PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : H04L

A2

(11) International Publication Number: WO 00/25468

(43) International Publication Date: 4 May 2000 (04.05.00)

(21) International Application Number: PCT/IB99/01865

(22) International Filing Date: 27 October 1999 (27.10.99)

(30) Priority Data:
60/105,794 27 October 1998 (27.10.98) US

(71) Applicant: ULTRAMIND INTERNATIONAL LIMITED [IL/GB]; Saville House, 2 Lindsey Street, London EC1A 9HP (GB).

(72) Inventors: ORBACH, Tuvi; 5 Ravenscroft Avenue, London NW11 0SA (GB). DANIELI, Zeev; Hamitzpa 3/6, 90917 Giva't Zeev (IL). ALPERT, Jesse, Louis; Moshav Sde-Nitzan #5, 85470 D.N. HaNegev (IL).

(74) Agent: GALLAFENT, Richard, J.; Gallafent & Company, 9 Staple Inn, London WC1V 7QH (GB).

(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: METHOD AND APPARATUS FOR PROVIDING A DIRECT COMMUNICATION LINK BETWEEN A PLURALITY OF USERS INDEPENDENT OF A SERVER

(57) Abstract

A method and system for establishing a direct connection between at least two users having access to a registry server located on a global network. The method includes registering a first IP address associated with a proxy server with a registry server. The proxy server being locally connected to at least one first user station. The method also includes receiving a request from a second user station to communicate with the first user station and transmitting the first IP address to the second user station by the registry server. The method also includes establishing a direct communication link from the second user station to the first user station via the proxy server using the first IP address without having to connect to the registry server.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

METHOD AND APPARATUS FOR PROVIDING A DIRECT COMMUNICATION LINK BETWEEN A PLURALITY OF USERS INDEPENDENT OF A SERVER

CROSS-REFERENCE TO RELATED APPLICATION

The present application claims benefit of the filing date of provisional U.S. Patent Application No. 60/105,794 entitled "Method and Apparatus for Providing a Communication Link Between A Plurality of Users Independent of a Server", filed on October 27, 1998, which is herein incorporated by reference in its entirety.

Field Of The Invention

The present invention relates to direct communication links between a plurality of users.

BACKGROUND OF THE INVENTION

A business network that provides services to users globally on a global network typically includes one or more central servers, for example, for providing centralized access control, file access, and resource management to the plurality of users. Consequently, a user desiring to establish a communication link to another user employing the same service must direct all communications to the server. The server then relays the messages to the other user. Similarly, when a response is necessary, the other user must also direct all communications to the server, wherein the server relays the messages to the appropriate user(s) to receive the messages. Because all communications from the plurality of users must be handled and redirected by the server, a major bottleneck is frequently created at the server, degrading performance and causing failure of the server computer. As a result, the users are often prevented from communication with each other. Moreover, communicating through a server invariably creates an increase in network traffic resulting in time delay for users to communicate.

It is therefore, highly desirable to have a system for preventing an increased time delay between users and preventing the server from crashing. Accordingly, the present invention provides a direct communications link between at the two users independent of the common server once the direct communication link is established. Accordingly, the transmission of psychophysiological data in real time can be achieved.

SUMMARY OF THE INVENTION

An aspect of the present invention provides a method for establishing a direct connection between at least two users having access to a registry server located on a global network. The method includes registering a first IP address associated with a proxy server with a registry server. The proxy server being locally connected to at least one first user station. The method also includes receiving a request from a second user station to communicate with the first user station and transmitting the first IP address to the second user station by the registry server. The method also includes establishing a direct communication link from the second user station to the first user station via the proxy server using the first IP address without having to connect to the registry server.

Another aspect of the present invention provides a method for establishing a direct connection between at least two users having access to a registry server located on a global network to enable a first user to monitor a second user remotely. The method includes registering a first user station with a registry server using a proxy IP address. The first user station locally connected to a proxy server having the proxy IP address.

The method also includes initiating communications with at least one sensor for monitoring activities of a user such as physiological and psychophysiological activities at the first user station. The method also includes notifying a second user station remotely connected from the first user station to receive the proxy IP address to communicate with the first user station and establishing a direct communication link from the second user station to the first user station via the proxy IP address. The method also includes receiving data signals from the sensor as a result of monitoring the activities such as physiological and psychophysiological activities of the user at the first user station and transmitting the data signals to the second user station via the direct communication link.

Yet another aspect of the present invention provides a system for establishing a direct connection between at least two users having access to a registry server located on a global network. The system includes a registry server for registering IP addresses of at least one user and a proxy server having a first IP address connected to at least one first user station. When the proxy server receives a request from the first user station to register with the registry server, the proxy server registers the first IP address. Further, when the registry server receives a communications request from a second user station to communicate with the first user station, the registry server provides the second user station with the first IP address. Thus, the second user station is allowed to establish a direct communication link bypassing the registry server to the first user station via the proxy server.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is an exemplary embodiment of a system of the present invention for establishing a direct connection between a plurality of users.

FIG. 2 is an exemplary embodiment of a diagram illustrating establishing direct communication between a plurality of users in accordance with the present invention.

FIG. 3 is an exemplary embodiment of a system of the present invention for establishing direct communication between a health provider unit and client unit.

FIG. 4 is an exemplary embodiment of a block diagram for establishing a connection to receive data from a device independent of the server once the communication link is established in accordance with the present invention.

FIG. 5 illustrates an example of a block diagram showing the Data Server of the present invention having two devices and two client applications.

DETAILED DESCRIPTION

FIG. 1 illustrates an exemplary embodiment of a system for establishing a direct connection between a plurality of users not having well-known Internet Protocol (IP) addresses. In an exemplary embodiment, for example, users A-D 110, 120, 130, 140 may not have well-known IP addresses. Users 110, 120, 130, 140, for example may be clients or users of clients having a client program such as a user interface. The client program may run and register services on behalf of the client. Further, users 110, 120, 130, 140 may connect to the Internet, for example, using a dial up connection through an Internet provider. The Internet provider usually assigns a temporary Internet address to the user. This temporary address is usually unknown to other users, that may want to contact to another user. Further, multiple processes can run on one machine, each handling Internet connections. Thus, a user can sign on to a plurality of services under the user's name. A user can also specify other users that have access to a specific service which the user signed on. Other users not specified in an access control list, will not be provided with any information indicative of whether the specified service is available or the address of the service from, for example, a registry server such as Server H 180. In an exemplary embodiment of the present invention, the system is supplemented with password authentication.

Server H 180 may have a well-known IP address. Proxy Servers 150, 160, 170, however, need not have well-known IP addresses. For example, a proxy server may access Server H 180 through an ISP and be provided with a temporary Internet address. Proxy Servers, for example, establish and maintain connections between users of different networks and tracks local users and their local addresses. In an exemplary embodiment of the present invention, Proxy Servers 150, 160, 170 may include a program, for example, UMGate, designed to run on a proxy computer that connects a local area network (LAN) with the Internet. The LAN may be connected to the Internet via one or more stations, which can be used as a gateway (or proxy) by the other LAN stations. For example, User C 130 goes through Proxy Server F 160 and Proxy Server G 170 as shown in FIG. 1 to access Server H 180 which may be located at a web site on the Internet. Further, User C 130 may include a plurality of users on a LAN accessing the same Proxy Servers F and G. Accordingly, the other LAN users access the Internet through the proxy server and do not get an Internet address of their own.

In an exemplary embodiment of the present invention, proxy servers 150. 160, 170 are transparent, e.g., a specific communication link between two users may go through one or more proxy servers without the users being provided with information indicative that the proxy servers are involved. Regular operations including, but not limited to, listing online users, retrieving user data, and broadcasting, all work as if there were no proxy servers 150, 160, 170. Accordingly, all the messages are provided to the appropriate users whether they are connected directly to the Internet or behind one or more proxy servers.

Server H 180, for example, may include a registry server located on the Internet for registering IP addresses and tracking online services and a file transfer protocol(FTP) server for transferring files in response to requests. In accordance with the present invention, for example, a direct connection may be established between User A 110 and User B 120 independent of Server H 180 once the direct connection is established.

As shown in FIG. 1, User B, not having a well-known IP address, locally connects to Proxy Server E having an address such as a temporary IP address or a well-known IP address. Proxy Server E 150, for example, may receive a request from User B 120 to register with Server H 180. Accordingly, Proxy Server E 150 registers its IP address with Server H 180. As shown in FIG. 1, User A 110 communicates with Server H 180 without going through a proxy server. User A 110, for example, may be access Server H through an ISP and be provided with a temporary IP address. Upon Server H 180 receiving a communications request from User A 110 to communicate with User B 120, Server H 180 provides User A 110 with the well-known IP address of Proxy Server E 150 for allowing User A 110 to establish a direct communications link to User B 120 via

Proxy Server E 150 in which Server H 180 is bypassed. Accordingly, multiple users, directly or indirectly connected to a network such as the Internet may communicate with each other through a direct communication link independent of a registry server, e.g., Server H 180, once the direct communication link is established.

Further, users accessing Server H 180 without going through proxy servers can communicate with each other. Also, direct communication links may be established with more than two users, each having direct communication with each other. The direct communication being independent of the common server 180 once the direct communication is established. Accordingly, communication can be established with each other without having to continuously pass through increased network traffic at a common server.

FIG. 2 illustrates an exemplary embodiment of a method of the present invention for establishing a direct connection between at least two users enabling one of the users to monitor another one of the users. Each of the users 110, 120, 130, 140 have access to a registry server 180 located on a global network such as the Internet. As shown in FIG. 2, in 201-203 User B 120 is registered with Server H 180 by signing in 201 with Proxy Server E 150. In 202, Proxy Server E 150 transfers the sign in information from User B 120 and the IP address of Proxy Server E 150 to Server H 180. In 203, Server H 180 receives IP address information

In 204, User B 120 sets a device such as a sensor connected to or at the location of User B 120 for providing data corresponding to activity such as psychophysiological data of, for example, User B 120. Sensors may include, but are not limited to, EDA sensor, finger temperature sensor, core temperature sensor, pulse sensor, respiration Luzi sensor, blood pressure sensor, EKG sensor and weight sensor. Psychophysiological data reflects psychological behaviour and psychological state. Mind/body interactions, relaxation methods and diagnostic information with emphasis on self behaviour modification and improvement can be achieved.

In 205, Proxy Server E 150 receives the set device information and transfers the information to Server H 180 which records the device information in 206. Accordingly, communication is initiated with the device and User B 120. In 207, User B 120 provides information to Proxy Server E 150 allowing data provided by the device previously set to be viewed by User A 110. In 208, Proxy Server E 150 transfers the information provided by User B 120 to Server H 180 and Server H 180 record this information in 209. Accordingly, in an exemplary embodiment of the present invention, User B 120 can be registered in a client mode allowing, for example, the activities such as physiological and psychophysiological activities of the user as indicated by the device to be monitored by a specified user or users. In 210, Server H notifies User A 110, for example, remotely connected from User B 120, to receive the proxy IP address to establish a direct communication link between User A 110 and User B 120 independent of Server H 180 once the direct communication link is established. In 211, User A 110 signs in with Server H 180.

In 212, User B 120 sets and/or modifies user information with Proxy Server E 150 which transfers the user information to Server H 180 in 213 to be recorded in 214. In 215, User A 110 requests user data from Server H 180 which provides the user data to User A 110 in 216.

In 217, User B 120 receives the user data from Server H 180. The user data can be read whether or not the user, which originally stored the data, is currently online. User A 110 requests to monitor User B 120, for example, establish a communication link to receive data provided by the device previously set by User B 120 in 218 In response, User B 120 sets a local server and provides notification in 219. Raw data from the device is sent to User A 110 through IP address of Proxy Server E 150 in 220. In an exemplary embodiment of the present invention, in 221 the raw data may be displayed for User A 110 using a selected application. Accordingly, User A 110 may monitor the activities such as physiological and psychophysiological activities of User B 120 through a direct communication link. In an exemplary embodiment of the present invention, the information of User B 120 transmitted to Server H 180 may be retrieved by User A 110. User A 110 may, for example, analyse the information of User B 120 with the data signals received. Further, in 222a, 222b User A 110 and User B 120 may chat with each other through respective IP addresses associated with each other through the direct communication link independent of Server H 180 once the direct communication link is established.

An exemplary embodiment of the present invention as shown in FIG. 3, provides a Client/Server based environment that uses networks such as the Internet for a comprehensive connected health application. A plurality of users can be, for example, a doctor 310 and a patient 320. The doctor 310 and patient 320 can connect to the Internet through, for example, a direct connection or a LAN. The patient 320 can transfer data such as physiological data directly to the doctor 310 in real-time or place data on a server unit 330 that can be retrieved by other authorised users, for example, a doctor 310 which has access to the patient 320. The data placed on the server can be retrieved whether or not the user that placed the data is currently online. Further, a user can use the server unit 330 to broadcast a message to all users which are currently online and which have access to the broadcasting user. For example, patient data may be provided to, for example, the server unit 330 and/or other users through sensors associated with the patient 320. If any of the sensors are changed, a message such as the sensor being changed may be provided to all of the doctors 310 of the patient 320 currently online.

The health application including the attachment of several low cost, easy to use sensors that relate to a specific medical parameter as disclosed in European Patent Application EP 0 598 016 B1, filed on August 7, 1992, entitled "Apparatus for Monitoring Psychophysiological Condition," which is herein incorporated by reference in its entirety. The health application also includes registration of patients with appropriate data such as name, address, phone number, etc.; on the fly measurement and recording of medical data received from at least one of the sensors; viewing of physiological data from a session or historical perspective; sending the data received from the at least one sensor on the fly or at a later time; sending other medical data to the same server and/or user such as a health provider 310 and automatic process of notification between respective users such as a patient and a health provider. Table One identifies the communication between users such as the physicians 310 and patients 320, and the server unit 330. The server unit 330 may be a registry server 180 located at a web site that can be accessed, for example, through a dial-up connection and a continuous connection will be established with a local ISP.

TABLE ONE

Arrow Number	**Content**
301	text instructions, establish data channel
302	device(s) data, text (answers, questions, etc.), files (medical data and documents)
303	registration data, authorisation data, files for the health providers
304	list of health providers or clients for chatting, IP of a selected target for the chatting
305	registration data, query of user
306	list of authorised clients, IP of selected files of patients

In an exemplary embodiment of the present invention, the client unit 320 includes the following capabilities: registration, accepting and displaying other users such as health providers 310, sensor attachment and accepting physiological data therefrom by using, for example, a graphing application such as UMGraph; sending informative data and medical data; setting sensors; sending and receiving instructions and messages via chatting channel and view recorded data by using, for example, a viewing application such as UMView.

A user is identified through registration and establishes a password to log in to the server. Further, the first time a user registers additional information may be requested such as telephone number, address, birthday and names of health providers. For existing users, only a password may be required to enable *sign-in.* Existing users may modify their data previously provided by again using the registration process as shown in 212 of FIG. 2.

Further, the client unit 320 may receive the status of providers registered thereto. For example, an initial list may be provided to the client unit 320 and a modification notification may be provided each time the initial list changes. The patient 320 may be connected to a sensor and set the type of sensor. Graphs may be displayed by a graph application as shown in 221 of FIG. 2 and the raw data may be recorded for future assessments. Further, the user may send data files to the server addressing it to another user such as a health provider 310. If the health provider 310 is signed-in, it will receive an automatic notification about the sent files. The client unit 320 may send recorded medical data to the server unit 330 and health providers 310 of the client unit 320 and download files provided by the respective health providers 310. The recorded medical data may be, for example, a set of previously stored sessions, a file of an integrated external application and a specific session from previously recorded sessions for Internet data viewing. The client unit 320 defines the sensor, checks the connection and informs others as shown in 204 of FIG. 2. The client unit 320 receives and sends text data to a health provider 310 that is currently signed in and associated with the client unit 320. Further, health providers 310 may send messages to other health providers 310 or their patients 320. The patient 320 will be able to view the respective data that has already been recorded. Data that passed through the graph application may automatically be recorded.

The server unit 330 may be, for example, an Internet site system written in Java including the following services: accept data of users, accept registration requests, sign-in and out, accept and send information regarding sensors, connect users together for interactive operation, store data files of patients 320, send data files of patients 320 to their respective health providers 310 and accept user data. The server unit 330 receives registration data from the various users and records it. The registration data may include, for example user code name, first name, last name, birthday, health provider, name, telephone number and password. The server unit 330 accepts and stores registration in a database. The user IP will be kept for future connections. The sign-in/out process informs the server unit 330 of whether a user is presently connected. The server unit 330 manages a connected-user table that includes communication data such as a user name and associated IP address. The server unit 330 will send respective notification messages to users connected to the network. For example, a client unit 320 is notified when health providers 310 of the client unit 320 are signed in and the health provider 310 is notified that its patients 320 are signed in. Further, at the end of a sign-in process, the user signing in gets the initial data including the status of the associations of users. The server unit 330 provides the health provider 310 with the sensor type previously provided by the client unit 320 prior to the health provider 310 connecting to the client unit 320. Interactive operation between users is accomplished by at least one of the users getting the IP of the other user from the server unit 330. The server unit 330 stores and sends data files of patients 320 to the respective health providers 310.

The health provider unit 310 may include the following capabilities: registration; open data channel with a patient 320 to receive data on the fly; download patient data from the server unit 330; send and receive instructions and messages via chatting channel; receive sensor type; view recorded data; view external data files using, for example, a viewing application and send files such as reports or analysis of the patient data to patients 320. The health provider unit 310 may open a data channel with a patient 320 by selecting the patient 320 from a list of patients presently signed in. When a data channel is established the relevant client units 320 receive an automatic notification from the health provider 310 via the chatting mechanism. The health provider unit 310 may ask to download patient data form, for example, the Internet. The health provider unit 310, however, may be allowed to do so from its patients only. By using the chatting channel, the health provider unit 310 can direct the patient to take specific actions or obtain current information. For example, instruct the patient 320 to relax or to turn on a sensor. The health provider unit 310 will be able to view the data of patients 320 that has been sent. The health provider unit 310 selects the relevant patient 320 out of the list and gets a list of recorded sessions attached to the chosen patient. A sessions list may be provided and sorted by sensor type, session date or duration.

In an exemplary embodiment of the present invention, data from each of the respective sessions is stored through, for example, a header file including patient’s user name and sensor code to recognise patient and sensor type; management of several raw data files, a manual procedure for import from and export to external files and a session data table.

In an exemplary embodiment, the system of the present invention may include a viewing application, a graphing application and a displaying application. The viewing application, for example, may include enabling the recording and assessment mechanism to support several sensors and devices together with multiple patient management includes using header file to recognise patient and sensor type, selecting several raw data files, select patients and sensors, and viewing weight and EKG data of a patient. The graphing application may include selecting a device such as a sensor, saving raw data and scaling the graph at least based on the type of sensor. Each graph application window may be used with one sensor. For simultaneous display of several sensors, the user may run several windows, each set to one sensor. Further, the graph application saves the data including headers, events and raw data of non-EDA RP sensor. In an exemplary embodiment, the client data is saved only at the client unit 320. The health provider 310 may view that data after it has been sent from the client unit 320 and received by the health provider 310.

In an exemplary embodiment of the present invention, the display application may include displaying the core temperature measured from a sensor such as a forehead sensor over a channel. The core temperature measurement includes a sensor calibration stage and a measurement stage. The display application includes a setting function for setting and calibrating a sensor, measurement and result presentation, and saving results. In an exemplary embodiment, saving results includes indicating in a data file a beginning event, an ending event and a code event in which the final temperature will be stored. Further, respective notification messages may be sent to the server unit 330.

In an exemplary embodiment of the present invention, server unit 330 such as a Data Server is responsible for acquiring data from various devices such as sensors attached via serial ports and sending the data to client applications. Data Server provides communications to the devices and optional saving of device data. In an exemplary embodiment of the present invention, Data Server includes two components–DataServer.exe and DataServerDLL.dll. DataServer.exe is responsible for acquisition and buffering of the device data, while DataServerDLL.dll provides functional interface to the client. The client application loads the DataServerDLL into its (client) address space and directly calls functions for managing of the device. The calls to the function are made through a virtual function table and compliant with COM interface. Client notification about new data arriving from the device may include setting of a callback function or custom Windows Message notification. The callback function may be called from contexts of a thread, providing more uniform processing of data by client unit 320. Typically, a second thread is created in client process by DataServerDLL.

Figure 4 is a flow diagram 400 illustrating the operations of the Data Server of the present invention in one exemplary embodiment. At 402, connection to the DataServer.exe is made. The DataServerDLL checks if DataServer.exe is already running and if not, executes it. Then the Local Procedure Call (LPC) connection is established with DataServer.exe. The DataServerDLL also can communicate through Remote Procedure Call (RPC) with a remote computer, thus allowing retrieving of the data from the device connected to remote computer.

Communication with local or remote computer is transparent to client application. For example, the client only needs to provide the name of the remote computer. In an exemplary embodiment of the present invention, Data Server allows connection of multiple clients to any one of the devices. Each client is given its own buffering of data, and therefore, one client application runs independent of another.

At 404, device is opened for communication. The client typically specifies the name of the port the device is connected to, for example COM1. If the client doesn't specify the port, providing NULL pointer instead, the Data Server automatically looks for connected devices and opens the first one found. Typically, the four first COM ports are checked.

At 406, channel for data acquisition is opened. After a device is opened at 404, the client application creates the channel object. Using the channel object, the client may set the callback function or handle of window for notifications. The callback function is generally called each time the specified amount of the data is placed in the internal buffer in DataServer.exe. In an exemplary embodiment of the present invention, the Data Server may separate data and provide data in multiple channels separately.

At 408, data from the device is receive at the client. During data acquisition, the client may temporary disable and re-enable data acquisition on a specified channel. For example, the client may temporary disable the data acquisition on open channel by calling "EnableChannel" function with TRUE or FALSE. In this case, the client does not obtain notification and data is not transferred through RPC.

Moreover, the data may be saved locally by Data Server to file. The client may enable or disable the data save during the session. The data may be also transferred through a local network.

In an exemplary embodiment, the functionality of Data Server is provided through COM interface. For accessing of Data Server the application obtains a header file with declaration of interface and function for creating of first interface pointer, this function, for example, is called "DataServerCreate". This function creates the instance of CDataServer class, casts its pointer to IDataServer interface and returns it. Through this pointer the client application can open devices and create communication channels for data acquisition.

Briefly, as well known to those skilled in the art, the interface is an abstract base class with declarations of pure virtual functions. Using the interface pointer, the client can call these virtual functions. The server internally creates instance of the class derived from interface. This class includes implementation of functions declared in interface. The functions declared in interface are virtual functions, thus the call to interface functions is made through pointer in virtual function table. If the class that implements the interface is provided in DLL, the application doesn't need to implicitly import interface function from DLL. If the DLL implementing the interface is changed. the existing client application can work with new DLL.

Examples of COM interfaces of Data Server of the present invention are described hereinbelow.

<u>IDataServer - interface to Data Server Object</u>

interface IDataServer: public IUnknown

```
{virtual HRESULT
OpenDevice(LPCSTR1pszPortName,intdeviceID.IDevice*lplp
IDevice)=0
};
```

The client uses this interface to open available devices, for example, at 404. Typically, it opens the device on specified port and check if the device has specified ID. If the lpszPortName port name is NULL, then the Data Server looks for connected devices and opens the first device found. If there are no devices connected, it returns an error. Optionally, this interface may include emulation of devices and testing of available ports. For example, if instead of the port name the first parameter obtains name of the file, the device is emulated by the data from the file.

OpenDevice function first checks if the current port is already opened with some device, if not, it opens it. Then it checks that the valid data arrive from the device. If it obtains valid UM batch, then the interface pointer is returned to the client application.

The Release member function of IDataServer interface closes connection of client process to DS process and closes all communication channels and open devices created by client process. If the last client called to release, the DS process is unloaded from memory.

The pointer to IDataServer interface is typically obtained through a function that initiates the Data Server, for example, DataServerCreate function. DataServerCreate function is executed, for example, at 402, when connecting to the Data Server of the present invention:

```
HRESULT DataServerCreate (LPCSTR 1pszCompupterName,
IDataServer **1plpDataServer)
```

The first parameter is the name of the computer with Data Server Process. NULL pointer in this parameter indicates a local computer. Then this function first checks if DS Process is already running. If it cannot find already running DS process, then creates new and binds to it. Then DataServerCreate creates instance of DataServer object and return pointer to it. If this function is called the second time, it returns the same pointer and increases the lock count. The DataServerCreate has a static linkage to client applications 320. It typically calls CoCreateInstance API with class ID of IDataServer interface.

IDevice - interface to device object

In an exemplary embodiment of the present invention, a device may have a few data channels, one channel or none. The device is typically described by its internal ID and name of port to which it is connected. The data channel object controls flow from specific sensor to a client application. The interface for device is as follow:

```
interface IDevice : public IUnknown
{//-------Opens data channel with specified ID.  For
single channel devices, channelID should be zero.
virtual HRESULT OpenDataChannel(int channelID,
IDataChannel **lplpChannel) = 0; //-------Enable or
disable saving of device data to file.
virtual HRESULT EnableFileSave (BOOL IsEnable) = 0;
//-------Query device for it ID and return it.
Virtual HRESULT GetDeviceID(int *lpDeviceID);
//-------Fills the buffer with name of the port the
device connected to virtual HRESULT GetPortName (LPSTR
IpBufrer, int bufLen) = 0;
};
```

In an exemplary embodiment of the present invention, OpenDataChannel creates new data channel object and returns interface pointer to it. Through this pointer a client application may retrieve data from a sensor and get notification when data is ready. Few different processes may be attached to the same data channel in which case the process typically receive the same data concurrently. This call is used, for example, at 406, for opening data acquisition channels.

EnableFileSave enables or disables writing of data to file, for example, if the function is called with TRUE to enable and FALSE to disable. Initially the device interface pointer may be created with saving to file disabled.

Upon a request from a client application, the data received from the sensor may be saved into a data file. Typically, the data is saved in local directory with a predetermined or pre-specified file name.

The Release function of IDevice interface deletes the device object in current process and closes all its data channels.

GetDeviceID query the device (through serial port) for its ID and returns it.

GetPortName returns the port name to which the device is connected.

<u>IDataChannel - retrieving of data from device</u>

After Data Server object, i.e., pointer to IDataServer, is created and a device is opened, a communication channel to the device is created. With IDataChannel pointer an application may set notification callback function or a handle of a window to start receiving data as described with reference to 408. The interface is as follow:

```
interface IDataChannel: public IUnknown
```

```
{ virtual HRESULT EnableChannel() = 0;
virtual HRESULT DisableChannel() = 0;
virtual HRESULT SetNotificationHWND(HWND hWnd, int
numberOfSamples) = 0;
virtual HRESULT GetNotificationHWND(HWN *lphWnd,int
*lpNumberOfsamples) =0;
virtual HRESULT
SetNotificationCalllback(LPFN_ON_DATA_ACQUIRED lpProcFunc,
void *lpUserData, int numberOfSamples) = 0;
virtual RESULT
GetNotificationCallback(LPFN_ON_DATA_ACQUIRED *lpProcFunc,
void **lplpUserData, int * lpNumberOfSamples) = 0;
virtual HRESULT GetDeviceObject(IDevice **lplpDevice);
};
```

where LPFN_ON_DATA_ACQUIRED defined as follow:

```
typedef void (*LPFN_ON_DATA_ACQUIRED)(void *lpUserData,
int numberOfSamples, MAILBOX *lpAcquredData);
```

A ring data buffer is created for each new data channel. If the is already opened by other process, the client is attached to an existing ring buffer. The SetNotificationHWND function obtains the handle of window for notifications. The second parameter indicates the number of samples stored in a buffer before notification is sent to a client application.

In an exemplary embodiment, the notifications are sent to application in a form of WM_LJN1D_DATA message. The WPARAM includes the number of data items acquired, while LPARAM points to the data.

The notification callback function works exactly in the same way as message response function for WM_UMD_DATA message. It also obtains number of samples and pointer to sampled data. It also gets the pointer to user data, the same pointer as was provided in

SetNotificationCallback. The main difference between processing of WM_UMD_DATA message and use of callback function is that the last one is called from contents of additional thread, thus the data can be processed concurrently with updating of the screen or other user operations. DisableChannel stops data acquisition on channel or sending of data ready notifications for this channel.

Implementation of the Data Server of the present invention will now be described in greater detail. In an exemplary embodiment, the Data Server include two components. The first component includes the process that implements the data acquisition and buffering and is referred to, for example, as a Data Server Process ("DS Process"). Functions such as the opening of the serial port, reading, validation and buffering of data is typically provided in this process.

The second component, the DLL includes an implementation of interface functions. In an exemplary embodiment of the present invention, the DLL is directly loaded into address space of client process and provides the communication with DS Process through Local Procedure Call (LPC) or Remote Procedure Call (RPC). Generally, the code for LPC and RPC is the same except for the difference in transport protocol name when registering and binding to a server. If the performance of LPC is not satisfactory, for example, the call time may take more than 1 to 1.5 milliseconds. The transferring of data from DS Process to application may be done through memory mapped file. The advantage of LPC is that the same server can work on local as well as on remote computer.

Transfer of raw data from server to client application is based on standard producer-consumer model.

In this model, the producer thread adds the data to the buffer while the consumer thread is blocked until in the buffer contains enough data for single consumer notification. Each time the producer places new data in the buffer, it checks whether some clients have enough data for notification. Each client thread is blocked on event, waiting for new data. Each time there is enough data, the event is pulsed by provider thread for the client to get its chance to read its data from the buffer.

Briefly, the circular buffer in the present invention includes a buffer with data, one write pointer and dynamic array of read pointers for each client. For each client, the following data structure is maintained: (1) current "read" pointer; (2) event object signaling that enough data for notification of specific client 320 is placed in the buffer; and (3) minimum number of data items for notification.

Figure 5 illustrates an example of a block diagram 500 showing the Data Server of the present invention having two devices and two client applications. The DS process 516 opens one provider thread 518, 520 for each open serial port or device 502, 504. This provider thread 518, 520 is responsible for reading data from a serial port, multiplexing the data according to channels and placing the data to a corresponding channel. Also, it pulses or signals an event for the clients that have enough data for notification. Each client 512, 514 has the consumer thread 522, 524, 526, 528 for each opened channel 506, 508, 510. These consumer threads are typically created by DS DLL attached to the client process. The consumer thread 522, 524, 526, 528 is blocked on event, waiting for new data. The producer thread 518, 520 pulses or signals the event, if there is enough data in the circular buffer. The consumer thread 522, 524 reads the data from the buffer and calls the callback in application with pointer to acquired data or sends a message to the application window.

In Figure 5, two devices 502, 504 have three channels 506, 508, 510, and two client applications 512, 514 retrieve the data from the channels 506, 508, 510. In this example, each client application 512, 514 retrieves data from two channels 506, 508 and 508, 510.

For each serial port the DS process creates separate thread that acquires data from the port. This thread is blocked waiting while data arrives on serial port. The data is analyzed, validated, sorted according to channel and stored in ring buffer. Thus data is stored continuously for each channel in its own ring buffer. Creating of separate thread for each serial device enables independence of data acquisition on separate devices.

The Data Server provides data from UM sensor to client applications. It acquire and buffers data from serial port. This data is polled by client applications. In an exemplary embodiment, data acquisition is based on producer-consumer scheme. The producer thread reads the data from serial port, checks its validity and places data into the buffers for each channel. The producer thread waits on the channel for new data from the serial port.

The producer thread is implemented in class "CDevice". In an exemplary embodiment, for each new device connected to serial port new instance of "CDevice" is created and new thread is run. The "CDevice" class is based on CWinThread class and "IDevice" interface; IDevice is based in IUnknown.

RunDevice function of the CDevice,

BOOL CDevice::RunDevice(LPCSTR lpszPortName, int BaudRate, int deviceID),
obtains the port name, BaudRate and device signature. This function tries to open specified port and checks whether it gets the valid data from port. It returns TRUE only if the open of port successful and valid UM batch is received. Inside of this function the new thread for data acquisition is created.

The CDevice is based in IDevice - thus it implements the IDevice interface and supports reference count. The instance of this class is deleted through call to Release function that decreases the reference count, breaks the acquisition loop if necessary and deletes the CDevice object.

CDevice includes the instance of "CUMDSerPort" class, based on CSerial.Port. Serial port in DataServer is opened for overlapped I/O. This class is used for encapsulation of data and function for serial port. It includes the function, OpenDevCommun, that obtains the name of the port and baud rate. It opens the port in overlapped I/O mode. The data from port is read through the Receive function.

int CSerialPort: :Receive (void* IpBuf, DWORD nBufLen, DWORD timeout, DWORD retryPeriod)

Receive function reads the data from port synchronously, even in the case the port is open for overlapped I/O. It obtains the buffer, number of bytes to read, time out and retry check period. The function returns either a time out or a specified number of bytes received on serial port. It returns a number of bytes actually read from the serial port. The port is automatically closed by constructor of CSerialPort.

Advantageously, because CDevice class is derived from CWinThread it has three key overridables -

InitInstance.Run and ExitInstance. The Run function includes the main loop with acquisition. The loop is broken only when the reference count of the device reaches zero. In the loop, data is read from serial port, the validity of the data checked and the data put to channel(s) buffers, one buffer for each channel. Also in the loop, data saving to a file may be performed.

In an exemplary embodiment, the DataServer can automatically find a sensor device on one of the serial port and open it. This action is supplied through global function,

```
HRESULT OpenDevice (/* [string][in]*/ _LPCSTR
lpszPortName, int deviceID, /* [out]* IDEV_ PTR_ RPC_FAR
*lplpIDevice)
```

This function obtains the name of serial port (if the string is "NULL", four first ports are searched), device signature (currently not checked) and address of variable with result. It returns S_OK if the device was found and run successfully.

The acquired data from UMDevice is kept in the buffer until the client applications retrieve it. The data is kept in instances of CChannel class. Each instance of this class keeps buffer with corresponding data. This buffer is updated by producer thread, run by CDevice::Run function.

Instance of CDevice class keeps dynamic array of pointers to instances of CChannel class. The loop inside of CDevice::Run dispatches data between the buffer in instances of different channels.

Each instance of CChannel class keeps the buffer for the data and list of clients retrieving the data from this channel.

Constructor of CChannel obtains the integer ChanneID and the size of array with buffer.

```
CChannel::CChannel (int channelID, int samples_size);
```

All the data is kept in the array of mailboxes, declared according Moshe specifications:

```
CArray <MAILBOX, MAILBOX& > m_SamplesBuffer;
```

In the constructor the proper size of array with mail boxes is allocated. Each instance of CChannel keeps the information about clients connected to specific channels. Information about clients is kept in the CClientInfo structure, the CChannel keeps the dynamic array of pointers to those structures. Each time new client is connected, the new instance of CClientInfo is created, the client application obtains its index in the array. When the client application tries to get the data it calls the ReadSamples function:

```
HRESULT CChannel::ReadSamples (int hClient, int samples,
MAILBOX *pMailBoxes, int *pFilledBoxes )
```

ReadSamples function obtains the index of the client in the array of CClientInfo structures, number of samples the client intends to get, pointer to buffer (pMailBoxes), and address of the integer to be filled with actual number of data read. It waits on synchronization object, so the CPU time is not wasted. Depending on the number of requested samples, it waits and then obtains the batch with a few samples in the buffer pointed by pMailBoxes.

The CChannel class keeps the current "write" position, while each client has the "read" position in CClientInfo structure.:

```
struct CClientInfo
{public:
//-------Current read position of client
  UINT    m_ReadPos;
//-------Minimum number of samples for notification
  UINT    m_MinSampReady;
//-------Number of lost samples
  UTNT m_lostSamples,
//-------Event that there is enough data
  CEvent m_DataReadyEvn;
//-------If TRUE, the client is given only the last data
  BOOL m_ AllowDataLoss;
//-------If FALSE, channel disabled and not accumulating
data
  BOOLm_ IsEnabled;
};
```

In general, the client keeps the index of its CClientInfo structure and this structure keeps information about the client for DataServer.

Each time the new client tries to connect to the specific channel the function, HRESULT CChannel::AddNewClient (int *hClient) is called. This function return the index in new CClientInfo through the hClient parameter, this index is used later in ReadSamples function.

The DataServer application keeps the lock count of the client connections as a global variable _ - m_ NumberOfClients - public data member of the CDataServerApp class based on CWinApp. Two global functions used for managing of reference count include ULONG AddRef() and ULONG Release(). When Release function is called it decrements a reference count. When the reference count reaches zero, the function posts WM_COMMAND with IDOK to the dialog, which operation closes the DataServer application.

In the InitInstance of the CDataServerApp, the DataServer is registered as RPC server through the function, RegisterDataServer(). This function reads the name of the protocol from umd.ini file. The function

UnregisterDataServer() called during shutdown of DataServer closes the RPC service.

While the invention has been particularly shown and described with respect to an embodiment thereof, it will be understood by those skilled in the art that the foregoing and other changes in form and details may be made therein without departing from the spirit and scope of the invention.

What is claimed is:

1. A method for establishing a direct connection between at least two users having access to a registry server located on a global network, the method comprising:

registering a first IP address associated with a proxy server with a registry server, the proxy server locally connected to at least one first user station;

receiving by the registry server, a request from a second user station to communicate with the first user station;

transmitting by the registry server, the first IP address to the second user station; and

establishing a direct communication link from the second user station to the first user station via the proxy server using the first IP address without having to connect to the registry server.

2. A method for establishing a direct connection between at least two users having access to a registry server located on a global network to enable a first user to monitor a second user remotely, the method comprising:

registering a first user station with a registry server using a proxy IP address, the first user station locally connected to a proxy server having the proxy IP address;

initiating communications with at least one sensor and the first user station, the sensor for monitoring activities of a user at the first user station;

notifying a second user station remotely connected from the first user station to receive the proxy IP address to communicate with the first user station;

establishing a direct communication link from the second user station to the first user station via the proxy IP address;

receiving data signals from the sensor as a result of monitoring the activities of the user at the first user station;

transmitting the data signals to the second user station via the direct communication link.

3. The method of claim 2, further including:

establishing a chatting session between the first user station and the second user stations via the direct communication link.

4. A method of claim 3, further including:

transmitting user information to the registry server, wherein the second user station retrieves the user information for analysis with the data signals received.

5. The method of claim 3, wherein the registry server is located on the Internet.

6. A system for establishing a direct connection between at least two users having access to a registry server located on a global network, the system comprising:

a registry server for registering IP addresses of at least one user;

a proxy server connected to at least one first user station, the proxy server having a first IP address;

wherein when the proxy server receives a request from the first user station to register with the registry server, the proxy server registers the first IP address, and when the registry server receives a communications request from a second user station to communicate with the first user station, the registry server provides the second user station with the first IP address for allowing the second user station to establish a direct communications link to the first user station via the proxy server, bypassing the registry server.

7. The system of claim 6, further including:

a client interface associated with the first user station for allowing a user to register with the registry server at the first user station.

8. The system of claim 7, further including:

at least one sensor connected to the first user station for monitoring and transforming a psychophysiological condition of the user at the first user station into data signals for transmission.

9. The system of claim 8, wherein the data signals representing the psychophysiological condition of the user are transmitted to the second user station.

10. The system of claim 9, wherein the data signals represent at least one of electrodermal activity (EDA), peripheral temperature, heart rate (R to R interval), heart rate variability (HRV) and breathing measurements.

11. The system of claim 8, wherein the data signals are interpreted to produce screen animation presented to the user.

12. The system of claim 11, wherein the animation is imparted by changes in the psychophysiological condition of the user.

13. The system of claim 12, wherein the user receives a stimuli at the first user station if the psychophysiological condition of the user is different from a predetermined condition.

14. The system of claim 13, wherein the stimuli are consciously-viewable images.

15. The system of claim 13, wherein the stimuli are unconsciously-viewable subliminal images.

16. The system of claim 13, wherein the stimuli are audible sounds.

17. The system of claim 13, wherein the stimuli include instructions to the user at the first user station, wherein responses of the user to the instructions can be monitored at the second user station.

18. The system of claim 9, wherein the data signals can be monitored at the second user station as at least one of images, graphs, numbers, audio and video sequences.

19. The system of claim 9, wherein the data signals are transmitted to a plurality of stations at different locations.

20. The system of claim 9, further including a plurality of the proxy servers having at least one first user station.

21. The system of claim 20, wherein at least one first user station transmits the data signals simultaneously.

22. The system of claim 17, wherein the instructions, the responses, and the data signals representing the psychophysiological condition of the user are transmitted over one communications link contemporaneously.

FIG. 1

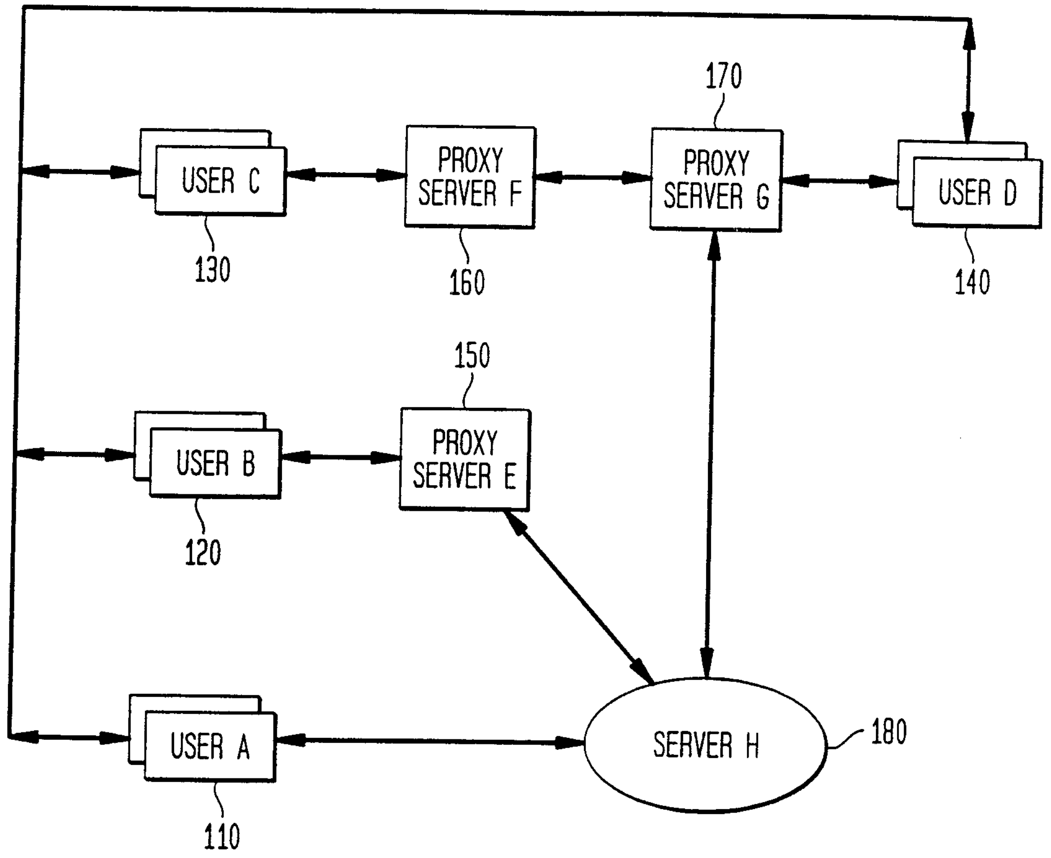

170
USER C
PROXY SERVER F
PROXY SERVER G
USER D
130
160
140
150
USER B
PROXY SERVER E
120
USER A
SERVER H
180
110

FIG. 2

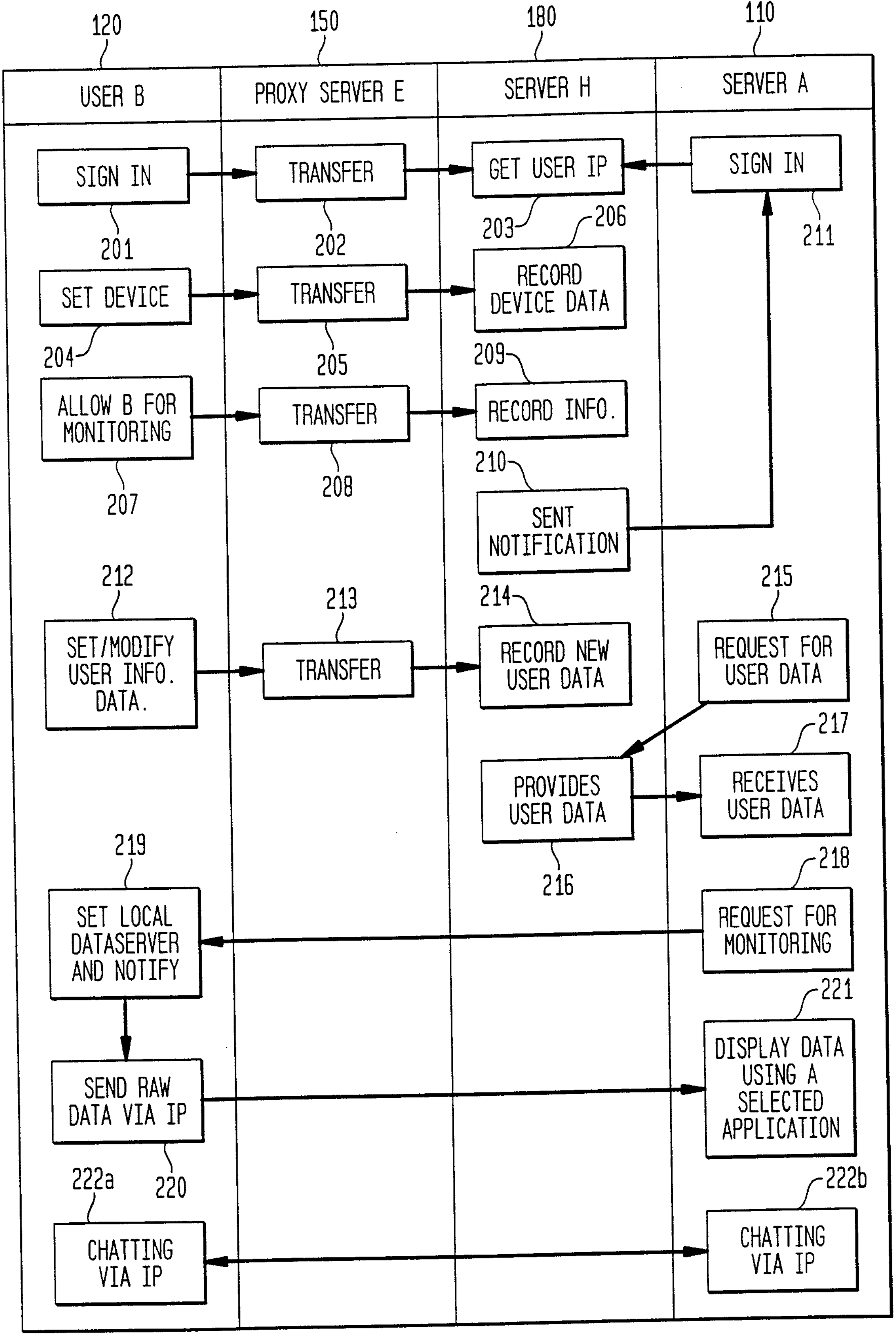

120
150
180
110
USER B
PROXY SERVER E
SERVER H
SERVER A
SIGN IN
TRANSFER
GET USER IP
SIGN IN
201
202
203
206
211
SET DEVICE
TRANSFER
RECORD DEVICE DATA
204
205
209
ALLOW B FOR MONITORING
TRANSFER
RECORD INFO.
207
208
210
SENT NOTIFICATION
212
213
214
215
SET/MODIFY USER INFO. DATA.
TRANSFER
RECORD NEW USER DATA
REQUEST FOR USER DATA
217
PROVIDES USER DATA
RECEIVES USER DATA
219
216
218
SET LOCAL DATASERVER AND NOTIFY
REQUEST FOR MONITORING
221
SEND RAW DATA VIA IP
DISPLAY DATA USING A SELECTED APPLICATION
222a
220
222b
CHATTING VIA IP
CHATTING VIA IP

FIG. 3

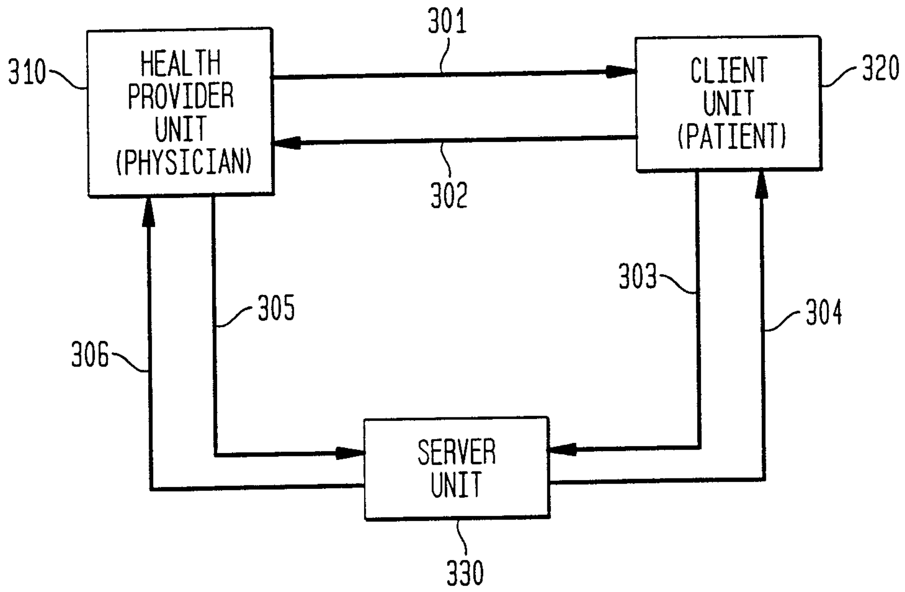

HEALTH PROVIDER UNIT (PHYSICIAN)
310
301
302
CLIENT UNIT (PATIENT)
320
303
304
305
306
SERVER UNIT
330

FIG. 4

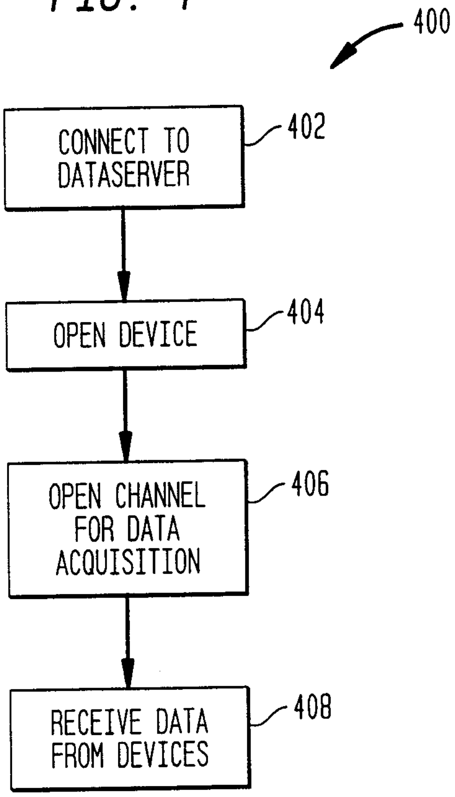
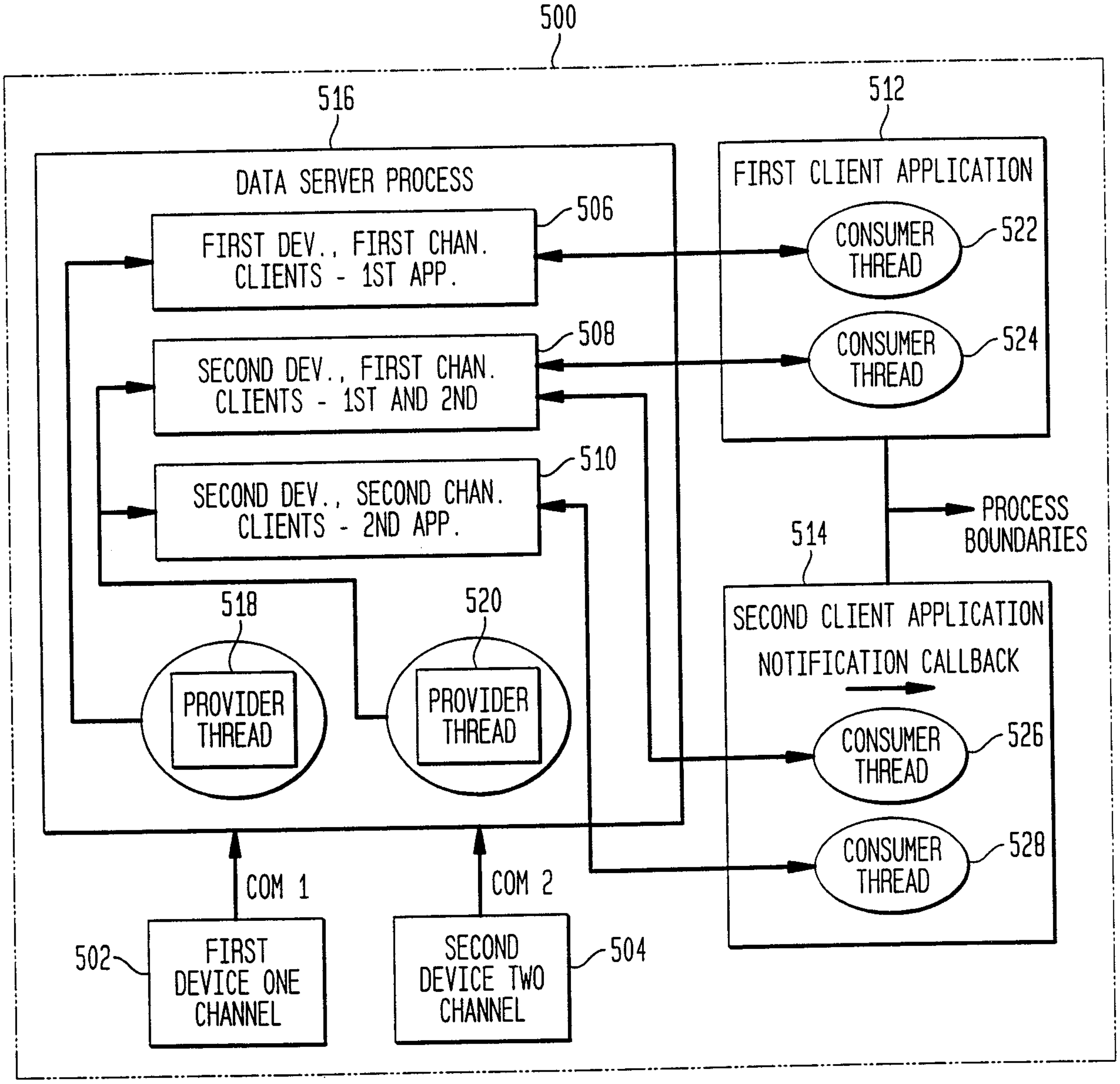

400
CONNECT TO DATASERVER
402
OPEN DEVICE
404
OPEN CHANNEL FOR DATA ACQUISITION
406
RECEIVE DATA FROM DEVICES
408

FIG. 5


500
516
DATA SERVER PROCESS
FIRST DEV., FIRST CHAN. CLIENTS - 1ST APP.
506
SECOND DEV., FIRST CHAN. CLIENTS - 1ST AND 2ND
508
SECOND DEV., SECOND CHAN. CLIENTS - 2ND APP.
510
518
PROVIDER THREAD
520
PROVIDER THREAD
512
FIRST CLIENT APPLICATION
CONSUMER THREAD
522
CONSUMER THREAD
524
PROCESS BOUNDARIES
514
SECOND CLIENT APPLICATION
NOTIFICATION CALLBACK
CONSUMER THREAD
526
CONSUMER THREAD
528
COM 1
COM 2
502
FIRST DEVICE ONE CHANNEL
SECOND DEVICE TWO CHANNEL
504