MFP was configured to use the predetermined external functions each time a new external function was added to a network accessible by the MFP, a service person was required to physically visit the site of the MFP to upgrade the configuration of the MFP to include the newly added external function.

[0007] Conventionally, when a user wanted to install a new service on a multi-function printer ("MFP") such as a file storage application, a new application had to be installed to the MFP that made these services available, as shown in Figure 19.

[0008] In order to leverage these applications in an MFP context, the applications are installed by a service person that would travel to the physical site of the multifunction printer and install the applications manually. In addition if a user or a retailer wanted to remove an application from the MFP a service person was required to visit the physical site of the MFP and physically uninstall the application.

[0009] The present inventors have determined that users of conventional multifunction devices are unable to determine until after printing or saving if a scan on a multifunction device had been successful or if the input pages need to be scanned again. In addition, the inventors have determined that users of conventional devices are unable to verify the scan quality of the job and verify things such as barcodes, signatures, anti-copying marks or watermarks that developed on the scan.

[0010] Over the past several years, there has been an increase in the number and types of document-related applications available over networks. These applications can include document management systems, such as those specializing in managing documents of various specific contents, for example medical, legal, financial, marketing, scientific, educational, etc. Other applications include various delivery systems, such as e-mail servers, facsimile servers, and/or regular mail delivery. Yet other applications include document processing systems, such as format conversion and optical character recognition systems. Further applications include document management systems used to store, organize, and manage various documents. These document management systems used to store, organize, and manage various documents may be referred to as "backend" applications.

[0011] Various systems for accessing these network applications from image processing devices (e.g., scanners, printers, copy machines, cameras) have been contemplated. One system associates a computer with each image processing device for managing the documents with the network applications. The computers communicate with the various network applications to enable the use of the applications by the user of the image processing devices. For example, the computers request and receive from the network applications information about the format and content of the data required by the applications to manage the documents. The computers process this information and configure the image processing devices to provide the correct format and content.

[0012] The image processing devices also typically incorporate some type of monitoring system to track the resource usage of the image processing device. These monitoring systems authenticate a user and provide the ability to track copy, print and fax activities based on attributes such as document name, printer, port, date and time, paper size, finishing options and choice between black and white or color. Such a process allows billing reports, invoices, etc. to be generated based on the authenticated user's actions at the image processing device. Thus, before operating an image processing device that includes such a monitoring system, the user must first be authenticated with the monitoring system. Such authentication typically involves the entry of some sort of personal information or data from the user.

[0013] Once a user is granted access to the image processing device, an additional authentication step typically is performed to gain access to one or more of the above noted backend applications. For example, the user may additionally log into a server or network to which the image processing device is connected in order to gain access to this server or network.

[0014] Thus, the present inventors have realized that current systems may require a user to log in multiple times at a single image processing device in order to both gain access to the image processing device and a back-

end application associated with the image processing device. The inventors have realized that such redundancy is burdensome for the user of the image processing device, and may force users to memorize different user authentication information associated with each necessary log in procedure.

SUMMARY OF THE INVENTIONS

[0015] The present inventions provide, inter alia, a method including the step of launching a host application of an image handling device. The image handling device includes at least one embedded function and a network interface. The method further includes determining external functions for the image handling device. The external functions utilize the network interface and including operations which are performed remotely from the image handling device. The method also includes storing information regarding available external functions determined by the determining in a configuration file and presenting a graphical interface that includes selectable graphical indicia representing each function accessible on the image handling device including the at least one embedded function and the available external functions. In addition, the method includes executing the at least one embedded function and the determined external functions based on a selection of the corresponding graphical indicia.

[0016] Also provided by the present inventions is an image handling device that includes a host application configured to provide a core service of the image handling device and including at least one embedded function and a network interface. Also included in the image handling device is an external function unit configured to determine external functions for the image handling device, the external functions utilizing the network interface and including operations which are performed remotely from the image handling device. A configuration file configured to store information regarding available external functions determined by the determining is included in the image handling device along with a display configured to present a graphical interface that includes selectable graphical indicia representing each function accessible on the image handling device including the at least one embedded function and the available external functions. In addition, an input unit configured to receive user input and to execute the selected at least one embedded function or the determined external function based on the selection of the corresponding graphical indicia is included in the image handling device.

[0017] The present inventions provide, inter alia, a method that includes launching a host application of an image handling device. The image handling device includes at least one plug-in and a corresponding set of services. The method further includes accessing a configuration file of the image handling device. The configuration file includes information regarding the activation status of each service corresponding to the at least one plug-in. Also included in the method is launching the at least one plug-in based on the information regarding the activation of each service. The plug-in provides the corresponding set of activated services to the host application. The method includes presenting a graphical interface that includes a graphical indicia of each activated service corresponding to each activated plug-in.

[0018] Also included in the present invention is a method that includes launching a host application on an image handling device. The image handling device includes an application layer, hardware and an operating system. The method also includes launching an activation manager. The activation manager determines which installed plug-ins and which services corresponding to the installed plug-ins to activate. The method includes reading a configuration file stored on the image handling device. The configuration file includes information identifying the image handling device, the user of the image handing device and the services corresponding to the installed plug-ins. The method includes determining the activation status of each service in the configuration file and updating the configuration file based on the determining. The configuration file is updated to include information regarding the activation status of each service. The method also includes generating a project array based on the number of installed plug-ins, generating a service array for each project and displaying a project array window. The project array window graphically displays each project included in the project array. The method includes determining the activation status of each project selected by a user and each corresponding service and displaying a main window and a default service window when a project is selected in the project array window and is determined to be active. The main window includes graphical indicia of the activated project services.

[0019] Also included in the present invention is an image handling device which includes a host application configured to provide the core service of the image handling device, a plug-in application configured to be programmatically invoked by the host application, an activation manager configured to control access to the plug-in application and a configuration file updated by the activation manager stored in a memory and including information regarding activation of the plug-in application and functions corresponding to the plug-in application, the host application configured to programmatically invoke the plug-in in accordance with information regarding activation in the configuration file.

[0020] The present invention provides, inter alia, a method of modifying images including the steps of scanning at least one document having multiple pages, displaying a thumbnail image of at least one of the pages of the scanned document, displaying a selectable graphical indicia corresponding to at least one operation for modifying at least one image of the scanned document, selecting the selectable graphical indicia and modifying the at least one image in accordance with said selecting.

[0021] Also included in the present invention is a meth-

od that includes scanning at least one document having multiple pages, displaying a preview image of at least one of the pages of the scanned document, displaying a selectable graphical indicia corresponding to at least one operation for modifying the preview image, selecting the selectable graphical indicia and modifying the preview image in accordance with said selecting.

[0022] The present inventors have determined that there is a need for a more efficient and customizable login procedure for image processing devices and the systems associated therewith. Specifically, the present invention is directed to using authorization information and user credentials based on an authentication procedure at the monitoring system for authentication at a second server and various backend applications without user intervention.

[0023] The present invention, therefore, is directed to a system and method for authenticating a user of an image processing system. User credentials are received at an authentication device corresponding to an image processing device, and transmitted to a first server remote from the authentication device. The validity of the user credentials are judged by comparing the received user credentials to authentication information stored at the first server, and a result of the judging is transmitted to the image processing device. The image processing device then requests access to a second server remote from the image processing device, and the second server transmits a request for the user credentials to the first server. After receiving the user credentials from the first server, the second server performs user authentication.

[0024] It is to be understood that both the foregoing general description of the invention and the following detailed description are exemplary, but are not restrictive, of the invention.

BRIEF DESCRIPTION OF SEVERAL VIEWS OF THE DRAWINGS

[0025] Other objects, features and advantages of the present invention will become more apparent from the following detailed description when read in conjunction with the accompanying drawings, in which:

Fig.1 is a block diagram showing an office set-up which includes several MFPs;

Fig. 2 is a block diagram of an application layer of an MFP according to an exemplary embodiment of the present invention;

Fig. 3 is a block diagram showing a Unified Client Application according to an embodiment of the present invention;

Fig. 4 is a block diagram showing an embedded function according to an exemplary embodiment of the present invention;

Fig. 5 is a block diagram showing an example of the embedded function of the Document Mall application;

Figs. 6A and 6B are process diagrams showing exemplary software architecture;

Figs. 7A, 7B, 7C, 7D(i), 7D(ii), 7E, 7F(i),7F(ii) and 7F(iii) are flowcharts showing a procedure of the process of the Server/Serverless Unified Client Main thread;

Figs. 8A and 8B are a flowchart of the Server/Serverless Unified Client Upload thread;

Fig. 9 shows an exemplary project array window which allows the user to select a project;

Fig. 10 shows a main window which allows the user to select different services of the selected project;

Figs. 11A-11F show an example of a config.xml file;

Fig. 12 is exemplary user interface of an eCabinet (TM) embedded function in which the eCabinet project main window and the eCabinet owner service window are displayed;

Fig. 13 is an exemplary user interface of an eCabinet embedded function in which the ecabinet project main window and eCabinet Folder service window are displayed;

Fig. 14 is an exemplary user interface of an eCabinet embedded function in which the eCabinet project main window and the scan settings service window are displayed;

Fig. 15 is an exemplary user interface of an eCabinet embedded function in which the eCabinet project main window and the scan settings service scan size sub-window are displayed;

Fig. 16 is an exemplary user interface of an eCabinet embedded function in which the eCabinet project main window and the job log service window are displayed;

Fig. 17 is an exemplary user interface of an Email embedded function in which the Email project main window and the Email selection window are displayed;

Fig. 18 is a block diagram showing a typical software configuration for a multi-function printer.

Fig.19 is a block diagram showing a typical application layer included on a conventional MFP;

Fig. 20 is a block diagram of an application layer of an MFP according to an exemplary embodiment of the present invention;

Fig. 21 is a block diagram showing a Unified Client Application according to an embodiment of the present invention;

Fig. 22 is a block diagram showing a plug-in according to an exemplary embodiment of the present invention;

Fig. 23 is a block diagram showing an example of a plug-in for a Document Mall application;

Figs. 24A and 24B are process diagrams showing exemplary software architecture;

Figs. 25A, 25B, 25C, 25D and 25E are flowcharts showing a procedure of the process of the Unified Client Main thread;

Figs. 26A and 26B are a flowchart of the Unified Cli-

ent Upload thread;
Fig. 27 shows an exemplary project array window which allows the user to select a project;
Fig. 28 shows a main window which allows the user to select different services of the selected project;
Figs. 29A-29B show an example of a config.xml file;
Fig. 30 is exemplary user interface of an eCabinet (TM) plug-in in which the eCabinet project main window and the eCabinet Owner service window are displayed;
Fig. 31 is an exemplary user interface of an eCabinet plug-in in which the eCabinet project main window and eCabinet Folder service window are displayed;
Fig. 32 is an exemplary user interface of an eCabinet plug-in in which the eCabinet project main window and the scan settings service window are displayed;
Fig. 33 is an exemplary user interface of an eCabinet plug-in in which the eCabinet project main window and the scan settings service scan size sub-window are displayed;
Fig. 34 is an exemplary user interface of an eCabinet plug-in in which the eCabinet project main window and the job log service window are displayed;
Fig. 35 shows a general configuration screen displayed by the Unified Client remote configuration web page;
Fig. 36 shows an eCabinet server screen displayed by the Unified Client remote configuration web page;
Fig. 37 shows a default scan settings screen displayed by the Unified Client remote configuration web page;
Fig. 38 is a block diagram showing a software configuration of the multi-function printer relative to the hardware and operating system according to an embodiment of the present invention;
Fig. 39 is a hardware configuration of the image forming apparatus according to an embodiment of the present invention; and
Fig. 40 is a block diagram showing a typical software configuration for a multi-function printer.
Figure 41 is a diagram showing an overview of the relationship between input documents, a multi-function device, a document preview, output documents and a server according to an exemplary embodiment of the present invention;
Figure 42A shows an exemplary MFD main window in addition to an exemplary preview page including a preview button;
Figure 42B shows an exemplary MFD main window in addition to an exemplary preview page including a highlighted preview button;
Figure 42C shows an exemplary MFD main window in addition to an exemplary preview page including an expanded Preview Page Range drop-down box;
Figure 43A shows an exemplary Thumbnail selection window;
Figure 44 shows a second exemplary Thumbnail selection window;
Figure 45 shows an exemplary preview detail window including a preview window in which the scanned page is displayed;
Figure 46 shows an exemplary deletion confirmation pop-up box that is displayed when the delete button is selected in the preview detail window;
Figure 47 shows an exemplary cancellation confirmation pop-up box that is displayed when the cancel button is selected in the single preview detail window or thumbnail selection window;
Figure 48 shows an exemplary undo confirmation pop-up box that is displayed when the undo button is selected in the single preview detail window or thumbnail selection window;
Figures 49A & 49B illustrate a manage services window used to remotely manage services on a GlobalScan server;
Figures 50A & 50B illustrate a server configuration settings window for the image preview function used to remotely change image preview function settings on the GlobalScan server;
Figures 51A & 51B illustrates a profile settings window used to services with profiles on the GlobalScan server;
Figures 52A & 52B illustrates a profile settings window with a image preview function related service linked with a profile on the GlobalScan server;
Figure 53 is a flowchart which illustrates the interaction between a MFD and a server for creating the preview window;
Figures 54A-C are an example of the return xml sent from the server to the MFD in response to an API Get Available Services query from the MFD;
Figures 55A and 55B are a flowchart which illustrates the interaction between a MFD and a server for performing a number of preview operations on the scanned job;
Figure 56 is a flowchart that illustrates the different types of function calls submitted by the MFD to the server;
Figure 57 is a flowchart showing the complete image preview process between the user, MFD and GlobalScan server;
Figure 58A is a flowchart showing the image preview initialization process between the user, MFD and GlobalScan server;
Figure 58B illustrates the return XML data sent by the server in response to the initialization request;
Figure 59A is a flowchart showing the image preview rotate operation process between the user, MFD and GlobalScan server;
Figure 59B illustrates the rotate request syntax and the return XML data sent by the server in response to the rotate request;
Figure 60A is a flowchart showing the image preview zoom operation process between the user, MFD and GlobalScan server;
Figure 60B illustrates the zoom request syntax and

the return XML data sent by the server in response to the zoom request;
Figure 61A is a flowchart showing the image preview pan operation process between the user, MFD and GlobalScan server;
Figure 61B illustrates the pan request syntax and the return XML data sent by the server in response to the pan request;
Figure 62A is a flowchart showing the image preview delete operation process between the user, MFD and GlobalScan server;
Figure 62B illustrates the delete request syntax and the return XML data sent by the server in response to the delete request;
Figure 63A is a flowchart showing the image preview next/previous operation process between the user, MFD and GlobalScan server;
Figure 63B illustrates the next/previous request syntax and the return XML data sent by the server in response to the next/previous request;
Figure 64A is a flowchart showing the image preview submit or cancel operation process between the user, MFD and GlobalScan server;
Figure 64B illustrates the submit, cancel or restore request syntax and the return XML data sent by the server in response to the submit, cancel or restore request; and
Figure 65 is a hardware configuration of the image forming apparatus according to an embodiment of the present invention.
Figure 66 is a block diagram showing an overall system configuration according to one embodiment of the present invention;
Figure 67 is a block diagram illustrating components of the image processing device and document manager server according to one embodiment of the present invention;
Figure 68 shows an example of a process for performing authentication at the monitoring system according to one embodiment of the present invention;
Figures 69 shows an example of a process for performing authentication at the monitoring system according to one embodiment of the present invention;
Figures 70A and 70B is a flowchart illustrating the steps for performing user authentication at the document manager server and backend applications according to one embodiment of the present invention;
Figure 71 is a flowchart showing a forced logout process according to one embodiment of the present invention;
Figure 72 is a diagram illustrating the overall system configuration of the system according to one embodiment of the present invention;
Figure 73 is a block diagram illustrating an image processing device according to one embodiment of the present invention;
Figure 74 is a schematic representation of an image processing device according to one embodiment of the present invention;
Figure 75 is a block diagram illustrating a server according to one embodiment of the present invention; and
Figure 76 is a schematic representation of a server according to one embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTIONS

[0026] Referring now to Figures 1-18 of the drawings wherein like reference numbers designate identical or corresponding parts throughout the several views and more particularly to Figure 1 thereof, this is illustrated a typical set-up including a main office 511 and a branch office 510 remote from the main office 511. The branch office 510 includes a branch office MFP 500 and a number of branch office personal computers (PCs) 501-502. Connected via a network is the main office 511. The main office 511 includes a GlobalScan(TM) server 504, a main office MFP 505 and a main office PC 506. Also included external to the main office 511 and the branch office is a document mall server 503.

[0027] An embodiment of the present invention enables seamless integration of both embedded and external functions on the MFP 500. Embedded functions are the functions that are installed on the MFP and are performed locally on the MFP. The embedded functions can be implemented used a plug-in. External functions are functions that utilize an external server such as a globalscan server to perform the function. The MFP is any printer or copier which includes multiple functions such as scanning, printing and/or faxing. Additionally, the MFP described above may include a copier that scans and prints a document in response to a single command, as scanning and printing are distinct functions.

[0028] The embedded functions of the MFP are configured as follows. Figure 2 shows an application layer 1 of an MFP including a unified client application 5. The unified client application 5 installed on a MFP includes a core application 6, the core application being an application that includes primary routines that serve the application. These primary routines typically carry out basic functions of the MFP including scanning, printing, copying, faxing, and communicating, for example. Included below the core application 6 is the activation manager 6b. The activation manager is the portion of the unified client application that determines the activation status of functions and corresponding services. In addition, the activation manager 6b generates a config.xml file 7 based on the determination of the activation status. The configuration file may be any type of configuration file including an extensible markup language such as XML, Standard Generalized Markup Language (SGML), GML, RDF/XML, RSS, Atom, MathML, XHTML, SVG, DSDL, XUL, MXML, EAD or Klip.

[0029] It should also be noted that according to one embodiment, the configuration file is able to be used in

a mixed brand environment. Even if, for example, several different brands of copiers are used in an environment such as an office or a building, each unique brand will be able to utilize the configuration file. In addition each different MFP may be able to load the unified client architecture and the embedded functions. Thus, each copier or multi-function device will be able to have the same basic interface and commands limited only be the functionality of the specific copier or multi-function printer in question. According to an alternate embodiment, different configuration files can be used by different manufactures or models.

[0030] The activation manager 6b provides the ability to activate or de-activate any embedded functions installed on the MFP or any external function operable by the MFP. As a result, all available embedded functionalities can be pre-installed on the MFP at the factory. When a user takes delivery of the MFP, the MFP may have some of the embedded functions activated or none of the embedded functions activated. If none of the embedded functions are activated, the user can activate the embedded functions as the user sees fit, such as when the need arises.

[0031] Different types of activation schemes are also available. For example, the user may be able to activate embedded or external functions for a limited time on a trial basis. Alternatively, the user may be able to buy, lease, or license the use of the embedded or external functions for a one time use or a time-based use such as for a week or a month. This could be useful for organizations that have higher demand during certain times of the year such as during tax season.

[0032] Additionally, the activation manager enables different fee options to be used on the MFP. For example, a user may pay a monthly subscription fee for use of embedded or external functions. When the user no longer needs the embedded or external functions, the use can be discontinued via a central server using the activation manager 6b. Additionally, embedded or external functions can be de-activated based on expiration dates. For example, a user could buy a six month subscription to an embedded or external function, and once the six months have expired the embedded or external function can be deactivated.

[0033] Further, if the user has the ability to activate embedded or external functions (such as the user has financial decision making abilities) the activation manager 6b enables the user to activate embedded or external functions in different ways. For example, using a device attached to the MFP, a user's ability to buy activation could be verified. Devices such as a smart card, biometrics, PIN code, magnetic strip card or proximity card could be used as well as other existing ID verification systems to enable the purchase/activation.

[0034] With respect to the activation, in the event that a user has control over a number of MFPs, the activation process can be accomplished remotely and collectively. Thus, a large number of MFPs can have embedded or external functions activated virtually simultaneously using a remote station. This enables uniformity in an organization and also saves a significant amount of time as there is no need to visit each MFP to perform an activation or deactivation.

[0035] The config.xml file 7 includes settings regarding unified client application 5 in addition to activation information. Additionally, embedded functions 8a...n are controlled by the core application 6.

[0036] Different types of embedded functions can be installed in the unified client application 6. For example, in the present example depicted in Figure 2, a Document Mall embedded function 8a, an eCabinet embedded function 8b and a generic embedded function 8n are installed.

[0037] Document Mall is an application for creating a secure online document storage, enabling an online shared workspace. Document Mall combines security with web-based document management and collaboration features delivered as an on-demand, document management and document imaging service.

[0038] Likewise, eCabinet is a network document repository that integrates with multi-function printers. eCabinet provides users with the ability to capture documents and automatically index them, providing archive security coupled with fast retrieval.

[0039] An embedded function generally includes programs or code for operating the hardware of the multi-function printer via the core application 6. An alternate illustration of the Unified Client application 5 shown in Figure 2 is set forth in Figure 3 which includes the core application 6, the activation manager 6b which, according to one exemplary embodiment, is separate from the core application, the config.xml file 7 including activation information and the embedded functions 8, the embedded functions including an activation reading part 9.

[0040] Figure 4 shows an example of an internal structure of an embedded function 8 according to one embodiment of the present invention. For example, the embedded function 8 may include a single service window 10a or may also include a number of service windows such as 10a...10n. Each service window 10a...n is a user interface that enables the user to interface with the service that corresponds to the service window 10a...n. The service window 10a...n also includes a corresponding activation reading part 15a....n. The activation reading part 15a...n is the first function performed when a service window 10a...n is executed and checks to see if the service window is active before any other functions are performed. Further explanation of the service window 10a...n will be discussed below with respect to Figures 12-16. The embedded function 8 may also include a single service data 11a or a number of service data elements 11a...n. The service data elements 11a...n also include an activation reading part 16a ... n. As noted above with regard to the service window activation reading part 15a...n, the activation reading part 15a...n ensures that the service data elements are activated. Each service

window 10a...n has corresponding service data 11a...n. In addition, the activation reading part of the service window 15a...n corresponds to the activation reading part 16a...n of the service data 11a...n. The service data 11a...n generally includes service name, service id, configuration data corresponding to the service window 10a...n, default service window data and run time data entered by users through service window 10a...n.

[0041] The embedded function 8 also includes a service data handler 12 and optionally may include a login window 13 and login data 14, in other words, an authentication user interface. The service data handler 12 is the portion of the unified client application that uploads data from the MFP to a receiving device. Included in the service data handler 12 is an activation reading part 17. The activation reading part 17 checks the activation information in the config.xml 7 file to ensure that the service data handler 12 is activated before any corresponding functions of the service data handler 12 are preformed. In each embedded function 8, there may be multiple service windows 10a...n and service data elements 11a...n. However, according to one preferred embodiment, there is only one service data handler 12. Other embodiments may have more than one service data handler 12.

[0042] Figure 5 depicts an example of the Document Mall embedded function 8a. The Document Mall service can be installed on the core application 6 as an embedded function 8. When the Document Mall embedded function 8a is installed in the unified client application 5, the services provided by Document Mall are extended to the MFP in which the unified client application 5 is installed. The Document Mall embedded function 8a preferably includes the optional login window 23 and login data 24. These options allow user names, passwords and accounts to be input and utilized by the embedded function 8a, allowing the embedded function 8a to restrict unauthorized users from use of the embedded function 8a.

[0043] The Document Mall embedded function 8a further includes several different service windows and service data. For example, in the Document Mall embedded function 8b, an e-mail service window 20a and a folder service window 20b are included. The email service window 20a is a user interface enabling a user to enter a Document Mall stored email address as a scan destination, while the folder service window 20b is a user interface that enables a user to select a Document Mall folder as the scan destination. Further, an e-mail service data 21a and folder service data 21b are also included. The e-mail service data 21a and the folder service data 21b correspond to the data generated by the e-mail service window 20a and the folder service window 20b, respectively. Both the service windows 20a and 20b and the service data elements 21a and 21b include activation reading parts 25a...b and 26a...b. The activation reading parts are the first functions performed by the service window/data element pairs and are used to ensure that the corresponding service window/data is activated and able to perform a function. The Document Mall embedded function 8b also includes a service data handler 22. In the example of the Document Mall 8a the service data handler 22 is used as an upload handler that merges both the e-mail service data 21a and the folder service data 21b into one upload.xml file, and sends the upload file to a Document Mall server through an https post command, for example. Other uses for the service data handler 22 not mentioned in this example are also possible. The service data handler 22 also includes an activation reading part 27. The activation reading part 27 allows the service data handler 22 to ensure activation before performing any functions.

[0044] Figure 6a shows the unified client software architecture structure. The unified client application 5, shown in Figures 2 and 3, is launched by a unified client main thread 30. In Figure 6a, the unified client main thread 30 is shown as including an activation reading part 30a, a project array 31 and a project array window 32. The main thread 30 initializes the core application 6 and uses the activation reading part 30a to read the config.xml file 7 in order to create the project array 31 based on the activation information found in the config.xml file 7. The config.xml file 7 includes activation information regarding several projects 33a...n, each project 33a...n corresponding to at least one embedded function 8, one external function or a set including embedded and external functions.

[0045] The project array 31 is a list of projects that are found to be active in the unified client application. The project array 31 is constructed by reading <project> tags included in the config.xml file 7. Further, the main thread 30 creates service arrays 34a...n for each project 33a...n by reading <service> tags and activation information included in the config.xml file 7. The service array is a list of the activated services installed under a respective project. The main thread 30 also displays the project array window 32. The project array window 32 is the first screen displayed when using or executing the unified client application 6. However, according to one embodiment of the invention, if only one project 33b is installed on the system, the project array window 32 will be bypassed. The project array window 32 displays project buttons for the user to select. When a project button is selected, the corresponding project 33a...n is invoked. It should also be noted that the project array window 32 may or may not display un-activated projects depending on the activation information found in the config.xml file 7. For example, in one configuration if no function is found to be activated by the main thread 30 for a project 33a.., n, a project 33a...n may not be created for the function making it seem to the user as if the project does not physically exist on the MFP. In contrast, in another configuration if a function is found not to be activated, the main thread 30 may create a project 33a...n corresponding to the function. However, when the user attempts to execute the project 33a...n via the project array window 32 instead of loading the corresponding main window 35 the project

33a...n, the system will give the user the ability to activate the project 33a...n. Additionally, the system may give the user the ability to see a demonstration of the project 33a...n or use the project for a limited time. A more detailed discussion of the activation process can be found below with reference to Figure 7E. The project array window 32 will be discussed in further detail below with respect to Figure 9.

[0046] Several projects 33a...n are shown in Figure 6a connected to the project array 31. Each project 33a...n includes an activation reading part 28a. The activation reading part 28a determines how the project 33a...n will be executed and which services 38a...n are included in a service array 34a...n. Each project 33a...n can manage a login/logout process of the project 33a...n through a corresponding login data 36 and login window 37. For example, if authentication is needed in the project 33a...n, a login window 37 can be used to display a login window which will be displayed before the user can begin accessing the project 33a...n, Once the login/logout button is pressed, a corresponding login and logout handler used by login data 36 will be called.

[0047] Further, the project 33a...n can control the post login process. For example, each service 38a...n can define its own post login process for its service window displayed by the service window 40a...n. When the authentication succeeds, the post login process of each service 38a...n will be called sequentially.

[0048] The login window displayed by the login window 37 described above, is an example of an authentication user interface ("UI") display. The login window, displayed by the login window 37, interfaces with the login data which is included in the login data 36 and includes an authentication process definition. Additionally, the login window used by the login window 37 can be implemented to request additional authentication information. As one example, for the Document Mall embedded function 8a, the Document Mall login window 23 may be implemented to include a place for users to enter account information. Other information may be utilized by the login window 37. Additionally, the login data can be accessed by each service window 40a...n and service data handler 12. Further, each project 33a...n includes a main window 35 and a service array 34a...n.

[0049] In Figure 6a, a main window 35 is associated with the project 33b. Although the main window 35 is only illustrated under project 33b, each project 33a...n may be implemented to include a main window. The main window 35 is used for service management for each service 38a...n which corresponds to a button, the button being a user selectable link to a service window, included in the main window 35. For example, in the Document Mall embedded function 8a example, the main window 35 includes buttons for scanned setting handling, document name input and login button handling. Another example of the main window 35 is discussed below with respect to Figure 10.

[0050] Included in each project 33a...n is a service array 34a...n. Each service array 34a...n includes a list of the activated services 38a...n. A service 38a...n is a function operable on the MFP. A project 33a...n may include a combination of embedded functions and external functions. A project 33a...n may also include only embedded functions or only external functions. In the case that a project includes both embedded and external functions and the functions conflict, such as both the external and embedded functions perform the same operation, priority information found in the config.xml file 7 is used to determine whether the embedded or external function is called when the service is selected in the main window 35. The present invention also includes the feature that if, for example, an external function has priority but the external function is unreachable due to a network problem the embedded function can seamlessly step in for the unavailable external function a will perform the function.

[0051] Each service 38a...n corresponding to an embedded function includes an activation reading part 29a...n, a service window 40a...n and service data 39a...n. The activation reading part 29a...n is the first operation activated by a service 38a...n and determines if the service is activated. A service window included in the embedded function 40a...n displays a service window user interface. Further, the service window 40a...n performs the post-login process or gets and sets default values in the service data 39a...n. For example, in the post-login process of the Document Mall embedded function 8a example, a Document Mall folder service downloads the user's folder list and sets the user's folder as the default folder destination. The service window 40a...n also performs interactive operations with the user to interact and update the service data in the service data 39a...n. The service window 40a...n is an abstract class and, as such, certain behaviors of the service window 40a...n are predefined in the code. However, a developer is able to add features to, or extend the service window depending on the needs of the developer. For example, in the Document Mall embedded function 8a example, a Document Mall e-mail service window supports both e-mail address search using the Document Mall service, and manual e-mail address entry.

[0052] The service data 39a...n is updated by the service window 40a...n based on user operations. Further, the service data 39a...n is accessed by an activated service data handler 12 when upload operations are performed. For example, if activated, the sending of e-mails or uploading to network folders may be performed by the service data handler 12 as is done in the Document Mall embedded function example 8a. As with the service window 40a...n, the service data 39a...n is an abstract class which can be updated or extended by developers to create further service related data. For example, in the Document Mall embedded function 8a example, the Document Mall e-mail service sends an e-mail based on the e-mail destination address that is saved in the service data included in the service data 39a...n.

[0053] Each service 41 corresponding to an external function also includes an activation reading part 29a...n, however there is no locally stored service data and the service window 42 only acts as a display interface for sending information to the external server corresponding to the external service.

[0054] Thus, the unified client main thread 30 includes a project array 31 which lists several projects 33a...n which may or may not include embedded or external functions, each project including the service array 34a...n which lists several services 38a...n corresponding to functions which may or may not be activated. As discussed above different embodiments of the present invention handle the inclusion of activated and non-activated projects and services in the corresponding project or service array. In one embodiment, only activated projects and services are included in the corresponding project and service arrays. In another embodiment the un-activated projects and services are included in the corresponding project and service arrays, when a user attempts to utilize the functionality of the inactivated projects and services the user is given the ability to buy or activate the service. Further detail regarding this feature will be discussed below with regard to the activation manager. The projects 33a...n found in the project array are displayed on a project array window 32 and each project includes a main window 35, and optionally a login window which is displayed before the main window 35, the login window could alternatively be displayed simultaneously with the main window 35. Further, each service 38a...n includes a service window included in the service window 40a...n.

[0055] It should also be noted that multiple functions can be associated with a single project 33a...n. For example, if the eCabinet Email function and the eCabinet Folder are included in a single project 33a...n, users will see links to both eCabinet Email and eCabinet Folder service windows 40a...n in main window 35. If a user enters all necessary information in corresponding service windows 40a...n, one scan job can be delivered using both the eCabinet Folder and eCabinet Email operations.

[0056] Turning now to Figure 6b, this figure connects to Figure 6a by symbol connector A. Once a scan by the MFP is completed, upload data 50 is created. The upload data 50 includes a document name, scan data, login data and service data, for example. The upload data 50 can also include any other information that can be uploaded by a service data handler 54a...n to a reception device. The service data handler 54a...n includes an activation reading part 55a...n and performs upload of data from the MFP, each service data handler being related to a project 33a...n. An upload thread/job monitor 51 includes a job queue 53. The upload thread/job monitor 51 is a background process that monitors the job queue 53 and processes the jobs when they become available. The upload thread/job monitor 51 is connected to the service data handler 54a...n. When a scan completes, the main thread 30 posts its final upload data 50 and adds it to the job queue 53.

[0057] For each job, the upload thread/job monitor 51 groups upload data 50 based on the corresponding service data handler 54a...n and invokes the corresponding service data handler 54a...n to process the upload data 50. For example, in the Document Mall embedded function 8a example, the upload thread/job monitor 51 passes generic data such as scan or image file related information, login data e-mail service data and folder service data to the Document Mall service data handler 54a...n to be processed. Once the upload thread/job monitor 51 has completed the above steps, the final steps are to get a job upload status and update a job log. The job upload status is the status of the upload by the service data handler 54a...n and the job log is the list of jobs processed by the upload thread/job monitor 51.

[0058] As described above, the service data handler 54a...n performs the upload of the upload data 50. However, the service data handler 54a...n will only perform its function if activation is first confirmed by the activation reading part 55a...n. For example, in the Document Mall embedded function 8a example, the activation reading part 55a...n will access the config.xml 7 to confirm that the service data handler 54a...n is activated. If activation is confirmed, then the service data handler 54a...n receives generic data, login data e-mail service data such as e-mail destinations and folder service data such as folder destinations. Finally the service data handler 54a...n composes the received upload data 50 into an upload.xml file and uploads the xml file to a Document Mall server designated in the config.xml file 7 via a http post process. Finally the service data handler 54a...n reports the upload status to the job monitor for job logging.

[0059] Any processes descriptions or blocks in flowcharts should be understood as representing modules, segments, portions of code which include one or more executable instructions for implementing specific logical functions or steps in the process, and alternate implementations are included within the scope of the exemplary embodiment of the present invention in which functions may be executed out of order from that shown or discussed, including substantially concurrently or in reverse order, depending upon the functionality involved, as would be understood by those skilled in the art.

[0060] Figures 7A-7F show a flowchart of the unified client main thread 30. After starting in Figure 7A, the unified client application 5 is initialized in step 60. The unified client application 5 is initialized by first initializing the core application 6. Next in step 61, the external functions are determined and the config.xml file 7 is updated. Further detail regarding the external functions determining process is found in Figure 7F. In step 62, the installed embedded functions are determined and the config.xml file 7 is updated. It should be noted that the config.xml is updated each time a new function is installed on the MFP

[0061] Next in step 63, the activation manager 6b determines activation status of each external and internal function found in the config.xml file located on the MFP,

further detail of this process is found in Figure 7E.

[0062] In step 64, priority is determined between two conflicting external and embedded functions. The process for determining priority is based on information received by the activation manager.

[0063] The flow then moves onto step 65 where the config.xml file 7 is read. The config.xml file 7 includes settings for the core application 6 and for the functions which are operable on the MFP via the host or core application 6. A project array 31 is then constructed in step 66 based on the functions determined in steps 61 and 62 above. Next in step 67, the service array 34a...n is constructed for each project 33a...n. Further a main window 35 is constructed in step 68. Figure 10, discussed in more detail below, shows an example of the main window 35. Flow then proceeds to process B in Figure 7B.

[0064] In Figure 7B, the project array window 32 is created in step 70 using the project array. The project array window 32 will only display projects that are activated or are available for activation by the user utilizing the MFP. Thus if the user utilizing the MFP does not have the authority to activate new projects, then only the previously activated projects will be available for selection. The project array window 32 is then displayed in step 71 and in step 72 a project is selected based on manual user input.

[0065] Once the project is selected by step 73, the flow then proceeds to step 73 where the user selection is stored in a log file. It should be noted that the present invention allows all user access and transactions within the MFP to be tracked and recorded. The log file can then be transmitted over the network to a central repository for billing uses, audit reporting or other purposes. It should also be noted that the log file stores use access and use information to be stored for both embedded and external functions accessed via the MFP.

[0066] When step 73 is accomplished, the system flow proceeds to step 74 where the selected project is initialized. From step 74 of Figure 7b, flow proceeds to process C of Figure 7C.

[0067] Turning now to Figure 7C, step 80 determines if the initialized project includes a login window 37. If no login window 37 is installed, flow proceeds to process E of Figure 7D. If the project includes a login window 37, then the flow proceeds to step 81 where it is determined if the login is activated. If the login is not activated, flow proceeds to process E of Figure 7D. If the login is activated then flow proceeds to step 82 where both the login window class and the login data class are loaded.

[0068] Once the class files have been loaded, the login window is displayed in step 83. The login window includes both a login button and a cancel button. Depending on which button is determined to have been pressed in step 84, the flow proceeds differently- When the login button is pressed, the flow proceeds to step 85 in which the process login function of the login window 37 is called. However, if the cancel button is pressed in step 84, the flow proceeds to process D in Figure 7B. Process D returns the flow to step 70.

[0069] If the login button is pressed in step 84 the login function of the login window 37 is called in step 85. Step 86 checks to see if the login was successful. If the login was not successful, the flow proceeds to step 87 to reset the login window and then returns to step 83. If the login was successful, then the flow proceeds to process E in Figure 7D.

[0070] Thus Figure 7C includes the general procedure of completing authentication if the login window 37 is installed in the selected project 33a...n. If no login window 37 is installed or the window 37 is not activated, then the entire login process is skipped.

[0071] Turning now to Figure 7D(i), in step 90, service data for each service is loaded. Once the service data is loaded, the flow proceeds to step 91 where the post-login function of each service is called. In step 92, the logout listener is set in the main window 35. Additionally, in step 93 a service button for each service 38a...n is created in the main window 35. The service window class for each service is then loaded in step 94 and the upload data 50 is initialized or created in step 95.

[0072] Once the upload data 50 is initialized in step 95, the main window 35 is displayed to the user in step 96. The default service is then selected in step 97. It should be noted that the default service will always be an activated service. Then the service window corresponding to the selected service is displayed in step 98. In step 99 service data is input in the service window displayed in step 98. The flow then proceeds to step 100 in Figure 7D(ii) which checks if the auto-logout time has expired. The auto-logout feature forces the flow to proceed to the logout step 104 if no user activity is detected for a predetermined period of time. If the auto-logout time is determined not to have expired in step 100, the flow proceeds to step 101 which determines if a button was pressed. If a button was pressed, the flow proceeds to step 102, if not the flow returns to step 100. Step 102 determined which button was pressed. If one of the service buttons was pressed, then the flow proceeds to step 103 where the selected service is set. The flow then returns to step 98 in Figure 7D(i) where the newly selected service window is displayed. If in step 102 the logout button is pressed the flow proceeds to step 104 where each service is reset, the main window 35 is reset and the upload data is reset.

[0073] If the MFP "start button" is pressed by the user in step 102 the flow proceeds to step 105. In step 105 the scan is completed. The flow then proceeds to step 106 where the upload data 50 is copied and added to the job queue 53.

[0074] Figure 7E shows a more detailed description of the activation manager 6a workflow. Specifically, Figure 7E shows details of the process of step 63 shown in Figure 7A. Step 63 is made up of five steps 63A-E. In step 63A the activation manager 6a starts up and reads the previously installed config.xml file 7. Included in the config.xml file 7 is MFP information, further discussion of the

contents of the config.xml file 7 can be found below with regard to Figures 11A-11E.

[0075] The activation manager then contacts an Activation Database over a network in step 63B and sends information regarding the MFP to the Activation Database to verify the MFP and account information in step 63C. In step 63D, the activation manager 6a retrieves activation information and priority information regarding the priority between conflicting external and embedded functions from the Activation Database based on the sent MFP information. The activation manager 6A then updates the activation information in the config.xml file 7 based on the received information in 63E.

[0076] In the present embodiment, the activation manager updates the config.xml file 7 by contacting an activation database. In an alternate embodiment, the activation information could be retrieved from another MFP in the network that had contacted the activation database at a previous time.

[0077] Figures 7F(i), 7F(ii) and 7F(iii) show a more detailed description of the external functions determination workflow. Specifically, Figures 7F(i), 7F(ii) and 7F(iii) show three embodiments of the internal process of step 61 shown in Figure 7A.

[0078] The first embodiment shown in Figure 7F(i) comprises steps 600-602. In step 600, the flow reads the config.xml to identify any services that are indicated as corresponding to external functions. Then in step 601, using the information in the config.xml file 7 the availability of the external functions are confirmed. For example, if several globalScan functions were previously stored in the config.xml file 7 such as globalscan email or globalscan fax, the flow would then confirm that the globalscan server still exists and that the functions previously included in the config.xml file are still available. In step 602 the flow then updates the config.xml file 7 and indicates if the functions are not available.

[0079] Embodiments two and three relate to the situation in which the external functions available to the MFP are not known and are not previously stored in the config.xml of the MFP. Thus, the MFP has the ability to discover external functions based user input using a search mechanism or automatically as is described in embodiments two and three.

[0080] The second embodiment shown in Figure 7F(ii) comprises steps 603 and 604. In step 603 the network is scanned using a discovery mechanism and available external functions are discovered. For example, referring back to Figure 1, the branch office MFP 500 could discover the globalscan server 504 and connect to the server 504 and discover a number of services that are available on globalscan server 504.

[0081] Once available external functions are discovered in step 603 the config.xml file 7 is updated in step 604.

[0082] The third embodiment shown in Figure 7F(iii) comprises steps 605-608. The third embodiment differs from the second embodiment in that the address of the server including the external functions is already known by the MFP. Thus, in step 605 the MFP connects to the external function server. In step 606 the MFP uploads MFP and account information to the external function server. This allows the external function server to confirm that the MFP has the authority to use the external functions provided by the server. In step 607 a list of available services is downloaded by the MFP and is used in step 608 to update the config.xml with available external functions.

[0083] However, it should be noted that even though in step 606 the MFP uploads information to the external function server the unified client platform uses a uniform security policy that allows the user to be authenticated for both embedded and external functions at the same time. Using the activation manager 6A this step is accomplished because although a function may be available to be accessed via the MFP the user may not have access to this function, thus the function is de-activated even if it is included in the config.xml file 7.

[0084] Turning now to Figures 8A and 8B, Figures 8A and 8B show a flowchart of the unified client upload thread 51. After starting, a job monitor initialization is performed in step 120. The system then checks if any jobs are in the job queue 53 in step 121. If no jobs are determined to exist in the job queue 53, flow proceeds back to the beginning of step 121. The system continues in this for a loop until a job is observed in the job queue 53.

[0085] When a job is determined to exist in the job queue 53 in step 121, flow proceeds to step 122 and gets the job from the job queue 53 and groups services included in the job based on the corresponding service data handler 54a...n. Next, the generic login data and corresponding service data is passed to the service data handler 54a...n in step 123. In step 124 it is determined if the service data handler 54a...n is activated. If the service data handler is not activated flow proceeds to step 128 where the job is not processed for the service data handler 54a...n. The flow then proceeds to step 129 where the job upload status is sent to the job monitor. Flow then proceeds to step 127.

[0086] If however in step 124 the service data handler 54a...n is activated, the service data handler 54a...n then processes the job upload data 50 in step 125.

[0087] Once the service data handler 54a...n has processed the job upload data 50, the job monitor 51 gets the job upload status from the service data handler 54a...n and updates the job log in step 126. Flow then proceeds to step 127 which checks to see if there are any more service data handlers 54a...n. If no service data handlers 54a...n remain for the job, then flow moves back to step 121 to check for new jobs in the queue. If more service data handlers 54a...n are included in 127, flow proceeds back to step 123 and processes steps 123-126 again. This loop continues until no service data handlers 54a...n remain for a job.

[0088] Thus the unified client upload thread includes two loops, the first checks for new jobs in the queue. The

second loop occurs once a job is determined to exist, in the second loop, the system loops through checks to make sure all of the activated service data handlers 54a...n in a job have been processed.

[0089] Moving now to Figure 9, there is shown an example of a project array window 32. The project array window 32 includes a line reserved for system messages 151. Also included is a unified client logo 152 and instructions to the user on how to use the project array window 153. The project array window 153 also includes several project buttons that are selectable by the user 154 and link the user to the main window 35 and default service window of the selected project 33a...n. Examples of such buttons are the Document Mall button 154, eCabinet button 154A, Email button 154B, Fax document button 154C, scan to folder 154D or other similar type projects buttons 154n. The scroll bar 155 allows a number of project buttons to be installed in the project array window 32. Thus the function of the project array window 35 is to allow a user to select which project 33a...n the user may like to use on the MFP.

[0090] Figure 10 shows an example of a main window 35. The main window 35 includes the unified client logo 161 as well as the document name 162 and a logout button 163. As discussed earlier with respect to Figure 7d, the logout button 163 allows the user to log out of the selected project and return to the project array window 32 described in Figure 9. The main window 35 also includes a number of buttons 156 through 159 which correspond to a number of services or alternatively one service if only one function is associated with the project 33a...n. The buttons displayed in the main window 35 correspond to the project 33a...n which was selected in the project array window 32. For example, when the Document Mall project is selected 154 in the project array window 35, several Document Mall related buttons are available. For example, the button 156 allows the user to open a scan to a Document Mall e-mail service window. Item 157 allows the user to open a scan to a folder service window. Item 158 is a button that open up the scan settings service window. While item 159 allows the user to open up the job log service window. The invention is not limited to the number of buttons included in Figure 10 or the services shown in Figure 10. Additionally arrow buttons 160a and 160b allow the user scroll through a number of service buttons. Thus, any type of service button can be installed on the main window 35.

[0091] Figures 11A-11E show an example a config.xml file 7 according to one embodiment of the present invention. It should be noted that Figures 11A-11E are not intended to be a comprehensive example of how a config.xml file 7 may be designed, instead Figures 11A-11E include one way that a config.xml file 7 might be written for activation of a unified client application 5.

[0092] Figure 11a begins the example of the config.xml file 7. In line 1 of Figure 11A, the config.xml file 7 begins with the root tag. Under the root tag are the jar file list tags which include different jar files that are installed on the system. A jar file is a single file that includes several class files. The class files each include portions of code that, in the present example, correspond to different services 38a...n. In the present embodiment, the jar files listed in the config.xml file 7 correspond to embedded functions that are installed in the unified client application 5. In this example, the eCabinet jar file, the Document Mall jar file and the EmbeddedEmail.jar file are installed. The eCabinet jar file corresponds to the eCabinet embedded function 8b and the Document Mall jar file corresponds to the Document Mal embedded function 8a. The EmbeddedEmail jar file corresponds to the embedded email function. Each embedded function is included in a project 33a...n.

[0093] The MFP section starting on line 7 includes several tags which relate the MFP and the account associated with the MFP. On line 8 is found the MFPSerialNo tag which includes the MFP serial number. The MFP serial number is a unique number which identifies the hardware of the MFP. On line 9 is found the MACAddress tag which includes the MFP MAC Address. The MAC address is a unique network identification code that identifies the network interface of the MFP. On line 10 is found the AccountName tag which includes the account name to which the MFP is registered. In the present example the account name is Ricoh.

[0094] On line 11 the UserName tag is found which includes the username of the user currently logged into the MFP. It should be noted that in alternate embodiments no UserName tag is used. In addition, the ModelName tag not shown in Figure 11A can be included in the MFP section. The ModelName tag identifies the model name of the MFP. Finally on line 12 the close MFP tag is found which identifies the end of the MFP section.

[0095] As noted above with regard to Figure 7E, the portion of the config.xml file 7 enclosed in the MFP tags is part of the MFP information sent to the activation database. The MFP information is then used by the activation database to determine which services are activated. Thus the data in the MFPSerialNo, MACAddress and ServiceName tags can be used as a unique key.

[0096] In line 13, the project tag begins the project 33a...n and in line 14 the project name is designated, in this example the project name is described as eCabinet. The default scan setting, included in line 15, is empty in this example but could include a number of different settings. In line 16 the default resolution setting tag is included, in this example, the default resolution is set to 200, which corresponds to 200 dpi. In line 17 the default double side scan setting is included. In this example the default double side scan setting is set to false. This setting allows the user to set if the multi-function printer will scan both sides of the paper instead of just a single side.

[0097] In line 19 the login setting for the project is included. In this example, the eCabinet project does not have a login. However, this setting could also be set to true. In one embodiment of the config.xml file 7, if in line 19 the login setting is set to true, in line 20, a login class may be included. Other embodiments may not include

the class file in this manner. In this example, no login class is included because the login setting in line 19 is set to false.

[0098] As discussed earlier, each project 33a...n includes a number of services 38a...n. In line 21, the service tag begins the section describing a service 38a...n. In line 22, the service's name is included and in line 23 the display name is also included. In this example, the service name is set to eCabinet and the display name is set to eCabinet Owner. The display name setting shows how the service is displayed in the service buttons in the main window 35.

[0099] Line 24 shows the Activation open tag. This tag begins the activation section of the service. Included in the activation section are several tags related to the activation of the service. The example shown in Figure 11A is one way of including activation information in the config.xml 7 file, other ways are also possible. On line 25 the ActivationRequired tag is found. This tag includes a boolean indicator which denotes whether or not activation is required for the service in question. In the present example, shown on line 25, the ActivationRequired tag is listed as "Y", however if the tag included a "N" or "F" indicator the service would always be available to be used on the MFP. The default value for this tag is "Y".

[0100] The next tag in the activation tag section is the Activated tag which is found on line 26. As with the ActivationRequired tag noted above, the Activation tag includes a Boolean indicator. The Boolean indicator corresponds to whether or not the service in question is activated. If the indicator shows "N" or "F" then the service will not be available to the user of the MFP unless the user goes through an activation process. In the present example the Activated tag includes a "Y" indicator. When a "Y" indicator is found in the Activated tag, the ActivationDate and ExpirationDate tags found on lines 27 and 28 list the date that the service was activated and the expiration date of the activation, respectively. The ExpirationDate tag is useful in the case that the Activation Database is unable to be contacted. If the Activation Database is unreachable, the activation manager can compare the internal date stamp of the MFP with the information found in the ExpirationDate tag of the config.xml file 7 to ensure that the activation is still valid. Finally, the Activation close tag is found on line 29.

[0101] In should be noted that several different tags may be used in the Activation Section depending on the type of activation used. In the present example, time-based activation is used, however, when different types of activation are used different tags may be used in the activation section.

[0102] For each project a data handler is included and for each service corresponding to an embedded function a service window class file is included. In this example, in lines 30-32 the service window class file is listed. The service window class file includes all the code necessary to display the service window. In lines 33-35 the data handler class file is listed. This includes all the code necessary for the data handler in this project.

[0103] Beginning on line 1 of Figure 11B, the configuration data for this service is included. In this example, on line 29 the eCabinet server address is listed as 11.11.11.111. The address 11.11.11.111 is an example of an address that may be used other addresses including IPv6 addresses or named addresses, such as domain names, can also be used. Further, the eCabinet server port is listed as port 81. Other port numbers could also be used in this example.

[0104] Beginning on line 6, the data handler configuration data is included. In this example, the data handler configuration data includes the eCabinet server address on lines 8-9 and the FTP port on line 10. If no FTP port is included in line 10 a default ftp port is used, such as 21. On line 11, the data handler configuration data tag is closed and on line 12 the service is closed. Thus the eCabinet service configuration in this example is described between lines 21 of Figure 11A and line 12 of Figure 11B.

[0105] On line 13 a second service included underneath the eCabinet project is described. Beginning on line 13, the service tag opens the service. The service name in this example is included on line 14 and is listed as eCabinet Folder. The display name included on line 15 is listed as eCabinet Folder.

[0106] Lines 16-21 show the Activation section for the eCabinetFolder service. As was described with regard to the eCabinet service above, the Activation section of the eCabinetFolder service includes open and close Activation tags, an ActivationRequired tag, an Activated tag, an ActivationDate tag and an ExpirationDate tag.

[0107] Additionally, as was the case in the eCabinet service described above, the eCabinet Folder service also includes a service window class shown on lines 22-24 and a data handler class included in lines 25-27. Also included in the eCabinet folder service are the eCabinet server address, on line 29, and the eCabinet server port included on line 31. Also in this example, the data handler configuration data tag area begins on line 33 and includes an eCabinet server address, on line 34, and a FTP port setting on line 1 of Figure 11C. The Data handler configuration data tag area is closed on line 2, the service is closed on line 3 and the project is closed on line 4. Thus in this example, the eCabinet project includes two services, the eCabinet service and the eCabinet Folder service.

[0108] Line 6 continues the example of the config.xml file 7. On line 6 of Figure 11C, a new project is opened with a project tag. The project name is described on line 7 as Document Mall. In this example, in line 8, the default scan setting tag is opened and closed denoting the default setting and, in line 9, the default resolution is set to 200. In line 10, the default double siding scan is set to true and in line 12 the login setting is set to true. As was noted in the discussion regarding eCabinet project above, because the login setting is set to true in line 12, in lines 13 and 14 the login class file is included. In con-

trast to the eCabinet project described above, in the Document Mall project example, the login class is included.

[0109] Beginning on line 15, the service tags included under the Document Mall project are described. The first service begins with a service tag, included on line 15. In line 16, the service name DMEmail is included and on line 17 the display name DM Email is also included.

[0110] Lines 18-23 show the Activation section for the DocumentMall Email service. As was described with regard to the other services described above, the Activation section of the DocumentMall Email service includes open and close Activation tags, an ActivationRequired tag, an Activated tag, an ActivationDate tag and an ExpirationDate tag.

[0111] The service window class is described at lines 24-26 and the data handler class is described in lines 27-29.

[0112] Beginning on line 30, the configuration data for the Document Mall email service is included. In this example, in line 31, the Document Mall server address is included as documentmall.com and in line 32 the configuration data tag is closed. In line 33 the data handler configuration data tag is opened. Within this tag, in lines 34-35, the Document Mall server address is included and on line 1 of Figure 11D the data handler configuration data tag is closed. In line 2 the Document Mall Email service is closed.

[0113] Line 3 begins a second service under the Document Mall project with the service tag. On line 4 the service name of DMFolder is included and in line 5 the display name DM folder is also included. In lines 6-12 the activation portion of the service is included. In lines 13-15 the service window class is described and in lines 16-18 the data handler class is described. In line 19, the configuration data begins with the configuration data tag. In line 20, the Document Mall server address is included. In line 21 the configuration data is closed. Lines 22-24 show the data handler configuration data, with line 23 showing the Document Mall server address. In line 25, the close service tag closes the service. In line 26, the close project tag closes the Document Mall project.

[0114] Line 28 includes a project tag which begins a new project. Line 29 includes the project name which is Email. The default scan setting, included in line 30, is empty in this example. In line 31 the default resolution setting tag is included, in this example, the default resolution is set to 200, which corresponds to 200 dpi. In lines 32-33 the default double side scan setting is included. In this example, the default double side scan setting is set to false.

[0115] In lines 34-35 the login setting for the project is included. The haslogin setting is false and the loginclass tag is empty. The Email project shown in the present example has two services, the embedded email service found in lines 1-25 of Figure 11E and the GlobalScanEmail service found in lines 26 to line 6 of Figure 11F. Both services include an external tag section which is found in lines 10-14 for the embedded email and line 35 to line 5 of Figure 11F for the GlobalScanEmail.

[0116] The external tag section includes Embedded, ExternalAddress and Priority tags. The Embedded tag is a Boolean value that describes how the service is performed using the MFP. If the Embedded tag is set to true, then the service corresponds to a function that is embedded or physically installed on the MFP. If the Embedded tag is set to F, then the service corresponds to an external function or a function in which the function is performed on an external server. For example, for the Email function described in lines 26 to line 6 of Figure 11F, an image is scanned on the MFP locally but the email itself is created and the scanned image is converted into PDF on an external server. Thus, the function of creating the Email is performed externally from the MFP.

[0117] The ExternalAddress tag is empty when the Embedded tag is set to true. However, when the Embedded tag is set to F, then the ExternalAddress tag includes the network address of the server on which the external function corresponding to the service is located.

[0118] The Priority tag is used to determine the priority for two services in the same project which are conflicting. In the present example, the embedded email function and GlobalScanEmail function are conflicting because both services perform the same function in the same project. Thus, the External tag section includes a tag that allows the MFP to know which service has priority over the other or whether the user will be using the embedded or external email function by default when the Email function is selected. As described earlier, if the external function is determined to have priority, but the external function server is unavailable, the embedded function can be enabled as a failsafe. The user will thus suffer no disruption in MFP functionality even though the external service is unreachable.

[0119] The Email project is then closed on line 7 of Figure 11F and in line 8, the close root tag closes the config.xml file 7.

[0120] A functional example of the unified client application 5 installed on a multi-function printer will now be described in Figures 12-17. In Figures 12-16 of this example, the unified client application 5 is installed with the eCabinet embedded function 8b. In Figure 17 the unified client application 5 also includes the Email external/embedded function. The unified client application 5 with an eCabinet embedded function 8b is developed using SDK/J and uses the CVM option on each MFP in which the unified client application 5 is installed. SDK/J is an embedded software architecture software development kit ("SDK") which allows in house developers, independent software vendors and system integrators to deliver customized JAVA based solutions on MFPs. The CVM option is the java virtual machine that is able to be installed on the MFP. Other types of virtual machines and/or programming languages can be used to create embedded functions associated with the unified client application.

[0121] The example of the unified client application 5

with the eCabinet embedded function 8b uses 2 SDK/J type applications, the 2 SDK/J applications are, for example, one Java xlet application which implements major unified client functionalities and one servlet application which allows user to update the config.xml file 7 remotely via a web browser. Some of the services or functions that are supported by the unified client application 5 with the eCabinet embedded function 8b are: scan to eCabinet server, scan to eCabinet folder, scan settings and job log viewing. These services are represented as service buttons in the eCabinet project main window 35.

[0122] In Figure 12, an example of a main window 35 and service window 173 is shown. The main window 35 and the eCabinet owner service window 173 are displayed. The main window 35 includes a logo 161 as well as the document name 162 and an end session or logout button 163. Further, several service buttons 164-167 are also included in the main window 35. In the eCabinet example, the first service button is the eCabinet owner button 164. In figure 12, this button is selected and as a result the corresponding eCabinet owner service window 173 is displayed. The left side of the eCabinet owner service window includes an order list window 168 and a refresh button 169. Further, on the left side of the eCabinet owner service window 173 there is a selected owner's window 170. Also included are a public 171 and reset button 172. The eCabinet owner button offers users the ability to select the eCabinet owner for use with the eCabinet Folder service and eCabinet Email service (not shown). The owner list is downloaded from the eCabinet server automatically and is displayed in the owner list window 168. Multiple owners can be selected if no eCabinet folder is selected in the eCabinet folder window 193. When an eCabinet folder is selected in eCabinet folder window 193, only a single owner selection is allowed. The owner list window 168 shows a list of the owners. The selected window 170 shows the destination owners. To add a destination owner, the user can highlight desired owners in the owner list window 168 and press the right arrow button 175. To delete a destination owner, the user can highlight the owner in the selected window 170 and press the left arrow 176. The refresh button 169, allows the user to download the owner list from the server again. The public button 171 allows the user to set the attribute of the scan document to public or private. The reset button 172 allows the user to remove all of the contents of the selected window.

[0123] Figure 13 shows an example of the main window 35 with the eCabinet folder service button 165 selected and the eCabinet folder service window 193 displayed. In this example, the eCabinet folder button 165 has been selected and as a result the eCabinet folder service window 193 is displayed. The eCabinet folder service window 193 includes a folder list window 189, a refresh button 190, a selected window 191, and a reset button 192. The eCabinet folder service offers users the ability to scan to the eCabinet folder service. The eCabinet folder list is downloaded from the eCabinet server automatically using the configuration settings included in the config.xml file 7. When a user selects the eCabinet folder button 165, the unified client application 5 prompts the user with a software keyboard allowing the user to enter a user name and a password. The unified client application 5 then downloads the user's folder tree and displays the tree in the folder list window. Note that using the eCabinet folder service requires single owner selection. If multiple owners have been selected in eCabinet owner service window 173 and the user presses eCabinet folder button 165, an error message will pop up stating eCabinet folder service requires single owner selection. The folder list window 189 shows a user's eCabinet folder tree. The user can browse the folder tree in the folder list window 189. To add a destination folder, the user can highlight the desired folder in the folder list window 189 and press the right arrow button 175. To delete a destination folder in the selected window 191, the user can highlight the desired folder in the selected window 191 and press the left arrow button 176. It should also be noted that multiple folders can be selected. The refresh button 190 allows the user to download the eCabinet folder list again from the eCabinet server. If the refresh button 190 is pressed, the user will be prompted for the user name and password entry again. The reset button 192 allows all the contents placed in the selected window 191 to be removed. It should also be noted that the eCabinet folder list, included in the folder list window 189, is dependent upon the owner selected in the eCabinet owner service window 173 included in Figure 12. The user that is selected and included in the selected window 170, is the user who corresponds to the folder list included in the folder list window 189.

[0124] Figure 14 shows the user interface for an example of when the scan settings button 166 is selected. When the scan settings 166 button is selected, the scan settings service window is displayed 218. The scan setting service window includes several options including resolution 209, original 211, image type 214 and file format 215. Under the resolution option 209, several different buttons relating to scanner resolution are used. In this example, DPI 200, 210a, 300 DPI, 210b, 400 DPI, 210c, or 600 DPI, 210n, are available to be selected. Other similar types of DPI options resolution options could also be used. The original option 211, includes two buttons. The first button 212a allows the one sided option to be selected. The second button 212n allows the two sided option to be selected. Further, a batch scan button is displayed 213. The image type option 214 also includes a drop-down menu listing a number of image types. In the current example, the text option is displayed. It should also be noted that in the image type drop-down box text, photo, gray scale or photo options are available. Similarly, under the file format option 215, a second drop-down box is included listing a number of different file formats. In the present example, the PDF option is displayed. However, in the file format drop-down box single page tiff, multi-page tiff, jpeg and PDF options are available.

Also included on the scan setting service window 218 is a scan size 216 button and a reset button 217.

[0125] The scan size button 216 opens a new window which is shown in Figure 15. The scan size window 219 is still part of the scan sittings service window 218. However, the scan size window 219 is displayed in place of the scan setting service window 218 under the main window 35. In the scan size window 219, several different options are available. For example, auto detect 239, 8x11 5-1/2 x 8-1/2 A5, 240a, 8-1/2x11 5-1/2x8-1/2 A5 240b, 11x17 a3, B4 JIS 270c, 8-1/2x13 A4 B5 JIS 240d, and 8-1/2x14 A4 B5 JIS 240n. Also included are a reset button 242 and a general button 241 which returns the user to the original scan settings service window 218.

[0126] Figure 16 shows the main window 35 and the job log service window 264 displayed when the job log button 167 is selected. In the job log service window 264, date and time 259, document name 260, pages 261 and status 262 titles are displayed. From the job log service window 264 users can check scan job upload status specifically through the date and time, the document name, number of pages and the status of the job. This concludes the MFP display example of the eCabinet Embedded Function 8b.

[0127] Figure 17 shows the default service window 300 and main window 35 for the email project corresponding to the email function of the MFP. When a user selects the Email project in the project array window 32, the window shown in figure 17 is shown. The window shown is the same regardless if the embedded email function or the embedded email function has priority. As a result, the use of external functions by the MFP is transparent to the user.

[0128] The main window 35 of the email project includes the Email service button 301, the ScanSetting service button 166, and the JobLog button 167. The ScanSetting service button 166 and the JobLog button 167 correspond to ScanSetting and JobLog service windows which are very similar to the windows described above with respect to the eCabinet embedded function. Thus these service windows are not illustrated in the present example.

[0129] The email service window 300 includes a subject button 302 which when selected allows the user to enter an email subject in subject field 306. When the subject button 302 is selected, a keyboard is displayed that allows a user to enter in the subject (not shown). The recipient field 307 includes the names of the recipients of the email. Search button 303 enables the names of the addresses of the email recipients to be searched using an email address database such as an LDAP database. Alternatively, the manual entry button 304 allows the address of the recipient to be manually entered. Clear Form button 305 clears all of the addresses from the recipient field 307. Once the user has finished adding the addresses to the recipient field 307, the addresses can be moved to the "To" field 309, the "CC" field 310, the "Bcc" field 311 or the "Reply to" field 312 using the arrow buttons 308. The reset button 313 clears the "To", "Cc", "Bcc" and "Reply to" fields. Also included in the email service window 300 are doc name 162 and end session 163 buttons, these buttons are discussed above with respect to the eCabinet embedded function.

[0130] Fig.18 shows an example of a hardware configuration of the MFP 499 according to an embodiment of the present invention. As shown in Fig.18, the MFP 499 includes a controller board 400, an operation panel 410, a fax control unit (FCU) 420, a USB 430, an IEEE 1394 port 440, and a printer 450. It should be also noted that other types of i/o interfaces could be included including IEEE 1394b, USB 2.0. The controller board 400 includes a CPU 402 for processing and several storage devices such as SDRAM 403, SRAM 408, flash memory (flash ROM) 404, flash card interface part 406 and HD 405 used to store data associated with the MFP 499. Each of these components are connected to the ASIC 401, the ASIC 401. is an application specific integrated circuit that is designed specifically for use in a MFP 499. Other types of storage devices are also possible as well as other types of data processors and integrated circuits. The operation panel 410 is directly connected to the ASIC 401 as is the communications interface 420. The communications interface 420 can also be connected to a network or any other similar type communications medium. The USB 430, the IEEE 1394 440 and the multifunction printer functions 450 such as scanning, printing, and faxing are connected to the ASIC 401 via the PCI bus 480.

[0131] The SRAM 408 is a nonvolatile RAM, other types of SRAM are also possible. A flashcard 407 can be inserted into a flash card interface part 406, so that data is sent/received between the ASIC 401 and the flashcard 407 via the flash card interface part 406.

[0132] The operation panel 410 includes an operation part used for key operation such as key input and button pushing and the like by the user, and a display part for displaying drawing data such as various screens. It should be appreciated that other types of hardware components can be used in the present invention.

[0133] Further with respect to a computer readable medium such as a floppy disk, magnetic tape, CD-ROM and the like, by installing the program stored in the computer readable medium into an MFP, the MFP can perform the functions of the present invention.

[0134] This invention has been described with respect to a multifunction printer, but is applicable to any image handling device such as a copier, digital copier, printer, scanner, digital camera, fax machine, or multi-function printer or any combination thereof. A general purpose computer is not considered an image handling device. Moreover, the invention is applicable to other special purpose devices such as navigation systems, global positioning systems, vending machines, metering systems, machine tools and other tools which operate using programming or a programmed processor, automobiles, other transportation devices such as trains, motorcycles,

planes, or boats, radar systems, radios, MP3 players, digital music players, and other audio systems, mobile phones, other communication devices and systems, and any other special purpose device which operates using a plug-in.

[0135] The present invention is not limited to the specifically disclosed embodiments, and variations and modifications may be made without departing from the scope of the present invention.

[System and Method to Allow the Removal and Addition of Functions on a Multi-Function Printer]

[0136] The present advancements relate, in part, to a method of executing applications installed on an image handling device. The image handling device is a MFP. The MFP is any printer or copier which includes multiple functions such as scanning, printing and/or faxing. Additionally, the MFP described above may include a copier that scans and prints a document in a single step as scanning and printing are distinct functions.

[0137] The present advancements also are related to a host application of the image handling device and a configuration file of the image handling device. The host application may be executable code that interfaces with the operating system of the MFP and accesses the hardware of the MFP through the operating system.

[0138] The configuration file includes activation information corresponding to at least one plug-in and a set of services corresponding to the at least one plug-in. In other words, the configuration file includes the information that determines which plug-in can be run by on the MFP. Even if the physical application is found on a memory or hard drive of the MFP, unless the configuration file allows activation, the application will not be executed.

[0139] The configuration file may be any type of configuration file including an extensible markup language such as XML, Standard Generalized Markup Language (SGML), GML, RDF/XML, RSS, Atom, MathML, XHTML, SVG, DSDL, XUL, MXML, EAD or Klip.

[0140] It should also be noted that the configuration file is able to be used in a mixed brand environment. Even if, for example, several different brands of copiers are used in an environment such as an office or a building, each unique brand will be able utilize the configuration file. In addition each different MFP may be able to load the unified client architecture and the plug-ins. Thus, each copier or multi-function device will be able to have the same basic interface and commands limited only be the functionality of the specific copier or multi-function printer in question.

[0141] The present advancements also are related to launching at least one plug-in based on the activation information in the configuration file. Each plug-in includes a number of services that increase the functionality of the MFP.

[0142] The method also includes presenting a graphical interface that includes a graphical indicia of each activated plug-in and each activated service of the set of services corresponding to the activated plug-in. The graphical indicia can be links or pages relating to the activated plug-ins and corresponding activated services. Examples of the graphical interface are discussed with reference to Figures 30-37 below.

[0143] Referring now to Figures 19-40 of the drawings wherein like reference numbers designate identical or corresponding parts throughout the several views and more particularly to Figure 20 thereof, there is illustrated an application layer 1 including a unified client application 5. The unified client application 5 installed on a MFP includes a core application 6, the core application being an application that includes primary routines that serve the application. These primary routines typically carry out basic functions of the MFP including scanning, printing, copying, faxing, and communicating, for example. Included below the core application 6 is the activation manager 6b. The activation manager is the portion of the unified client application that determines the activation status of plug-ins and corresponding services. In addition, the activation manager 6b generates a config.xml file 7 based on the determination of the activation status.

[0144] The activation manager 6b provides the ability to activate or de-activate any plug-in installed on the MFP. Thus, all available functionalities can be pre-installed on the MFP at the factory. When a user takes delivery of the MFP, the MFP may have some of the plug-ins activated or none of the plug-ins activated. If none of the plug-ins are activated, the user can activate the plug-ins as the user sees fit, such as when the need arises.

[0145] Different types of activation schemes are also available. For example, the user may be able to activate a plug-in for a limited time on a trial basis. Alternatively, the user may be able to buy, lease, or license the use of the plug-in for a one time use or a time-based use such as for a week or a month. This could be useful for organizations that have higher demand during certain times of the year such as during tax season.

[0146] Additionally, the activation manager enables different fee schemes to be used on the MFP. For example, a user may pay a monthly subscription fee for use of a plug-in. When the user no longer needs the plug-in, the use can be discontinued via a central server using the activation manager 6b. Additionally, plug-ins can be de-activated based on expiration dates. For example, a user could buy a six month subscription to a plug-in, once the six months have expired the plug-in can be deactivated.

[0147] Further, if the user has the ability to activate plug-ins (such as the user has financial decision making abilities) the activation manager 6b enables the user to activate plug-ins in different ways. For example, using a device attached to the MFP, a users ability to buy activation could be verified. Devices such as a smart card, biometrics, PIN code, magnetic strip card or proximity card could be used as well as other existing ID verification systems to enable the purchase/activation.

[0148] With respect to the activation, in the event that a user has control over a number of MFPs, the activation process can be accomplished remotely and collectively. Thus, a large number of MFPs can have plug-ins activated virtually simultaneously using a remote station. This enables uniformity in an organization and also saves a significant amount of time as there is no need to visit each MFP to perform an activation or deactivation.

[0149] The config.xml file 7 includes settings regarding unified client application 5 in addition to activation information. Additionally, plug-ins 8a...n are controlled by the core application 6.

[0150] Different types of plug-ins can be installed in the unified client application 6. For example, in the present example depicted in Figure 20, a Document Mall plug-in 8a, an eCabinet plug-in 8b and a generic plug-in 8n are installed.

[0151] A plug-in generally includes programs or code for operating the hardware of the multi-function printer via the core application 6. An alternate illustration of the Unified Client application 5 shown in Figure 20 is set forth in Figure 21 which includes the core application 6, the activation manager 6b which is separate from the core application, the config.xml file 7 including activation information and the plug-ins 8, the plug-ins including an activation reading part 8b.

[0152] Figure 22 shows an example of an internal structure of a plug-in 8. For example, the plug-in 8 may include a single service window 10a or may also include a number of service windows such as 10a...10n. The service window 10a...n is a user interface that enables the user to interface with the service that corresponds to the service window 10a...n. The service window 10a...n also includes an activation reading part 15a...n. The activation reading part 15a...n is the first function performed when a service window 10a...n is executed and checks to see if the service window is active before any other functions are performed. Further explanation of the service window 10a...n will be discussed below with respect to Figures 30-34. The plug-in 8 may also include a single service data 11a or a number of service data elements 11a...n. The service data elements 11a...n also include an activation reading part 16a...n. As noted above with regard to the service window activation reading part 15a...n, the activation reading part 15a...n ensures that the service data elements are activated. Each service window 10a...n has corresponding service data 11a...n. In addition, the activation reading part of the service window 15a...n corresponds to the activation reading part 16a...n of the service data 11a...n. The service data 11a...n generally includes service name, service id, configuration data corresponding to the service window 10a...n, default service window data and run time data entered by users through service window 10a...n.

[0153] The plug-in 8 also includes a service data handler 12 and optionally may include a login window 13 and login data 14, in other words, an authentication user interface. The service data handler 12 is the portion of the unified client application that uploads data from the MFP to a receiving device. Included in the service data handler 12 is a activation reading part 17. The activation reading part 17 checks the activation information in the config.xml 7 file to ensure that the service data handler 12 is activated before any corresponding functions of the service data handler 12 are preformed. In each plug-in 8, there may be multiple service windows 10a...n and service data elements 11a...n. However, according to one preferred embodiment, there is only one service data handler 12. Other embodiments may have more than one service data handler 12.

[0154] Figure 23 depicts an example of the Document Mall plug-in 8a. The Document Mall service can be installed on the core application 6 as a plug-in 8. When the Document Mall plug-in 8a is installed in the unified client application 5, the services provided by Document Mall are extended to the MFP in which the unified client application 5 is installed. The Document Mall plug-in 8a preferably includes the optional login window 23 and login data 24. These options allow user names, passwords and accounts to be input and utilized by the plug-in 8a, allowing the plug-in 8a to restrict unauthorized users from use of the plug-in 8a.

[0155] The Document Mall plug-in 8a further includes several different service windows and service data. For example, in the Document Mall plug-in 8a, an e-mail service window 20a and a folder service window 20b are included. The email service window 20a is a user interface enabling a user to enter a Document Mall stored email address as a scan destination, while the folder service window 20b is a user interface that enables a user to select a Document Mall folder as the scan destination. Further, an e-mail service data 21a and folder service data 21b are also included. The e-mail service data 21a and the folder service data 21b correspond to the data generated by the e-mail service window 20a and the folder service window 20b, respectively. Both the service windows 20a and 20b and the service data elements 21a and 21b include activation reading parts 25a...b and 26a...b. The activation reading parts are the first functions performed by the service window/data element pairs and are used to ensure that the corresponding service window/data is activated and able to perform a function. The Document Mall plug-in 8b also includes a service data handler 22. In the example of the Document Mall plug-in 8a the service data handler 22 is used as an upload handler that merges both the e-mail service data 21a and the folder service data 21b into one upload.xml file, and sends the upload file to a Document Mall server through an https post command, for example. Other uses for the service data handler 22 not mentioned in this example are also possible. The service data handler 22 also includes an activation reading part 27. The activation reading part 27 allows the service data handler 22 to ensure activation before performing any functions.

[0156] Figure 24a shows the unified client software architecture structure. The unified client application 5,

shown in Figures 20 and 21, is launched by a unified client main thread 30. In Figure 24a, the unified client main thread 30 is shown as including an activation reading part 30a, a project array 31 and a project array window 32. The main thread 30 initializes the core application 6 and uses the activation reading part 30a to read the config.xml file 7 in order to create the project array 31 based on the activation information found in the config.xml file 7. The config.xml file 7 includes activation information regarding several projects 33a...n, each project 33a...n corresponding to at least one activated plug-in 8.

[0157] The project array 31 is a list of projects that are found to be active in the unified client application. The project array 31 is constructed by reading <project> tags and activation information included in the config.xml file 7. Further, the main thread 30 creates service arrays 34a...n for each project 83a...n by reading <service> tags and activation information included in the config.xml file 7. The service array is a list of the activated services installed under a respective project. The main thread 30 also displays the project array window 32. The project array window 32 is the first screen displayed when using or executing the unified client application 6. However, according to one embodiment of the invention, if only one project 33b is installed on the system, the project array window 32 will be bypassed. The project array window 32 displays project buttons for the user to select. When a project button is selected, the corresponding project 33a...n is invoked. It should also be noted that the project array window 32 may or may not display un-activated projects depending on the activation information found in the config.xml file 7. For example, in one configuration if a plug-in 8 is not found to be activated by the main thread 30 a project 33a...n may not be created for the plug-in 8 making it seem to the user as if the plug-in 8 does not physically exist on the MFP. In contrast, in another configuration if a plug-in 8 is not found to be activated the main thread 30 may create a project 38a...n corresponding to the plug-in 8. However, when the user attempts to execute the project 33a...n via the project array window 32 instead of loading the corresponding main window 35 the project 33a...n will give the user the ability to activate the project 33a...n. Additionally, the project 33a...n may give the user the ability to see a demonstration of the project 33a...n or use the project for a limited time. A more detailed discussion of the activation process can be found below with reference to Figure 25E. The project array window 32 will be discussed in further detail below with respect to Figure 27.

[0158] Several projects 33a...n are shown in Figure 24a connected to the project array 31. Each project 33a...n includes an activation reading part 28a. The activation reading part 28a determines how the project 33a...n will be executed and which services 38a...n are included in a service array 34a...n. Each project 33a...n can manage a login/logout process of the project 33a...n through a corresponding login data plug-in 36 and login window plug-in 37. For example, if authentication is needed in the project 33a...n, a login window plug-in 37 can be used to display a login window which will be displayed before the user can begin accessing the project 33a...n. Once the login/logout button is pressed, a corresponding login and logout handler used by login data plug-in 36 will be called.

[0159] Further, the project 33a...n can control the post login process. For example, each service 38a...n can define its own post login process for its service window displayed by the service window plug-in 40a...n. When the authentication succeeds, the post login process of each service 38a...n will be called sequentially.

[0160] The login window displayed by the login window plug-in 37 described above, is an example of an authentication user interface ("UI") display. The login window, displayed by the login window plug-in 37, interfaces with the login data which is included in the login data plug-in 36 and includes an authentication process definition. Additionally, the login window used by the login window plug-in 37 can be implemented to request additional authentication information. As one example, for the Document Mall plug-in 8a, the Document Mall login window 23 may be implemented to include a place for users to enter account information. Other information may be utilized by the login window plug-in 37. Additionally, the login data can be accessed by each service window 40a...n and service data handler 12. Further, each project 33a...n includes a main window 35 and a service array 34a...n.

[0161] In Figure 24a, a main window 35 is associated with the project 33b. Although the main window 35 is only illustrated under project 33b, each project 33a...n may be implemented to include a main window. The main window 35 is used for service management for each service 38a...n which corresponds to a button, the button being a user selectable link to a service window, included in the main window 35. For example, in the Document Mall plug-in 8a example, the main window 35 includes buttons for scanned setting handling, document name input and login button handling. Another example of the main window 35 is discussed below with respect to Figure 28.

[0162] Included in each project 33a...n is a service array 34a...n. Each service array 34a...n includes a list of the activated services 38a...n. A service 38a...n is a function relating to an installed plug-in. Each service 38a...n includes an activation reading part 29a...n, a service window plug-in 40a...n and a service data plug-in 39a...n. The activation reading part 29a...n is the first function activated by a service 38a...n and determines if the service is activated. A service window included in the service window plug-in 40a...n displays a service window user interface. Further, the service window plug-in 40a...n performs the post-login process or gets and sets default values in the service data plug-in 39a...n. For example, in the post-login process of the Document Mall plug-in 8a example, a Document Mall folder service downloads the user's folder list and sets the user's folder as the default folder destination. The service window plug-in 40a...n al-

so performs interactive operations with the user to interact and update the service data in the service data plug-in 39a...n. The service window plug-in 40a...n is an abstract class and, as such, certain behaviors of the service window plug-in 40a...n are predefined in the code. However, a developer is able add features to, or extend the service window plug-in depending on the needs of the developer. For example, in the Document Mall plug-in 8a example, a Document Mall e-mail service window supports both e-mail address search using the Document Mall service, and manual e-mail address entry.

[0163] The service data included in the service data plug-in 39a...n is updated by the service window plug-in 40a...n based on user operations. Further, the service data included in the service data plug-in 39a...n is accessed by an activated service data handler 12 when upload operations are performed. For example, if activated, the sending of e-mails or uploading to network folders may be performed by the service data handler 12 as is done in the Document Mall plug-in example 8a. As with the service window plug-in 40a...n, the service data plug-in 39a...n is an abstract class which can be updated or extended by plug-in developers to create further service related data. For example, in the Document Mall plug-in 8a example, the Document Mall e-mail service sends an e-mail based on the e-mail destination address that is saved in the service data included in the service data plug-in 39a...n.

[0164] Thus, the unified client main thread 30 includes a project array 31 which lists several projects 33a...n which may or may not be activated, each activated project including the service array 34a...n which lists several services 38a...n which may or may not be activated. As discussed above different embodiments of the present invention handle the inclusion of activated and non-activated projects and services in the corresponding project or service array. In one embodiment, only activated projects and services are included in the corresponding project and service arrays. In another embodiment the un-activated projects and services are included in the corresponding project and service arrays, when a user attempts to utilize the functionality of the inactivated projects and services the user is given the ability to buy or activate the service. Further detail regarding this feature will be discussed below with regard to the activation manager. The projects 33a...n found in the project array are displayed on a project array window 32 and each project includes a main window 35, and optionally a login window which is displayed before the main window 35, the login window could alternatively be displayed simultaneously with the main window 35. Further, each service 38a...n includes a service window included in the service window plug-in 40a...n.

[0165] It should also be noted that multiple plug-ins 8 can be associated with a single project 33a...n. For example, if the Document Mall plug-in 8a and eCabinet plug-in 8b are included in a single project 33a...n, users will see both Document Mall and eCabinet service windows 40a...n in main window 35. If a user enters all necessary information in corresponding service windows 40a...n, one scan job can be delivered to both the Document Mall and eCabinet servers. In the case that multiple plug-ins 8 are associated with one project each unique function corresponding to the plug-in has its own activation reading part. For example, if only the Document Mall function is activated the eCabinet functions would either not be available or would need to be activated before use.

[0166] Turning now to Figure 24b, this figure connects to Figure 24a by symbol connector A. Once a scan by the MFP is completed, upload data 50 is created. The upload data 50 includes a document name, scan data, login data and service data, for example. The upload data 50 can also include any other information that can be uploaded by a service data handler 54a...n to a reception device. The service data handler 54a...n includes an activation reading part 55a...n and performs upload of data from the MFP, each service data handler being related to a project 33a...n. An upload thread/job monitor 51 includes a job queue 53. The upload thread/job monitor 51 is a background process that monitors the job queue 53 and processes the jobs when they become available. The upload thread/job monitor 51 is connected to the service data handler 54a...n. When a scan completes, the main thread 30 posts its final upload data 50 and adds it to the job queue 53.

[0167] For each job, the upload thread/job monitor 51 groups upload data 50 based on the corresponding service data handler 54a...n and invokes the corresponding service data handler plug-in 54a...n to process the upload data 50. For example, in the Document Mall plug-in 8a example, the upload thread/job monitor 51 passes generic data such as scan or image file related information, login data e-mail service data and folder service data to the Document Mall service data handler plug-in 54a...n to be processed. Once the upload thread/job monitor 51 has completed the above steps, the final steps are to get a job upload status and update a job log. The job upload status is the status of the upload by the service data handler 54a...n and the job log is the list of jobs processed by the upload thread/job monitor 51.

[0168] As described above, the service data handler 54a...n performs the upload of the upload data 50. However, the service data handler 54a...n will only perform its function if activation is first confirmed by the activation reading part 55a...n. For example, in the Document Mall plug-in 8a example, the activation reading part 55a...n will access the config.xml 7 to confirm that the service data handler 54a...n is activated. If activation is confirmed, then the service data handler 54a...n receives generic data, login data e-mail service data such as e-mail destinations and folder service data such as folder destinations. Finally the service data handler 54a...n composes the received upload data 50 into an upload.xml file and uploads the xml file to a Document Mall server designated in the config.xml file 7 via a http post

process. Finally the service data handler 54a...n reports the upload status to the job monitor for job logging.

[0169] Any processes descriptions or blocks in flow charts should be understood as representing modules, segments, portions of code which include one or more executable instructions for implementing specific logical functions or steps in the process, and alternate implementations are included within the scope of the exemplary embodiment of the present invention in which functions may be executed out of order from that shown or discussed, including substantially concurrently or in reverse order, depending upon the functionality involved, as would be understood by those skilled in the art.

[0170] Figures 25A-25E show a flowchart of the unified client main thread 30. After starting in Figure 25A, the unified client application 5 is initialized in step 60. The unified client application 5 is initialized by first initializing the core application 6. Next in step 61, the activation manager 6b determines activation status of each plug-in and corresponding service found in the config.xml file located on the MFP, further detail is found in Figure 25E. In step 62, the flow determines if at least one plug-in 8 is activated.

[0171] If no plug-in is activated the flow proceeds to an activation window 68. The activation screen enables the user of the MFP to activate, through purchase or trial, at least one plug-in on the MFP. Once a user has activated at least one plug-in the flow would return to step 61 where the activation manager would determine if at least one plug-in 8 is activated.

[0172] If it is determined that at least one plug-in is activated, the flow moves onto step 64 where the config.xml file 7 is read. The config.xml file 7 includes settings for the core application 6 and for the plug-ins 8 which are associated with the host or core application 6. A project array 31 is then constructed in step 65 based on the number of installed plug-ins 8. Next in step 66, the service array 34a...n is constructed for each activated project 33a...n. Further a main window 35 is constructed in step 67. As noted earlier, Figure 28, discussed in more detail below, shows an example of the main window 35. Flow then proceeds to process B in Figure 25B.

[0173] Figure 25E shows a more detailed description of the activation manager 6a workflow. Specifically, Figure 25E shows the internal process of step 61 shown in Figure 25A. Once step 60 in Figure 25A is complete, the flow moves to step 61. Step 61 is made up of five steps 61A-C. In step 61A the activation manager 6a starts up and reads the previously installed config.xml file 7. Included in the config.xml file 7 is MFP information, further discussion of the contents of the config.xml file 7 can be found below with regard to Figures 29A-29B.

[0174] The activation manager then contacts an Activation Database over a network in step 61B and sends information regarding the MFP to the Activation Database to verify the MFP and account information in step 61C. The Activation Database may be a remote cross referenced database that stores information regarding the activation status of each project and service included on the MFP. In step 61D the activation manager 6a retrieves activation information from the Activation Database based on the sent MFP information. The activation information is information regarding the activation status of the projects and services included on the MFP. The activation manager 6a then updates the activation information in the config.xml file 7 based on the received information.

[0175] It should be noted that although in the present embodiment the activation manager updates the config.xml file 7 by contacting an activation database. In an alternate embodiment the activation information could be retrieved from another MFP in the network that had contacted the activation database at a previous time.

[0176] In Figure 25B, the project array window 32 is created in step 70 using the project array. The project array window 32 will only display projects that are activated or are available for activation by the user utilizing the MFP. Thus if the user utilizing the MFP does not have the authority to activate new projects then only the previously activated projects will be available for selection. The project array window 32 is then displayed in step 71 and in step 72 a project is selected based on manual user input.

[0177] Once the project is selected by step 72, step 73 checks to see if the selected project is activated. If the answer is no then the flow proceeds to step 74 where an activation window will allow the user to activate the selected un-activated project. In step 75 it is determined if the user has decided to activate the project. If the user decides to not to activate the selected project the flow will return to step 71 where a new project can be selected. However, if the user decides to activate the selected project in step 74 then flow will proceed to step 76 where the config.xml file will be updated to include the activation information for the newly activated project. Once the config.xml file is updated in step 76 the service array is constructed for the newly activated project in step 77 and the flow proceeds to step 78.

[0178] Returning to step 73, if the selected project is activated then the flow proceeds to step 78 where the selected project is initialized.

[0179] As a general procedure, steps 70-78 perform a process that checks to see if the selected project is activated. If the selected project is not activated then the system allows activation.

[0180] Turning now to Figure 25C, from C in Figure 25B step 80 determines if the initialized project includes a login window plug-in 37. If no login window plug-in 37 is installed, flow proceeds to process E of Figure 25D. If the project includes a login window plug-in 37, then the flow proceeds to step 81 where it is determined if the login plug-in is activated. If the login plug-in is not activated, flow proceeds to process E of Figure 25D. If the login plug-in is activated then flow proceeds to step 82 where both the login window class and the login data class are loaded.

[0181] Once the class files have been loaded, the login window is displayed in step 83. The login window includes both a login button and a cancel button. Depending on which button is pressed in step 84 the flow proceeds differently. When the login button is pressed the flow proceeds to step 85 in which the process login function of the login window plug-in 37 is called. However, if the cancel button is pressed in step 84, the flow proceeds to process D in Figure 25B. Process D returns the flow to step 70.

[0182] If the login button is pressed in step 84 the login function of the login window plug-in 37 is called in step 85. Step 86 checks to see if the login was successful. If the login was not successful, the flow proceeds to step 87 to reset the login window and then returns to step 83. If the login was successful, then the flow proceeds to process E in Figure 25D.

[0183] Thus Figure 25C includes the general procedure of completing authentication if the login window plug-in 37 is installed in the selected project 33a...n. If no login window plug-in 37 is installed or the plug-in 37 is not activated, then the entire login process is skipped.

[0184] Turning now to Figure 25D, in step 89, service data for each service is loaded. Once the service data is loaded, the flow proceeds to step 90 where the post-login function of each service is called. In step 91, the logout listener is set in the main window 35. In step 92, the activation status of each service is checked. At least one service will be activated for each activated project. If the user is not able to activate services only the activated services will be available in step 93, otherwise all installed services will be available. Additionally, in step 93 a service button for each service 38a...n is created in the main window 35. The service window class for each service is then loaded in step 94 and the upload data 50 is initialized or created in step 95.

[0185] Once the upload data 50 is initialized in step 95, the main window 35 is displayed to the user in step 96. The default service is then selected in step 97. It should be noted that the default service will always be an activated service. Then the service window corresponding to the selected service is displayed in step 98. In step 99 service data is input in the service window displayed in step 98. The flow then proceeds to step 100 which checks if the auto-logout time has expired. The auto-logout feature forces the flow to proceed to the logout step 101 if no user activity is detected for a predetermined period of time. If the auto-logout time is determined not to have expired in step 100 the flow proceeds to step 101 which determines if a button was pressed. If a button was pressed the flow proceeds to step 102, if not the flow returns to step 100. Step 102 determined which button was pressed. If one of the service buttons was pressed, then the flow proceeds to step 107 where it is determined if the selected service is activated. If the selected service is not activated then flow proceeds to step 108 where the user have the ability to activate the un-activated selected service. Flow then proceeds to step 109 where it is determined if the selected service was activated in step 108. If the service was not activated in step 108 the flow returns to step 98 where the default service window is again displayed. If the selected service was activated in step 108 then flow proceeds to step 103.

[0186] Returning to step 107, if the selected service is activated then flow proceeds to step 103 where the selected service is set. The flow then returns to step 98 where the newly selected service window is displayed. If in step 102 the logout button is pressed the flow proceeds to step 101 where each service is reset, the main window 35 is reset and the upload data is reset.

[0187] If the MFP "start button" is pressed by user in step 102 the flow proceeds to step 105. In step 105 the scan is completed. The flow then proceeds to step 106 where the upload data 50 is copied and added to the job queue 53.

[0188] Turning now to Figures 26A and 26B, Figures 26A and 26B show a flowchart of the unified client upload thread 51. After starting, a job monitor initialization is performed in step 120. The system then checks if any jobs are in the job queue 53 in step 121. If no jobs are determined to exist in the job queue 53, flow proceeds back to the beginning of step 121. The system continues in this for a loop until a job is observed in the job queue 53.

[0189] When a job is determined to exist in the job queue 53 in step 121, flow proceeds to step 122 and gets the job from the job queue 53 and groups services included in the job based on the corresponding service data handler 54a...n. Next, the generic login data and corresponding service data is passed to the service data handler 54a...n in step 123. In step 124 it is determined if the service data handler 54a...n is activated. If the service data handler is not activated flow proceeds to step 128 where the job is not processed for the service data handler 54a...n. The flow then proceeds to step 129 where the job upload status is sent to the job monitor. Flow then proceeds to step 127.

[0190] If however in step 124 the service data handler 54a...n is activated, the service data handler 54a...n then processes the job upload data 50 in step 125.

[0191] Once the service data handler 54a...n has processed the job upload data 50, the job monitor 51 gets the job upload status from the service data handler 54a...n and updates the job log in step 126. Flow then proceeds to step 127 which checks to see if there are any more service data handlers 54a.._n. If no service data handlers 54a...n remain for the job, then flow moves back to step 121 to check for new jobs in the queue. If more service data handlers 54a...n are included in 127, flow proceeds back to step 123 and processes steps 123-126 again. This loop continues until no service data handlers 54a...n remain for a job.

[0192] Thus the unified client upload thread includes two loops, the first checks for new jobs in the queue. The second loop occurs once a job is determined to exist, in the second loop, the system loops through checks to make sure all of the activated service data handlers

54a...n in a job have been processed.

[0193] Moving now to Figure 27, there is shown an example of a project array window 32. The project array window 32 includes a line reserved for system messages 151. Also included is a unified client logo 152 and instructions to the user on how to use the project array window 153. The project array window 153 also includes several project buttons that are selectable by the user 154 and link the user to the main window 35 and default service window of the selected project 33a ... n. Examples of such buttons are the Document Mall button 154, the eCabinet button 154a or other similar type projects buttons 154b, 154n. The scroll bar 155 allows a number of project buttons to be installed in the project array window 32. Thus the function of the project array window 35.is to allow a user to select which project 33a...n the user may like to use on the MFP.

[0194] Figure 28 shows an example of a main window 35. The main window 35 includes the unified client logo 161 as well as the document name 162 and a logout button 163. As discussed earlier with respect to Figure 25d, the logout button 163 allows the user to log out of the selected project and return to the project array window 32 described in Figure 27. The main window 35 also includes a number of buttons 156 through 159 which correspond to a number of services. The buttons displayed in the main window 35 correspond to the project 33a...n which was selected in the project array window 32. For example, when the Document Mall project is selected 154 in the project array window 35, several Document Mall related buttons are available. For example, the button 156 allows the user to open a scan to a Document Mall e-mail service window. Item 157 allows the user to open a scan to a folder service window. Item 158 is a button that open up the scan settings service window. While item 159 allows the user to open up the job log service window. The invention is not limited to the number of buttons included in Figure 28 or the services shown in Figure 28. Additionally arrow buttons 160a and 160b allow the user scroll through a number of service buttons. Thus, any type of service button can be installed on the main window 35.

[0195] Figures 29A-29B show an example a config.xml file 7 which includes activation information. It should be noted that Figures 29A-29B are not intended to be a comprehensive example of how a config.xml file 7 may be designed. Instead Figures 29A-29B include one way that a config.xml file 7 might be written for activation of a unified client application 5.

[0196] Figure 29B begins with MFP tag on line 1 which opens the MFP section of the config.xml file 7. The MFP section includes several tags which relate the MFP and the account associated with the MFP. On line 2 is found the MFPSerialNo tag which includes the MFP serial number. The MFP serial number is a unique number which identifies the hardware of the MFP. On line 3 is found the MACAddress tag which includes the MFP MAC Address. The MAC address is a unique network identification code that identifies the network interface of the MFP. On line 4 is found the AccountName tag which includes the account name to which the MFP is registered. In the present example the account name is Ricoh.

[0197] On line 5 the UserName tag is found which includes the username of the user currently logged into the MFP. It should be noted that in alternate embodiments no UserName tag is used. In addition, the ModelName tag not shown in Figure 29A can be included in the MFP section. The ModelName tag identifies the model name of the MFP. Finally on line 6 the close MFP tag is found which identifies the end of the MFP section. As noted above with regard to Figure 25E, the portion of the config.xml file 7 enclosed in the MFP tags is the MFP information sent to the activation database. The MFP information is then used by the activation database to determine which services are activated. Thus the data in the MFPSerialNo, MACAddress and ServiceName tags can be used as a unique key.

[0198] Line 7 of Figure 29A includes a service open tag and begins a new service section in the config.xml file 7. On line 8 is found the ServiceName tag which includes the service name of the service. In the present example, the service name DMEmail denotes the DocumentMall Email Service. On line 9 the DisplayName tag includes the display name of the service. In the present example the display name "Document Mall Email" is shown. The display name setting shows how the service will be displayed in the service buttons in the main window 35.

[0199] Line 10 shows the Activation open tag. This tag begins the activation section of the service. Included in the activation section are several tags related to the activation of the service. The examples shown in Figures 29A and 29B are one way of including activation information in the config.xml 7 file, other ways are also possible. On line 11 the ActivationRequired tag is found. This tag includes a boolean indicator which denotes whether or not activation is required for the service in question. In the present example, shown on line 11, the ActivationRequired tag is listed as "Y", however if the tag included a "N" or "F" indicator the service would always be available to be used on the MFP. The default value for this tag is "'Y".

[0200] The next tag in the activation tag section is the Activated tag which is found on line 12. As with the ActivationRequired tag noted above, the Activation tag includes a boolean indicator. The Boolean indicator corresponds to whether or not the service in question is activated. If the indicator shows "N" or "F" then the service will not be available to the user of the MFP unless the user goes through an activation process. In the present example the Activated tag includes a "Y"' indicator. When a "Y"' indicator is found in the Activated tag, the ActivationDate and ExpirationDate tags found on lines 13 and 14 list the date that the service was activated and the expiration date of the activation, respectively. The ExpirationDate tag is useful in the case that the Activation

Database is unable to be contacted. If the Activation Database is unreachable, the activation manager can compare the internal date stamp of the MFP with the information found in the ExpirationDate tag of the config.xml file 7 to ensure that the activation is still valid. Finally, the Activation close tag is found on line 15.

[0201] In should be noted that several different tags may be used in the Activation Section depending on the type of activation used. In the present example, time-based activation is used, however, when different types of activation are used different tags may be used in the activation section.

[0202] In lines 16-19, the service window class file is listed. The service window class file includes all the code necessary to display the service window. In lines 20-23 the data handler class file is listed. This includes all the code necessary for the data handler in this service. Beginning on line 24, configuration data for this service is included. In this example, on line 25, the DocumentMall server address is listed as documentmall.com. The address documentmall.com is an example of an address that may be used, other addresses including IPv6 addresses or IPv4 addresses can also be used. Beginning on line 27, the data handler configuration data is included. In this example, the data handler configuration data is listed as optional. However, information such as FTP port or other similar data can be listed in this tag. On line 26-27, the data handler configuration data tag is closed and on line 29 the service is closed for the above noted service.

[0203] Line 30 includes a new service tag which corresponds to a new service. It should be noted that although the present example only includes two service sections, a service section corresponding to each service found on the MFP may be included in the config.xml file 7. On line 31 the ServiceName tag is found which, in the present example, shows the eCabinetFolder service and on line 32 the DisplayName tag is found which includes the name "eCabinet Scan to Folder."

[0204] Lines 33 of Fig 11A to Line 1 of Fig 11B show the Activation section for the eCabinetFolder service. As was described with regard to the DMEmail service above, the Activation section of the eCabinetFolder service includes open and close Activation tags, an ActivationRequired tag, an Activated tag, an ActivationDate tag and an ExpirationDate tag.

[0205] In lines 2-4 of Figure 29B the service window class file is listed. The service window class file includes all the code necessary to display the service window. In lines 5-8 the data handler class file is listed. This includes all the code necessary for the data handler in this service. Beginning on line 9, configuration data for the DMEmail service is included. In this example, on line 10 the eCabinet server address is listed as eCabinet.com. The address eCabinet.com is an example of an address that may be used, other addresses including IPv6 addresses or IPv4 addresses may also be used. Beginning on line 12, the data handler configuration data is included. In this example, the data handler configuration data is listed as optional. However, information such as FTP port or other similar data can be listed in this tag. On line 12-13, the data handler configuration data tag is closed and on line 14 the service is closed for the DMEmail service.

[0206] It should be noted that the foregoing example does not show project tags or corresponding project activation sections, the config.xml file 7 may include activation sections in subordinate to the project tag in a similar manner to the placement of the activation section in each service section described above and illustrated in Figures 29A-B.

[0207] A functional example of the unified client application 5 installed on a multi-function printer will now be described in Figures 30-34. In this example, the unified client application 5 is installed with the eCabinet plug-in 8b. The unified client application 5 with an eCabinet plug-in 8b is developed using SDK/J and uses the CVM option on each MFP in which the unified client application 5 is installed. SDK/J is an embedded software architecture software development kit ("SDK") which allows in house developers, independent software vendors and system integrators to deliver customized JAVA based solutions on MFPs. The CVM option is the java virtual machine that is able to be installed on the MFP. Other types of virtual machines and/or programming languages can be used to create plug-ins associated with the unified client application.

[0208] The example of the unified client application 5 with the eCabinet plug-in 8b uses 2 SDK/J type applications, the 2 SDK/J applications are, for example, one Java xlet application which implements major unified client functionalities and one servlet application which allows user to update the config.xml file 7 remotely via a web browser. Some of the services that are supported by the unified client application 5 with the eCabinet plug-in 8b are: scan to eCabinet server, scan to eCabinet folder, scan settings and job log viewing. These services are represented as service buttons in the eCabinet project main window 35. In the case that the unified client application 5 only includes one project 33a...n installed, such as in the present example, a default service window is the first window displayed along with the main window 35.

[0209] In Figure 30, an example of a main window 35 and service window 173 is shown. The main window 35 and the eCabinet owner service window 173 are displayed. The main window 35 includes a logo 161 as well as the document name 162 and an end session or logout button 163. Further, several service buttons 164-167 are also included in the main window 35. In the eCabinet example, the first service button is the eCabinet owner button 164. In figure 30, this button is selected and as a result the corresponding eCabinet owner service window 173 is displayed. The left side of the eCabinet owner service window includes an order list window 168 and a refresh button 169. Further, on the left side of the eCabinet owner service window 173 there is a selected owner's window 170. Also included are a public 171 and reset

button 172. The eCabinet owner button offers users the scan to eCabinet owner service. The owner list is downloaded from the eCabinet server automatically and is displayed in the owner list window 168. Multiple owners can be selected if no eCabinet folder is selected in the eCabinet folder window 193. When an eCabinet folder is selected in eCabinet folder window 193, only a single owner selection is allowed. The owner list window 168 shows a list of the owners. The selected window 170 shows the destination owners. To add a destination owner, the user can highlight desired owners in the owner list window 168 and press the right arrow button 175. To delete a destination owner, the user can highlight the owner in the selected window 170 and press the left arrow 176. The refresh button 169, allows the user to download the owner list from the server again. The public button 171 allows the user to set the attribute of the scan document to public or private. The reset button 172 allows the user to remove all of the contents of the selected window.

[0210] Figure 31 shows an example of the main window 35 with the eCabinet folder service button 165 selected and the eCabinet folder service window 193 displayed. In this example the eCabinet folder button 165 has been selected and as a result the eCabinet folder service window 193 is displayed. The eCabinet folder service window 193 includes a folder list window 189, a refresh button 190, a selected window 191, and a reset button 192. The eCabanet folder service offers users the ability to scan to the eCabinet folder service. The eCabinet folder list is downloaded from the eCabinet server automatically using the configuration settings included in the config.xml file 7. When users select the eCabinet folder button 165 the unified client application 5 prompts user with a software keyboard to enter a user name and a password. The unified client application 5 then downloads the user's folder tree and displays the tree in the folder list window. Note that using the eCabinet folder service requires single owner selection. If multiple owners have been selected in eCabinet owner service window 173 and the user presses eCabinet folder button 165 an error message will pop up stating eCabinet folder service requires single owner selection. The folder list window 189 shows a user's eCabinet folder tree. The user can browse the folder tree in the folder list window 189. To add a destination folder, the user can highlight the desired folder in the folder list window 189 and press the right arrow button 175. To delete a destination folder in the selected window 191, the user can highlight the desired folder in the selected window 191 and press the left arrow button 176. It should also be noted that multiple folders can be selected. The refresh button 190 allows the user to download the eCabinet folder list again from the ecabinet server. If the refresh button 190 is pressed, the user will be prompted for the user name and password entry again. The reset button 192 allows all the contents placed in the selected window 191 to be removed. It should also be noted that the eCabinet folder list included in the folder list window 189 is dependent upon the owner selected in the eCabinet owner service window 173 included in Figure 30. The user that is selected and included in the selected window 170 is the user who corresponds to the folder list included in the folder list window 189.

[0211] Figure 32 shows the user interface for an example of when the scan settings button 166 is selected. When the scan settings 166 button is selected, the scan settings service window is displayed 218. The scan setting service window includes several options including resolution 209, original 211, image type 214 and file format 215. Under the resolution option 209, several different buttons relating to scanner resolution are used. In this example, DPI 200, 210a, 300 DPI, 210b, 400 DPI, 210c, or 600 DPI, 210n, are available to be selected. Other similar types of DPI options resolution options could also be used. The original option 211, includes two buttons. The first button 212a allows the one sided option to be selected. The second button 212n allows the two sided option to be selected. Further, a batch scan button is displayed 213. The image type option 214 also includes a drop-down menu listing a number of image types. In the current example, the text option is displayed. It should also be noted that in the image type drop-down box text, photo, gray scale or photo options are available. Similarly, under the file format option 215, a second drop-down box is included listing a number of different file formats. In the present example, the PDF option is displayed. However, in the file format drop-down box single page tiff, multi-page tiff, jpeg and PDF options are available. Also included on the scan setting service window 218 is a scan size 216 button and a reset button 217.

[0212] The scan size button 216 opens a new window which is shown in Figure 33. The scan size window 219 is still part of the scan sittings service window 218. However, the scan size window 219 is displayed in place of the scan setting service window 218 under the main window 35. In the scan size window 219, several different options are available. For example, auto detect 239, 8x11 5-1/2 x 8-1/2 A5, 240a, 8-1/2x11 5-1/2x8-1/2 A5 240b, 11x17 a3, B4 JIS 270c, 8-1/2x13 A4 B5 JIS 240d, and 8-1/2x14 A4 B5 JIS 240n. Also included are a reset button 242 and a general button 241 which returns the user to the original scan settings service window 218.

[0213] Figure 34 shows the main window 35 and the job log service window 264 displayed when the job log button 167 is selected. In the job log service window 264 date and time 259, document name 260, pages 261 and status 262 titles are displayed. From the job log service window 264 users can check scan job upload status specifically through the date and time, the document name, number of pages and the status of the job. This concludes the MFP display example of the eCabinet plug-in 8b.

[0214] Figures 35-37 show an example display for managing the settings of the eCabinet plug-in 8b remotely. The unified client application 5 with the eCabinet plug-in 8b can be configured remotely through web access. For security purposes the website is protected by a pass-

word. Figures 35-37 show an example of the remote configuration website for eCabinet plug-in 8b.

[0215] Figure 35 shows an example of the general configuration window 300 on the remote configuration website of the unified client application 5 with the eCabinet plug-in 8b installed. When a user accesses the website shown in Figure 35 three options will be shown. The general window button 271 the eCabinet server button 272 or the default scan settings button 273. These three options correspond to three screens: the general screen 300, the eCabinet server configuration screen 290 and the default setting configuration screen 270. Several settings are configurable through the general configuration screen 300 that is selected by the general button 271 and is the default screen loaded. First the enable/disable document name with time stamp suffix 302 is checkable. The machine reset timer seconds 303 is available to be changed. The machine reset timer seconds 303 setting relates to an auto session logout with refresh timer. In this example, 600 seconds is placed in the auto session refresh timer. The change administrator password option 304 is also available to be selected; this setting allows the user to change the administrative password for using the remote configuration service. It should also be noted that reinstallation can reset the password to the default. Also included are update 282 and cancel 283 buttons which allow the user to update and to apply the changes that the user has made in the general configuration window 300 or cancel the changes.

[0216] Figure 36 shows the result when the eCabinet server button 272 is selected. The eCabinet server button displays the eCabinet server window 290 which allows the following options: eCabinet server address and FTP port. Both of these are required fields as is shown in 281. The eCabinet server address 291 and FTP port 292 can both be entered by the user. The FTP port 292 is automatically filled with the default ftp port. As with Figure 35, update 282 and cancel 283 buttons are available.

[0217] Figure 37 illustrates the example of when the default scan settings button 273 is selected. When the default scanning settings button is selected the default scan settings window 270 is displayed. In the default scan settings window 270 a number of options are displayed. First the default scan resolution 275 is available to be changed. In this example, 200 dpi is selected. In the unified client application 5 with eCabinet plug-in 8b installed the default scan resolutions of 200 dpi, 300 dpi, 400 dpi and 600 dpi are available and are displayed in the drop-down box in item 275. The default batch scan option is also selectable 276 along with the duplex 277 option. A default image type 278 is also available to be selected in the eCabinet plug-in 8b example. The default image types available in the drop-down box of item 278 are text, print, text photo, photo or grayscale. These image types correspond to different qualities of the scanned image. The next option available is the default text photo file format 279. In this example, the multi-page tiff option is selected. In the unified client application 5 with the eCabinet plug-in 8b installed the options available for the drop-down box of 279 are single page tiff, multi-page tiff or pdf. Single page tiff is a tiff image file that only includes a single image per file. The multi-page tiff is an image file that includes several images. PDF is a proprietary format to Adobe Systems which includes multiple page fixed-layout documents. The final option is default grayscale/color file format 280. In this example, jpg is selected as the default grayscale/color file format but the pdf option is also available in the unified client application 5 with eCabinet plug-in 8b installed. Item 281 shows required fields that must be selected. As with the Figures 35 and 36, update 282 and cancel 283 buttons are available. The update button 282 and the cancel button 283 allow the user to apply the changes with the update button 282 or cancel the changes with the cancel button 283.

[0218] Fig.38 is a block diagram of an MFP according to an embodiment of the present invention. As shown in Fig.38, the MFP 300 includes an application layer 301, an OS 302, and hardware resources 303.

[0219] The application layer 301 includes the unified client application 304 as well as several plug-ins 305a...n which are included in the unified client application 304. It should also be noted that the unified client application includes an activation manager 306 that interfaces with the plug-ins and limits the plug-ins 305 ability to be accessed by the unified client application 304.

[0220] The application layer 301 is a position in a software hierarchy in which applications installed on the application layer 301, such as the unified client application 304, access the hardware 303 through the OS 302. Further the plug-ins 305a...n access the OS 302 and the hardware 303 via the unified client application 304 installed on the application layer 301.

[0221] It is also important to note that the application layer 301 is independent from the OS 302 and although the application layer 301 accesses the hardware through the OS 302 it is not a part of the OS 302.

[0222] The OS 302 is any operating system that accesses the hardware 303. Further, the OS 302 acts as a conduit for allowing applications that are installed on the application layer 301 to access the hardware 303.

[0223] The hardware 303 is the physical components of the multi-function printer. For example, hardware 303 can include a scanner, a printer, a fax or any other hardware component.

[0224] Fig.39 shows an example of a hardware configuration of the MFP 499 according to an embodiment of the present invention. As shown in Fig.39, the MFP 499 includes a controller board 400, an operation panel 410, a fax control unit (FCU) 420, a USB 430, an IEEE 1394 port 440, and a printer 450. It should be also noted that other types of i/o interfaces could be included including IEEE 1394b, USB 2.0. The controller board 400 includes a CPU 402 for processing and several storage devices such as SDRAM 403, SRAM 408, flash memory (flash ROM) 404, flash card interface part 406 and HD 405 used to store data associated with the MFP 499.

Each of these components are connected to the ASIC 401, the ASIC 401 is an application specific integrated circuit that is designed specifically for use in a MFP 499. Other types of storage devices are also possible as well as other types of data processors and integrated circuits. The operation panel 410 is directly connected to the ASIC 401 as is the communications interface 420. The communications interface 420 can also be connected to a network or any other similar type communications medium. The USB 430, the IEEE 1394 440 and the multifunction printer functions 450 such as scanning, printing, and faxing are connected to the ASIC 401 via the PCI bus 480.

[0225] The SRAM 408 is a nonvolatile RAM, other types of SRAM are also possible. A flashcard 407 can be inserted into a flash card interface part 406, so that data is sent/received between the ASIC 401 and the flashcard 407 via the flash card interface part 406.

[0226] The operation panel 410 includes an operation part used for key operation such as key input and button pushing and the like by the user, and a display part for displaying drawing data such as various screens. It should be appreciated that other types of hardware components can be used in the present invention.

[0227] Further with respect to a computer readable medium such as a floppy disk, magnetic tape, CD-ROM and the like, by installing the program stored in the computer readable medium into an MFP, the MFP can perform the functions of the present invention.

[0228] This invention has been described with respect to a multifunction printer, but is applicable to any image handling device such as a copier, digital copier, printer, scanner, digital camera, fax machine, or multi-function printer or any combination thereof. A general purpose computer is not considered an image handling device. Moreover, the invention is applicable to other special purpose devices such as navigation systems, global positioning systems, vending machines, metering systems, machine tools and other tools which operate using programming or a programmed processor, automobiles, other transportation devices such as trains, motorcycles, planes, or boats, radar systems, radios, MP3 players, digital music players, and other audio systems, mobile phones, other communication devices and systems, and any other special purpose device which operates using a plug-in.

[0229] The present invention is not limited to the specifically disclosed embodiments, and variations and modifications may be made without departing from the scope of the present invention.

[SYSTEM AND METHOD FOR IMAGE THUMBNAIL/ PREVIEW ON AN IMAGE PROCESSING DEVICE]

[0230] Referring now to Figures 41-65 of the drawings, wherein like reference numerals designate identical or corresponding parts throughout the several views and more particularly to Figure 41 thereof, there is illustrated a typical example in which input documents 2 are input into a multi-functional device 1 which delivers output documents 3. An embodiment of the present invention enables a preview of the input document 2 before it is output as an output document 3 or sent to a GlobalScan server 5. The preview image 4 of the scanned input documents 2 enables the user to easily review the scan of the input document 2. Additionally, if the GlobalScan server 5 is being used along with the multifunction device 1, the image preview allows the user to review multiple scanned images before a job is committed to the Global Scan server 5 for final processing. The GlobalScan server 5 serves as a digital document routing system that accepts, organizes and controls scanned documents in addition to creating digital files for electronic transmission or for storage. Further, information on the GlobalScan server 5 and system can be found in related applications 11/092,831, "System and Method for Authenticating a User of an Image Processing System," filed March 30, 2005, 11/092,836, "System and Method for Managing Documents with Multiple Network Applications," filed March 30, 2005 and 11/092,829, "System and Method for Compensating for Resource Unavailability in an Image Processing System," filed March 30, 2005.

[0231] According to an embodiment of the present invention, a user previews an input document 2 on the multi-function device 1 before it is submitted to the GlobalScan server 5 for final processing. It should be noted that, according to an embodiment of the invention, the image preview features of the multi-function device 1 ("MFD") maybe the same, even if different devices are used for processing the preview operations. The MFD 1 may be used to locally handle the preview operations. Alternatively, the GlobalScan server 5 may be used to perform each operation in the preview process that is performed before final submission. In addition, the administrator is able to activate and de-activate the preview/thumbnail function on the MFD 1 remotely and the administrator can determine if the preview function's operations will be performed using the GlobalScan server 5 or the MFD 1.

[0232] The user interface for operating the image thumbnail/preview function on the multifunction device will now be described with reference to Figures 42A-C.

[0233] Figure 42A illustrates a preview page 10 which is displayed below a GlobalScan main window 9. The preview page 10 is displayed when a user selects a project and then selects or presses a preview tab 13, which is located on the GlobalScan main window 9. Also included in the GlobalScan main window 9 are a scan to folder tab 14, a scan settings tab 15, a job log tab 16, a document name button 11, a project button 12 and scroll buttons 21a and 21b. The scroll buttons 21a and 21b are used to scroll between available tabs if there are a greater number of tabs then there is space available on the display window. The document name button 11 allows the user to change the name of the input document 2 being scanned. The project button 12 allows the user to exit the currently selected project and select a new project.

The Email tab 14 allows the user to scan to email. The Scan settings tab 15 allows the user to change the localized settings for the scan such as resolution etc. The job log tab 16 allows the user to see a list of previously preformed scan jobs on the MFD 1.

[0234] As noted earlier, when the preview tab 13 is pressed on the GlobalScan main window 9 the preview window 10 is displayed. The preview window 10 includes a preview button 20a. In order to perform the preview function the preview button 20a is preferably, although not necessarily, highlighted. The preview button 20a is placed into and out of a highlighted state by pressing or clicking the preview button 20a. If the preview button 20a is pressed before a scan job is commenced, after the scan job is completed the preview function will be started automatically. If the preview button 20a is not selected the preview function will not be performed. Figure 42B shows an example of when the preview button 20a is highlighted.

[0235] Also included on the preview window 10 is the preview page range dropdown box 19a. The preview page range dropdown box 19a enables the user to select how many of the input document's pages he/she desires to preview. The options that are available to the user are pre-set on the server side, for example. In the example illustrated in Figure 42A, the optional "all" is selected in the preview page range drop-down box 19a. However, in Figure 42B the option "none" is selected in the preview page range dropdown box 19a. Figure 42C shows all of the options in this example that are available in the preview page range dropdown box 19a. In this example, the options are "none", "all", "first page only", "first five pages" or "last page only". However, other types of restrictions on the number of pages that are shown may also be set.

[0236] On the left side of the preview window 10 of Figure 42A there is included a color type button 17. In this example, the color type button shows a value of 2. Thus, the color type portion of the preview window 10 enables the user to select the thumbnail's color type.

[0237] When the image preview functions are engaged and the user selects the start button initiating a scan, the MFD 1 scans the input document, uploads the job to the server 5, retrieves thumbnails and displays them, as is described below with respect to Figures 43 and 44. According to one embodiment, the server 6 preferably generates thumbnails only for the scanned pages the user has selected in the preview page range dropdown box 19a. Alternatively, the system may be able generate thumbnails for any or all scanned pages in the job. In this embodiment, the operations that are performed by the user on the preview images are performed only on those selected pages that the user previews. However, when the preview is accepted, all pages in the scan job are sent to the job monitor, even though only the pages that were previewed were changed. Alternatively, the user may be able to make a change to one previewed page and apply this change to each page or a number of selected pages, even if the page or pages were not previewed by the user.

[0238] Figure 43A illustrates a thumbnail selection window 34 which is displayed after the preview button 20A is highlighted and a scan job is performed. The thumbnail selection window 34 includes a number of thumbnails which each represent one of the scanned pages or images of the scan job performed on the MFD 1. In one embodiment, the thumbnail size is predetermined and stored in the MFD 1. Alternatively in another embodiment the thumbnail size may be determined by settings on the GlobalScan server 5. As shown in thumbnail window 34, thumbnails 35A-35D are selectable icons which the user can select to further process the specified page or image of the current scan job. If a user's scan job includes only one document, the thumbnail selection window 34 will not be shown according to one embodiment.

[0239] In addition to a number of thumbnails, the thumbnail selection window 34 includes a scroll bar 37 which, when pressed or selected, shows additional image thumbnails that could not be shown in the display window. Figure 44 shows the additional thumbnail 35E that could not be shown in the original window in Figure 43A. In order to return to the window shown in Figure 43A, an up button on the scroll bar 37 may be used.

[0240] When an image thumbnail is selected in Figures 43 or 44, a preview detail window 60 shown in Figure 45 is brought up. The preview detail window 60 includes a back button 61 which returns the user to the thumbnail selection window 34 shown in Figures 43 and 44. The image preview detail window 60 includes a preview window 51 which shows a preview of the scan document. Also included in the image preview detail window 60 are up and down pan buttons 42 and right and left pan buttons 43. These pan buttons allow the user to pan through the previewed image. On the left side of the image preview detail window 60 above the pan buttons is a page information window 41 which includes information about the selected previewed page or image. In the example shown in Figure 45, the page information window 41 includes an identifying number of the thumbnail, an image size and a color mode of the thumbnail image. Other information may also be displayed in the page information window 41, such as, process history, page resolution, page identification, bar code information from the page, watermarks, or other types of information.

[0241] In addition, several buttons are included in right side of the preview image detail window 60. The first button included is a left arrow or previous button 44. The previous button 44 enables the user to see the page that was scanned directly before the currently displayed page. A right arrow or next button 45 is also included which allows the user to view the page that was scanned directly after the currently displayed page. Also included are a rotate forward button 46 that allows the user to rotate the image 90 degrees clockwise and a rotate back button 47 that allows the user to rotate image 90 degrees counterclockwise. A zoom-in button 48 that allows the user to zoom in on the image in preview window 51 is

included directly above the rotate forward button 46. A zoom-out button 49 that allows the user to zoom-out from the image in the preview window 51 is included next to the zoom-in button 48. Finally, a delete button 50 that allows the user to delete the currently displayed image from of current job is included. The delete function initiated by the deleted button 50 is useful for deleting blank pages or for deleting pages mistakenly scanned in the scan job.

[0242] With respect to the rotation forward button 46 and rotate back button 47, these buttons change the orientation of the final scanned page. Once the preview job is submitted, the rotated result will be seen in the final electronic or paper documents. In addition, another feature of the rotation function is a "follow-me" rotational feature. The "follow-me" rotational feature allows a user to rotate an entire image or page while keeping the image or page in view on the display screen. In one embodiment of the present invention the display screen may be a LCD display screen or alternatively the display screen may be any type of display screen that is available to be used with the MFD 1. The size of the display screen is only limited by the physical characteristics of the MFD 1 and can be placed in any feasible location on the MFD 1. Although not illustrated in Figure 45, the preview image detail window 60 may also include horizontal and vertical flipping functions which enable the user to flip the currently displayed image.

[0243] The zoom-in 48 and zoom-out 49 buttons make the previewed image larger or smaller on the window. The zoom-in feature enables the users to enlarge portions of the page. Each previewed image or page can be zoomed based on a percentage that is, preferably in one embodiment, predetermined by the administrator on the server. Alternatively, the user may be able to change the percentage of the zoom locally. In one embodiment, the zoom function does not affect the original document and only allows the user to see more closely the previewed image. Alternatively, the zoom function may be used to permanently change the zoom of the scanned page. When the zoom-in button 48 and zoom-out button 49 are used, the up and down panning scroll bar 42 and the left and right panning scroll bar 43 may be used to pan through a zoomed portion of the image.

[0244] Figure 46 shows an illustration of a pop-up box 70 that is displayed when the delete button 50, shown in Figure 45, is selected. If the user does not desire the currently displayed page to be included in the final job set, the current page may be deleted with the delete function initiated by selecting the delete button 50. When the delete function is used, the current page preferably is not generated in the final document. Once the delete button 50 is selected, the pop-up window shown in Figure 46 is displayed. The user may select either the ok button 71 to agree with the deletion or the cancel button 72 which returns the user back to the preview window. If the user presses the ok button 71 the current page is deleted and the preview window displays the next page of the scan. If the deleted page is the last page in the total scan, the next image shown in the preview window is the previous image in the scan.

[0245] After deleting an image or selecting the back button 61, the user is returned to the thumbnail selection window 34 shown in Figures 43 and 44. When the user has finished previewing the pages the user may select one of a restore button 31, a cancel button 32 or a submit button which are shown in the top right corner of the window shown in Figures 43 and 44.

[0246] When the cancel button 32 shown in Figures 43 and 44 is pressed, a cancel job pop-up box 80 shown in Figure 47 is displayed. The cancel job pop-up box 80 includes an ok button 81 and a cancel 82 button. When the user selects the ok button 81, the preview operation, as well as the scan job, is cancelled and no document is generated at the job's destination. In contrast, if the user selects the restore button 31, a pop-up box 90 shown in Figure 48 will be displayed. The pop-up box 90 also has an ok button 91 and a cancel button 92. If the ok button of the undo pop-up box 90 is selected, then the preview operation is canceled and the user is sent back to the beginning to do the preview operation again. When the preview operation is canceled, all rotations, deletions and other such changes are removed from the record and the scan job image set will be returned to its first scanned state. Finally, the submit button 33 shown in Figures 43 and 44 is used to submit the preview changes to the GlobalScan server 5 or to a predetermined output portion of the MFD 1.

[0247] Figures 49-52 illustrate a number of settings that may be defined on the server side before the user uses the preview function on the MFD 1. Specifically, an administrator is able to change some aspects of the image thumbnail preview user interface discussed above. The administrator is able to change the settings on the GlobalScan server 5 in several different ways including locally and remotely via software or via a web-browser. In the present example illustrated in Figures 49-52, a web-browser is used to access the GlobalScan server 5 via an http protocol.

[0248] In Figures 49A&B, a manage services window 100 is selected by selecting a manage services button from a GlobalScan menu 107. The manage profile window 100 includes information about a number of services. Specifically, each service includes a service name 102, a service description 103 and number of informational check boxes 104 and a configure button 105. The order of the display of the services in the manage services window 100 can be changed using the sort bar 101. In the sort bar 101 is included a sort button a "sort by" drop-down box and a "sort direction" drop-down box. The "sort by" box enables the user to select which category (i.e. name 103, description 103 or one of the information check-boxes 104 such as "Job") is used to sort the list of services. The "sort direction" drop-down box lets the user select an ascending or descending order of the sort.

[0249] When the user selects the configure button 105,

a service (plug-in) configuration window 160, shown in Figures 50A&B, is displayed. Included in the service configuration window 160 are several settings related to the selected service. In the example shown in Figures 50A&B, the image preview service (plug-in) includes a threshold setting 111, a zoom level setting 112, and a page options setting 113-118. Also, included are an update button 119, a close button 120 and a delete button 121.

[0250] The threshold setting 111 includes a drop-down box which enables the administrator to set a value for the previewed image's contrast. The lower the number selected by the administrator, the less contrast and as a result less detail shown on the preview image. It should be noted that this setting does not affect the contrast of the scanned image.

[0251] The zoom levels setting 112 includes a number of zoom percentages that are able to be selected for use by the administrator. In the zoom levels selection box includes zoom magnifications that range from 20% to 200%. The zoom levels are listed in the box in ascending order and in whole integers. The zoom level 164 corresponds to how much zoom is applied when the zoom-in 48 or zoom-out 49 buttons are selected in Figure 45.

[0252] The page options setting 113-118 enables the administrator to select which options will be available in the preview page range dropdown box 19a shown in Figures 42A-C. In page option selection box 113, the administrator is able to select which options will be available in the preview page range dropdown box 19a. In the present example, the options available for selection are "none", "all", "First X", "Last X" and "First X, Last Y". The values for the "X" and "Y" variables are editable in option boxes 117 and 118. Theses X 117 and Y 118 settings correspond to the X and Y variables in the page option selection box 113. The values may be any integral value and are preferably kept small as a number less than 10 should be adequate. Also included in the Page option setting are the default page selection option 114. Using the add 115 and delete 116 buttons the administrator can add and delete the default selection shown in box 114.

[0253] When the update button 119 is selected, the changes made in the service configuration window 160 are stored in a configuration file that is used to update the available options of the MFD 1. When the close button 120 is selected the administrator is returned to the manage services window 100 shown in Figures 49A&B. When the delete button 121 is selected the settings are cleared.

[0254] It should also be noted that when the configure button 105 is selected for the RicohScanSettings service (Image preview function) the service configuration window 160 displayed corresponds to the default system service configuration. Once these settings are saved by selecting the update button 119, each unchanged profile on the GlobalScan server 5 is updated to have settings corresponding to the default system service configuration. However, if the administrator desires to customize the settings of a specific profile or project, this profile or project can be selected from the selection toolbar 110. For example, in the selection toolbar 110 the administrator could select the "2 - RS" profile. When this profile is selected the administrator could then adjust the settings in the service configuration window 160 for this profile. This can also be accomplished for the projects included in the selection toolbar 110.

[0255] In order to add or remove a profile or project from the selection toolbar 110 the Project/Profile service window shown in Figures 51A&B and 12A&B may be used. After selection is made in the GlobalScan menu 107 of a specific project or profile, the Project/Profile service window is displayed. In the Project/Profile service window the project/profile id 139 is displayed along with the project/profile name 140. Also included in the Project/Profile service window are update buttons 130/133, delete checked items buttons 131/134 and delete all buttons 132/135. The update buttons 130/133 save the changes made my the administrator, the delete checked items buttons 131/134 remove the checked services from the listed services and the delete all buttons 132/135 removes all the services from the service list. the service list includes all of the services that are available to the profile in question. The service list includes for each included service, a service name 102, a number of service settings 104, a display sequence 136, a processing order 137, a required check-box 138, a configure button 105 and a delete button 106. The delete button 106 deletes the service from the service list. The configure button 105 takes the administrator to the service configuration window 160 shown in Figures 50A&B.

[0256] In order to add a service to the service list, the add-service drop-down box 141 lists all available services. When a service is selected the administrator is able to select the add button 107 which adds the service to the service list. Once the service is added to the service list for a project or for a profile, the profile/project shows up on the selection toolbar 110. Figures 52A&B shows an example of the Project/Profile service window after the RicohImaging service is added to the service list.

[0257] Figure 53 illustrates the process of how the GlobalScan server 5 controls the functions of the MFD 1. The MFD 1 may have difficulty processing native image files such as JPEG or TIFF files as the MFD 1 may not be configured to easily manage any other type of image file other than bitmap images (BMP), making image modifications difficult. In order to overcome these obstacles, a GlobalScan server 5 may be used to perform the image operations in place of the MFD 1. Although using an external GlobalScan server 5 solves the above noted problems, it also creates additional obstacles. For example, when an external server is used, a user session may need cleaning up when a user unexpectedly leaves due to an unexpected disconnection. Additionally, network bandwidth increases with each image preview operation and the images in the native image format JPEG or TIFF need to be converted into Bitmap images and back again.

[0258] To overcome these issues, the GlobalScan server 5 employs several techniques which include adding to the MFD 1 a cancel job feature and that may be the default server action if session timeout occurs. In addition, the server 5 can perform down-sampling of bitmap files as, for example, a 400 dpi full color JPEG can generate a 90 megabyte bitmap file and performing previews and rotations on such a large file is not efficient and adds overhead to the network. In addition, the GlobalScan server 5 is able to down-sample high quality images to a quality that makes best use of the LCD panel found on the MFD. This eliminates the possibility that the user may notice the change in quality. In addition, the server tracks a history of operations on pages or image files such as, for example, rotations or deletions, and performs these operations on the down-sampled images during the preview session. The server may then perform all operations at once at the end of the preview session on the original images. This allows for multiple rotations to be combined into one operation or deletions to delete the image all together.

[0259] Figure 53 illustrates the flow for setting up the preview page 10 and preview tab 13 illustrated in Figures 42A-C. The flow begins in 200 when the MFD 1 calls a GetAvailableServices function. The GetAvailableServices function is called through a APIGetAvailableServices.aspx active server page 214. When the MFD 1 uses this aspx file to make a request to the server 5, the GlobalScan server 5 generates a typical services xml structure but also includes a new service ID that is created for the preview tab 13. Additionally, the server 5 embeds all localization information for the image preview in this call. The server 5 then returns an xml file containing the normal available services information as well as the data for the preview tab 13 in 216. In step 204, the MFD 1 prepares the preview tab 13 from the information obtained by the GetAvailableServices function including any server defined default values. In step 206, before scanning, the user selects the preview tab 13 and highlights the Preview button 20a. In step 208, the user fills out the requisite information in other tabs (14-16). Finally, in step 210, after placing the preview button 20a in a highlighted state and completing the additional settings in the other tabs, the user presses start to initiate a scan.

[0260] Figures 54A-C illustrate an xml information retrieved by the GetAvailableServices function run on the GlobalScan server 5 and initiated by the MFD 1. In line 1, a screen_data tag opens the file. Lines 2-23 include the settings for an option drop-down box shown as the Preview Page Range box 19a in Figures 42A-C. Each item included in the drop-down box is listed, including: none, all, first five, first three, last three. Line 24 of Figure 54A through line 11 of Figure 54C include the localization options which are comprised of a variety of different settings. Specifically, in one embodiment lines 25-31 of Figure 54A include a thumbnail width and a thumbnail height display values. In another embodiment the thumbnail width and thumbnail height values may be predetermined and stored in the MFD 1. In line 32, a page select tag lists the page range. In lines 33-35, a title of the thumbnail selection page is listed. In lines 4-12 of Figure 54B, the restore, cancel and submit buttons titles are included. Lines 13-21 correspond to the data in the page information window 41 shown in Figure 45. In addition, in lines 22-24 of Figure 54B, the back button, shown in the single image preview window 40 when only one page is scanned in the job, is included. Also included in lines 25 through line 10 of Figure 54C are the previous, next, rotate forward, rotate back, zoom-in, zoom-out and delete button titles that are illustrated in Figure 45.

[0261] Figures 55A and 55B illustrate the process between the MFD 1 and the GlobalScan server 5 once the scan job, with image preview button 21a selected, is completed. In step 300, the scan job is finished and an upload.xml is created which sends the scan job including all scanned pages or images and embeds the image preview settings and image capabilities and sends this information to the server 5 using an APIThumbInit.aspx page in step 302. In step 304, the GlobalScan server 5 generates thumbnails based on the thumbnail size specification set inside the upload.xml. Exactly what pages are to be thumbnailed and how many will be sent are defined inside the upload.xml file. In one embodiment of the invention, the server 5 is able to automatically detect blank pages and will not include these blank pages in the return xml. Alternatively, the server 5 may not automatically remove blank pages from the scan job. In addition, in one embodiment of the invention when the GlobalScan server 5 receives the scan job, the scan job is stored in a temporary location on the server 5. Alternatively, in another embodiment the scan job is stored in a more permanent storage location on the server 5, such as on a memory card or on a hard drive.

[0262] The GlobalScan server 5 then returns an xml file containing the requested thumbnail bitmaps in step 306. In step 308, the MFD 1 displays the thumbnails. In step 310, the user chooses a thumbnail and in step 312 the preview function is automatically performed since the user has selected a thumbnail in step 310. As soon as the preview function is selected, an APISessionMgr.aspx?action = preview&action_input = op = get 314 call is sent to the GlobalScan server 5. In step 316, the server 5 creates and sends back an image at the zoom and crop level requested. This data is returned via xml in step 318.

[0263] When the user performs a rotation action in step 320, the APISessionMgr.aspx function is called again but this time with the settings ?action = preview&action_input = op = rotate in step 322. When the GlobalScan server 5 receives the function call in step 324, the server 5 sends back a rotated image at the zoom and crop level requested via xml in step 326.

[0264] When the user performs the delete action in step 328, the APISessionMgr.aspx is called with the settings op = delete in step 330. When the GlobalScan server 5 receives this command, the server 5 sends back, in step 332, an acknowledgement with the delete logically

performed to allow for undo. This acknowledgement is returned via xml in step 334.

[0265] When the user performs the pan function in step 336, the APISessionMgr.aspx with settings op = pan is called in step 338. When the global scan server 5 receives this command in step 340, the server 5 sends back a crop result of the pan via the xml in step 342.

[0266] When user performs the zoom function in step 344, the APISessionMgr.aspx with settings op = zoom is called in 346. The server 5 then sends back the crop result of the zoom in 348, via xml in 350.

[0267] When the user performs the action accept in step 352, the APISessionMgr.aspx with settings op = accept is called in step 354. The server 5 then sends back a response of a job ID in 356 via xml in 358.

[0268] Finally, when the user performs the cancel action in step 360, the APISessionMgr.aspx with settings op = cancel is called in 362. In step 364, the server 5 sends an acknowledgement of the cancel via xml in 366.

[0269] Figures 53-55 describe the GetAvailableServices function, the ApiThumbInit function and the APISessionMgr function all of which operate via interactions between the MFD 1 and the server 5. Figure 56 continues this description. Thus, in step 400 of Figure 56, the MFD 1 calls the GetAvailableServices function using an APIGetAvailableServices.aspx page in step 402. The aspx page called in step 402 also may include additional command line settings such as profile ID, project ID, language, machine serial number or manufacturing version. This data is sent to the GlobalScan server 5, where the server 5 generates a services.xml file that is of the normal structure as it includes data regarding services to be available on the MFD 1. However, the generated service.xml file also includes a new service ID that is created for the preview tab 13. This is accomplished in step 404 where, in addition to the above data, the server 5 embeds all localization information for the image preview in the xml file. The xml file is then returned to the MFD 1 in step 406.

[0270] In step 408, the MFD 1 calls the ApiThumbInit.aspx function. Using this active server page, the MFD 1 sends a scanned image in a modified upload.xml file in 410. The server 5 then, in step 412, parses the upload.xml file and generates appropriate thumbnails from the image data found inside the upload.xml file. The server 5 then returns the images to the MFD 1 in an encoded fashion via xml in 414. The xml file sent to the MFD 1 also contains some control information such as whether or not there are only thumbnails included in the xml file and how many images are being returned.

[0271] In step 416, the MFD 1 calls the APISessionMgr.aspx active server page. All options for this page are passed via a query string i.e., the get method of http in step 418. In step 420, the server 5 parses the query string mainly branching on the op variable. Depending on the op variable's value, other options may also be passed. In step 422, the return.xml file, which contains images and control information based on the op variable value, is sent to the MFD 1.

[0272] Figure 57 is an example of the operation and flow of communications of a system in which a user, the MFD 1 and the GlobalScan server 5 interact. Beginning in step 450, the preview button 20a is pressed by the user on the operation panel of the MFD 1 or via some other method such as via a handheld device which connects to the MFD 1 via a wireless connection. As was noted earlier, when the preview button 20a is pressed, the button 20a is selected and highlighted as is shown in Figure 42B. After the preview button 20a is highlighted, when the user selects the start button in step 452, the MFD 1 begins scanning the input images or pages. Once the scan is finished in step 454, the MFD 1 sends an initial preview image file request to the GlobalScan server 5 in step 456. The GlobalScan server 5 then returns thumbnail image files in step 458 which are shown by the MFD 1 in step 460 to the user. When the user selects a thumbnail in step 462, the MFD 1 sends a get preview bitmap image request in step 464 to the GlobalScan server 5.

[0273] The GlobalScan server 5 returns a preview image in step 466 and in step 468, the preview image is shown to the user. In step 470, when the user selects a preview operation, the MFD 1 forwards a preview operation request in step 472 to the GlobalScan server 5. The GlobalScan server 5 returns a bitmap file which has had the selected operation performed on it in 474. In step 476, the bitmap file is shown to the user. This process is repeated until the user is finished performing operations of the preview images. The user, in step 478, then confirms or cancels the change or changes and this confirmation or cancellation request is then sent to the GlobalScan server 5 in step 480. In step 482, the GlobalScan server 5 sends a job finish command to the MFD 1.

[0274] Steps 470-476 will now be described in detail in Figures 58-64. However, before turning to the modifying functions in Figure 59-64, an initialization function will first be described with reference to Figures 58A and 58B. The initialized function initializes the thumbnail images and establishes session variables that will apply to all image preview functions for the current session. After a scan job is performed for which the image preview is enabled by the user, the initialization operation is performed. This operation notifies the server 5 and transfers the scanned TIFF, JPEG, etc images to the server 5 by embedding them in an upload xml file. After the server 5 processes the request, the MFD 1 receives from the server 5 the initial thumbnail image files which are displayed on the thumbnail list window shown in Figure 43A. The data is posted to the server 5 via an upload xml file using http post and the data is returned from the server 5 in thumbnail bitmap files used for display on the thumbnail window.

[0275] In Figure 58A, beginning in step 500 when the MFD 1 finishes a scan, an initialization request is sent in an upload.xml file in 502 to the GlobalScan server 5. The xml request includes the selection the user made on the

preview window, a MFD's thumbnail size and a batch size. In step 504, the GlobalScan server 5 responds with xml data containing a thumbnail image and a job ID.

[0276] Figure 58B shows an example of the response xml data returned from the GlobalScan server 5. In line 1, a root tag begins the xml file. An error_code tag in line 2 includes any error codes that might be generated and in line 3, an error_description tag includes a description of any error codes that might have been generated. In line 7, a total page number includes the total number of images or pages scanned in the batch job. In line 8, a page number tag includes the number of pages selected for the operation. Lines 10-13 and lines 14-17 include examples of thumbnails with ID numbers of 1 and 2 respectively. Also included is a data tag and inside the data tag is a page type tag.

[0277] The page type tag is used to store further information about the scanned pages. For example, page type tag includes information that states whether the scanned page or image was originally on A4 landscape or an A4 portrait type paper. The server is able to determine the paper type from the scanned image and retrieves the information necessary to make this determination from a database included on the server. The page type values can be, for example, any of the following: paper auto-detect, paper 8 x 11 portrait, paper 8 x 11 landscape, paper 8 x 17 landscape, paper 8 x 14 landscape, paper 8 x 13 landscape, paper 5 x 8 portrait, paper 5 x 8 landscape, paper 8 x 3 landscape, paper 8 x 4 portrait, paper 8 x 4 landscape, paper 8 x 5 portrait, paper 8 x 5 landscape, paper B4 landscape, paper B5 portrait, paper B5 landscape or any other type of paper.

[0278] The information that is included in a bmpdata tag found on lines 11 and 15 is base 64 encoded data which is comprised of image data. The image ID is found in lines 10 and 14 inside the data tag and is used as a unique identifier for the image. It should be noted that, in one embodiment of the invention, the image ID is not required to be numeric value; examples of the image ID are I1, I2 or I3.

[0279] Returning to Figure 58A, in step 506, the thumbnail image is shown in the thumbnail selection window 34. In order to reduce network traffic, the thumbnail images are implemented in batches which are based on the thumbnail size established by the MFD 1 in the initialization request sent in step 502. The APIThumbInit.aspx returns the first page thumbnail images from the server 5. In order to receive the rest of the thumbnail images from the server 5 (as all thumbnails may not fit on the first page) another protocol is used to retrieve the remainder of the thumbnails which uses the APIThumb.aspx?op=getset&batch_number=b command and which retrieves the set using the batch number. The batch number signifies the number of thumbnails to be displayed. For example, if the number of available thumbnails is 50 and the batch size is 10 with a batch number of 3, the function may return 20 to 30 images, depending on how many images are remaining and how many thumbnails were able to be displayed originally.

[0280] Figures 59A and 59B begin the first of the modification type functions. A modification type function is a function which modifies image data. The rotate function is the first modification type function described. After the user presses the rotate button on the preview image detail window 60, the preview image in the preview window 51 either rotates forward 46 or rotates back 47. In order to bring about this action, the rotation request may be sent to the server 5. After the server processes this request, a rotated preview page and thumbnail will be sent from the server 5 to the MFD 1 where it is displayed.

[0281] According to one embodiment, the originally scanned TIFF or JPEG file is not be modified at this moment but a recording of the fact that the image was rotated is saved on the server 5. Later, when the user chooses to accept the changes made in the preview, the actual JPEG/TIFF file is rotated, although other implementations are possible. Each time the user presses the rotate button, the image in the preview window 51 is rotated 90 degrees. The rotation algorithm rotates the image from the center of the LCD screen. Accordingly, while after submission the original image is merely rotated, the preview of the rotated image is rotated and zoomed, such that the location of the original pixel at the center of the port still becomes the same pixel at the center of the rotated image. However, different reference points for rotation can be used.

[0282] In Figure 59A, when the user presses the rotate button in step 510, a rotate request 512, made on the MFD 1, is sent to the GlobalScan server 5 in an upload.xml file. Included in the rotate request 512, are several different settings. An image number, degrees of rotation, a request image setting, a zoom setting and a follow me setting, explained below, are all possible settings that can be included in the upload.xml file. The image number setting corresponds to the image identifier generated during the initialization call. For example, if the user wants to rotate image 1, the ID of the image is sent along with the request. The degrees specify the number of degrees to rotate the referenced image. For example, the degrees setting could be 90, 180 or 270 if the user requested rotate forward one, two or three times, respectively. The follow me setting is a flag that determines a reference point for rotation. For example, a follow me value of zero could equal rotate with reference to the center of the original image. A follow me value of 1 could equal rotating with reference to the center of the view port. Finally the zoom setting determines the zoom percentage that the image is zoomed before returning to the MFD 1.

[0283] Returning now to Figure 59B, in step 514 the GlobalScan server 5 returns the xml data response that contains a preview image and job ID. In step 516, the MFD 1 shows preview images modified by the GlobalScan server 5 on the preview window 51 of the MFD 1 for the user to see.

[0284] Figure 59B shows, in lines 1-7, the data format

of the rotate request sent from the MFD 1 to the GlobalScan server 5. As noted earlier, the http request includes an APISessinMgr.aspx?action = preview&action_input=op=rotate request as well as several settings which are included in the query string. The possible settings included in the query string are image number, degrees, request image, zoom and follow me. Lines 10-25 show the contents of the xml data returned from the server 5 in step 514. The xml data includes a thumbnail only tag on line 16 which includes a Boolean value which is set to zero in this example. If the setting was set to 1, only a thumbnail would be included in the xml file. Lines 17-19 include the thumbnail of the image and lines 21-25 include the actual preview full sized image which includes width and height values as well as a zoom value. The width and height values are determined to be the most suitable size for the preview image to be shown on the MFD 1 within the view port of the MFD 1. It should also be noted that in one embodiment lines 17 and 23 both include image data encoded in Base64 format. In another embodiment, other encoding formats may be used, including secure formats, if the need were to arise.

[0285] Figures 60A and 60B illustrate an interaction between the MFD 1 and the GlobalScan server 5 when the user selects a Zoom in/out function by selecting the zoom in button 48 or the zoom out button 49. Each time the user selects the zoom in 48 or zoom out 49 buttons a zoom request will be sent to the server 5. The server 5 will then crop the preview image based on the current center of the viewport coordinates and return the image to the MFD 1 for display.

[0286] Figure 60A illustrates this process beginning with step 520 in which the zoom button is pressed by the user. Once the zoom button 48/49 is pressed, the MFD 1 forwards the zoom request and upload.xml to the server 5 in step 522. On receipt of the zoom request and upload.xml, the server 5 performs the zooming function and returns a response XML data containing the zoomed preview image and the Job ID in step 524. Finally, in step 526 the MFD 1 displays the zoomed preview image in the preview window 51.

[0287] Figure 60B illustrates the data posted to the server 5 in lines 1-8 and the xml data returned to the MFD 1 by the server 5 in lines 10-26. In step 522 of Figure 60A the zoom request and upload.xml data are sent to the server 5 using the APISessionMgr.aspx active server page. Appended to the APISessionMgr.aspx is the query string op=get. Additionally several options are also available to be passed in the query string, including a request_image, an image_no, a sizeX, a sizeY, a direction and a zoom option. Lines 3-8 illustrate these options. In line 7, the request image option is included. The request image (request_image) option indicates the type of image or images to return. When the option equals 0, this denotes no image. When the option equals 1, this denotes a thumbnail. When the option equals 2, this denotes a preview image. Finally, when the option equals 3, this denotes both thumbnail and preview images.

[0288] The image number (image_no) setting corresponds to the image identifier generated during the initialization call. The setting sizeX indicates the width of the viewport on the MFD 1 while the setting sizeY indicates the height of the viewport. The zoom setting specifies the zoom value desired. A zero value indicates the best fit size for the first time the preview image is shown. Different input paper may have different best-fit sizes and, as a result, zoom values for different types of input paper maybe different. Finally, the direction option shown on line 8 indicates a direction of any panning that was performed as a result of the zoom.

[0289] Lines 10-26 include an example of the XML file that is returned from the server 5 in step 524 of Figure 60A. In line 9, the root tag opens the xml file. Line 11 includes any error codes that might be generated and in line 12 the error_description tag includes a description of any error codes that might have been generated. In line 15 the server status is returned to the MFD 1. In line 16, the xml data includes a thumbnail only tag which includes a Boolean value which is set to zero in this example. If this setting was set to 1, this would indicate that only a thumbnail and no preview image would be included in the xml file. Lines 17-21 include the preview image data. In lines 17-18 the data tag includes a type, a preview, an id, a width and height, the page type (discussed above with respect to Figure 59B) and a zoom init value setting. Line 19 includes the preview image encoded in base64 format. Lines 22-25 include the thumbnail image and the data type and the thumbnail ID. Finally, in line 25 the xml file is closed with the close root tag.

[0290] Figures 61A and 61B illustrate an interaction between the MFD 1 and the GlobalScan server 5 when the user selects a pan left/right or pan up/down function by selecting the pan left/right buttons 43 or the pan up/down buttons 42. When the user zooms the preview image, the user may not be able to view the entire preview image. Thus since only part the zoomed preview image files is viewable, the preview image may need to be moved so that the user can see other parts of the zoomed image. The panning function provides the user the ability to see other parts of the image when the image is zoomed. It should be noted that in one embodiment of the invention this function is not performed on the actual image but is used only for benefit of a user using the viewfinder on the MFD 1 to preview an image. Alternatively, this function may be performed on the final image.

[0291] Figure 61A illustrates this process beginning with step 530 in which one of the pan buttons is pressed by the user. Once the MFD 1 receives a command from the user to pan the preview image, the pan request and an upload.xml file are uploaded to the GlobalScan server 5 in step 532. Once the GlobalScan server 5 receives the pan request and the upload.xml file, the GlobalScan server 5 processes the preview image and returns a response XML data including the preview image and a Job ID in step 534. Once the MFD 1 has received the panned preview image, the preview image is displayed on the

preview window 51 in step 536.

[0292] Figure 61B illustrates the data posted to the server 5 in lines 1-7 and the xml data returned to the MFD 1 by the server 5 in lines 9-21. In step 532 of Figure 61A, the pan request and upload.xml data are sent to the server 5 using APISessionMgr.aspx active server page.

[0293] Appended to the APISessionMgr.aspx is the query string op=get. Additionally several options are also available to be passed in the query string, including request_image, image_no, sizeX, sizeY, zoom and direction. It should be noted that the pan function includes all the same settings as the zoom function as well as the added setting of direction. Lines 3-8 illustrate these options. In line 7, the request image option is included. The request image option indicates the type of image or images to return. When the option equals 0, this denotes no image. When the option equals 1, this denotes a thumbnail. When the option equals 2, this denotes a preview image. Finally, when the option equals 3, this denotes both thumbnail and preview images. The image number setting corresponds to the image identifier generated during the initialization call. The setting sizeX indicates the width of the viewport on the MFD 1 while the setting sizeY indicates the height of the viewport. The zoom setting, shown in line 4, specifies the zoom value desired. A zero value indicates the best fit size for the first time the preview image is shown. Different input paper may have different best-fit sizes and, as a result, zoom values for different types of input paper may be different. Finally, a direction setting found on line 8 informs the server 5 of the direction in which the user has chosen to pan the preview image. The value of 0 indicates no panning, a value of 1 indicates pan up, a value of 2 indicates pan down, a value of 3 indicates pan left and a value of 4 indicates pan right.

[0294] Lines 10-22 include an example of the XML file that is returned from the server 5 in step 534 of Figure 61A. In line 9, the root tag opens the xml file. Line 11 includes any error codes that might be generated and in line 12 the error_description tag includes a description of any error codes that might have been generated. In lines 15 the server status is returned to the MFD 1. In line 16, the xml data includes a thumbnail only tag which includes a Boolean value which is set to zero in this example. If the setting is set to 1, this indicates that only a thumbnail and no preview image are included in the xml file. Lines 17-21 include the preview image data. In lines 17-18 the data tag includes type, preview, id, width and height, page type (discussed above with respect to Figure 59B) and zoom init value settings. Line 19 includes the preview image itself encoded in base64. Finally, in line 22 the xml file is closed with the close root tag.

[0295] Figures 62A and 62B illustrate an interaction between the MFD 1 and the GlobalScan server 5 when the user selects the delete button 50. When a user previews a page and finds that this page is not needed, the delete function can be used to remove the currently previewed page. However, selecting the delete button does not affect the image ID. When the delete button is pressed in the preview window, the MFD 1 will send a delete request to the server. After the server process the delete request, the current image will be marked for deletion in the image mapping file and the next image in line after the current image will be sent from the server. As noted earlier, if the current image is the last image in the set, the current image's predecessor will be sent from the server 5.

[0296] Figure 62A illustrates this process beginning with step 540 in which the delete button is pressed by the user. Once the MFD 1 receives a command from the user to delete the preview image, the delete request and an upload.xml file are uploaded to the GlobalScan server 5 in step 542. Once the GlobalScan server 5 recieves the delete request and the upload.xml file, the GlobalScan server 5 marks the scanned image for deletion and returns a response XML data including the next preview image and a Job ID in step 544. Once the MFD 1 has received the next preview image, the next preview image is displayed on the preview window 51 in step 546.

[0297] Figure 62B illustrates the data posted to the server 5 in lines 1-4 and the xml data returned to the MFD 1 by the server 5 in lines 6-18. In step 542 of Figure 62A the delete request and upload.xml data are sent to the server 5 using APISessionMGR.aspx active server page.

[0298] Appended to the APISessionMGR.aspx is the query string op=delete as well as the setting image_no. The image number (image_no) setting indicates which image number will be deleted. The image number corresponds to the image identifier generated during the initialization call.

[0299] Lines 6-18 include an example of the XML file that is returned from the server 5 in step 544 of Figure 62A. In line 6, the root tag opens the xml file. Line 7 includes any error codes that might be generated and in line 8 the error_description tag includes a description of any error codes that might have been generated. In line 11 the server status is returned to the MFD. In line 12, the xml data includes a thumbnail only tag which includes a Boolean value which is set to zero in this example. If the setting was set to 1, this would indicate that only a thumbnail and no preview image would be included in the xml file. Lines 13-17 include the preview image data. In lines 13-14 the data tag includes the type, preview, the id, the width and height, the page type (discussed above with respect to Figure 59B) and zoom init value settings. Line 15 includes the next preview image itself encoded in base64. Finally, in line 18 the xml file is closed with the close root tag.

[0300] Figures 63A and 63B illustrate an interaction between the MFD 1 and the GlobalScan server 5 when the user selects the previous button 44 or the next button 45. In the preview window 51 the customer may only be able to see one page, however the user may want to see the current page's previous and next pages. Accordingly, the previous and next functions enable to user to slide between pages without returning to the thumbnail selec-

tion window 34. Each time the previous 44 or next 45 button is selected, a get request is sent to the server 5 from the MFD 1.

[0301] Figure 63A illustrates this process beginning with step 550 in which the next/prev button is pressed by the user. Once the MFD 1 receives a command from the user to go to the next or previous page, a get request and an upload.xml file are uploaded to the GlobalScan server 5 in step 552. Once the GlobalScan server 5 recieves the get request and the upload.xml file, the GlobalScan server 5 then returns a response XML data including the next or previous preview image and a Job ID in step 554. Once the MFD 1 has received the next preview image, the next preview image is displayed on the preview window 51 in step 556. It should be noted that in one embodiment of the invention if the user is on the last image in a set, the next button returns the first image in the set. Alternatively, in another embodiment the system indicates that there is no previous or next image.

[0302] Figure 63B illustrates the data posted to the server 5 in lines 1-8 and the xml data returned to the MFD 1 by the server 5 in lines 10-26. In step 552 of Figure 63A, the get request and upload.xml data are sent to the server 5 using the APISessionMGR.aspx active server page.

[0303] Appended to the APISessionMGR.aspx is the query string op=get as well as a number of settings. The several options that are available to be passed in the query string include request_image, image_no, sizeX, sizeY, direction and zoom. Lines 3-8 illustrate these options. In line 7, the request image option is included. The request image option indicates the type of image or images to return. When the option equals 0, this denotes no image. When the option equals 1, this denotes a thumbnail. When the option equals 2, this denotes a preview image. Finally, when the option equals 3, this denotes both thumbnail and preview images.

[0304] The image number setting, found in line 3, corresponds to the image identifier generated during the initialization call. The setting sizeX indicates the width of the viewport on the MFD 1 while the setting sizeY indicates the height of the viewport. The direction setting indicates a direction of any pan that may need to be performed. Finally the zoom setting, found on line 4, specifies the zoom value desired. A zero value indicates the best fit size for the first time the preview image is shown. Different input paper may have different best-fit sizes and, as a result, zoom values for different types of input paper maybe different.

[0305] Lines 10-26 include an example of the XML file that is returned from the server 5 in step 554 of Figure 63A. In line 10, the root tag opens the xml file. Line 11 includes any error codes that might be generated and in line 12 the error_description tag includes a description of any error codes that might have been generated. In line 15 the server status is returned to the MFD 1. In line 16, the xml data includes a thumbnail only tag which includes a Boolean value which is set to zero in this example. If the setting is set to 1, this indicates that only thumbnails and no preview image would be included in the xml file. Lines 17-21 include the preview image data. In lines 17-18 the data tag includes type, preview, id, width and height, page type (discussed above with respect to Figure 59B) and zoom init value settings. Line 19 includes the next or previous preview image encoded in base64. Lines 22-25 include the thumbnail data corresponding to the preview image included in line 19. In addition, the thumbnail image data is found in line 23. Finally, in line 26 the xml file is closed with the close root tag.

[0306] Figures 64A and 64B illustrate an interaction between the MFD 1 and the GlobalScan server 5 when the user selects the restore 31, cancel 32 and submit 33 buttons.

[0307] When the submit button 33 is selected, a set function is called by the MFD 1. The set function is used to confirm the changes that have been made in the preview window. Once the user has made all the changes to the scanned image set the user needs to confirm the changes. When the submit button is selected, the set function is called and the server 5 makes all the changes to the original image.

[0308] Figure 64A illustrates this process beginning with step 560 in which the submit button 33 is pressed by the user. Once the MFD 1 receives a command from the user to submit the changes, the set request and an upload.xml file are uploaded to the GlobalScan server 5 in step 562. Once the GlobalScan server 5 recieves the set request and the upload.xml file, the GlobalScan server 5 then returns response XML data including a Job ID in step 564. Once the MFD 1 has received the xml data, the preview window is closed in step 566.

[0309] Figure 64B illustrates the code used by the server 5 and the MFD 1 after the submit 33, cancel 32 and restore 31 buttons are selected.

[0310] With respect to the submit button 33, Figure 64B illustrates the data posted to the server 5 in line 1 and the xml data returned to the MFD 1 by the server 5 in lines 1-8. In step 562 of Figure 64A the set request and upload.xml data are sent to the server 5 using the APISessionMgr.aspx active server page. Appended to the APISessionMgr.aspx is the query string op=set.

[0311] Although not shown in Figure 64B an example of the XML file that is returned from the server 5 in step 564 of Figure 64A is described below. First, the root tag opens the xml file. Next is included any error codes that might be generated and an error_description tag includes a description of any error codes that might have been generated. Next the server status is returned to the MFD 1, the server status tag is used to communicate to the MFD 1 that the set operation is complete. Finally, the close root tag closes the xml file.

[0312] With respect to the cancel button 32, Figure 64B illustrates the data posted to the server 5 in lines 4-5 when the cancel button is selected. When the user selects the cancel button, the preview operation is cancelled and the user loses all the scanned image files. The

operation is called using the APISessionMgr.aspx active server page with the appended query string op=cancel. When the server 5 receives this command, the job is erased and the server returns an XML file which includes the server status similar to the submit operation described previously.

[0313] With respect to the restore button 31, Figure 64B illustrates the data posted to the server 5 in lines 7-8 and the xml data returned to the MFD 1 by the server 5 in lines 10-27. When the user selects the restore button, an undo operation is initialized that undoes all the preview operations previously performed and returns the scan job back to the original initialized status. The operation is called using the APISessionMgr.aspx active server page with the appended query string op=restore.

[0314] Lines 10-37 include an example of the XML file that is returned from the server 5 in response to the restore call. The restore call places the MFD 1 in the same position as the MFD 1 was at the time of the original initialization command.

[0315] In line 10, the root tag opens the xml file. Line 11 includes any error codes that might be generated and in line 12 the error_description tag includes a description of any error codes that might have been generated. In line 15 the server status is returned to the MFD 1. In line 16, the total page number includes the total number of images or pages scanned in the batch job. In line 17, the page_number tag includes the number of pages selected for the operation. In line 18, the xml data includes a thumbnail only tag which includes a Boolean value which is set to 1 in this example. If the setting was set to 0, this would indicate that no preview image would be included in the xml file.

[0316] Lines 19-22 and lines 23-26 include examples of thumbnails with ID numbers of 1 and 2 respectively. Also included inside the data tag is the page type tag (discussed above with respect to Figure 59B) In addition, the thumbnail image data is found in lines 20 and 24. Finally, in line 27 the xml file is closed with the close root tag.

[0317] In addition, in another embodiment of the present invention, a user is able to undo a single operation instead of every operation performed by the user if the operation is a mistake or the user decides that the change operation is not needed.

[0318] Further, in another embodiment of the present invention the user is able to insert images in the scan job if during preview the user realizes an image or page is missing. These images may be inserted by scanning an input document or using an image previously stored on the GlobalScan server 5, the MFD 1 or some other networked storage device.

[0319] In another embodiment of the present invention, the preview page range dropdown box 19a may include an option entitled auto-detect. This option allows the MFD 1 or server 5 to detect which pages should be available for image preview. This determination can be made based on percentage of stray copy marks or if the majority of the job has one orientation while a few pages have a different orientation. In addition, the auto-detect could detect the presence of water-marks or bar-codes that may cause a problem to the user.

[0320] Fig.65 shows an example of a hardware configuration of the MFD 1 according to an embodiment of the present invention. As shown in Fig.65, the MFD 1 includes a controller board 600, an operation panel 610, a fax control unit (FCU) 620, a USB 630, an IEEE 1394 port 640, and a printer 650. It should be also noted that other types of I/O interfaces could be included including IEEE 1394b, USB 2.0. The controller board 600 includes a CPU 602 for processing and several storage devices such as SDRAM 603, SRAM 608, flash memory (flash ROM) 604, flash card interface part 606 and HD 605 used to store data associated with the MFD 1. Each of these components are connected to the ASIC 601, the ASIC 601 is an application specific integrated circuit that is designed specifically for use in a MFP 699. Other types of storage devices are also possible as well as other types of data processors and integrated circuits. The operation panel 610 is directly connected to the ASIC 601 as is the communications interface 620. The communications interface 620 can also be connected to a network or any other similar type communications medium. The USB 630, the IEEE 1394 640 and the multi-function printer functions 650 such as scanning, printing, and faxing are connected to the ASIC 601 via the PCI bus 680.

[0321] The SRAM 608 is a nonvolatile RAM; other types of SRAM are also possible. A flashcard 607 can be inserted into a flash card interface part 606, so that data is sent/received between the ASIC 601 and the flashcard 607 via the flash card interface part 606.

[0322] The operation panel 610 includes an operation part used for key operation such as key input and button pushing and the like by the user, and a display part for displaying drawing data such as various screens. It should be appreciated that other types of hardware components can be used in the present invention.

[0323] Further with respect to a computer readable medium such as a floppy disk, magnetic tape, CD-ROM and the like, by installing the program stored in the computer readable medium into an MFD, the MFD can perform the functions of the present invention.

[0324] In addition, several different types of bitmap image can be used in the present invention including, but not limited to, 24-bit bitmap, 256 color bitmap, 16 color bitmap and monochrome bitmap. These types of bitmaps are smaller and do not incur the network overhead that other types of image files might incur.

[0325] This invention has been described with respect to a multifunction device such as is described above with respect to Fig.65, but is applicable to any image processing device such as a copier, digital copier, printer, scanner, fax machine, or multi-function printer or any combination thereof. Moreover, the invention is applicable to other special purpose devices such as navigation systems, global positioning systems, vending machines,

metering systems, machine tools and other tools which operate using programming or a programmed processor, automobiles, other transportation devices such as trains, motorcycles, planes, or boats, radar systems, radios, MP3 players, digital music players, and other audio systems, mobile phones, other communication devices and systems, and any other special purpose device which operates using a plug-in.

[0326] The present invention is not limited to the specifically disclosed embodiments, and variations and modifications may be made without departing from the scope of the present invention. Obviously, numerous modifications and variations of the present invention are possible in light of the above teachings. It is therefore to be understood that within the scope of the appended claims, the invention may be practiced otherwise than as specifically described herein.

[SYSTEM AND METHOD FOR AUTHENTICATING A USER OF AN IMAGE PROCESSING SYSTEM]

[0327] Referring now to Figures 66-76 of the drawings, wherein like reference numerals designate identical or corresponding parts throughout the several views, Figure 66 is a block diagram of a system 5 for managing documents according to the present invention, and in particular to allow a document manager server 40 to manage documents and files by processing information related to applications, which can be grouped in different groups I-III. The system 5 includes a network 100 that interconnects at least one, but preferably a plurality of image processing devices which may be implemented as multifunction devices (MFDs) 10-30, to a document manager server 40. The network 100 preferably uses TCP/IP (Transmission Control Protocol/Internet Protocol), but any other desirable network protocol such as, for example IPX/SPX (Internetwork Packet Exchange/Sequential Packet Exchange), NetBEUI (NetBIOS Extended User Interface), or NetBIOS (Network Basic Input/Output System) is possible. The network 100 can be a local area network, a wide area network, any type of network such as an intranet, an extranet, the Internet or a combination thereof. Other communications links for the network 100, such as a virtual private network, or a wireless link, or any other suitable substitute may be used as well.

[0328] As shown in Figure 66, the devices 10, 20, 30 can be multi-function devices, or "MFDs." An MFD may incorporate or be any one of a plurality of a scanner, a copy machine, a printer, a fax machine, other office devices, and combinations thereof. Any one or combinations of these devices are referred to as a MFD, generally. Various types of MFDs are commonly known in the art and share common features and hardware with the MFDs of the present invention. Such an MFD combines digital imaging and Internet capabilities so that one can capture still images, sounds or videos and share such multimedia using wired or wireless connections from various locations. The MFD can create web pages, send and receive e-mails with attachments, edit images, FTP files, surf the Internet, and send or receive a fax. In another embodiment, the MFD is one of a combination of a scanner, photocopier and printer.

[0329] The MFD also includes or is connected to a user authentication device 15, 25, 35 which is configured to accept information input via a keypad, from an electronic card or memory, and/or a biometric device configured to sense biometric information input by a user. Examples of suitable biometric devices include, but are not limited to, retinal scanners, fingerprint readers, voice scanners or any other type of biometric reader device. Other devices used to authenticate users may include a proximity scanner, automatic tollbooth payment devices, cell phones, etc. More generally, the authentication devices may be any suitable device which is capable of identifying a user for the purposes of performing user authentication at the monitoring system, MFD, document manager server or backend application.

[0330] It should be noted that while the term "smartcard" may be used in the application, this term refers to any type of card or memory device for storing user information and capable of being read by an electronic device. Also, the card and the device used to read the card may be a scan sensor used to read directly from the card, or alternatively a proximity sensor configured to read data from the device without physically making contact with the card.

[0331] As depicted at Figure 72, the user authentication devices may be located within or near the image processing device, and may, or may not be, in communication with the image processing device. In one embodiment, the authentication devices may communicate directly with the MFD to perform user authentication. In another embodiment, the authentication device may be connected to a network to send received authentication information to any one of a number of systems connected to the network, which are capable of performing user authentication. In such a configuration, the user's credentials are transmitted from the authentication device to, for example, the monitoring system 45 or document manager server 40. The MFD is then informed of the result of the authentication, thereby permitting or denying the user access to the MFD. The authentication result may be sent to the MFD directly from the monitoring system 45, or may be forwarded from the authenticating device after it receives an authentication result. As will be discussed in greater detail below, authentication at the monitoring system 45 may be used by the document manager server 40, and the various backend applications connected thereto to perform a single sign-on authentication procedure.

[0332] Each of the MFDs and user authentication devices may be connected by any suitable type of wired or wireless connection to transfer information. Moreover, the communications between the monitoring system 45 and the authentication devices may be similar to the connection between the MFD and the document manager

via network 100, discussed above. Regardless, any other connection between the authentication devices and the monitoring system 45 suitable for the exchange of user credentials and authentication results may be employed.

[0333] In an exemplary embodiment the authentication devices are configured to interact with the Equitrac Office™ system. This system allows for a user to be authenticated with the Equitrac system by entering user authentication information at one of the authentication devices, which is then sent to an Equitrac Server (e.g., monitoring system 45). Once a user is authenticated with the Equitrac system, the system tracks copy, print and fax activities based on attributes such as document name, printer, port, date and time, paper size, finishing options and choice between black and white or color. Other types and brands of authentication devices and cost accounting systems may also be used with the invention.

[0334] As shown in Figure 66, the document manager server 40 is connected to a directory/address book server 60 (or "directory server" or "global directory"). The directory server 60 can include information such as the names, addresses, network addresses, e-mail addresses, phone/fax numbers, other types of destination information, and authorization of individuals. Other information can also be included in the directory server 60. Examples of directory servers 60 compatible with the present invention include, but are not limited to, Lotus Notes™, Microsoft Exchange™, and LDAP ("Lightweight Directory Access Protocol") enabled directory servers. LDAP is a software protocol that enables a user to perform network authentication, locate organizations, individuals, files, devices in a network. The directory server is configured to receive user information entered at the authentication device or image processing device and authenticate the user for the network.

[0335] The document manager server 40 can also be connected to a network domain controller 50 that controls authentication of the MFD user to a network. The network domain controller 50 is, for example, a server that responds to security authentication requests, such as logging in, within its domain. The network domain controller 50 may be backed up by one or more backup network domain controllers that can optionally also handle security authentication. Examples of a directory server 60 and a network domain controller 50 are disclosed in U.S. Application Serial No. 10/243,645, filed September 16, 2002.

[0336] Briefly, the system 5 provides access for the users of the MFDs 10-30 to the information stored at the directory server 60 via the document manager server 40 when the user is authenticated at the image processing device. The directory server 60 is capable of retrieving preference information related to the user's credentials and transmits this preference information to the MFDs 10-30. This preference information may include information relating to scan settings, such as resolution, density, scan mode, color, paper size, file format, or any additional settings that can be adjusted at the MFD. The preference information may also include information related to a destination of the processed image, including a specific e-mail address, a backend application, a middle processing system, or any other network application configured to accept the processed data. A middle processing system may include a file formation conversion system, optical character recognition, or any similarly suited system as will be described in greater detail below. Also, the preference information may include a software plug-in, which will be discussed in greater detail below, or any other information related to changing the functionality of the MFD. These plug-ins also allow the document manager server to access user credential information stored at the MFD following a successful authentication at the monitoring system 45, and subsequently the document manager server 40 as discussed below.

[0337] A user can also request a search of the company's global directory stored at the directory server 60. The document manager server 40 can pass the search request to the directory server 60 and can receive the search results (e.g., e-mail addresses and/or fax numbers) from the directory server 60. The document manager server 40 can pass the search results to the MFD 20, which can temporarily store and display them. The user can select a displayed result (e.g., an e-mail addresses or a fax number), scan a document, and request that the scanned document be transmitted, e-mailed and/or faxed to the selected destination.

[0338] The document manager server 40 can be configured to act as an intermediate agent, or a gateway between a plurality of network applications 45, 50, 60, 70, 80, and 90 and the MFDs. The applications 70, 80, and 90 can include for example an e-mail server, a fax server, a file format conversion system, an optical character recognition (OCR) system, a document management system and a file storage system or any combination of multiples thereof. The document management server 40 is capable of supporting a plurality of backend applications such as various document management systems, or file storage systems. In a preferred embodiment, the e-mail server is incorporated into the document manager server 40. An example of a document management system is disclosed in U.S. Application Serial No. 09/795,438, filed March 1, 2001; and in U.S. Application Serial No. 10/116,162, filed April 5, 2002.

[0339] The applications can be grouped, for example in Groups I-III. Group I can be a delivery system group including an e-mail server and a fax server; Group II can be a middle processing group including a file format conversion system and an optical character recognition system; and Group III can be a backend application group including a document management system and a file storage system. Groups I-III can include a plurality of devices from each category. For example, the document management server 40 can be connected to a plurality of applications from each group. The document manager server 40 can direct documents to several applications within each group. In a preferred embodiment, the doc-

ument manager server 40 delivers a document to several of the applications within the delivery system group, but delivers the document to one or a plurality of the application within the middle processing group and to one or a plurality of the applications of the backend application group. For example, the document manager server 40 can deliver a document to the e-mail and fax servers, to the OCR system, and to a document management system. Other combinations are possible in other embodiments.

[0340] In a preferred embodiment, the MFDs 10-30 and the document manager server 40 exchange data using the protocol HTTP ("Hypertext Transfer Protocol") or HTTPS (HTTP over Secure Socket Layer) over the network 100. Other protocols such as TCP/IP, IPX/SPX, NetBEUI, or NetBIOS, for example can equivalently be used with the present invention. Preferably, the MFDs 10-30 and the document manager server 40 exchange data using the format XML ("Extensible Markup Language"). Other formats, such as HTML, can equivalently be used with the present invention.

[0341] In one embodiment, the document manager server 40 can include an MFD profiler 280 (shown in Figure 67) that manages profiles for the MFDs 10-30. The administrator of the system 5 can create, change and maintain the profiles via a profile user interface on the document manager server 40. A profile includes information (e.g., parameters) sent from the document manager server 40 to an MFD. Based on this information, the MFD can adjust its user interface and functions so as to properly interface with the document manager server 40. The information may also include software plug-ins processed by the MFD to allow the operation of the MFD to be modified based on the existence or introduction of a backend application. The document manager server 40 includes software plug-ins corresponding to the backend applications connected to the document manager server 40. For example, the MFD can display selections allowing a user to select options (e.g., a particular delivery system, a middle processing system, or a backend application) available to the MFD via the document manager server 40. Information included in the profile can be the identity of the various applications 70-90 connected to the document manager server 40. The profiler 280 receives identification information from an MFD (e.g., the serial number) and uses this identification information to check whether the MFD is registered within a register, e.g., a data table stored in a memory of the document manager server 40. If registered, the profiler sends the MFD a profile assigned to the MFD. If the MFD is not registered, the profiler can register the MFD and send the MFD a profile. The profiler can store more than one profile. In a preferred embodiment, one profile is assigned to each MFD, and more than one MFD can share the same profile. While the term "software plug-in" has been used, any type of software, programming, or chip can be used to modify the operation of the MFD.

[0342] Examples of parameters in a profile include, but are not limited to:

a profile ID, which identifies the profile;
an LDAP Enabled parameter, which indicates whether or not the LDAP tree search is enabled on the document manager server 40 using the directory server 60;
a Base Domain Name (DN) parameter, which provides a default field of search for the LDAP tree when the LDAP search is enabled;
a Network Authentication parameter, which indicates whether or not network authentication is enabled using the network domain controller 40;
a Time-Out parameter, which indicates the time period that should elapse before the MFD resets and requires the user to enter login information;
a Max Result Count parameter, which determines the maximum number of LDAP query results returned;
a Fax Option parameter, which indicates whether or not a fax server is connected to the document manager server 40;
a Post Scan Processing parameter, which indicates what post scan processing system is connected to the document manager server 40, post scan processing systems may include, for example an e-mail server, a file format conversion system, an optical character recognition system, etc.;
a Backend parameter, indicating which backend applications are connected to the document manager server 40 and are able to be accessed by the MFD, such backend applications may include, a document management system or a file storage system, or another similar type of system; and
a Software Plug-in which contains and executable file allowing the image processing device to perform specific processing tasks (e.g., user authentication) related to one or a plurality of backend applications.

[0343] Other parameters can also be included in the profile. For example, parameters reflecting specific user ID, default size of papers, scanning resolution setting, condition of the document feeder, department code for billing image processing operations, additional scanning job parameters for the specific user ID, or any additional parameters may be used.

[0344] The Backend parameter may also initiate an authentication step to determine if a user has already logged into the network and been automatically authenticated to operate the backend application based on the network authentication. If the Backend parameter indicates that a software plug-in is required for the MFD device to properly interface with the backend application, then the MFD transmits data to the document manager server 40 requesting the receipt of a software plug-in.

[0345] In the context of the authentication procedure disclosed below with reference to Figures 70A-70B, the profile may be accessed when the document manager

receives a request from the MFD including identification information corresponding to the MFD. Alternatively, the profile may be accessed once the user is authenticated at the document manager server and associated backend applications. Regardless, the plug-ins corresponding to each backend application are used to form the login template discussed in Figures 70A-70B.

[0346] Figure 67 illustrates an MFD 20's browser 25 configured to exchange information between the MFD 20 and the document manager server 40 according to one embodiment of the present invention. An example of a browser 25 is disclosed in U.S. Application Serial No. 10/243,643, filed September 16, 2002. Further details of the browser 25 are set forth below. Figure 67 shows the software components of the document manager server 40, which includes an authentication device 260 configured to perform the authentication functions discussed below. The document manager server 40 also includes an administration device 265 which allows the system administrator to administer the system 5. For example, the administrator of the system can access the profiler 280 via the administration device 265 to set user profiles and/or the MFD profiles for the MFDs 10-30 connected to the document manager server 40. A system administrator may also access the administration device 265 to set the single sign-on feature disclosed below with reference to Figures 70A and 70B. A directory gateway 270 is also included within the document manager server 40 and is configured to communicate with the directory server 60. The document manager server 40 also includes a document router 275 configured to route the documents received from the MFDs to the appropriate applications 70, 80 and 90.

[0347] As shown in Figure 67, the MFD 20 includes an engine control service (ECS) 200 that controls, for example, the scanning engine of the MFD 20. A memory control service (MCS) 205 controls access to the memory of the MFD 20. This MCS 205 also stores a user credentials used to log into the monitoring system 45 or any other external authentication system. As discussed below, this user credential information may be accessed by the authentication device 260 and/or profiler 280 to perform authentication at the document manager server 40 and various backend applications connected to the document manager server.

[0348] An operation panel control service (OCS) 215 generates outputs which are displayed on the touch-panel type liquid crystal display (LCD) of the MFD 20. It should be noted that the display and user interface of the MFD 20 is not limited to an LCD display, but may also be any other suitable device, or combination of devices, such as but not limited to LCDs, light-emitting diode (LED) displays, cathode ray tube (CRT) displays, plasma displays, keypads, and/or keyboards.

[0349] A system control service (SCS) 225 controls and/or monitors sensors within the MFD 20. For example, the SCS 225 controls the touch screen sensors, paper jam sensors and scanning operation sensors. Accordingly, the SCS 225 can manage the status of the MFD 20 based on the information from the sensors.

[0350] A network control service (NCS) 220 controls communication between the browser 25 and the document manager server 40. Optionally, a secure socket layer (SSL) 230, in the form of a communication formatting device or routine, provides added security for communications between the NCS 220 and the browser 25.

[0351] A command input service (CIS) 240 processes input information, for example, from the LCD touch panel and/or a keypad of the MFD 20. A user of the MFD can enter information and commands using the LCD touch panel and the keypad. The CIS 240 can process such information and commands entered by a user (e.g., forwarded to the CIS 240 by the SCS 225). The CIS 240 can generate a command (e.g., a display command) based on such processing and transmit the command to other components of the MFD (e.g., to the OCS 215 to display a graphic on the LCD). The CIS 240 can also exchange information and commands with the NCS 220 for processing with the browser 25 in connection with the server 40.

[0352] Conventional MFDs include ECSs, MCSs, OCSs, NCSs, SCSs, and CISs which are firmware for implementing and controlling each hardware component of the MFD. In the present invention, however, the NCS 220 is configured to communicate with the browser 25. For instance, the NCS 220 has additional capabilities for communicating using the HTTP protocol. The NCS 220 is also configured to communicate with the server 40 so that the NCS 220 exchanges data between the browser 25 and the server 40. For example, The NCS 220 can exchange user information with the server 40 and receive a profile, can transmit a request for an e-mail address and can receive from the server 40 a selected e-mail address, or the NCS 220 can exchange user credential information with the monitoring system 45 and can receive authentication confirmation from the monitoring system 45 (and from the directory server 60) during an authentication process. The NCS 220 is also capable of receiving plug-in information from the document manager server 40 which is capable of performing the authentication procedure described below or altering the user interface of the MFDs.

[0353] The browser 25 includes an HTTP command processor 235 that communicates with the network control service (NCS) 220 of the MFD 20. For example, a request for an e-mail address entered by the user via the MFD keypad, or a request for displaying information on the LCD can be passed from the NCS 220 to the browser 25 by the HTTP command processor 235. The HTTP command processor 235 can exchange data in the HTML format with the browser's HTML parser 250, and can exchange data in the XML format with the XML parser 255. The parsers 250 and 255 can check the data from the HTTP command processor 235 for syntax and process the data for HTTP command processor 235. The present invention can include conventional parsers, which are

conventionally included as part of a compiler.

[0354] The HTTP command processor 235 can be provided with a program code, or software plug-in, for implementing a specific application, such as user authentication processing as discussed below. The HTTP command processor 235 can process information based on definitions of the specific application. For example, the HTTP command processor 235 can process information provided by the user, such as user credentials (e.g., username, password, biometric identification, etc.), and generate an HTTP request based on this processing for the server 40. The HTTP command processor 235 can transmit this HTTP request to the NCS 220 to be transmitted to the server 40. The HTTP command processor 235 can also receive plug-in information relating to specific backend application functionalities, or authentication processes necessary for gaining access to the document manager server 40 or a backend application connected to the document manager server. These plug-ins also allow users to add processing instructions, metadata, and other indexing information to the image file transmitted to the document manager server 40.

[0355] The HTTP command processor 235 can also process information received from the server 40 (via the NCS 220). For example, the HTTP command processor 235 can receive an HTTP response generated by the server 40 which includes a profile with parameters or software plug-ins for operating the MFD. These software plug-ins also indicate user credential information that may be necessary for a user to be authenticated at the document manager server 40, or any system connected to the document manager server. As noted above, and as discussed below in detail, a plug-in may also be associated with the authentication procedure performed for the monitoring system 45. This information may be obtained by the document manager server 40, and be used to fill in the user credential and authentication information needed for the plug-ins that require additional user authentication. The HTTP command processor 235 can process this information and generate commands to control the MFD in accordance with the information, e.g., can request the MFD to display a menu with the appropriate buttons, or to scan according to the scanning job parameters for the specific user ID. As another example, the HTTP command processor 235 can generate a graphic drawing command for the LCD panel. The HTTP command processor 235 can transmit the commands to the appropriate MFD firmware (e.g., the OCS 215) to be executed.

[0356] Figures 68 and 69 are flowcharts depicting exemplary steps performed while performing authentication at an external system, for example monitoring system 45 via any one of the authentication devices 15, 25, 35.

[0357] Specifically, the process depicted at Figure 68 illustrates a method of performing authentication at the monitoring system 45 using a card-type authentication device. At step 300 the user inserts a card into the authentication device. As depicted in Figure 72, the authentication device 15-35 may be located within the MFD 10-30 or it may be located externally to the MFD 10-30, and is configured to communicate with an external system, such as monitoring system 45. Other types of authentication devices can be used, such as an optical reader, and it is not necessary to actually insert a card into an authentication device, depending on the type and the design of the authentication device. The authentication device in this embodiment is, for example, a card reader and an interface which allows a user to enter a PIN number or other personal information associated with the user. Further, the monitoring system may perform the authentication process individually and communicate the result of the authentication with the MFD 10-30.

[0358] The user is prompted for a personal identification number (PIN) at step 305. This prompting can be done before, after, or simultaneous to the card reading. Alternatively, the user may be required to enter biometric information related to a physical attribute of the user. This may include reading the user's fingerprint, scanning a user's retina, sensing a user's voice, or performing a facial recognition on the user. This entered biometric information may then be transformed into a mathematical representation which is compared to a mathematical model of the user's specified biometric information stored in the card or at the monitoring system.

[0359] At step 310, the card information and associated PIN or biometric information (e.g. credentials) are sent to the monitoring system. The monitoring system includes a database that associates user credentials with specific users, allowing identification and authentication of a user based on the received credential information. At step 315, the received user credential information is mapped to a user stored in the database at the monitoring system. At step 320, the monitoring system determines if the received credential information corresponds to an authorized or unauthorized user. If, at step 325, the user is not found or is not authorized to access the MFD, then the monitoring system notifies the MFD to prevent the user from accessing the device. Alternatively, if the user is found to be authorized to access the system at step 320, the user's credentials, and optionally additional information related to the user, is cached at the monitoring system 45 to later be accessed by the document manager server 40. At step 335, the monitoring system 45 informs the MFD device that the user is authorized to access the system. At step 340, the MFD grants the user access to the device and unlocks the user interface allowing the user to access other systems via the document manager server.

[0360] One example of user credential information that may be stored at the card and correlated with a user at the database of the monitoring system is a digital signature. The monitoring system may determine the validity of the received digital signature and authorize or disable access to the MFD accordingly. This digital signature is then cached at the monitoring system so that is can easily

be accessed by a plug-in of the document manager server for subsequent authentication procedures and to retrieve a profile corresponding to the user.

[0361] Figure 69 depicts a process similar to that depicted in Figure 68, but is directed to performing user authentication using a biometric authentication device. As noted above, the authentication device 15-35 may be any one, or a combination of, a retinal scanner, fingerprint reader, voice scanner, or any other type of biometric device.

[0362] At step 400 the user presents biometric information to the authentication device, which detects the biometric information using any one of the above-noted biometric scanning/detecting mechanisms. Other types of authentication devices can be used, such as an optical reader, and it is not necessary to actually insert a card into an authentication device, depending on the type and the design of the authentication device. At step 405 the biometric information is detected and mapped to a mathematical equivalent, by conventional well known methods, before being transmitted to the monitoring system for identification and/or authentication. At step 410 the monitoring system 45 searches a database for a match to the received detected biometric information. If a match is not found at step 415, the monitoring system 45 informs the MFD that the user is not authorized to access the device. The MFD accordingly denies access to the user.

[0363] If, however, a match is found at step 415 the monitoring system determines if the user corresponding to the received biometric information is authorized to access the MFD. If the user is authorized, at step 425 the monitoring device retrieves the user's credentials from the database and stores (step 430) the credentials in a cache memory to be later accessed by a document manager server 40 for subsequent authentication. The credential information that is cached may only include the biometric information, but preferably also includes additional information stored in the database of the monitoring system that matches the received biometric information. This additional stored information may include the identity of a user, a username of a user, password, or other additional user information used by the document manager server 40 for authentication purposes. Step 435, which may be performed before or after the user credential information is cached (i.e., steps 425 and 430), includes informing the MFD that the user is authorized to access the MFD. The MFD may then allow the user to access the device by unlocking the user interface.

[0364] Figures 68 and 69, as discussed above, relate to performing user authentication using an authentication device and the external monitoring system. User credentials, either received from a user, or retrieved from a database of the monitoring system, are then cached at the monitoring system to allow for efficient access by an authentication procedure at the document manager server 40. Figures 70A and 70B illustrate a process in which the cached user credential information is accessed to allow for easier user authentication at the document manager server 40 and the various services connected to the document manager server.

[0365] The process begins with authentication at an external system such as the monitoring system 45, as noted above (i.e., step 500). At step 502 the user interface of the MFD is unlocked and an user is presented with an interface allowing the user to operate the MFD. The options presented on the user interface may include conventional processing functions of a MFD, as discussed above, and may also include options to login and access backend services via the document manager server.

[0366] Step 505 determines whether the user wishes to access the document manager server and related backend applications. If the user does not request access to services related to the document manager server 40, flow proceeds to step 510 and the user is permitted to access the MFD to perform conventional operations. The process then ends. It should be noted that the user may choose to access the document manager services at another time, at which point the process would pick-up at step 515. If step 505 determines that access is requesting flow proceeds to step 507, which sends a request to a document manager server. This request includes information identifying the requesting MFD, such as an IP address, MAC address, MFD serial number, or other similar identification information. Further, the request may identify the selected backend services for which the user of the MFD has requested access. The MFD may also access a profile based on the received information identifying the MFD, as discussed in detail above, to automatically determine which services correspond to the user profile identified as corresponding to the requesting MFD.

[0367] At step 515, the document manager determines, based on a user input at the MFD or the profile retrieved from the profiler 280, which services should be made available to the requesting MFD. At step 520, the document manager server determines the requested services that require authentication. If a service does not require authentication, the user is granted access to this service without the need for additional authentication. Alternatively, at step 525, if the service requires authentication, then a template is generated by the plug-in corresponding to this service indicating the user credentials needed for authentication. At step 530, the document manager server determines if any default credentials are available for authentication templates corresponding to each requested service. These default credentials may be part of the plug-in corresponding to the service, or may be filled in based on the above mentioned profile information corresponding to an identified MFD. When step 530 determines that there are default credentials available for service, flow proceeds to step 535 which fills in default credentials. The templates for each service may be unique and some may only need a username, while others may require a username and password, and others may need no user credentials whatsoever.

[0368] Once the authentication templates correspond-

ing to each service (e.g., plug-in) are completed to the extent that they can based on default login information and profile information corresponding to the requesting MFD, the monitoring system is accessed to retrieve any additional user credentials. At step 540, the document manager server secures the retrieved unique information corresponding to the requesting MFD, noted above (e.g., IP address, MAC address, machine serial number, etc.), and at step 545 uses this information to retrieve the additional user credential information cached at the monitoring system 45. The additional stored user credential information includes any credentials submitted to the monitoring system via the authentication device, as well as any additional user credential information corresponding to the user that is retrieved from the database of the monitoring system based on the received user credentials. Such information may include the identification of a user, a username, password, or any additional credentials that may be used to complete the authentication templates for each requested service. Since the user has already been authentication via the monitoring system 45, the document manager server considers this additional user credential data as trusted data and allows this data to be used to fill in any additional necessary user credentials.

[0369] At step 550, the additional user credential information retrieved from the monitoring system is used by the document manager server to complete the individual templates, and the document manager server generates a master template including the user credential information necessary to complete the login procedure. This master template is typically what would be used to generate a display at the MFD for user login. Step 555 determines whether the master template including all necessary user credentials is completed. If the template is not complete, a flag is set at step 560 indicating that additional user credentials are needed for the user to be authenticated to all requested services. If the master template is completed, the flag is not set. At step 565 the master template is sent to the MFD.

[0370] After receiving the master authentication template, at step 570 the MFD determines whether the template is flagged for display. If the template is flagged for display, a user interface is displayed on the MFD at step 580 and at step 585 the user is prompted to enter additional user credentials to complete the authentication process. At step 590, after entering the additional credentials, the user selects a displayed login button and at step 595 the additional user credentials are sent back to the document manager server for authentication. Alternatively, if no additional credentials are required by the user, step 595 sends the credentials are sent back to the document manager server for authentication. It should be noted that if the master template is complete, the document manager server may optionally not send the template back to the MFD, but instead simply perform user authentication and send data corresponding to the authenticated user's profile to the MFD, as discussed above. Further, even if the master template is complete, the MFD may display a login button forcing the user to submit the automatically filled in user credential information to the document manager server. Once the authentication procedure is complete, the document manager server would provide services according to the profile corresponding to the authenticated MFD and/or user as disclosed in U.S. Application Serial No. 11/092,831, filed March 30, 2005.

[0371] Figure 71 illustrates the steps performed during a logout operation. At step 600, a service connected to the document manager server (e.g., backend application), or the monitoring system 45, generates a logout request for the authenticated user and/or MFD. A logout request may be generated because the user's account has insufficient funds to continue the requested processing, the communication with the service has timed out, or for any other situation in which logout is desirable. At step 605, the document manager receives the logout request. Next, step 610 determines if logout request should be rejected based on the status of the MFD. Specifically, the logout request may be denied under the following exemplary conditions: when there are temporary communication threads running that are communicating with the document manager, when the user is accessing the interface of the MFD, when there is a scanning operation in progress, when there are pending jobs, etc. If a denial of the logout request is allowed, then the document manager has the option at step 615 of denying the logout request. If, however, step 610 determines that the logout request should not be rejected, flow proceeds to step 620 which ends communications with all services from the MFD. At step 625 the user and/or MFD is logged out from all services based on the request generated at the backend service or monitoring system 45.

[0372] The forced logout system coupled with the single sign-on capability allows the monitoring system 45 and document manager server 40 to perform coordinated user authentication and forced logout procedures at a system level.

[0373] Figure 72 illustrates an overview of the hardware used to implement the present invention. An authentication device 1205 is located in, at, or around the MFD 10-30. As stated previously, the authentication device 1205 may be located at a position outside of the MFD 10-30 and provide communications only to the MFD 10-30 when necessary. As previously stated, devices such as memory readers, proximity sensors, biometric sensors or any other desired device may be used as the authentication device. The authentication device 1205 and/or biometric sensing device 1200, the MFD 10-30 and the monitoring system 45 are in communication via a wireless or wired connection 100 using well know protocols and signal transmission techniques. It should be noted that the authentication device 1205 may also be implemented in conjunction with a biometrics device 1200 to provide multi-factor user authentication. The biometric detection device 1200 may include a mechanism

for detecting user characteristics such as fingerprints, a retinal scan, voice recognition, facial recognition component, or any other desired characteristic. This entered biometric information may be compared against a biometric parameter stored on the card itself, or with biometric data stored at the monitoring system 45. If the entered biometric information matches the biometric information stored in the card or the monitoring system 45 then the user is successfully granted access to the system. The interaction between these devices and the roles of each device has been described in detail above. Figure 72 also illustrates the document manager server 40, LDAP server 60 and network application server 70-90 which are described in greater detail below.

[0374] Figures 73-74 illustrate an example of the MFD 20, which includes a central processing unit (CPU) 1305, and various elements connected to the CPU 1305 by an internal bus 1310. The CPU 1305 services multiple tasks while monitoring the state of the MFD 20. The elements connected to the CPU 1305 include a read only memory (ROM) 1345, a random access memory (RAM) 1315, a hard disk drive (HDD) 1320, a floppy disk drive (FDD) 1350 capable of receiving a floppy disk 1355, a communication interface (I/F) 1330, and a modem unit 1360. In addition, a control panel 1375, a scanner unit 1370, a printer unit 1335, and an image processing device 1340 can be connected to the CPU 1305 by the bus 1310. Both the I/F 1330 and the modem unit 1360 are connected to a communication network 100.

[0375] In a preferred embodiment, the program code instructions for the MFD 20 are stored on the HDD 1320 via an IC card. Alternatively, the program code instructions can be stored on the floppy 1355 so that the program code instructions may be read by the FDD 1350, transferred to the RAM 1315 and executed by the CPU 1305 to carry out the instructions. These instructions can be the instructions to perform the MFD's functions described above. These instructions permit the MFD 20 to interact with the document manager server 40 via browser 25 and to control the control panel 1335 and the image processing units of the MFD 20.

[0376] During a start-up of the MFD 20, the program code instructions may be read by the CPU 1305, transferred to the RAM and executed by the CPU 1305. Alternatively, the program code instructions may be loaded to the ROM 1345. It is therefore understood that in the present invention any of the floppy disk 1355, the HHD 1330, the RAM 1315, and the ROM 1345 correspond to a computer readable storage medium capable of storing program code instructions. Other devices and medium that can store the instructions according to the present invention include for example magnetic disks, optical disks including DVDs, magneto-optical disks such as MOS, and semiconductor memory cards such as PC cards, compact flash cards, smart media, memory sticks, etc.

[0377] In a preferred embodiment, the control panel 1375 includes a user interface that displays information allowing the user of the MFD 20 to interact with the document manager server 40. The display screen can be a LCD, a plasma display device, or a cathode ray tube CRT display. The display screen does not have to be integral with, or embedded in, the control panel 1375, but may simply be coupled to the control panel 1375 by either a wire or a wireless connection. The control panel 1375 may include keys for inputting information or requesting various operations. Alternatively, the control panel 1375 and the display screen may be operated by a keyboard, a mouse, a remote control, touching the display screen, voice recognition, or eye-movement tracking, or a combination thereof.

[0378] Figure 75 is a block diagram of a server 40, 50, 60 or the server corresponding to the monitoring system 45 according to one embodiment of the present invention. Figure 76 is a schematic representation of the server. The server 40, 45, 50, 60 includes a central processing unit 101 (CPU) that communicates with a number of other devices by way of a system bus 150. The server 40, 45, 50, 60 includes a random access memory (RAM) 190 that hosts temporary storage values used in implementing the authenticating, routing and managing functions of documents.

[0379] A conventional personal computer or computer workstation with sufficient memory and processing capability may also be configured to operate as the server 40, 45. The central processing unit 101 is configured for high volume data transmission and performing a significant number of mathematical calculations in processing communications and database searches.

[0380] The ROM 180 is preferably included in a semiconductor form although other read-only memory forms including optical media may be used to host application software and temporary results. The ROM 180 connects to the system bus 150 for use by the CPU 101. The ROM 180 includes computer readable instructions that, when executed by the CPU 101, can perform the different authenticating, routing and managing functions discussed above associated with scanned documents from MFDs. An input controller 160 connects to the system bus 150 and provides an interface with peripheral equipment, including a keyboard 161 and a pointing device such as a mouse 162. The input controller 160 may include different ports such as a mouse port in the form of a PS2 port or, for example, a universal serial bus (USB) port. The keyboard port for the input controller 160 is in the form of a mini-DIN port although other connectors may be used as well. The input controller 160 provides sound card connections so that external jacks on the sound card allow users to attach microphone speakers or an external sound source. The input controller 160 also may include serial ports or parallel ports as well.

[0381] A disk controller 140 is in the form of an IDE controller and connects via ribbon cables to a floppy disk drive 141 as well as a hard disk drive 142, a CD-ROM drive 118 and a compact disk 119. In addition, a PCI expansion slot is provided on the disk controller 140 or

mother board that hosts the CPU 101. An enhanced graphic port expansion slot is provided and provides 3-D graphics with fast access to the main memory. The hard disk 121 may also include a CD-ROM that may be readable as well as writeable. A communication controller 130 provides a connection, for example by way of an Ethernet connection to a network 131, which can be the network 101. In one embodiment, the network 131 and the connection to the communication controller 130 are made by way of a plurality of connections including a cable-modem connection, DSL connection, dial-up modem connection, and the like that connect to the communication controller 130.

[0382] An input/output controller 120 also provides connections to external components such as an external hard disk 121, printer 122, which can be MFD 10-3, for example, by way of an RS 232 port, a SCSI bus, an Ethernet or other network connection which supports any desired network protocol such as, but not limited to TCP/IP, IPX, IPX/SPX, or NetBEUI.

[0383] A display controller 110 interconnects the system bus 150 to a display device, such as a cathode ray tube (CRT) 111. While a CRT is shown, a variety of other display devices may be used such as an LCD, or plasma display device.

[0384] The mechanisms and processes set forth in the present description may be implemented using a conventional general purpose microprocessor(s) programmed according to the teachings of the present specification, as will be appreciated to those skilled in the relevant arts. Appropriate software coding can readily be prepared by skilled programmers based on the teachings of the present disclosure, as will also be apparent to those skilled in the software art. In particular, the computer program product for authenticating, routing, and managing documents according to the present invention can be written in a number of computer languages including but not limited to C, C++, Fortran, and Basic, as would be recognized by those of ordinary skill in the art. The invention may also be implemented by the preparation of applications specific integrated circuits or by interconnecting an appropriate network of conventional component circuits, as will be readily apparent to those skilled in the art. Thus, the invention is not limited to the implementations shown in the specification, and ordinary programming and methods of generating interfaces which are alternative to web interfaces, http, etc. may be used.

[0385] The present invention thus also includes a computer-based product that may be hosted on a storage medium and include instructions that can be used to program a computer to perform a process in accordance with the present invention. This storage medium can include, but is not limited to, any type of disk including floppy disks, optical disks, CD-ROM, magneto-optical disks, ROMs, RAMs, EPROMs, EEPROMs, Flash Memory, Magnetic or Optical Cards, or any type of media suitable for storing electronic instructions.

[0386] Advantageously, the present invention can be incorporated with the system and method for managing documents disclosed in U.S. Application Serial No. 11/092,836, filed March 30, 2005, U.S. Application Serial No. 11/092,831 filed March 30, 2005, U.S. Application Serial No. 11/092,829 filed March 30, 2005, U.S. Application Serial No. 09/795,438, filed March 1, 2001; U.S. Application Serial No. 10/243,645, filed September 16, 2002; and U.S. Application Serial No. 10/294,607, filed November 15, 2002.

[0387] Obviously, numerous additional modifications and variations of the present invention are possible in light of the above teachings. It is therefore to be understood that within the scope of the appended claims the present invention may be practiced otherwise than as specifically described herein.

Claims

1. A method, **characterized by**:

   launching a host application of an image handling device, the image handling device including at least one embedded function and a network interface;
   determining external functions for the image handling device, the external functions utilizing the network interface and including operations which are performed remotely from the image handling device;
   storing information regarding available external functions determined by the determining in a configuration file;
   presenting a graphical interface that includes selectable graphical indicia representing each function accessible on the image handling device including the at least one embedded function and the available external functions; and
   executing the at least one embedded function and the determined external functions based on a selection of the corresponding graphical indicia.

2. The method according to claim 1, **characterized in that** the determining scans a network connected to the network interface to determine available external functions.

3. The method according to claim 1, **characterized in that** the determining connects to an external server that identifies available external functions and transmits information regarding the available external functions to the image handling device.

4. The method according to claim 1, **characterized in that** the determining uses a previously created configuration file as a basis for confirming the availability of external functions.

5. The method according to claim 4, further **characterized by**:

determining priority of functions when an embedded function conflicts with an available external function.

6. The method according to claim 5, **characterized in that** the presenting presents graphical indicia based on the result of the determining priority of functions.

7. The method according to claim 6, **characterized in that** the graphical indicia presented corresponds only to function determined to have priority.

8. The method according to claim 5, **characterized in that** accessibility of the conflicting functions is used to determine priority.

9. The method according to claim 1, further **characterized by** :

accessing the configuration file of the image handling device, the configuration file including information regarding the activation status of each function for the image handling device including both embedded and external functions; and
determining the accessibility of each function for the image handling device including both embedded and external functions using the configuration file.

10. The method according to claim 9, **characterized in that** the accessibility is determined based on the activation status of each function in the configuration file.

11. The method according to claim 1, **characterized in that** a log of the executing is stored on the image handling device.

12. The method according to claim 11, **characterized in that** the log is transmitted over the network for storage in a central repository.

13. The method according to claim 1, **characterized in that** the configuration file is an extensible markup language configuration file.

14. An image handling device, **characterized by**:

a host application configured to provide a core service of the image handling device and including at least one embedded function and a network interface;
an external function unit configured to determine external functions for the image handling device, the external functions utilizing the network interface and including operations which are performed remotely from the image handling device;
a configuration file configured to store information regarding available external functions determined by the determining;
a display configured to present a graphical interface that includes selectable graphical indicia representing each function accessible on the image handling device including the at least one embedded function and the available external functions; and
an input unit configured to receive user input and to execute the selected at least one embedded function or the determined external function based on the selection of the corresponding graphical indicia.

15. The image handling device according to claim 14, **characterized in that** the external function unit scans a network connected to the network interface to determine available external functions.

16. The image handling device according to claim 14, **characterized in that** the external function unit connects to an external server that identifies available external functions and transmits information regarding the available external functions to the image handling device.

17. The image handling device according to claim 14, **characterized in that** the external function unit uses a previously created configuration file as a basis for confirming the availability of external functions.

18. The image handling device according to claim 17, further **characterized by**:

a priority unit configured to determine a priority of functions when an embedded function conflicts with an available external function.

19. The image handling device according to claim 18, **characterized in that** the display presents graphical indicia based on the result of the determining of the priority unit.

20. The image handling device according to claim 19, **characterized in that** the graphical indicia presented corresponds only to function determined to have priority.

21. The image handling device according to claim 18, **characterized in that** accessibility of the conflicting functions is used to determine priority.

22. The image handling device according to claim 14,

further **characterized by**:

an activation unit configured to access the configuration file of the image handling device, the configuration file including information regarding the activation status of each function for the image handling device including both embedded and external functions; and
an accessibility unit configured to determine the accessibility of each function for the image handling device including both embedded and external functions using the configuration file.

23. The image handling device according to claim 22, **characterized in that** the accessibility unit determines the accessibility of each function based on the activation status of each function in the configuration file.

24. The image handling device according to claim 14, **characterized in that** a log of each function executed on the image handling device is stored on the image handling device.

25. The image handling device according to claim 24, **characterized in that** the log is transmitted over the network for storage in a central repository.

26. The image handling device according to claim 14, **characterized in that** the configuration file is an extensible markup language file.

27. A method, **characterized by**:

launching a host application of an image handling device, the image handling device including at least one plug-in and a corresponding set of services;
accessing a configuration file of the image handling device, the configuration file including information regarding the activation status of each service corresponding to the at least one plug-in;
launching the at least one plug-in based on the information regarding the activation of each service, the plug-in providing the corresponding set of activated services to the host application; and
presenting a graphical interface that includes a graphical indicia of each activated service corresponding to each activated plug-in.

28. The method according to claim 27, **characterized in that** the information regarding the activation of each service is generated utilizing information obtained by connecting to a database containing information regarding the activation of each service stored on the image handling device.

29. The method according to claim 27, **characterized in that** the information regarding the activation of each service includes a use time limit.

30. The method according to claim 27, **characterized in that** the configuration file is an XML file.

31. The method according to claim 27, **characterized in that** the information regarding the activation of each service includes a limit on the number of operations that can be performed on the image handling device.

32. The method according to claim 27, **characterized in that** the information regarding activation of each service is remotely input into the image handling device.

33. The method according to claim 27, **characterized in that** a user of the image handling device can modify the information regarding activation of each service by purchasing activation.

34. The method according to claim 33, **characterized in that** the user can purchase activation using an external device.

35. The method according to claim 34, **characterized in that** the external device is one of biometrics, a pin code, proximity card, smart card, or magnetic swipe card.

36. A method, **characterized by**:

launching a host application on an image handling device, the image handling device including an application layer, hardware and an operating system;
launching an activation manager, the activation manager determining which installed plug-ins and which services corresponding to the installed plug-ins to activate;
reading a configuration file stored on the image handling device, the configuration file including information identifying the image handling device, the user of the image handing device and the services corresponding to the installed plug-ins;
determining the activation status of each service in the configuration file;
updating the configuration file based on the determining, the configuration file being updated to including information regarding the activation status of each service;
generating a project array based on the number of installed plug-ins;
generating a service array for each project;
displaying a project array window, the project

array window graphically displaying each project included in the project array;
determining the activation status of each project selected by a user and each corresponding service; and
displaying a main window and a default service window when a project is selected in the project array window and is determined to be active, the main window including graphical indicia of the activated project services.

37. The method according to claim 36, further **characterized by**:

displaying options in the corresponding service window for entry and selection when one of the graphical indicia corresponding to the activated project services is selected;
adding upload data produced by the hardware of the image handling device and options entered and selected in the service window to a job queue;
identifying activated service date handlers using the activation information in the configuration file; and
processing upload data in the job queue using the activated service data handlers, the service data handlers being services for sending the upload data from the image handling device to a destination included in a configuration file.

38. A computer readable medium storing program code for causing an image handling device to perform a method,
characterized by:

launching a host application on an image handling device, the image handling device including an application layer, hardware and an operating system;
launching an activation manager, the activation manager determining which installed plug-ins and which services corresponding to the installed plug-ins to activate;
reading a configuration file stored on the image handling device, the configuration file including information identifying the image handling device, the user of the image handing device and the services corresponding to the installed plug-ins;
determining the activation status of each service in the configuration file;
updating the configuration file based on the determining, the configuration file being updated to including information regarding the activation status of each service;
generating a project array based on the number of installed plug-ins;
generating a service array for each project;
displaying a project array window, the project array window graphically displaying each project included in the project array;
determining the activation status of each project selected by a user and each corresponding service; and
displaying a main window and a default service window when a project is selected in the project array window and is determined to be active, the main window including graphical indicia of the activated project services.

39. An image handling device, **characterized by**:

a host application configured to provide the core service of the image handling device;
a plug-in application configured to be programmatically invoked by the host application;
an activation manager configured to control access to the plug-in application;
a configuration file updated by the activation manager stored in a memory and including information regarding activation of the plug-in application and functions corresponding to the plug-in application, the host application configured to programmatically invoke the plug-in in accordance with information regarding activation in the configuration file.

40. The device according to claim 39, **characterized in that** the activation manger controls access to the plug-in application by connecting to a database including activation information.

41. The device according to claim 39, **characterized in that** the information regarding activation includes a use time limit.

42. The device according to claim 39, **characterized in that** the configuration file is an XML file.

43. The device according to claim 39, **characterized in that** the information regarding activation includes a limit on the number of operations that can be performed on the image handling device.

44. The device according to claim 39, **characterized in that** the information regarding activation is remotely input into the image handling device.

45. The device according to claim 39, **characterized in that** a user of the image handling device can modify the information regarding activation by purchasing activation.

46. The device according to claim 45, **characterized in that** the user can purchase activation using an ex-

ternal device.

47. The device according to claim 46, **characterized in that** the external device is one of biometrics, a pin code, proximity card, smart card, or magnetic swipe card.

48. An image handling device, **characterized by**:

a display device;
a memory configured to store a configuration file associated with a host application, the configuration file including:

an identification of at least one activated plug-in associated with the host application;
at least one project corresponding to the activated plug-in;
at least one activated service corresponding to the at least one project and including data indicating the functions of the image handling device;

an activation manager configured to determine the activation status of each plug-in and corresponding services using information regarding activation included in the configuration file;
a controller configured to interface with the host application and activation manager to present a graphical interface including activated projects and the corresponding set of activated services,

wherein the controller displays activated project selection data enabling the user to select an activated project and displays activated service selection data, corresponding to the selected activated project, enabling the user to select an activated service.

49. A method of modifying images, **characterized by**:

scanning at least one document having multiple pages;
displaying a thumbnail image of at least one of the pages of the scanned document;
displaying a selectable graphical indicia corresponding to at least one operation for modifying at least one image of the scanned document;
selecting the selectable graphical indicia; and
modifying the at least one image in accordance with said selecting.

50. The method of modifying images according to claim 49, further **characterized by**:

selecting a thumbnail image displayed by the displaying based on user input;
displaying a preview image of at least one of the pages of the scanned document corresponding to the selected thumbnail image.

51. The method of modifying images according to claim 50, **characterized in that** the at least one operation for modifying at least one image of the scanned document also modifies the preview image.

52. The method of modifying images according to claim 51, further **characterized by**:

modifying the preview image in accordance with said selecting.

53. The method of modifying images according to claim 52, **characterized in that** the at least one image of the scanned document is modified subsequent to the modification of the preview image.

54. The method of modifying images according to claim 49, **characterized in that** the at least one operation for modifying includes one of a zoom, pan, undo, insert, rotate or delete operation.

55. The method of modifying images according to claim 49, further **characterized by**:

displaying a plurality of thumbnail images, each thumbnail image corresponding to a page of the scanned document.

56. The method of modifying images according to claim 55, **characterized in that** a number of the plurality of thumbnail images is set by user input.

57. The method of modifying images according to claim 56, **characterized in that** the number of the plurality of thumbnail images is set by user input from a limited number of predetermined options.

58. The method of modifying images according to claim 49,
characterized in that the modifying the at least one image is performed by a remote server.

59. The method of modifying images according to claim 52,
characterized in that the modifying the preview image is performed by a remote server.

60. The method of modifying images according to claim 49, **characterized in that** the modifying the at least one image is performed by a multi-function device.

61. A method of modifying images, **characterized by**:

scanning at least one document having multiple pages;
displaying a preview image of at least one of the

pages of the scanned document;
displaying a selectable graphical indicia corresponding to at least one operation for modifying the preview image;
selecting the selectable graphical indicia; and
modifying the preview image in accordance with said selecting.

62. The method of modifying images according to claim 61, further **characterized by**:

modifying an image of at least one of the pages of the scanned document in accordance with the modifying the preview image.

63. The method of modifying images according to claim 61, further **characterized by**:

displaying a thumbnail image of at least one of the pages of the scanned document; and
selecting a thumbnail image based on user input.

64. The method of modifying images according to claim 63, further **characterized by**:

displaying a preview image of at least one of the pages of the scanned document in accordance with the selecting a thumbnail image based on user input.

65. The method of modifying images according to claim 63, **characterized in that** each preview image corresponds to a thumbnail image.

66. The method of modifying images according to claim 62,
characterized in that the modifying of the image of at least one of the pages of the scanned document is performed by a remote server.

67. The method of modifying images according to claim 61,
characterized in that the modifying the preview image is performed by a remote server.

68. The method of modifying images according to claim 61, **characterized in that** the modifying the preview image is performed by a multi-function device.

69. The method of modifying images according to claim 62, **characterized in that** the image of at least one of the pages of the scanned document is modified subsequent to the modification of the preview image.

70. The method of modifying images according to claim 61, **characterized in that** the at least one operation for modifying includes at least one of a zoom, pan, undo, insert, rotate or delete operation.

71. An image processing device, **characterized by**
a scanner configured to produce an image of at least one document having multiple pages;
a communications interface configured to connect the image processing device to a server via a network;
a display unit configured to display a thumbnail image of at least one of the pages of the scanned document; and
a user interface configured to display a selectable graphical indicia corresponding to at least one operation for modifying at least one image of the scanned document.

72. The image processing device according to claim 71, **characterized in that** the communications interface is configured to transmit the at least one image of the scanned document to a remote server on which the at least one operation for modifying is performed.

73. The image processing device according to claim 71, **characterized in that** the image processing device is further configured to include a modification unit for performing the at least one operation for modifying.

74. The image processing device according to claim 73, **characterized in that** the modification unit is configured to perform at least one of a zoom, pan, undo, insert, rotate or delete.

75. The image processing device according to claim 71, **characterized in that** the thumbnail image of at least one of the pages of the scanned document is a user selectable image corresponding to a respective one of the pages of the scanned document.

76. The image processing device according to claim 71, **characterized in that** the user interface is part of the display unit.

77. An image processing device, **characterized by**:

a scanner configured to produce an image of at least one document having multiple pages;
a communications interface configured to connect the image processing device to a server via a network;
a display unit configured to display a preview image of at least one of the pages of the scanned document; and
a user interface configured to display a selectable graphical indicia corresponding to at least one operation for modifying the preview image.

78. The image processing device according to claim 77, **characterized in that** the image processing device

further includes a modification unit configured to modify the image of at least one of the pages of the scanned document in accordance with the modifying of the preview image.

79. The image processing device according to claim 77, **characterized in that** the communications interface is configured to transmit the at least one image of the scanned document to a remote server on which the at least one operation for modifying the preview image is performed.

80. The image processing device according to claim 78, **characterized in that** the modification unit is configured to perform at least one of a zoom, pan, undo, insert, rotate or delete.

81. The image processing device according to claim 77, **characterized in that** the user interface is part of the display unit.

82. A computer readable medium storing program code for causing an image processing device to perform a method,
characterized by:

scanning at least one document having multiple pages;
displaying a thumbnail image of at least one of the pages of the scanned document;
displaying a selectable graphical indicia corresponding to at least one operation for modifying at least one image of the scanned document;
selecting the selectable graphical indicia; and
modifying the at least one image in accordance with said selecting.

83. A computer readable medium storing program code for causing an image processing device to perform a method,
characterized by:

scanning at least one document having multiple pages;
displaying a preview image of at least one of the pages of the scanned document;
displaying a selectable graphical indicia corresponding to at least one operation for modifying the preview image;
selecting the selectable graphical indicia; and
modifying the preview image in accordance with said selecting.

84. An image processing device, **characterized by**:

a scan means for scanning at least one document having multiple pages;
a display means for displaying a thumbnail image of at least one of the pages of the scanned document and displaying a selectable graphical indicia corresponding to at least one operation for modifying at least one image of the scanned document;
a selection means for selecting the selectable graphical indicia; and
a modification means for modifying the at least one image in accordance with said selecting.

85. An image processing device, **characterized by**:

a scan means for scanning at least one document having multiple pages;
a display means for displaying a preview image of at least one of the pages of the scanned document and displaying a selectable graphical indicia corresponding to at least one operation for modifying the preview image;
a selection means for selecting the selectable graphical indicia; and
a modification means for modifying the preview image in accordance with said selecting.

86. A method for authenticating a user of an image processing system, **characterized by**:

receiving user credentials at an authentication device corresponding to an image processing device;
transmitting the user credentials to a first server;
judging, at the first server, the validity of the user credentials by comparing the received user credentials to stored user data;
allowing access to a first server using a result of said judging; and
allowing access to a second sever based on the user credentials.

87. The method of Claim 86, **characterized in that** the allowing access to a second server step further comprises:

transmitting a result of the judging step from the first server to at least one of the image processing device and the second server.

88. The method of Claim 86, **characterized in that** the allowing access to a second server step further comprises:

requesting, at the image processing device, access to the second server.

89. The method of Claim 88, **characterized in that** the request for access to the second server comprises a request to access at least one of a plurality of back-end applications connected to the second server.

90. The method of Claim 88, **characterized in that** the request for access to the second server comprises at least one of an Internet Protocol (IP) address, Media Access Control (MAC) address, and serial number corresponding to the image processing device.

91. The method of Claim 89, **characterized in that** the allowing access to a second server step further comprises:

determining, at the second server, that the at least one of a plurality of backend applications requires user authentication.

92. The method of Claim 91, **characterized in that** the allowing access to a second server step further comprises:

generating a template indicating additional user credential information necessary for a user to be authenticated at the at least one of a plurality of backend applications.

93. The method of Claim 92, **characterized in that** the allowing access to a second server step further comprises:

transmitting, from the second sever, a request for at least the user credentials to the first server.

94. The method of Claim 93, **characterized in that** the allowing access to a second server step further comprises:

completing the template indicating additional user credential information using the user credentials received from the first server.

95. The method of Claim 94, **characterized in that** the allowing access to a second server step further comprises:

determining that the template can not be completed based on the user credentials received from the server;
transmitting the template to the image processing device; and
displaying a login interface at the image processing prompting a user to enter additional user credential information missing from the template.

96. The method of Claim 86, **characterized in that** the step of receiving user credentials comprises:

receiving one of a card input, retinal scan, fingerprint, voice, and personal identification number corresponding to a user.

97. A system for authenticating a user of an image processing system, **characterized by**:

an authentication device corresponding to an image processing device and configured to receive user credentials;
a first server configured to receive the user credentials transmitted from the authentication device;
a first module, at the first server, configured to judge the validity of the user credentials by comparing the received user credentials to stored user data;
the first server configured to allow access based on a result of said judging; and
a second server configured to allow access based on the user credentials.

98. The system of Claim 97, further **characterized by**:

a first communications interface at the first server configured to transmit a result of the judging to at least one of the image processing device and the second server.

99. The system of Claim 97, further **characterized by**:

a second communications interface at the image processing device configured to request access to the second server.

100. The system of Claim 99, further **characterized by**:

at least one of a plurality of backend applications connected to the second server,

wherein the request for access to the second server identifies at least one of the plurality of backend applications connected to the second server.

101. The system of Claim 99, **characterized in that** the request for access to the second server comprises at least one of an Internet Protocol (IP) address, Media Access Control (MAC) address, and serial number corresponding to the image processing device.

102. The system of Claim 100, further **characterized by**:

a second module at the second server configured to determine that the at least one of a plurality of backend applications requires user authentication.

103. The system of Claim 102 **characterized in that** the second module is further configured to generate a

template indicating additional user credential information necessary for a user to be authenticated at the at least one of a plurality of backend applications.

104. The system of Claim 103, further **characterized by**:

a third communications interface at the second server configured to transmit a request for at least the user credentials to the first server.

105. The system of Claim 104, **characterized in that** the second module at the second server is further configured to complete the template indicating additional user credential information using the user credentials received from the first server.

106. The system of Claim 105, further **characterized by**:

a third module at the second server configured to determine that the template can not be completed based on the user credentials received from the server;
the third communications interface at the second server configured to transmit the template to the image processing device; and
a display at the image processing device configured to display a login interface a prompting a user to enter user credential information missing from the template.

107. The system of Claim 97, **characterized in that**:

the authentication device is one of a card reader, retinal scanner, fingerprint reader, voice scanner, proximity scanner and keypad device.

108. A system for authenticating a user of an image processing system, **characterized by**:

means for receiving user credentials at an authentication device corresponding to an image processing device;
means for transmitting the user credentials to a first server;
means for judging, at the first server, the validity of the user credentials by comparing the received user credentials to stored user data;
means for allowing access to a first server using a result of said judging; and
means for allowing access to a second sever based on the user credentials.

FIG.1

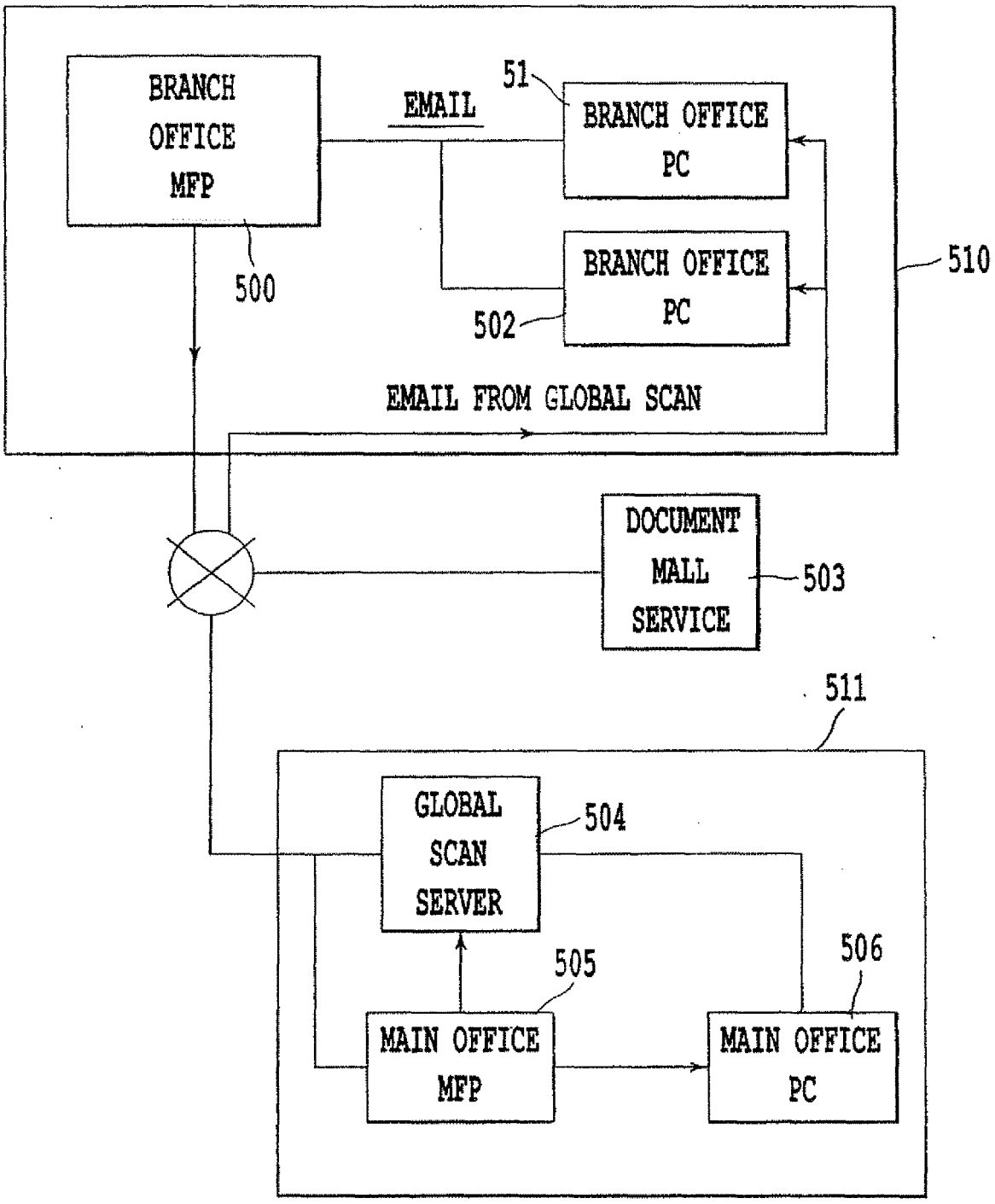

BRANCH OFFICE MFP
EMAIL
51
BRANCH OFFICE PC
BRANCH OFFICE PC
500
502
510
EMAIL FROM GLOBAL SCAN
DOCUMENT MALL SERVICE
503
511
GLOBAL SCAN SERVER
504
505
506
MAIN OFFICE MFP
MAIN OFFICE PC

FIG.2

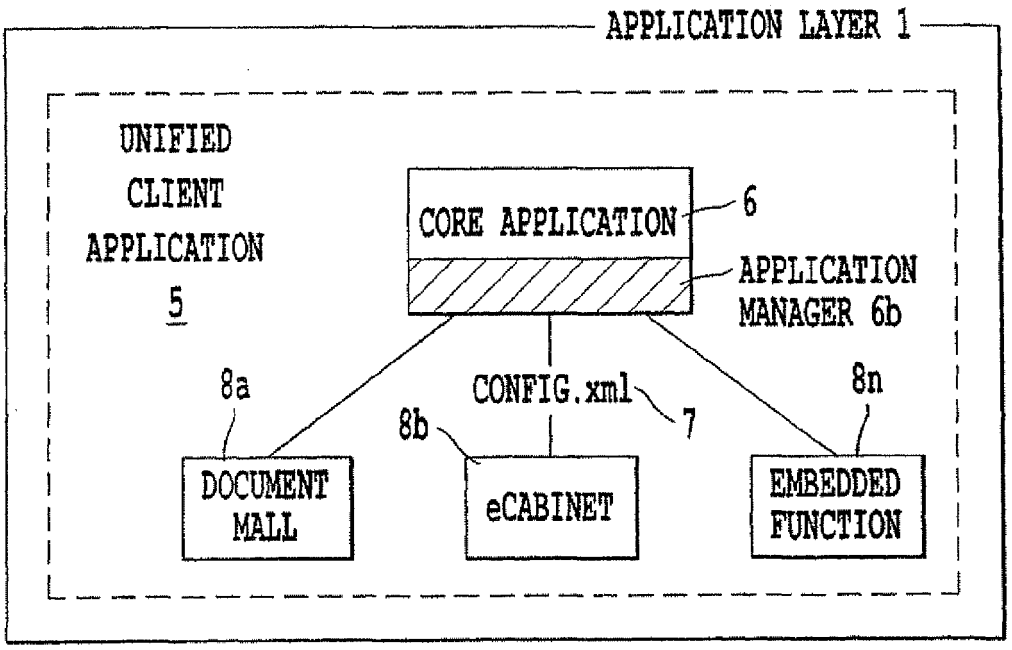

APPLICATION LAYER 1
UNIFIED CLIENT APPLICATION 5
CORE APPLICATION
6
APPLICATION MANAGER 6b
CONFIG.xml
7
8a
8b
8n
DOCUMENT MALL
eCABINET
EMBEDDED FUNCTION

FIG.3

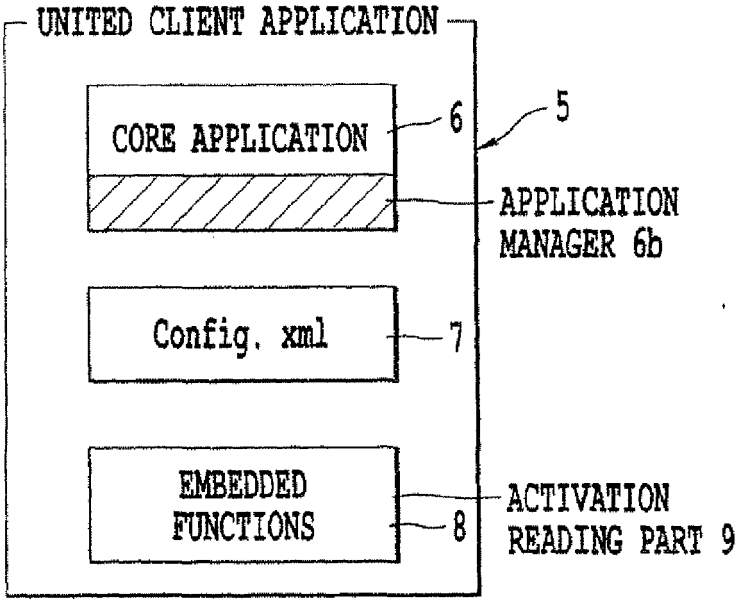

UNITED CLIENT APPLICATION
CORE APPLICATION
6
5
APPLICATION MANAGER 6b
Config. xml
7
EMBEDDED FUNCTIONS
8
ACTIVATION READING PART 9

FIG.4

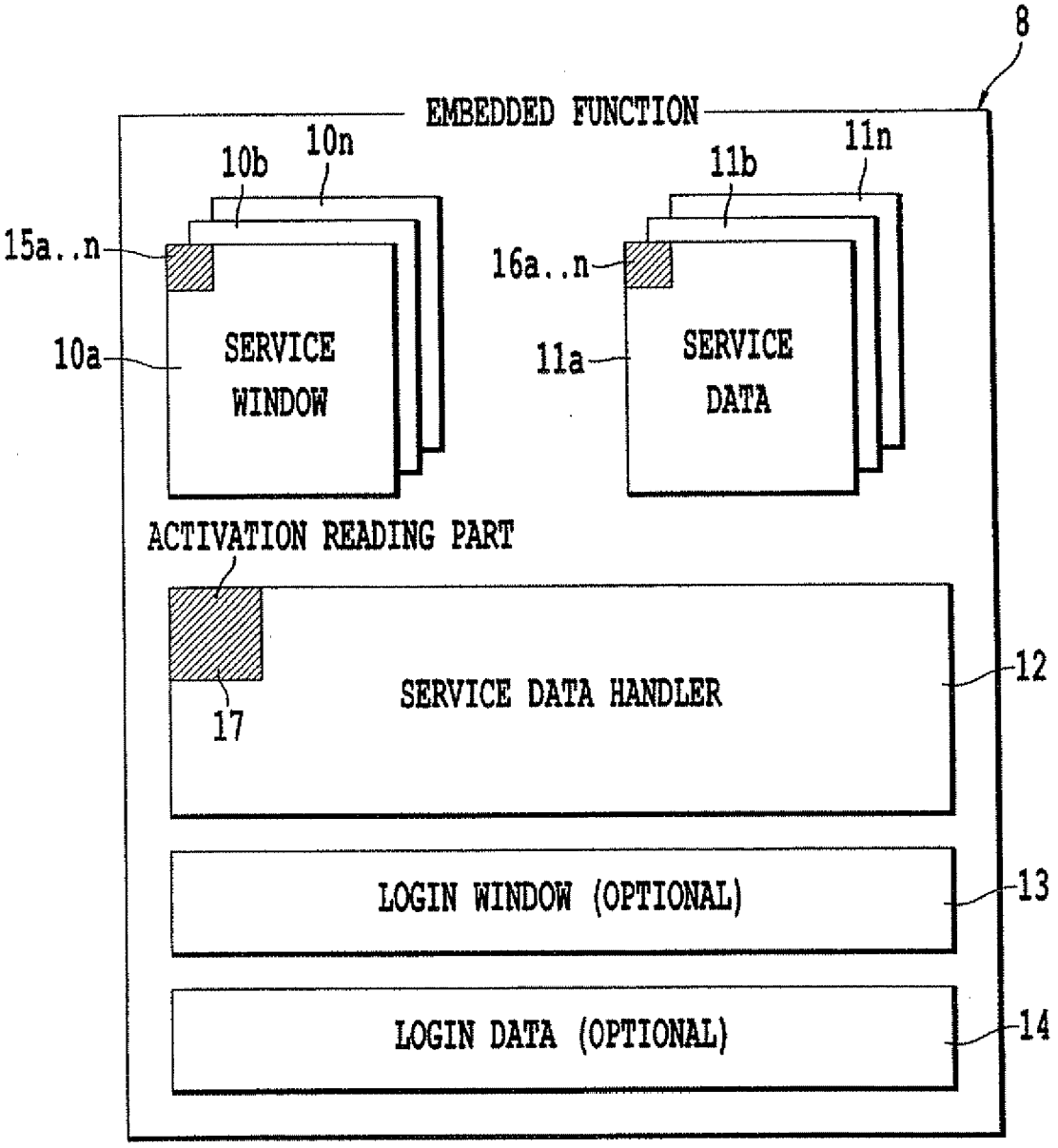

8
EMBEDDED FUNCTION
10b
10n
11b
11n
15a..n
16a..n
10a
SERVICE WINDOW
11a
SERVICE DATA
ACTIVATION READING PART
17
SERVICE DATA HANDLER
12
LOGIN WINDOW (OPTIONAL)
13
LOGIN DATA (OPTIONAL)
14

FIG.5

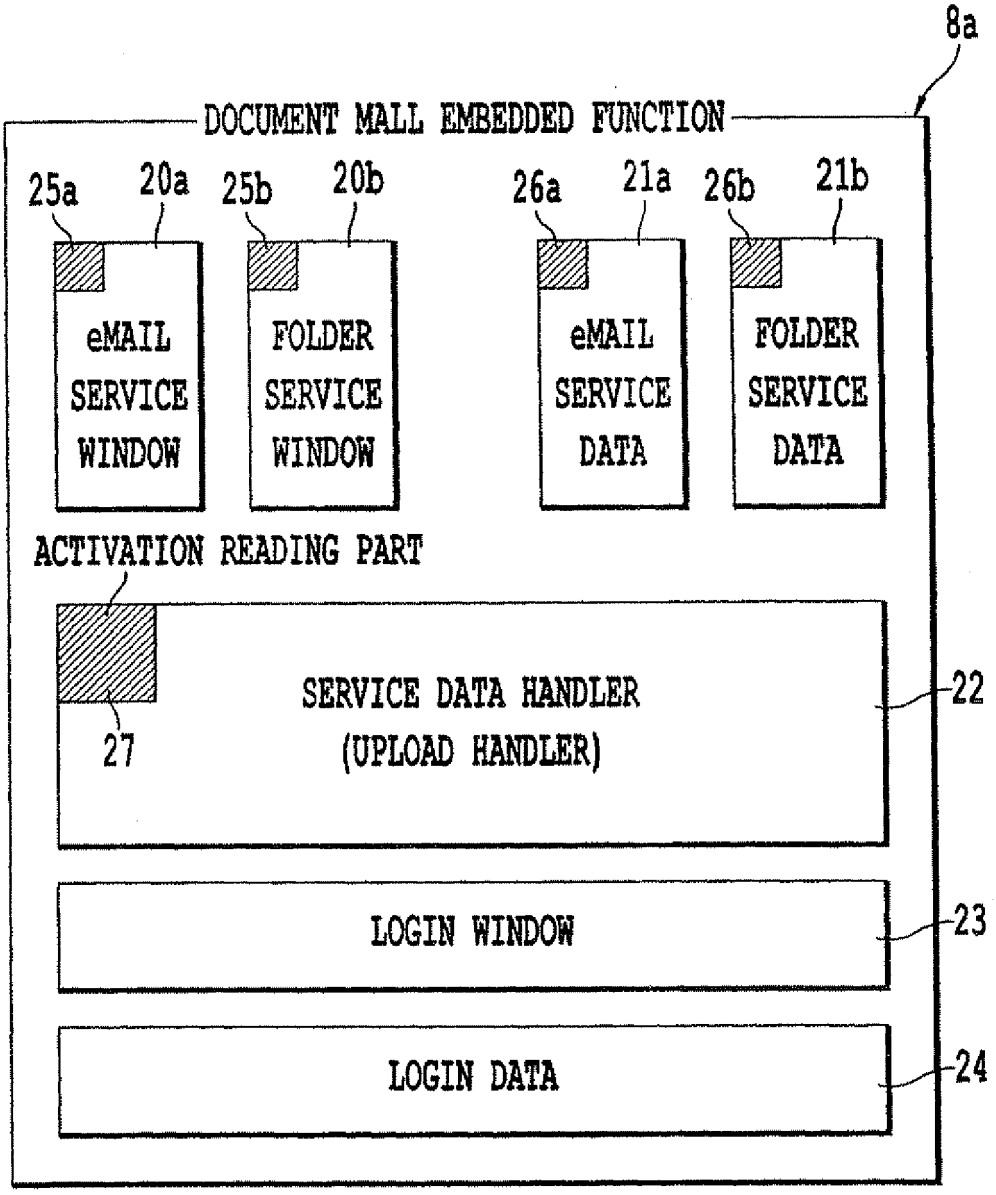

8a
DOCUMENT MALL EMBEDDED FUNCTION
25a
20a
25b
20b
26a
21a
26b
21b
eMAIL SERVICE WINDOW
FOLDER SERVICE WINDOW
eMAIL SERVICE DATA
FOLDER SERVICE DATA
ACTIVATION READING PART
27
SERVICE DATA HANDLER (UPLOAD HANDLER)
22
LOGIN WINDOW
23
LOGIN DATA
24

FIG.6A

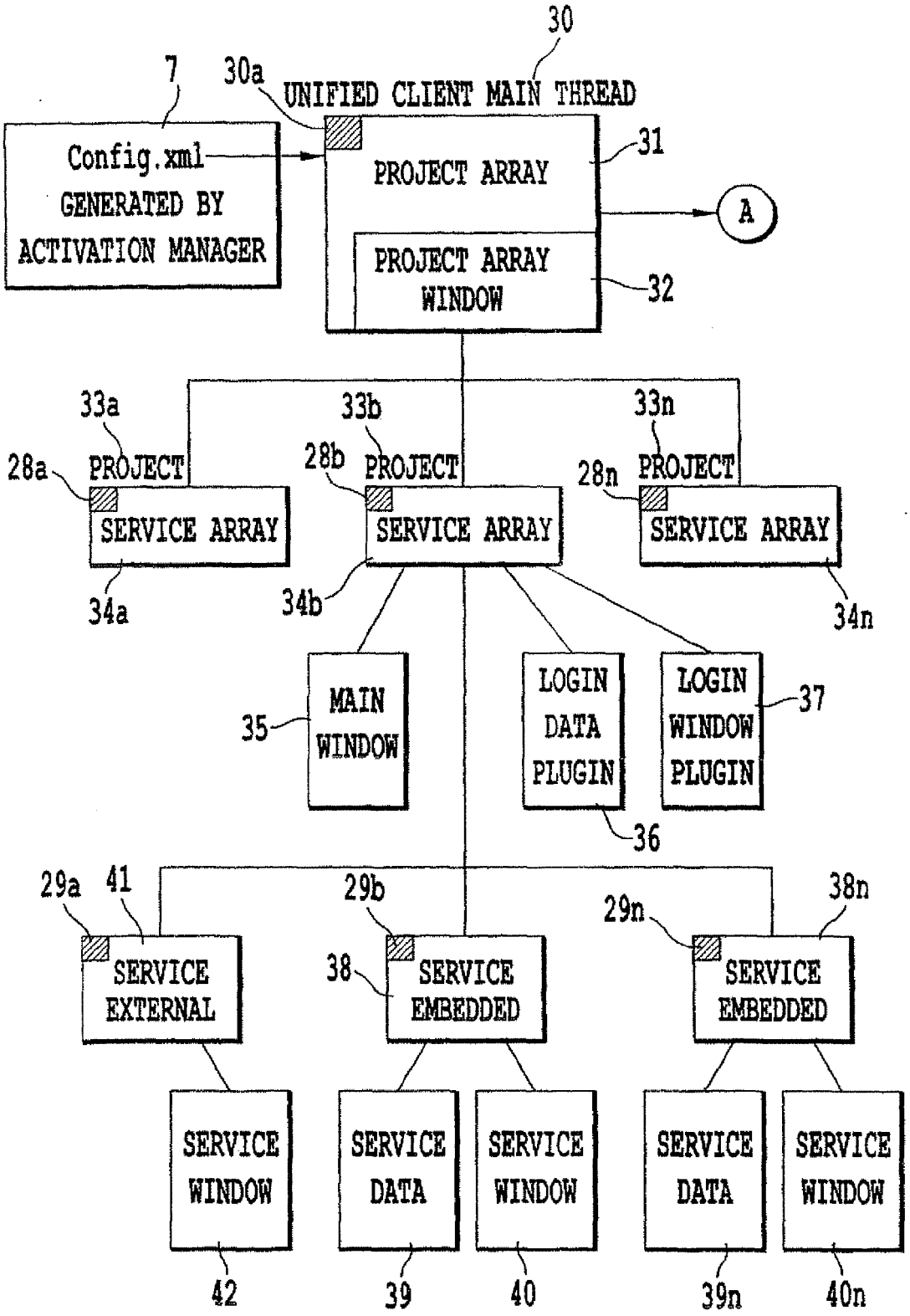

30
7
30a
UNIFIED CLIENT MAIN THREAD
Config.xml
GENERATED BY
ACTIVATION MANAGER
PROJECT ARRAY
31
A
PROJECT ARRAY
WINDOW
32
33a
33b
33n
28a
PROJECT
28b
PROJECT
28n
PROJECT
SERVICE ARRAY
SERVICE ARRAY
SERVICE ARRAY
34a
34b
34n
MAIN
WINDOW
35
LOGIN
DATA
PLUGIN
LOGIN
WINDOW
PLUGIN
37
36
29a
41
29b
29n
38n
SERVICE
EXTERNAL
38
SERVICE
EMBEDDED
SERVICE
EMBEDDED
SERVICE
WINDOW
SERVICE
DATA
SERVICE
WINDOW
SERVICE
DATA
SERVICE
WINDOW
42
39
40
39n
40n

FIG.6B

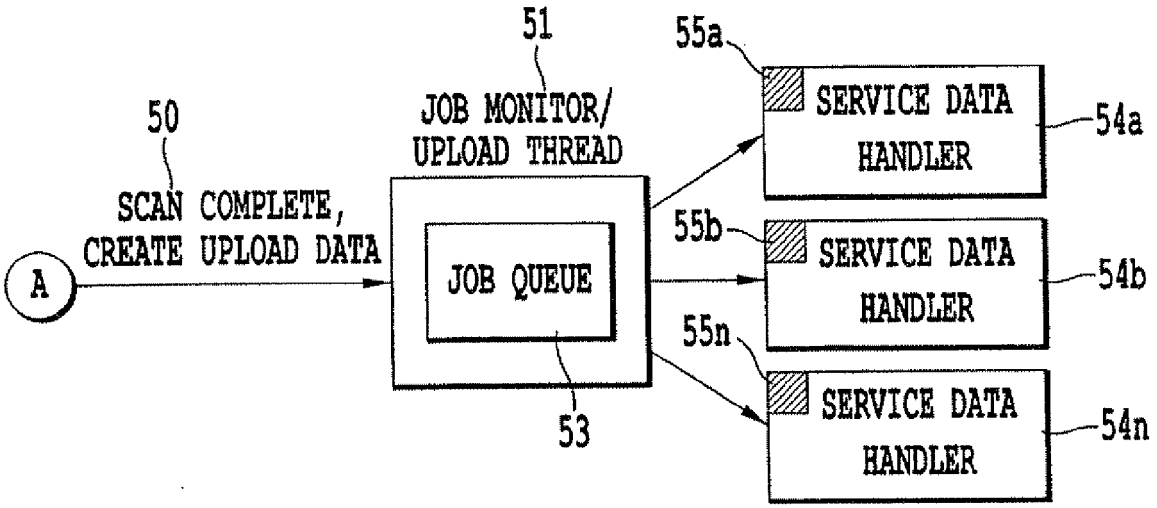

51
55a
50
JOB MONITOR/
UPLOAD THREAD
SERVICE DATA
HANDLER
54a
SCAN COMPLETE,
CREATE UPLOAD DATA
A
JOB QUEUE
55b
SERVICE DATA
HANDLER
54b
55n
SERVICE DATA
HANDLER
54n
53

FIG.7A

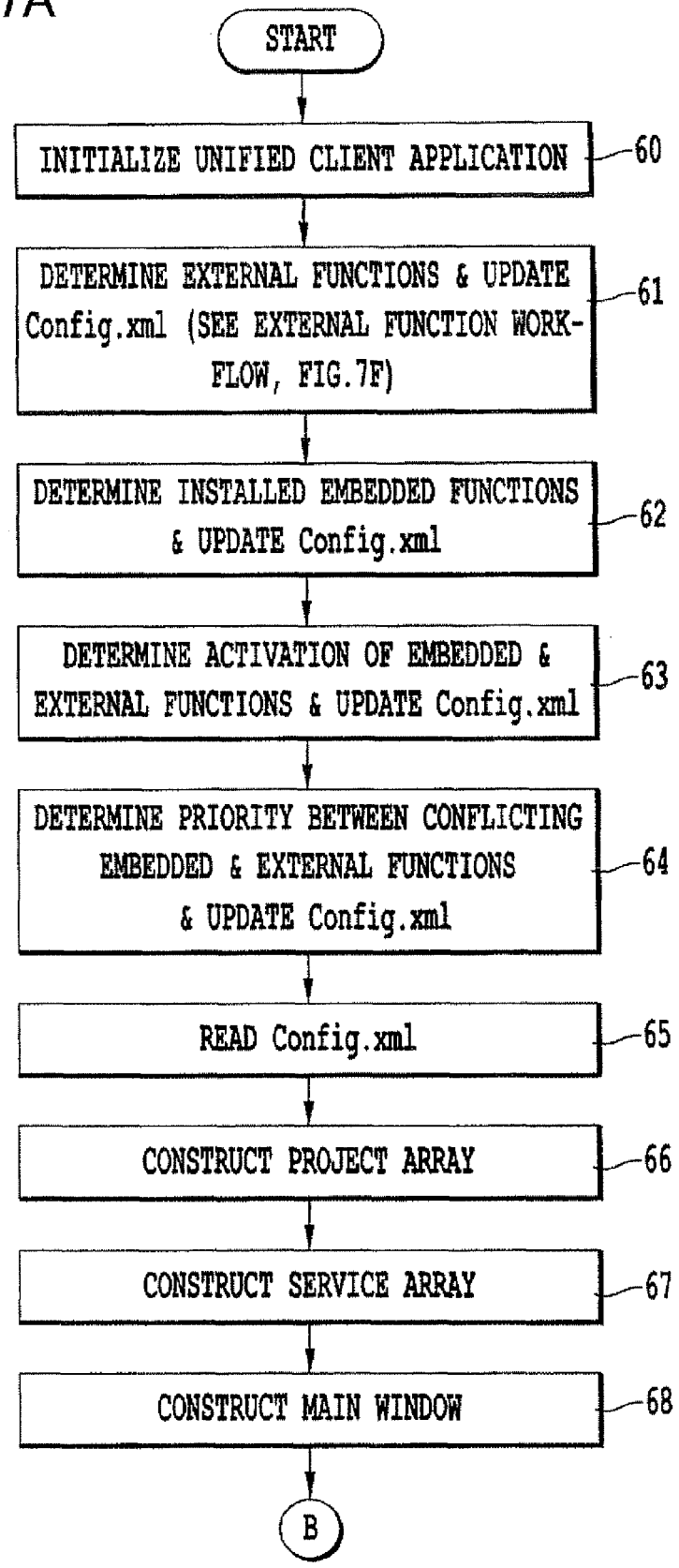

START
INITIALIZE UNIFIED CLIENT APPLICATION
60
DETERMINE EXTERNAL FUNCTIONS & UPDATE Config.xml (SEE EXTERNAL FUNCTION WORK-FLOW, FIG.7F)
61
DETERMINE INSTALLED EMBEDDED FUNCTIONS & UPDATE Config.xml
62
DETERMINE ACTIVATION OF EMBEDDED & EXTERNAL FUNCTIONS & UPDATE Config.xml
63
DETERMINE PRIORITY BETWEEN CONFLICTING EMBEDDED & EXTERNAL FUNCTIONS & UPDATE Config.xml
64
READ Config.xml
65
CONSTRUCT PROJECT ARRAY
66
CONSTRUCT SERVICE ARRAY
67
CONSTRUCT MAIN WINDOW
68
B

FIG.7B

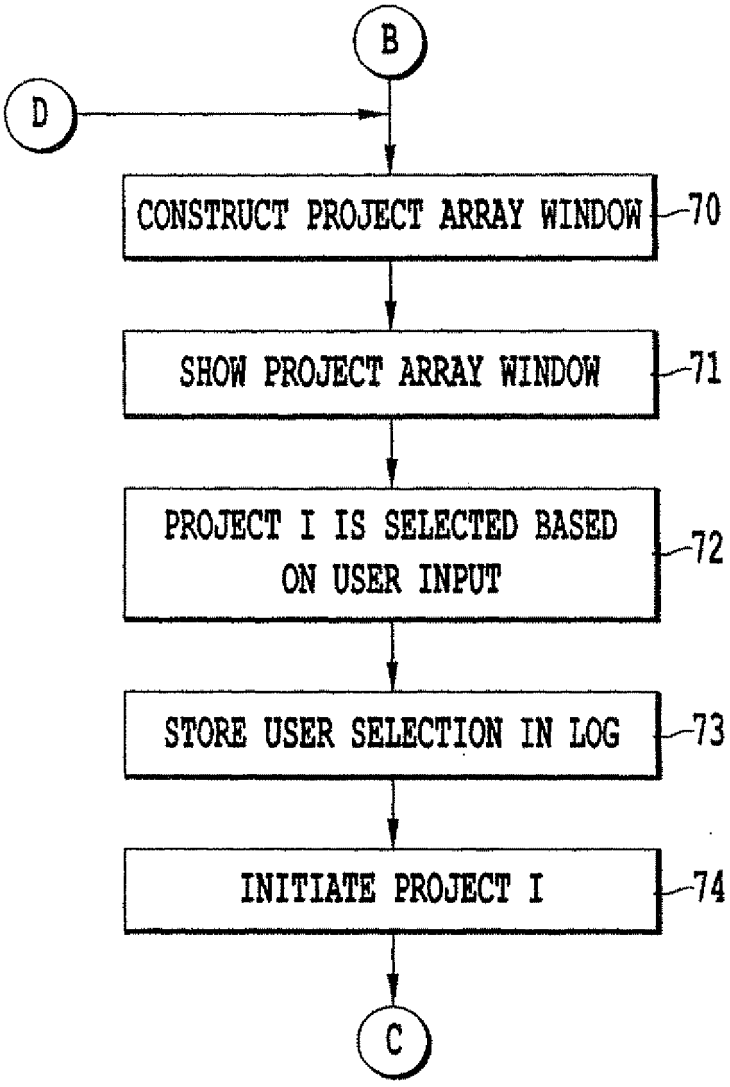

B
D
CONSTRUCT PROJECT ARRAY WINDOW
70
SHOW PROJECT ARRAY WINDOW
71
PROJECT I IS SELECTED BASED ON USER INPUT
72
STORE USER SELECTION IN LOG
73
INITIATE PROJECT I
74
C

FIG.7C

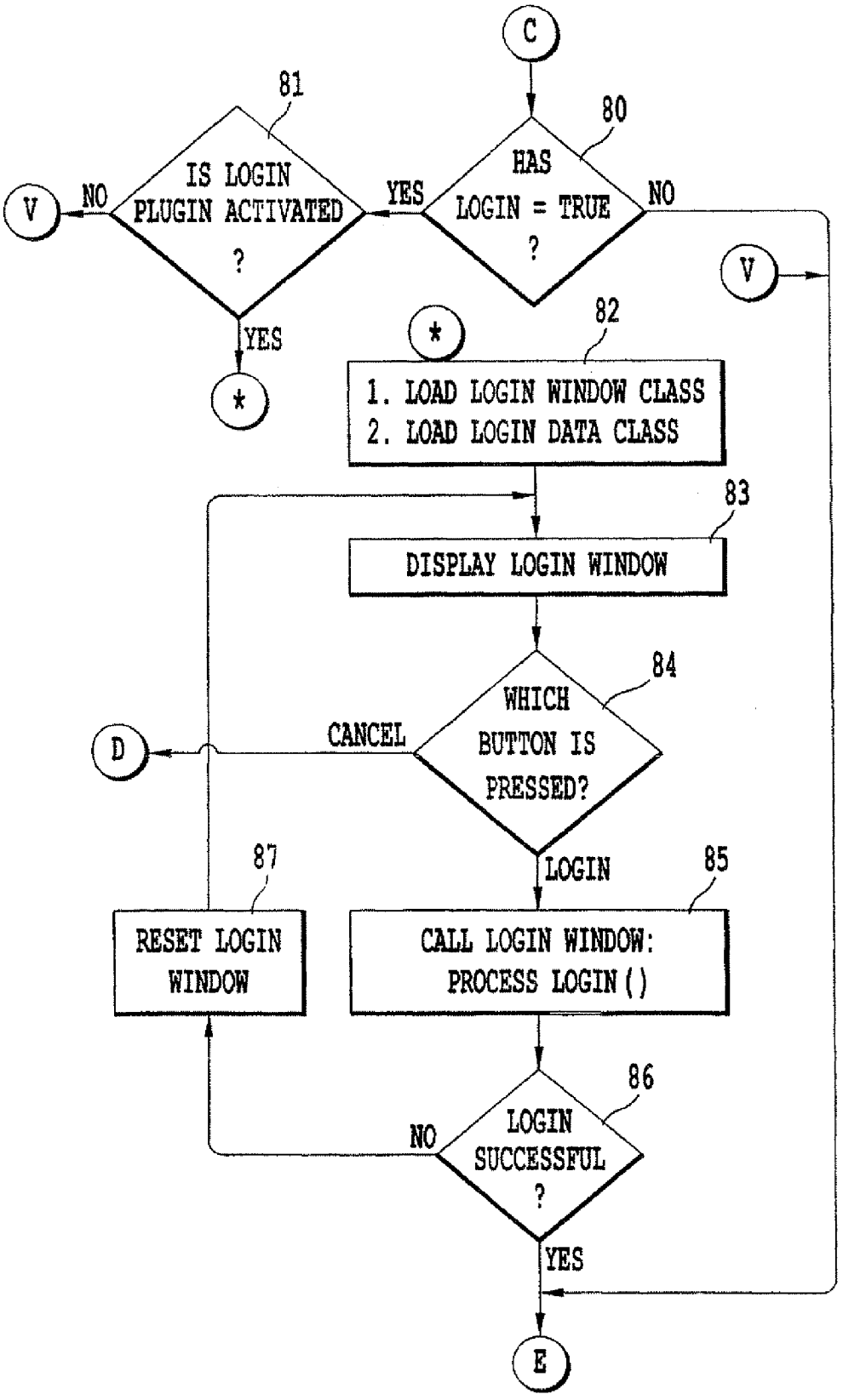

C
80
HAS LOGIN = TRUE ?
NO
YES
81
IS LOGIN PLUGIN ACTIVATED ?
NO
V
YES
*
V
*
82
1. LOAD LOGIN WINDOW CLASS
2. LOAD LOGIN DATA CLASS
83
DISPLAY LOGIN WINDOW
84
WHICH BUTTON IS PRESSED?
CANCEL
D
LOGIN
85
CALL LOGIN WINDOW: PROCESS LOGIN ( )
87
RESET LOGIN WINDOW
86
LOGIN SUCCESSFUL ?
NO
YES
E

FIG.7D(i)

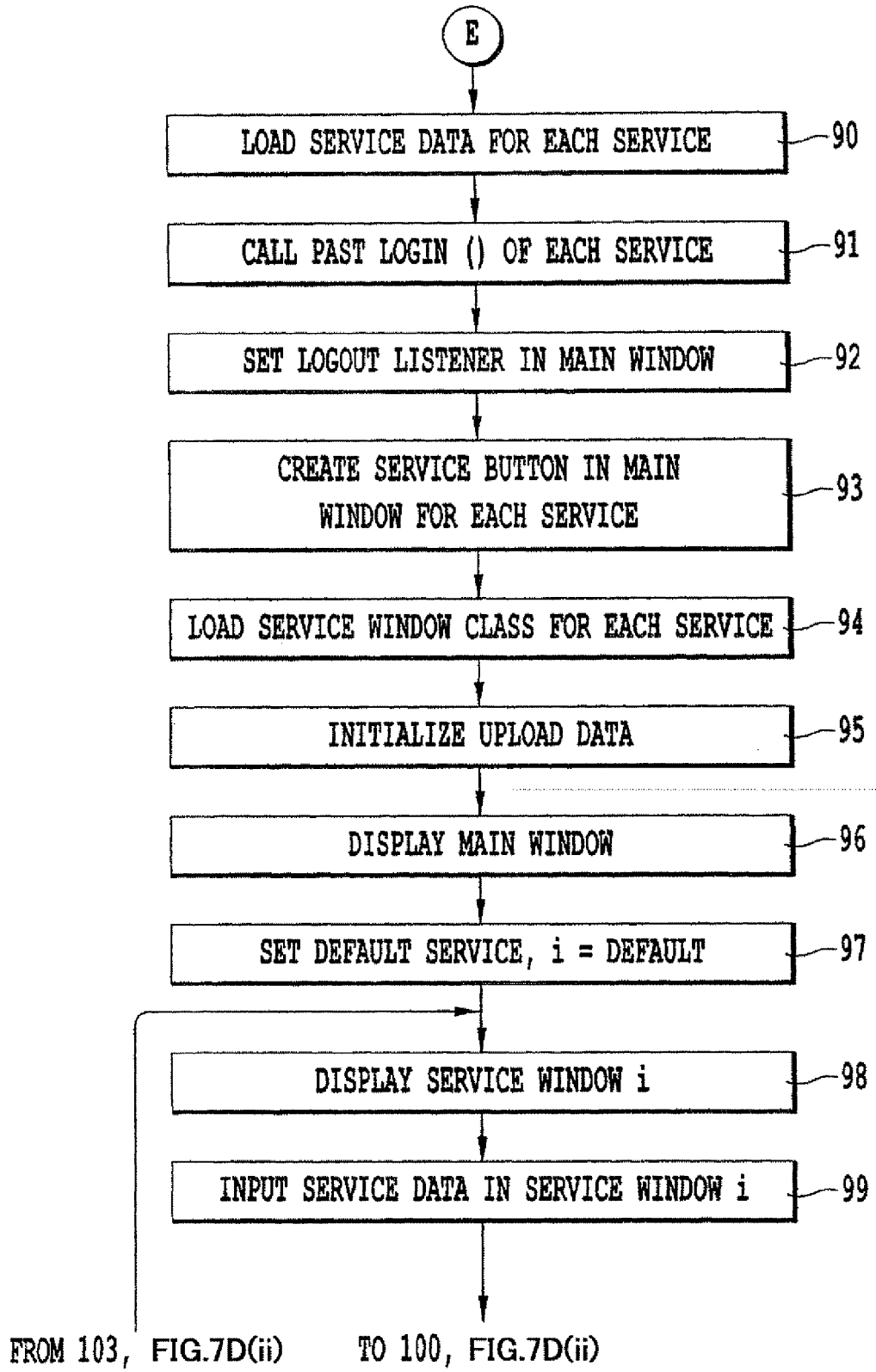

E
LOAD SERVICE DATA FOR EACH SERVICE
90
CALL PAST LOGIN () OF EACH SERVICE
91
SET LOGOUT LISTENER IN MAIN WINDOW
92
CREATE SERVICE BUTTON IN MAIN WINDOW FOR EACH SERVICE
93
LOAD SERVICE WINDOW CLASS FOR EACH SERVICE
94
INITIALIZE UPLOAD DATA
95
DISPLAY MAIN WINDOW
96
SET DEFAULT SERVICE, i = DEFAULT
97
DISPLAY SERVICE WINDOW i
98
INPUT SERVICE DATA IN SERVICE WINDOW i
99
FROM 103, FIG.7D(ii)
TO 100, FIG.7D(ii)

FIG.7D(ii)

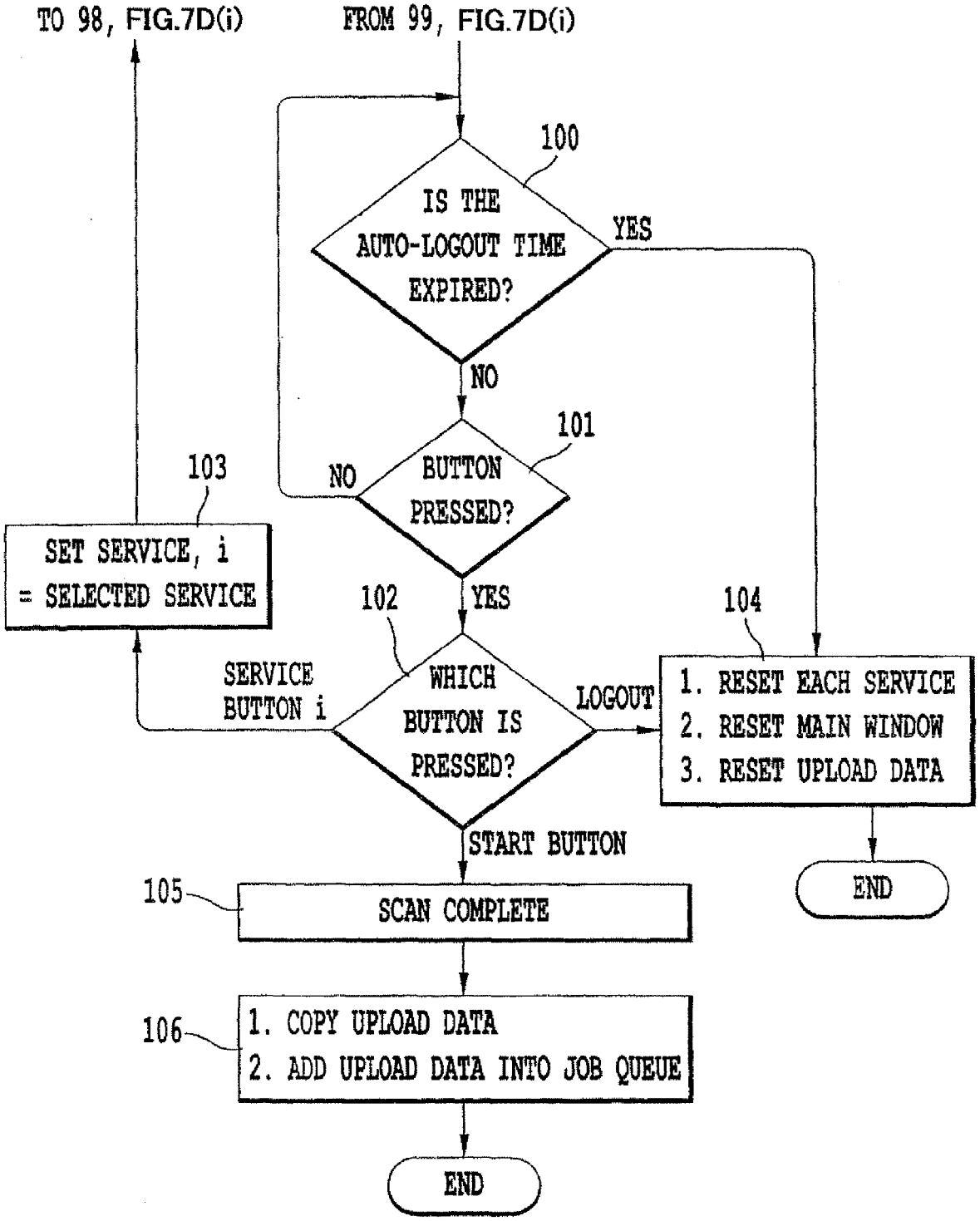

TO 98, FIG.7D(i)
FROM 99, FIG.7D(i)
100
IS THE AUTO-LOGOUT TIME EXPIRED?
YES
NO
101
BUTTON PRESSED?
NO
103
SET SERVICE, i = SELECTED SERVICE
102
YES
WHICH BUTTON IS PRESSED?
SERVICE BUTTON i
LOGOUT
104
1. RESET EACH SERVICE
2. RESET MAIN WINDOW
3. RESET UPLOAD DATA
END
START BUTTON
105
SCAN COMPLETE
106
1. COPY UPLOAD DATA
2. ADD UPLOAD DATA INTO JOB QUEUE
END

FIG.7E

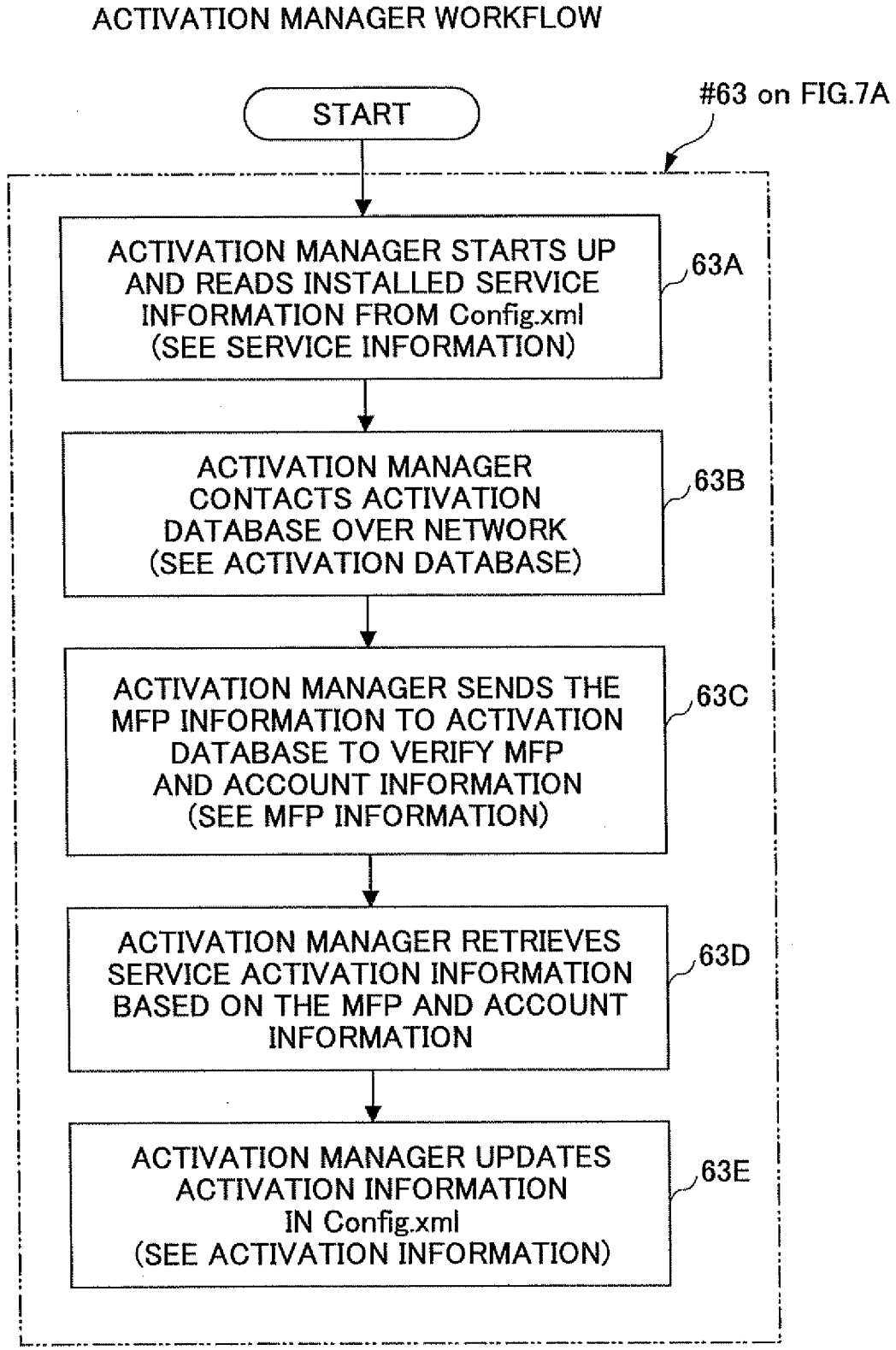

ACTIVATION MANAGER WORKFLOW
START
#63 on FIG.7A
ACTIVATION MANAGER STARTS UP AND READS INSTALLED SERVICE INFORMATION FROM Config.xml (SEE SERVICE INFORMATION)
63A
ACTIVATION MANAGER CONTACTS ACTIVATION DATABASE OVER NETWORK (SEE ACTIVATION DATABASE)
63B
ACTIVATION MANAGER SENDS THE MFP INFORMATION TO ACTIVATION DATABASE TO VERIFY MFP AND ACCOUNT INFORMATION (SEE MFP INFORMATION)
63C
ACTIVATION MANAGER RETRIEVES SERVICE ACTIVATION INFORMATION BASED ON THE MFP AND ACCOUNT INFORMATION
63D
ACTIVATION MANAGER UPDATES ACTIVATION INFORMATION IN Config.xml (SEE ACTIVATION INFORMATION)
63E

FIG.7F(i)

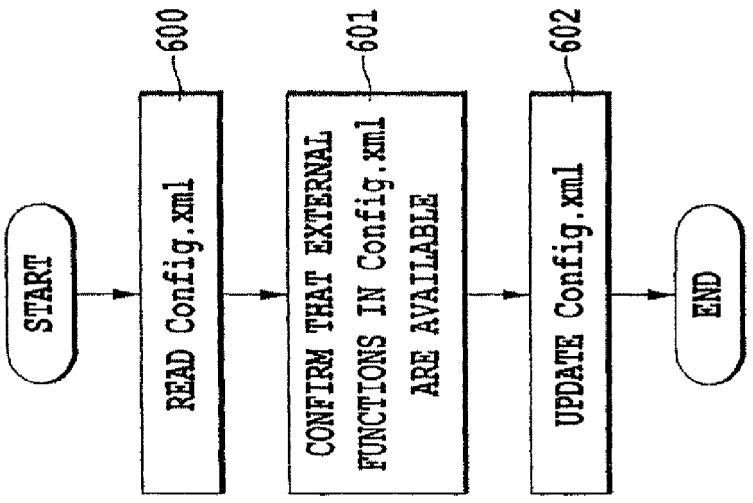


FIG.7F(ii)

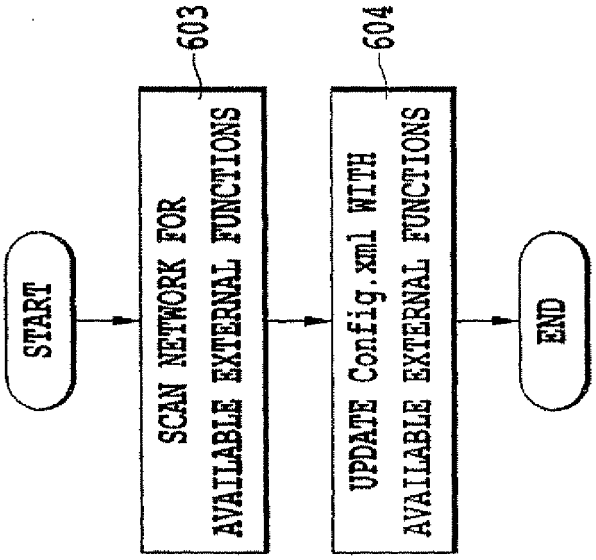


FIG.7F(iii)

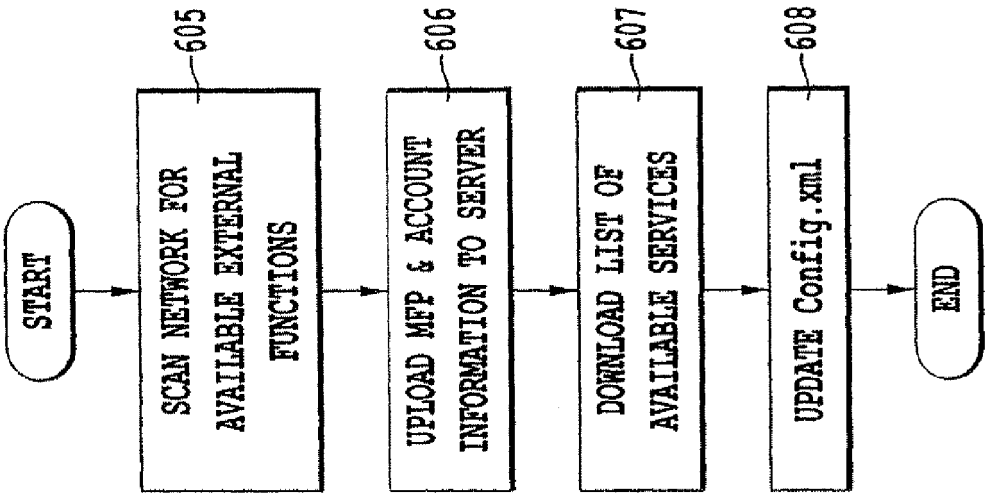


FIG.8A

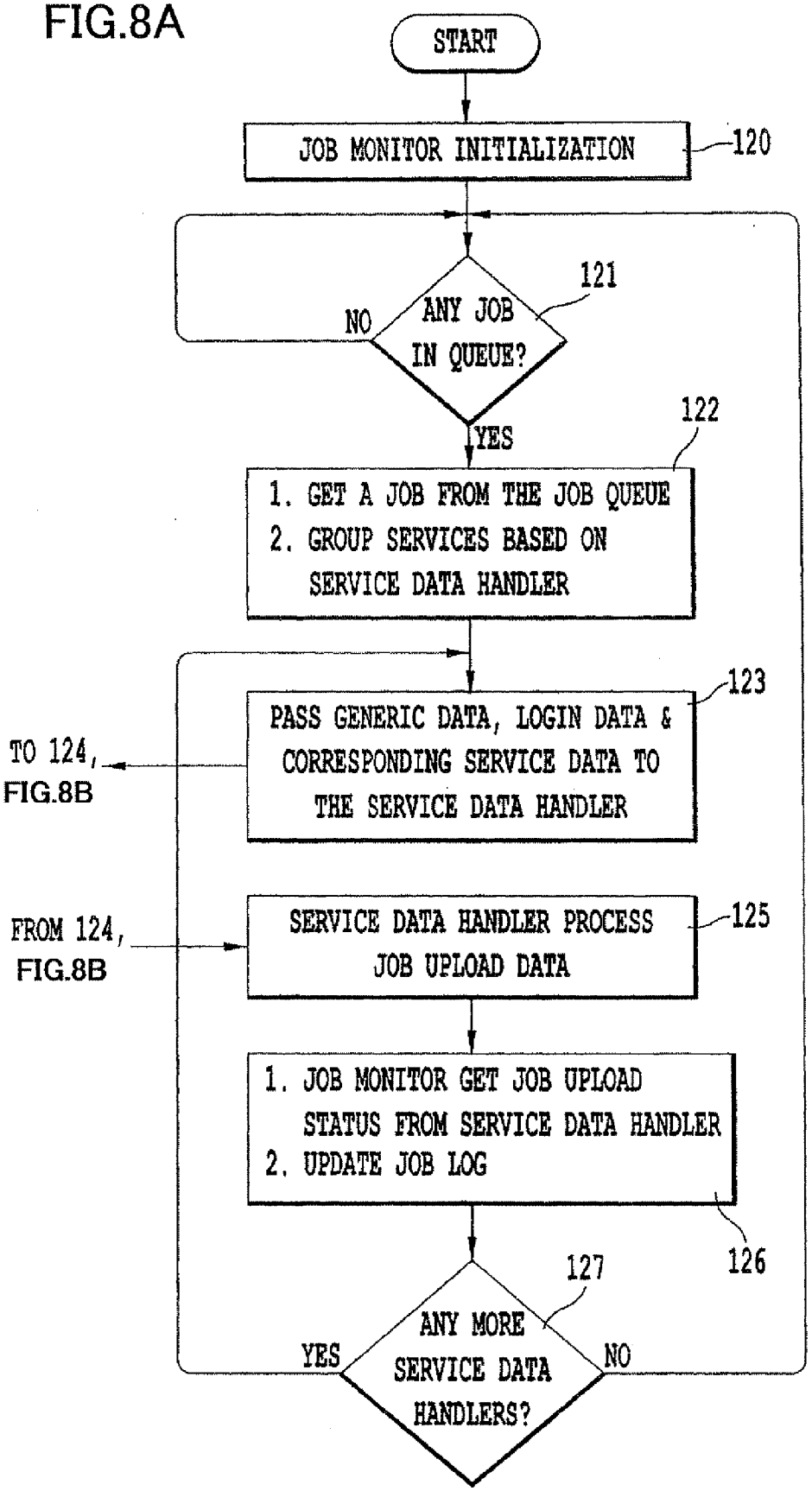

START
JOB MONITOR INITIALIZATION
120
121
ANY JOB IN QUEUE?
NO
YES
122
1. GET A JOB FROM THE JOB QUEUE
2. GROUP SERVICES BASED ON SERVICE DATA HANDLER
123
PASS GENERIC DATA, LOGIN DATA & CORRESPONDING SERVICE DATA TO THE SERVICE DATA HANDLER
TO 124, FIG.8B
FROM 124, FIG.8B
SERVICE DATA HANDLER PROCESS JOB UPLOAD DATA
125
1. JOB MONITOR GET JOB UPLOAD STATUS FROM SERVICE DATA HANDLER
2. UPDATE JOB LOG
126
127
ANY MORE SERVICE DATA HANDLERS?
YES
NO

FIG.8B

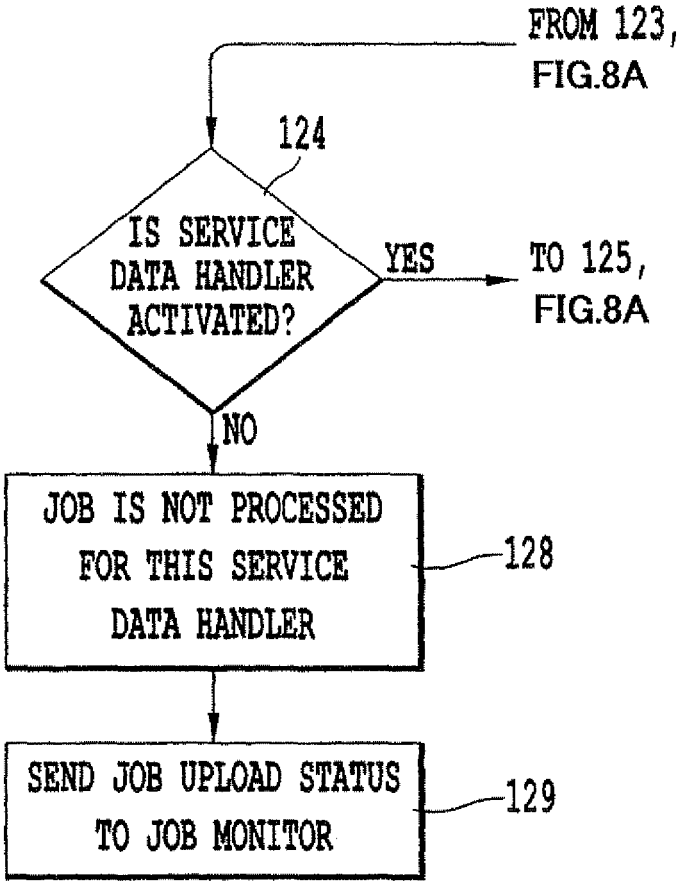

FROM 123, FIG.8A
124
IS SERVICE DATA HANDLER ACTIVATED?
YES
TO 125, FIG.8A
NO
JOB IS NOT PROCESSED FOR THIS SERVICE DATA HANDLER
128
SEND JOB UPLOAD STATUS TO JOB MONITOR
129

FIG.9

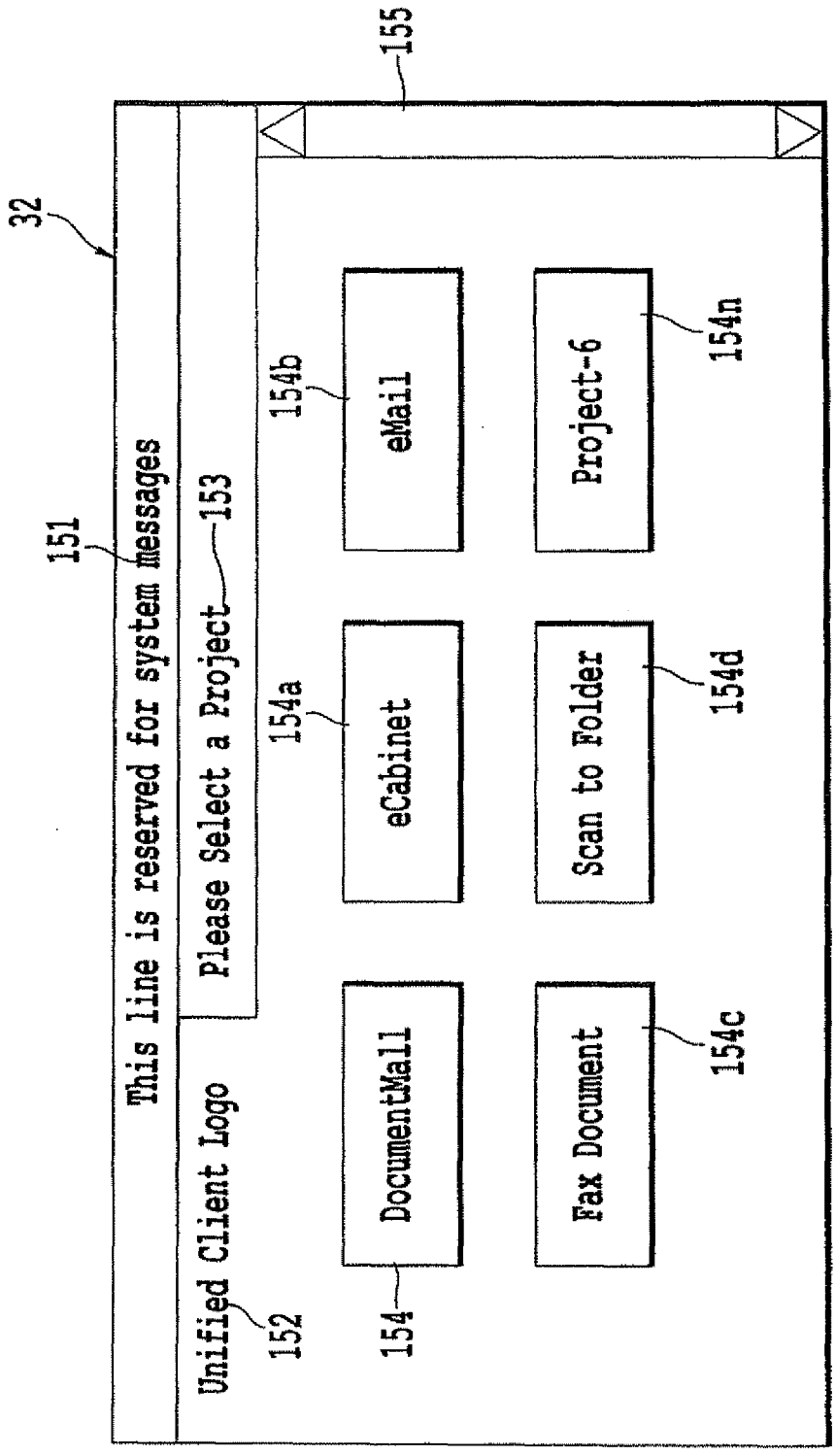

32
151
This line is reserved for system messages
Unified Client Logo
152
Please Select a Project
153
155
154a
eCabinet
154b
eMail
154n
Project-6
154
DocumentMall
154c
Fax Document
154d
Scan to Folder

FIG.10

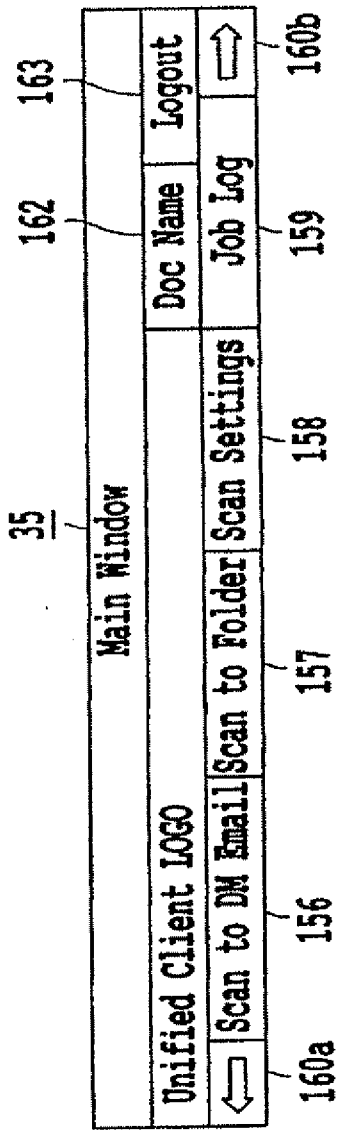

35
Main Window
Unified Client LOGO
Scan to DM Email
Scan to Folder
Scan Settings
Doc Name
Job Log
Logout
156
157
158
159
160a
160b
162
163

FIG.11A

```
<root>
    <JarFileList>
        <Jar>eCabinet.jar</Jar>
        <Jar>DocumentMall.jar</jar>
        <Jar>EmbeddedEmail.jar</jar>
    </JarFileList>
<MFP>
    <MFPSerialNo>ABC112223333</MFPSerialNo>
    <MACAddress>11-22-33-44-55-66</MACAddress>
    <AccountName>Ricoh</AccountName>
    <UserName>Ricoh User</UserName>
</MFP>
    <Project>
        <ProjectName>eCabinet</ProjectName>
        <DefaultScanSetting></DefaultScanSetting>
        <DefaultResolution>200</DefaultResolution>
        <DefaultDoubleSidedScan>false</DefaultDoubleSid
        edScan>
        <HasLogin>false</HasLogin>
        <LoginClass></LoginClass>
        <Service>
            <ServiceName>eCabinet</ServiceName>
            <DisplayName>eCabinet Owner</DisplayName>
            <Activation>
            <ActivationRequired>N</ActivationRequired>
                <Activated>Y</Activated>
                <ActivationDate>1-22-07</ActivationDate>
                <ExpirationDate></ExpirationDate>
            </Activation>
            <ServiceWindowClass>com.ricoh.UnifiedClient.eC
            abinet.ECabinetWindow.class</ServiceWindowCla
            ss>
            <DataHandlerClass>com.ricoh.UnifiedClient.eCabi
            net.ECabinetServiceDataHandler.class</DataHan
            dlerClass>
```

FIG.11B

```
<ConfigurationData>
<eCabinetServer>11.11.11.111</eCabinetServ
er>
<eCabinetServerPort>81</eCabinetServerPort>
</ConfigurationData>
<DataHandlerConfigurationData>

<eCabinetServer>11.11.11.111</eCabinetServ
er>
<ftpPort></ftpPort>
</DataHandlerConfigurationData>
</Service>
<Service>
<ServiceName>eCabinetFolder</ServiceName>
<DisplayName>eCabinet Folder</DisplayName>
<Activation>
<ActivationRequired>N</ActivationRequired>
<Activated>Y</Activated>
<ActivationDate>1-22-07</ActivationDate>
<ExpirationDate></ExpirationDate>
</Activation>
<ServiceWindowClass>com.ricoh.UnifiedClient.eC
abinet.ECabinetFolderWindow.class</ServiceWin
dowClass>
<DataHandlerClass>com.ricoh.UnifiedClient.eCabi
net.ECabinetServiceDataHandler.class</DataHan
dlerClass>
<ConfigurationData>
<eCabinetServer>11.11.11.111</eCabinetServ
er>
<eCabinetServerPort>81</eCabinetServerPort>
</ConfigurationData>
<DataHandlerConfigurationData>
<eCabinetServer>11.11.11.111</eCabinetSer
ver>
```

FIG.11C

```
                <ftpPort></ftpPort>
            </DataHandlerConfigurationData>
        </Service>
    </Project>

<Project>
    <ProjectName>DocumentMall</ProjectName>
    <DefaultScanSetting></DefaultScanSetting>
    <DefaultResolution>200</DefaultResolution>
    <DefaultDoubleSidedScan>true</DefaultDoubleSidedS
    can>
    <HasLogin>true</HasLogin>
    <LoginClass>com.ricoh.UnifiedClient.DocumentMall.D
    MLoginWindow.class</LoginClass>
    <Service>
        <ServiceName>DMEmail</ServiceName>
        <DisplayName>DM Email</DisplayName>
        <Activation>
        <ActivationRequired>Y</ActivationRequired>
            <Activated>Y</Activated>
            <ActivationDate>1-22-07</ActivationDate>
            <ExpirationDate>4-22-07</ExpirationDate>
        </Activation>
        <ServiceWindowClass>com.ricoh.UnifiedClient.Do
        cumentMall.DMEmailWindow.class</ServiceWind
        owClass>
        <DataHandlerClass>com.ricoh.UnifiedClient.Docu
        mentMall.DMServiceDataHandler.class</DataHan
        dlerClass>
                <ConfigurationData>
        <DMServer>documentmall.com</eCabinetServer>
                </ConfigurationData>
                <DataHandlerConfigurationData>
            <DMServer>documentmall.com</eCabinetSe
        rver>
```

FIG.11D

```
1          </DataHandlerConfigurationData>
2       </Service>
3       <Service>
4          <ServiceName>DMFolder</ServiceName>
5             <DisplayName>DM Folder</DisplayName>
6           <Activation>
7
8          <ActivationRequired>N</ActivationRequired>
9             <Activated>Y</Activated>
10            <ActivationDate>1-22-07</ActivationDate>
11            <ExpirationDate></ExpirationDate>
12         </Activation>
13         <ServiceWindowClass>com.ricoh.UnifiedClient.Doc
14         umentMall.DMFolderWindow.class</ServiceWindo
15         wClass>
16         <DataHandlerClass>com.ricoh.UnifiedClient.Docum
17         entMall.DMServiceDataHandler.class</DataHandler
18         Class>
19         <ConfigurationData>
20           <DMServer>documentmall.com</DMServer>
21         </ConfigurationData>
22         <DataHandlerConfigurationData>
23           <DMServer>documentmall.com</DMServer>
24         </DataHandlerConfigurationData>
25       </Service>
26    </Project>
27
28    <Project>
29    <ProjectName>Email</ProjectName>
30    <DefaultScanSetting></DefaultScanSetting>
31    <DefaultResolution>200</DefaultResolution>
32    <DefaultDoubleSidedScan>true</DefaultDoubleSidedSc
33    an>
34    <HasLogin>false</HasLogin>
35    <LoginClass> </LoginClass>
```

FIG.11E

```
<Service>
    <ServiceName>Embedded Email</ServiceName>
    <DisplayName>Email</DisplayName>
    <Activation>
    <ActivationRequired>Y</ActivationRequired>
        <Activated>Y</Activated>
        <ActivationDate>1-22-07</ActivationDate>
        <ExpirationDate>4-22-07</ExpirationDate>
    </Activation>
    <External>
        <Embedded>Y</Embedded>
        <ExternalAddress></ExternalAddress>
    <Priority>2</Priority>
    </External>
    <ServiceWindowClass>com.ricoh.UnifiedClient.E
    mail.EmailWindow.class</ServiceWindowClass>
    <DataHandlerClass>com.ricoh.UnifiedClient.Email
    .EmailServiceDataHandler.class</DataHandlerCla
    ss>
    <ConfigurationData>
        <LDAPServer> </ LDAPServer>
    </ConfigurationData>
    <DataHandlerConfigurationData>
    </DataHandlerConfigurationData>
 </Service>
<Service>
    <ServiceName>GlobalScanEmail</ServiceName>
    <DisplayName>Email</DisplayName>
    <Activation>
    <ActivationRequired>Y</ActivationRequired>
        <Activated>Y</Activated>
        <ActivationDate>1-22-07</ActivationDate>
        <ExpirationDate>4-22-07</ExpirationDate>
    </Activation>
    <External>
```

FIG.11F

```
1                   <Embedded>N</Embedded>
2                   <ExternalAddress>GlobalscanServer</Extern
3               alAddress>
4               <Priority>1</Priority>
5               </External>
6           </Service>
7           </Project>
8       </root>
```

FIG.12

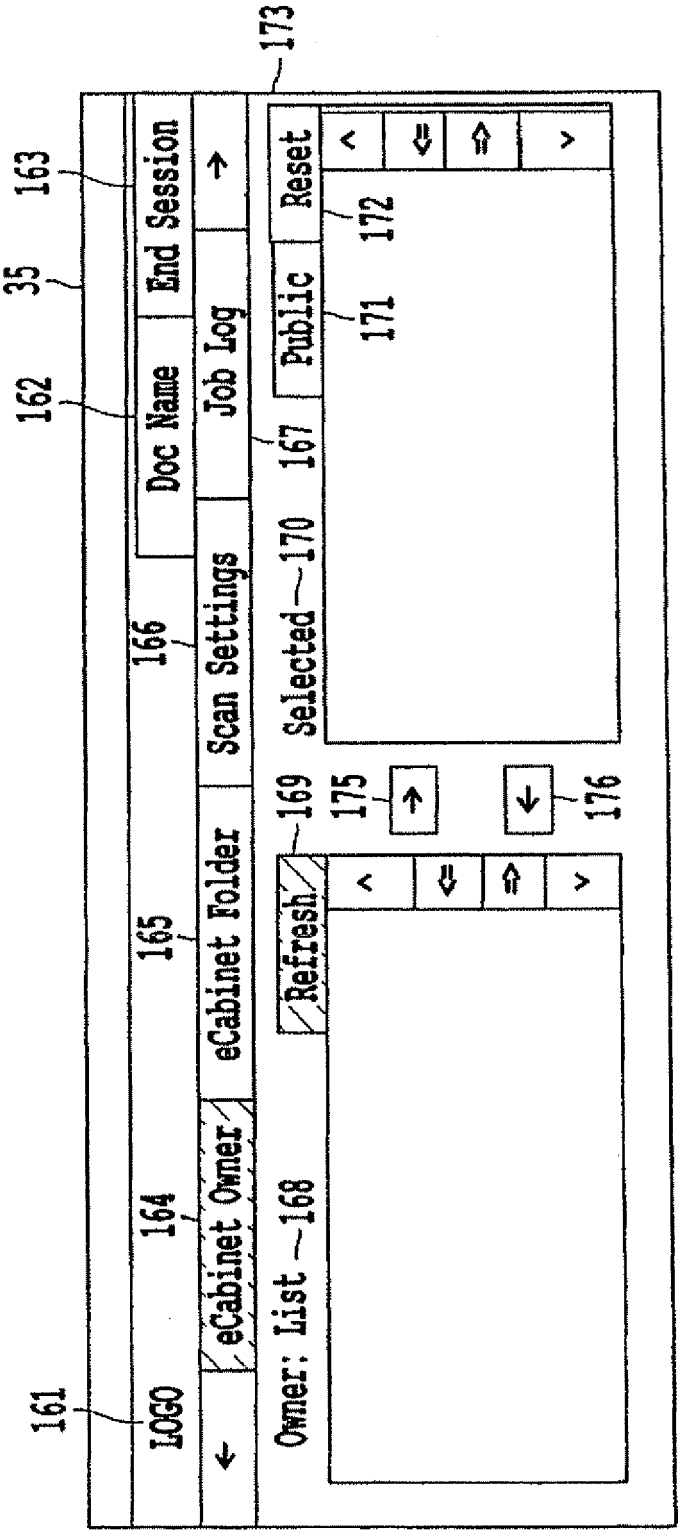

161
LOGO
164
eCabinet Owner
165
eCabinet Folder
166
Scan Settings
162
Doc Name
167
Job Log
35
163
End Session
173
Reset
172
Public
171
Selected
170
169
Refresh
Owner: List
168
175
176

FIG.13

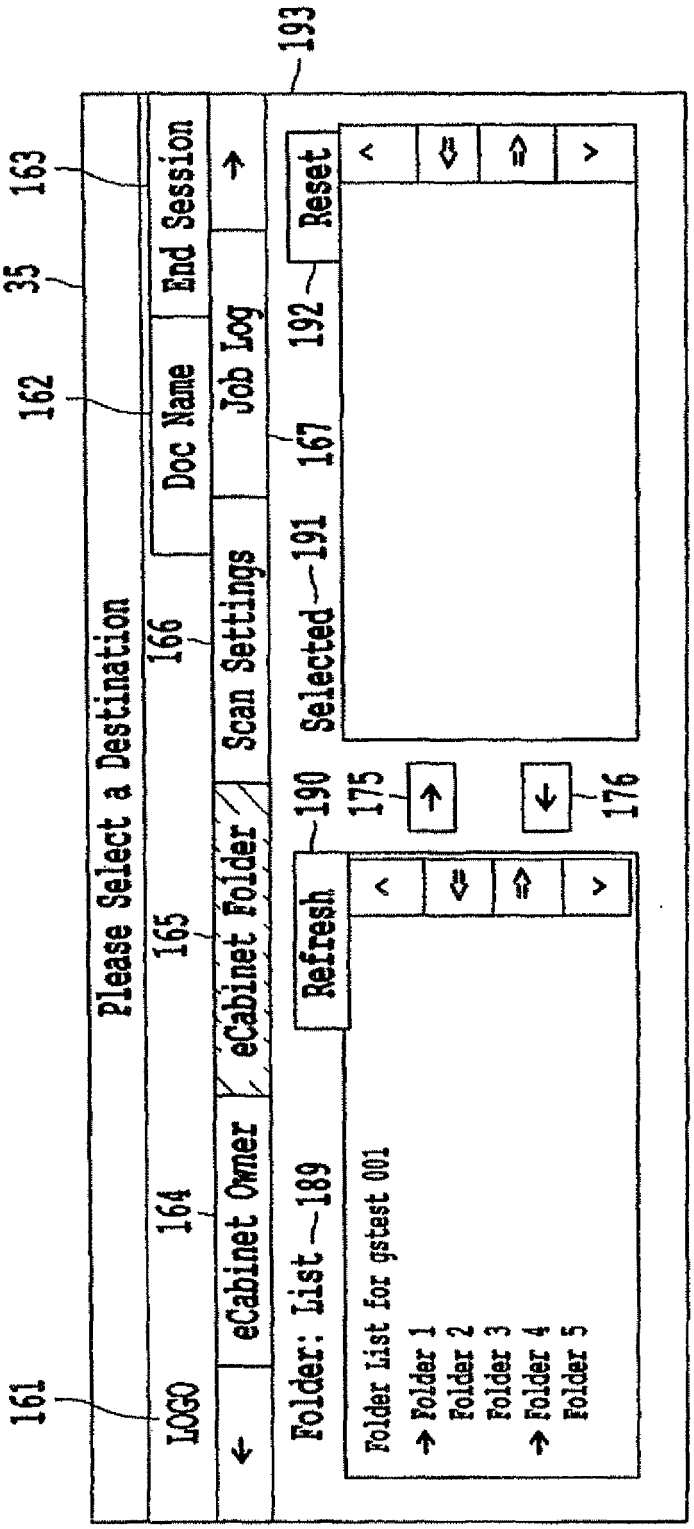

193
163
35
162
161
Please Select a Destination
LOGO
164
eCabinet Owner
165
eCabinet Folder
166
Scan Settings
Doc Name
Job Log
End Session
Folder: List
189
Refresh
190
Folder List for gstest 001
Folder 1
Folder 2
Folder 3
Folder 4
Folder 5
175
176
Selected
191
167
192
Reset

FIG.14

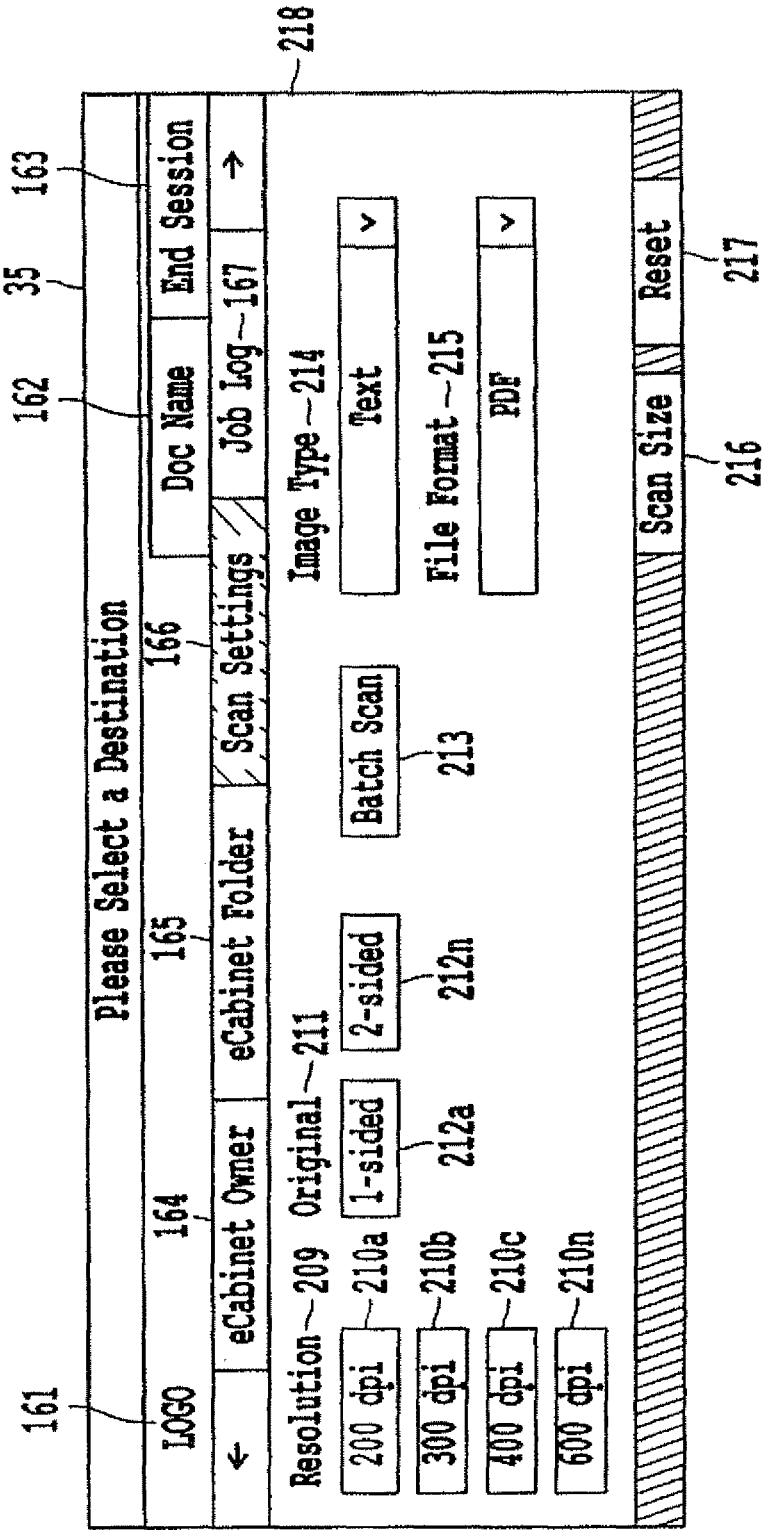

Please Select a Destination
LOGO
eCabinet Owner
eCabinet Folder
Scan Settings
Job Log
End Session
Doc Name
Image Type
Text
File Format
PDF
Batch Scan
Resolution
200 dpi
300 dpi
400 dpi
600 dpi
Original
1-sided
2-sided
Scan Size
Reset
161
162
163
164
165
166
167
209
210a
210b
210c
210n
211
212a
212n
213
214
215
216
217
218
35

FIG.15

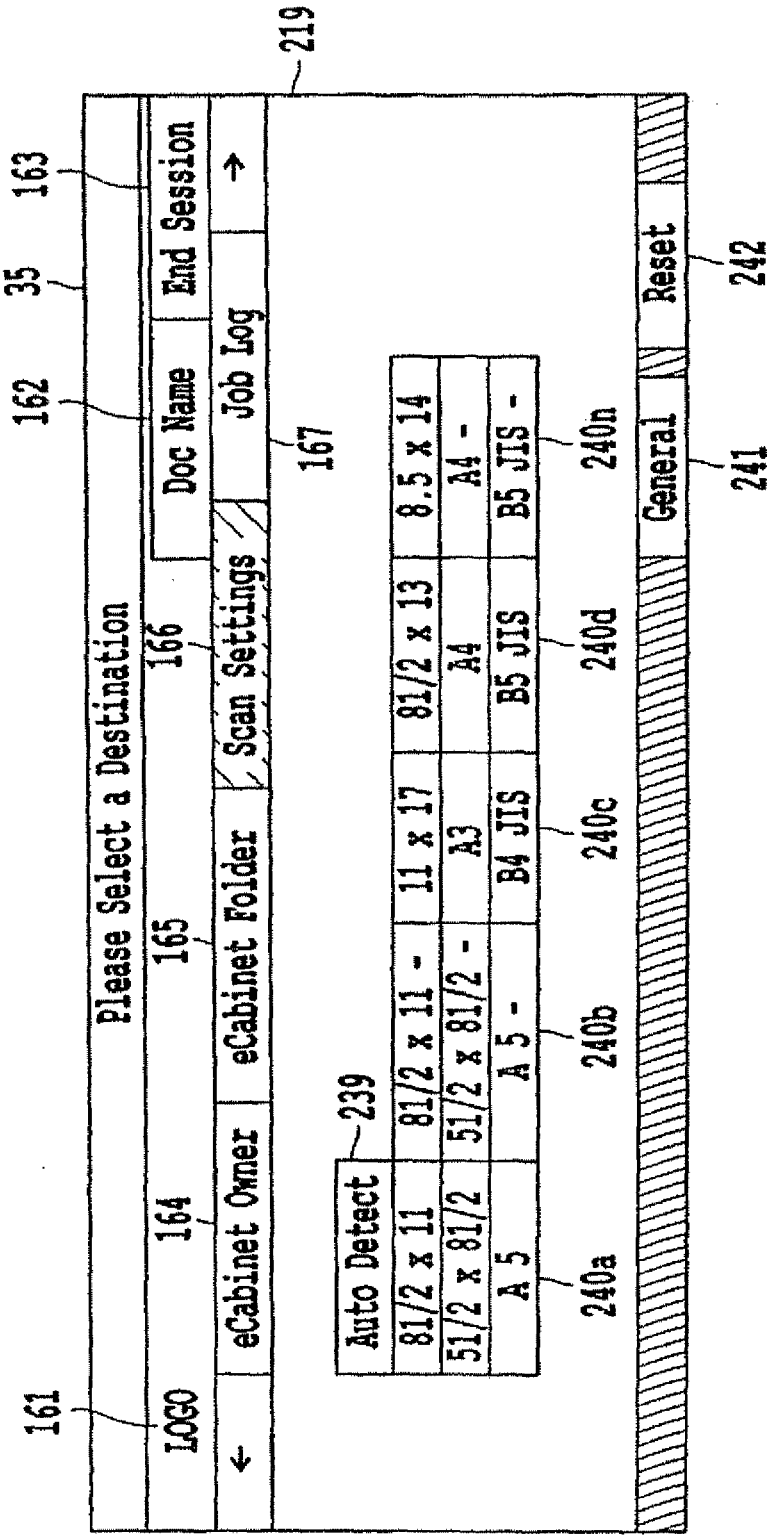

219
163
35
162
161
Please Select a Destination
LOGO
164
eCabinet Owner
165
eCabinet Folder
166
Scan Settings
Doc Name
End Session
Job Log
167
Auto Detect
239
81/2 x 11
51/2 x 81/2
A 5
240a
81/2 x 11 -
51/2 x 81/2 -
A 5 -
240b
11 x 17
A3
B4 JIS
240c
81/2 x 13
A4
B5 JIS
240d
8.5 x 14
A4 -
B5 JIS -
240n
General
241
Reset
242

FIG.16

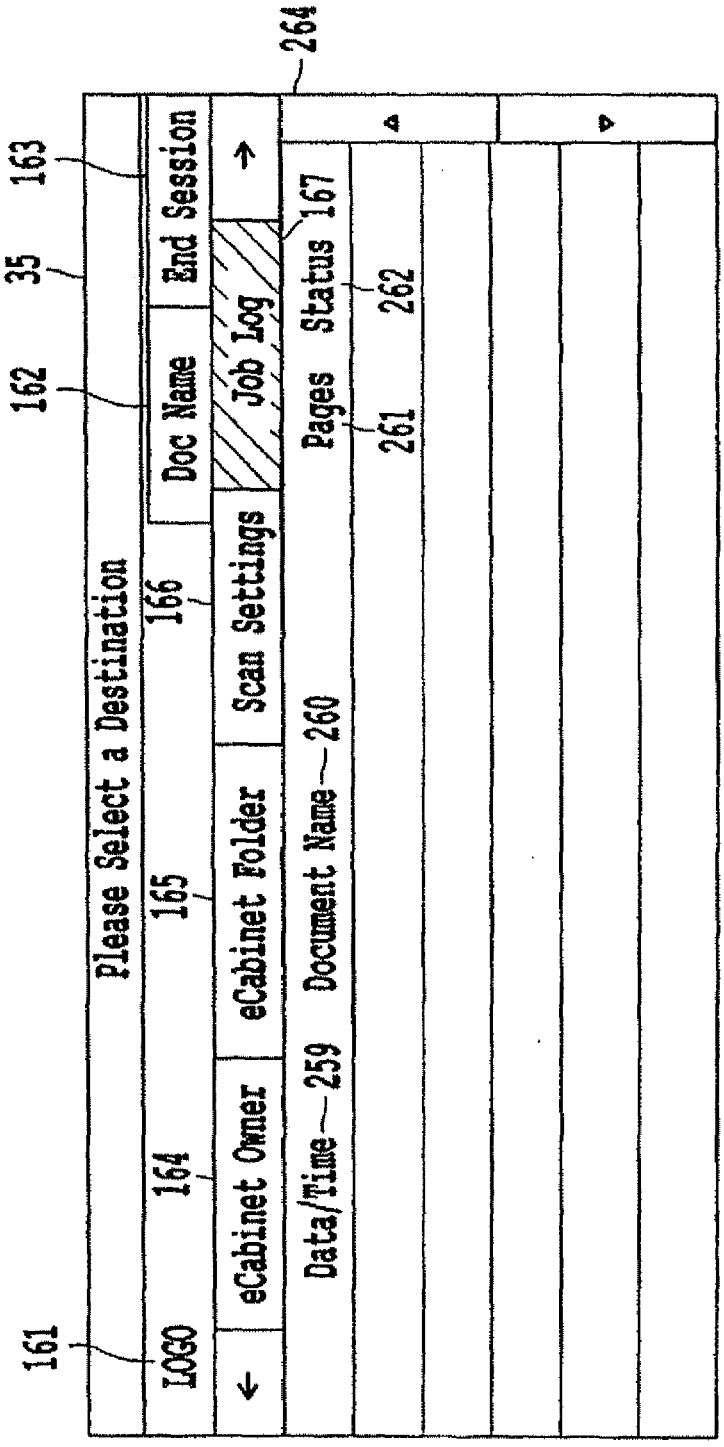

161
163
35
162
264
Please Select a Destination
LOGO
164
eCabinet Owner
165
eCabinet Folder
166
Scan Settings
End Session
Doc Name
Job Log
Data/Time
259
Document Name
260
Pages
261
Status
262
167

FIG.17

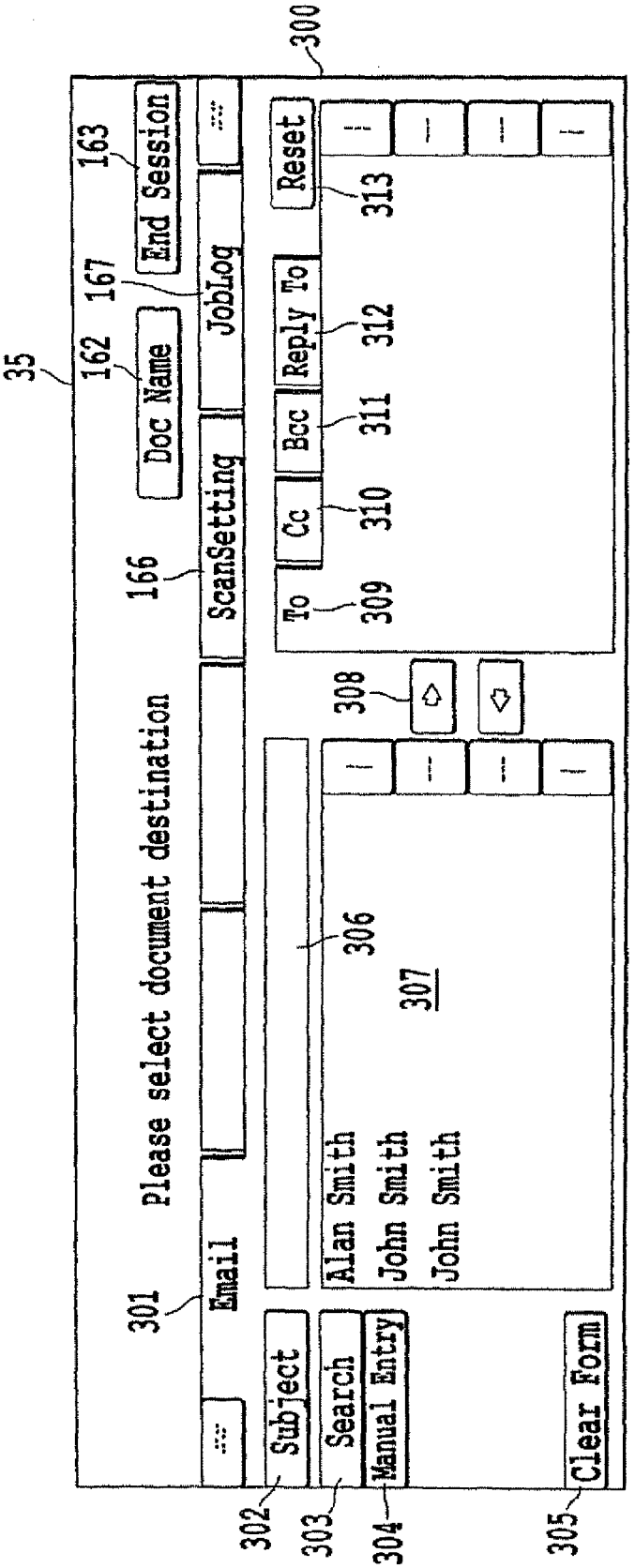

300
35
Please select document destination
Email
Subject
Search
Manual Entry
Clear Form
Alan Smith
John Smith
John Smith
306
307
301
302
303
304
305
308
ScanSetting
166
Doc Name
162
JobLog
167
End Session
163
To
309
Cc
310
Bcc
311
Reply To
312
Reset
313

FIG.18

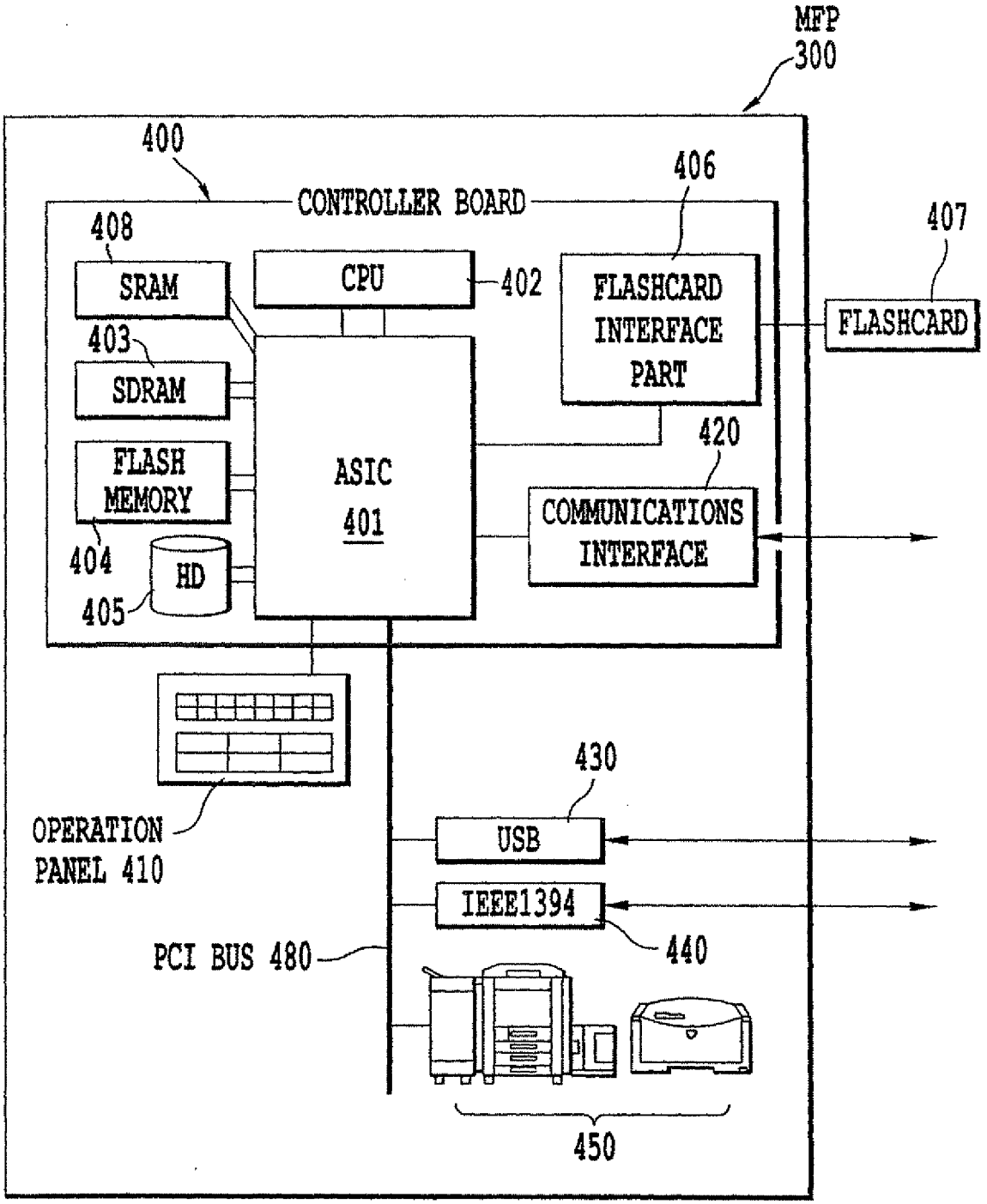

MFP
300
400
CONTROLLER BOARD
406
407
408
SRAM
CPU
402
FLASHCARD INTERFACE PART
FLASHCARD
403
SDRAM
FLASH MEMORY
ASIC
401
420
COMMUNICATIONS INTERFACE
404
HD
405
430
OPERATION PANEL 410
USB
IEEE1394
440
PCI BUS 480
450

FIG.19 RELATED ART

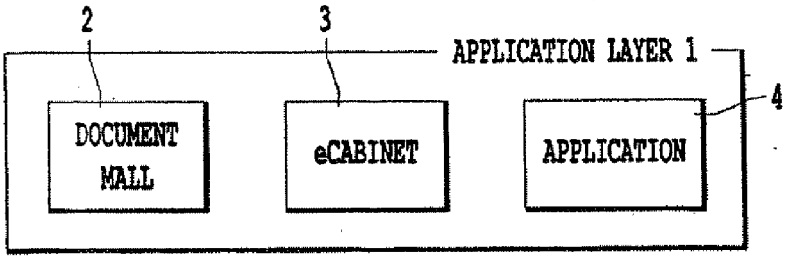


FIG.20

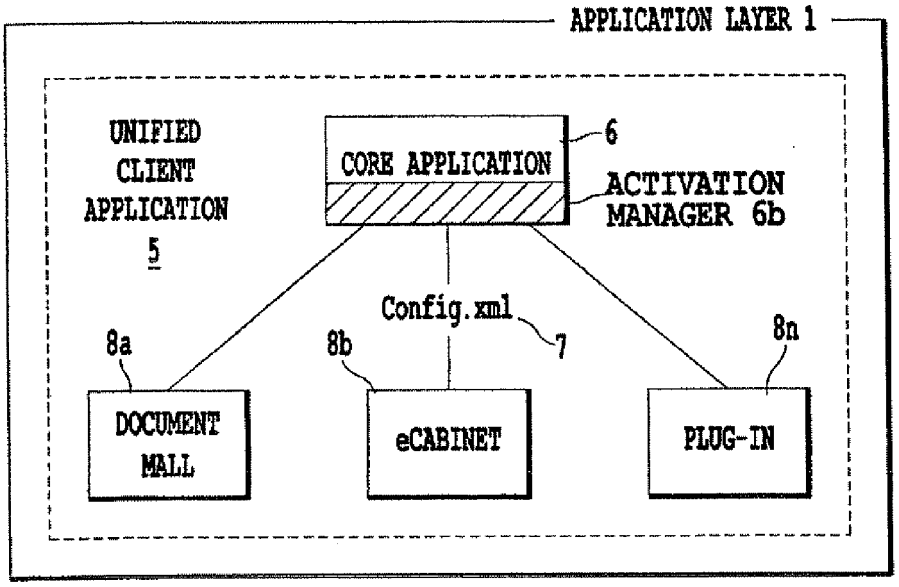


FIG.21

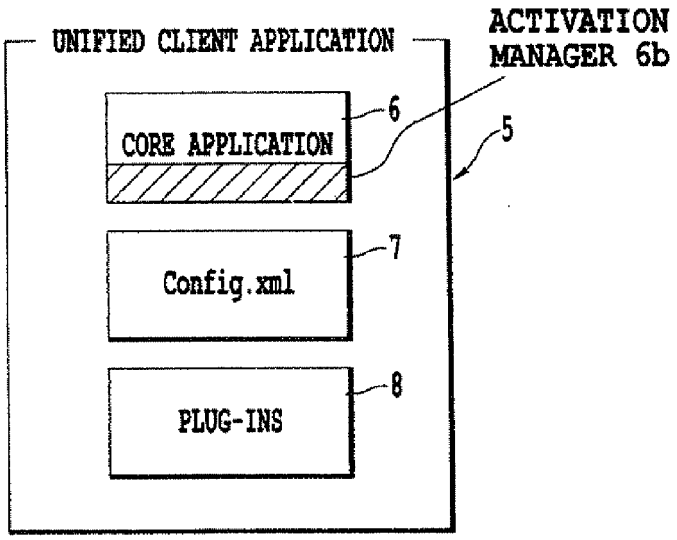


FIG.22

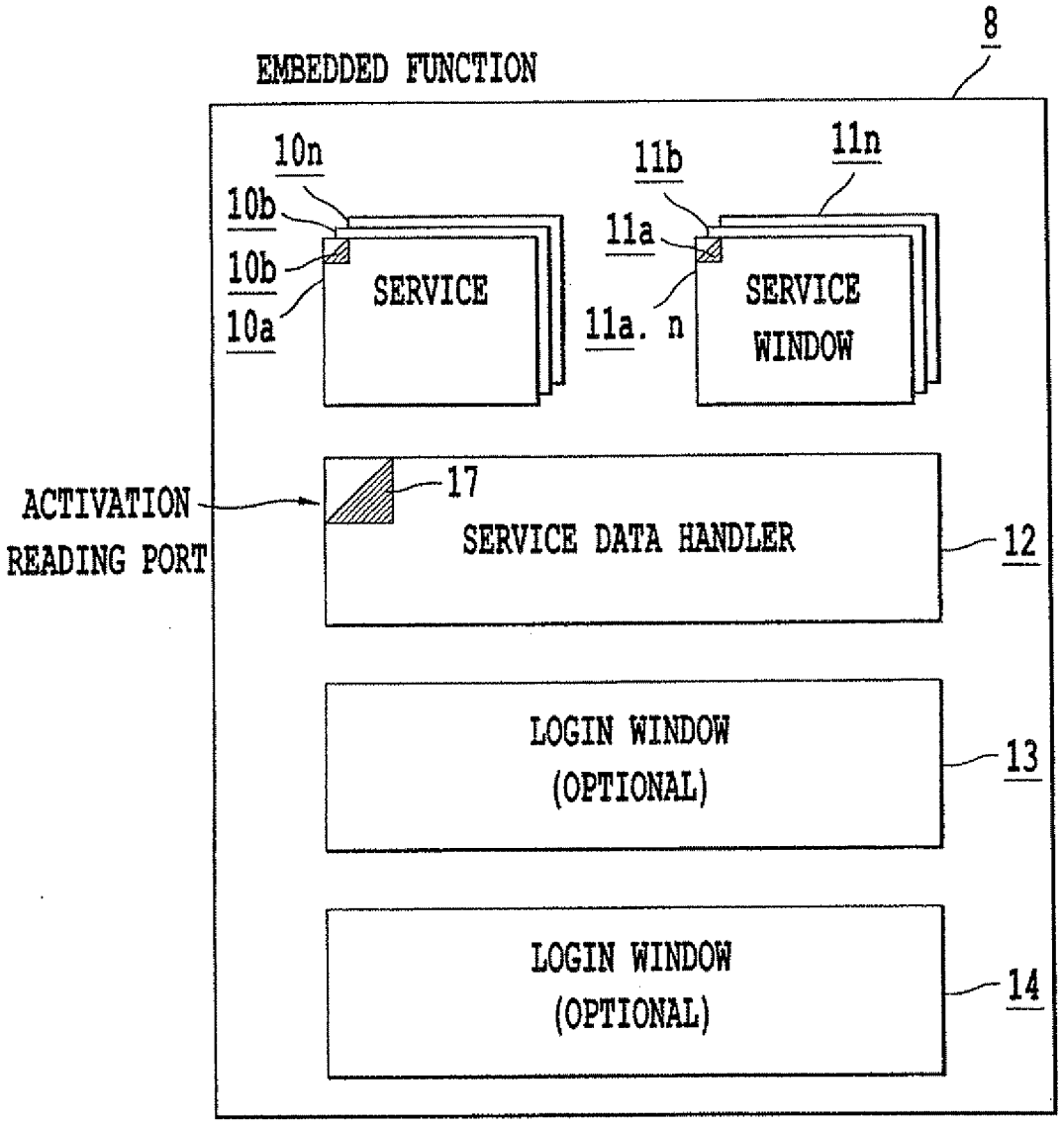

8
EMBEDDED FUNCTION
10n
10b
10b
10a
SERVICE
11b
11n
11a
11a. n
SERVICE WINDOW
ACTIVATION READING PORT
17
SERVICE DATA HANDLER
12
LOGIN WINDOW (OPTIONAL)
13
LOGIN WINDOW (OPTIONAL)
14

FIG.23

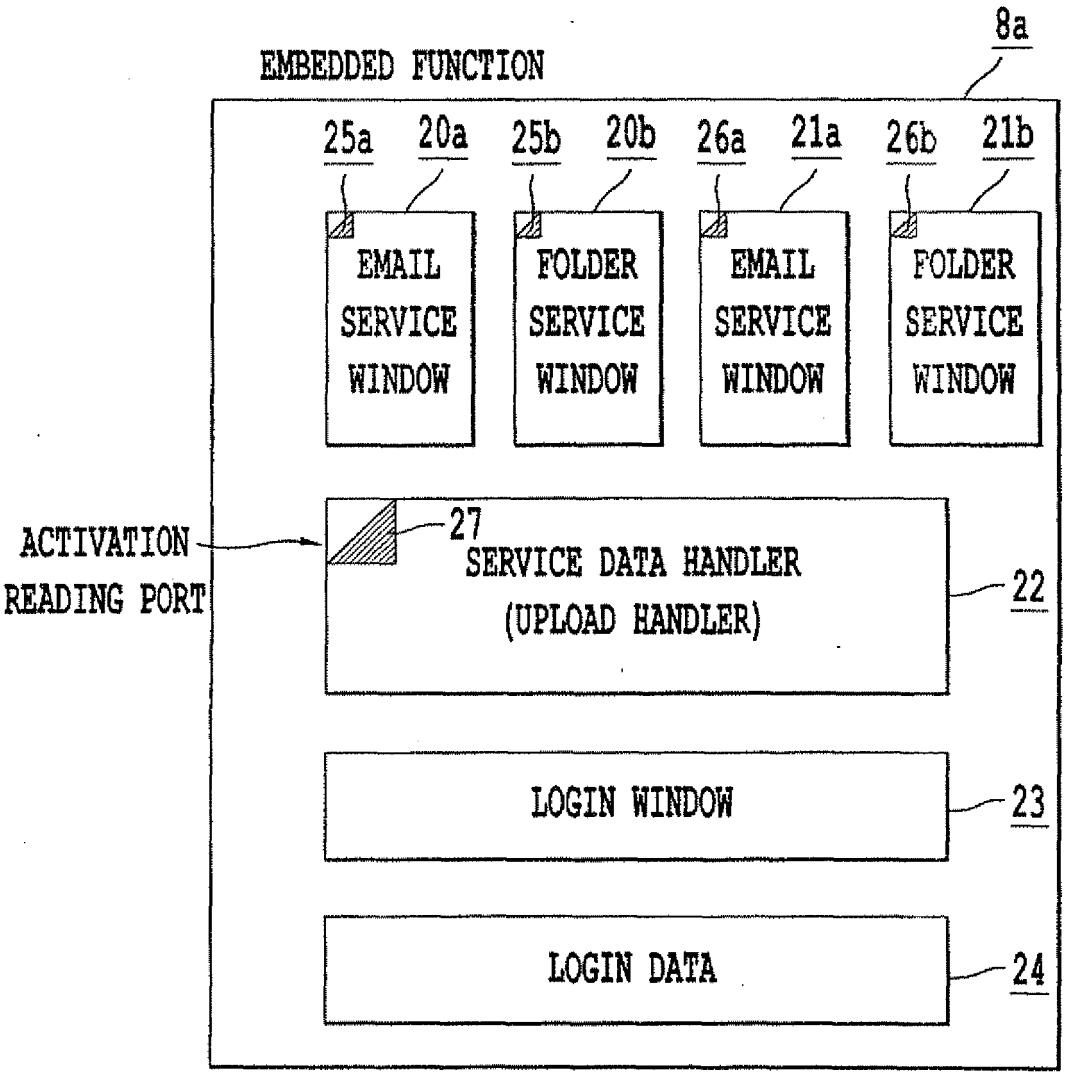

8a
EMBEDDED FUNCTION
25a
20a
25b
20b
26a
21a
26b
21b
EMAIL SERVICE WINDOW
FOLDER SERVICE WINDOW
EMAIL SERVICE WINDOW
FOLDER SERVICE WINDOW
ACTIVATION READING PORT
27
SERVICE DATA HANDLER (UPLOAD HANDLER)
22
LOGIN WINDOW
23
LOGIN DATA
24

FIG.24A

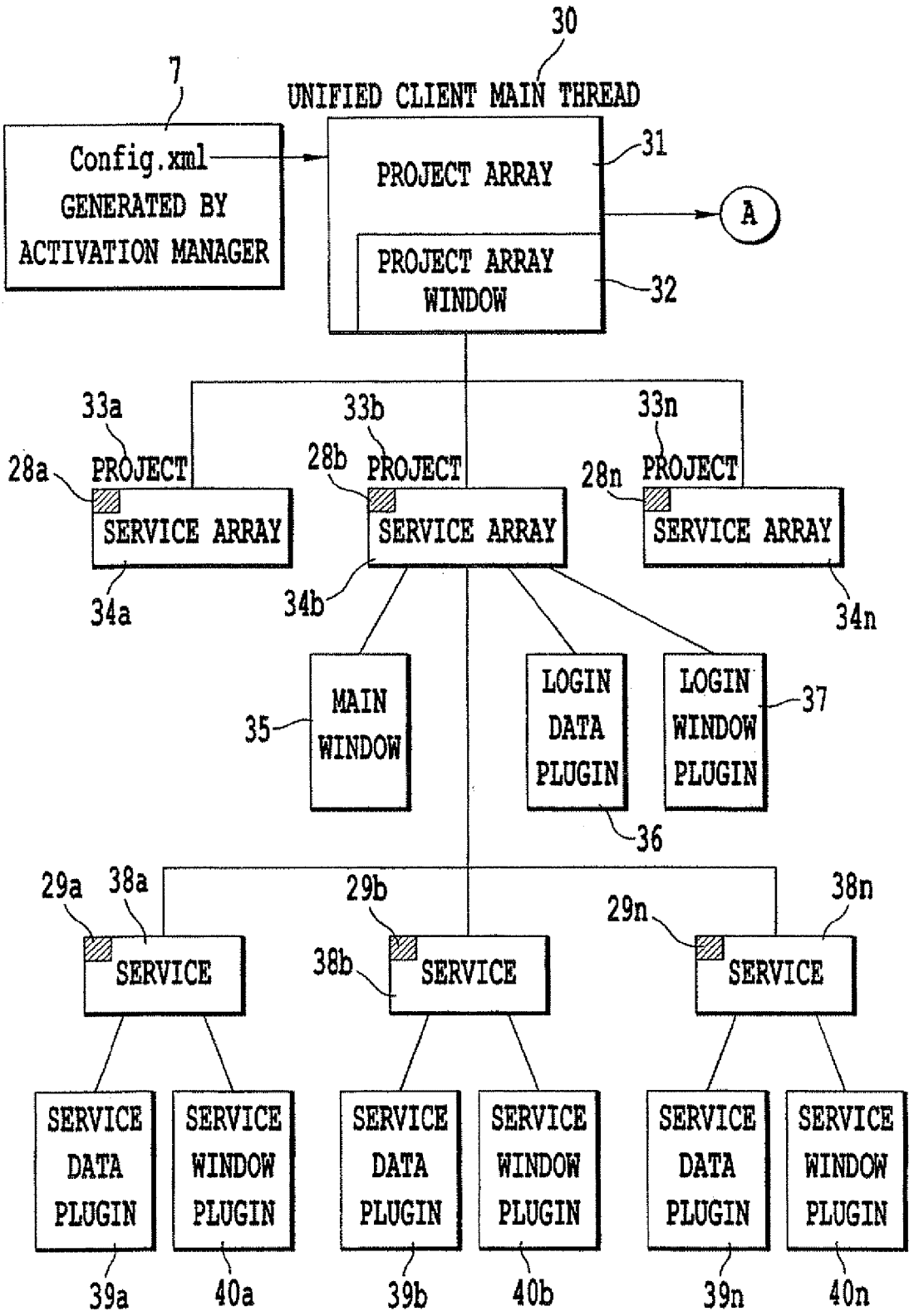

30
7
UNIFIED CLIENT MAIN THREAD
Config.xml
GENERATED BY
ACTIVATION MANAGER
PROJECT ARRAY
31
A
PROJECT ARRAY
WINDOW
32
33a
33b
33n
28a
PROJECT
28b
PROJECT
28n
PROJECT
SERVICE ARRAY
SERVICE ARRAY
SERVICE ARRAY
34a
34b
34n
MAIN
WINDOW
35
LOGIN
DATA
PLUGIN
LOGIN
WINDOW
PLUGIN
37
36
29a
38a
29b
38n
29n
38b
SERVICE
SERVICE
SERVICE
SERVICE
DATA
PLUGIN
SERVICE
WINDOW
PLUGIN
SERVICE
DATA
PLUGIN
SERVICE
WINDOW
PLUGIN
SERVICE
DATA
PLUGIN
SERVICE
WINDOW
PLUGIN
39a
40a
39b
40b
39n
40n

FIG.24B

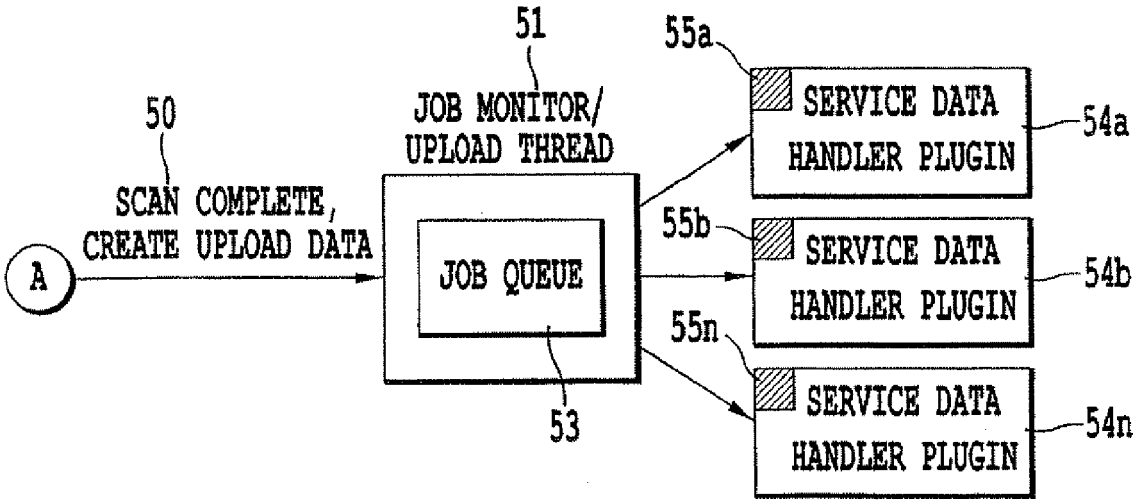

51
55a
50
JOB MONITOR/
UPLOAD THREAD
SERVICE DATA
HANDLER PLUGIN
54a
SCAN COMPLETE,
CREATE UPLOAD DATA
55b
A
JOB QUEUE
SERVICE DATA
HANDLER PLUGIN
54b
55n
SERVICE DATA
HANDLER PLUGIN
54n
53

FIG.25A

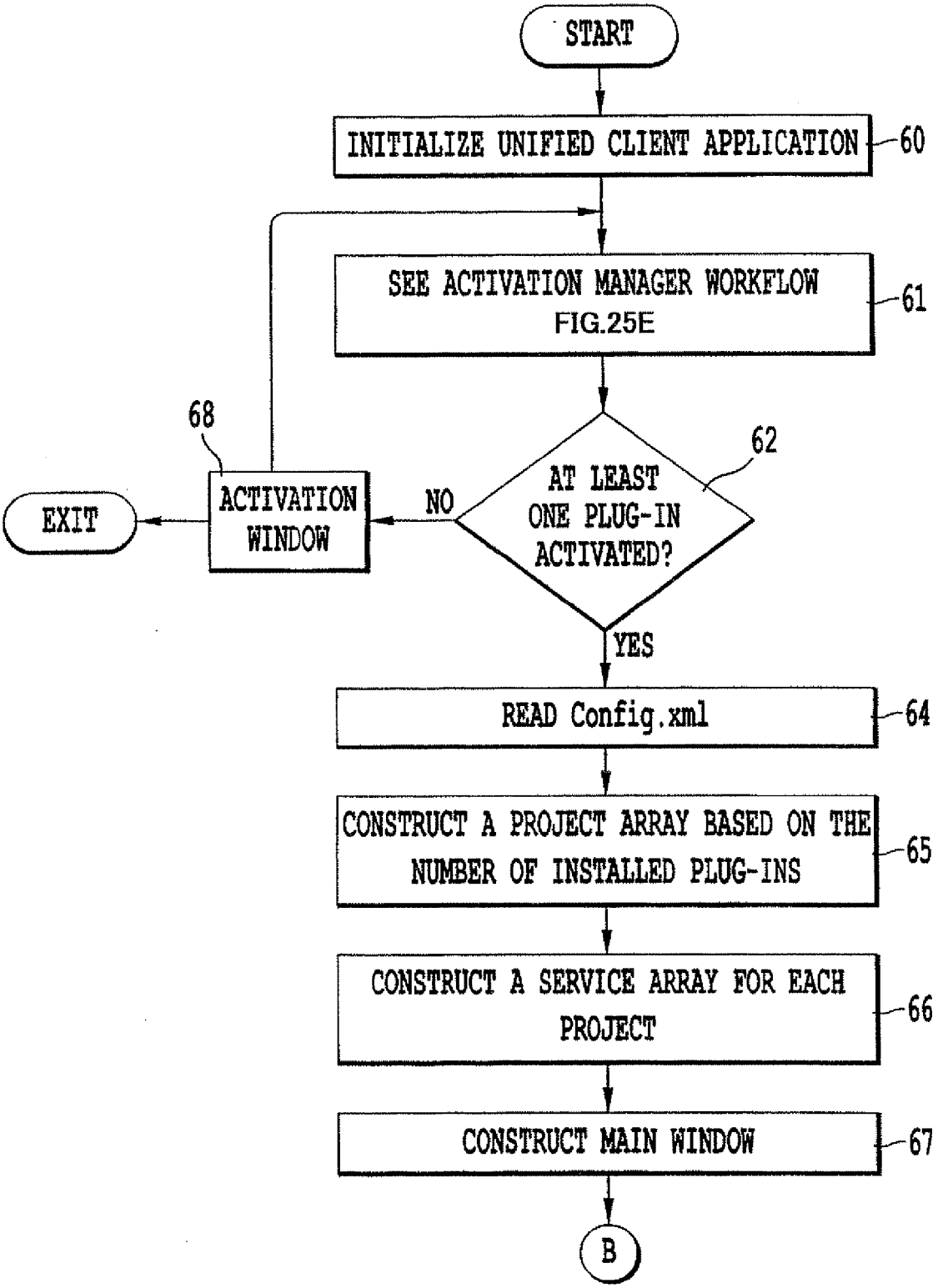

START
INITIALIZE UNIFIED CLIENT APPLICATION
60
SEE ACTIVATION MANAGER WORKFLOW FIG.25E
61
62
AT LEAST ONE PLUG-IN ACTIVATED?
NO
YES
68
ACTIVATION WINDOW
EXIT
READ Config.xml
64
CONSTRUCT A PROJECT ARRAY BASED ON THE NUMBER OF INSTALLED PLUG-INS
65
CONSTRUCT A SERVICE ARRAY FOR EACH PROJECT
66
CONSTRUCT MAIN WINDOW
67
B

FIG.25B

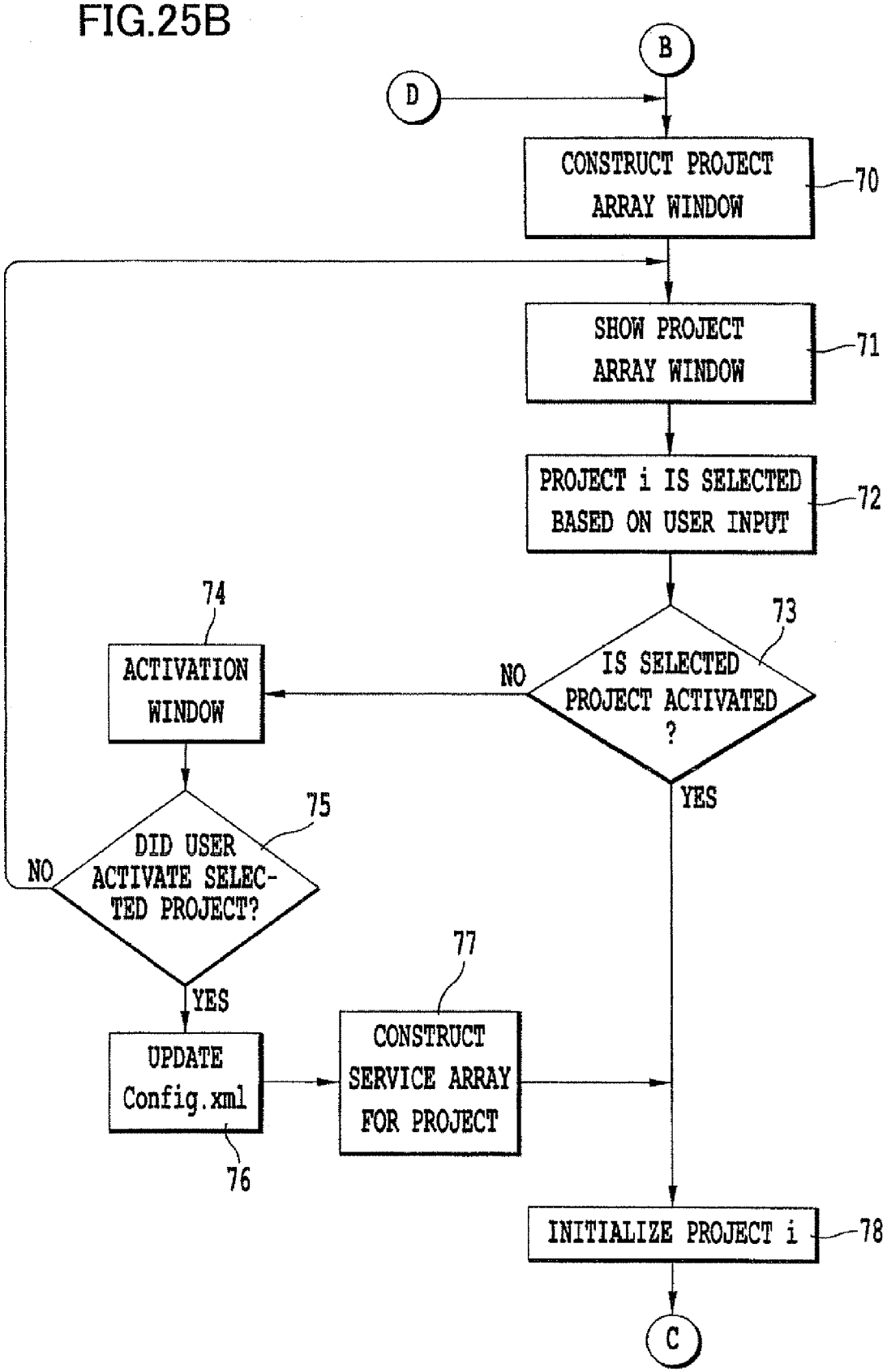

B
D
CONSTRUCT PROJECT ARRAY WINDOW
70
SHOW PROJECT ARRAY WINDOW
71
PROJECT i IS SELECTED BASED ON USER INPUT
72
IS SELECTED PROJECT ACTIVATED ?
73
NO
YES
ACTIVATION WINDOW
74
DID USER ACTIVATE SELECTED PROJECT?
75
NO
YES
UPDATE Config.xml
76
CONSTRUCT SERVICE ARRAY FOR PROJECT
77
INITIALIZE PROJECT i
78
C

FIG.25C

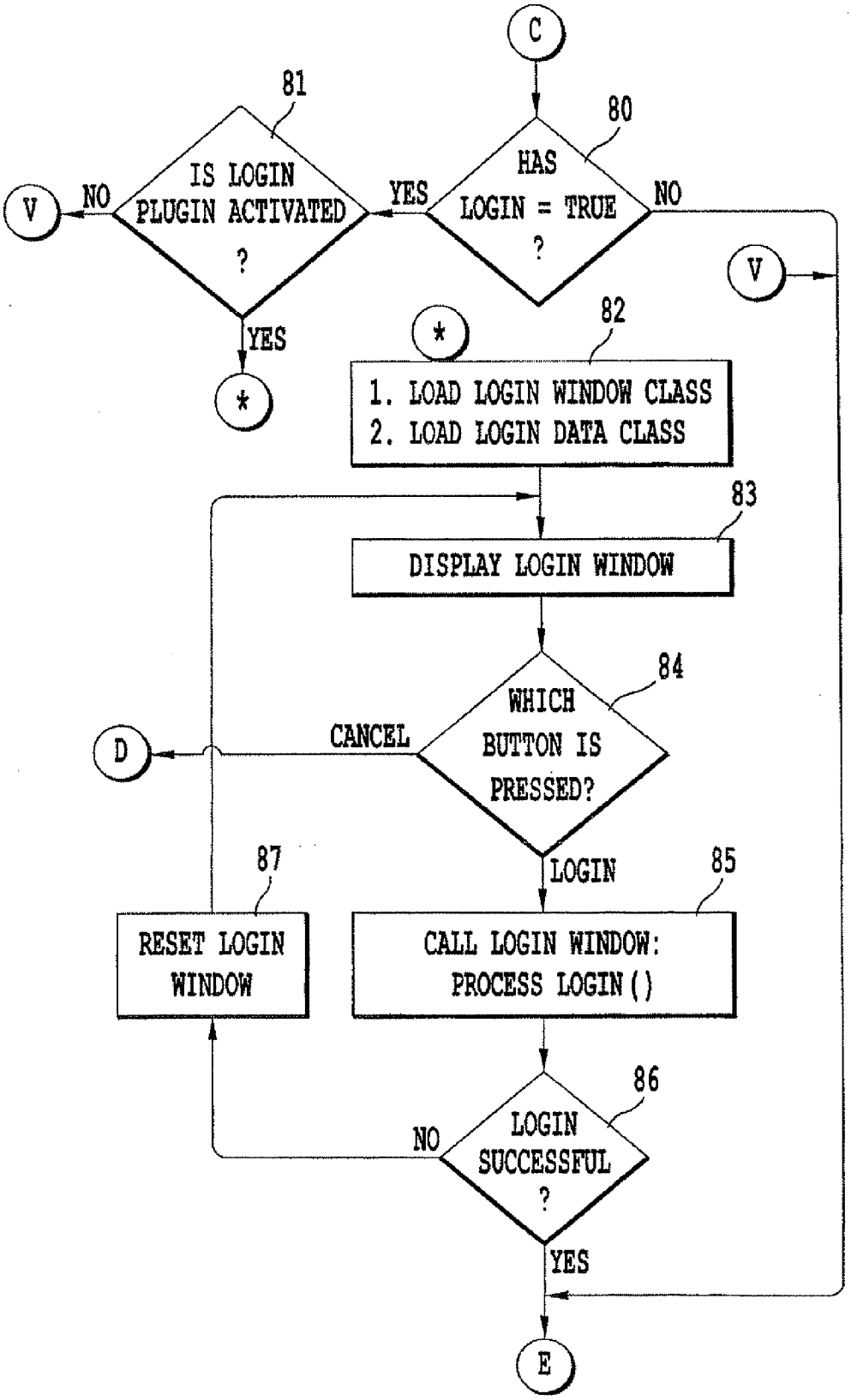

C
80
HAS LOGIN = TRUE ?
NO
YES
81
IS LOGIN PLUGIN ACTIVATED ?
NO
V
YES
*
V
*
82
1. LOAD LOGIN WINDOW CLASS
2. LOAD LOGIN DATA CLASS
83
DISPLAY LOGIN WINDOW
84
WHICH BUTTON IS PRESSED?
CANCEL
D
LOGIN
85
CALL LOGIN WINDOW: PROCESS LOGIN ()
87
RESET LOGIN WINDOW
86
LOGIN SUCCESSFUL ?
NO
YES
E

FIG.25D(i)

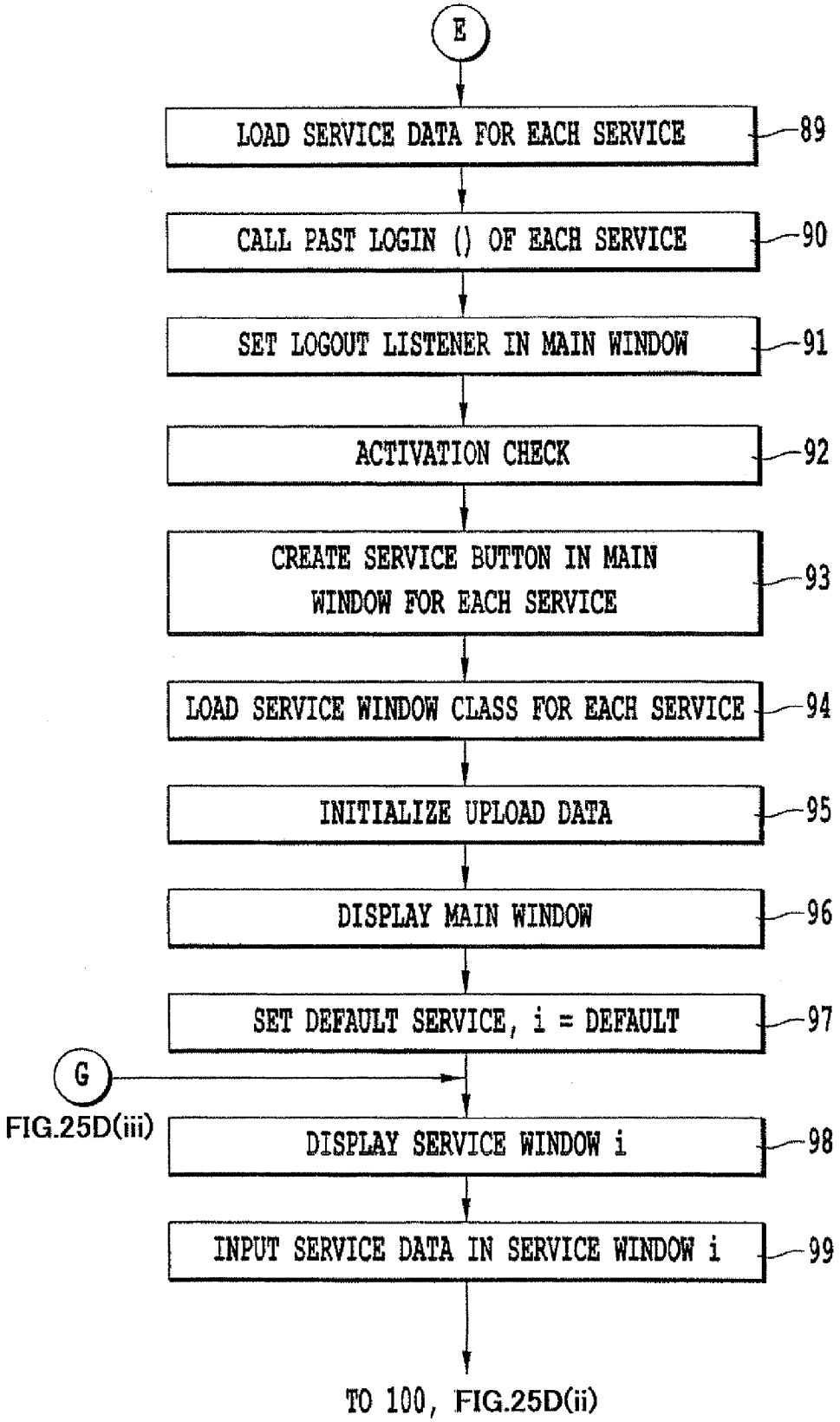


FIG.25D(iii)

FIG.25D(ii)

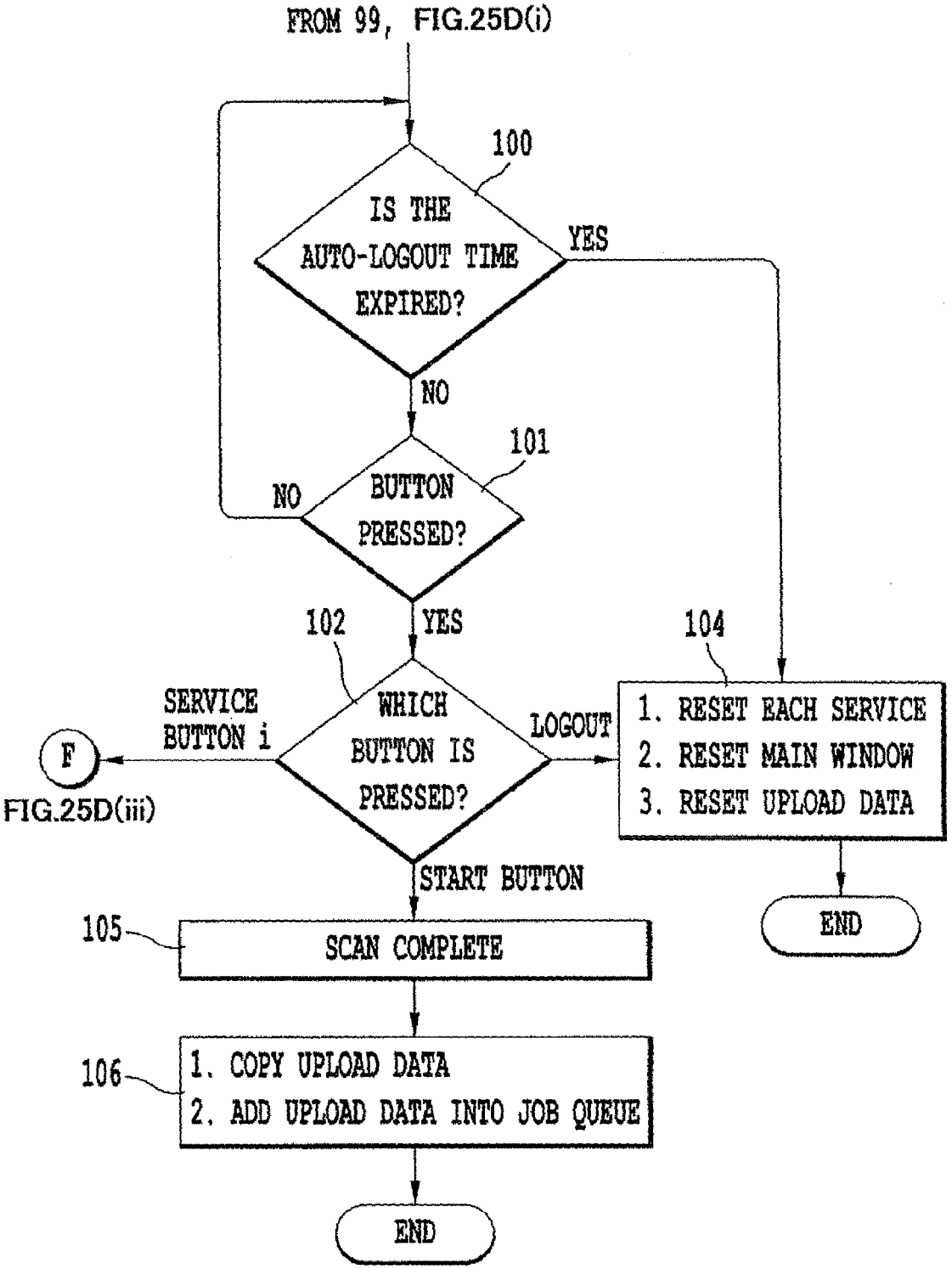

FROM 99, FIG.25D(i)
100
IS THE AUTO-LOGOUT TIME EXPIRED?
YES
NO
101
BUTTON PRESSED?
NO
YES
102
WHICH BUTTON IS PRESSED?
SERVICE BUTTON i
F
FIG.25D(iii)
LOGOUT
104
1. RESET EACH SERVICE
2. RESET MAIN WINDOW
3. RESET UPLOAD DATA
END
START BUTTON
105
SCAN COMPLETE
106
1. COPY UPLOAD DATA
2. ADD UPLOAD DATA INTO JOB QUEUE
END

FIG.25D(iii)

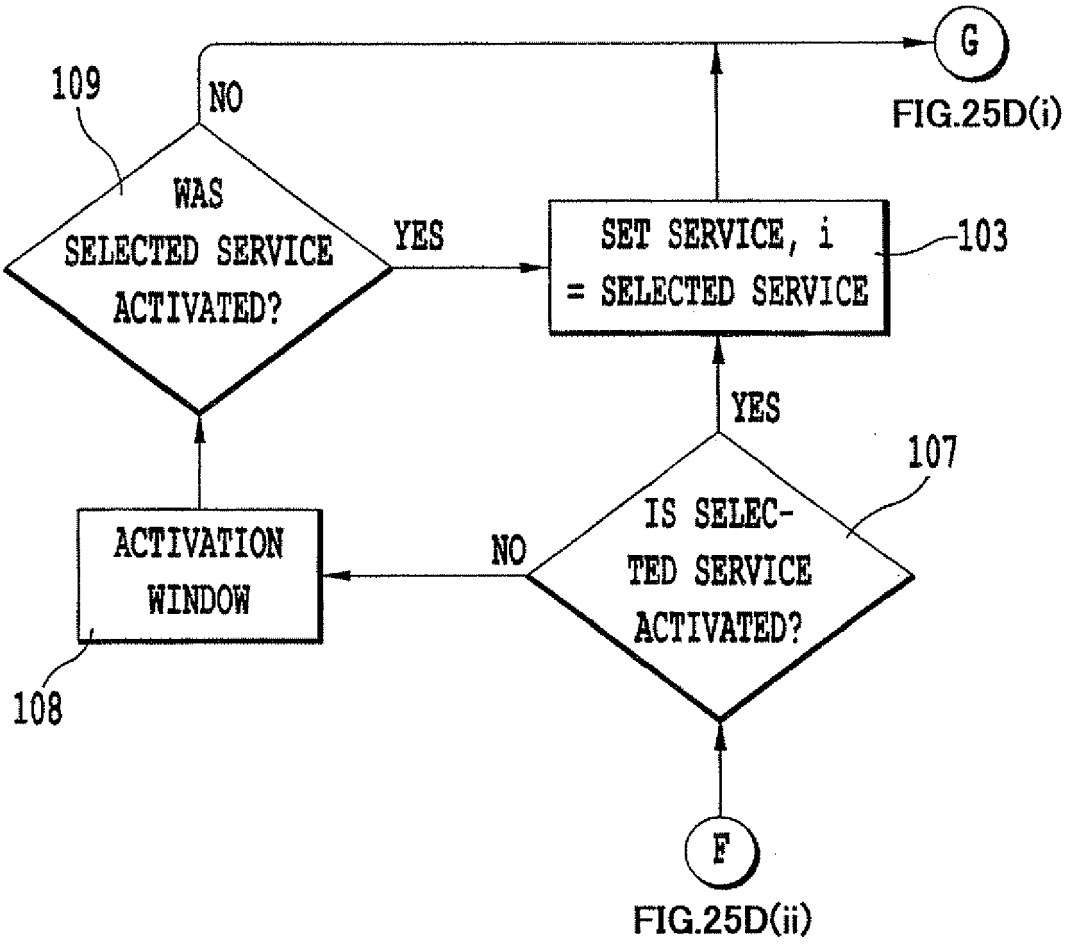

G
FIG.25D(i)
109
NO
WAS SELECTED SERVICE ACTIVATED?
YES
SET SERVICE, i = SELECTED SERVICE
103
YES
107
IS SELEC-TED SERVICE ACTIVATED?
NO
ACTIVATION WINDOW
108
F
FIG.25D(ii)

FIG.25E

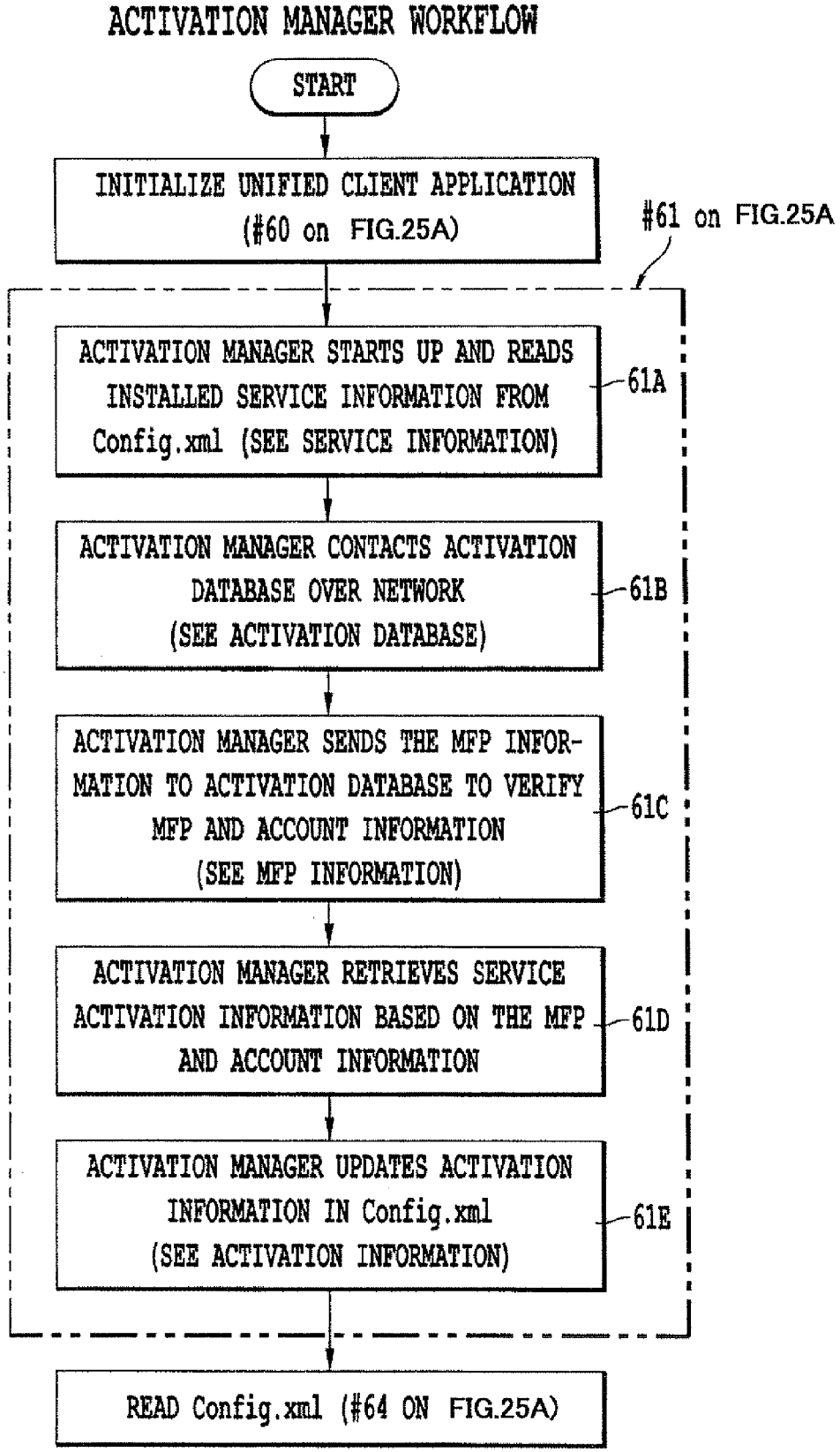

ACTIVATION MANAGER WORKFLOW
START
INITIALIZE UNIFIED CLIENT APPLICATION (#60 on FIG.25A)
#61 on FIG.25A
ACTIVATION MANAGER STARTS UP AND READS INSTALLED SERVICE INFORMATION FROM Config.xml (SEE SERVICE INFORMATION)
61A
ACTIVATION MANAGER CONTACTS ACTIVATION DATABASE OVER NETWORK (SEE ACTIVATION DATABASE)
61B
ACTIVATION MANAGER SENDS THE MFP INFORMATION TO ACTIVATION DATABASE TO VERIFY MFP AND ACCOUNT INFORMATION (SEE MFP INFORMATION)
61C
ACTIVATION MANAGER RETRIEVES SERVICE ACTIVATION INFORMATION BASED ON THE MFP AND ACCOUNT INFORMATION
61D
ACTIVATION MANAGER UPDATES ACTIVATION INFORMATION IN Config.xml (SEE ACTIVATION INFORMATION)
61E
READ Config.xml (#64 ON FIG.25A)

FIG.26A

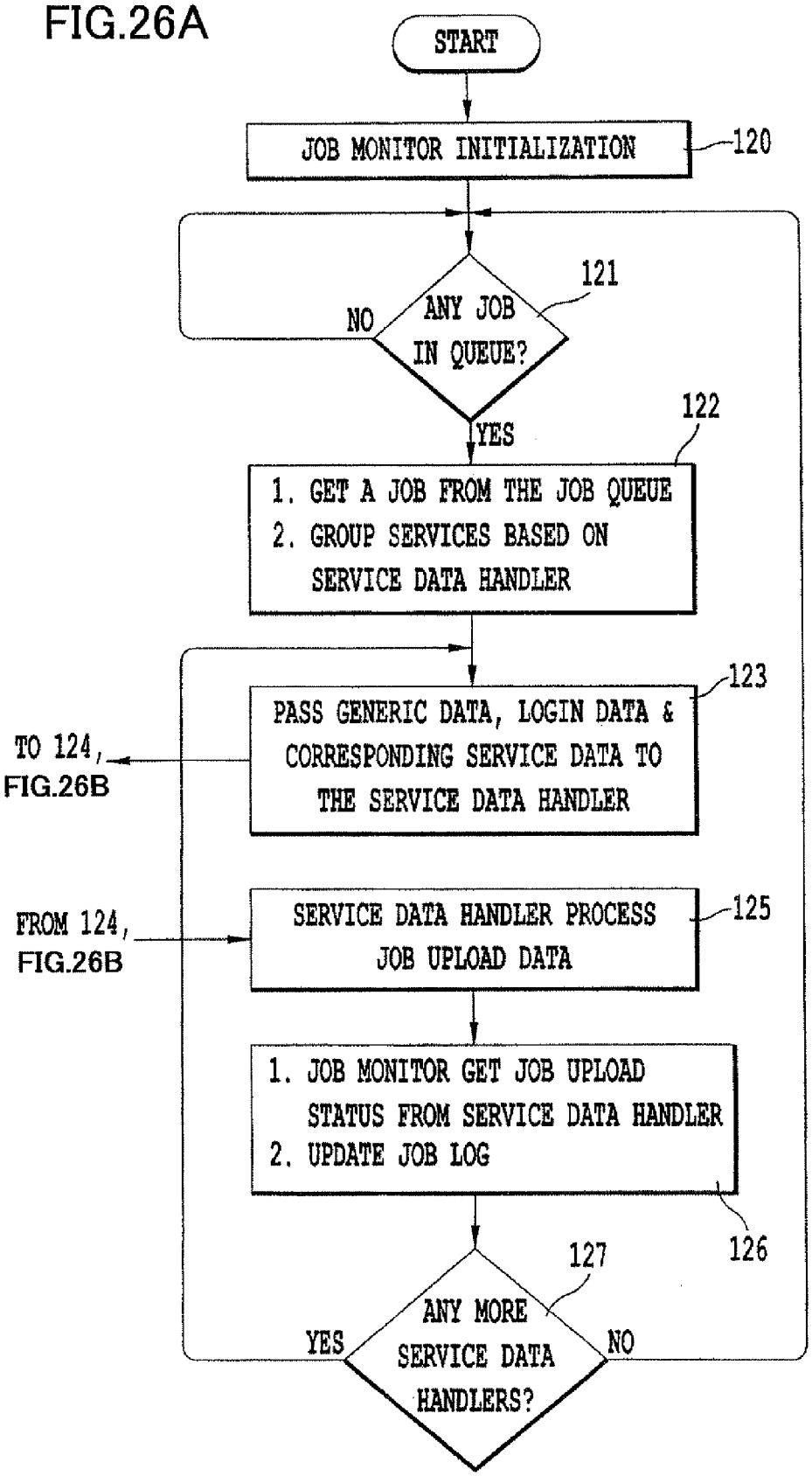

START
JOB MONITOR INITIALIZATION
120
ANY JOB IN QUEUE?
121
NO
YES
122
1. GET A JOB FROM THE JOB QUEUE
2. GROUP SERVICES BASED ON SERVICE DATA HANDLER
123
PASS GENERIC DATA, LOGIN DATA & CORRESPONDING SERVICE DATA TO THE SERVICE DATA HANDLER
TO 124, FIG.26B
FROM 124, FIG.26B
SERVICE DATA HANDLER PROCESS JOB UPLOAD DATA
125
1. JOB MONITOR GET JOB UPLOAD STATUS FROM SERVICE DATA HANDLER
2. UPDATE JOB LOG
126
127
ANY MORE SERVICE DATA HANDLERS?
YES
NO

FIG.26B

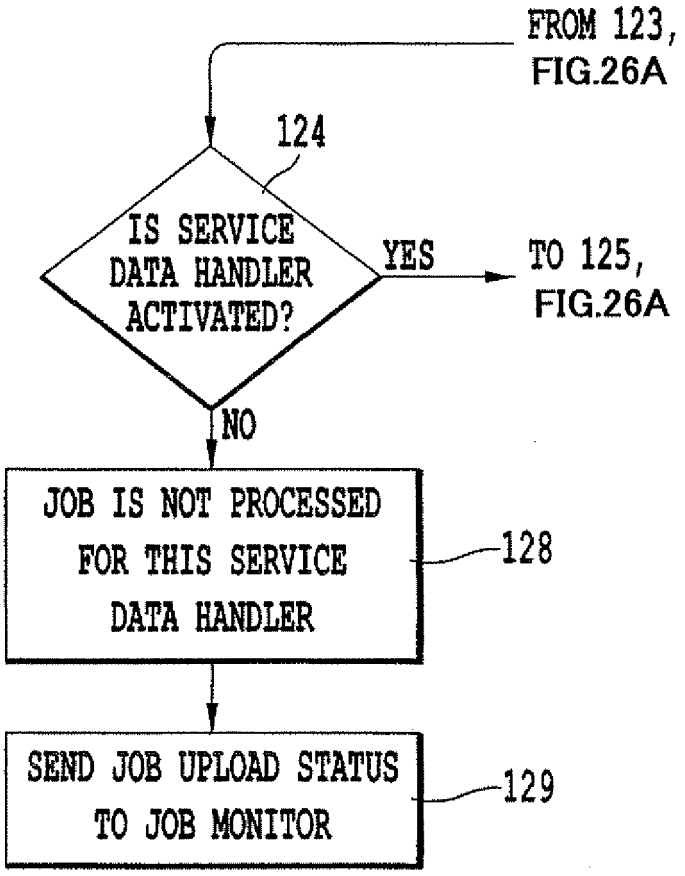

FROM 123, FIG.26A
124
IS SERVICE DATA HANDLER ACTIVATED?
YES
TO 125, FIG.26A
NO
JOB IS NOT PROCESSED FOR THIS SERVICE DATA HANDLER
128
SEND JOB UPLOAD STATUS TO JOB MONITOR
129

FIG.27

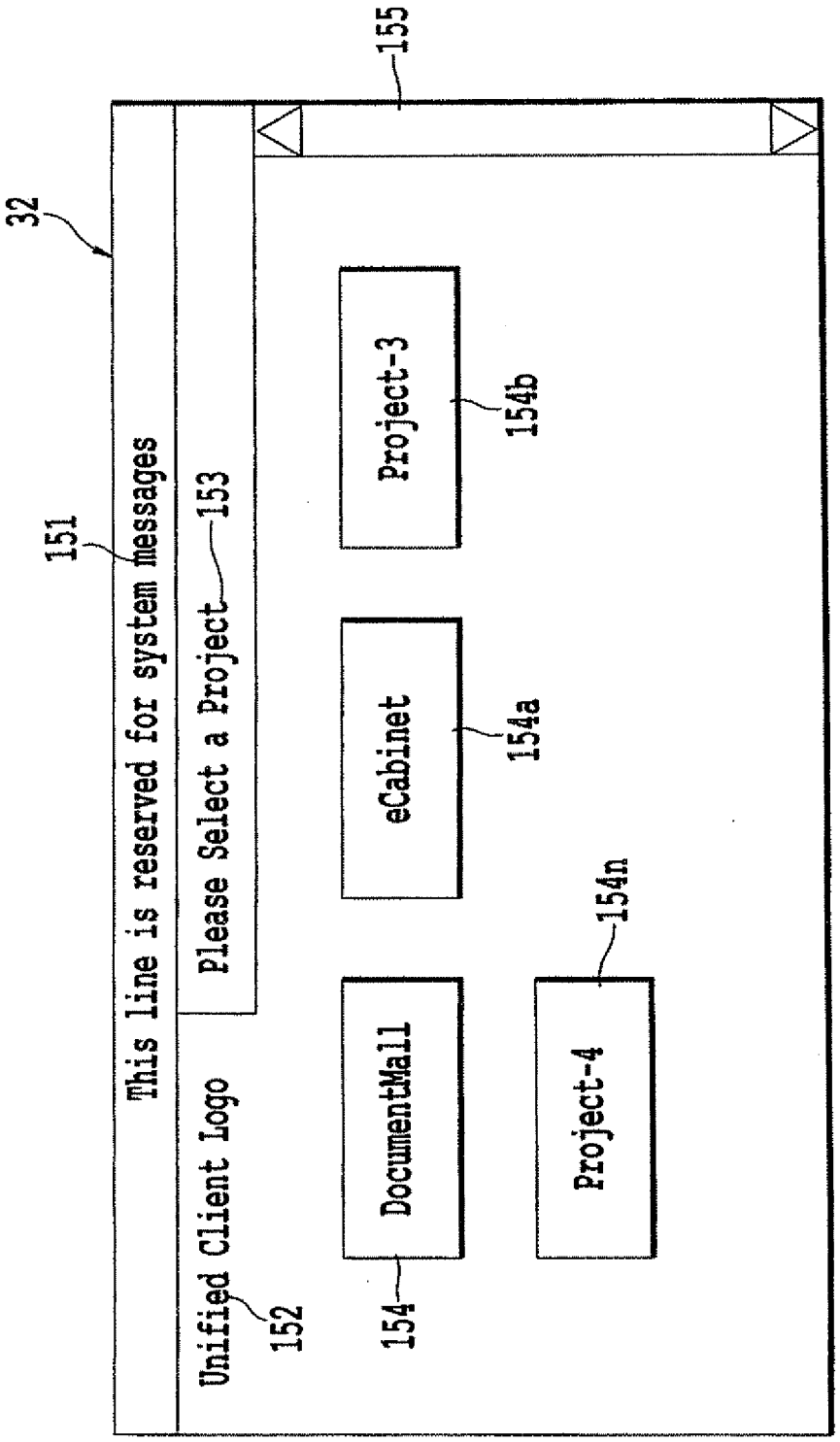

32
151
This line is reserved for system messages
153
Please Select a Project
155
Unified Client Logo
152
154
DocumentMall
eCabinet
154a
Project-3
154b
Project-4
154n

FIG.28

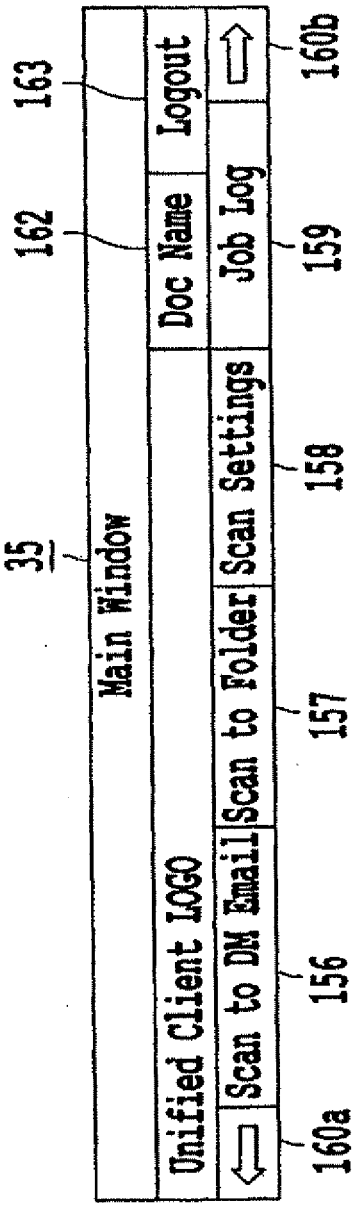

35
Main Window
Unified Client LOGO
Scan to DM Email
Scan to Folder
Scan Settings
Job Log
Doc Name
Logout
160a
156
157
158
159
162
163
160b

FIG.29A

```
<MFP>
    <MFPSerialNo>ABC112223333</MFPSerialNo>
    <MACAddress>11-22-33-44-55-66</MACAddress>
    <AccountName>Ricoh</AccountName>
    <UserName>Ricoh User</UserName>
</MFP>
<Service>
    <ServiceName>DMEmail</ServiceName>
    <DisplayName>Document Mall Email</DiplayName>
    <Activation>
        <ActivationRequired>Y</ActivationRequired>
        <Activated>Y</Activated>
        <ActivationDate>1-22-07</ActivationDate>
        <ExpirationDate>4-22-07</ExpirationDate>
    </Activation>
    <ServiceWindowClass>
        com.ricoh.UnifiedClient.DocumentMall.DMEmailWindow
        .class
    </ServiceWindowClass>
    <DataHandlerClass>
        com.ricoh.UnifiedClient.DocumentMall.DMServiceDataH
        andler.class
    </DataHandlerClass>
    <ConfigurationData>
        <DMServer>documentmall.com</DMServer>
    </ConfigurationData>
    <DataHandlerConfigurationData>optional</DataHandlerConfi
gurationData>
</Service>
<Service>
    <ServiceName>eCabinetFolder</ServiceName>
    <DisplayName>eCabinet Scan to Folder</DiplayName>
    <Activation>
        <ActivationRequired>N</ActivationRequired>
        <Activated>Y</Activated>
        <ActivationDate>1-22-07</ActivationDate>
        <ExpirationDate></ExpirationDate>
```

FIG.29B

```
1       </Activation>
2       <ServiceWindowClass>
3           com.ricoh.UnifiedClient.eCabinet.eCabinetWindow.class
4       </ServiceWindowClass>
5       <DataHandlerClass>
6           com.ricoh.UnifiedClient.eCabinet.eCabinetServiceData
7           Handler.class
8       </DataHandlerClass>
9       <ConfigurationData>
10          <eCabinetServer>eCabinet.com</eCabinetServer>
11      </ConfigurationData>
12      <DataHandlerConfigurationData>optional</DataHandlerConfi
13  gurationData>
14  </Service>
```

FIG.30

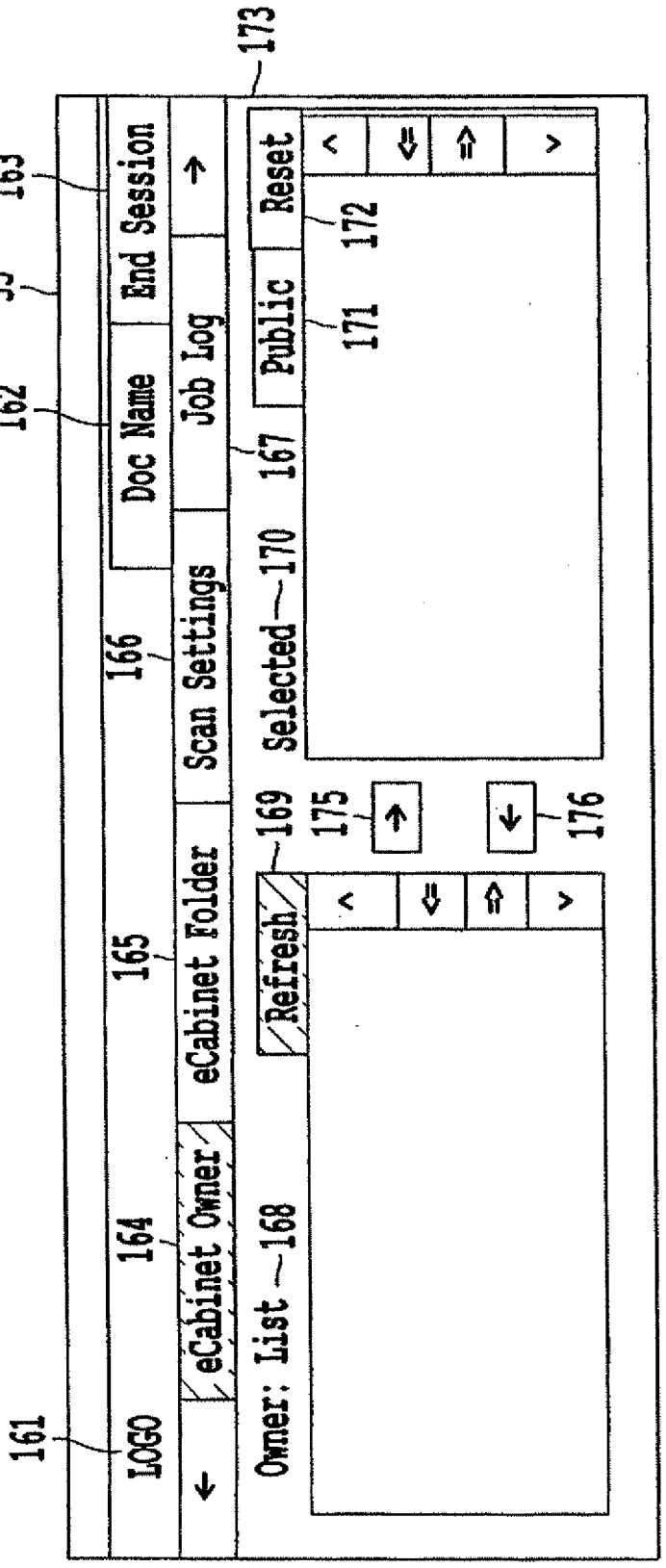

161
LOGO
164
eCabinet Owner
165
eCabinet Folder
166
Scan Settings
162
Doc Name
Job Log
35
163
End Session
173
Reset
172
Public
171
Selected
170
167
169
Refresh
Owner: List
168
175
176

FIG.31

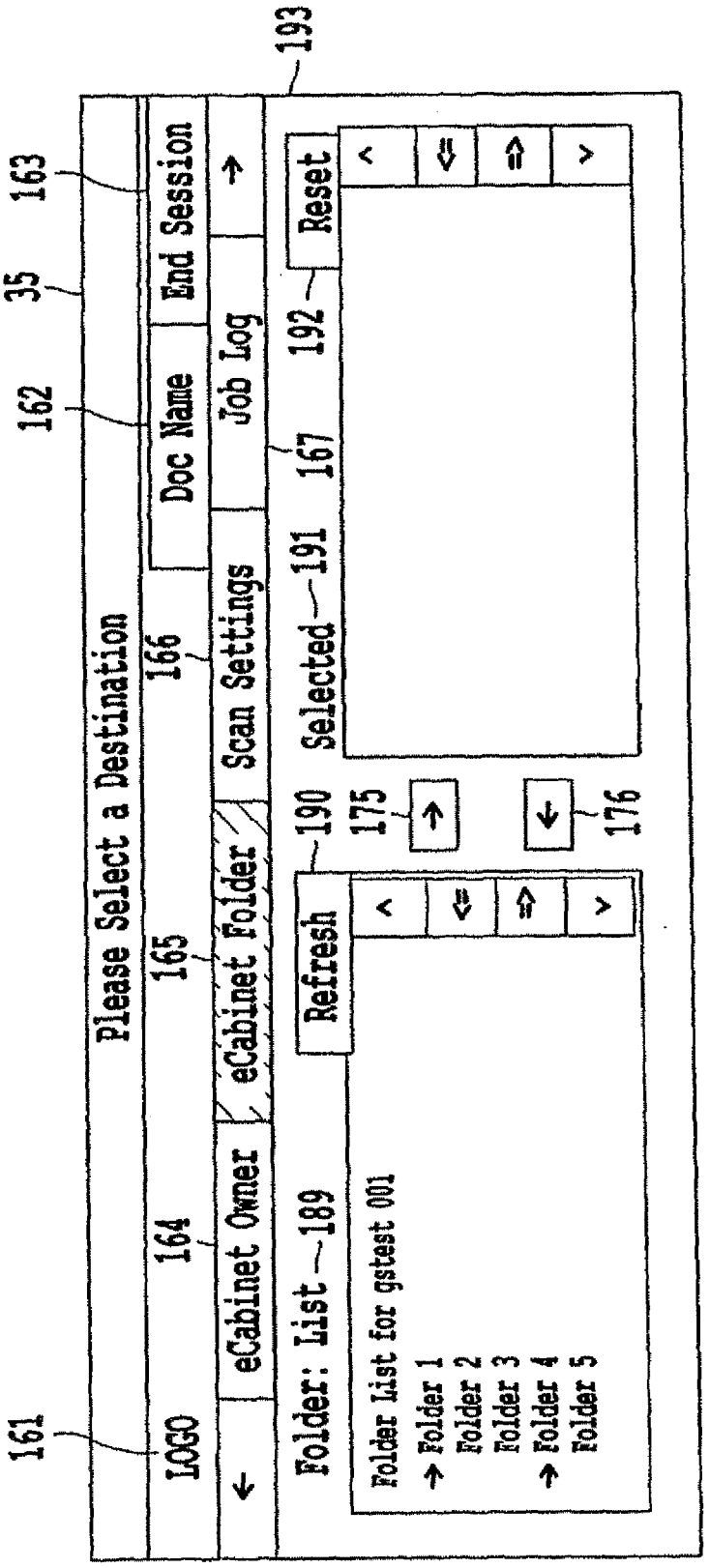

193
163
35
162
161
Please Select a Destination
LOGO
164
eCabinet Owner
165
eCabinet Folder
166
Scan Settings
Doc Name
End Session
Job Log
167
Folder: List
189
Refresh
190
Folder List for gstest 001
Folder 1
Folder 2
Folder 3
Folder 4
Folder 5
175
176
Selected
191
Reset
192

FIG.32

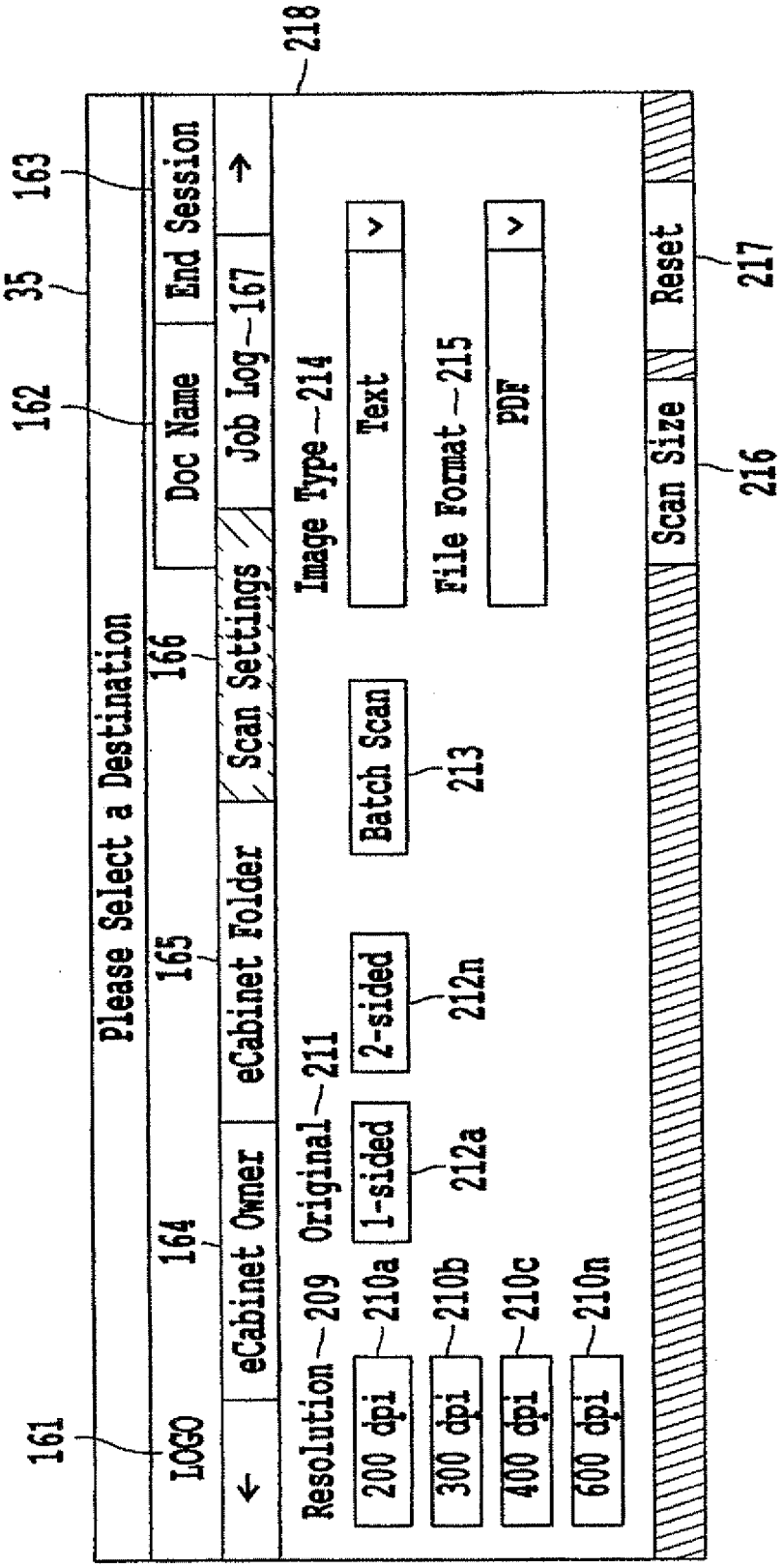

Please Select a Destination
LOGO
eCabinet Owner
eCabinet Folder
Scan Settings
Job Log
End Session
Doc Name
Resolution
200 dpi
300 dpi
400 dpi
600 dpi
Original
1-sided
2-sided
Batch Scan
Image Type
Text
File Format
PDF
Scan Size
Reset
161
162
163
164
165
166
167
209
210a
210b
210c
210n
211
212a
212n
213
214
215
216
217
218
35

FIG.33

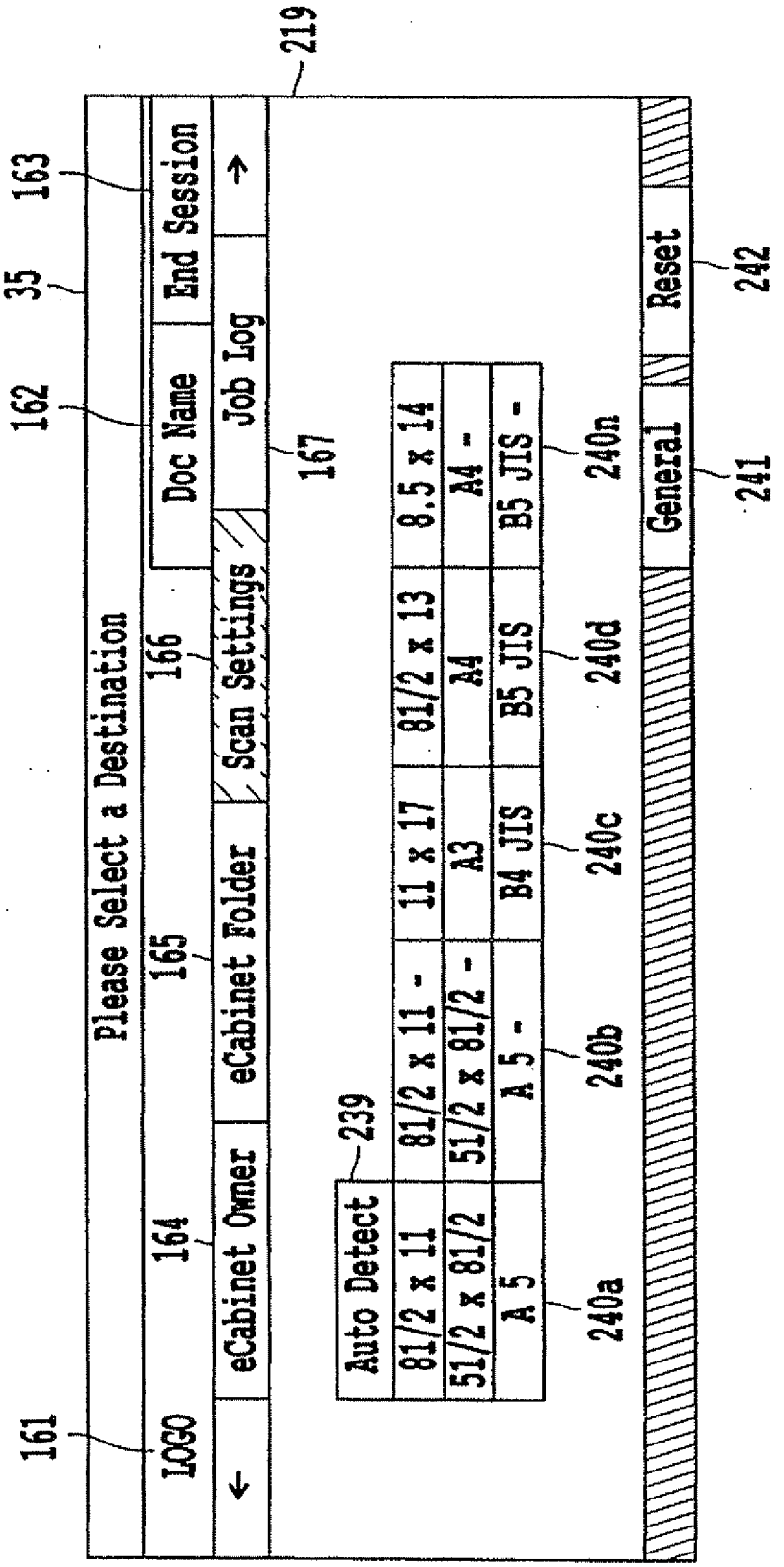

219
163
35
162
161
Please Select a Destination
LOGO
164
eCabinet Owner
165
eCabinet Folder
166
Scan Settings
Doc Name
End Session
Job Log
167
Auto Detect
239
81/2 x 11
51/2 x 81/2
A 5
240a
81/2 x 11 -
51/2 x 81/2 -
A 5 -
240b
11 x 17
A3
B4 JIS
240c
81/2 x 13
A4
B5 JIS
240d
8.5 x 14
A4 -
B5 JIS -
240n
241
General
Reset
242

FIG.34

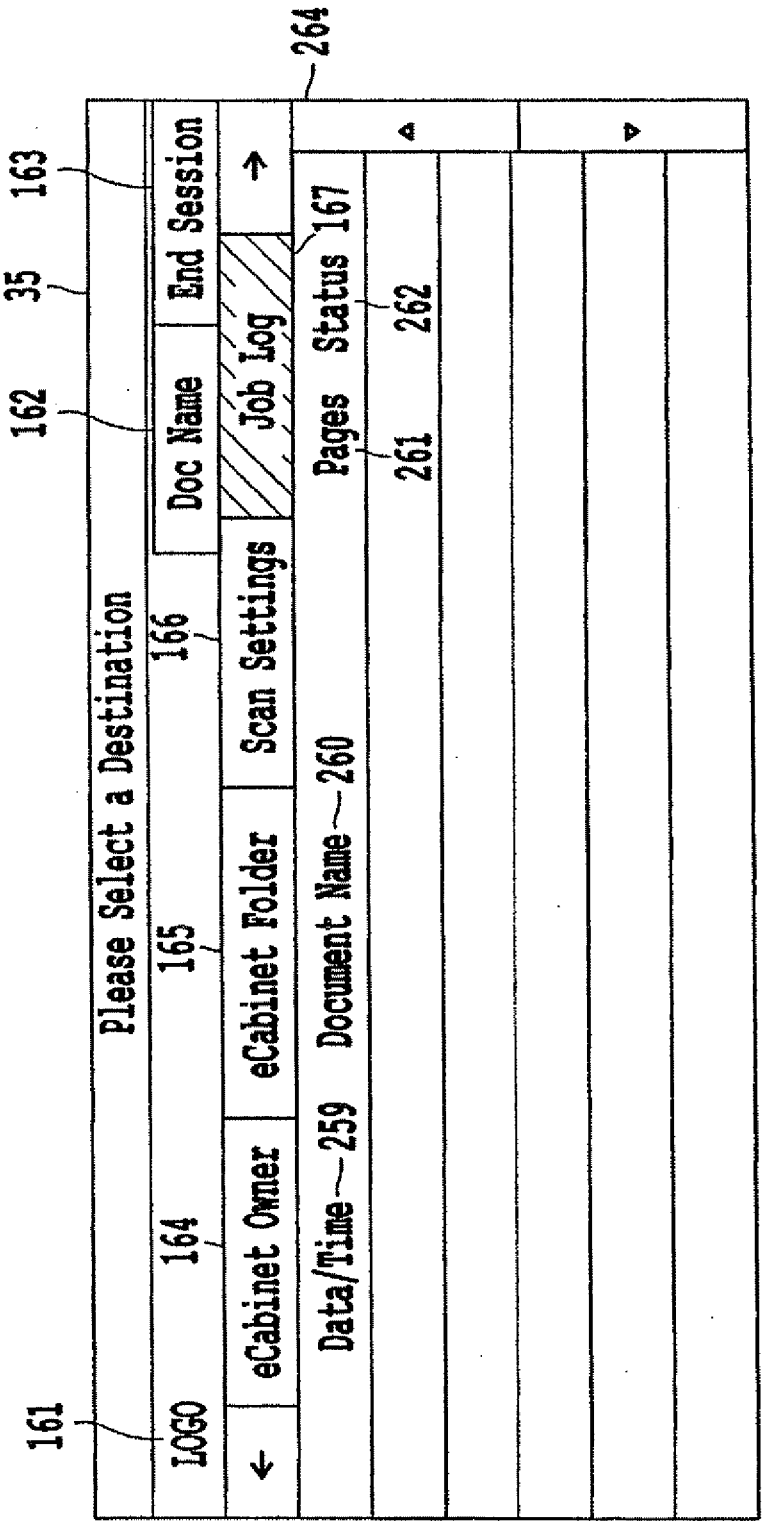

161
163
35
162
264
Please Select a Destination
LOGO
164
eCabinet Owner
165
eCabinet Folder
166
Scan Settings
Doc Name
End Session
Job Log
167
Data/Time
259
Document Name
260
Pages
261
Status
262

FIG.35

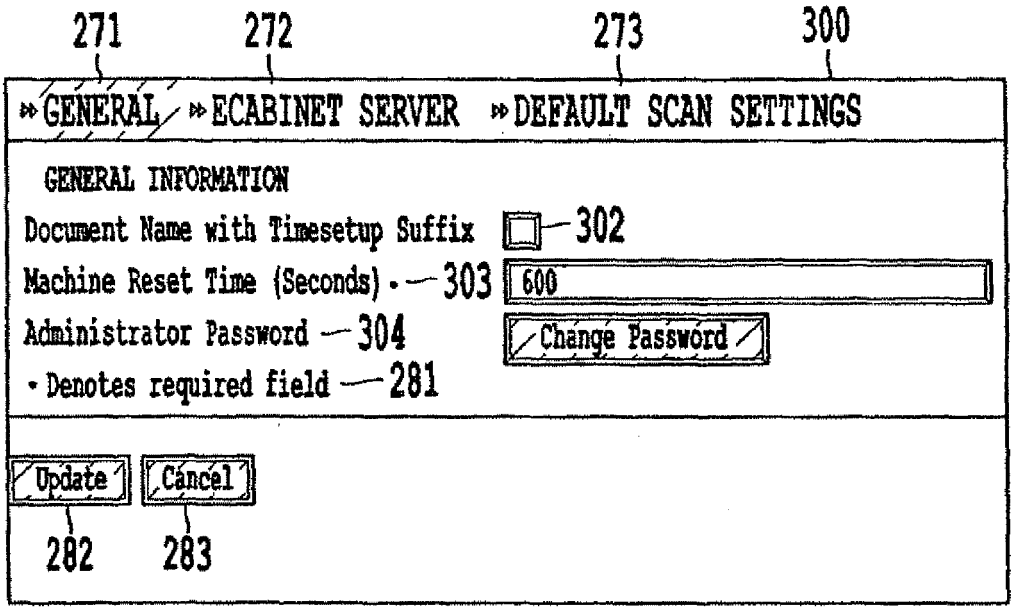

271
272
273
300
»GENERAL
»ECABINET SERVER
»DEFAULT SCAN SETTINGS
GENERAL INFORMATION
Document Name with Timesetup Suffix
302
Machine Reset Time (Seconds) •
303
600
Administrator Password
304
Change Password
• Denotes required field
281
Update
Cancel
282
283

FIG.36

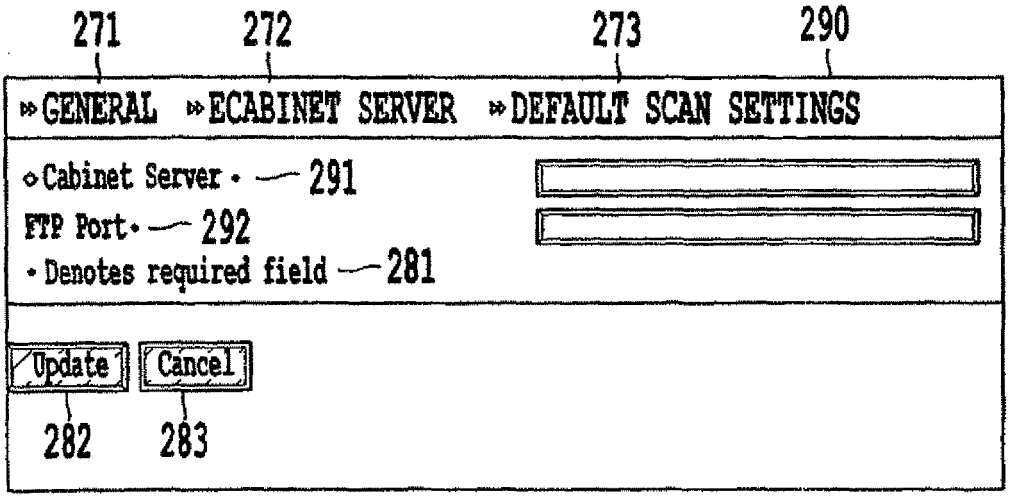

271
272
273
290
»GENERAL
»ECABINET SERVER
»DEFAULT SCAN SETTINGS
◇Cabinet Server •
291
FTP Port•
292
• Denotes required field
281
Update
Cancel
282
283

FIG.37

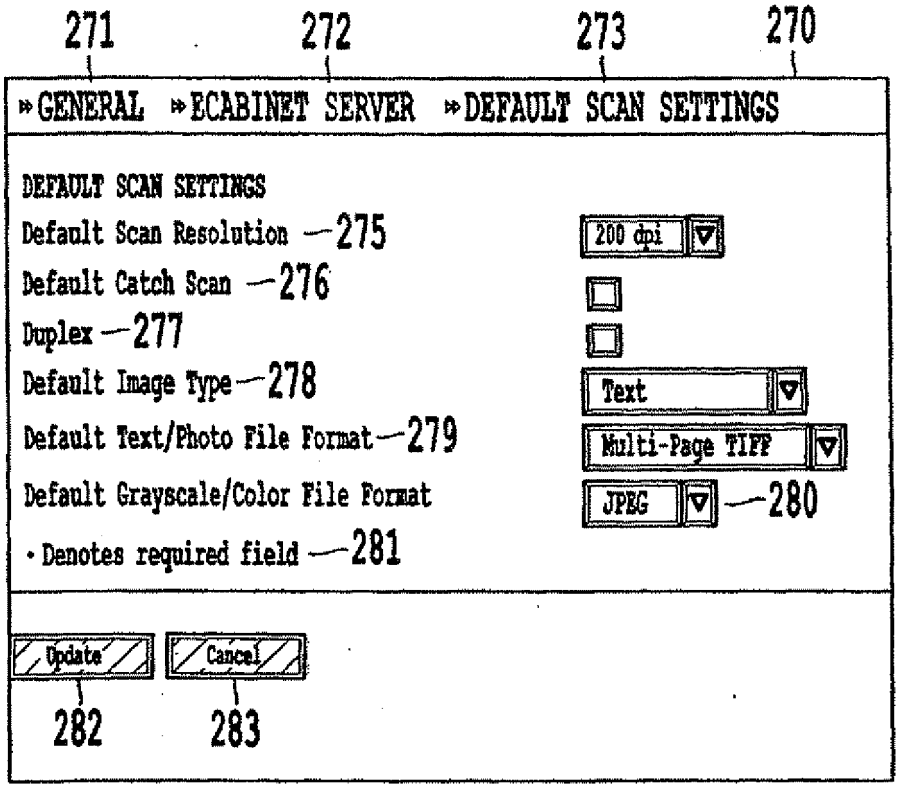

271
272
273
270
»GENERAL »ECABINET SERVER »DEFAULT SCAN SETTINGS
DEFAULT SCAN SETTINGS
Default Scan Resolution —275
200 dpi
Default Catch Scan —276
Duplex —277
Default Image Type —278
Text
Default Text/Photo File Format —279
Multi-Page TIFF
Default Grayscale/Color File Format
JPEG
—280
• Denotes required field —281
Update
Cancel
282
283

FIG.38

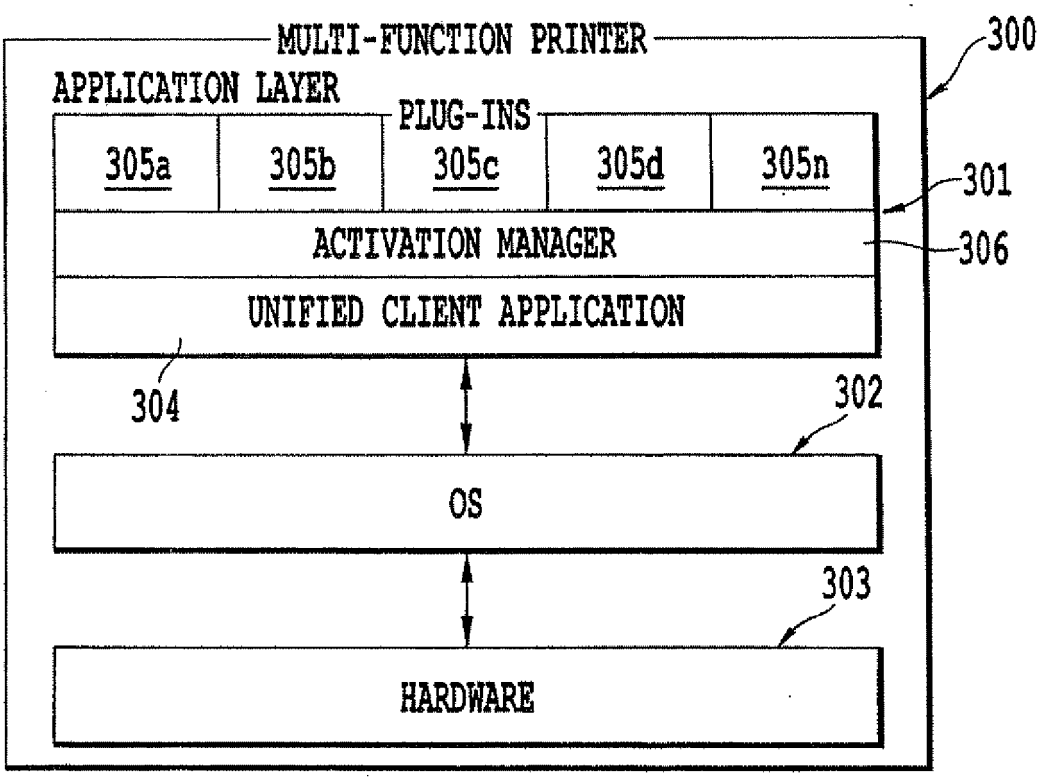

MULTI-FUNCTION PRINTER
300
APPLICATION LAYER
PLUG-INS
305a
305b
305c
305d
305n
301
ACTIVATION MANAGER
306
UNIFIED CLIENT APPLICATION
304
302
OS
303
HARDWARE

FIG.39

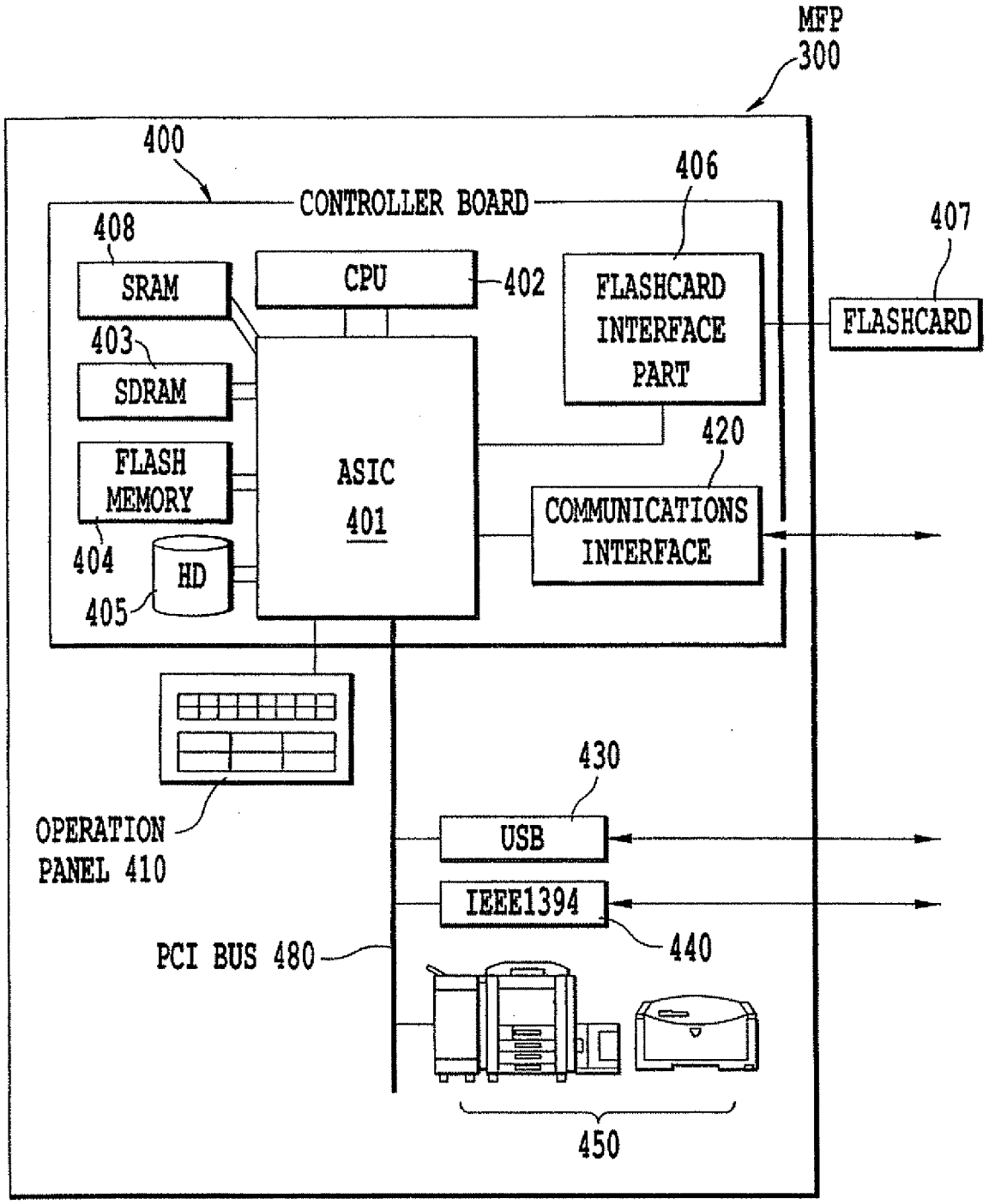

MFP
300
400
CONTROLLER BOARD
406
407
408
SRAM
CPU
402
FLASHCARD INTERFACE PART
FLASHCARD
403
SDRAM
420
FLASH MEMORY
ASIC
401
COMMUNICATIONS INTERFACE
404
HD
405
430
OPERATION PANEL 410
USB
IEEE1394
440
PCI BUS 480
450

FIG.40 RELATED ART

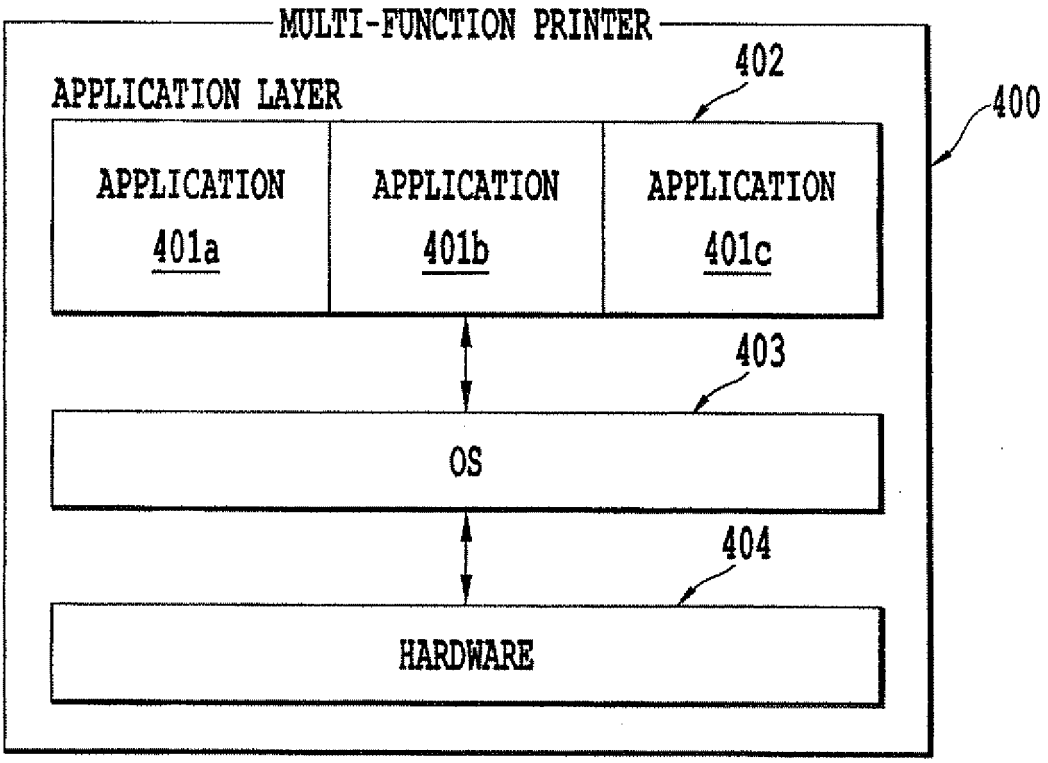


FIG.41

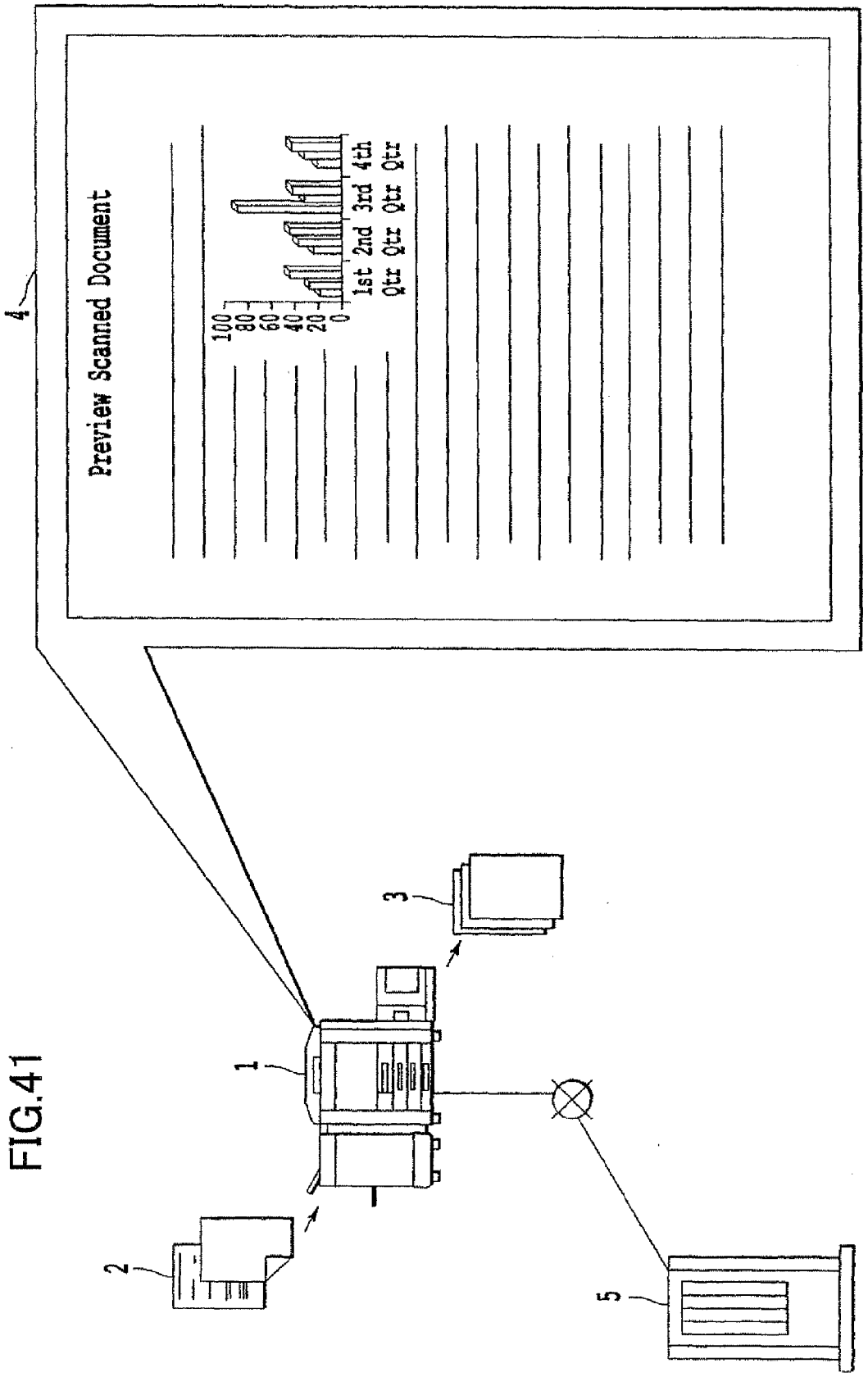

Preview Scanned Document
100
80
60
40
20
0
1st Qtr
2nd Qtr
3rd Qtr
4th Qtr
1
2
3
4
5

FIG.42A

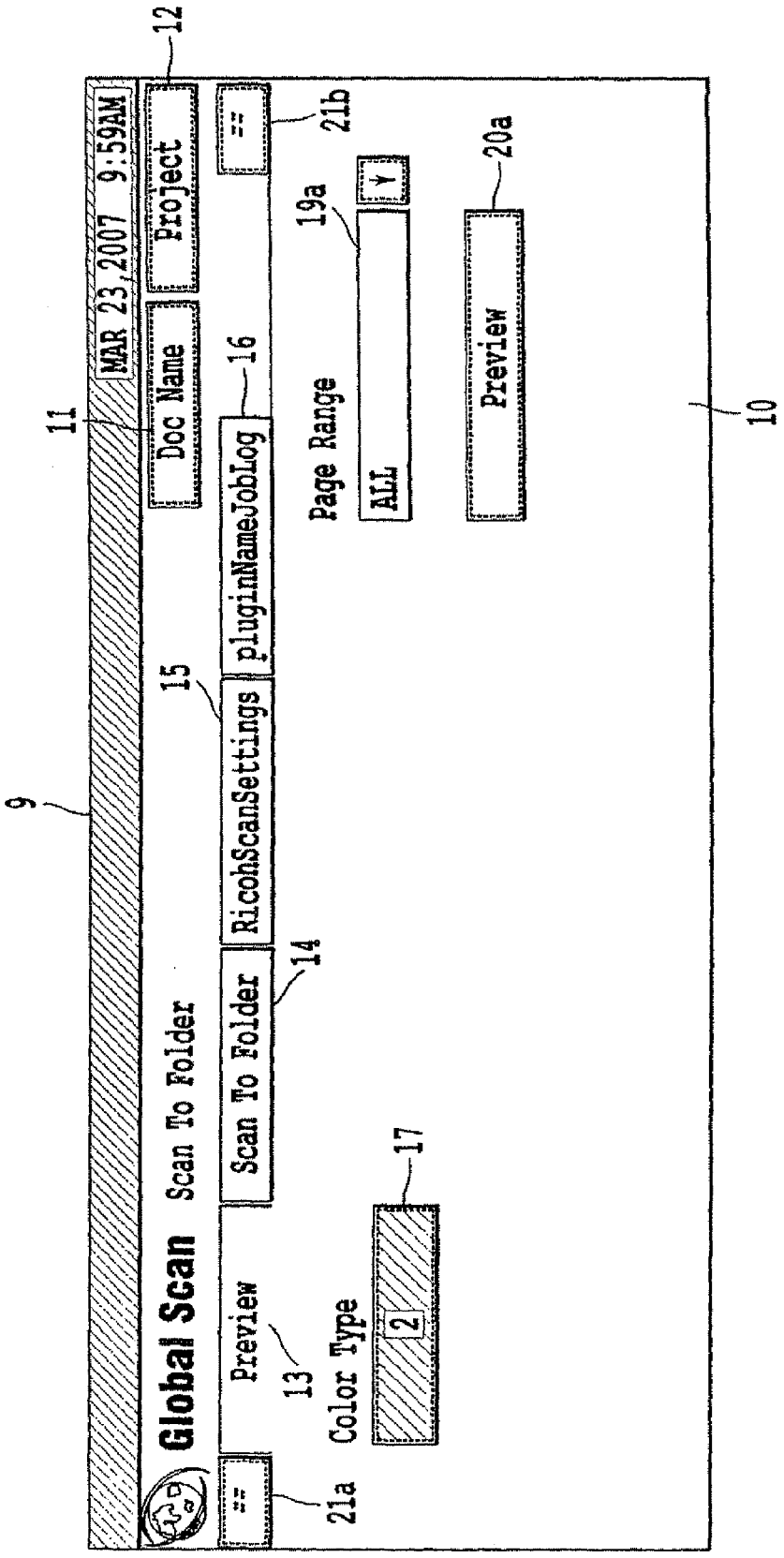

9
11
12
MAR 23 2007 9:59AM
Project
Doc Name
16
==
21b
Global Scan
Scan To Folder
15
RicohScanSettings
pluginNameJobLog
Page Range
19a
ALL
Y
20a
Preview
10
Scan To Folder
14
Preview
13
Color Type
17
2
==
21a

FIG.42B

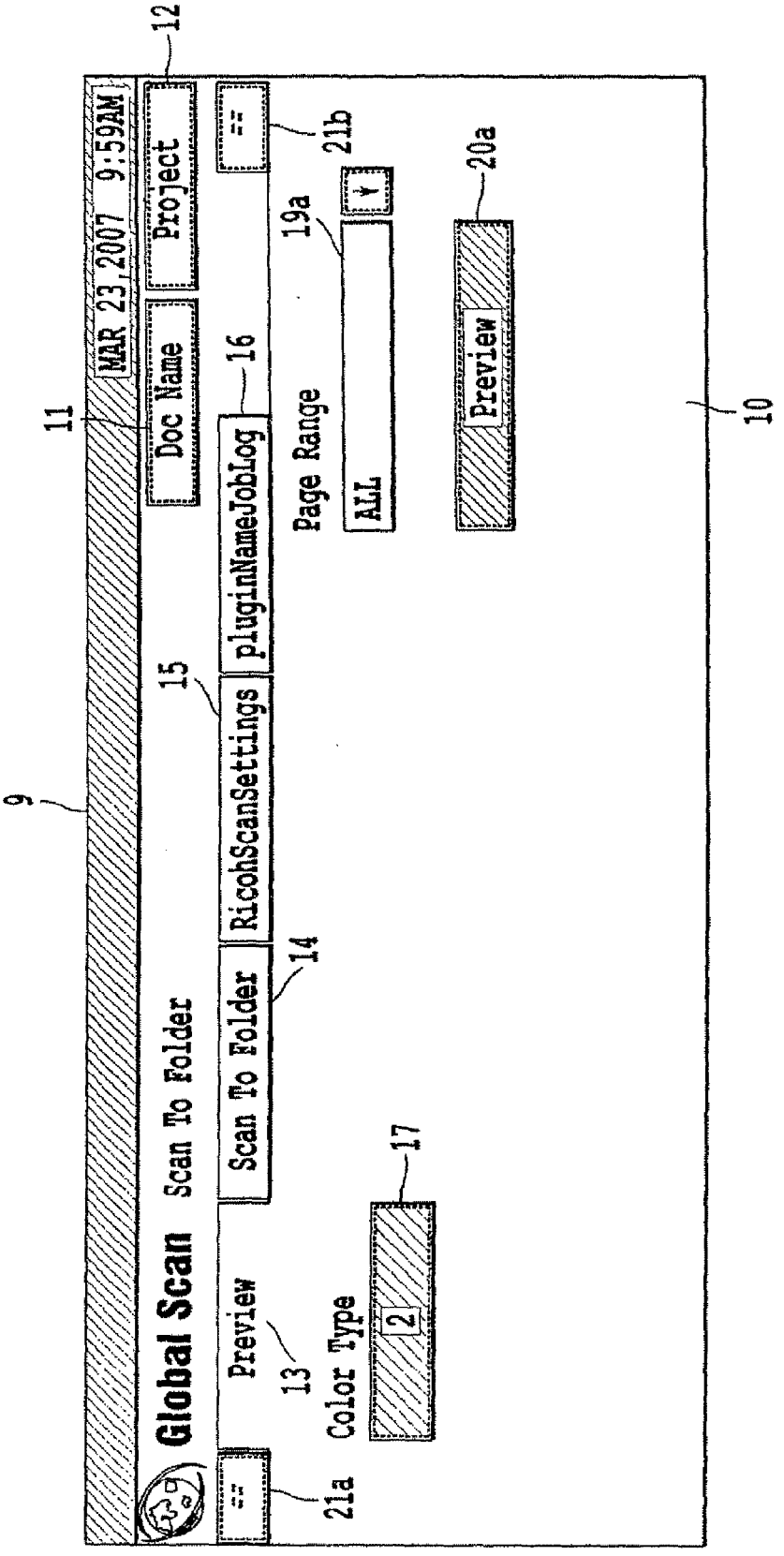

MAR 23 2007 9:59AM
Project
Doc Name
16
pluginNameJobLog
Page Range
ALL
Preview
RicohScanSettings
Scan To Folder
Global Scan
Scan To Folder
Preview
Color Type
2
9
10
11
12
13
14
15
17
19a
20a
21a
21b

FIG.42C

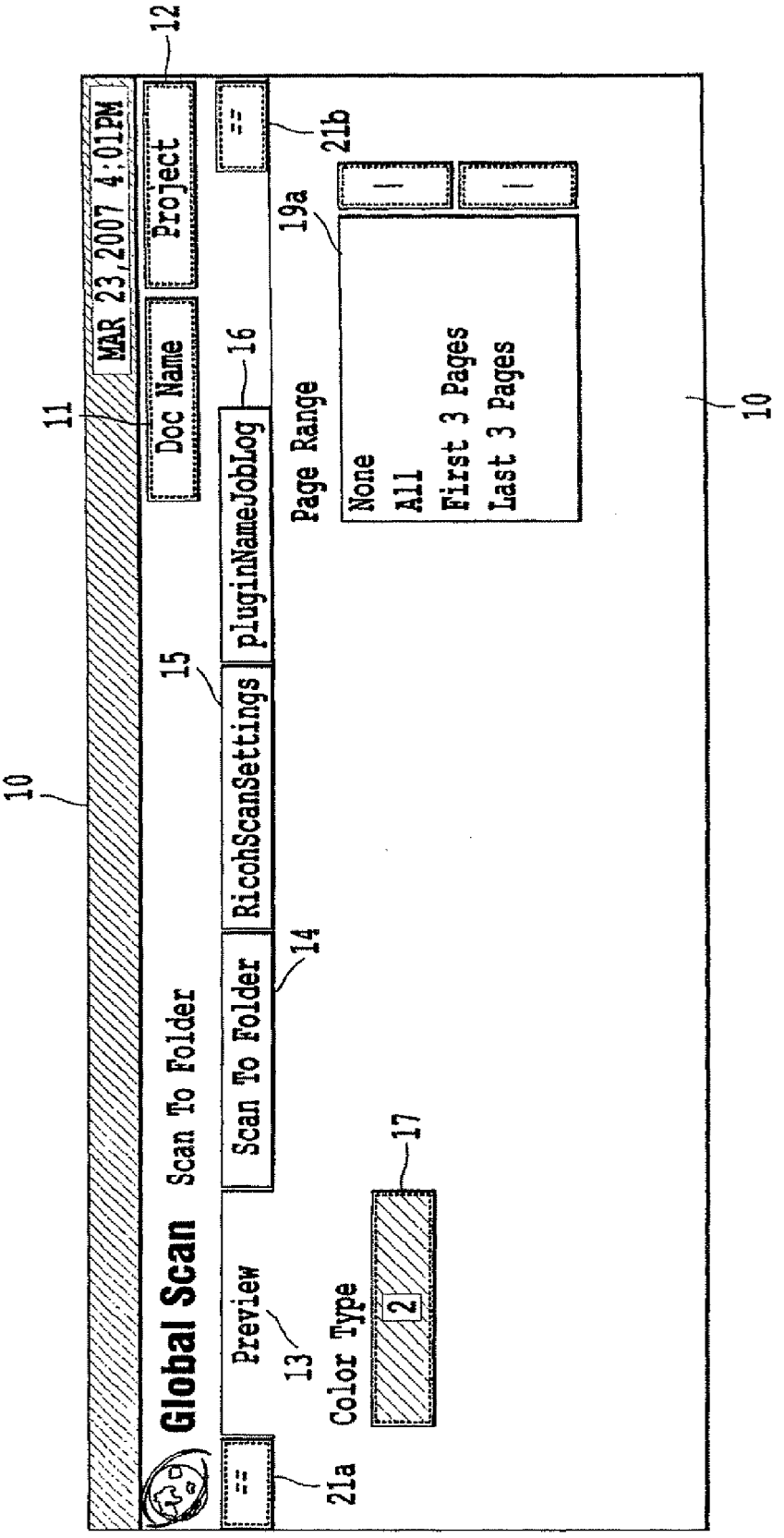

10
11
12
15
16
14
13
17
21a
21b
19a
MAR 23,2007 4:01PM
Project
Doc Name
pluginNameJobLog
RicohScanSettings
Scan To Folder
Preview
Global Scan
Scan To Folder
Color Type
2
==
==
Page Range
None
All
First 3 Pages
Last 3 Pages
1
1
10

FIG.43A

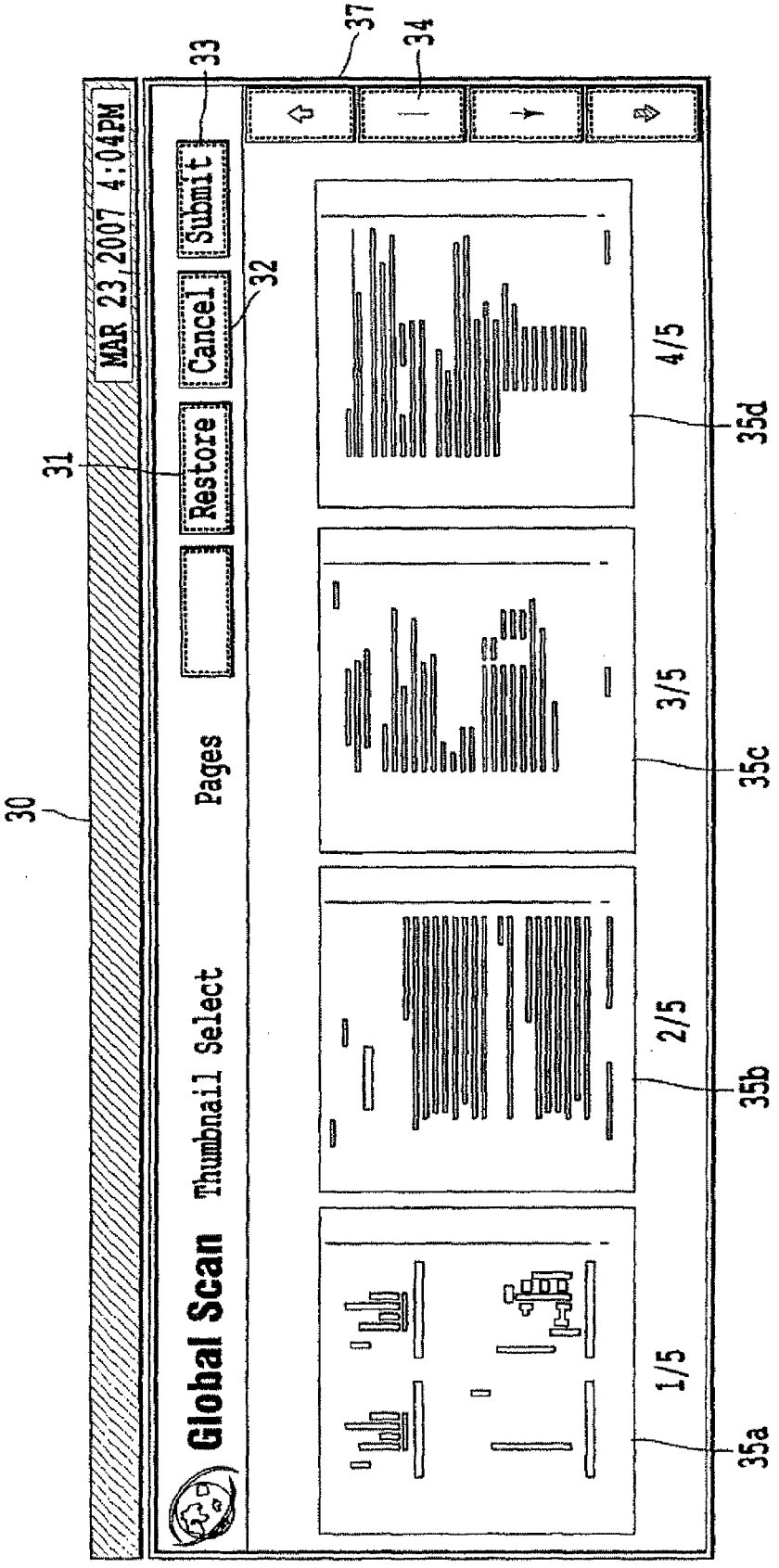

30
31
32
33
34
35a
35b
35c
35d
37
MAR 23,2007 4:04PM
Restore
Cancel
Submit
Global Scan
Thumbnail Select
Pages
1/5
2/5
3/5
4/5

FIG.44

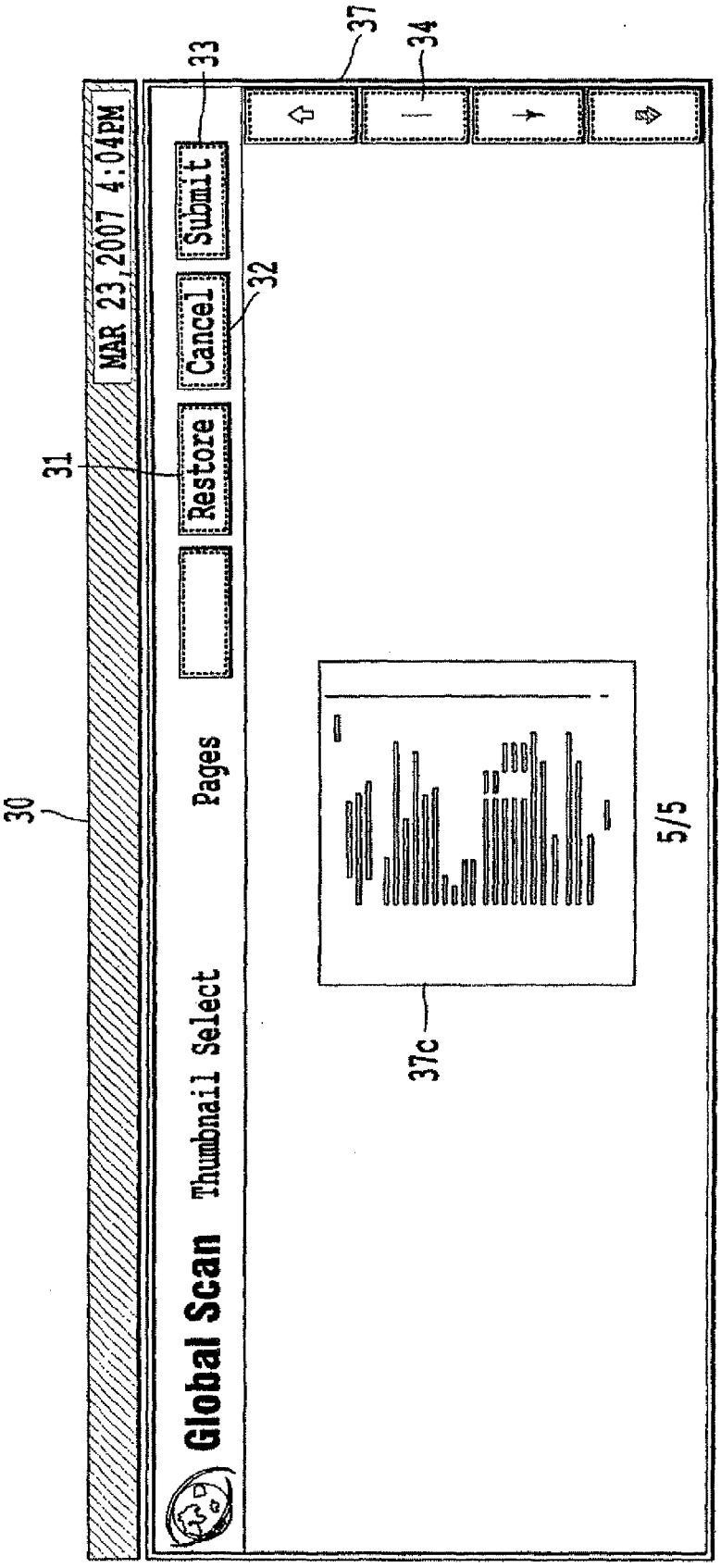

MAR 23,2007 4:04PM
Global Scan
Thumbnail Select
Pages
Restore
Cancel
Submit
5/5
30
31
32
33
34
37
37c

FIG.45

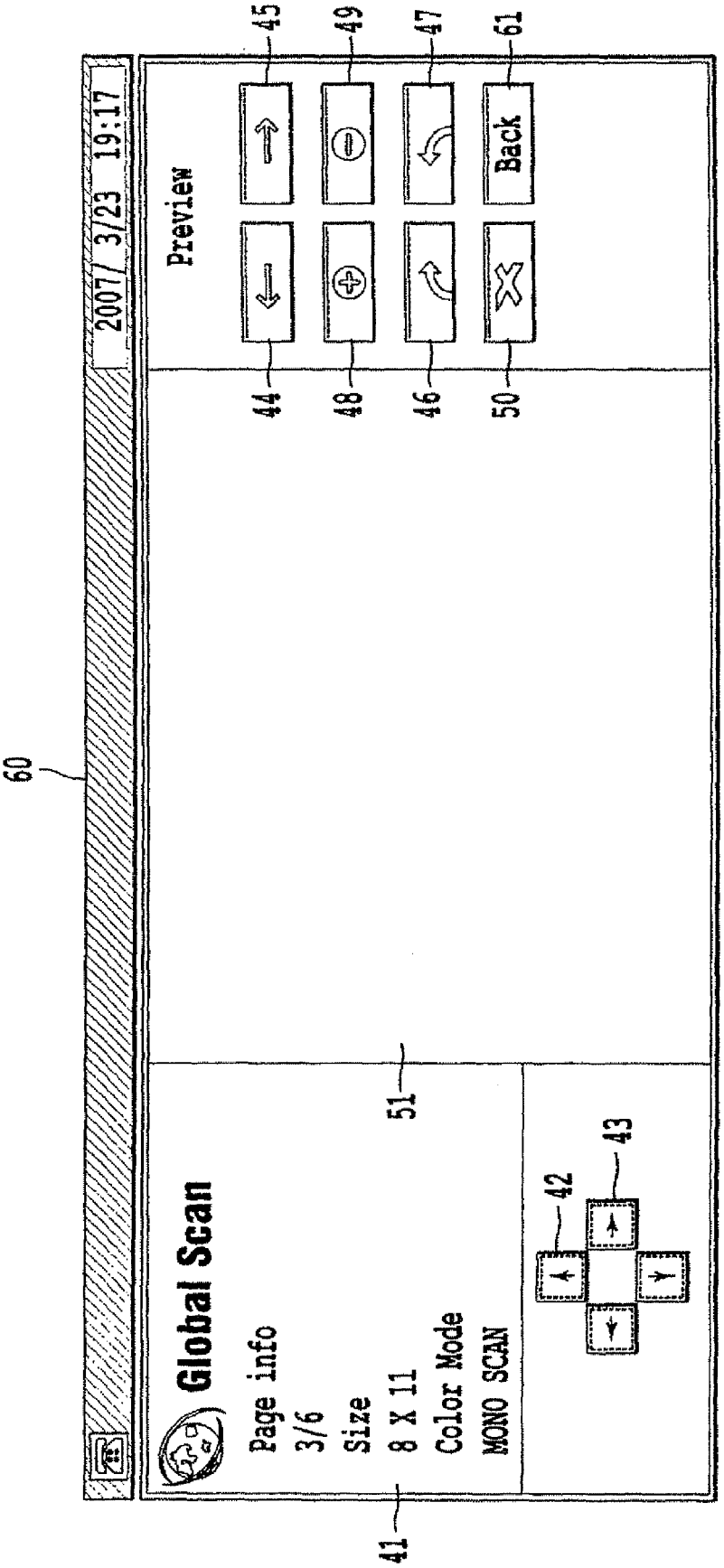

60
2007/ 3/23 19:17
Preview
45
49
47
61
44
48
46
50
Back
Global Scan
Page info
3/6
Size
8 X 11
Color Mode
MONO SCAN
51
41
42
43

FIG.46

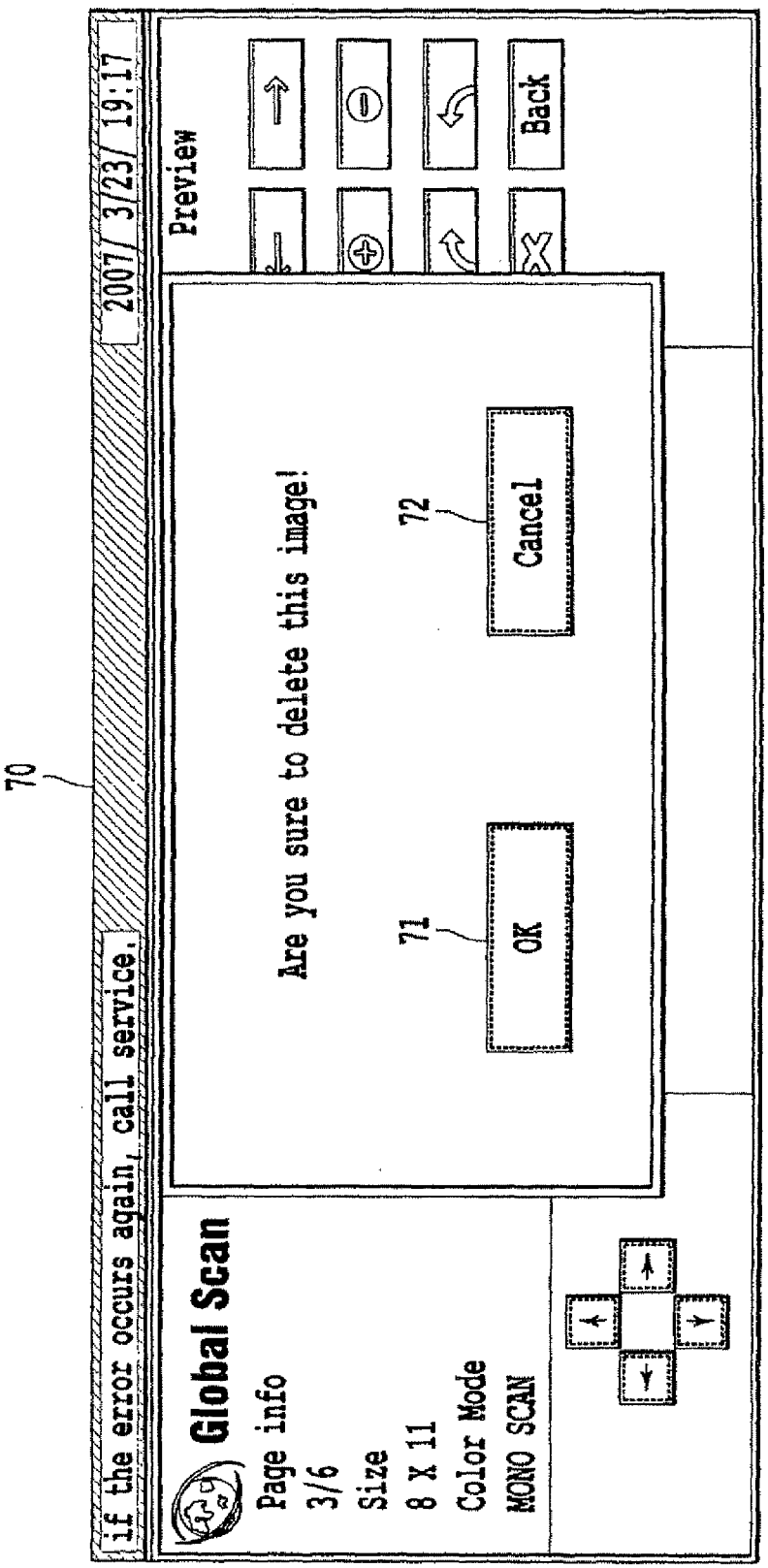

70
if the error occurs again, call service.
2007/ 3/23/ 19:17
Preview
Back
Are you sure to delete this image!
71
OK
72
Cancel
Global Scan
Page info
3/6
Size
8 X 11
Color Mode
MONO SCAN

FIG.47

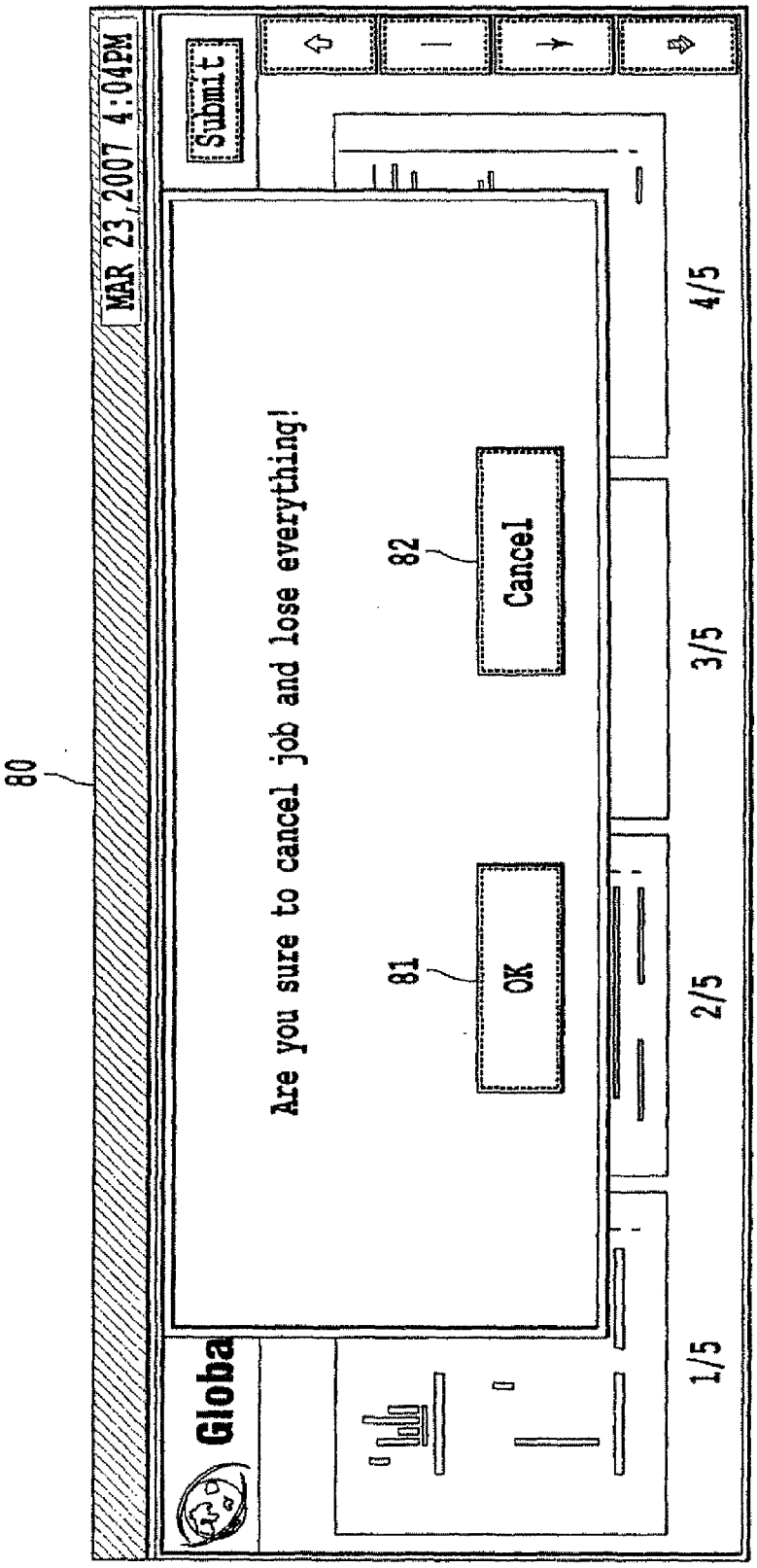

80
MAR 23,2007 4:04PM
Submit
Are you sure to cancel job and lose everything!
81
OK
82
Cancel
1/5
2/5
3/5
4/5

FIG.48

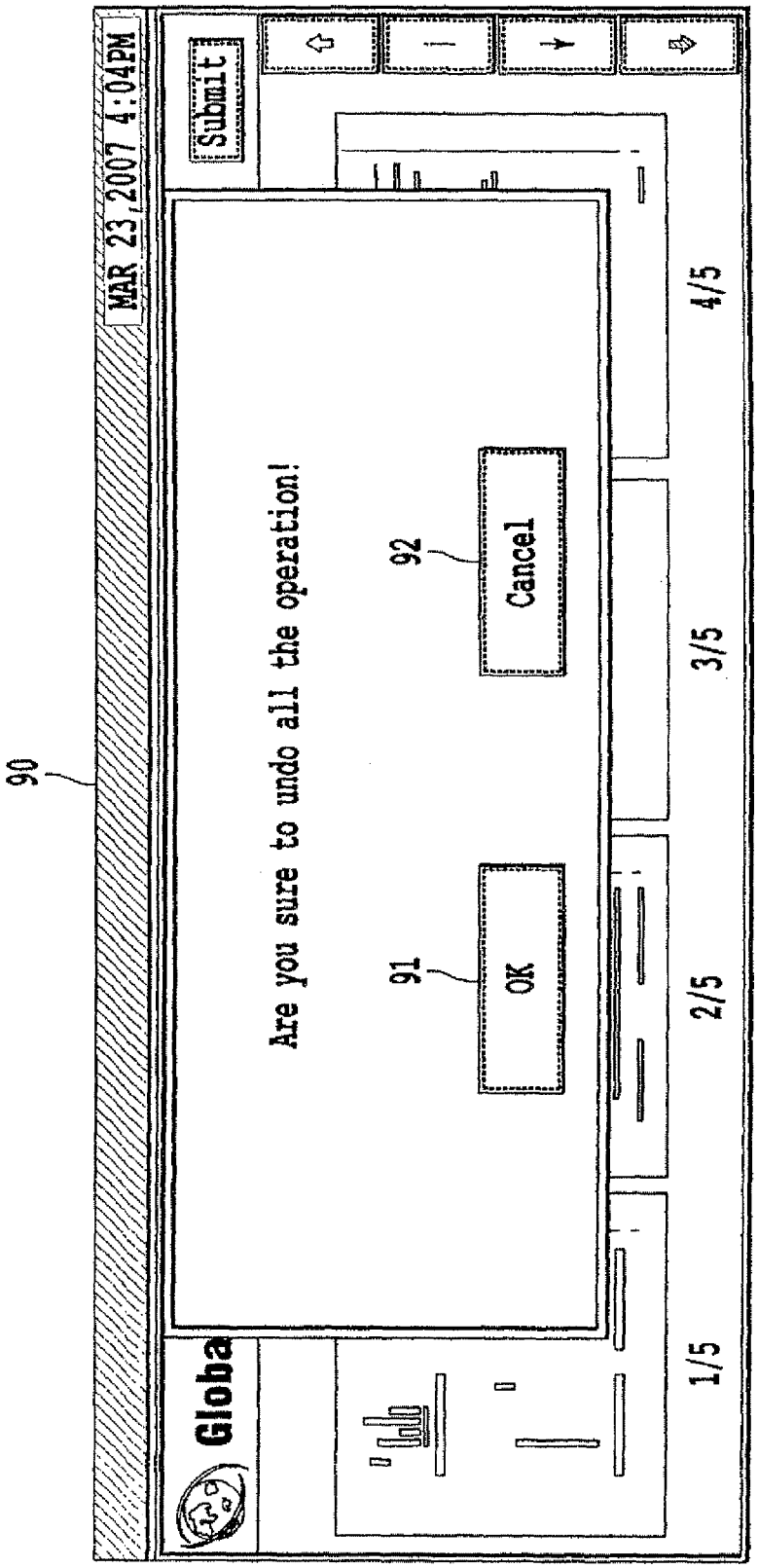

90
MAR 23,2007 4:04PM
Submit
Are you sure to undo all the operation!
91
OK
92
Cancel
1/5
2/5
3/5
4/5

FIG.49A

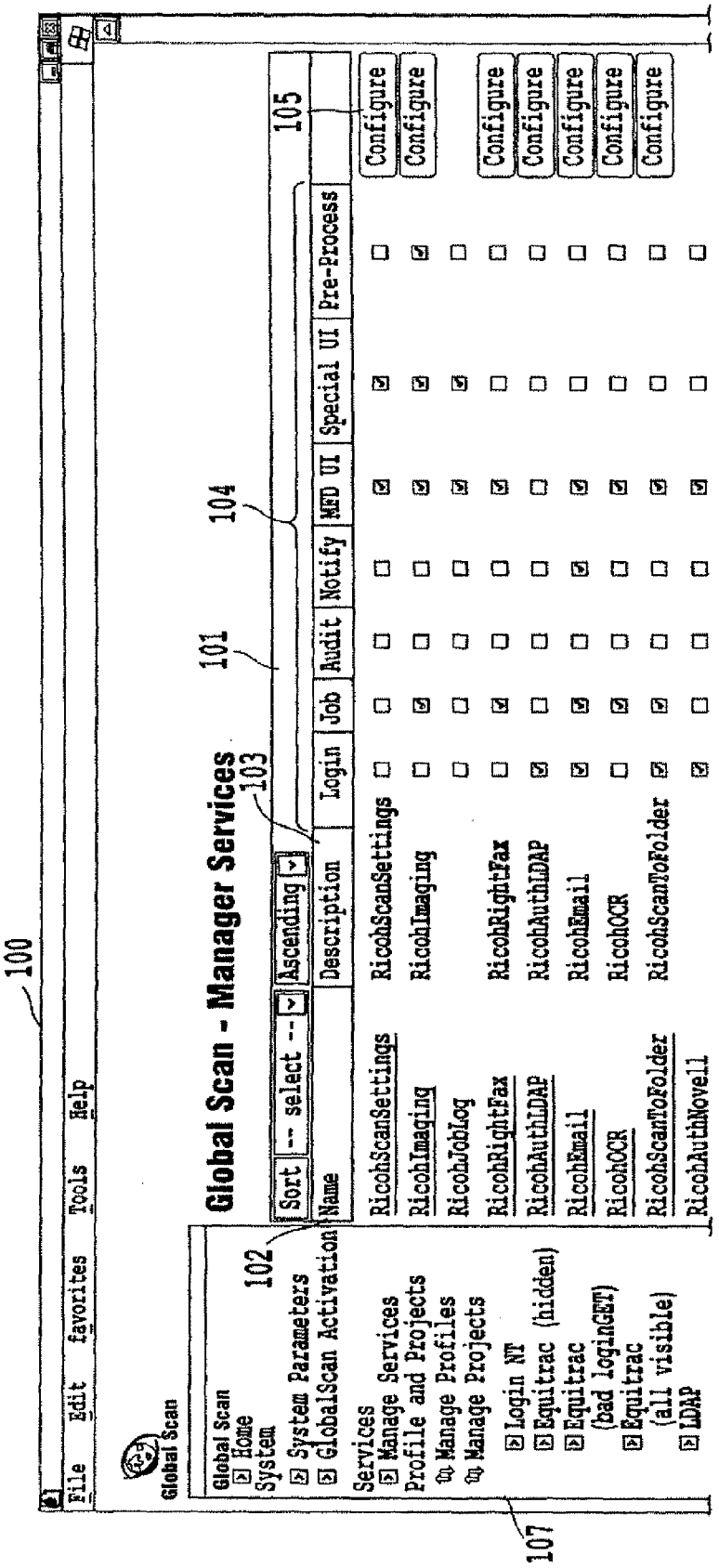

100
File Edit favorites Tools Help
Global Scan
Global Scan
Home
System
System Parameters
GlobalScan Activation
Services
Manage Services
Profile and Projects
Manage Profiles
Manage Projects
Login NT
Equitrac (hidden)
Equitrac (bad loginGET)
Equitrac (all visible)
LDAP
102
107
Global Scan - Manager Services
101
103
104
105
Sort
-- select --
Ascending
Name
Description
Login
Job
Audit
Notify
MFD UI
Special UI
Pre-Process
RicohScanSettings
RicohImaging
RicohJobLog
RicohRightFax
RicohAuthLDAP
RicohEmail
RicohOCR
RicohScanToFolder
RicohAuthNovell
Configure

FIG.49B

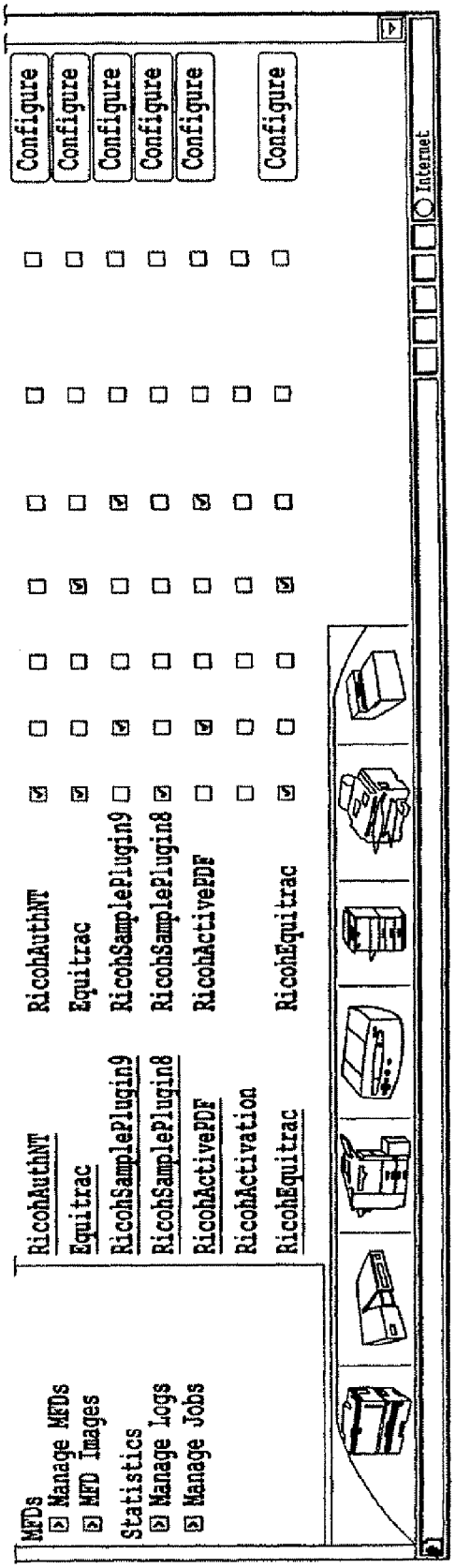

MFDs
Manage MFDs
MFD Images
Statistics
Manage Logs
Manage Jobs
RicohAuthNT
Equitrac
RicohSamplePlugin9
RicohSamplePlugin8
RicohActivePDF
RicohActivation
RicohEquitrac
Configure
Configure
Configure
Configure
Configure
Configure
Internet

FIG.50A

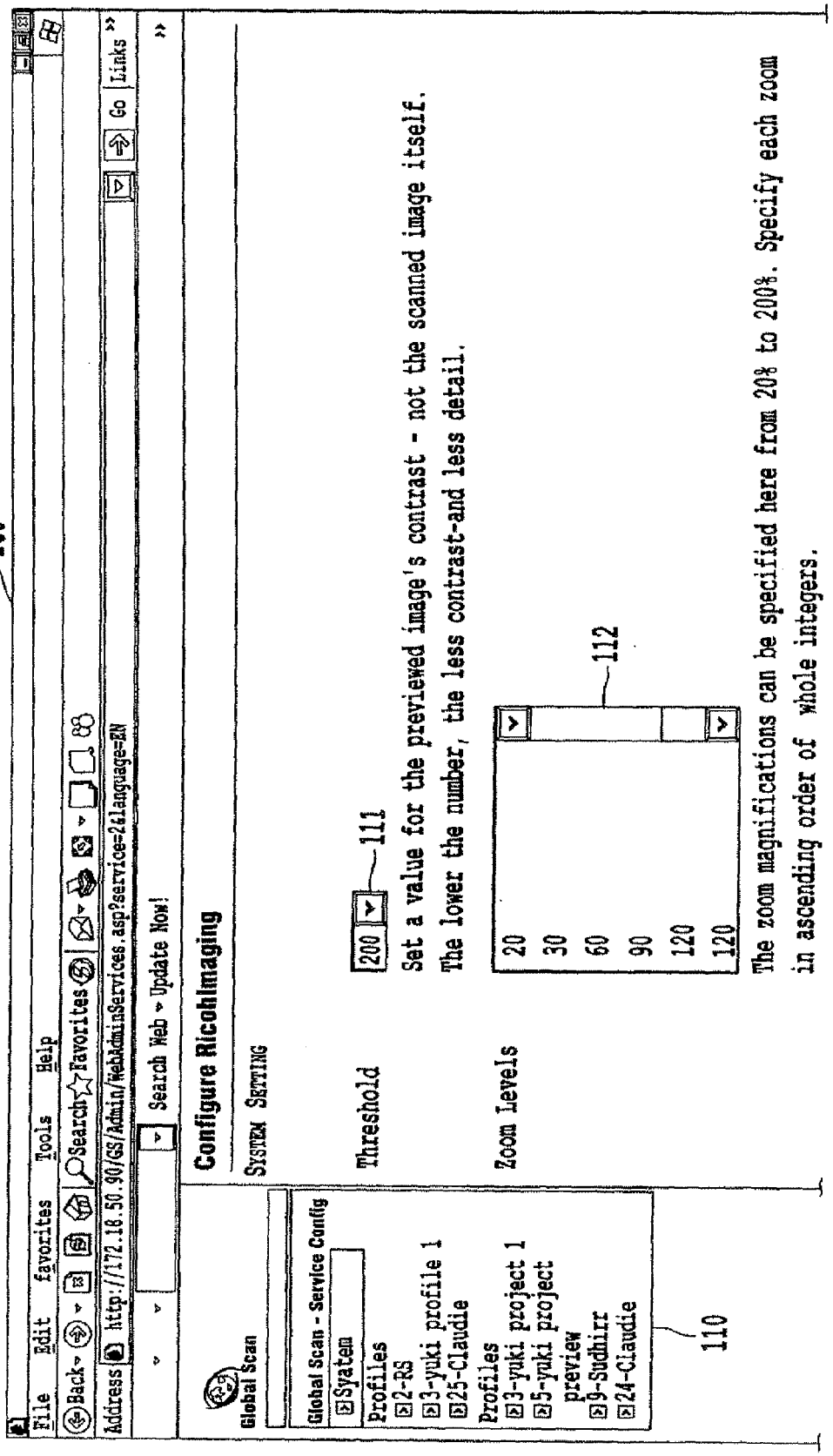

160
File Edit Favorites Tools Help
Back Search Favorites
Address http://172.18.50.90/GS/Admin/WebAdminServices.asp?service=2&language=EN
Go Links
Search Web Update Now!
Global Scan
Configure RicohImaging
Global Scan - Service Config
SYSTEM SETTING
Syatem
Profiles
2-RS
3-yuki profile 1
25-Claudie
Profiles
3-yuki project 1
5-yuki project preview
9-Sudhirr
24-Claudie
110
Threshold
200
111
Set a value for the previewed image's contrast - not the scanned image itself.
The lower the number, the less contrast-and less detail.
Zoom Levels
20
30
60
90
120
120
112
The zoom magnifications can be specified here from 20% to 200%. Specify each zoom in ascending order of whole integers.

FIG.50B

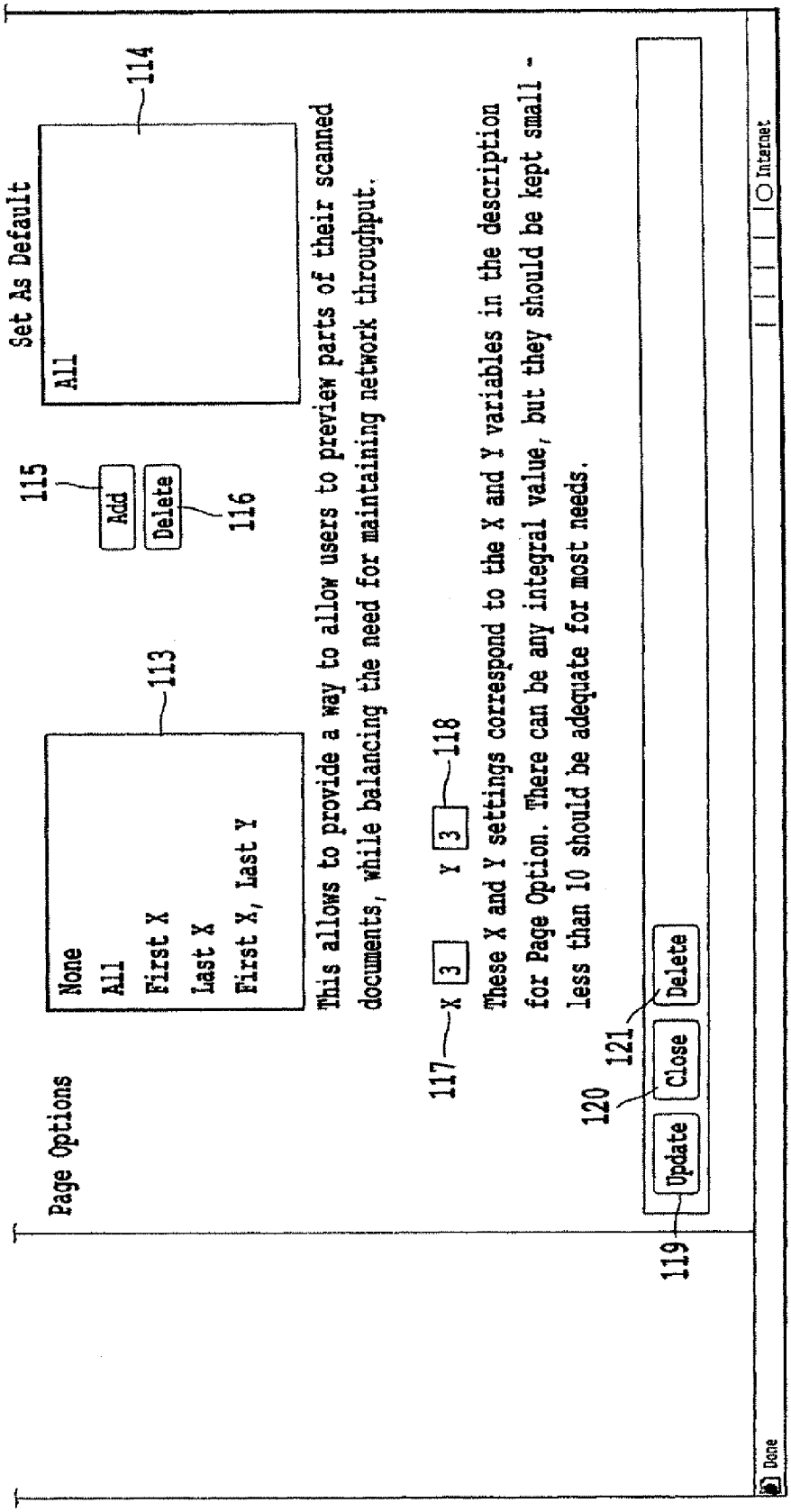

Page Options
None
All
First X
Last X
First X, Last Y
113
Add
Delete
115
116
Set As Default
All
114
This allows to provide a way to allow users to preview parts of their scanned documents, while balancing the need for maintaining network throughput.
X 3
Y 3
117
118
These X and Y settings correspond to the X and Y variables in the description for Page Option. There can be any integral value, but they should be kept small - less than 10 should be adequate for most needs.
120
121
Update
Close
Delete
119
Done
Internet

FIG.51A

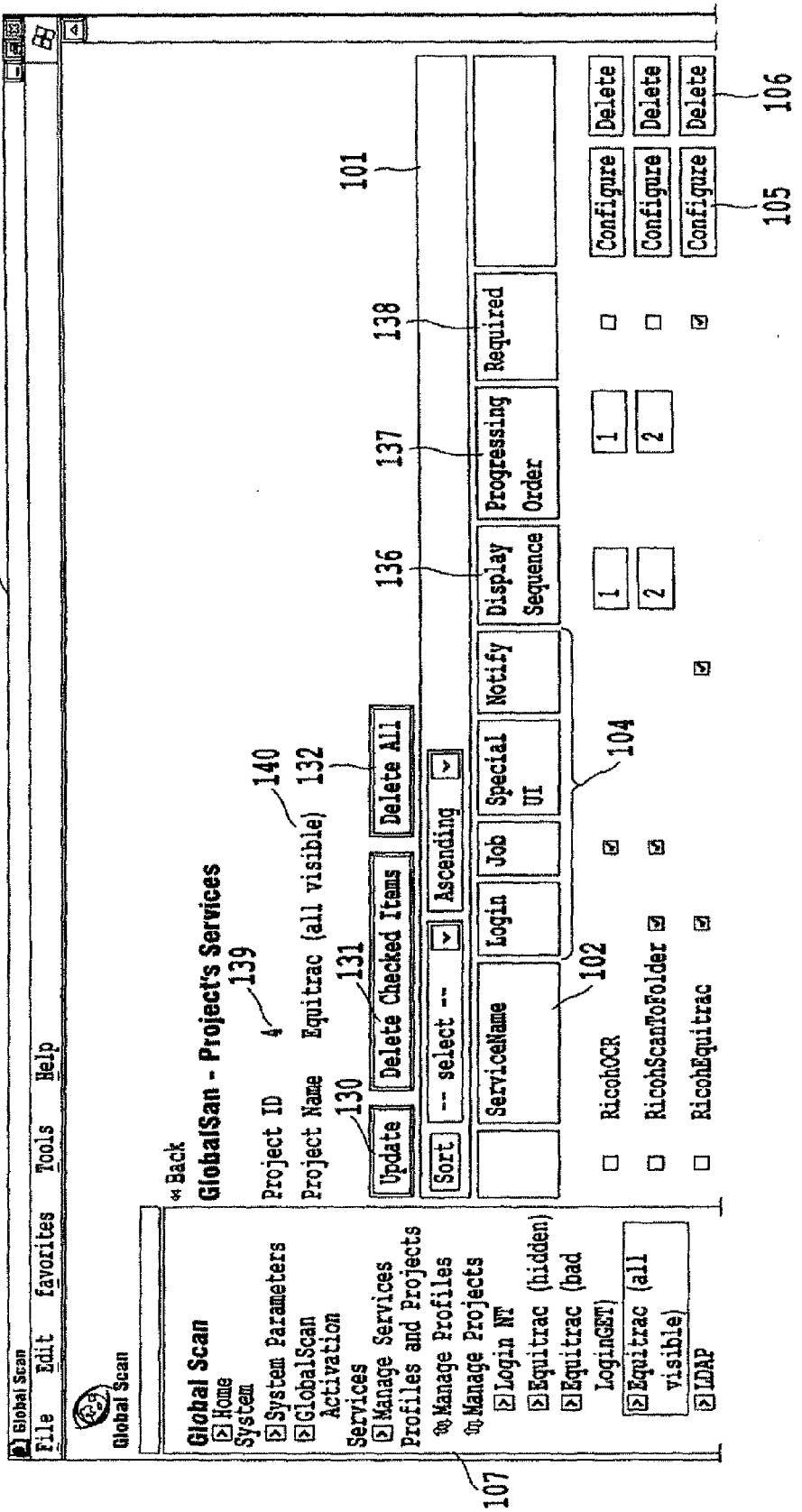

110
Global Scan
File Edit favorites Tools Help
Global Scan
Global Scan
Home
System
System Parameters
GlobalScan Activation
Services
Manage Services
Profiles and Projects
Manage Profiles
Manage Projects
Login NT
Equitrac (hidden)
Equitrac (bad LoginGET)
Equitrac (all visible)
LDAP
107
Back
GlobalSan - Project's Services
139
Project ID 4
140
Project Name Equitrac (all visible) 132
131
130
Update
Delete Checked Items
Delete ALL
136
138
137
101
Sort
-- select --
Ascending
ServiceName
Login
Job
Special UI
Notify
Display Sequence
Progressing Order
Required
102
104
RicohOCR
RicohScanToFolder
RicohEquitrac
1
2
1
2
Configure
Configure
Configure
Delete
Delete
Delete
105
106

FIG.51B

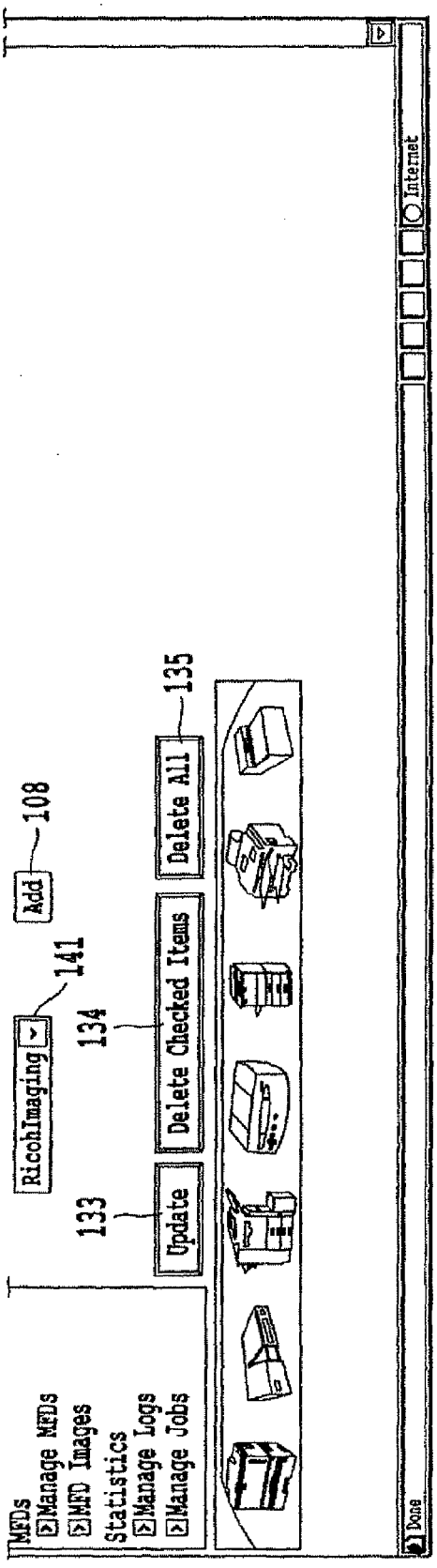

MFDs
Manage MFDs
MFD Images
Statistics
Manage Logs
Manage Jobs
Update
133
RicohImaging
141
Delete Checked Items
134
Add
108
Delete All
135
Internet
Done

FIG.52A

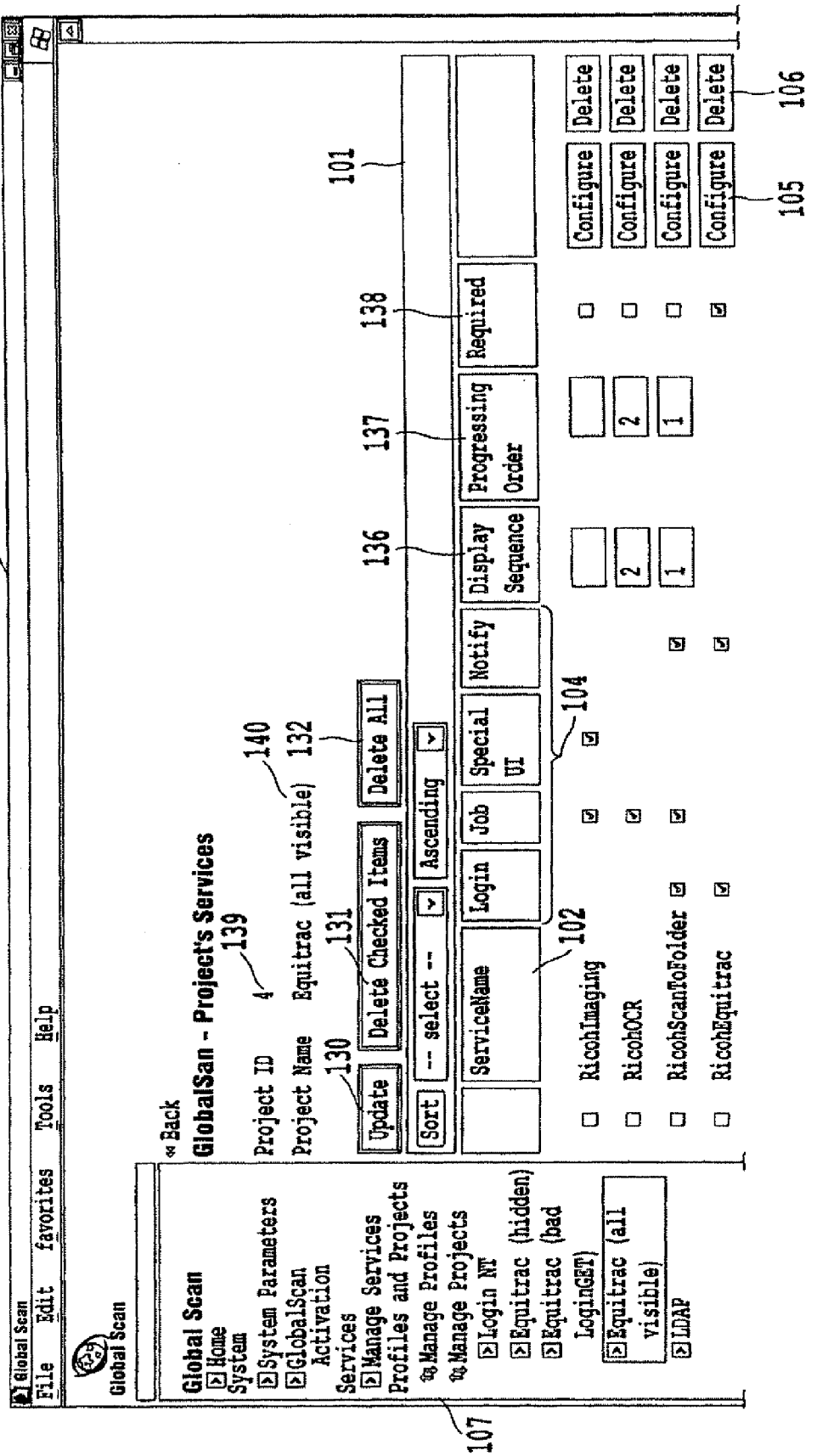
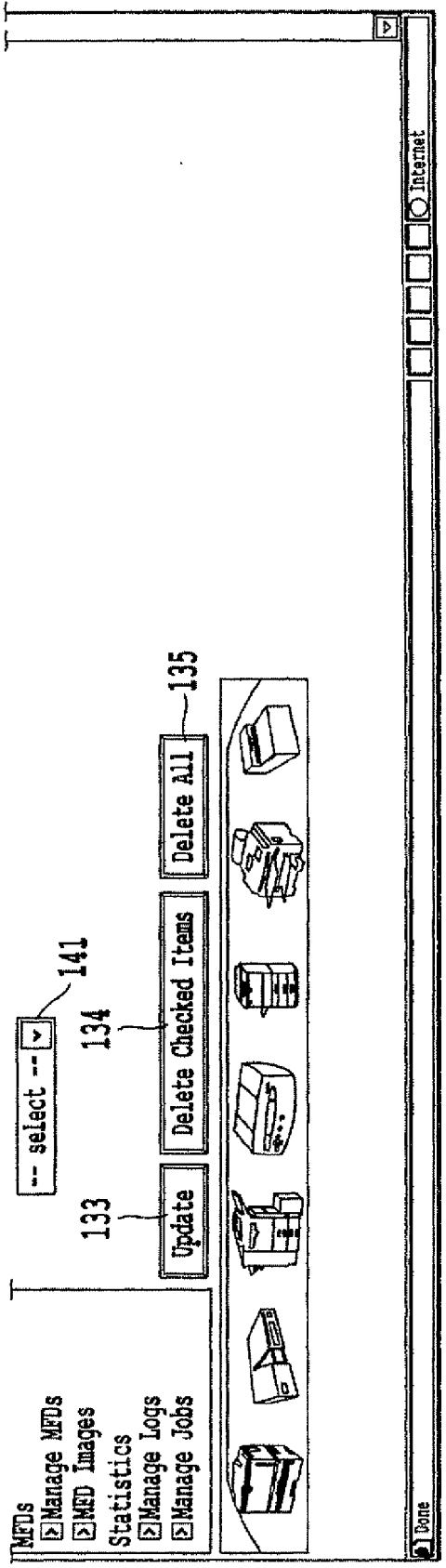

110
Global Scan
File Edit favorites Tools Help
Global Scan
Global Scan
Home
System
System Parameters
GlobalScan Activation
Services
Manage Services
Profiles and Projects
Manage Profiles
Manage Projects
Login NT
Equitrac (hidden)
Equitrac (bad LoginGEM)
Equitrac (all visible)
LDAP
107
Back
GlobalSan - Project's Services
139
Project ID 4
140
Project Name Equitrac (all visible)
132
130
Update
131
Delete Checked Items
Delete All
Sort
-- select --
Ascending
ServiceName
Login
Job
Special UI
Notify
Display Sequence
136
Progressing Order
137
Required
138
101
102
104
RicohImaging
RicohOCR
RicohScanToFolder
RicohEquitrac
Configure
Delete
105
106

FIG.52B


MFDs
Manage MFDs
MFD Images
Statistics
Manage Logs
Manage Jobs
Update
-- select --
Delete Checked Items
Delete All
133
134
135
141
Internet
Done

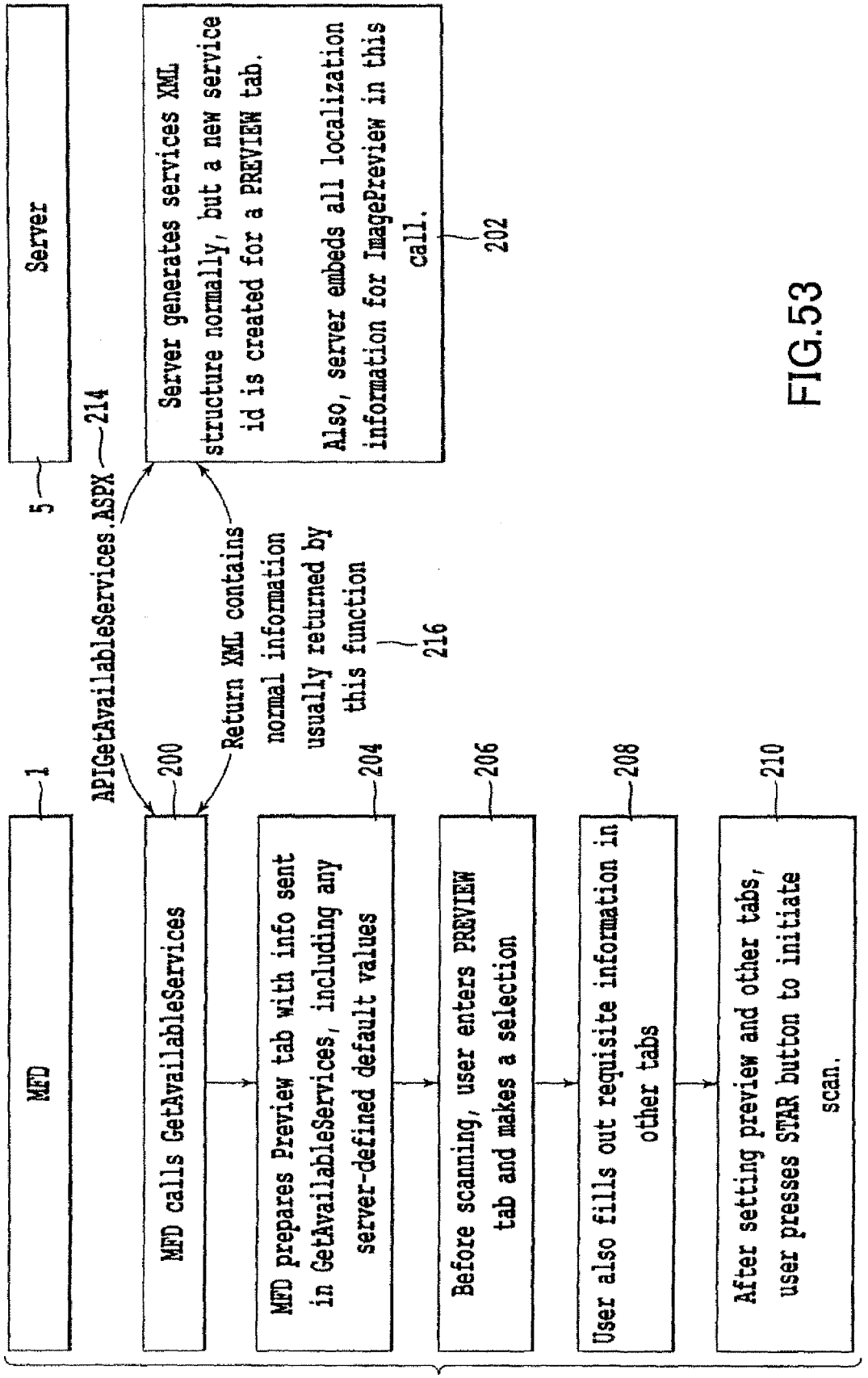

MFD
1
Server
5
APIGetAvailableServices.ASPX
214
MFD calls GetAvailableServices
200
Server generates services XML structure normally, but a new service id is created for a PREVIEW tab.
Also, server embeds all localization information for ImagePreview in this call.
202
Return XML contains normal information usually returned by this function
216
MFD prepares Preview tab with info sent in GetAvailableServices, including any server-defined default values
204
Before scanning, user enters PREVIEW tab and makes a selection
206
User also fills out requisite information in other tabs
208
After setting preview and other tabs, user presses START button to initiate scan.
210

FIG.53

FIG.54A

```
<screen_data>
  <options>
    <item>
      <item_name default="1">None</item_name>
      <item_value>0</item_value>
    </item>
    <item>
      <item_name default="0">All</item_name>
      <item_value>all</item_value>
    </item>
    <item>
      <item_name default="0">First 5</item_name>
      <item_value>F5</item_value>
    </item>
    <item>
      <item_name default="0">Last 3</item_name>
      <item_value>L3</item_value>
    </item>
    <item>
      <item_name default="0">First 3, Last 2</item_name>
      <item_value>F3,L2</item_value>
    </item>
  </options>
    <localization>
      <preview type="text">Preview</preview>
      <thumbnail_widthtype="text">
          ThumbNail Width
      <thumbnail_width>
      <thumbnail_height type="text">
          ThumbNail Height
      <thumbnail_height>
      <page_select type = "text">Page Range<page_select>
      <thumbnail_title type= "text">
          Thumbnail Select
      </thumbnail_title>
```

FIG.54B

```
<thumbnail_pages type="text">
    Pages
</thumbnail_pages>
<thumbnail_restore type="text"/"image">
    Restore
</thumbnail_restore>
<thumbnail_cancel type="text"/"image">
    Cancel
</thumbnail_cancel>
<thumbnail_submit type="text"/"image">
    Submit
</thumbnail_submit>
<page_information type="text">
    Page Information
</page_information>
<page_size type="text">
    Size
<page_size>
<page_color_mode type="text">
    Color Mode
</page_color_mode>
<preview_back type="text"/"image">
    Back
</preview_back>
<preview_previous type="text"/"image">
    Previous
</preview_previous>
<preview_next type="text"/"image">
    Next
</preview_next>
<preview_rotate_clockwise type="text"/"image">
    Rotate Clockwise
</preview_preview_rotate_clockwise>
<preview_rotate_anticlockwise type="text"/"image">
    Rotate Anti Clockwise
```

FIG.54C

```
1                </preview_preview_rotate_anticlockwise>
2                <preview_zoomin type="text"/"image">
3                    Zoom In
4                <preview_zoomin>
5                <preview_zoomout type="text"/"image">
6                    Zoom Out
7                <preview_zoomout>
8                <preview_delete type="text"/"image">
9                    Delete
10               <preview_delete>
11             </localization>
12       <zoom_levels>
13          <zoom_value>20</zoom_value>
14          <zoom_value>30</zoom_value>
15          <zoom_value>60</zoom_value>
16          <zoom_value>90</zoom_value>
17          <zoom_value>100</zoom_value>
18       </zoom_levels>
19   </screen_data>
```

FIG.55A

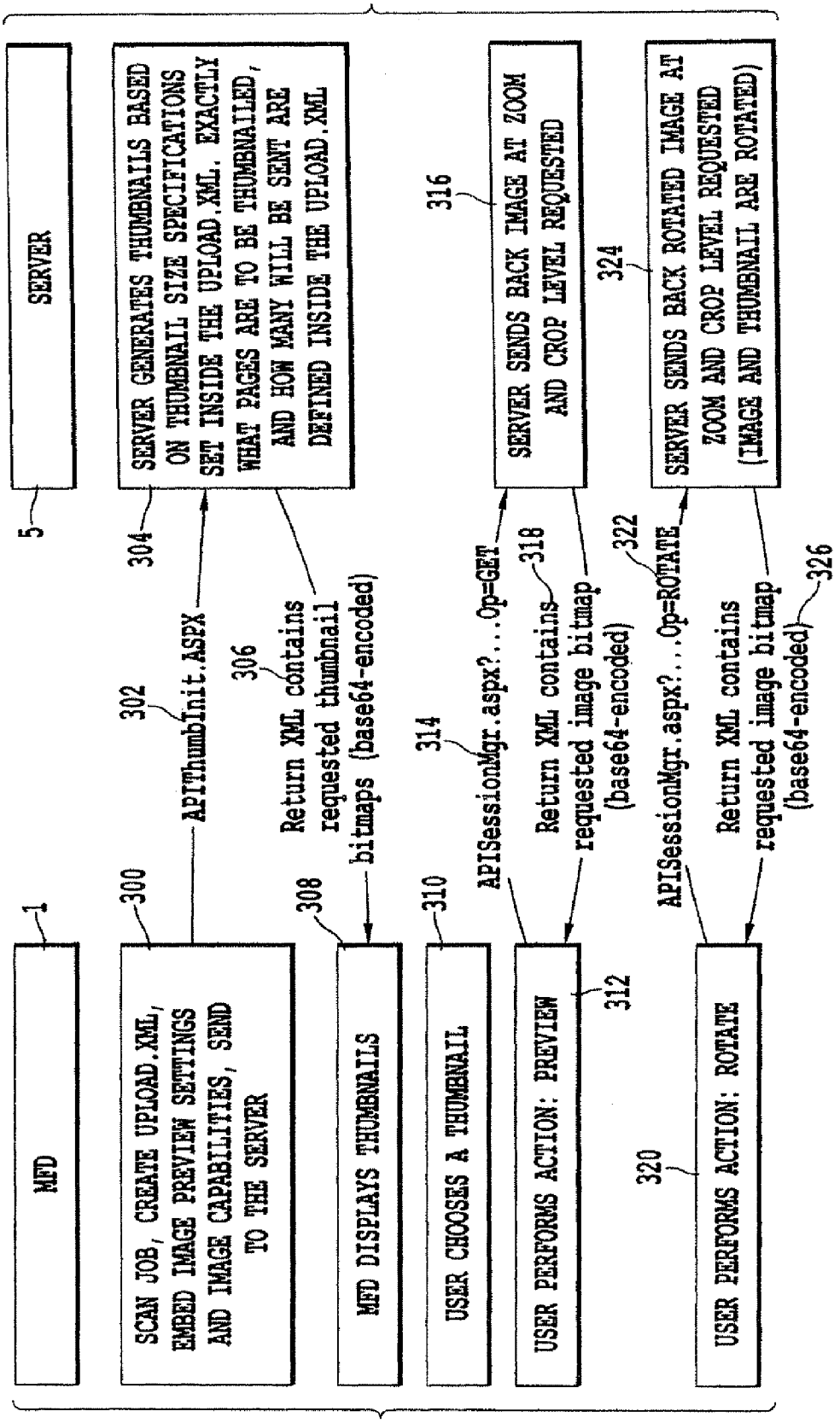
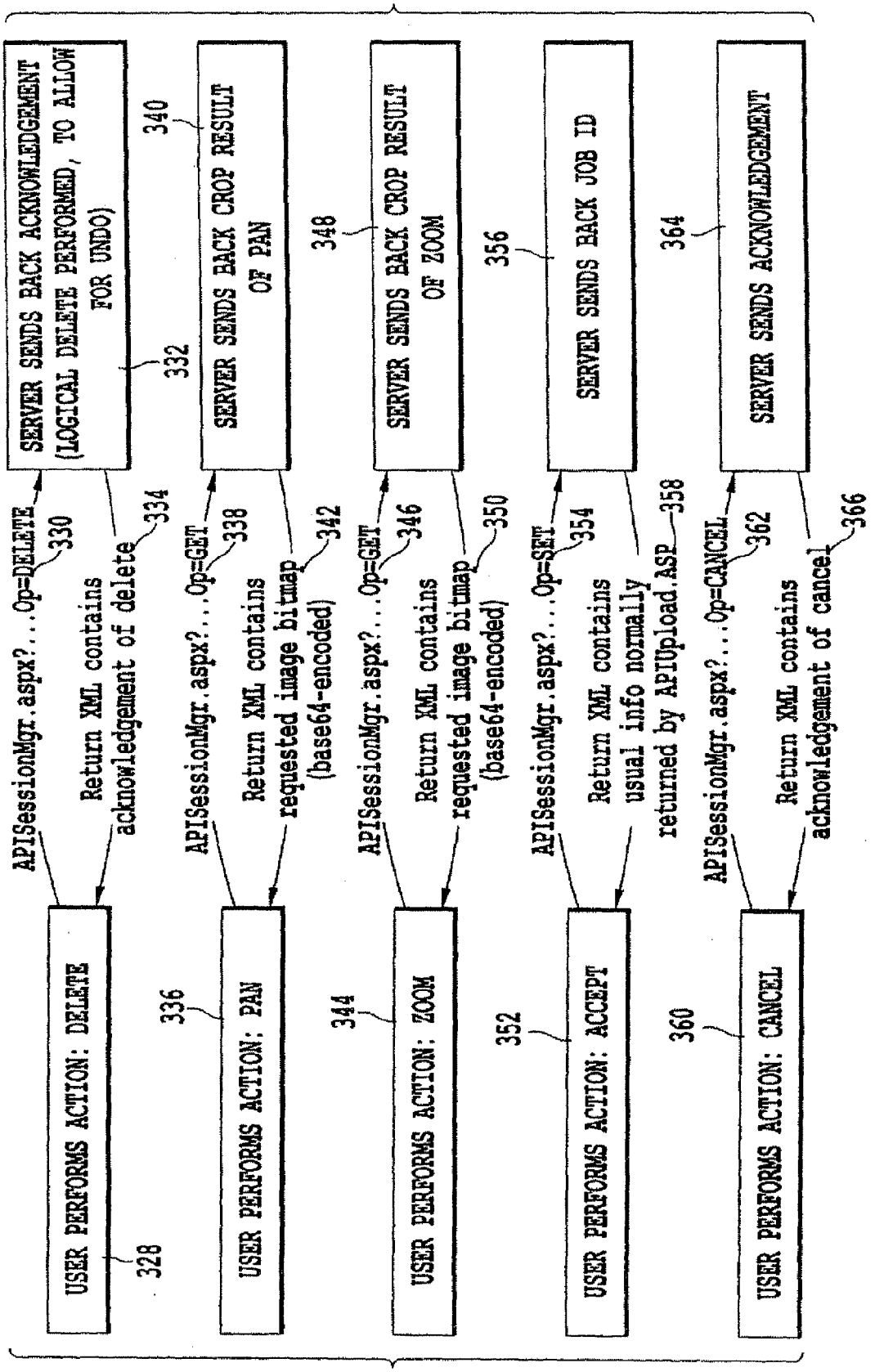

MFD
SERVER
1
5
SCAN JOB, CREATE UPLOAD.XML, EMBED IMAGE PREVIEW SETTINGS AND IMAGE CAPABILITIES, SEND TO THE SERVER
SERVER GENERATES THUMBNAILS BASED ON THUMBNAIL SIZE SPECIFICATIONS SET INSIDE THE UPLOAD.XML. EXACTLY WHAT PAGES ARE TO BE THUMBNAILED, AND HOW MANY WILL BE SENT ARE DEFINED INSIDE THE UPLOAD.XML
300
304
APIThumbInit.ASPX
302
Return XML contains requested thumbnail bitmaps (base64-encoded)
306
308
MFD DISPLAYS THUMBNAILS
USER CHOOSES A THUMBNAIL
310
APISessionMgr.aspx?...Op=GET
314
SERVER SENDS BACK IMAGE AT ZOOM AND CROP LEVEL REQUESTED
316
Return XML contains requested image bitmap (base64-encoded)
318
USER PERFORMS ACTION: PREVIEW
312
APISessionMgr.aspx?...Op=ROTATE
322
SERVER SENDS BACK ROTATED IMAGE AT ZOOM AND CROP LEVEL REQUESTED (IMAGE AND THUMBNAIL ARE ROTATED)
324
Return XML contains requested image bitmap (base64-encoded)
326
USER PERFORMS ACTION: ROTATE
320

FIG.55B


USER PERFORMS ACTION: DELETE
328
APISessionMgr.aspx?....Op=DELETE
330
Return XML contains acknowledgement of delete
334
SERVER SENDS BACK ACKNOWLEDGEMENT (LOGICAL DELETE PERFORMED, TO ALLOW FOR UNDO)
332
340
USER PERFORMS ACTION: PAN
336
APISessionMgr.aspx?....Op=GET
338
Return XML contains requested image bitmap (base64-encoded)
342
SERVER SENDS BACK CROP RESULT OF PAN
348
USER PERFORMS ACTION: ZOOM
344
APISessionMgr.aspx?....Op=GET
346
Return XML contains requested image bitmap (base64-encoded)
350
SERVER SENDS BACK CROP RESULT OF ZOOM
356
USER PERFORMS ACTION: ACCEPT
352
APISessionMgr.aspx?....Op=SET
354
Return XML contains usual info normally returned by APIUpload.ASP
358
SERVER SENDS BACK JOB ID
364
USER PERFORMS ACTION: CANCEL
360
APISessionMgr.aspx?....Op=CANCEL
362
Return XML contains acknowledgement of cancel
366
SERVER SENDS ACKNOWLEDGEMENT

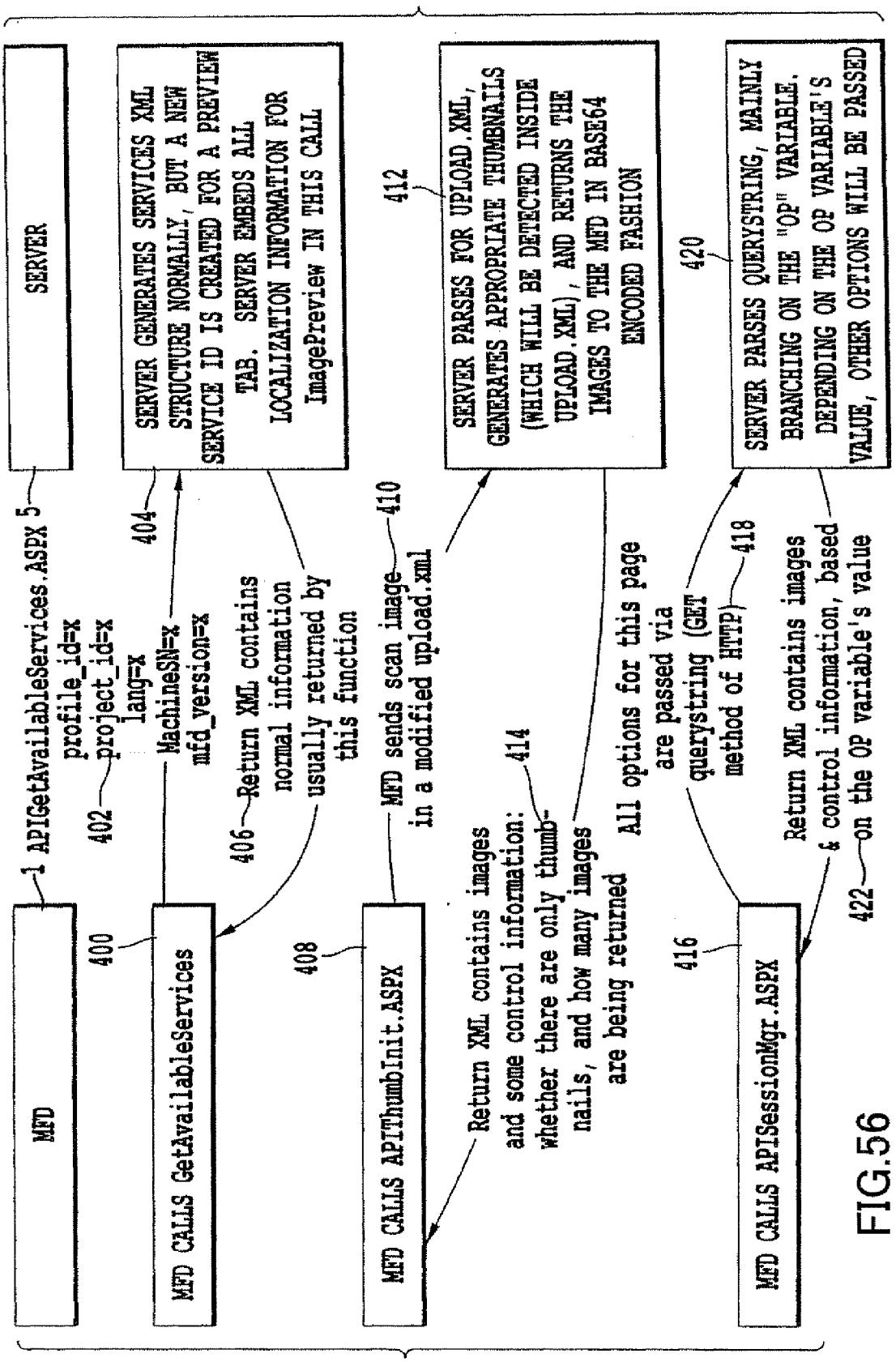

MFD
SERVER
400
1 APIGetAvailableServices.ASPX 5
profile_id=x
project_id=x
lang=x
MachineSN=x
mfd_version=x
402
404
MFD CALLS GetAvailableServices
SERVER GENERATES SERVICES XML STRUCTURE NORMALLY, BUT A NEW SERVICE ID IS CREATED FOR A PREVIEW TAB. SERVER EMBEDS ALL LOCALIZATION INFORMATION FOR ImagePreview IN THIS CALL
406 Return XML contains normal information usually returned by this function
408
MFD CALLS APIThumbInit.ASPX
MFD sends scan image in a modified upload.xml 410
412
SERVER PARSES FOR UPLOAD.XML, GENERATES APPROPRIATE THUMBNAILS (WHICH WILL BE DETECTED INSIDE UPLOAD.XML), AND RETURNS THE IMAGES TO THE MFD IN BASE64 ENCODED FASHION
Return XML contains images and some control information: whether there are only thumbnails, and how many images are being returned 414
All options for this page are passed via querystring (GET method of HTTP) 418
420
SERVER PARSES QUERYSTRING, MAINLY BRANCHING ON THE "OP" VARIABLE, DEPENDING ON THE OP VARIABLE'S VALUE, OTHER OPTIONS WILL BE PASSED
416
MFD CALLS APISessionMgr.ASPX
Return XML contains images & control information, based on the OP variable's value 422

FIG.56

FIG.57

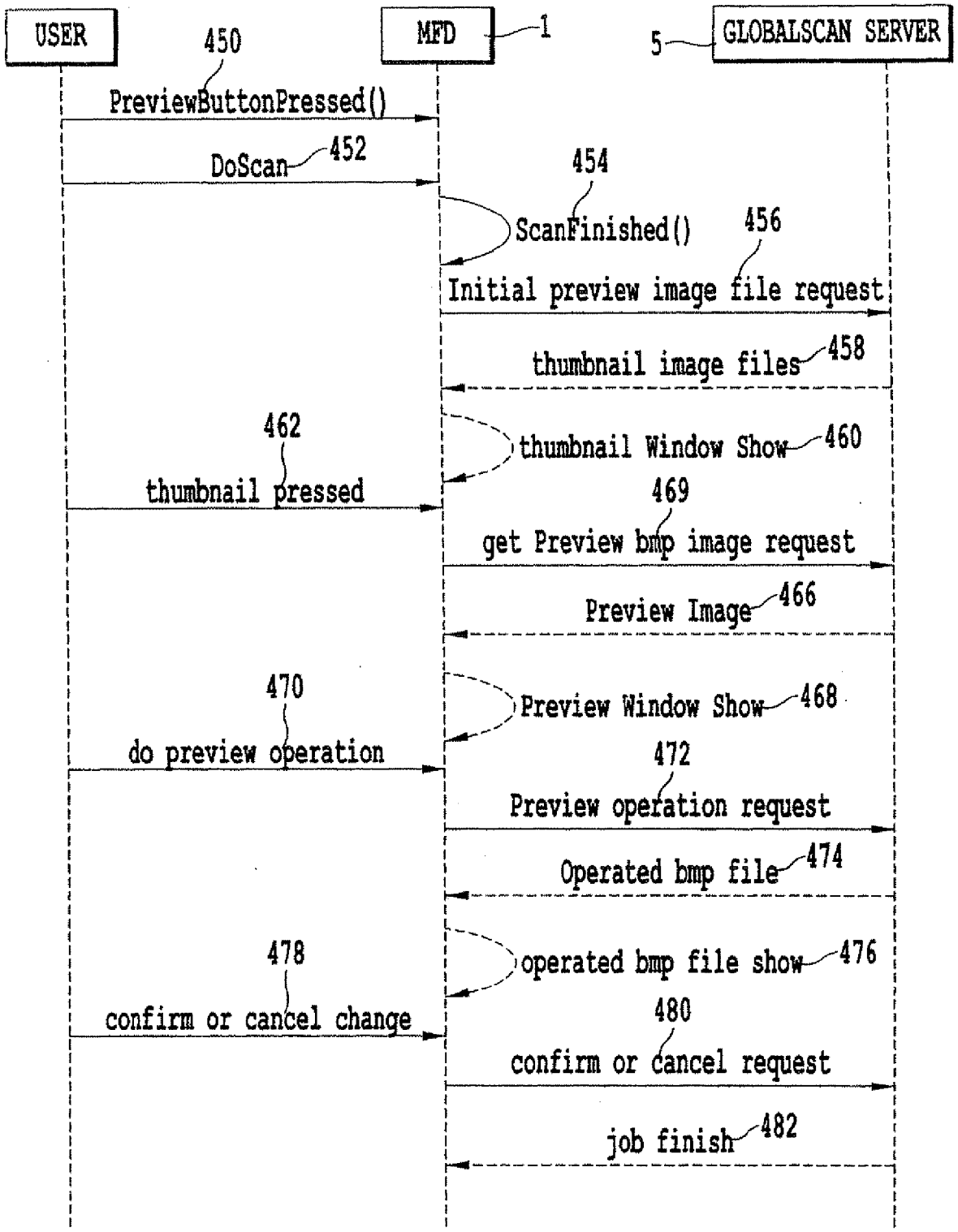

USER
MFD
1
5
GLOBALSCAN SERVER
450
PreviewButtonPressed()
DoScan
452
454
ScanFinished()
456
Initial preview image file request
thumbnail image files
458
462
thumbnail Window Show
460
thumbnail pressed
469
get Preview bmp image request
Preview Image
466
470
Preview Window Show
468
do preview operation
472
Preview operation request
Operated bmp file
474
478
operated bmp file show
476
confirm or cancel change
480
confirm or cancel request
job finish
482

FIG.58A

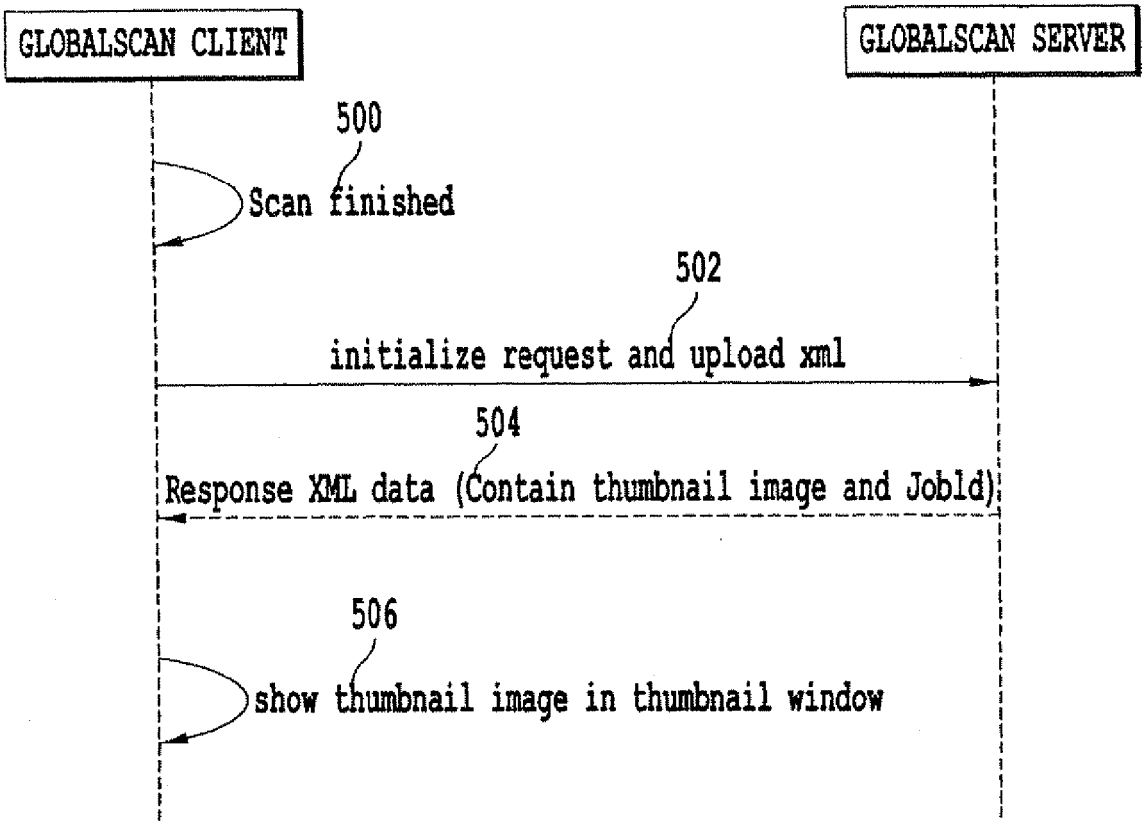

GLOBALSCAN CLIENT
GLOBALSCAN SERVER
500
Scan finished
502
initialize request and upload xml
504
Response XML data (Contain thumbnail image and JobId)
506
show thumbnail image in thumbnail window

FIG.58B

```
<root>
    <error_code/>
    <error_description/>
    <primary/>
    <secondary/>
    <server_status/>
    <total_page_number>25</total_page_number>
    <page_number>2</page_number>
    <thumbnail_only>1</thumbnail_only>
    <data type="thumbnail" id="1" pagetype="*********">
        <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXXX<
        /bmpdata>
    </data>
    <data type="thumbnail" id="2" pagetype="*********">
        <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXXX<
        /bmpdata>
    </data>
</root>
```

FIG.59A

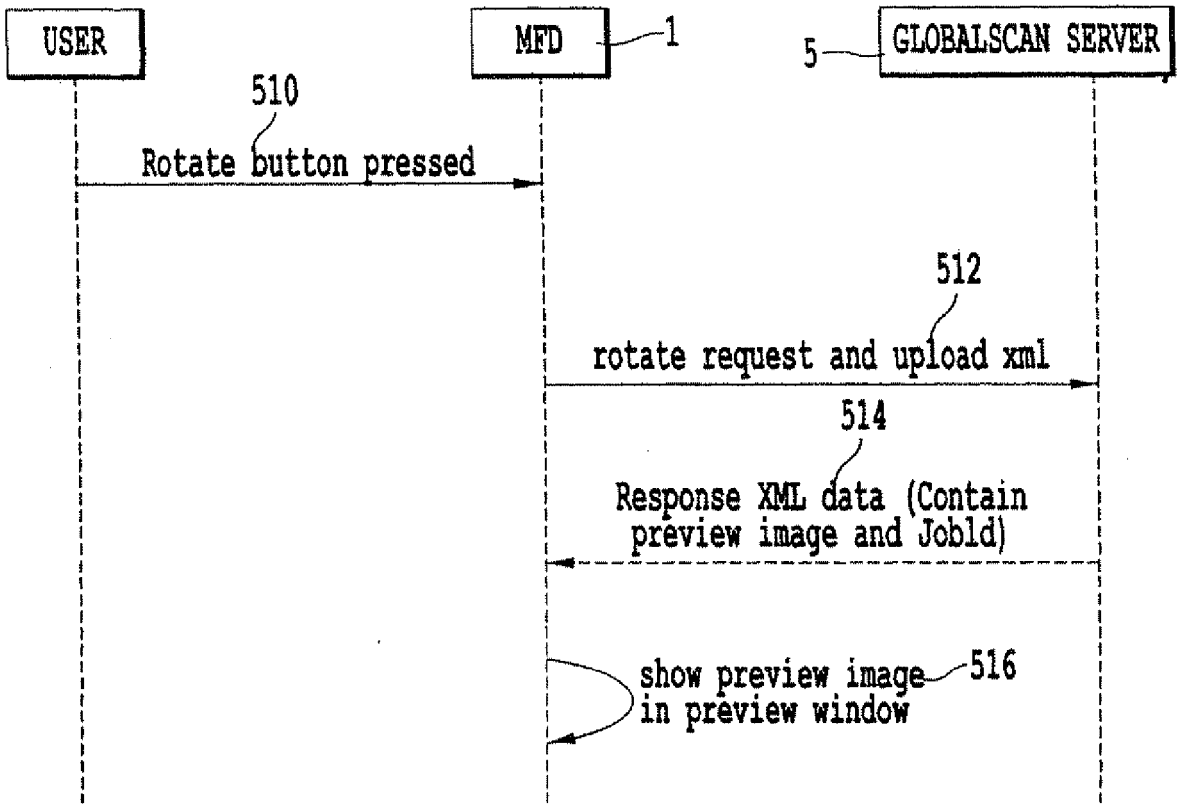

USER
MFD
1
5
GLOBALSCAN SERVER
510
Rotate button pressed
512
rotate request and upload xml
514
Response XML data (Contain preview image and JobId)
show preview image in preview window
516

FIG.59B

```
HTTP{S}://server{:port}/GSClient/APISessionMgr.aspx?
action=preview&action_input=op=rotate
&image_no=
&degrees=
&request_image=
&zoom=
&follow_me=


<root>
    <error_code/>
    <error_description/>
    <primary/>
    <secondary/>
    <server_status/>
    <thumbnail_only>0</thumbnail_only>
    <data type="thumbnail" id="1" >
        <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXXX<
        /bmpdata>
    </data>
    <data type="preview" id="1" width="80" height="100"
    zoom_init="30" pagetype="**********">
        <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXXX<
        /bmpdata>
    </data>

```

FIG.60A

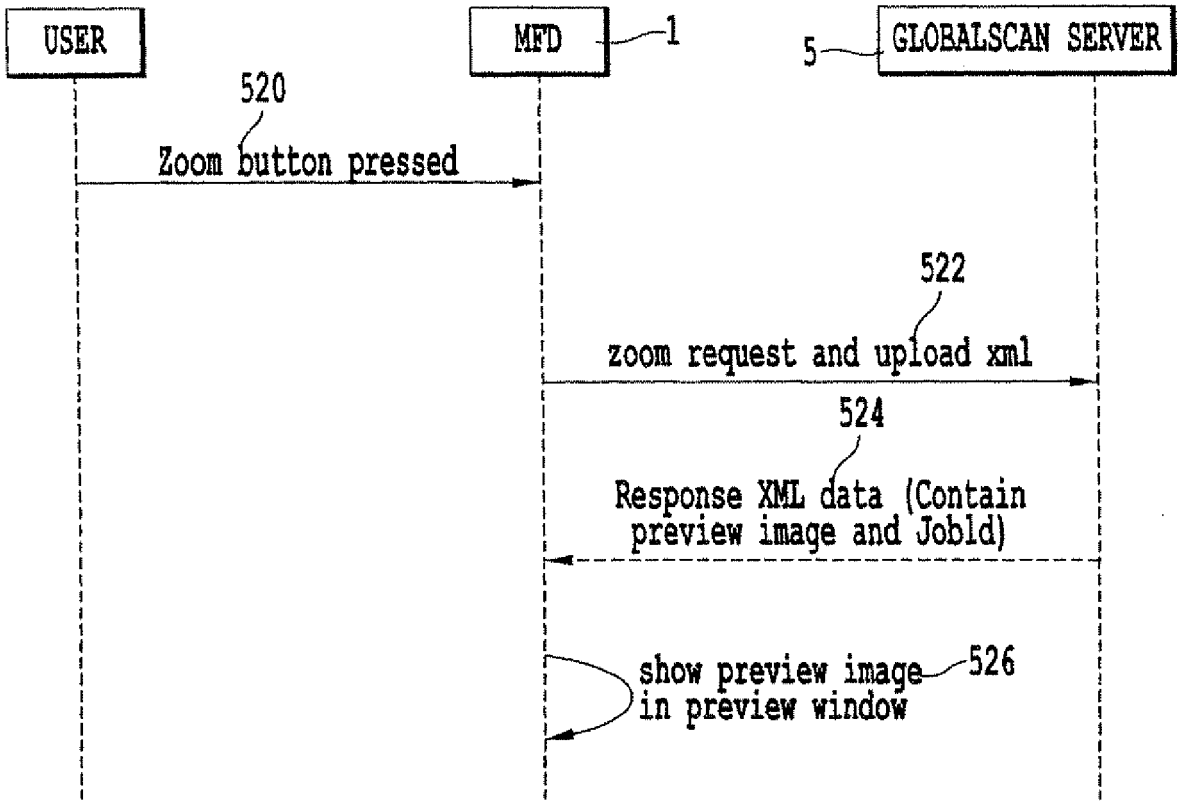

USER
MFD
1
5
GLOBALSCAN SERVER
520
Zoom button pressed
522
zoom request and upload xml
524
Response XML data (Contain preview image and JobId)
show preview image in preview window
526

FIG.60B

```
HTTP{S}://server{:port}/GSClient/APISessionMgr.aspx?
action=preview&action_input=op=get
&image_no=
&zoom=
&sizeX=
&sizeY=
&request_image=
&direction=

<root>
    <error_code/>
    <error_description/>
    <primary/>
    <secondary/>
    <server_status/>
    <thumbnail_only>0</thumbnail_only>
    <data type="preview" id="1" width="80" height="100"
    zoom_init="30" pagetype="********">
        <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXXX<
        /bmpdata>
    </data>
    <data type="thumbnail" id="1">
        <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXXX<
        /bmpdata>
    </data>
</root>
```

FIG.61A

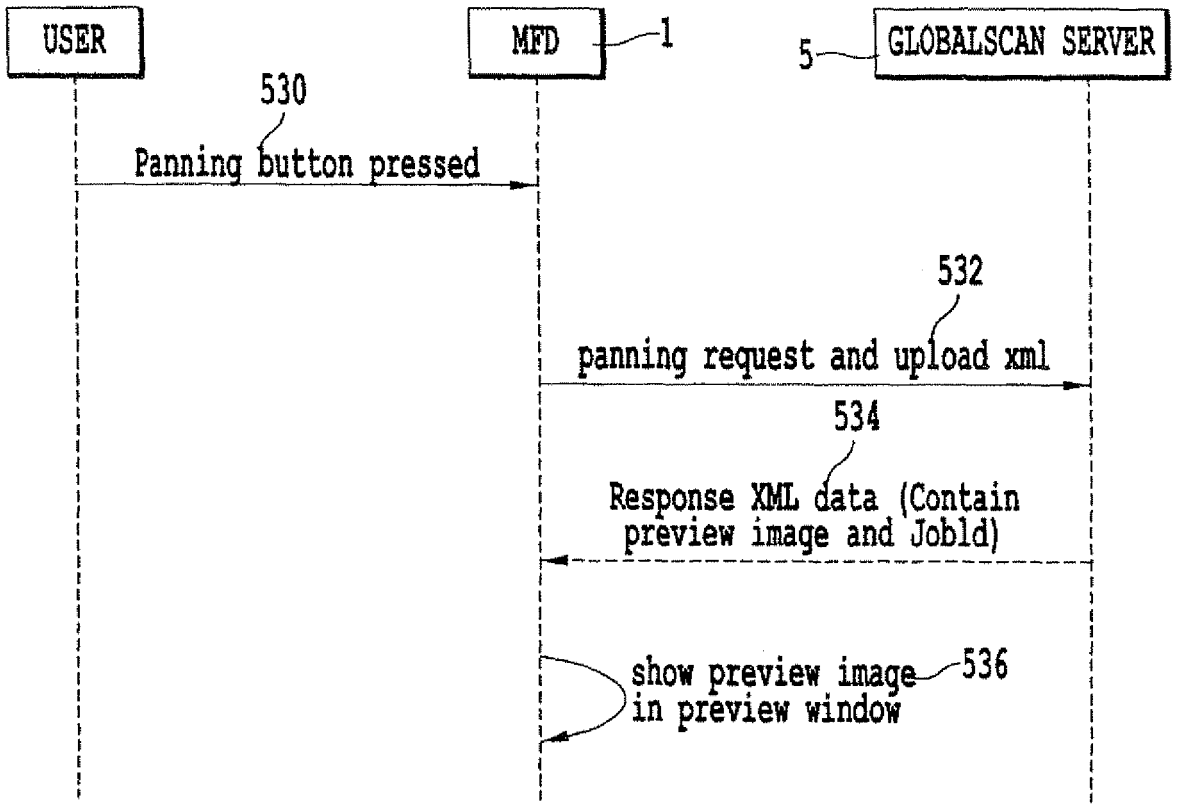

USER
MFD
1
5
GLOBALSCAN SERVER
530
Panning button pressed
532
panning request and upload xml
534
Response XML data (Contain preview image and JobId)
show preview image in preview window
536

FIG.61B

```
HTTP{S}://server{:port}/GSClient/APISessionMgr.aspx?
action=preview&action_input=op=get
&image_no=
&zoom=
&sizeX=
&sizeY=
&request_image=
&direction=

 <root>
      <error_code/>
      <error_description/>
      <primary/>
      <secondary/>
      <server_status/>
      <thumbnail_only>0</thumbnail_only>
      <data type="preview" id="1" width="80" height="100"
      zoom_init="30" pagetype="********">
           <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXXX
           </bmpdata>
      </data>
 </root>
```

FIG.62A

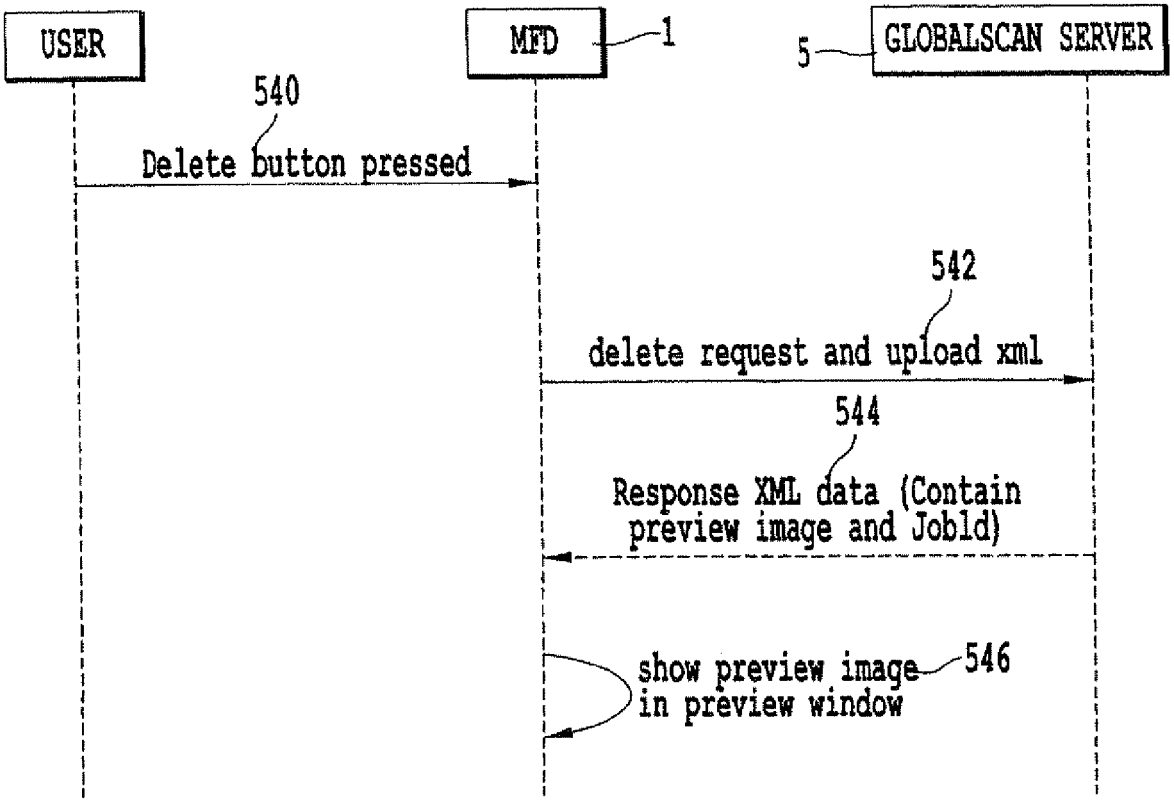

USER
MFD
1
5
GLOBALSCAN SERVER
540
Delete button pressed
542
delete request and upload xml
544
Response XML data (Contain preview image and JobId)
show preview image in preview window
546

FIG.62B

```
HTTP{S}://server{:port}/GSClient/APISessionMgr.aspx?
action=preview&action_input=
    op=delete
    &image_no=

    <root>
        <error_code/>
        <error_description/>
        <primary/>
        <secondary/>
        <server_status/>
        <thumbnail_only>0</thumbnail_only>
        <data type="preview" id="1" width="80" height="100"
        zoom_init="30" pagetype="3">
            <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXX
            X</bmpdata>
        </data>
    </root>
```

FIG.63A

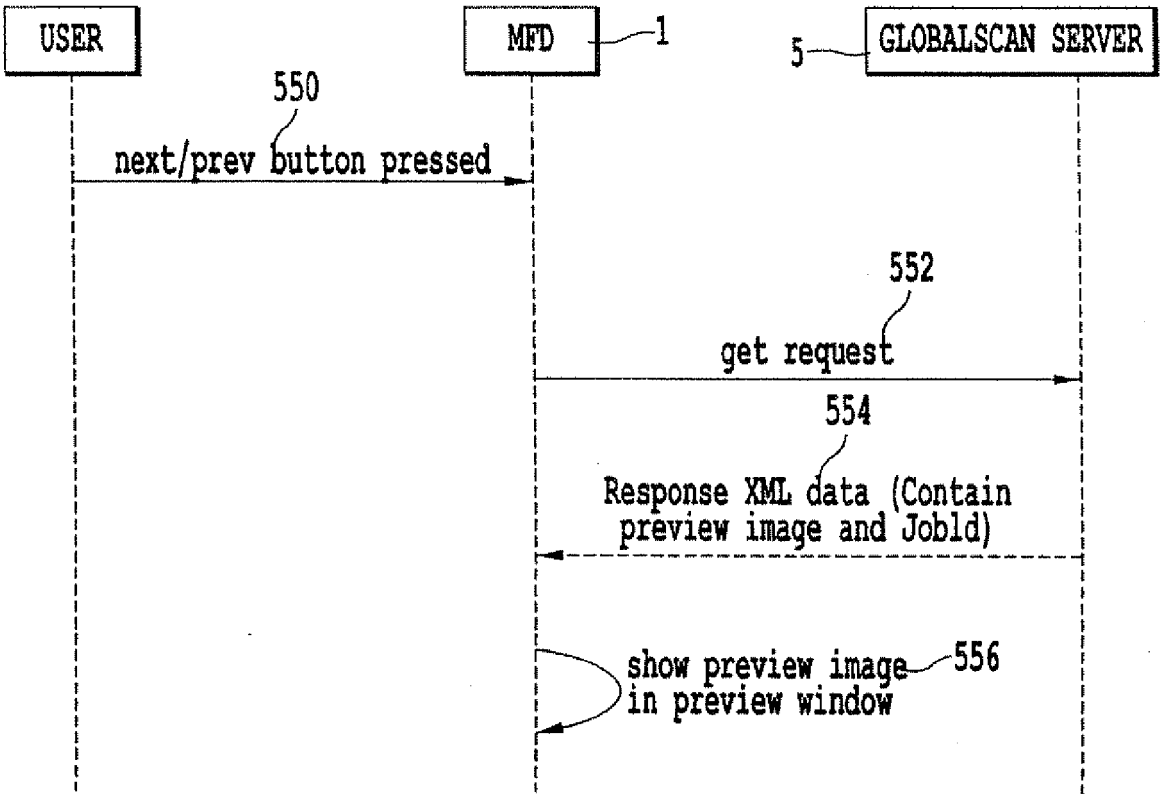

USER
MFD
1
5
GLOBALSCAN SERVER
550
next/prev button pressed
552
get request
554
Response XML data (Contain preview image and JobId)
show preview image in preview window
556

FIG.63B

```
HTTP{S}://server{:port}/GSClient/APISessionMgr.aspx?
action=preview&action_input=op=get
&image_no=
&zoom=
&sizeX=
&sizeY=
&request_image=
&direction=

    <root>
        <error_code/>
        <error_description/>
        <primary/>
        <secondary/>
        <server_status/>
        <thumbnail_only>0</thumbnail_only>
        <data type="preview" id="1" width="80" height="100"
        zoom_init="30" pagetype="********">
            <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXX
            X</bmpdata>
        </data>
        <data type="thumbnail" id="1">
            <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXX
            X</bmpdata>
        </data>
    </root>
```

FIG.64A

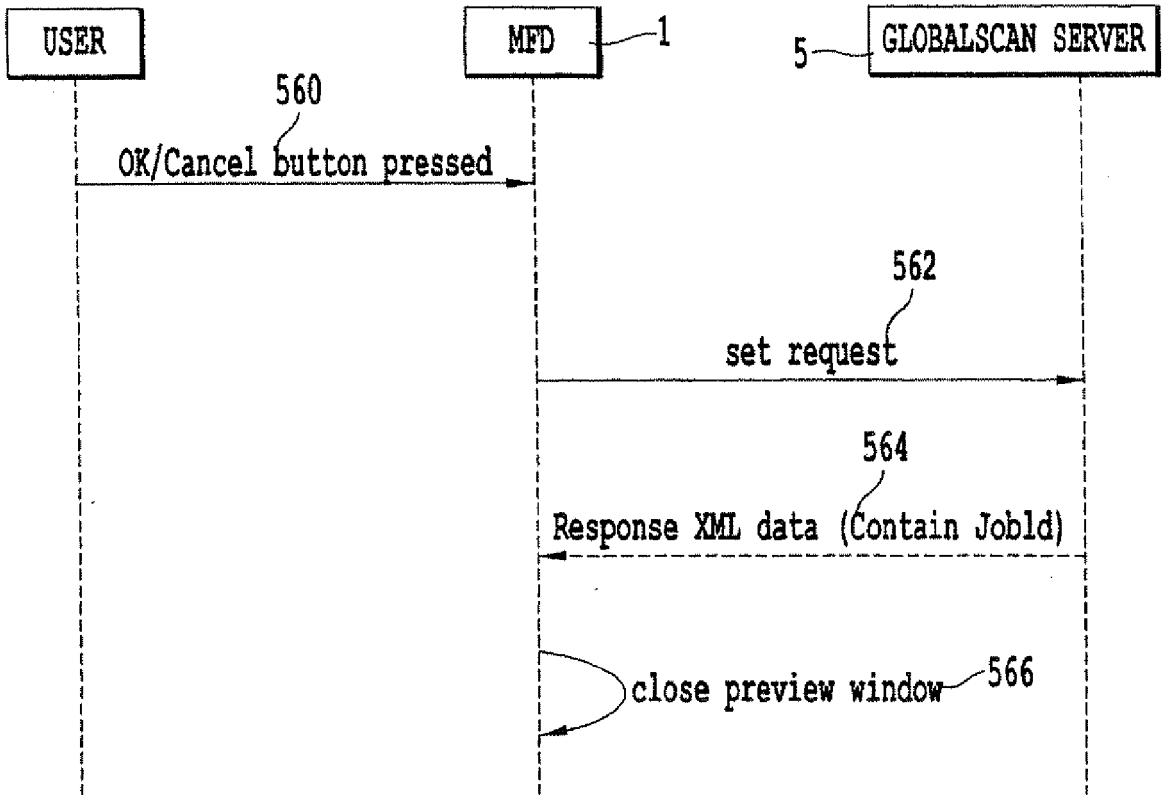

USER
MFD
1
5
GLOBALSCAN SERVER
560
OK/Cancel button pressed
562
set request
564
Response XML data (Contain JobId)
close preview window
566

FIG.64B

```
HTTP{S}://server{:port}/GSClient/APISessionMgr.aspx?
action=preview &action_input=  op = set

HTTP{S}://server{:port}/GSClient/APISessionMgr.aspx?
action=preview &action_input= op = cancel

HTTP{S}://server{:port}/GSClient/APISessionMgr.aspx?
action=preview &action_input= op = restore

    <root>
        <error_code/>
        <error_description/>
        <primary/>
        <secondary/>
        <server_status/>
        <total_page_number>25</total_page_number>
        <page_number>2</page_number>
        <thumbnail_only>1</thumbnail_only>
        <data type="thumbnail" id="1" pagetype="********">
            <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXX
            X</bmpdata>
        </data>
        <data type="thumbnail" id="2" pagetype="********">
            <bmpdata>XXXXXXXXXXXXXXXXXXXXXXXX
            X</bmpdata>
        </data>
    </root>
```

FIG.65

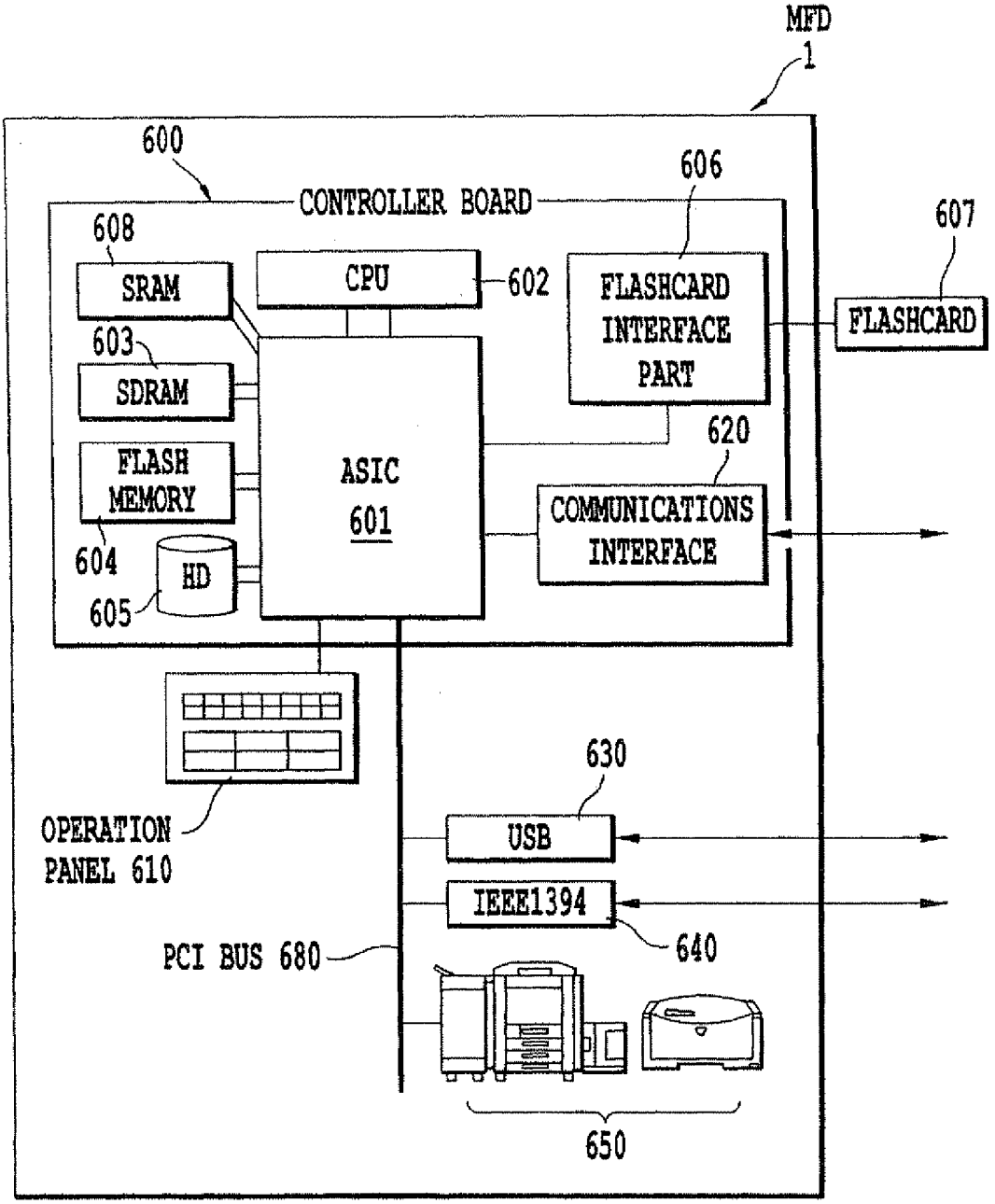
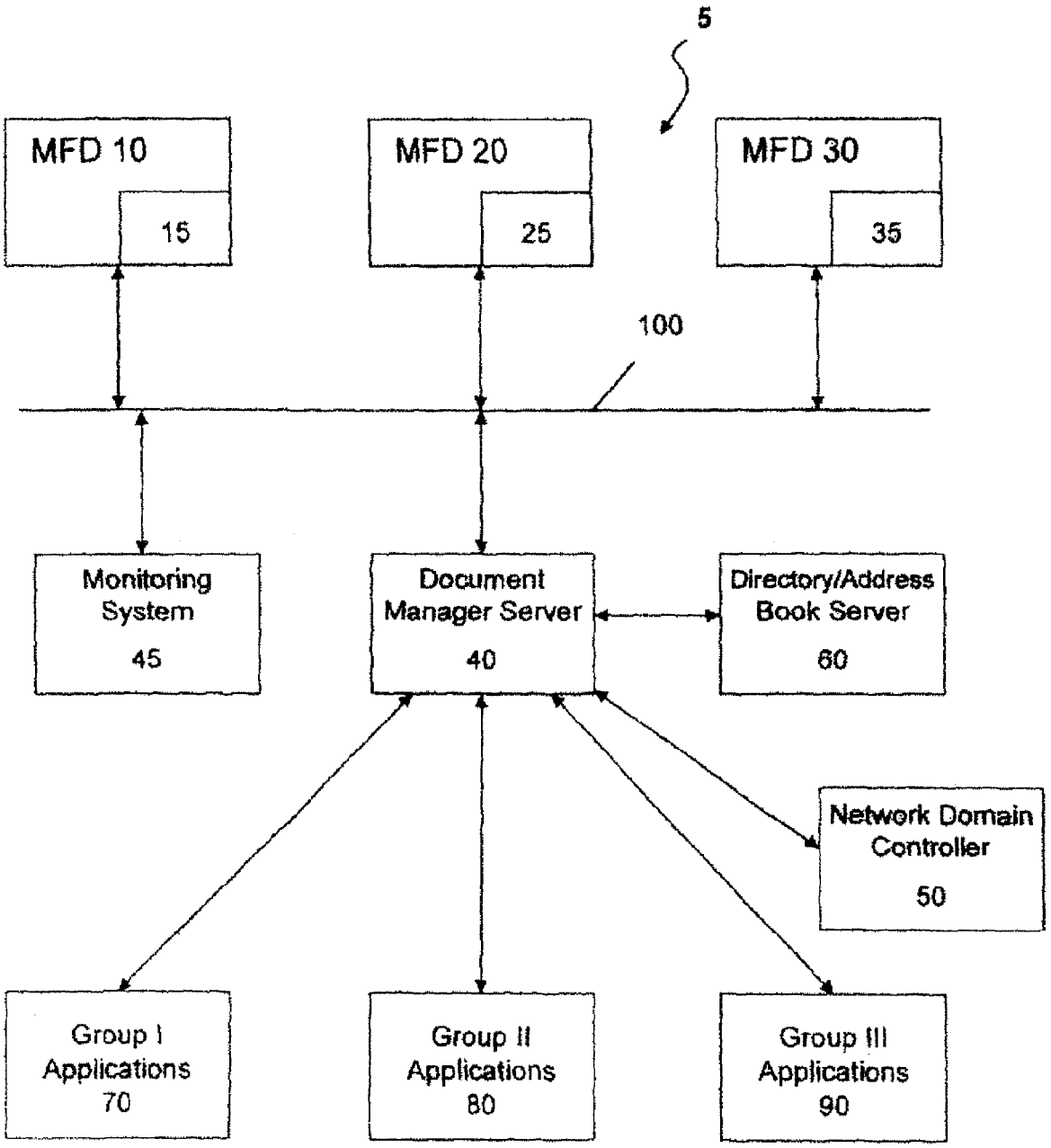

MFD
1
600
CONTROLLER BOARD
606
607
608
SRAM
CPU
602
FLASHCARD INTERFACE PART
FLASHCARD
603
SDRAM
FLASH MEMORY
ASIC
601
620
COMMUNICATIONS INTERFACE
604
HD
605
630
OPERATION PANEL 610
USB
IEEE1394
640
PCI BUS 680
650

FIG.66


5
MFD 10
15
MFD 20
25
MFD 30
35
100
Monitoring System
45
Document Manager Server
40
Directory/Address Book Server
60
Network Domain Controller
50
Group I Applications
70
Group II Applications
80
Group III Applications
90

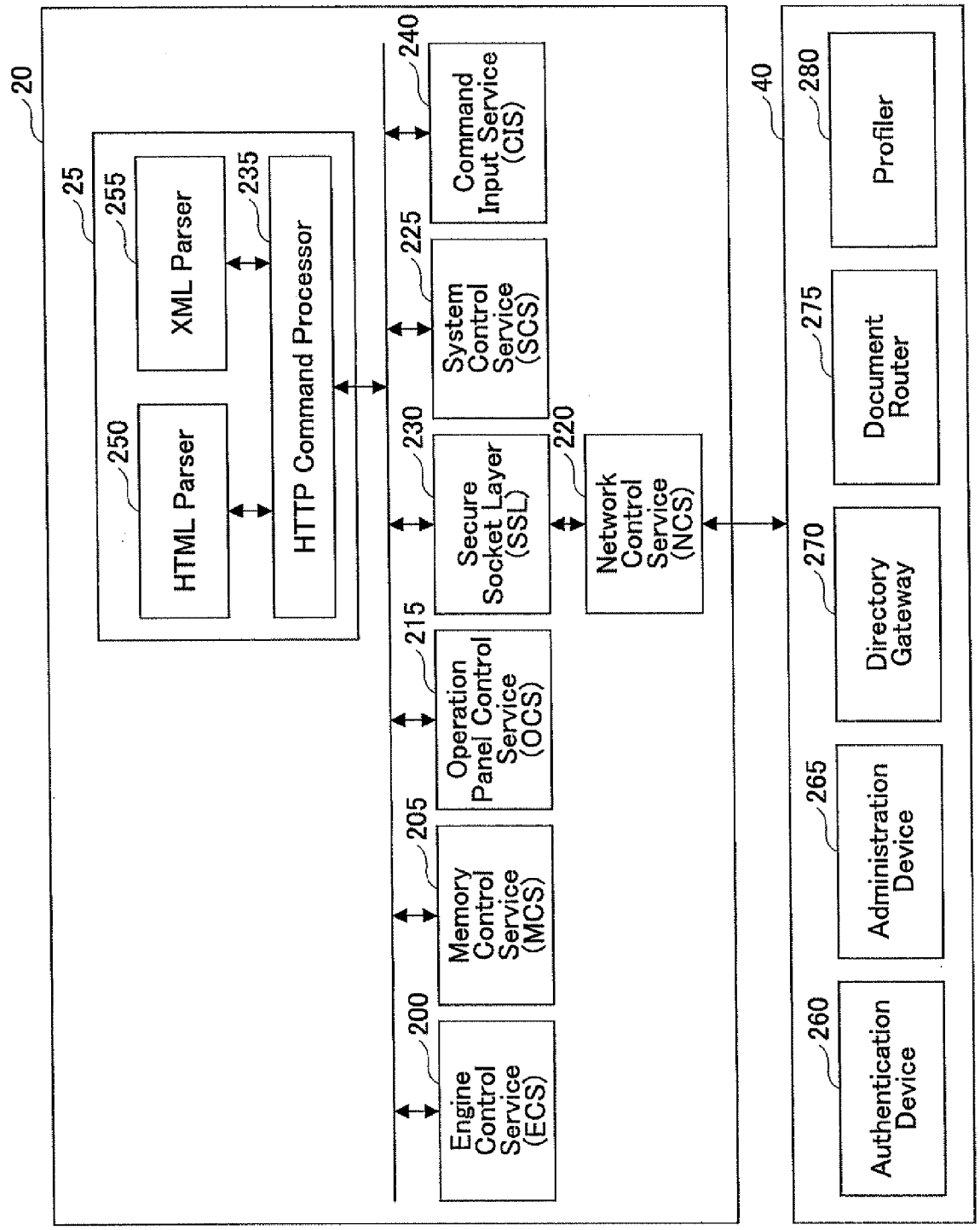

20
25
255
XML Parser
235
250
HTML Parser
HTTP Command Processor
240
Command Input Service (CIS)
225
System Control Service (SCS)
230
Secure Socket Layer (SSL)
220
Network Control Service (NCS)
215
Operation Panel Control Service (OCS)
205
Memory Control Service (MCS)
200
Engine Control Service (ECS)
40
280
Profiler
275
Document Router
270
Directory Gateway
265
Administration Device
260
Authentication Device

FIG.67

FIG.68

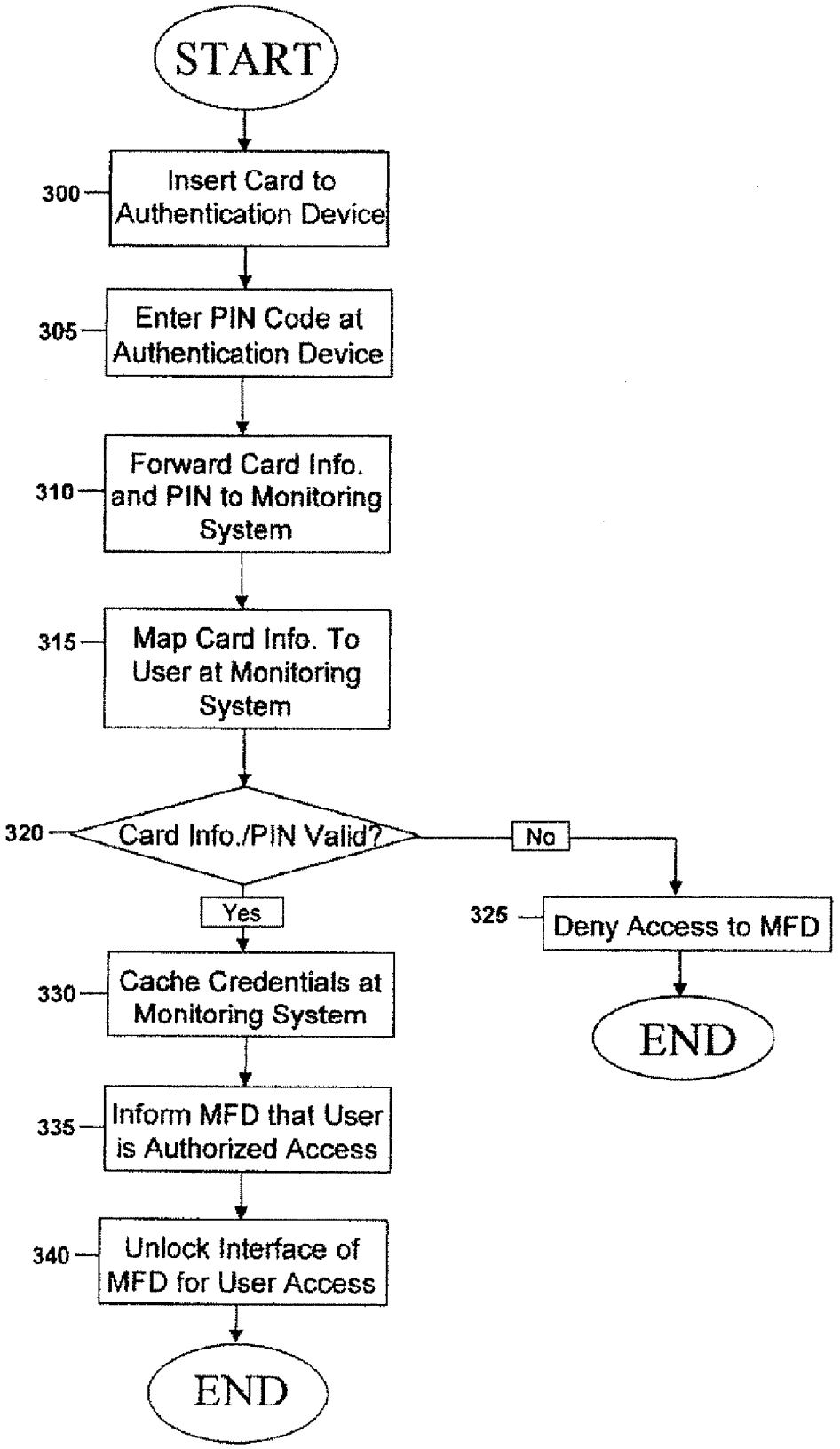

START
300
Insert Card to Authentication Device
305
Enter PIN Code at Authentication Device
310
Forward Card Info. and PIN to Monitoring System
315
Map Card Info. To User at Monitoring System
320
Card Info./PIN Valid?
No
325
Deny Access to MFD
END
Yes
330
Cache Credentials at Monitoring System
335
Inform MFD that User is Authorized Access
340
Unlock Interface of MFD for User Access
END

FIG.69

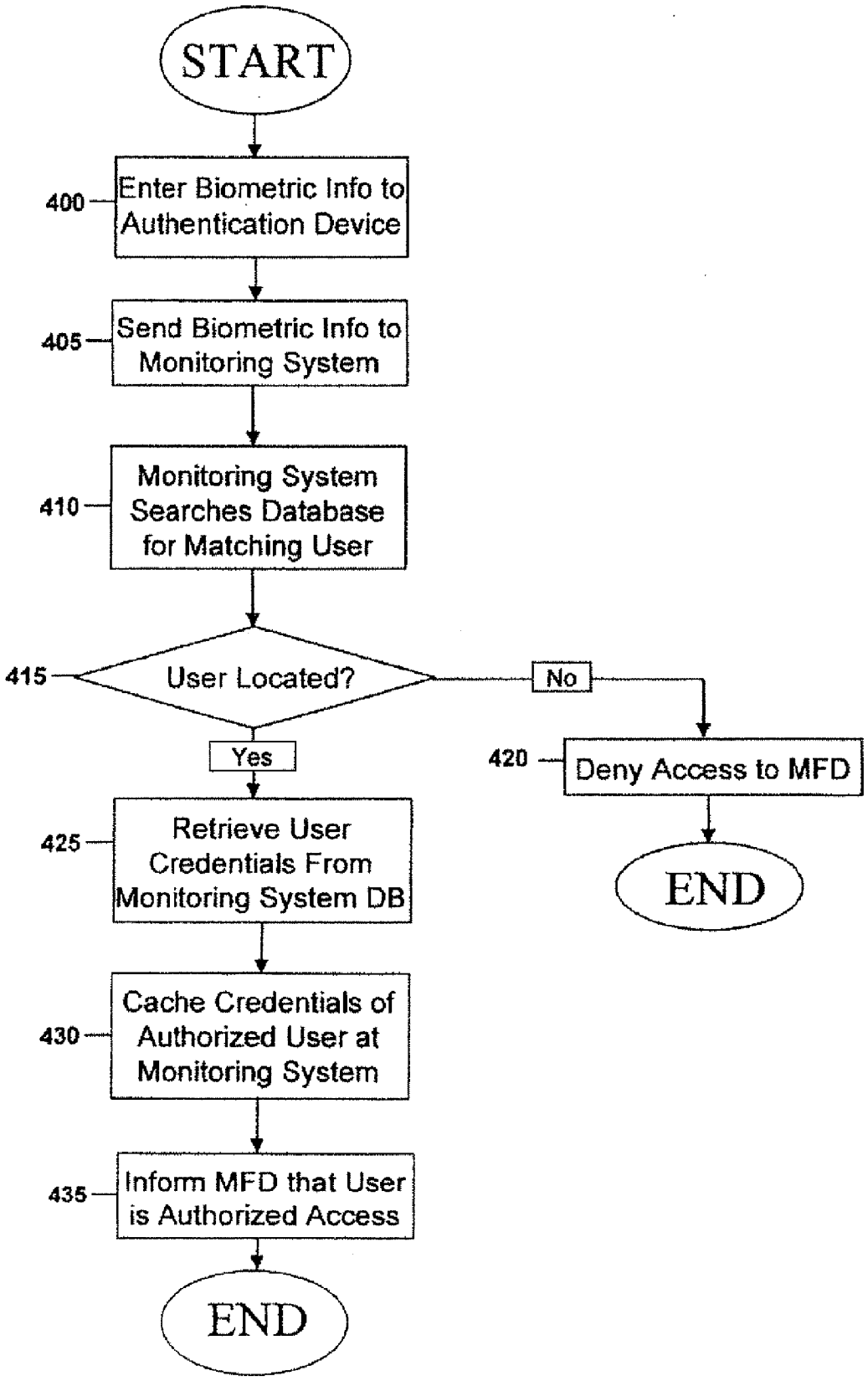

START
400
Enter Biometric Info to Authentication Device
405
Send Biometric Info to Monitoring System
410
Monitoring System Searches Database for Matching User
415
User Located?
No
420
Deny Access to MFD
END
Yes
425
Retrieve User Credentials From Monitoring System DB
430
Cache Credentials of Authorized User at Monitoring System
435
Inform MFD that User is Authorized Access
END

FIG.70A

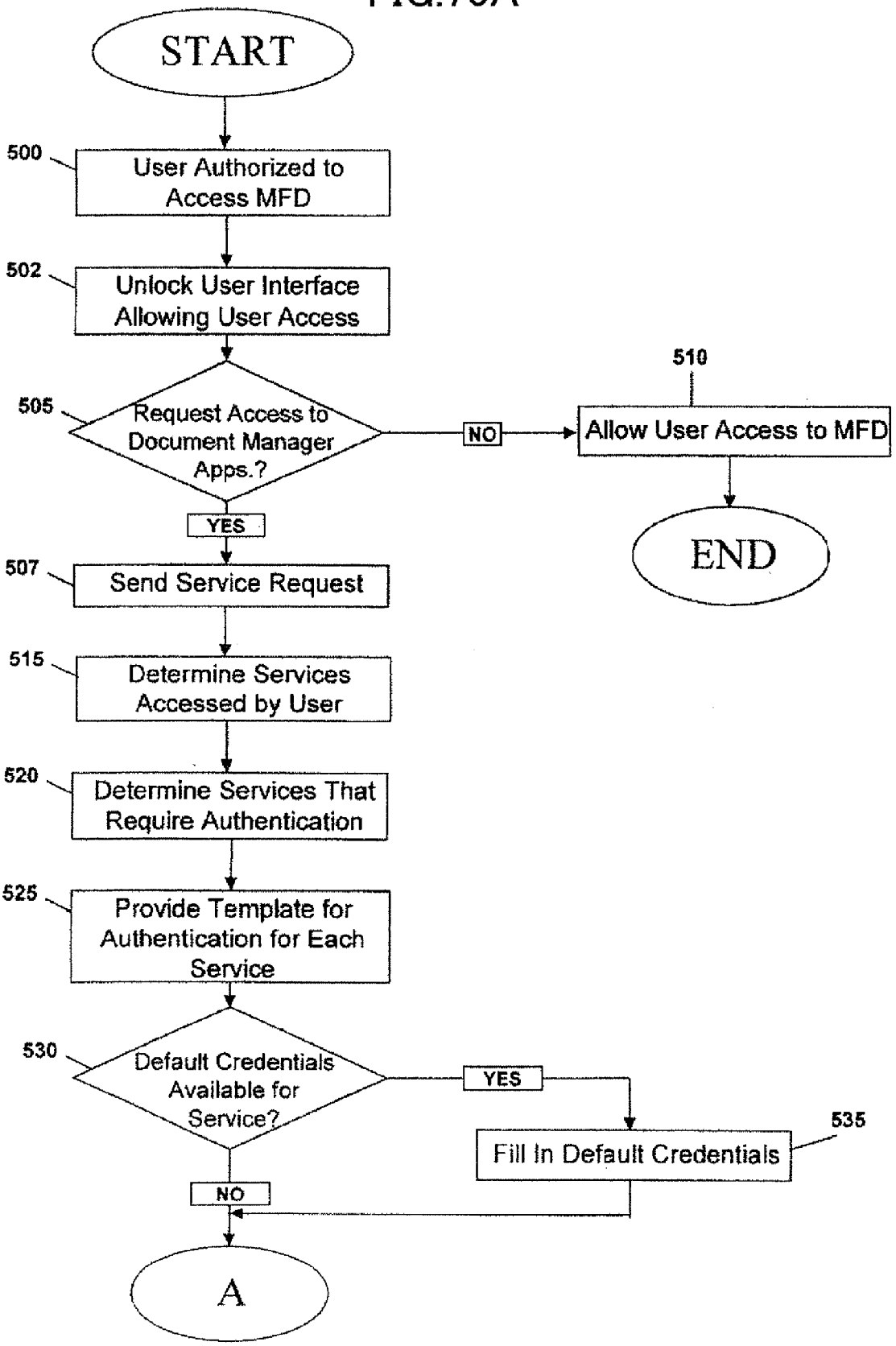

START
500
User Authorized to Access MFD
502
Unlock User Interface Allowing User Access
505
Request Access to Document Manager Apps.?
NO
510
Allow User Access to MFD
END
YES
507
Send Service Request
515
Determine Services Accessed by User
520
Determine Services That Require Authentication
525
Provide Template for Authentication for Each Service
530
Default Credentials Available for Service?
YES
535
Fill In Default Credentials
NO
A

FIG.70B

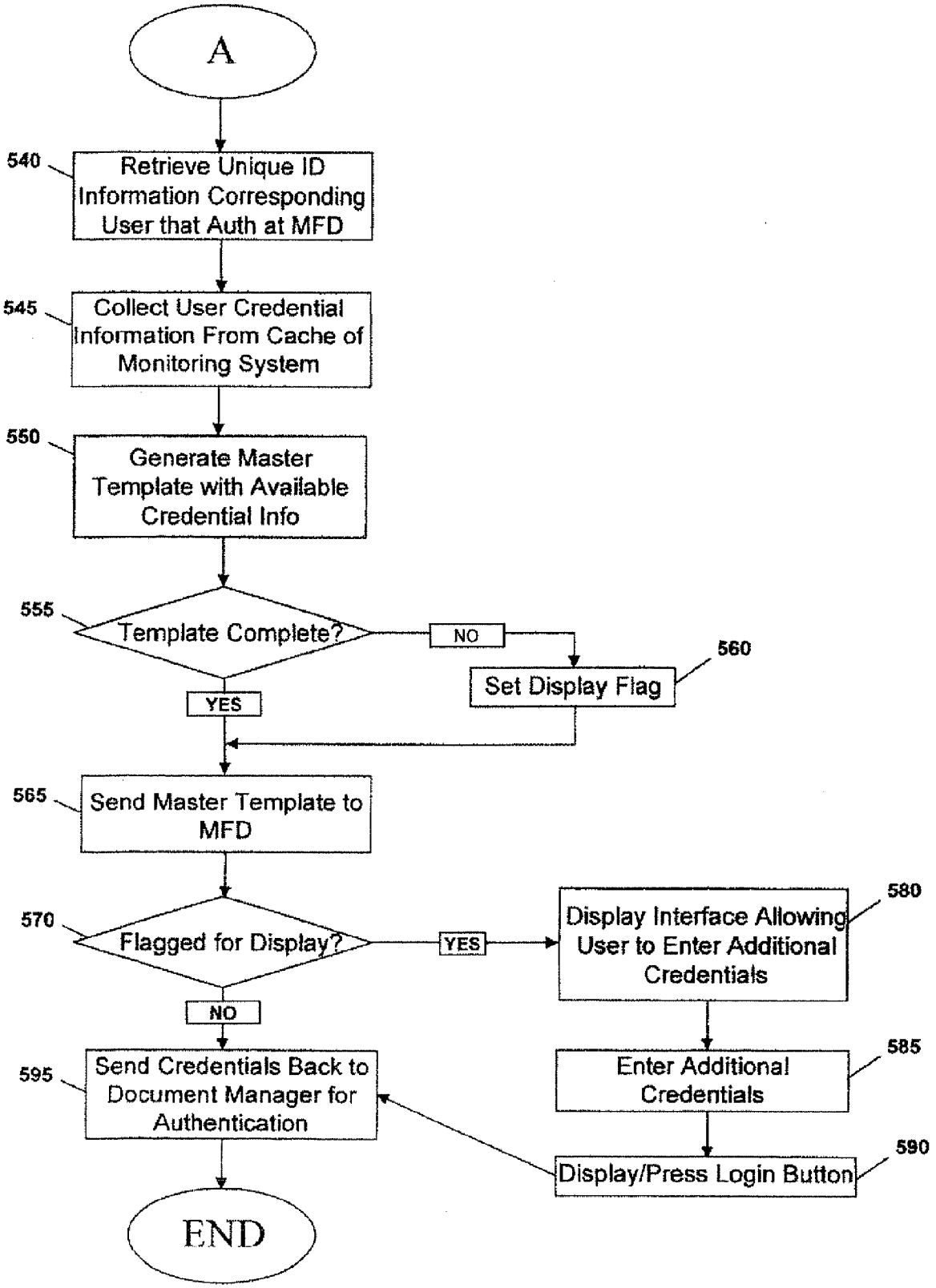

A
540
Retrieve Unique ID Information Corresponding User that Auth at MFD
545
Collect User Credential Information From Cache of Monitoring System
550
Generate Master Template with Available Credential Info
555
Template Complete?
NO
560
Set Display Flag
YES
565
Send Master Template to MFD
570
Flagged for Display?
YES
580
Display Interface Allowing User to Enter Additional Credentials
NO
585
Enter Additional Credentials
590
Display/Press Login Button
595
Send Credentials Back to Document Manager for Authentication
END

FIG.71

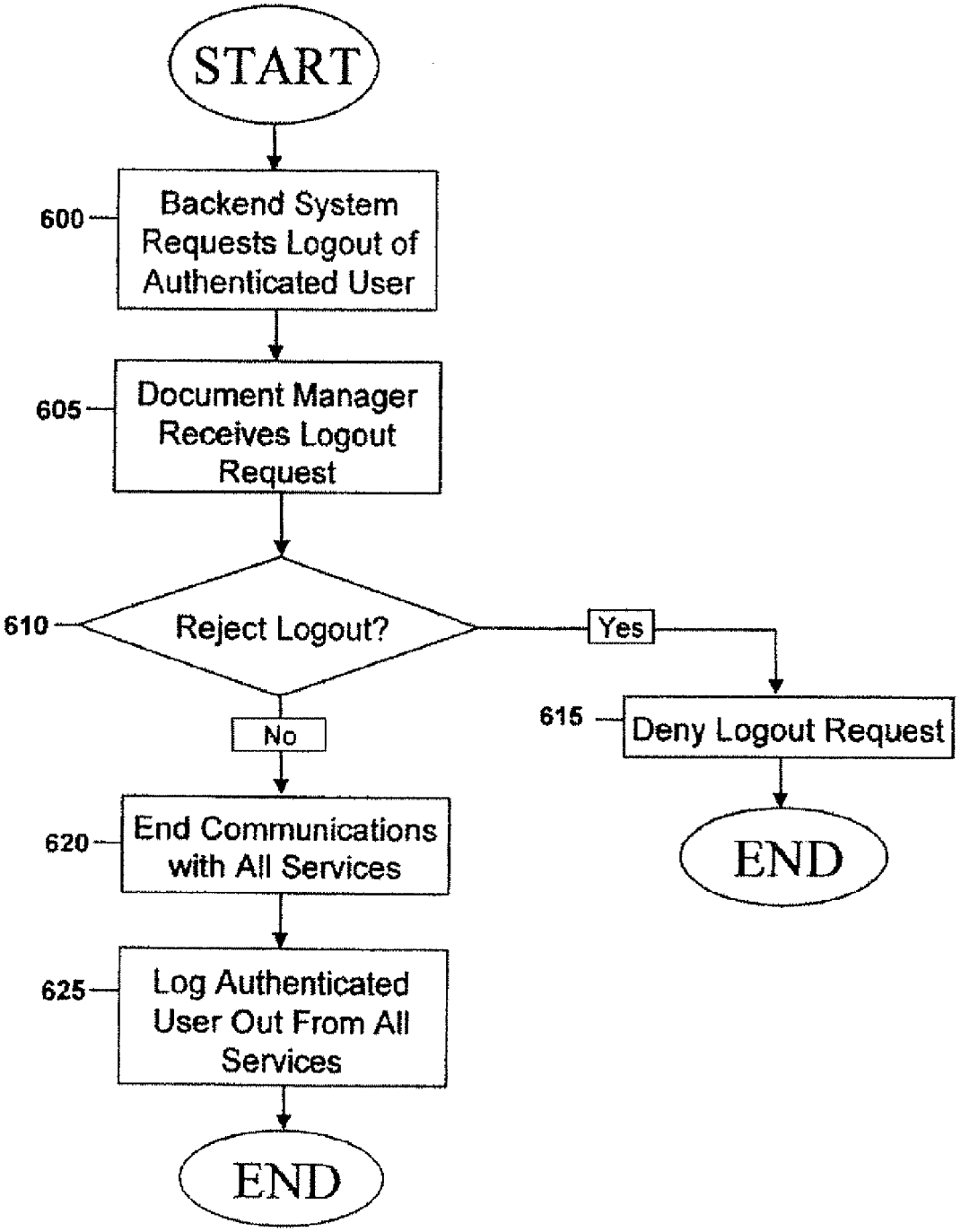

START
600
Backend System Requests Logout of Authenticated User
605
Document Manager Receives Logout Request
610
Reject Logout?
Yes
615
Deny Logout Request
END
No
620
End Communications with All Services
625
Log Authenticated User Out From All Services
END

FIG.72

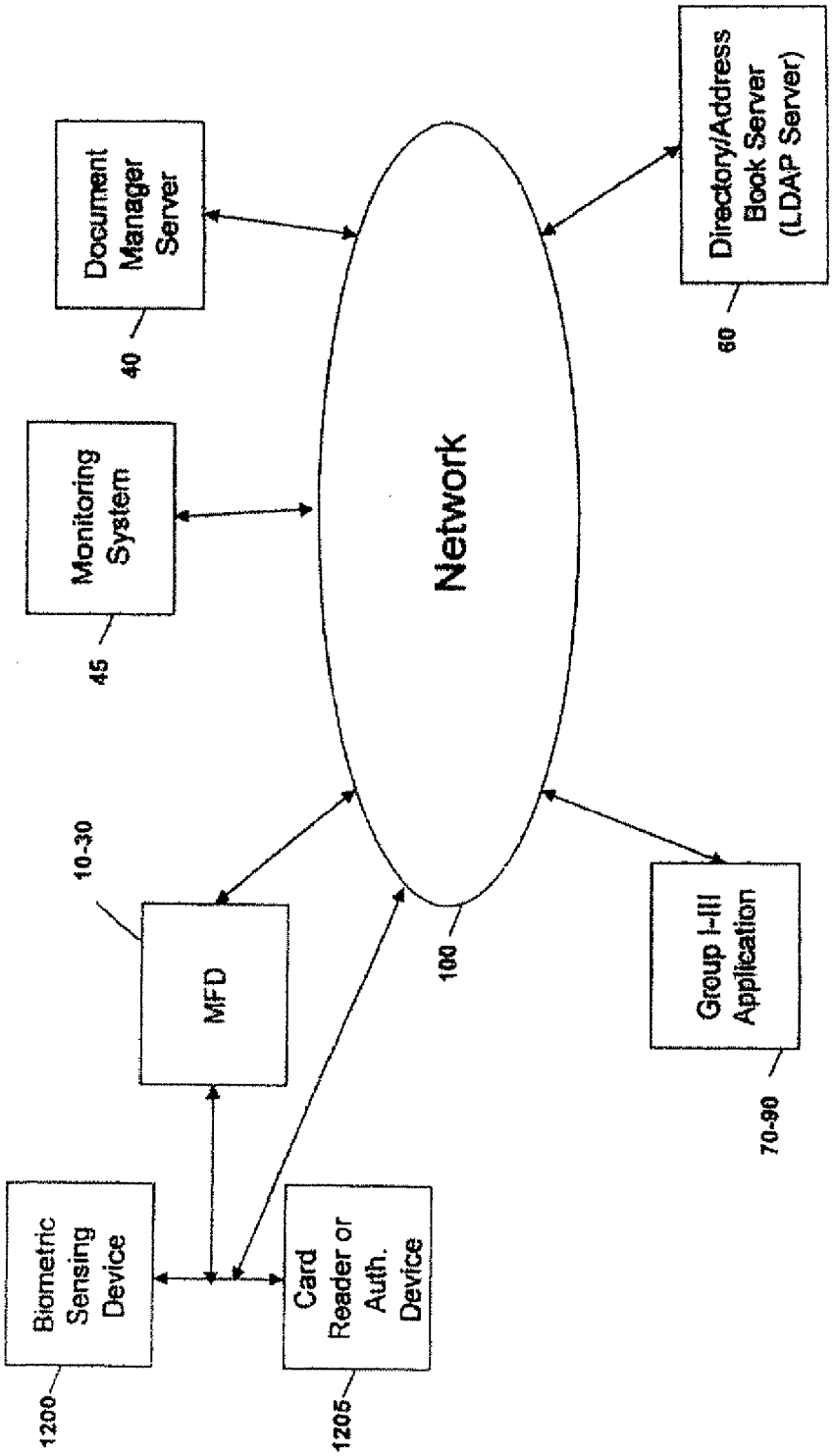

Network
100
Document Manager Server
40
Directory/Address Book Server (LDAP Server)
60
Monitoring System
45
MFD
10-30
Group I-III Application
70-90
Biometric Sensing Device
1200
Card Reader or Auth. Device
1205

FIG.73

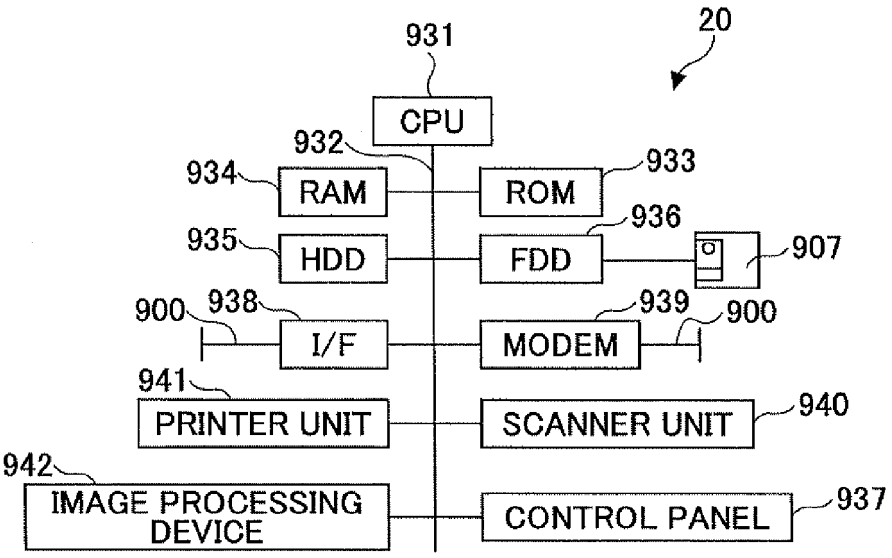

931
20
CPU
932
934
RAM
ROM
933
935
HDD
FDD
936
907
900
938
I/F
MODEM
939
900
941
PRINTER UNIT
SCANNER UNIT
940
942
IMAGE PROCESSING DEVICE
CONTROL PANEL
937

FIG.74

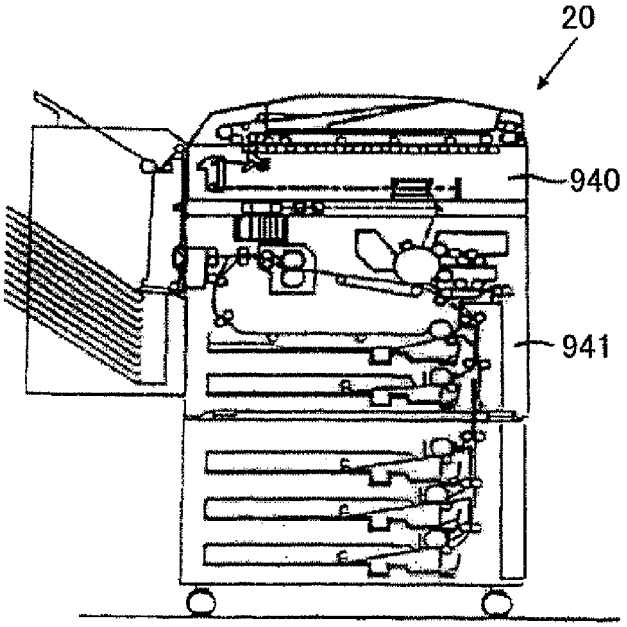

20
940
941

FIG.75

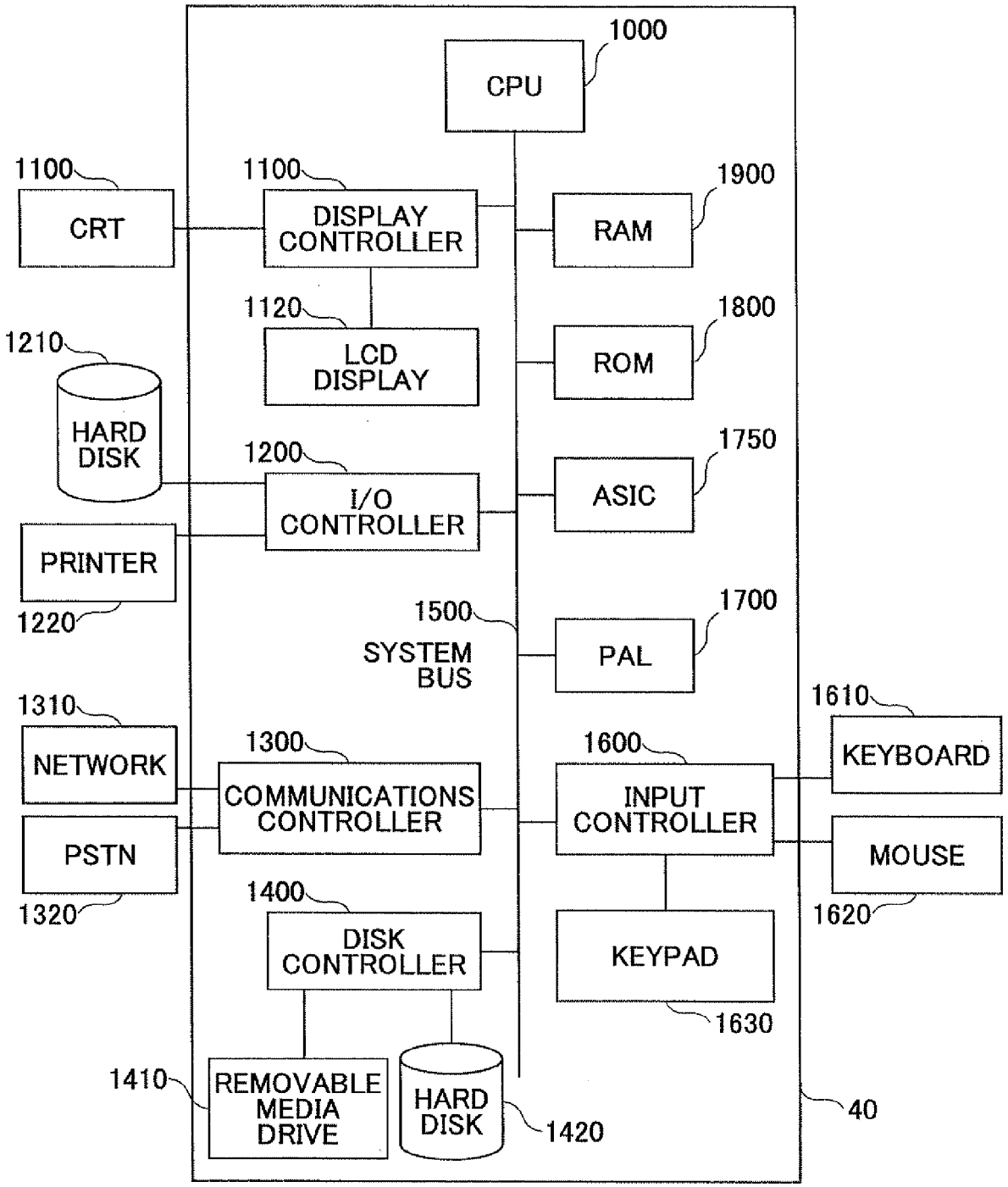
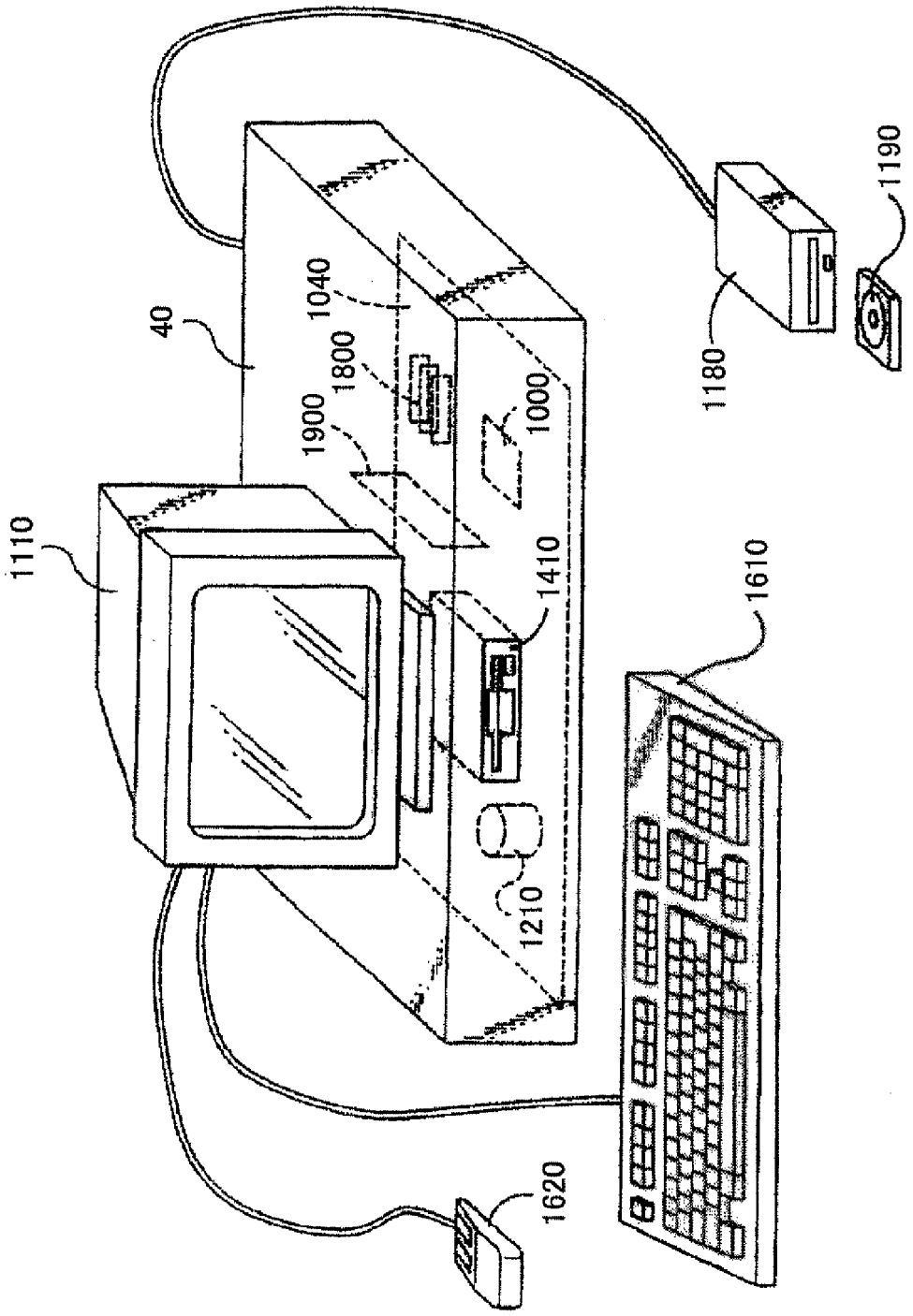

1000
CPU
1100
CRT
1100
DISPLAY CONTROLLER
1900
RAM
1120
LCD DISPLAY
1800
ROM
1210
HARD DISK
1200
I/O CONTROLLER
1750
ASIC
PRINTER
1220
1500
SYSTEM BUS
1700
PAL
1610
KEYBOARD
1310
NETWORK
1300
COMMUNICATIONS CONTROLLER
1600
INPUT CONTROLLER
PSTN
1320
MOUSE
1620
1400
DISK CONTROLLER
KEYPAD
1630
1410
REMOVABLE MEDIA DRIVE
HARD DISK
1420
40

FIG.76



European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 08 15 0752

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
X	US 2005/039126 A1 (KATANO SEIICHI [US]) 17 February 2005 (2005-02-17) * paragraph [0023] - paragraph [0028] * * figures 1-4 *	1-108	INV. G03G15/00
A	US 2004/216058 A1 (CHAVERS A GREGORY [US] ET AL) 28 October 2004 (2004-10-28) * paragraph [0030] - paragraph [0036] * * figures 1,2 *	1-108	

TECHNICAL FIELDS SEARCHED (IPC)

G03G

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
Munich	6 May 2008	Götsch, Stefan

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 08 15 0752

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

06-05-2008

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2005039126 A1	17-02-2005	US 2008010600 A1	10-01-2008
US 2004216058 A1	28-10-2004	WO 2004096564 A2	11-11-2004

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- WO 11092831 A **[0230]**
- WO 11092836 A **[0230]**
- WO 11092829 A **[0230]**
- US 24364502 A **[0335] [0386]**
- US 79543801 A **[0338] [0386]**
- US 11616202 A **[0338]**
- US 24364302 A **[0346]**
- US 09283105 A **[0370] [0386]**
- US 09283605 A **[0386]**
- US 09282905 A **[0386]**
- US 29460702 A **[0386]**