US 20090118596A1

(19) **United States**

(12) **Patent Application Publication**

(10) **Pub. No.: US 2009/0118596 A1**

(43) **Pub. Date: May 7, 2009**

Khanuja et al.

(54) **METHOD AND APPARATUS FOR REMOTELY MONITORING THE CONDITION OF A PATIENT**

(75) Inventors: **Sukhwant Singh Khanuja**, Chicago, IL (US); **Sandeep Garg**, Naperville, IL (US); **Irwin Preet Singh**, Prospect Heights, IL (US)

Correspondence Address:
MCANDREWS HELD & MALLOY, LTD
500 WEST MADISON STREET, SUITE 3400
CHICAGO, IL 60661

(73) Assignee: **Carematix, Inc.**, Chicago, IL (US)

(21) Appl. No.: **12/259,905**

(22) Filed: **Oct. 28, 2008**

Related U.S. Application Data

(63) Continuation of application No. 10/414,326, filed on Apr. 15, 2003, now Pat. No. 7,448,996.

(60) Provisional application No. 60/372,894, filed on Apr. 16, 2002.

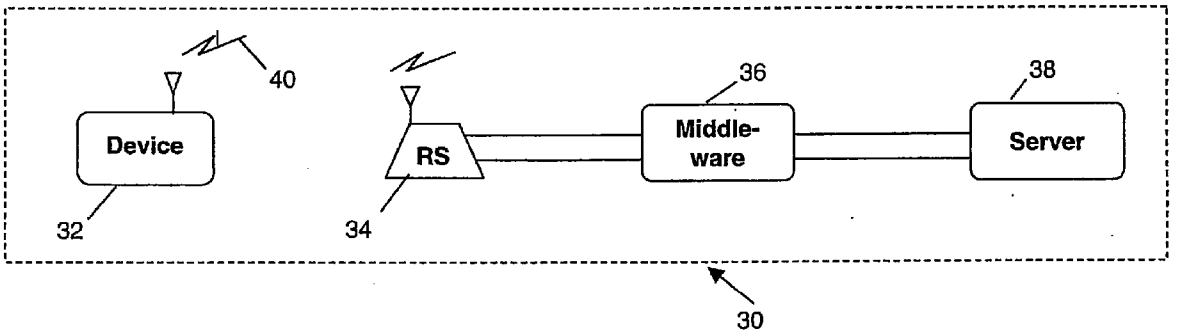
Publication Classification

(51) **Int. Cl.**
A61B 5/02 (2006.01)
A61B 5/00 (2006.01)

(52) **U.S. Cl.** **600/301**; 600/300

(57) **ABSTRACT**

A patient monitoring system provides enhanced functional capability relative to known systems and provides a wireless communication link between a patient monitoring device, worn by a patient, and a local hub. The patient monitoring system is adapted to monitor various patient physiological characteristics, such as blood pressure, pulse rate, blood glucose, weight, pulse oximetry and others. The data from the patient monitoring device is wirelessly transmitted to a local hub, which, in turn, is configured to automatically transfer the data to a remote server, for example, over a public or private communications network. In one embodiment of the invention, the server is configured as a web portal to selectively allow access to such patient physiological data by designated third parties, such as physicians, clinicians, relatives and the patient themselves.



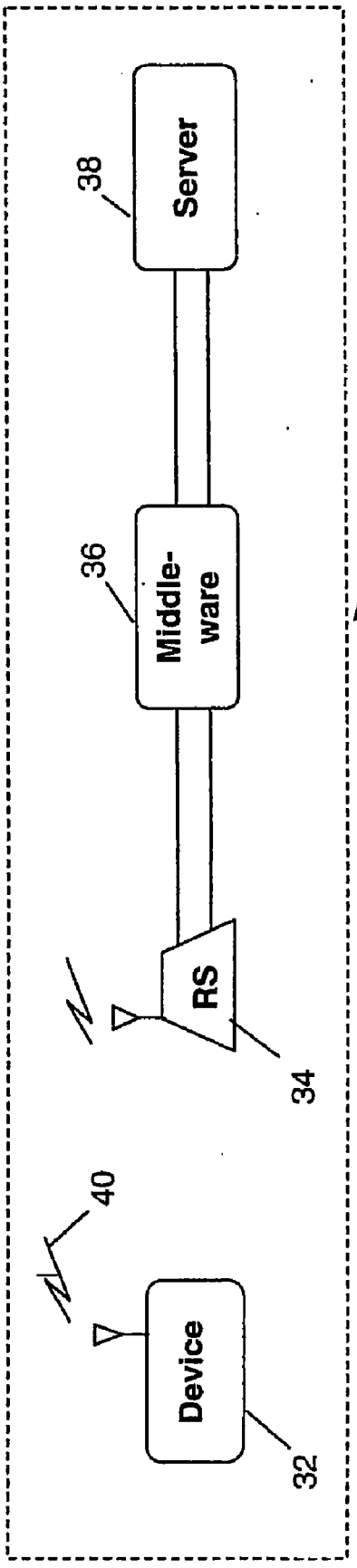


Fig. 1

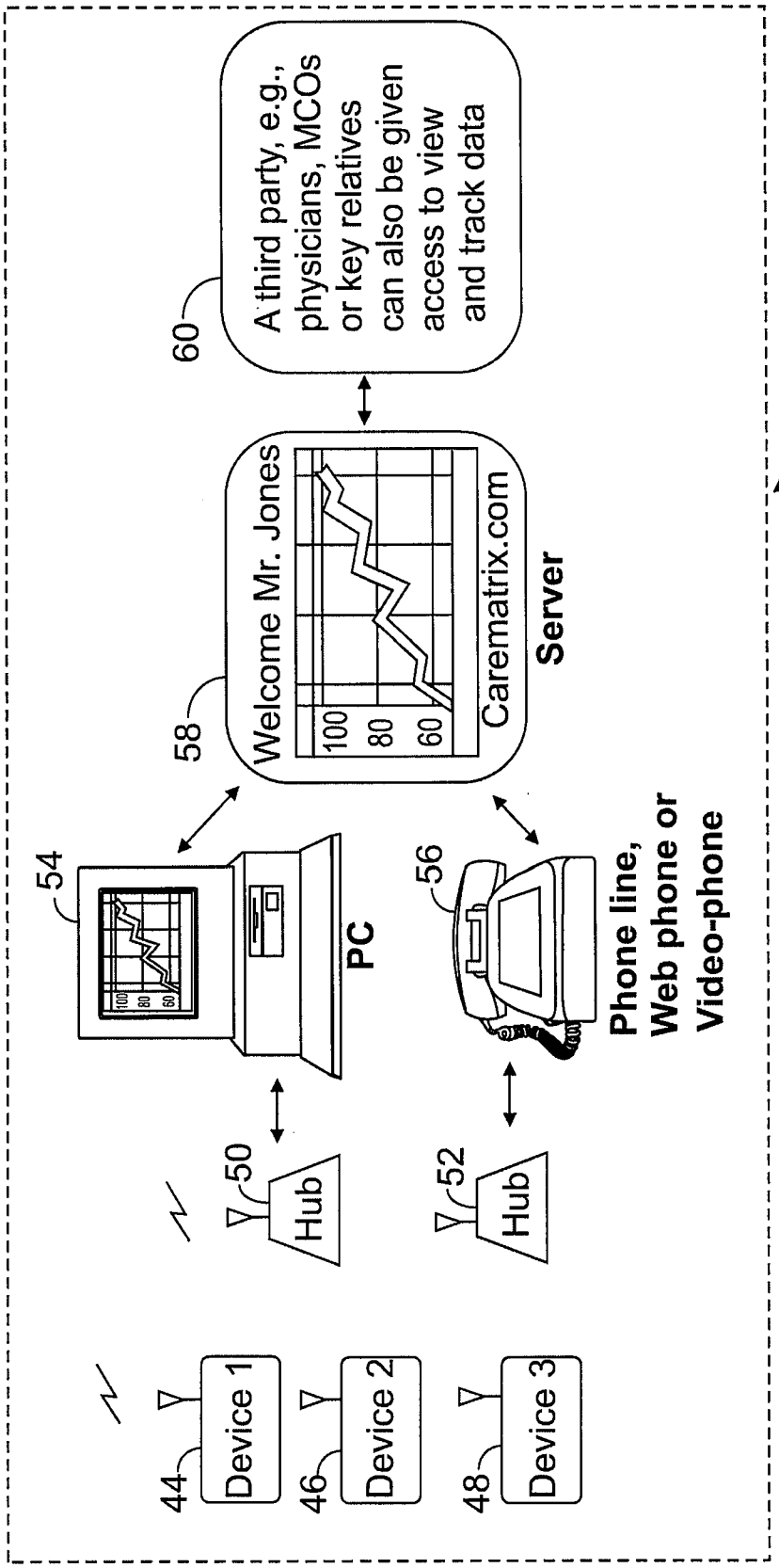

42
60
A third party, e.g., physicians, MCOs or key relatives can also be given access to view and track data
58
Welcome Mr. Jones
100
80
60
Carematrix.com
Server
54
PC
56
Phone line, Web phone or Video-phone
50
Hub
52
Hub
44
Device 1
46
Device 2
48
Device 3

FIG.2

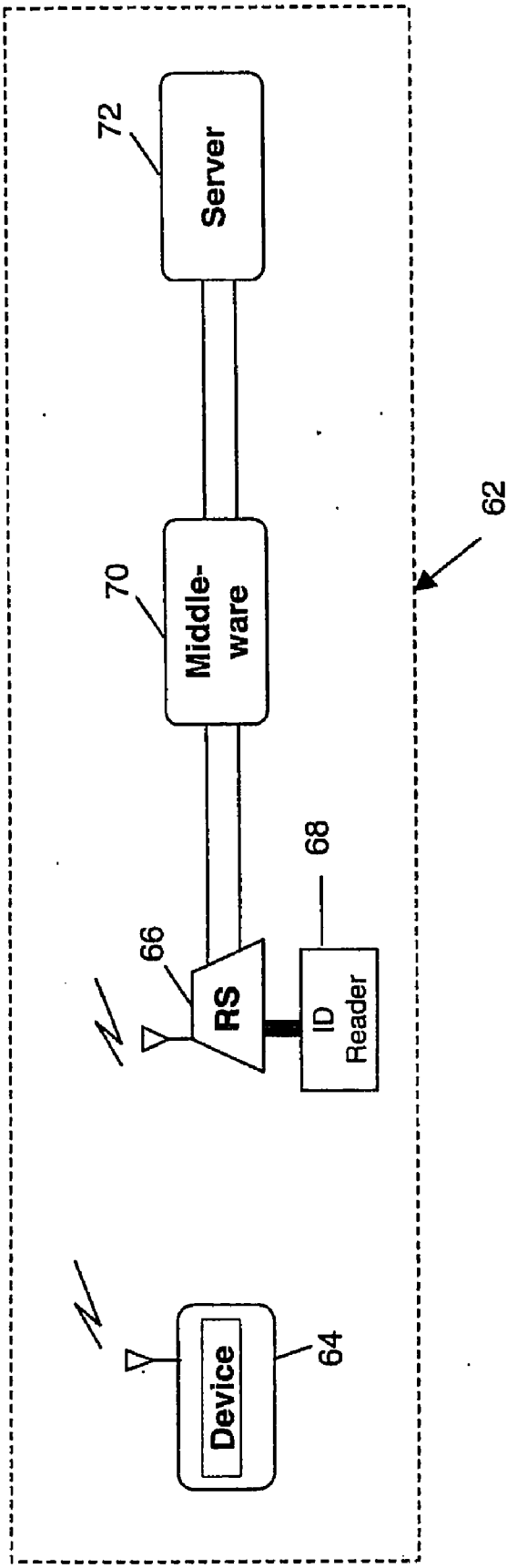


Fig. 3

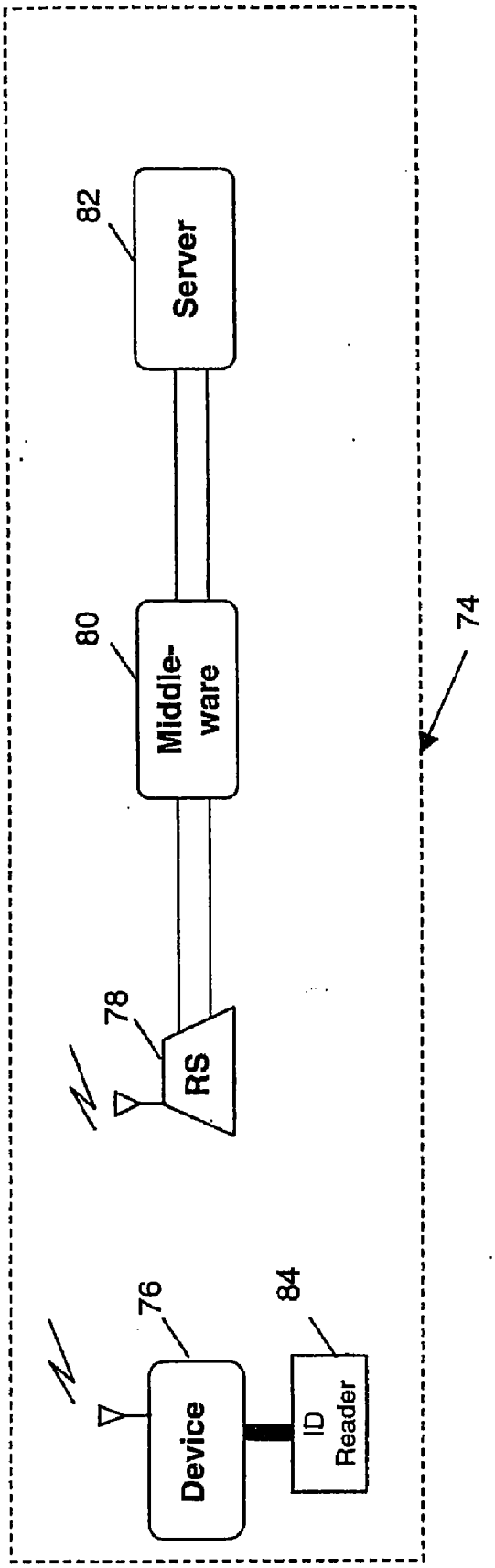


Fig. 4

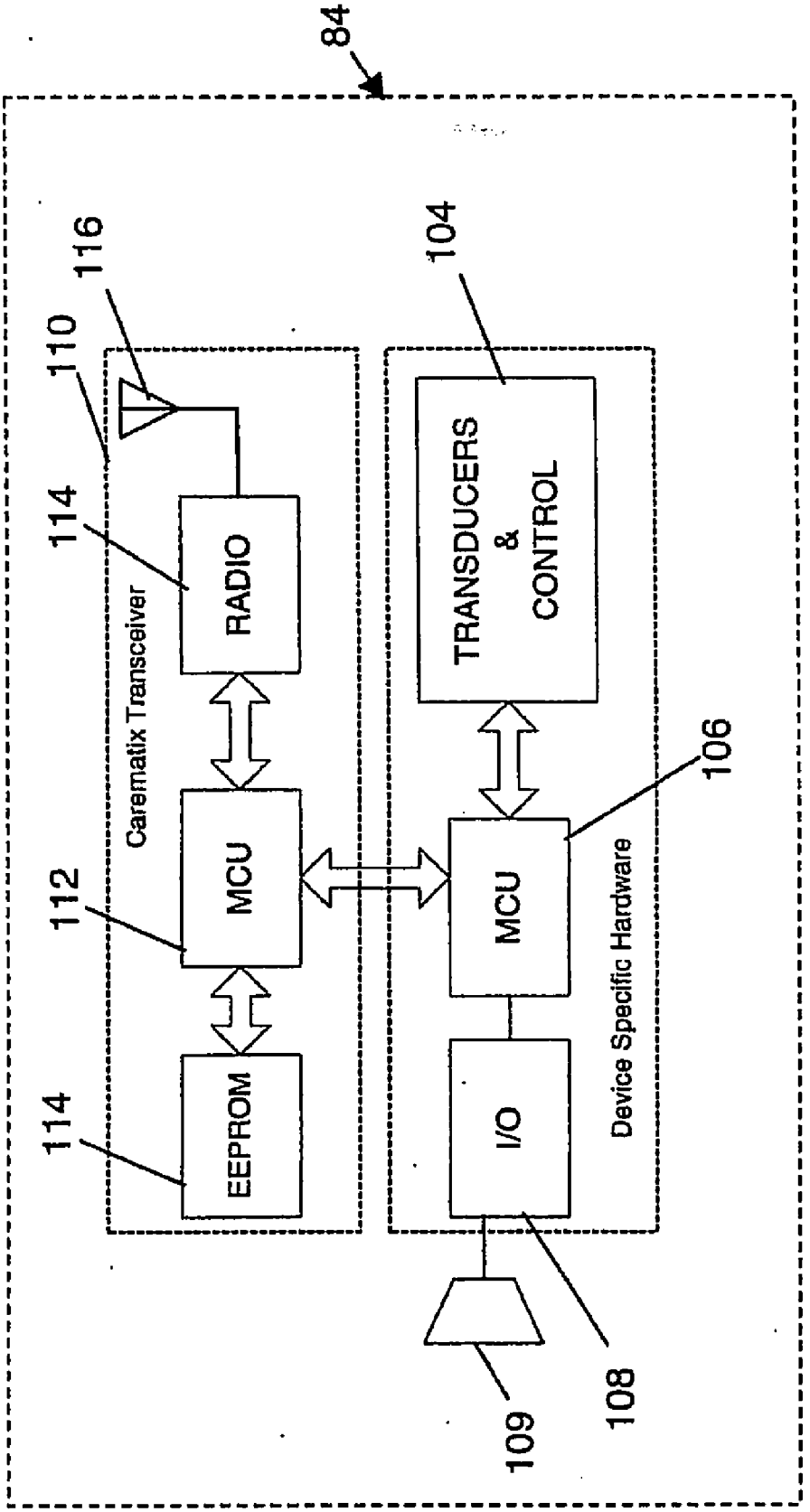

84
116
110
114
112
114
104
106
108
109
Caremattix Transceiver
RADIO
MCU
EEPROM
TRANSDUCERS & CONTROL
MCU
I/O
Device Specific Hardware

Fig. 5

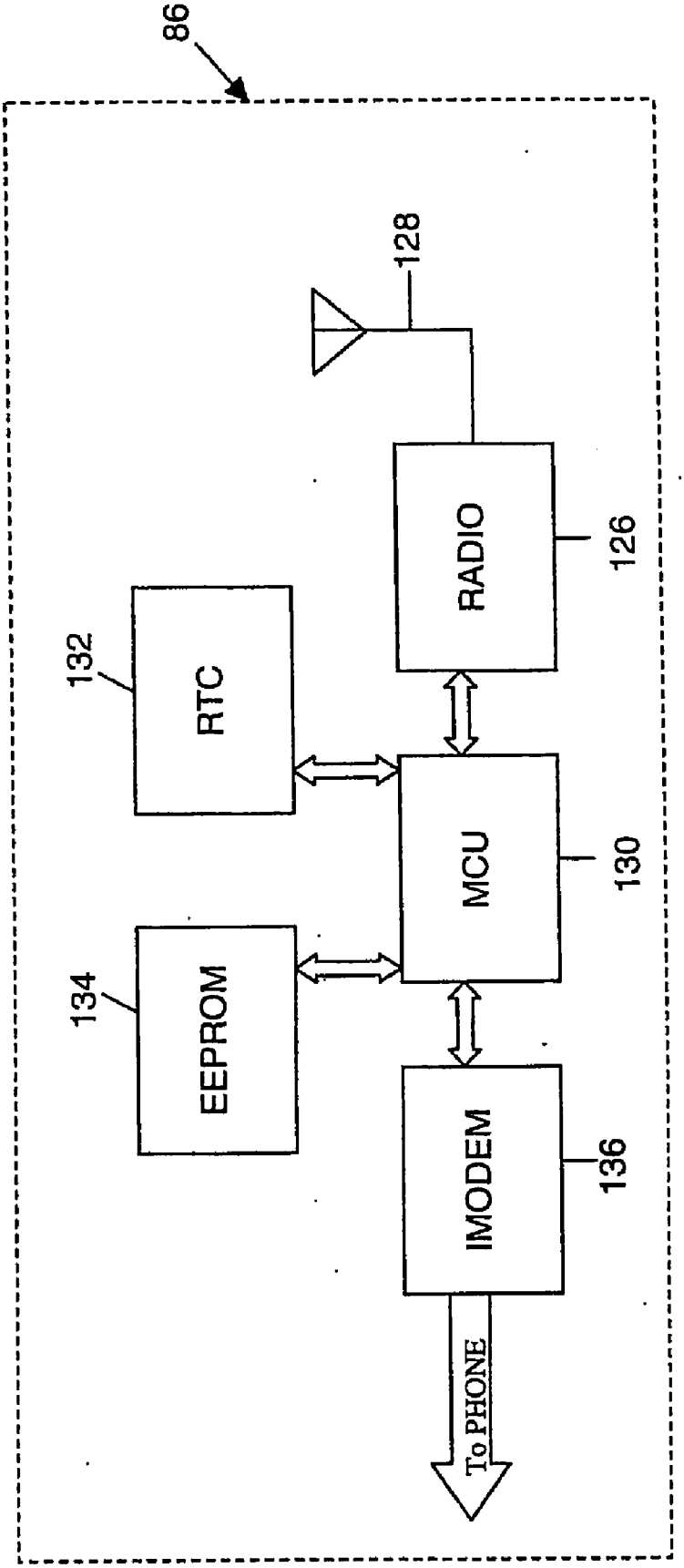

86
128
RADIO
126
132
RTC
MCU
130
134
EEPROM
IMODEM
136
To PHONE

Fig. 6

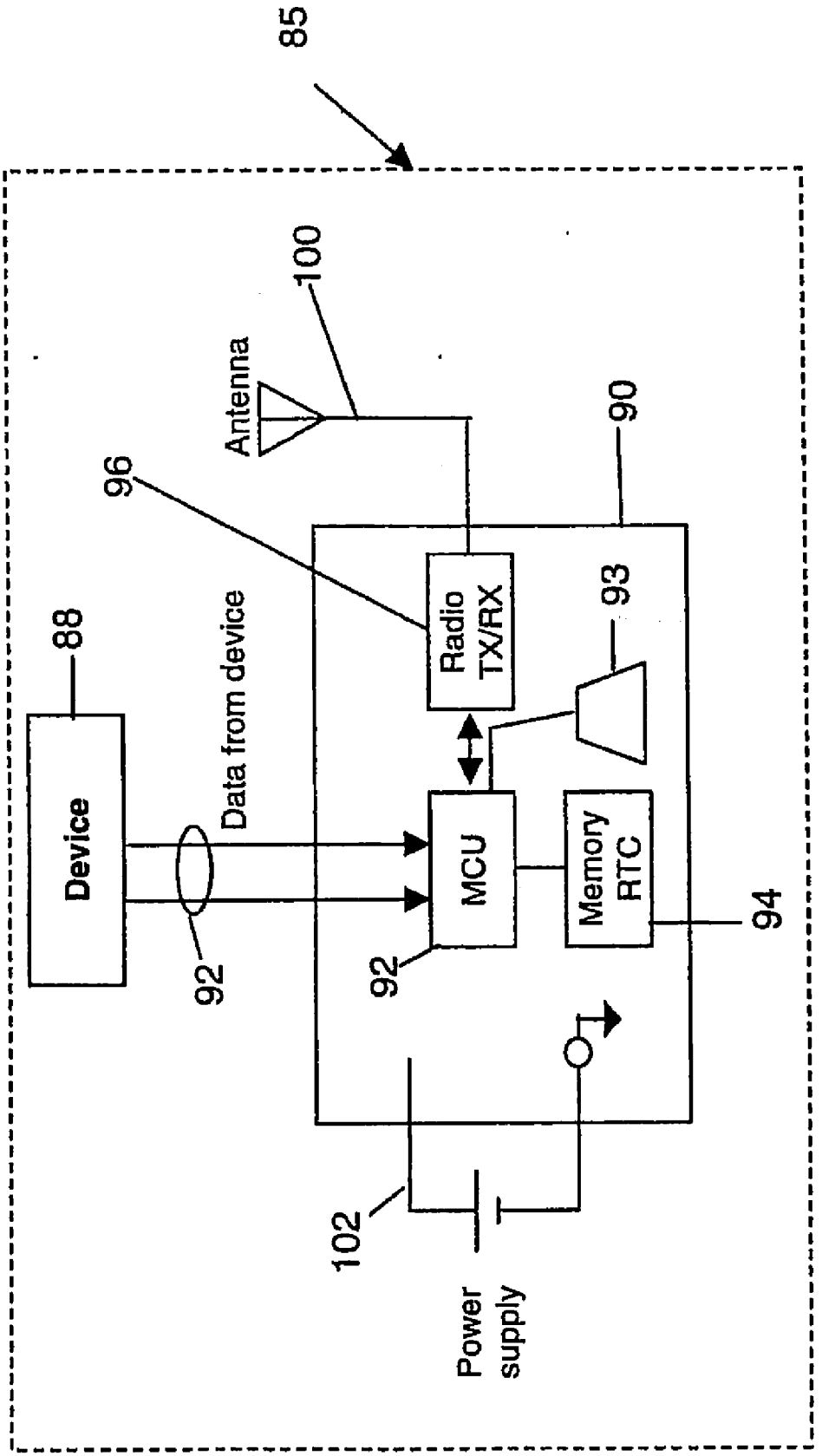

85
100
Antenna
90
96
Radio TX/RX
93
88
Device
Data from device
MCU
Memory RTC
94
92
92
102
Power supply

Fig. 7

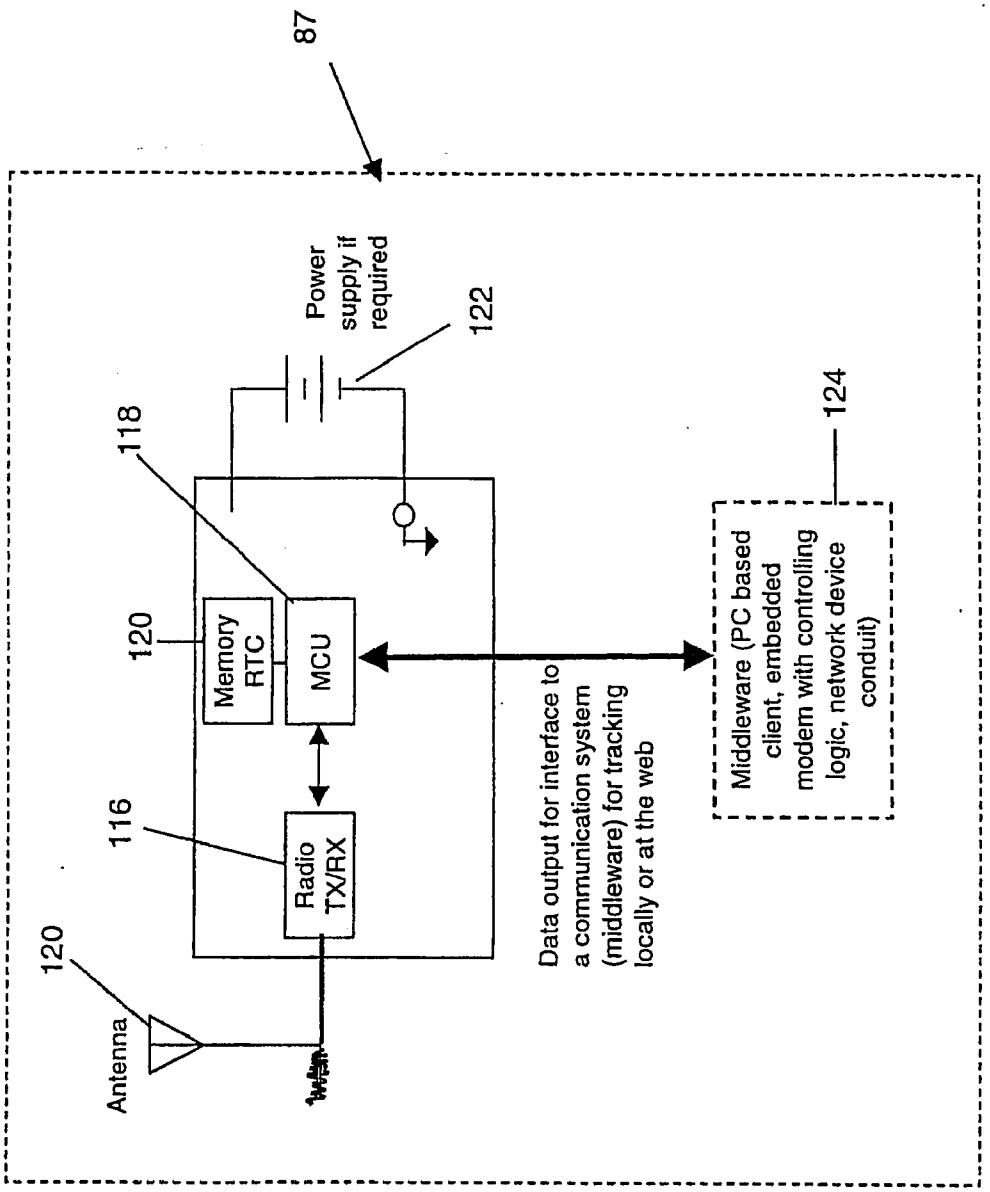

87
Power supply if required
122
118
120
Memory RTC
MCU
116
Radio TX/RX
120
Antenna
Data output for interface to a communication system (middleware) for tracking locally or at the web
124
Middleware (PC based client, embedded modem with controlling logic, network device conduit)

Fig. 8

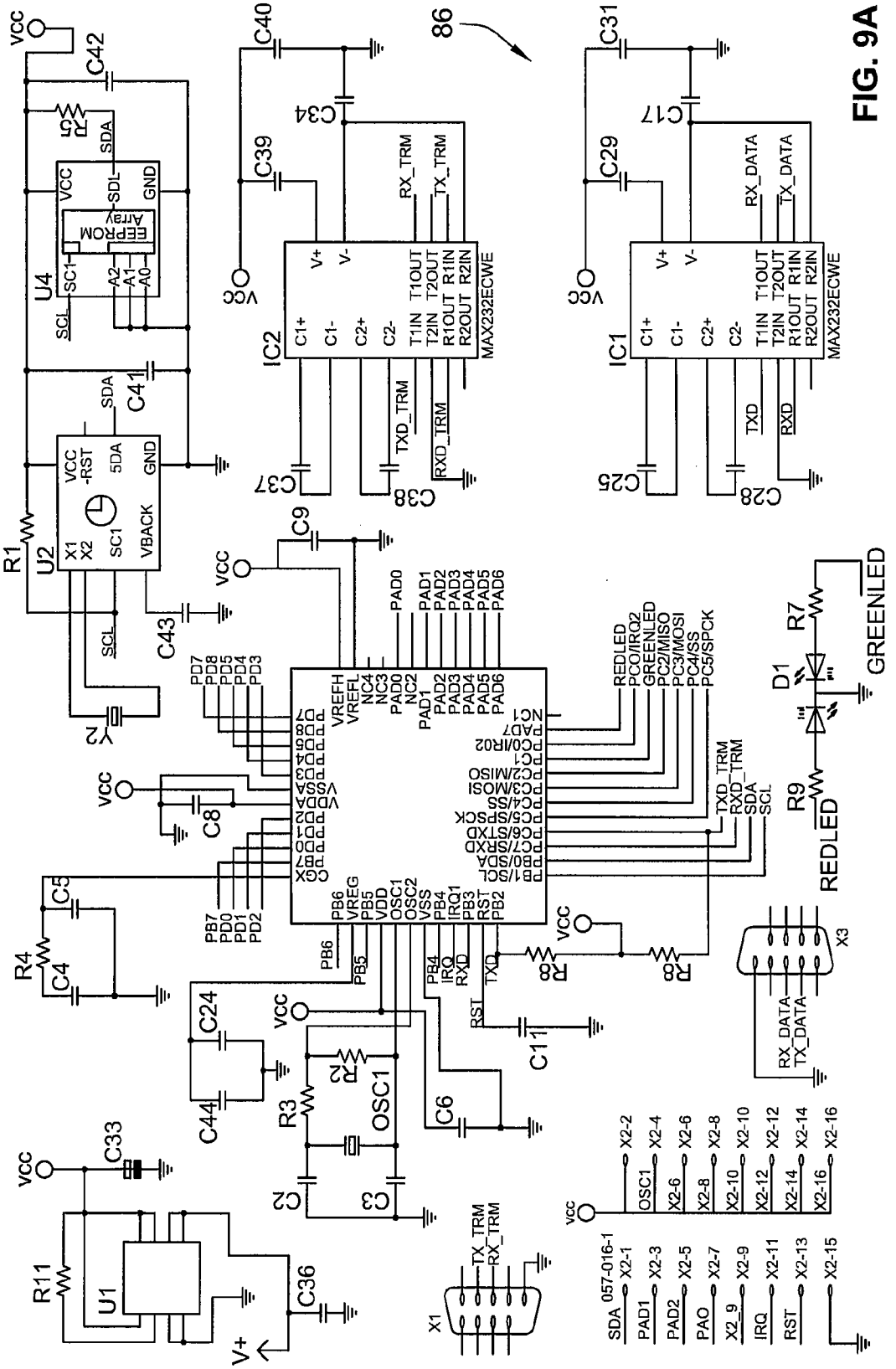

VCC
C42
R5
SDA
SDL
GND
EEPROM Array
U4
SC1
A2
A1
A0
SCL
C41
SDA
VCC -RST
5DA
GND
U2
X1
X2
SC1
VBACK
R1
C43
SCL
Y2
C9
VCC
C8
C5
R4
C4
C24
C44
R3
R2
OSC1
C6
C33
R11
U1
C36
V+
C2
C3
C40
C34
C39
86
C31
C17
C29
IC2
IC1
C1+
C1-
C2+
C2-
V+
V-
T1IN T2IN R1OUT R2OUT
T1OUT T2OUT R1IN R2IN
MAX232ECWE
RX_TRM
TX_TRM
TXD_TRM
RXD_TRM
RX_DATA
TX_DATA
TXD
RXD
C37
C38
C25
C28
PAD0 PAD1 PAD2 PAD3 PAD4 PAD5 PAD6
PD7 PD8 PD5 PD4 PD3
VREFH VREFL NC4 NC3 NC2
REDLED PC0/IRQ2 GREENLED PC2/MISO PC3/MOSI PC4/SS PC5/SPSCK
PC1 PC2/MISO PC3/MOSI PC4/SS PC5/SPSCK PC6/STXD PC7/SRXD PB0/SDA PB1/SCL
VDDA VSSA
PD2 PD1 PD0 PB7
CGX
PB6 VREG PB5 VDD OSC1 OSC2 VSS PB4 IRQ1 PB3 RST PB2
PB7 PD0 PD1 PD2
PB6 PB5
PB4 IRQ RXD TXD
RST
R8
R7
D1
GREENLED
R9
REDLED
X3
X1
TX_TRM RX_TRM
SDA 057-016-1 X2-1
PAD1 X2-3
PAD2 X2-5
PA0 X2-7
X2_9 X2-9
IRQ X2-11
RST X2-13
X2-15
OSC1 X2-2 X2-4 X2-6 X2-8 X2-10 X2-12 X2-14 X2-16

FIG. 9A

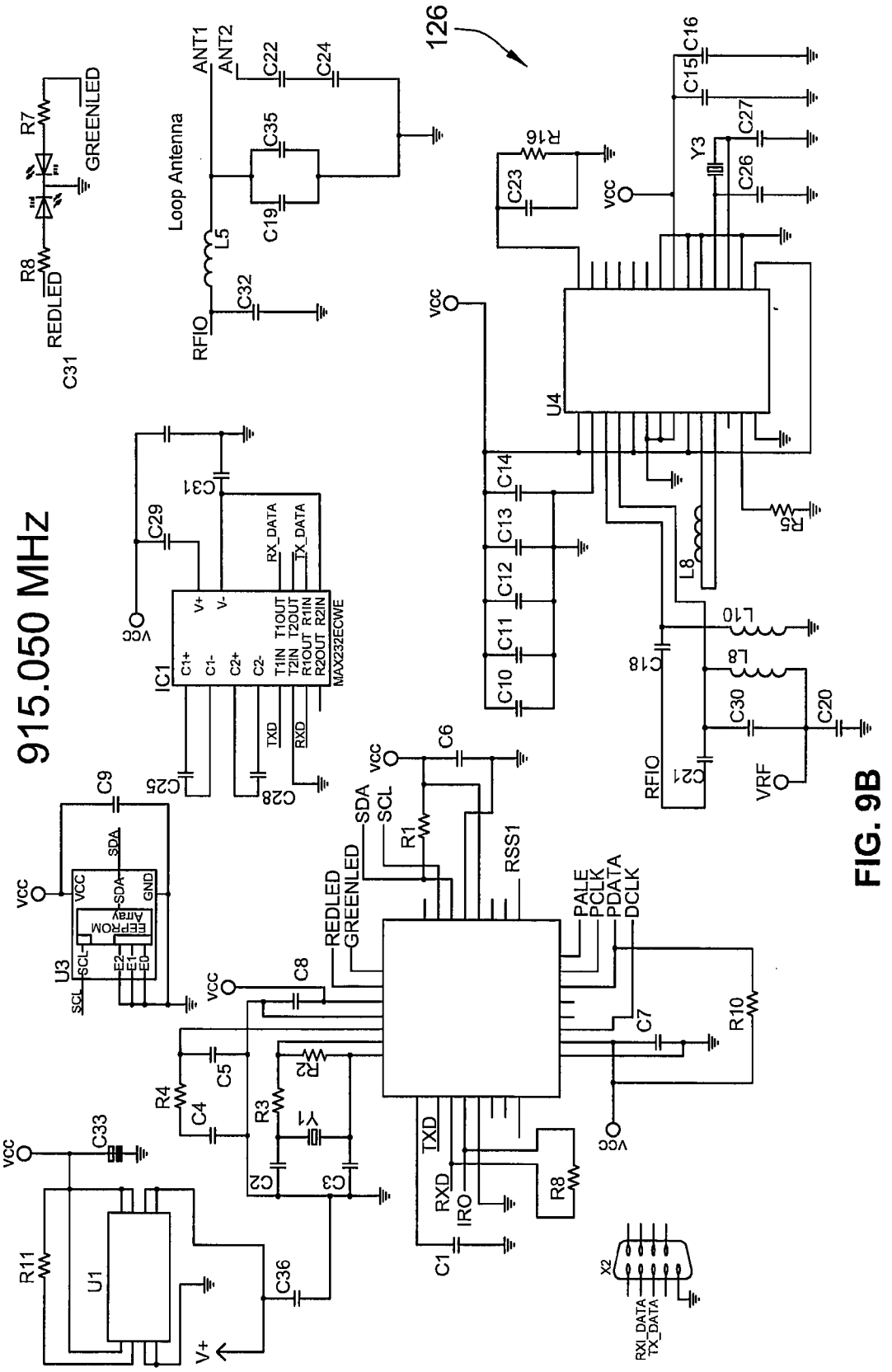

915.050 MHz
126
Loop Antenna
ANT1
ANT2
RFIO
L5
C19
C22
C24
C35
C32
R7
R8
GREENLED
REDLED
C31
U4
VCC
C14
C13
C12
C11
C10
C15
C16
C27
C26
Y3
C23
R16
L8
L10
C18
C30
C20
VRF
C21
R5
IC1
MAX232ECWE
C1+
C1-
C2+
C2-
V+
V-
T1IN
T2IN
R1OUT
R2OUT
T1OUT
T2OUT
R1IN
R2IN
RX_DATA
TX_DATA
TXD
RXD
C29
C31
C25
C28
U3
SCL
SDA
VCC
GND
E2
E1
E0
EEPROM
Array
C9
C6
C8
R1
RSS1
REDLED
GREENLED
PALE
PCLK
PDATA
DCLK
C7
R10
R2
R3
R4
C4
C5
Y1
C2
C3
R8
IRQ
C1
X2
RXI_DATA
TX_DATA
U1
R11
C33
C36
V+

FIG. 9B

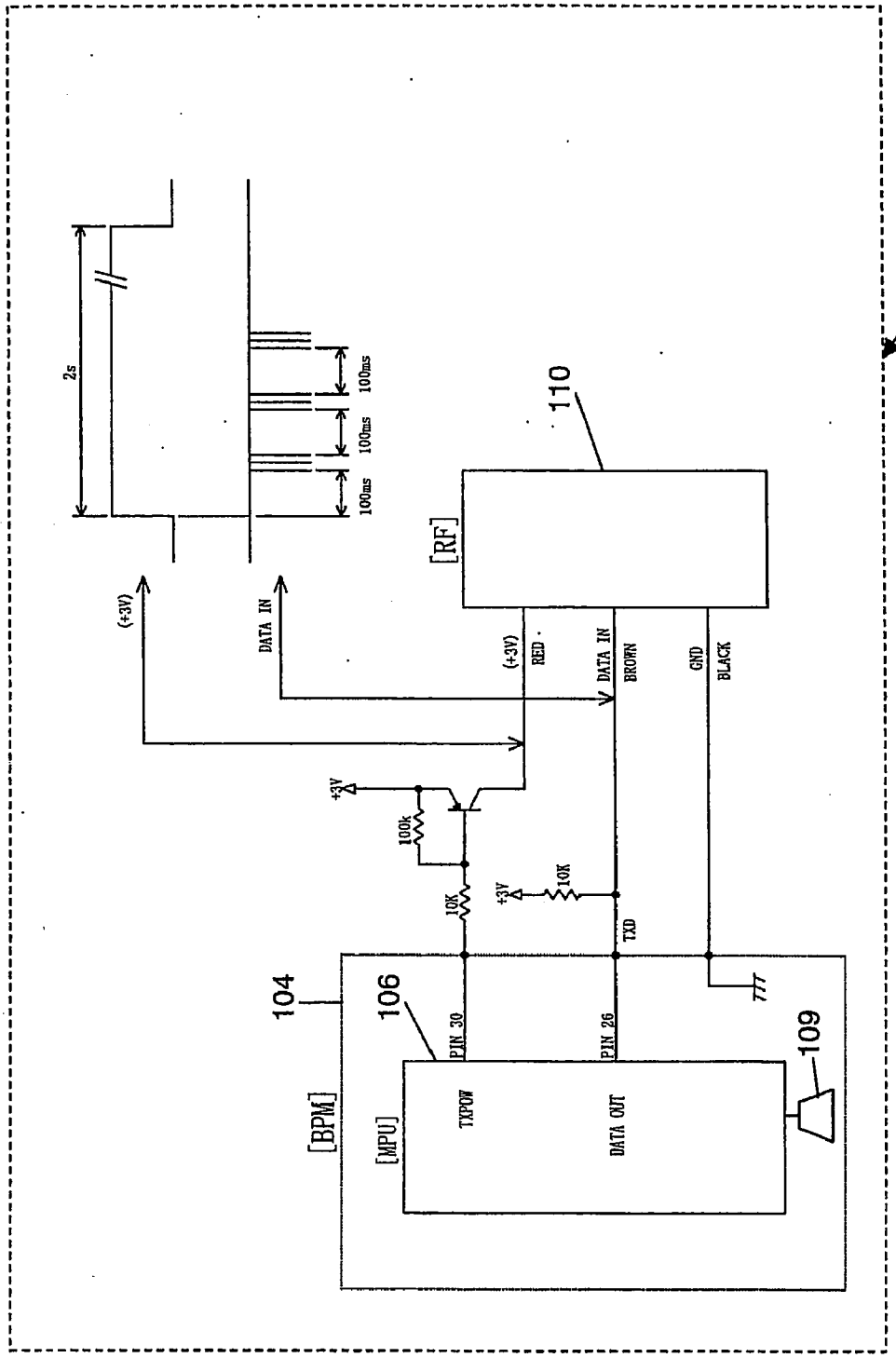

84
110
[RF]
(+3V)
DATA IN
RED
BROWN
BLACK
GND
2s
100ms
100ms
100ms
100K
10K
10K
+3V
+3V
TXD
104
106
109
[BPM]
[MPU]
PIN 30
PIN 26
TXPOW
DATA OUT

Fig. 10

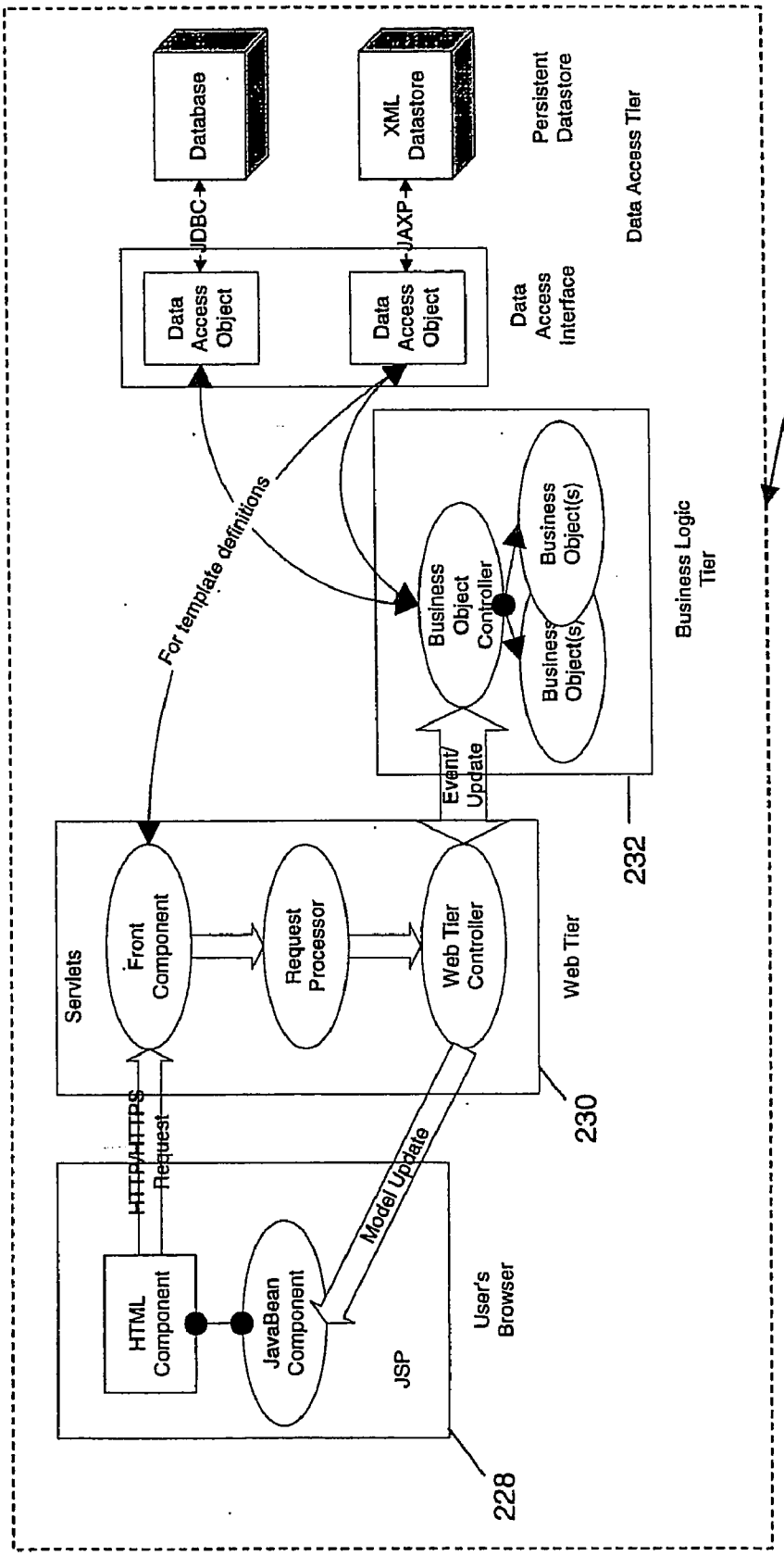

Database
XML Datastore
Persistent Datastore
Data Access Tier
JDBC
JAXP
Data Access Object
Data Access Object
Data Access Interface
234
For template definitions
Business Object Controller
Business Object(s)
Business Object(s)
Business Logic Tier
Event/ Update
232
Servlets
Front Component
Request Processor
Web Tier Controller
Web Tier
230
HTTP/HTTPS Request
Model Update
HTML Component
JavaBean Component
JSP
User's Browser
228

Fig. 11

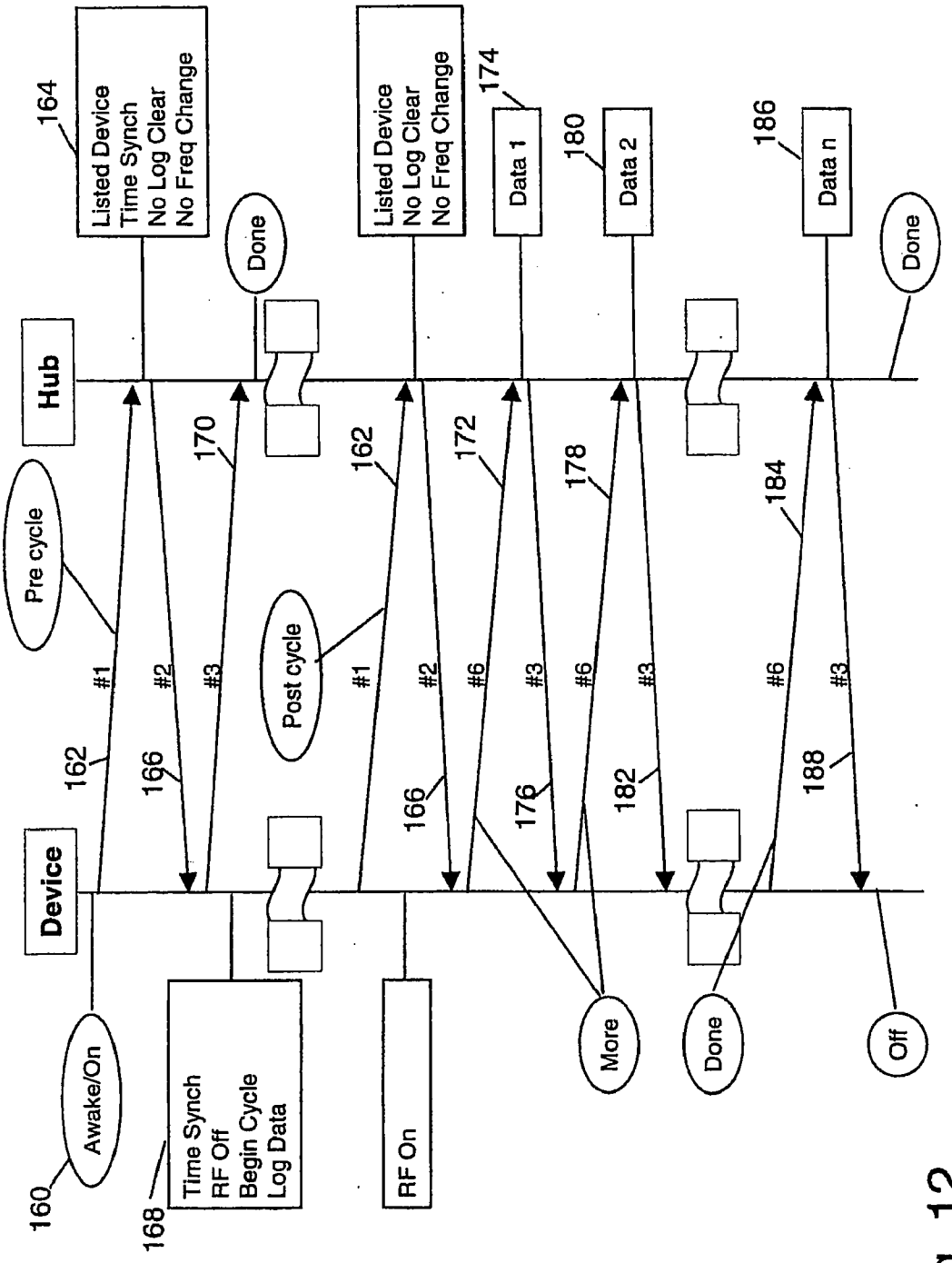

Hub
Device
Pre cycle
Post cycle
Awake/On
Time Synch
RF Off
Begin Cycle
Log Data
RF On
Listed Device
Time Synch
No Log Clear
No Freq Change
Listed Device
No Log Clear
No Freq Change
Data 1
Data 2
Data n
Done
Done
Done
More
Done
Off
#1
#2
#3
#1
#2
#6
#3
#6
#3
#6
#3
160
162
164
166
168
170
172
174
176
178
180
182
184
186
188

Fig. 12

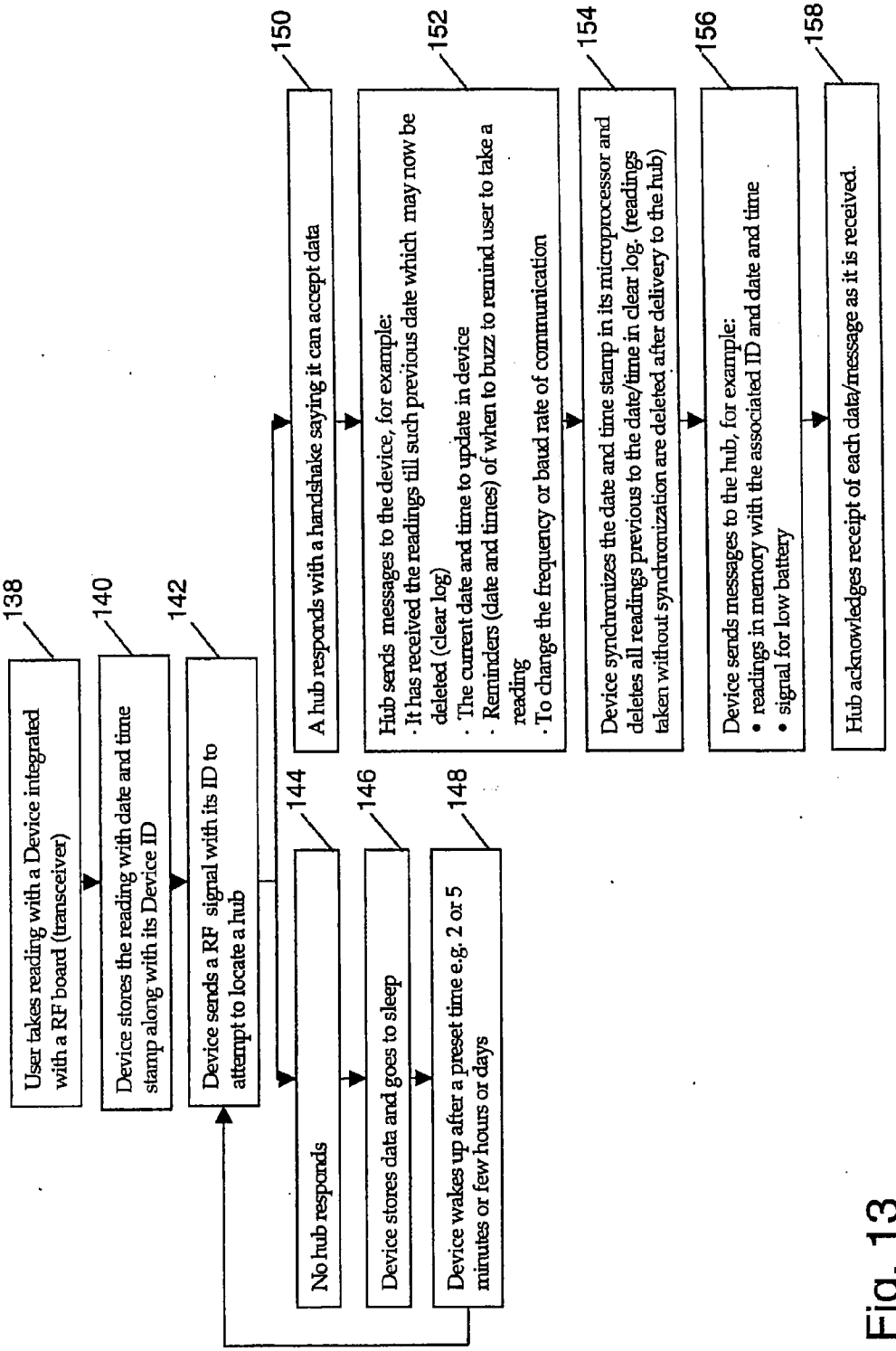

138
User takes reading with a Device integrated with a RF board (transceiver)
140
Device stores the reading with date and time stamp along with its Device ID
142
Device sends a RF signal with its ID to attempt to locate a hub
144
No hub responds
146
Device stores data and goes to sleep
148
Device wakes up after a preset time e.g. 2 or 5 minutes or few hours or days
150
A hub responds with a handshake saying it can accept data
152
Hub sends messages to the device, for example:
· It has received the readings till such previous date which may now be deleted (clear log)
· The current date and time to update in device
· Reminders (date and times) of when to buzz to remind user to take a reading
· To change the frequency or baud rate of communication
154
Device synchronizes the date and time stamp in its microprocessor and deletes all readings previous to the date/time in clear log. (readings taken without synchronization are deleted after delivery to the hub)
156
Device sends messages to the hub, for example:
• readings in memory with the associated ID and date and time
• signal for low battery
158
Hub acknowledges receipt of each data/message as it is received.

Fig. 13

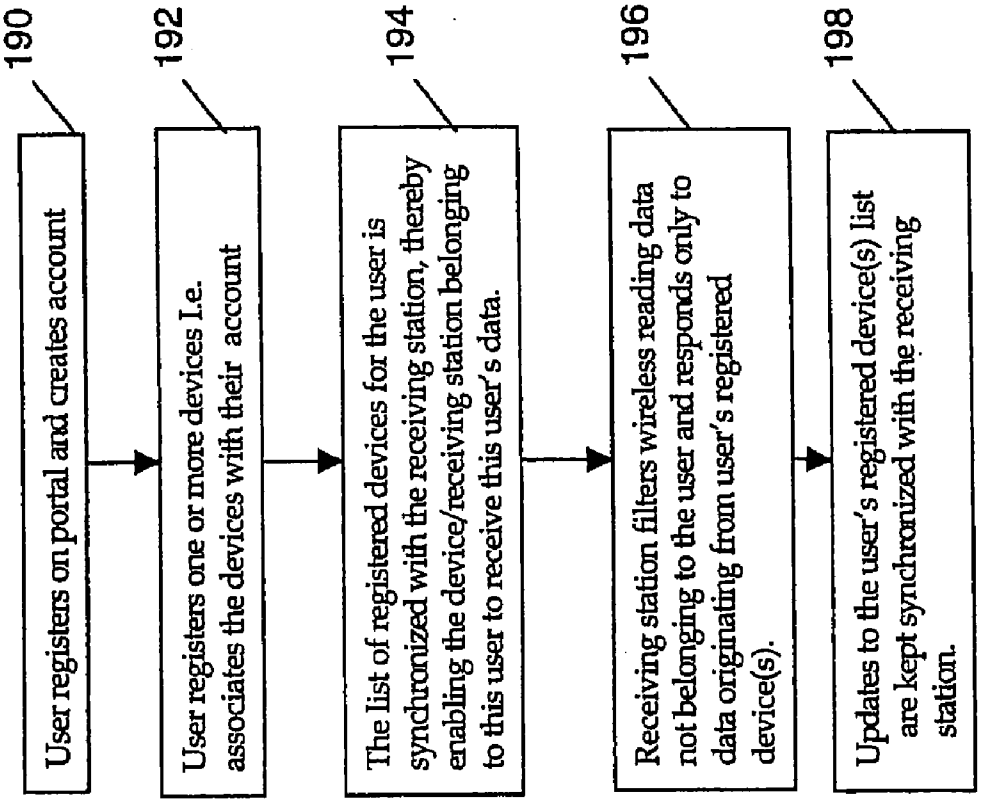

190
User registers on portal and creates account
192
User registers one or more devices I.e. associates the devices with their account
194
The list of registered devices for the user is synchronized with the receiving station, thereby enabling the device/receiving station belonging to this user to receive this user's data.
196
Receiving station filters wireless reading data not belonging to the user and responds only to data originating from user's registered device(s).
198
Updates to the user's registered device(s) list are kept synchronized with the receiving station.

Fig. 14

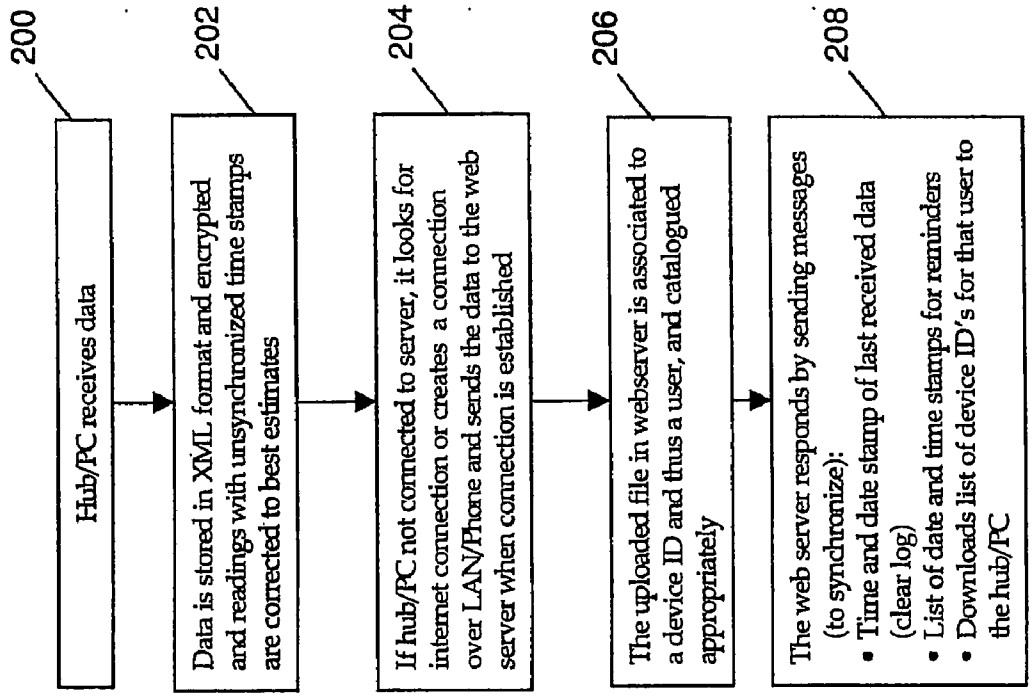

200
Hub/PC receives data
202
Data is stored in XML format and encrypted and readings with unsynchronized time stamps are corrected to best estimates
204
If hub/PC not connected to server, it looks for internet connection or creates a connection over LAN/Phone and sends the data to the web server when connection is established
206
The uploaded file in webserver is associated to a device ID and thus a user, and catalogued appropriately
208
The web server responds by sending messages (to synchronize):
• Time and date stamp of last received data (clear log)
• List of date and time stamps for reminders
• Downloads list of device ID's for that user to the hub/PC

Fig. 15

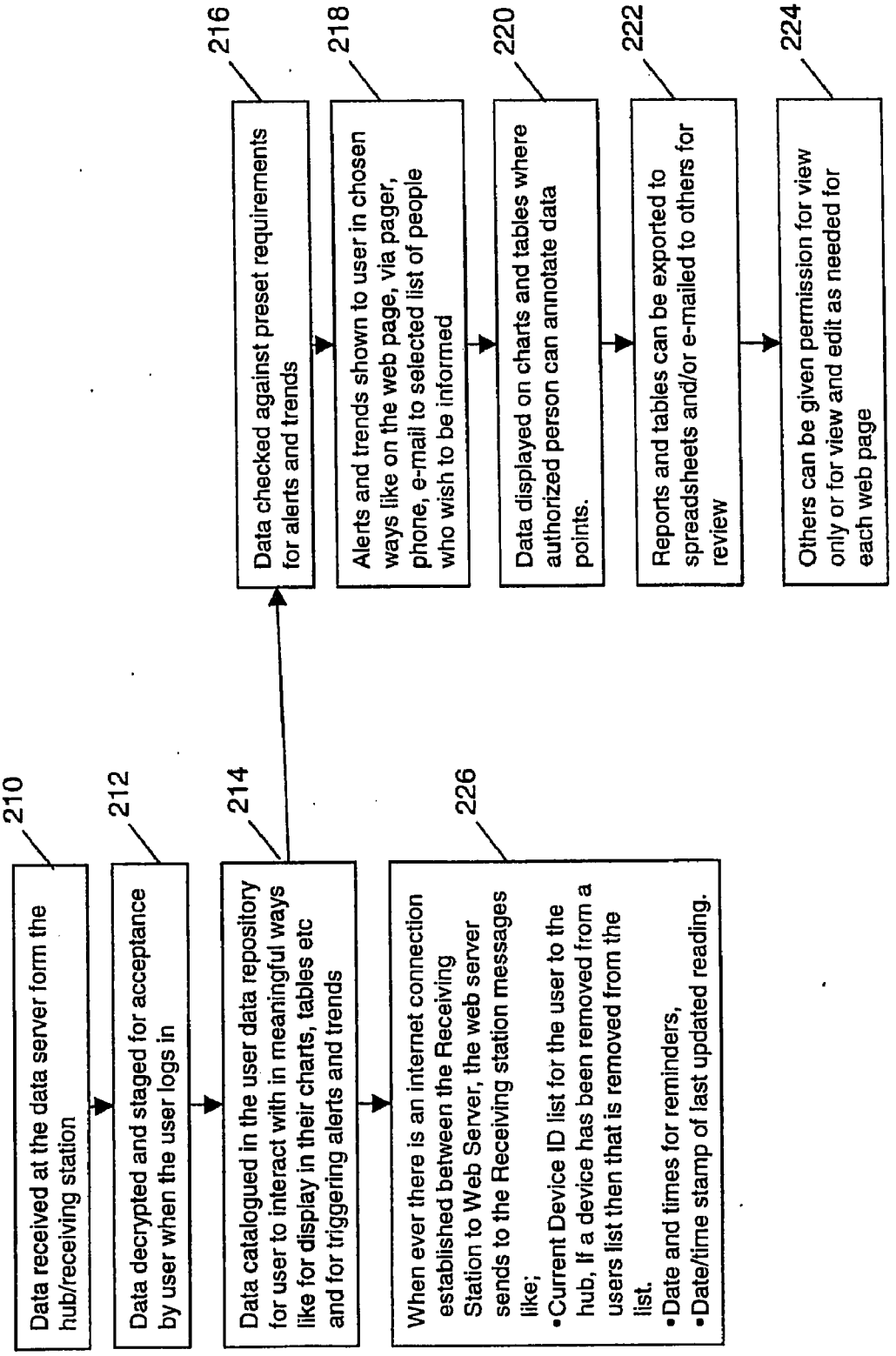

210
Data received at the data server form the hub/receiving station
212
Data decrypted and staged for acceptance by user when the user logs in
214
Data catalogued in the user data repository for user to interact with in meaningful ways like for display in their charts, tables etc and for triggering alerts and trends
216
Data checked against preset requirements for alerts and trends
218
Alerts and trends shown to user in chosen ways like on the web page, via pager, phone, e-mail to selected list of people who wish to be informed
220
Data displayed on charts and tables where authorized person can annotate data points.
222
Reports and tables can be exported to spreadsheets and/or e-mailed to others for review
224
Others can be given permission for view only or for view and edit as needed for each web page
226
When ever there is an internet connection established between the Receiving Station to Web Server, the web server sends to the Receiving station messages like;
•Current Device ID list for the user to the hub, If a device has been removed from a users list then that is removed from the list.
•Date and times for reminders,
•Date/time stamp of last updated reading.

Fig. 16

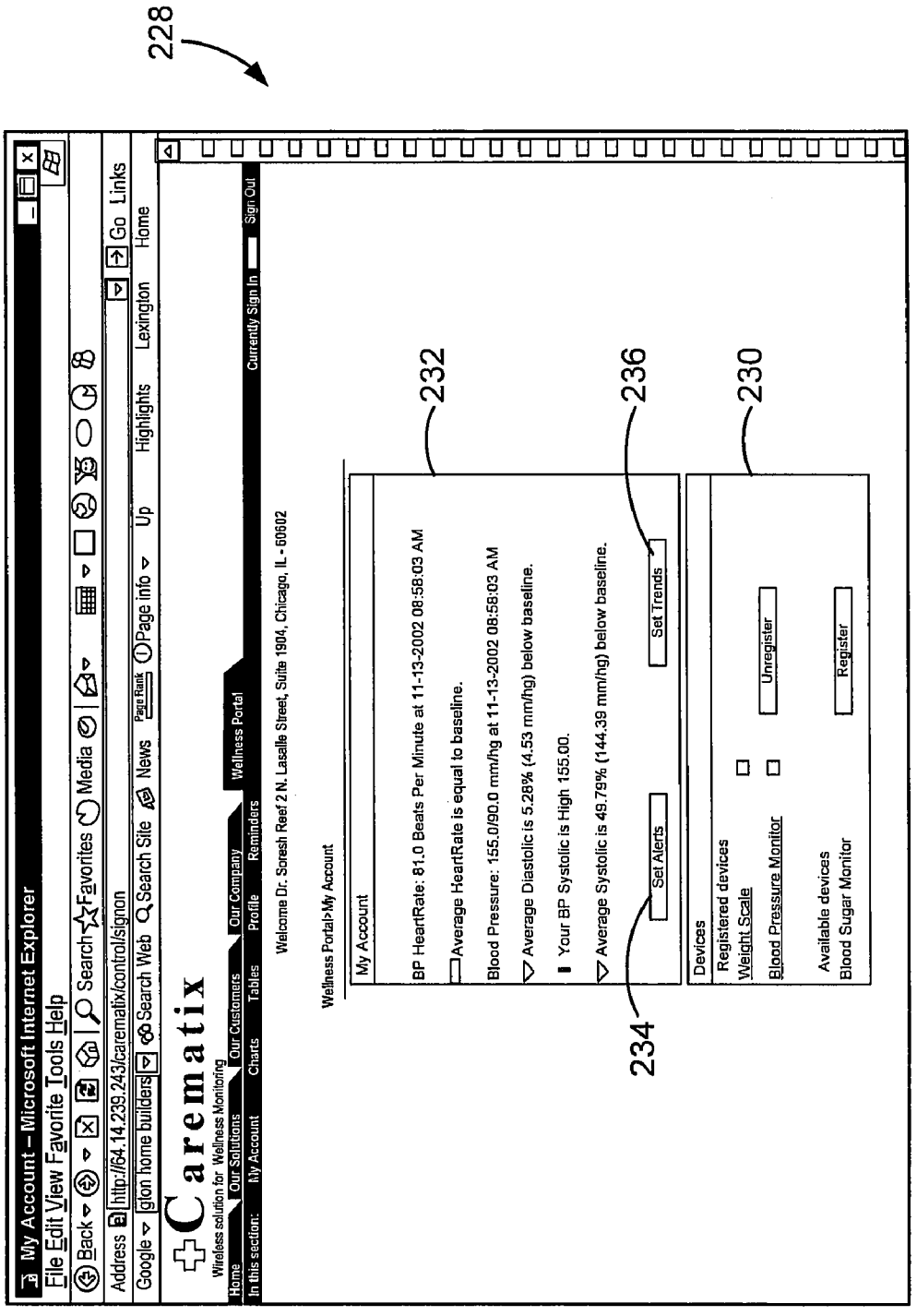

228
My Account – Microsoft Internet Explorer
File Edit View Favorite Tools Help
Address http://64.14.239.243/carematix/control/signon
Carematix
Wireless solution for Wellness Monitoring
Home
In this section: My Account Charts Tables Profile Reminders
Our Solutions Our Customers Our Company Wellness Portal
Currently Sign In Sign Out
Welcome Dr. Soresh Reef 2 N. Lasalle Street, Suite 1904, Chicago, IL - 60602
Wellness Portal>My Account
My Account
BP HeartRate: 81.0 Beats Per Minute at 11-13-2002 08:58:03 AM
Average HeartRate is equal to baseline.
Blood Pressure: 155.0/90.0 mm/hg at 11-13-2002 08:58:03 AM
Average Diastolic is 5.28% (4.53 mm/hg) below baseline.
Your BP Systolic is High 155.00.
Average Systolic is 49.79% (144.39 mm/hg) below baseline.
Set Alerts
Set Trends
Devices
Registered devices
Weight Scale
Blood Pressure Monitor
Unregister
Available devices
Blood Sugar Monitor
Register
232
236
230
234

FIG. 17

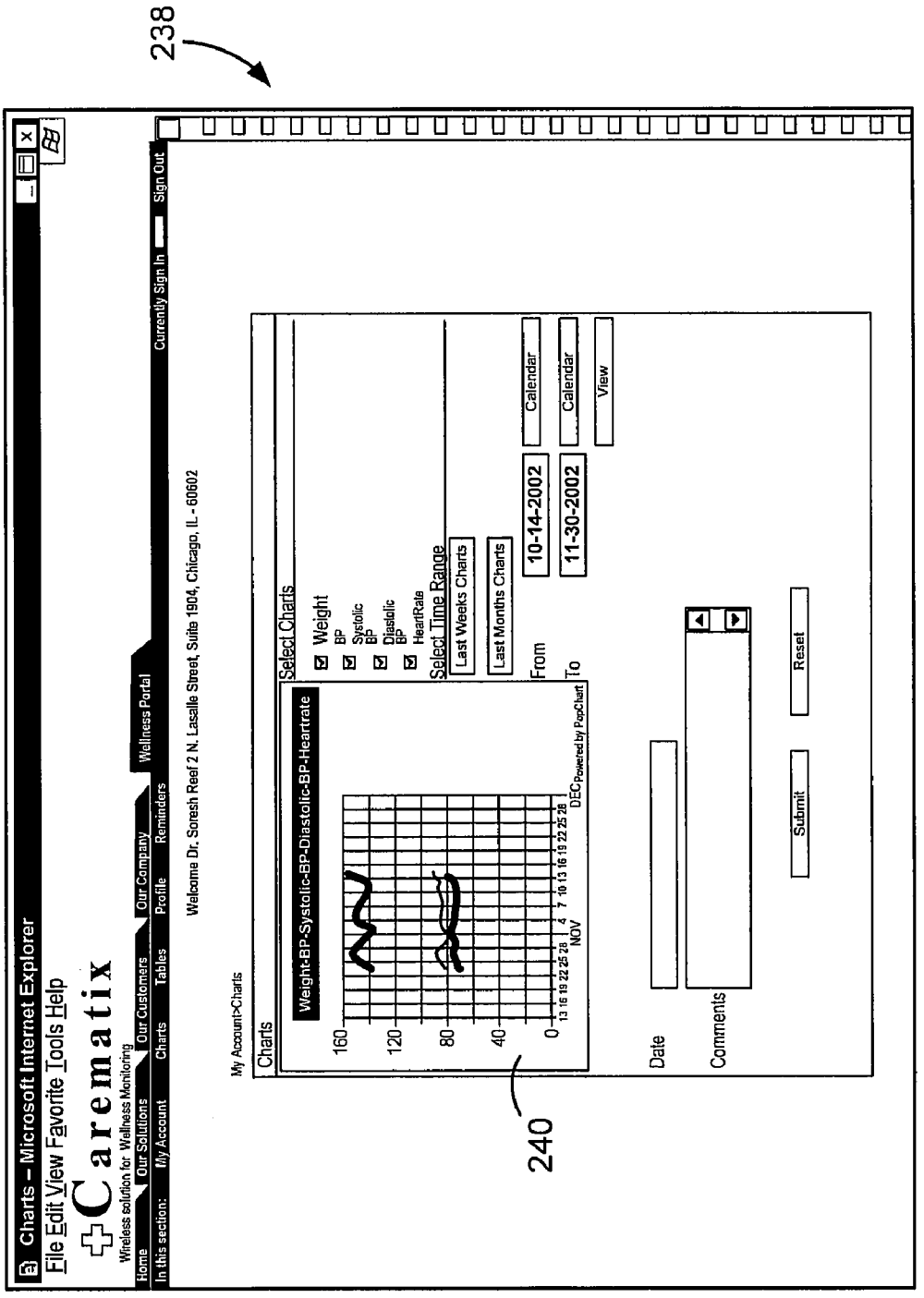

238
240
Charts - Microsoft Internet Explorer
File Edit View Favorite Tools Help
Carematix
Wireless solution for Wellness Monitoring
Home
In this section:
My Account
Our Solutions
Our Customers
Charts
Our Company
Tables
Profile
Wellness Portal
Reminders
Currently Sign In
Sign Out
Welcome Dr. Soresh Reef 2 N. Lasalle Street, Suite 1904, Chicago, IL - 60602
My Account>Charts
Charts
Select Charts
Weight-BP-Systolic-BP-Diastolic-BP-Heartrate
Weight
BP
Systolic BP
Diastolic BP
HeartRate
Select Time Range
Last Weeks Charts
Last Months Charts
From
10-14-2002
Calendar
To
11-30-2002
Calendar
View
Reset
Submit
Date
Comments
160
120
80
40
0
NOV
DEC
Powered by PopChart

FIG. 18

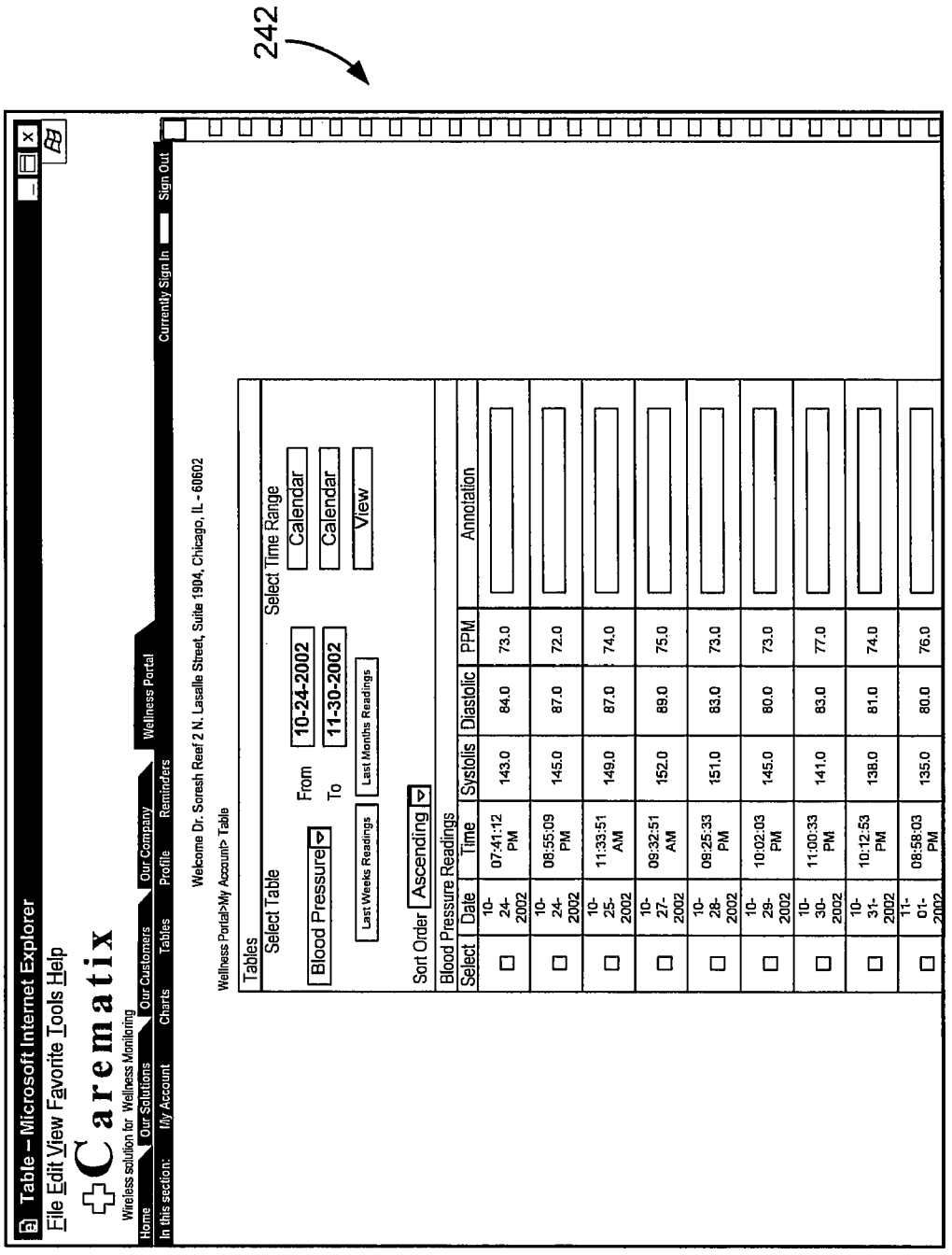

242
Table – Microsoft Internet Explorer
File Edit View Favorite Tools Help
Caremetrix
Wireless solution for Wellness Monitoring
Home
Our Solutions
Our Customers
Our Company
Wellness Portal
In this section:
My Account
Charts
Tables
Profile
Reminders
Currently Sign In
Sign Out
Welcome Dr. Soresh Reef 2 N. Lasalle Street, Suite 1904, Chicago, IL - 60602
Wellness Portal>My Account> Table
Tables
Select Table
Blood Pressure
Select Time Range
From
10-24-2002
Calendar
To
11-30-2002
Calendar
Last Weeks Readings
Last Months Readings
View
Sort Order
Ascending
Blood Pressure Readings
Select
Date
Time
Systolic
Diastolic
PPM
Annotation
10-24-2002 07:41:12 PM 143.0 84.0 73.0
10-24-2002 08:55:09 PM 145.0 87.0 72.0
10-25-2002 11:33:51 AM 149.0 87.0 74.0
10-27-2002 09:32:51 AM 152.0 89.0 75.0
10-28-2002 09:25:33 PM 151.0 83.0 73.0
10-29-2002 10:02:03 PM 145.0 80.0 73.0
10-30-2002 11:00:33 PM 141.0 83.0 77.0
10-31-2002 10:12:53 PM 138.0 81.0 74.0
11-01-2002 08:58:03 PM 135.0 80.0 76.0

FIG. 19

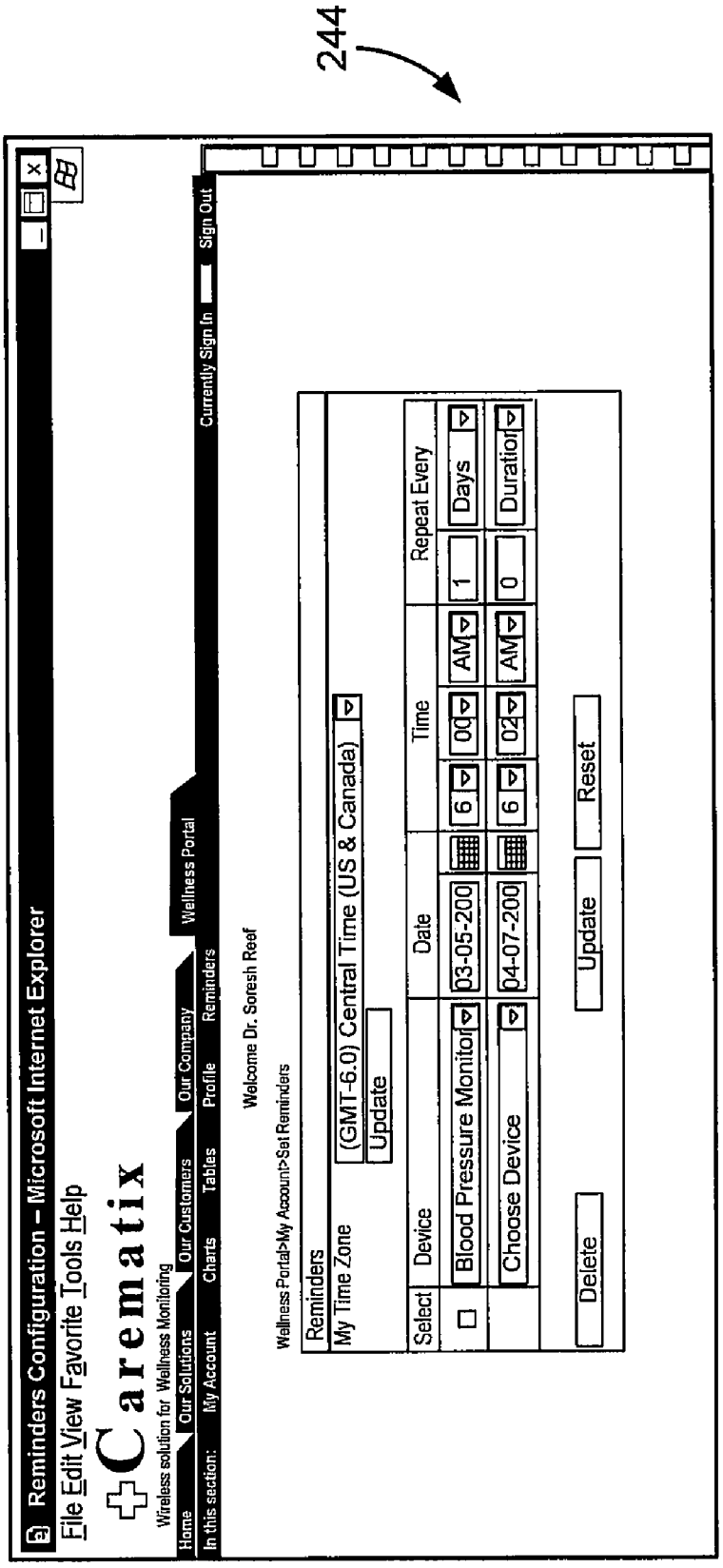

244
Reminders Configuration – Microsoft Internet Explorer
File Edit View Favorite Tools Help
Caremantix
Wireless solution for Wellness Monitoring
Home
In this section:
My Account
Our Solutions
Charts
Our Customers
Tables
Our Company
Profile
Wellness Portal
Reminders
Currently Sign In
Sign Out
Welcome Dr. Soresh Reef
Wellness Portal>My Account>Set Reminders
Reminders
My Time Zone
(GMT-6.0) Central Time (US & Canada)
Update
Select
Device
Date
Time
Repeat Every
Blood Pressure Monitor
03-05-200
6
00
AM
1
Days
Choose Device
04-07-200
6
02
AM
0
Duration
Delete
Update
Reset

FIG. 20

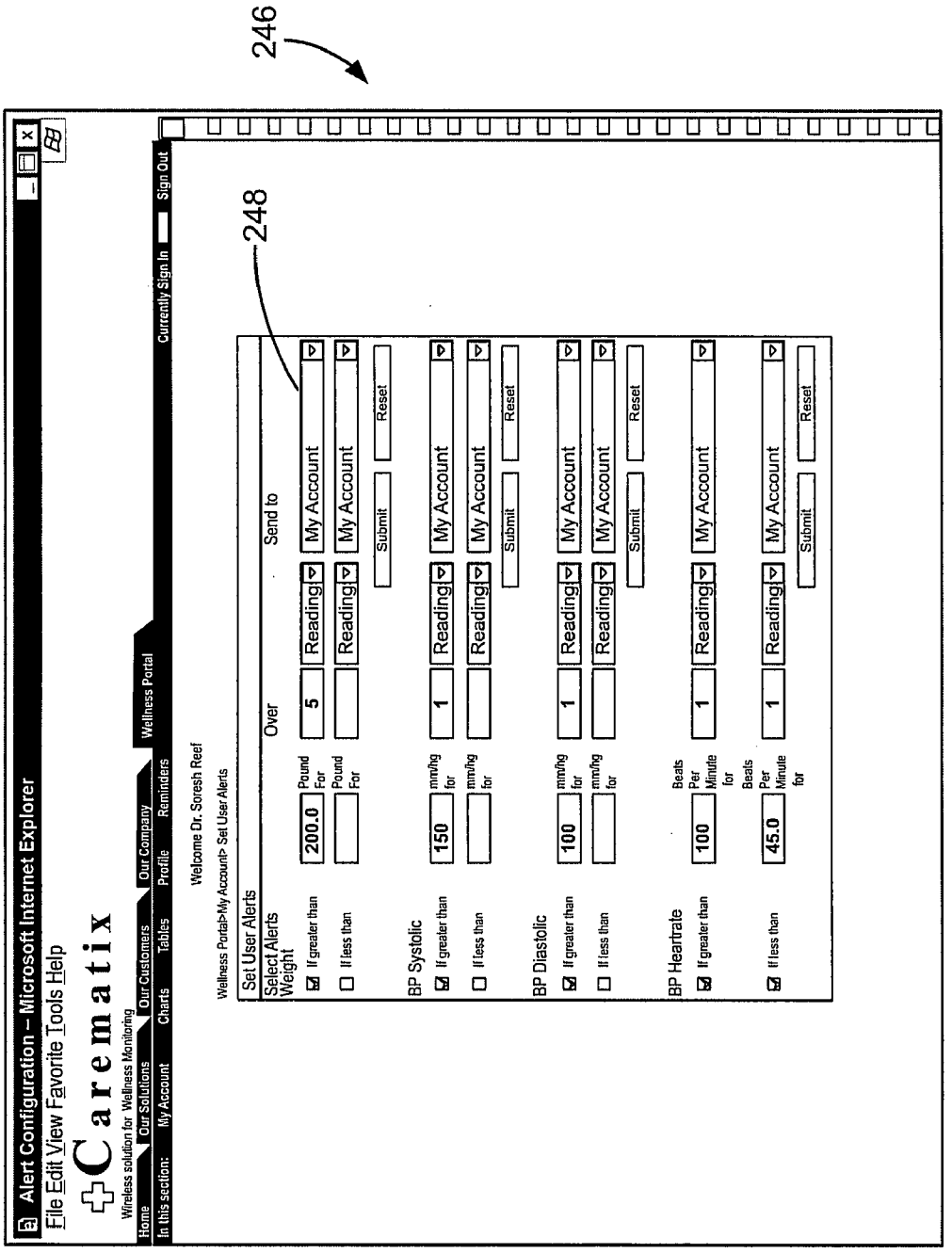

246
248
Alert Configuration – Microsoft Internet Explorer
File Edit View Favorite Tools Help
Carematix
Wireless solution for Wellness Monitoring
Home
Our Solutions
Our Customers
Our Company
Wellness Portal
In this section:
My Account
Charts
Tables
Profile
Reminders
Currently Sign In
Sign Out
Welcome Dr. Soresh Reef
Wellness Portal>My Account> Set User Alerts
Set User Alerts
Select Alerts
Over
Send to
Weight
If greater than
200.0
Pound For
5
Reading
My Account
If less than
Pound For
Reading
My Account
Submit
Reset
BP Systolic
If greater than
150
mm/hg for
1
Reading
My Account
If less than
mm/hg for
Reading
My Account
Submit
Reset
BP Diastolic
If greater than
100
mm/hg for
1
Reading
My Account
If less than
mm/hg for
Reading
My Account
Submit
Reset
BP Heartrate
If greater than
100
Beats Per Minute for
1
Reading
My Account
If less than
45.0
Beats Per Minute for
1
Reading
My Account
Submit
Reset

FIG. 21

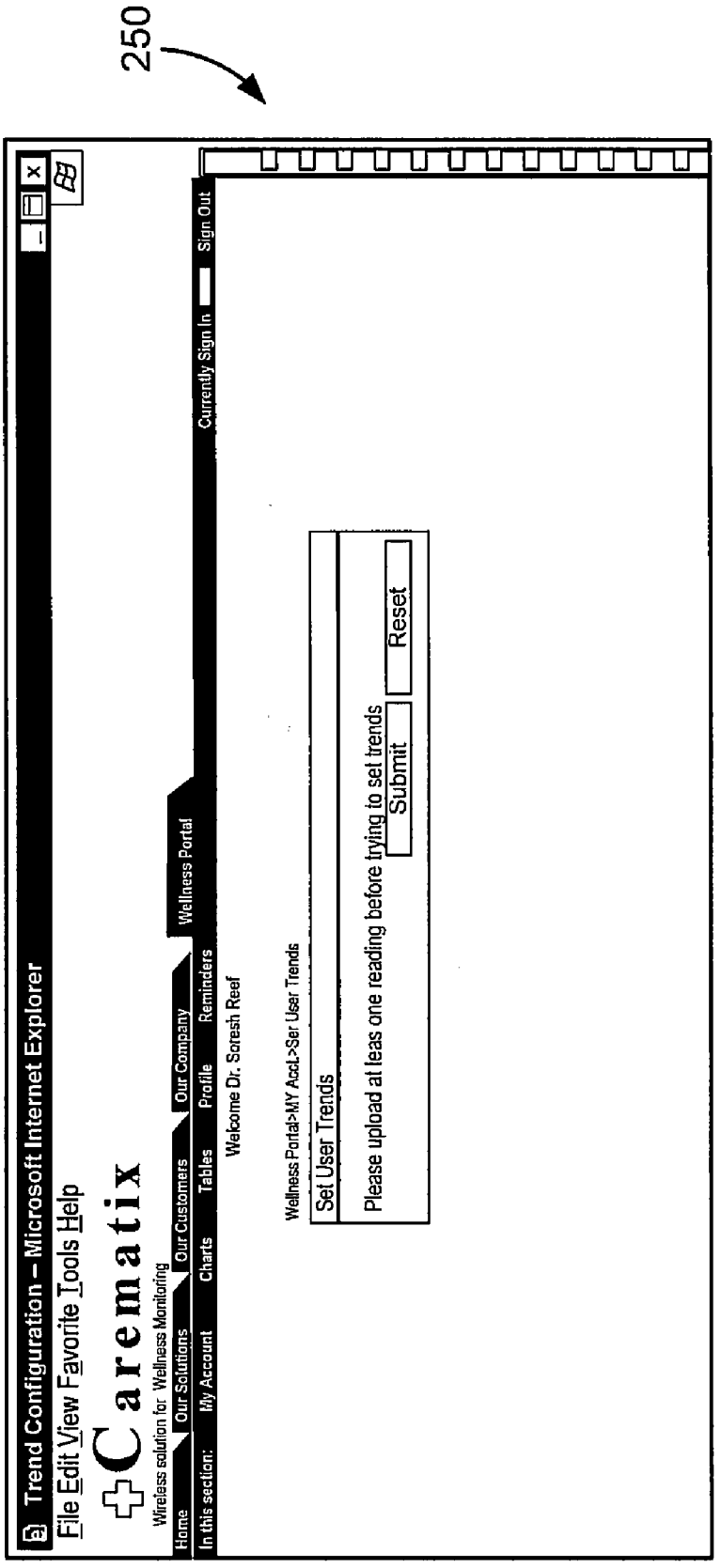

250
Trend Configuration – Microsoft Internet Explorer
File Edit View Favorite Tools Help
Caremantix
Wireless solution for Wellness Monitoring
Currently Sign In
Sign Out
Home
Our Solutions
Our Customers
Our Company
Wellness Portal
In this section:
My Account
Charts
Tables
Profile
Reminders
Welcome Dr. Soresh Reef
Wellness Portal>MY Acct.>Ser User Trends
Set User Trends
Please upload at leas one reading before trying to set trends
Submit
Reset

FIG. 22

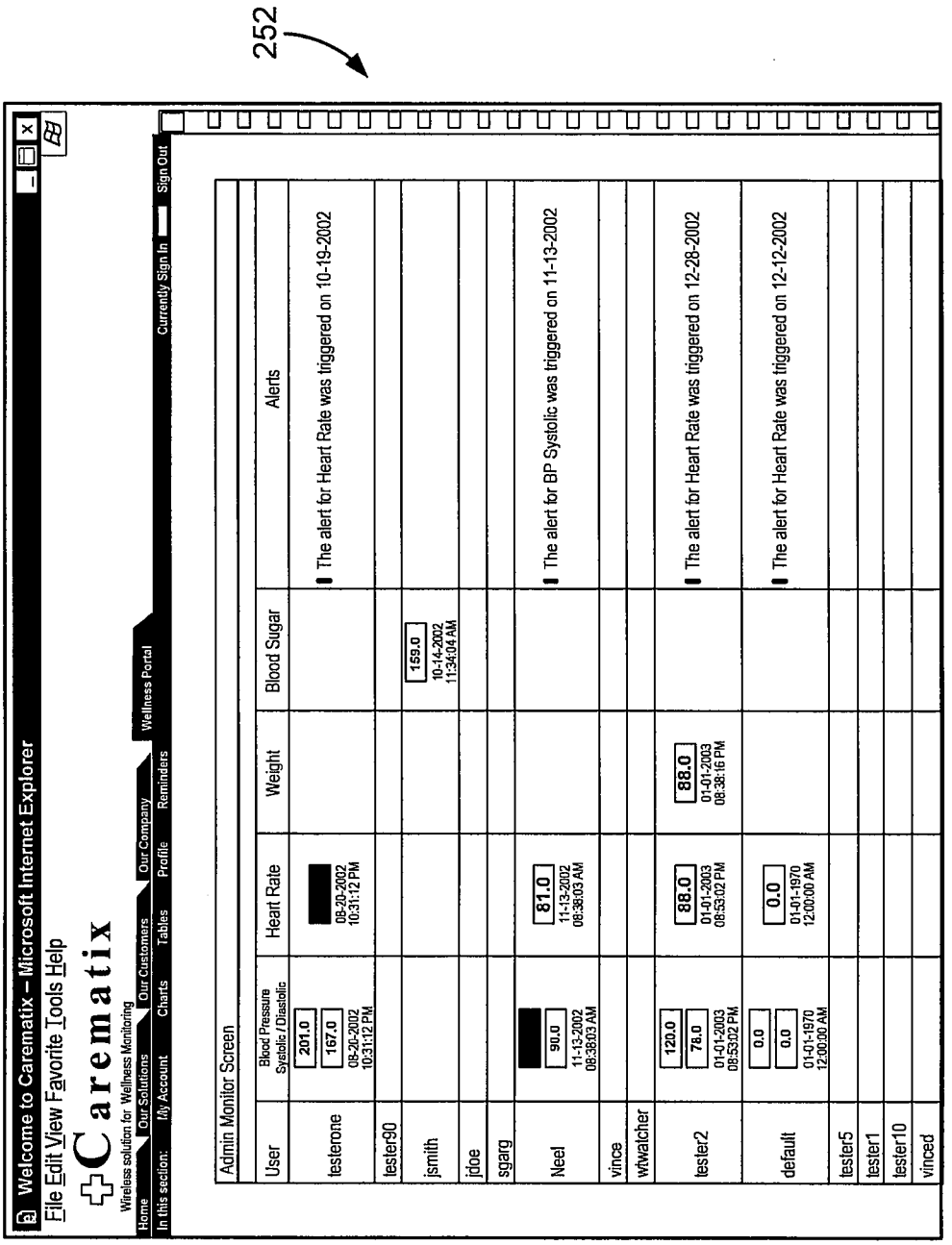

252
Welcome to Carematix – Microsoft Internet Explorer
File Edit View Favorite Tools Help
Carematix
Wireless solution for Wellness Monitoring
Home
Our Solutions
Our Customers
Our Company
Wellness Portal
In this section:
My Account
Charts
Tables
Profile
Reminders
Currently Sign In
Sign Out
Admin Monitor Screen
User
Blood Pressure Systolic / Diastolic
Heart Rate
Weight
Blood Sugar
Alerts
testerone
201.0
167.0
08-20-2002 10:31:12 PM
08-20-2002 10:31:12 PM
The alert for Heart Rate was triggered on 10-19-2002
tester90
jsmith
159.0
10-14-2002 11:34:04 AM
jdoe
sgarg
Neel
90.0
11-13-2002 08:38:03 AM
81.0
11-13-2002 08:38:03 AM
The alert for BP Systolic was triggered on 11-13-2002
vince
wtwatcher
tester2
120.0
78.0
01-01-2003 08:53:02 PM
88.0
01-01-2003 08:53:02 PM
88.0
01-01-2003 08:38:16 PM
The alert for Heart Rate was triggered on 12-28-2002
default
0.0
0.0
01-01-1970 12:00:00 AM
0.0
01-01-1970 12:00:00 AM
The alert for Heart Rate was triggered on 12-12-2002
tester5
tester1
tester10
vinced

FIG. 23

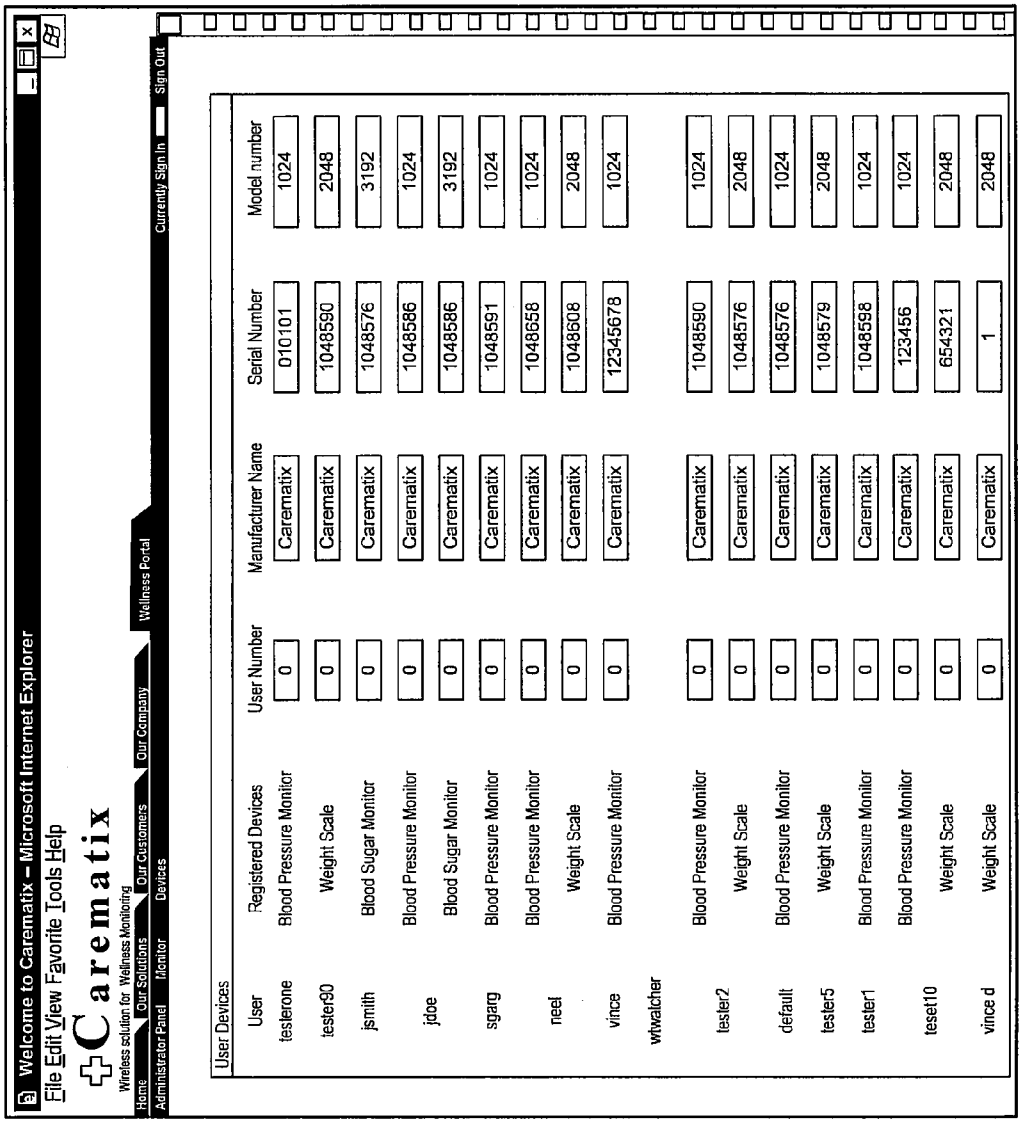

254
Welcome to Carematix – Microsoft Internet Explorer
File Edit View Favorite Tools Help
Carematix
Wireless solution for Wellness Monitoring
Home
Our Solutions
Our Customers
Our Company
Wellness Portal
Administrator Panel
Monitor
Devices
Currently Sign In
Sign Out
User Devices
User
Registered Devices
User Number
Manufacturer Name
Serial Number
Model number
testerone
Blood Pressure Monitor
0
Carematix
010101
1024
tester90
Weight Scale
0
Carematix
1048590
2048
jsmith
Blood Sugar Monitor
0
Carematix
1048576
3192
Blood Pressure Monitor
0
Carematix
1048586
1024
jdoe
Blood Sugar Monitor
0
Carematix
1048586
3192
sgarg
Blood Pressure Monitor
0
Carematix
1048591
1024
Blood Pressure Monitor
0
Carematix
1048658
1024
neel
Weight Scale
0
Carematix
1048608
2048
vince
Blood Pressure Monitor
0
Carematix
12345678
1024
wtwatcher
Blood Pressure Monitor
0
Carematix
1048590
1024
tester2
Weight Scale
0
Carematix
1048576
2048
default
Blood Pressure Monitor
0
Carematix
1048576
1024
tester5
Weight Scale
0
Carematix
1048579
2048
tester1
Blood Pressure Monitor
0
Carematix
1048598
1024
Blood Pressure Monitor
0
Carematix
123456
1024
teset10
Weight Scale
0
Carematix
654321
2048
vince d
Weight Scale
0
Carematix
1
2048

FIG. 24

METHOD AND APPARATUS FOR REMOTELY MONITORING THE CONDITION OF A PATIENT

RELATED APPLICATIONS

[0001] This is a continuation application which claims the benefit and priority of U.S. patent application Ser. No. 10/414,326, filed Apr. 15, 2003, which claimed the benefit of U.S. provisional application No. 60/372,894, filed on Apr. 16, 2002.

BACKGROUND OF THE INVENTION

Computer Listing Appendix

[0002] This application includes a Computer Listing Appendix on compact disc, hereby incorporated by reference.

1. FIELD OF THE INVENTION

[0003] The present invention relates to a patient monitoring system and more particularly to a patient monitoring system for monitoring various physiological characteristic data of a patient, such as blood pressure, pulse rate, blood glucose, weight and others, which wirelessly transmits such data to a hub, located near the patient, which, in turn transfers the data automatically to a remote server, for example, over a public or private communications network, which, in one embodiment, the remote server is configured as a web portal which selectively allows access to patient data by selected third party users, such as physicians, clinicians, patients, and/or relatives, and, in addition provides increased functionality relative to known systems by enabling trends of the patient data to be developed as well as automatically generate communications with the patient or other third parties by way of e-mail or pager when predetermined thresholds for selected physiological characteristics are exceeded and/or to remind the patient to take physiological measurements.

2. DESCRIPTION OF THE PRIOR ART

[0004] Healthcare costs have been increasing at a tremendous rate for the past decade, far exceeding the rate of inflation. The compound average growth rate for healthcare spending over the past decade was 6%, amounting to nearly $1.3 trillion in the year 2001. Chronic disease patients, whose numbers have doubled during the same decade, account for nearly $700 billion of this spending. Managed care organizations have begun to seek help from disease management companies to contain the spending on chronic diseases. Disease management companies thus have developed systems to monitor the chronically ill patients and help lower healthcare spending by improving patient compliance to medication schedules thereby lowering the hospitalization rates.

[0005] Various types of patient monitoring systems are known. For example, U.S. Pat. Nos. 5,810,747 and 5,694,940 disclose patient monitoring systems in which the patient monitoring devices are hard wired to a local hub, which, in turn, is connected to a remote station, for example, over a communication network. The fact that the patient monitors are hard wired to the local station significantly reduces the utility of such systems. For example, such systems are obviously not suitable for ambulatory patients and many applications where it may be desired to remotely monitor the physiological characteristics of a patient outside of a non-clinical environment.

[0006] Accordingly, various systems have been developed in which the patient monitoring device, normally worn by a patient, is connected by way of a wireless link to a local hub, which, in turn, is connected to a remote station or server by way of a public communication network, such as the Plain Old Telephone System (POTS). Examples of such systems are disclosed in U.S. Pat. Nos. 3,882,277; 5,522,396; and 6,093,146. Each of these systems include a patient monitoring device, connected to a local hub by way of a wireless link, which, in turn, is connected to a remote server by way of a public communication link, such as POTS. Such systems depend on the use of telephone modems which require patients to place a phone receiver into a modem cradle and dial up the remote server. Such systems have been found to be far too complicated and difficult for elderly and critically ill patients.

[0007] In order to resolve this problem, patient monitoring systems, for example, as disclosed in U.S. Pat. No. 6,336,900, have been developed, which not only provide a wireless link between the patient monitor and the local hub, but also automatically dial and connect to the remote server to facilitate the transfer of the physiological data to the remote server. Unfortunately, the functional capability of such systems is relatively limited. For example, such systems only provide limited access to the patient data. In addition, such systems can not be used to provide reminders to patients to take readings or provide messages to the patients or third parties when the physiological characteristics of a patient exceed predetermined thresholds. Thus, there is a need for a patient monitoring system for monitoring the physiological characteristics of a patient that provides enhanced functionality and expanded access to the healthcare data while not tethering the patient to a local hub, thereby improving patient compliance, affording the healthcare provider the ability to capture adverse events sooner and avoid costly emergency care and reduce costly home health visits.

SUMMARY OF THE INVENTION

[0008] Briefly, the present invention relates to a patient monitoring system which provides enhanced functional capability relative to known systems and provides a wireless communication link between a patient monitoring device, worn by a patient, and a local hub. The patient monitoring system in accordance with the present invention is adapted to monitor various patient physiological characteristics, such as blood pressure, pulse rate, blood glucose, weight, pulse oximetry and others. The data from the patient monitoring device is wirelessly transmitted to a local hub, which, in turn, is configured to automatically transfer the data to a remote server, for example, over a public or private communications network. In one embodiment of the invention, the server is configured as a web portal to selectively allow access to such patient physiological data by designated third parties, such as physicians, clinicians, relatives and the patient themselves. In accordance with another important aspect of the invention, the system provides for enhanced functionality relative to known systems and allows trends of the physiological data to be selectively generated. In addition, any third party can set thresholds to automatically notify the patient or other third parties by various communication methods including e-mail and/or pager when a particular physiological characteristic

exceeds a predetermined threshold. The system is also configured to provide reminders to patients to take readings.

DESCRIPTION OF THE DRAWINGS

[0009] These and other advantages of the present invention will be readily understood from the following specification and attached drawing wherein:

[0010] FIG. **1** is a block diagram of the patient monitoring system in accordance with the present invention.

[0011] FIG. **2** is a block diagram of an alternate embodiment of the invention, illustrated in FIG. **1**, illustrating the use of the invention in a multi-user, multi-hub environment.

[0012] FIG. **3** is a block diagram of an alternate embodiment of the invention, illustrated in FIG. **1**, for a multi-user environment, single hub environment.

[0013] FIG. **4** is a block diagram of another alternate embodiment of the invention illustrated in FIG. **1** for a multi-user, single hub environment.

[0014] FIG. **5** is a block diagram of an exemplary physiological transducer for relatively complex transducers such as an electrocardiograph for use with the present invention.

[0015] FIG. **6** is a block diagram of exemplary self contained hub or base station for use with the present invention.

[0016] FIG. **7** is an exemplary schematic diagram of a relatively simple physiological transducer such as a blood pressure monitor, for use with the present invention.

[0017] FIG. **8** is an exemplary schematic diagram of an alternative hub or base station that is adopted to be connected to a middleware device, for use with the present invention.

[0018] FIG. **9**A is a schematic diagram of the hub illustrated in FIG. **6**.

[0019] FIG. **9**B is a schematic diagram of the transceiver incorporated into the hub illustrated in FIG. **9**A.

[0020] FIG. **10** is a schematic diagram of the exemplary physiological transducer illustrated in FIG. **7**.

[0021] FIG. **11** is an exemplary block diagram of the server application architecture in accordance the present invention.

[0022] FIG. **12** is a diagram illustrating an exemplary message exchange between a patient monitoring device and a hub for the system in accordance with the present invention.

[0023] FIG. **13** is an exemplary software flow diagram for synchronizing the patient monitoring device to a hub in accordance with the present invention.

[0024] FIGS. **14** and **15** are exemplary flow diagrams illustrating the hub to web server synchronization.

[0025] FIG. **16** is an exemplary flow diagram illustrating the data server/portal to user interaction in accordance with the present invention.

[0026] FIGS. **17-24** are exemplary screen shots for use with a web-based embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0027] The present invention relates to a patient monitoring system for monitoring various physiological characteristics of a patient, such as blood pressure, pulse rate and others, as discussed below. The system includes a portable physiological transducer or patient monitoring device that can be worn by a patient in both single user and multi-user applications. The patient monitoring device is adapted to monitor physiological characteristic data of the patient and wirelessly transmit the data to a local hub or base station. In order to reduce the complexity of the system, the hub or base station is configured to automatically transfer the data to a remote server by way of a public or private communication network. In accordance with an important aspect of the invention, the system is configured to selectively enable access to a patients' physiological data by various selectable third party users, as well as the patient. In accordance with another aspect of the present invention, the patient monitoring system provides enhanced functionality relative to known systems. For example, the system in accordance with the present invention enables selectable third party users, as well as the patient, to monitor trends in the data for the physiological characteristics, as well as set alarms, which can automatically generate communications with the patient and/or third parties when a physiological characteristic exceeds a predetermined value. These communications can be, for example, by e-mail or pager. In one embodiment of the invention, a bi-directional communication link is provided between the patient monitoring device and the local hub. This bi-directional communication link allows communications to be sent to the patient monitoring device, for example, to remind a patient to take a physiological reading. The bi-directional communication link also allows handshaking between the patient monitoring device and the local hub to insure the transfer of physiological data to the local hub.

[0028] As will be discussed in more detail below, the system in accordance with the present invention can be utilized in a wide variety of applications. For example, FIG. **1** illustrates a single user/single hub application. FIG. **2** illustrates a multiple user/multiple hub application. FIGS. **3** and **4** illustrate a multiple user/single hub environment, for example, a cardiac rehabilitation facility in which each of the patients wear a portable heart monitor during cardiac rehabilitation.

[0029] FIGS. **5-10** illustrate exemplary hardware diagrams for the portable physiological transducer and the hub for use with the present invention. In particular, FIG. **5** illustrates an exemplary block diagram of a portable physiological transducer for relatively complex transducers, such as a electrocardiograph. FIG. **7** illustrates an exemplary block diagram of a relatively simple portable physiological transducer, for example, a blood pressure transducer. Both transducers illustrated in FIGS. **5** and **7** may include an audio or visual indicating device to enable reminder messages to be sent to the patient monitoring device. FIG. **6** illustrates a block diagram of a exemplary self contained hub or base station for use with the present invention. FIG. **8** illustrates a block diagram of an exemplary hub that is adapted to be connected to a middleware device. FIG. **9**A is a schematic diagram of a self contained hub illustrated in FIG. **6**. FIG. **9**B is a schematic diagram of the transceiver incorporated into the hub, illustrated in FIG. **9**A. FIG. **10** illustrates an exemplary schematic diagram for a relatively simple patient monitoring device, such as a blood pressure monitor.

[0030] FIGS. **11-16** relate to the software. FIG. **11** is a block diagram of the server application architecture. FIG. **12** is an exemplary block diagram illustrating an exemplary message exchange between a portable physiological transducer or patient monitoring device and the self contained hub in accordance with the present invention. FIG. **13** is an exemplary software flow diagram illustrating the synchronization of the patient monitoring device with the hub in accordance with the present invention. FIGS. **14** and **15** illustrate the synchronization of the hub to the web server. FIG. **16** illustrates an exemplary flow diagram illustrating the data server to user interaction in accordance with the present invention.

[0031] In one exemplary embodiment of the invention, the system is configured as a web portal. In this embodiment, the server is configured as a web server and selectively allows third party access as well as access by the patient to the physiological data transmitted by the portable patient monitoring device. In addition, the system can be used to provide aural and/or visual signals to the patient monitoring device to remind a patient to take readings. As discussed above and as will be discussed in more detail below, various third parties, such as the physician and/or relatives, as well as the patient, can not only access the physiological data of the patient but also view trends of such data and also set alarms which automatically generate messages to either the patient and/or other third parties when the physiological characteristics of the patient exceed a preset value. In accordance with this embodiment, various screen shots as illustrated in FIGS. **17-23** are provided.

[0032] System

[0033] FIGS. **1-4** illustrate various system applications for the patient monitoring system in accordance with the present invention. In particular, single user/single hub applications, as well as multiple user/single hub and multiple user/multiple hub applications are within the broad scope of the present invention. As such, the patient monitoring system in accordance with the present invention can be used in a relatively wide variety of applications from non-clinical applications, such as a home healthcare and health club applications, as well as various clinical applications.

[0034] Referring first to FIG. **1**, an embodiment of a patient monitoring system is illustrated and generally identified with the reference numeral **30**. The patient monitoring system **30** illustrates a single user/single hub application, for example, a home healthcare application. In this application, the patient monitoring system **30** includes a portable physiological monitor or patient monitoring device, identified with the reference numeral **32**, a local hub **34**, a middleware device **36**, which may or may not be incorporated into the hub **34**, as well as a remote server **38**. The patient monitoring device **32** can be one of multiple portable physiological transducer, such as a blood pressure monitor, heart rate monitor, weight scale, thermometer, spirometer, single or multiple lead electrocardiograph (ECG), a pulse oxymeter, a body fat monitor, a cholesterol monitor or a signal from an exercise machine, such as a heart rate. As will be discussed in more detail below, the patient monitoring device **32** is a portable device worn by the patient and includes a single or bi-directional wireless communication link, generally identified with the reference numeral **40**, for transmitting data from the patient monitoring device **32** to the local hub or receiving station **34** by way of a wireless radio frequency (RF) link using a proprietary or non-proprietary protocol.

[0035] In this embodiment of the invention, the system includes a local hub or receiving station **34** which wirelessly receives data from the patient monitoring device **32** and automatically communicates it to a middleware device **36**, which may be a personal computer, an Internet gateway, a home gateway, a phone, a video phone, or a phone modem. Alternatively, the middleware device **36** may not be required if the hub **34** includes its own processing equipment, as will be discussed below. In particular, the middleware device **36** may form part of the hub **34**.

[0036] The hub or receiving station **34** may be a receiver or transceiver for receiving data from the patient monitoring device **32**. Communication, if required, between the hub **34** and the middleware device **36**, may be over various communication links, such as a direct connection, such a serial connection, USB connection, Firewire connection or may be optically based, such as infrared or wireless based, for example, home RF, IEEE standard 802.11a/b, Bluetooth or the like.

[0037] The middleware device **36** transfers the data received by the hub or receiving station **34** to a remote server **38**. The communication link between the middleware device **36** and the remote server may be by direct connection, such as a serial connection, USB, Firewire or optically based such as infrared or wireless based such as home RF, IEEE standard 802.11b, Bluetooth or others. For longer distances, a communication link between the middleware device **36** and a remote server **38** may be by DSL, T-1 connection over a private communication network or a public information network, such as the Internet.

[0038] In accordance with one aspect of the invention, the remote server **38** may be configured to provide third party access to the patient physiological data forming a portal. The portal, for example, a web portal, allows the patient and/or third party users, such as relatives and physicians, to interact with the patient data in various ways as discussed below.

[0039] FIG. **2** illustrates a multi-user/multi-hub application of the present invention, generally identified with the reference numeral **42**. In this embodiment, three (3) exemplary patient monitoring devices **44**, **46** and **48** are used to transmit data to multiple hubs **50** and **52**. The hubs **50** and **52** transfer the data to middleware devices **54** and **56**, which, as discussed above may be a personal computer (PC) **54** or a phone line, web phone or video phone **56** or may be integrated into the hubs **50** and **52**. In this embodiment, the middleware devices **54** and **56**, in turn, transmit the data to a remote server **58**. In accordance with an important aspect of the invention, various third parties, such as a physician or relatives, can selectively access the patient physiological data in the server **58**, as indicated by the function block **60**.

[0040] FIGS. **3** and **4** illustrate a multi-user/single hub application, for use in multiple user applications, such as a retirement home. Two embodiments of this application are contemplated as illustrated in FIGS. **3** and **4**. Referring first to FIG. **3**, the patient monitoring system is generally identified with the reference numeral **62**. The patient monitoring system **62** includes a patient monitoring device **64**, a hub **66**, an ID reader **68**, a middleware device **70** and a remote server **72**. The patient monitoring device **64**, hub **66**, middleware device **70** and the server **72**, as well as the communication links therebetween, are as described above. In the exemplary application illustrated in FIG. **3**, the ID reader **68** is operatively connected to the hub **66** to accommodate different users in a multiple user environment. In such an application, each user is given a unique ID. This ID is input into the hub **66**. Various types of identification are contemplated. For example, the ID reader **68** may be a swipe card, RF tag or push button. In this embodiment, the user ID is read from the ID tag of the user at the same time physiological data is being read from the patient monitoring device **64**.

[0041] FIG. **4** illustrates an alternative embodiment of the invention illustrated in FIG. **3**. This embodiment is generally identified with the reference numeral **74**. The patient monitoring system **74** includes a patient monitoring device **76**, a hub **78**, a server **82** and an ID reader **84**. The patient monitoring system **74** is similar to the patient monitoring system **62**, illustrated in FIG. **3**, except that the patient monitoring

system **74** is configured such that the ID reader **84** is coupled to the patient monitoring device **76**. In this embodiment, the user ID and physiological data is transmitted together to the hub **78**. In lieu of the ID reader, the patient monitoring device **76** can be configured such that the device transmits an unique ID with the physiological data. Thus, different users wearing different devices can easily be correlated by the system.

[0042] Hardware

[0043] The hardware for the patient monitoring device, as well as the hub, is illustrated in FIGS. **5-10**. As mentioned above, FIGS. **5** and **7** relate to a patient monitoring devices, generally identified with the reference numerals **84** and **85**, for use with the present invention. FIGS. **6** and **8** illustrate exemplary hubs for use with the present invention. FIG. **10** is an exemplary schematic diagram of the patient monitoring device **84**, illustrated in FIG. **5**.

[0044] Exemplary Patient Monitoring Devices

[0045] Various embodiments of the patient monitoring devices **84** and **85** are contemplated. Both embodiments may optionally include a visual or audio indicating device to allow visual and/or audio communications to be sent from the system to the patient monitoring device. The patient monitoring device **85** (FIG. **7**) is representative of patient monitoring devices which utilize relatively simple transducers, such as a blood pressure monitor, heart rate monitor and others. The patient monitoring device **85** includes a transducer **88**, such as a blood pressure transducer, coupled to a communication module **90**. The communication module **90** may include a microprocessor **92**, a memory real time clock **94**, as well as a transmitter for unidirectional communications or a transceiver **96** which enables bi-directional communication between the patient monitoring device **85** and a local hub, as discussed above. An audio or visual indicating device **93** may also be provided to enable alerts or messages to be sent to the patient monitoring device **85**. The transducer **88** may be directly wired to the communication module **90**, as indicated by the electrical leads **98**. An antenna **100** is provided in order to provide a wireless link to the local hub, as discussed above. A power supply **102**, such as a battery may be provided.

[0046] The MCU **92** may be, for example, Motorola model number 68HC908GP32. The radio **96** may be, for example, a Xemics model number XE 1201. The memory **94** may be, for example, an Atmelmodel number CAT 24AA65. The real time clock may be a Dallas Semiconductor model number DS 1675.

[0047] More sophisticated patient monitoring devices **84**, as illustrated in FIG. **5**, such as electrocardiograph (ECG) devices, may include one or more transducers which directly communicate with a microprocessor **106**. An input/output (I/O) device **108** may be directly coupled to the microprocessor **106**, for example, for receiving data from electrodes attached to the patients' body by way of electrical leads (not shown). As is well known in the art, this data is received by the microprocessor **106** and configured by the transducer **104** and communicated to a communication module **110**. The communication module **110** includes its own microprocessor **112**, a memory **114** such as an electrically erasable programmable read only memory (EEPROM), a radio **114** for bi-directional communication or simply a transmitter for unidirectional communication, as well as an antenna **116**.

[0048] Various physiological transducers can be integrated into the system. These transducers are similar to various commercially available transducers such as: a blood pressure transducer, such as A&D blood pressure cuff, model number UA-767PC; a weight scale, such as an A&D scale, model number UC-321; a blood glucose meter, such as a LifeScan ONE TOUCH.RTM. Ultra; a pulse oximeter, such as a Palm-Sat Model 2500; a spirometer, such as a Micro Direct model MICRO DL; a prothromben time test for Coumadin.RTM. therapy, such as a PT/NR Protime Microcoagulation System; and a cholesterol monitor, such as a LSP 3101 Personal Cholesterol Monitor Kit. Virtually any physiological transducer which generates an electrical signal can be implemented in the system.

[0049] Hub Hardware

[0050] Exemplary block diagrams for the hub for use with the present invention is illustrated in FIGS. **6** and **8**. Referring first to FIG. **8**, the hub **87** includes a radio **116**, a microprocessor **118** and a memory/real time clock **120**. The hub **87** also includes and an antenna **120** and a power supply **122** if required. This embodiment of the hub **87** is for an embodiment in which there is a separate communication link as discussed above to a middleware device **122** for embodiments in which the middleware device **122** is separated from the hub **87**.

[0051] FIG. **6** illustrates an alternative embodiment of the hub, identified with the reference numeral **86** in which the middleware device is integrally formed with the hub. In this embodiment, the hub **86** includes a radio **126** and an antenna **128** for receiving data, wirelessly from the patient monitoring device. The hub **86** also includes a microprocessor **130**, a real time clock **132**, a memory device such as a EEPROM **134** and a modem **136**. The microprocessor **130**, radio **126**, real time clock **132** and memory **134** may be as discussed above. The modem may be a ConnectOne embedded iModem, Model Number iM336UC-EM.

[0052] FIG. **9**A is an exemplary schematic diagram of the hub **86** illustrated in FIG. **6**. FIG. **9**B is an exemplary schematic diagram of the transceiver **126** illustrated in FIG. **9**A.

[0053] Software

[0054] FIGS. **11-16** illustrate the system software. FIG. **11** illustrates the server application architecture. FIG. **12** illustrates the message exchange between a patient monitoring device and the hub. FIG. **13** illustrates the software for synchronizing data from the patient monitoring device to the hub. FIG. **14** illustrates the synchronization between the hub and the remote server and specifically relates to user registration. FIG. **15** illustrates the software flow diagram for interaction between the hub and the web server. FIG. **16** illustrates the software flow diagram for interaction between the user and data server/portal.

[0055] Referring first to FIGS. **13-16**, FIG. **13** illustrates the software for controlling the patient monitoring device and transmitting the data from the patient monitoring device to the hub. Initially, in step **138**, on power up, the patient monitoring device, either on its own or in response to patient input, begins taking readings of a patient physiological characteristic. These readings are stored in the patient monitoring device along with a time stamp and device ID in step **140**. In step **142**, the patient monitoring device sends out RF signals with its ID in an attempt to locate a local hub. If no hub responds as indicated in step **144**, the patient monitoring device stores the data and goes to sleep in step **146**. Subsequently, after a preset time, the device awakens in step **148** and returns to step **142** and attempts to locate a hub.

[0056] Once a hub is located, the hub responds with a handshake signal indicating that it is available to receive data as indicated in step **150**. After the handshake, the hub may

communicate various information to the patient monitoring device in step **152**. For example, the hub may send a message to the patient monitoring device indicating that previous readings have been received. The hub may also send a message to the patient monitoring device with the current date and time to update that device. In addition, the hub may send reminders at specified dates and times to remind a patient when to take readings for those patient physiological devices which do not constantly take readings. Finally, the hub may also send a message to the patient monitoring device to change the frequency or baud rate of the communication. Subsequently, in step **154**, the patient monitoring device synchronizes its date and time stamp and deletes all previous readings to the date and time in order to initialize the patient monitoring device. After the patient monitoring device is initialized in step **154**, the patient monitoring device then sends the data to the hub in step **156** with its associated ID and date and time. The patient monitoring device may also send data to the hub indicating that its battery is low. In step **158**, the hub acknowledges receipt of each message received.

[0057] The message exchange between the patient monitoring device and the hub is illustrated diagrammatically in FIG. **12**. Initially, as indicated in box **160**, the patient monitoring device on power up sends a query to the hub, as indicated by the message **162**. The hub in response to that query checks whether the ID number for the patient monitoring device is listed, as indicated by the block **164**. In addition, the hub may also communicate its local date and time as well as control information with respect to clearing the log and frequency change, as message **164**. This information is acknowledged and acted upon by the patient monitoring device, as indicated by the message **168**. After the patient monitoring device is time synchronized with the hub, the patient monitoring device begins sending data to the hub along with its ID and time stamp, as indicated by the message **170**.

[0058] The above message exchange may be identified as a pre-cycle message exchange. Such a pre-cycle message exchange is identified as a time period when the patient monitoring transducer is not actively recording data such as during a time period when a blood pressure cuff has not been inflated.

[0059] After the data has been transmitted from the patient monitoring device to the hub, post cycle messages include repeating the messages **162** and **166** after each data cycle has been completed. In addition, a message **172** may be sent from the patient monitoring device to the hub for the first piece of data, indicated by the box **174**. After the hub receives the data **174**, it sends a message back to the patient monitoring device indicating that the data **174** has been received. Once the patient monitoring device acknowledges from the hub that the first bit of data has been received, additional data is sent by way of a message **178** to the hub, as indicated by the data block **180**. The hub responds that the data **180** was received by way of a message **182** back to the patient monitoring device. After all of the data has been transmitted from the patient monitoring device to the hub, the patient monitoring device sends a message **184** indicating that the next bit of data is the last bit of data in the memory to be sent by the patient monitoring device by way of the message **184**. The hub then acknowledges receiving the last bit of data **186** by acknowledging by way of message **188** back to the patient monitoring device **188** that it has received that data **186**.

[0060] FIGS. **14** and **15** illustrate the synchronization between the hub and the remote server. In accordance with an important aspect of the invention, the server in this embodiment is configured as a portal, for example, a web portal, which enables third party users to selectively access patient physiological data transferred to the remote server. In this application, users register on the portal and create an account in step **190**. In step **192**, the user registers one or more devices (i.e. device ID's) with their account. Next, in step **194**, the list of registered devices for a particular user is synchronized with the hub thereby enabling the hub to receive data for the specified device IDs. In step **196**, the hub filters wireless communications from various patient monitor devices and only responds to data originating from registered devices. In step **198**, any updates of the registered devices with respect to a user's account are synchronized with the hub.

[0061] FIG. **15** illustrates a software diagram for transferring data between the hub and the remote server. Initially, in step **200**, data is received by the hub as discussed above. In step **202**, the data may be stored, for example, in XML format, for example, and optionally encrypted with readings with unsynchronized time stamps, which may be corrected to best estimates. In step **204**, if the hub is not connected to the server, the server looks for an Internet connection or creates a connection over a LAN/phone line automatically and sends the data to the web server when connection is established. In step **206**, data is uploaded to the web server and associated to a device ID and user. The web server responds in step **208** by sending messages to synchronize the data including a time and date stamp of last received data (clear log); a list of date and time stamps for reminders; and may include a down load list of device IDs associated with that hub.

[0062] FIG. **16** is a software flow diagram used to control the data server/portal in connection with user interaction. Initially, in step **210**, data is received by the data server from the hub. In step **212**, the data may be decrypted and staged for acceptance by the user after a user logs in. Subsequently, in step **214**, the data is cataloged in a user data repository or database to enable the user to interact with the data in meaningful ways like display charts, tables and trigger alarms and trends. As indicated in step **216**, the data may be checked against preset requirements for alerts and trends. These alerts and trends may be shown to the user in step **218** in various ways on a web page, a pager, a telephone, an e-mail to a selected list of people to be informed. The web server may then display the charts and tables in step **220**. The reports and tables can be exported to spreadsheets and/or e-mailed to third parties in step **222** for review. In step **224**, the server allows additional third party access for the various data in step **224**.

[0063] As indicated in step **226**, when connection, such as Internet connection, is established between the hub and the server, the server sends the hub various messages. These messages may include the current device ID list for the user to the hub. The messages may also include date and time stamps for reminders and the date/time stamp of the last updated reading.

[0064] Web Pages

[0065] In a exemplary embodiment, as discussed in detail below, the remote server may be configured as a web portal. Exemplary web pages for a patient monitoring system are illustrated in FIGS. **17-24**. In particular, FIG. **17** illustrates exemplary home page generally identified with the reference numeral **228**. This web page **228** includes a registration box **230** and an account box **232**. The registration box **230** allows patients to register and unregister various physiological characteristic measurements. The account box **232** provides a

summary of a patients' alert and trends. The account box **232** includes links **234** and **236** to screens where alerts and trends can be set.

[0066] FIG. **18** illustrates a exemplary web page **238** which allows a user, either a patient or a third party user, to track data by way of a display window **240**. The exemplary web page **238** allows a user to track data, view it by device type, change the time scale as well as annotate the data. FIG. **19** is an exemplary web page which allows a user to view data in tabular form. On this page, the time range and device data can be selected by the user. FIG. **20** is an exemplary web page **244** which enables a user to set reminders for the device to provide reminders, either aurally or visually at certain times or at periodic intervals. FIG. **21** is an exemplary web page **246** which allows a user to set alert parameters on absolute values and to specify how to receive the alerts via browser, e-mail, voicemail, pager, etc. In particular, "My Account" boxes generally identified with the reference numeral **248**, allow the particular message link to be selected. FIG. **22** illustrates an exemplary web page **250**. The web page **250** allows a user to set up parameters, baselines and trends. FIG. **23** illustrates an exemplary web page **252** which may be used by a clinical administrator to monitor a summary status of all assigned patients including the last reading and any alerts generated. FIG. **24** illustrates an exemplary web page **254** for use by an administrator to view the devices registered by various users in the system.

[0067] Server Application Architecture

[0068] FIG. **11** illustrates the server application architecture. The server application architecture may include four levels or tiers **228**, **230**, **232**, **234**. The tier **228** may be a user or client tier which allows users to interact with the underlying application. This client tier translates user actions and inputs them as server requests and formats a server response, usually visual. The client tier may be a client, such as a browser, or a custom application running on a hub or specific devices such as wired, wireless handheld Personal Digital Assistants (PDAs). The client tier may also be a non-user action, such as automated services, such as Internet Explorer.

[0069] The web tier **230** makes application functionality available on the worldwide web. This tier **230** accesses data and business functionalities, manages screen flow and often encapsulates some user interaction. It may be used to decouple the client from the business logic tier **232** to provide a uniform service model of client request. The web tier interacts with other tiers using standard protocols. Other applications may take place of the client programs accessing the underlying application through its web tie, such as Apache/Tomcat.

[0070] The business logic tier **232** encapsulates the business logic and comprises the software application functionality. The logic is organized as a set of interacting objects allowing for additions and enhancement of the logic at any point while still preserving the interface of other tiers. This business logic tier **232** may be implemented using J2EE, Microsoft.NET or a custom architecture, such as Tomcat.

[0071] The data access tier **234** may be used to integrate the application with other enterprise information systems. This tier **234** provides data storage and other information services to the application. It may consist of a database, enterprise resource planning systems, mainframe transaction processors, legacy systems and enterprise integration technologies. Other tiers access these databases using industry standard or custom protocols such as Oracle.

[0072] Web Portal Details

[0073] As mentioned above, in one exemplary embodiment of the invention, a web portal (hereafter referred to as the portal) provides meaningful access to a patient's personal health data, as collected by the patient monitoring device(s) and stored in a server-based data repository (hereafter referred to as the repository). Some key mechanisms of the portal are listed below and are explained in further detail in subsequent sections. These key mechanisms include:

[0074] A mechanism for self-registration of devices by the user.

[0075] A mechanism to synchronize data between the Repository and device(s) via the Receiving Station.

[0076] A mechanism to allow the user to preview the readings received from the device(s) and reject erroneous readings.

[0077] A mechanism to generate alerts against received data, based on user defined parameters.

[0078] A mechanism to determine trends in received data, based on user defined parameters.

[0079] A mechanism to match computed trends with visual perception of the trend.

[0080] A mechanism to set reminders on the device(s). This mechanism can be extended to set up any arbitrary "event" on the device(s) e.g. local alert generation.

[0081] A mechanism by which a clinician may monitor a group or a number of users via a summary "dashboard" of their readings data, with ability to drill-down into details for each user.

[0082] A mechanism to enhance user (healthcare) education by providing a healthcare questionnaire dynamically created (and updated) based upon certain rules applied to recent user readings data.

[0083] A mechanism to provide data feeds (i.e. export, either "raw" or aggregated) to external systems with transformations between appropriate data formats.

[0084] Device Registration Through Portal

[0085] Device registration is the process by which a patient monitoring device is associated with one or more users of the system. This mechanism is also used when provisioning devices for a user by a third party, such as a clinician (or their respective delegate). The mechanism is as follows:

[0086] 1. The user (or delegate) identifies himself/herself by logging into the portal. The user must have previously registered an account created either by themselves or by delegates.

[0087] 2. User (or delegate) registers a device by:

[0088] a. Choosing from a list of kinds of devices available for registration.

[0089] b. Providing identification information for the device such as manufacturer information, device model information, device serial number (available on device packaging) and optionally a hub number (available on hub packaging). The user may register more than one device at this point.

[0090] c. Optionally setting up a service subscription for device(s) usage. This includes selecting service plans and providing payment information.

[0091] 3. The device(s) are then associated with this user's account and a Device Control File (discussed below) comprised of device identification information is synchronized between the server and the Receiving Station on initialization (see the section Data Synchronization between Receiving Station and Repository). This enables the Receiving Station

to respond to and accept input from the devices that have their identification information in the control file.

[0092] 4. On each interaction between the hub and the server, the control file entries for the current user (or hub) are synchronized.

[0093] 5. Similarly, whenever the user registers a device, its entry is added to the control file on the hub if not already there on the next synchronization interaction with the server.

[0094] 6. No association information between the actual user (person) and the device is kept on the user's hub. This determination is made after the data is uploaded to the server.

[0095] 7. The hub may be shared between users and data for all users resident on the hub is uploaded to the server on synchronization.

[0096] Device Control File

[0097] This file contains records in the format

[0098] <device id>, <data update timestamp>

[0099] where <device id> is all the information required to uniquely identify a device to the system (such as manufacturer number, model number, serial number and user number) and the <data update timestamp> is the timestamp (including date) that data for this device was last received by the server. Note that this is not the timestamp for which data was last uploaded from this particular hub (on which the control file resides) but is the timestamp for which the server last received data (from any hub) for the particular (registered) device. Additional information for controlling the device(s) e.g. reminder information is also maintained in this file.

[0100] This file is internally maintained in XML format for ease of interfacing but is either kept encrypted or in a non-readable format on the hub for security reasons.

[0101] User Data File

[0102] This file contains records in the format

[0103] <device id>, <reading timestamp>, <reading data>

[0104] where <device id> is as described in the section Device Control File. The <reading timestamp> is the time of the reading and the <reading data> is a data record in a format specific to the type of reading. This file is internally maintained in XML format for ease of interfacing but is kept encrypted or in a non-readable format on the hub for security reasons.

[0105] Data Synchronization Between Receiving Station and Repository

[0106] The hub and the Repository frequently synchronize data. The hub may use one of various transportation methods to connect to the repository e.g. using a PC as conduit or via a connection established using an embedded modem (connected to a phone line) or via another network-connected device (such as, but not limited to, a web-phone, video-phone, embedded computer, PDA or handheld computer). The mechanism is as follows:

[0107] 1. When a user logs into the portal, the User Data File, resident on the hub, is uploaded to server. The server filters duplicate readings records.

[0108] 2. Alternatively, the Data Synchronization Client may upload the data from the hub to the Server without user intervention automatic data synchronization without user intervention also happens in the case of a stand-alone hub.

[0109] 3. The data may be decrypted on the server, assigned to each user (by determining which user the device is registered with) and put into a staging area. This staging area is so that the user may optionally preview the data before accepting the values. The server also filters for duplicate reading values.

[0110] 4. The device control file is uploaded to the server.

[0111] 5. The server downloads updated device control file with last updated timestamps for each device in list. If device has been removed from the system, then its entry in the file is deleted. The old device list file is backed up.

[0112] 6. The data file is cleared/deleted.

[0113] Data Synchronization Client

[0114] There is often the need for the reading data to be uploaded to the server without explicit user intervention. The Data Synchronization Client runs on the hub to upload the user's data to the server while synchronizing the information contained in the control file.

[0115] Alerts

[0116] Users may set up alerts that are triggered when one or more reading meet a certain set of conditions, depending on parameters defined by the user. The mechanism is as follows:

[0117] 1. The user sets up an alert by choosing the condition that they would like to be alerted to and by providing the parameters (e.g. threshold value for the reading) for alert generation.

[0118] 2. Each alert thus set up has an interval associated with it. This interval may be either the number of data points or a time duration in units such as hours, days, weeks or months, depending on the type of reading for which this alert is being set up.

[0119] 3. The values of readings lying within the interval specified by the user must all either positively or negatively exceed the specified threshold for the alert to be generated i.e. there is an implied "and" across all the readings within the interval.

[0120] 4. The user chooses the destination where the alert may be sent. This destination may include the user's portal, e-mail, pager, voice-mail or any combination of the above.

[0121] 5. Specific, preset alerts are provided based on medically established conditions in the case of common reading types e.g. heart rate, weight, blood sugar etc. The user may modify the parameter values for these alerts if they so desire.

[0122] Trends

[0123] The system computes trends in the user's data values over an interval. The mechanism is as follows:

[0124] 1. Trends are determined by applying mathematical and statistical rules (e.g. moving average and deviation) over a set of reading values. Each rule is configurable by parameters that are either automatically calculated or are set by the user.

[0125] 2. Currently the system tracks deviations from baseline values (see section Baselines) either automatically calculated using intelligent defaults or provided by the user.

[0126] 3. The user also supplies a range over which the trend is calculated. This is specified either as the number of data points or a numerical unit of time (e.g. hours, days, weeks etc, depending on reading type).

[0127] 4. This range is always applied backwards from the most current reading i.e. if the user specifies that the trend needs to be calculated over the past 5 days, then the system will calculate the trend of the past 5 days from current date.

[0128] 5. The system displays the deviation between the trend values computed from received data and the baseline value. This deviation is indicated both as a percentage as well as a numerical difference, along with icons ("+","−","=") to visually indicate the trend direction.

[0129] Baselines

[0130] 6. The system may be used to calculate a baseline value for each reading type using algorithms specific to each reading type. The user can override the default baseline average by changing the default parameters used to calculate the baseline e.g. the start date and the number of readings. Additionally, the user may directly provide baseline value(s) into the system, overriding the computed baseline values. If the user provides parameters such that there is insufficient data for calculating the baseline, then the system alerts the use to the fact. Previously set baseline values are not altered.

[0131] During early use of the system, there may be insufficient data for automatically calculating baselines for the user's reading types. In this case too, the system alerts the user to the fact.

[0132] Strength of Trend Correlation

[0133] The system may contain algorithms to compute the strength of the trend and correlate it to the visual perception of the trend as would be observed by a human observer.

[0134] Example Algorithm: In order for a weight trend to be visually perceived as increasing or decreasing, a certain positive or negative standard deviation from a moving average baseline must be observed over a pre-defined time period to be able to inform the user that their weight is either increasing or decreasing.

```
old-baseline = calculate moving average for initial data range
old-standard-deviation = 0
for each subsequent interval
    new-baseline = calculate moving average for updated data range
    new-standard-deviation = standard-deviation(new-baseline, old-
baseline)
    if (new-standard-deviation > 0)
        then trend is increasing
        if standard-deviation(new-standard-deviation, old-standard-
 deviation) > 0
            then strength of trend is increasing
        else strength of trend is decreasing
    else if (new-standard-deviation < 0)
        then trend is decreasing
        if standard-deviation(new-standard-deviation, old-standard-
deviation) > 0
            then strength of trend is increasing
        else strength of trend is decreasing
    old-baseline = new-baseline
    old-standard-deviation = new-standard-deviation
```

[0135] Reminders

[0136] The user may set up reminders on the device(s) via the portal. The device then draws the user's attention at the appointed time e.g. by sounding a buzzer. The mechanism is as follows:

[0137] 1. Each device associated with the user's account has its own independent set of reminders.

[0138] 2. Each reminder can be made repeatable. The user can set the reminder to repeat every n<units of time> (where <units of time> may be hours, days etc.) starting from a preset date and time.

[0139] 3. The reminders are associated with a user's account and are synchronized with the device(s) via the hub. The reminder information is maintained in the control file (see section Device Control File).

[0140] 4. The user can specify their time zone so as to account for differences between time on the server vs. the user's local time.

[0141] Buddy or Role Based Access

[0142] The user may give permission to others as needed to read or edit their personal data. This is useful, for example, for allowing a well-wisher to view ones data/charts, or receive alerts. The clinician could be allowed to edit data for example to annotate it, while the patient would have read-only privileges and that too for certain pages. An authorized person could set the reminders and alerts parameters with limited access to others.

[0143] The user or clinician could have a list of people that they want to monitor and have it show on their "My Account" page, which serves as the central monitoring station. The central monitoring concept could be particularly useful in an environment where on person wishes or needs to monitor multiple people.

[0144] Clinician Administrator

[0145] A clinician administrator (also referred to as Administrator in this context) may monitor the data for and otherwise administer a number of users of the system (also referred to as Patients in this context). The mechanism is as follows:

[0146] 1. A summary "dashboard" of readings from all Patients assigned to the Administrator is displayed upon log in to the Portal by the Administrator. At a minimum, the Patient's last reading received from all devices is shown (color coded to visually distinguish normal vs. readings that have generated an alert), along with description of the alert generated.

[0147] 2. The Administrator may drill down into the details for each Patient to further examine the readings data, view charts etc. in a manner similar to the Patient's own use of the system.

[0148] 3. The Administrator may also view a summary of all the devices registered to all assigned Patients, including but not limited to all device identification information.

[0149] 4. An Administrator has access only to information about Patients that have been assigned to the Administrator by a Super Administrator. This allows for segmenting the entire population of monitored Patients amongst multiple Administrators.

[0150] The Super Administrator may assign, remove and/or reassign Patients amongst a number of Administrators.

[0151] Patient Questionnaire

[0152] The Portal provides a mechanism to enhance the healthcare interaction and education of the users of the system (also referred to as Patients in this context). The mechanism is as follows:

[0153] 1. A questionnaire may be made available to the Patient containing questions pertinent to their health condition.

[0154] 2. The Patient's answers to the questions are stored and tracked by the time of each "session" (user interaction). Statistical summaries of these answers for various intervals of time are available for display.

[0155] 3. The Administrator (or delegate) defines the entire set of questions that may be made available to all Patients. Of these, pertinent questions are selected based on rules applied to the Patient's recent data in the Repository.

[0156] Data Export and Reporting

[0157] Data may be extracted, exported and reported from the Repository into a variety of formats, both application specific and industry standard. The mechanism is as follows:

[0158] 1. The data in the Repository is internally extracted into a neutral format represented in XML.

[0159] 2. This data is further processed to either aggregate information or to remove user identification information, as may be the specific requirement.

[0160] 3. The internal, neutral format may be transformed into custom, application specific formats or industry standard formats depending on specific application requirements.

Software Architecture Specification

[0161] Requirements

[0162] The architecture for the web application meets the requirements below. In particular, the architecture must be relatively independent of the user's client environment. At a minimum, the most common browsers (Netscape 4.0 and higher, IE 4.0 and higher) are supported on Windows (95, 98, NT, ME, 2000 now, XP in the near future). In addition, the architecture must be continuously adaptable to support newly emerging and evolving business requirements. This includes changes in the presentation, the business logic as well as the data tier.

[0163] In addition, the architecture should meet the following characteristics:

[0164] Scalable to meet performance levels as the user base grows, with the cost of scaling contained for at least 300,000 users.

[0165] Support rapid, iterative development. It is based on the J2EE Blueprints reference architecture with its timeline adapted to our schedule.

[0166] Not require expensive development tools. Most of the development tools are available free as either open source or extended evaluation versions.

[0167] Being based on industry-standard interfaces, it must allow for significant functionality to be purchased off-the-shelf (with integration and possible customization).

[0168] By adapting a standard reference architecture, it allows for the time required for developers to gain familiarity with the architecture to be compressed provided they are well versed in J2EE technology and architecture. This allows for the development effort to be outsourced at a later point in the development cycle. Moreover, the standard architecture, supports rapid and frequent changes to the look of the application. Such an architecture also supports the division of labor amongst the development team e.g. the application is partitioned such that the front-end designer requires minimal programming knowledge and vice versa.

[0169] The architecture should also partition the application to allow for the business logic to be accessed via a variety of interfaces e.g. the web (HTML) for now, wireless (WML) and/or voice response (VRU) in the future. It should also accommodate internationalization and localization of the application in the near future without significant changes. In addition, the architecture should leverage emerging technologies to provide a superior user experience e.g. it does not require the user to download, install and configure complicated software by utilizing emerging server-side technology. Moreover, the architecture should have a growth path to allow use of new technology as it becomes available over the application's lifespan (5 years). It should be designed to be resilient for at least the next 1-2 years while accommodating change. Finally, the architecture should not be tied to a particular set of products i.e. any J2EE compatible server can serve as application host. It also is relatively independent of database. This allows for flexibility in choosing and switching between hosting providers.

Technology

[0170] Platform

[0171] The Java 2 Enterprise Edition (J2EE) may be used as the standard technology platform for this application. Of the available J2EE components, the initial development phase may use Java Server Pages (JSPs), servlets, Java Database Connectivity (JDBC), Java API for XML Processing (JAXP). Other J2EE components, such as Enterprise Java Beans (EJBs), transactions APIs etc. may be used as required for subsequent development phases. Using J2EE components has the following advantages:

[0172] Simplified architecture and development

[0173] Scalability to meet demand variations

[0174] Integration with a variety of databases

[0175] Choices of servers, tools, components

[0176] Flexible security model

Relevant J2EE Features and Components

[0177] Scalability

[0178] J2EE provides mechanisms that support simplified scaling of distributed applications, without requiring significant effort on the part of the application development team. Because application services such as transaction support, database connections, life cycle management etc. are provided by the platform, the application can leverage these capabilities by adhering to the rules requiring it to be compatible with these services. For example, by using the appropriate JDBC APIs, connection pooling is automatically available for increased performance. Similarly, J2EE servers now provide many scaling mechanisms that were previously designed by the application developer.

[0179] Declarative Security

[0180] Security (access restriction, authentication) is configured declaratively at deployment time instead of being coded at development time. This allows for continuous refinement of the security mechanism (matching users with roles, with groups of users having specific access permissions) without requiring changes to the code followed by redeployment. For comparison, commercial products providing these kinds of capabilities for web applications (e.g. getAccess from Entrust Technologies) are typically priced at $15-$25 per user for quantities less than 5000 users.

[0181] Expandability

[0182] As the functionality provided by the J2EE expands, these capabilities are immediately available for use by the application. For example, the Connector architecture in the upcoming release of the J2EE will provide built-in capabilities of communicating with legacy systems (once commercial implementations are available).

[0183] Supported by Hosting Services

[0184] There are several commercial and open source implementations of J2EE compatible servers with the result that most hosting services offer reasonably priced hosting packages for J2EE applications.

Other Enabling Technologies and Products

Page Layout and Rendering

[0185] Flash

[0186] The pages should be as lightweight as possible while being aesthetically pleasing. This leads to using Flash as the rendering technology for items such as buttons and image maps etc. since this leads to compact graphics. However, there are no large animations.

[0187] CSS

[0188] The pages should not be hard-coded for a particular user screen size i.e. pages may be designed using Cascading Style Sheets (CSS).

[0189] Charting

[0190] Popchart (http://www.popchart.com) from Corda may be used to provide charting functions. It runs as an image server and services requests for charts from the web server over a pre-configured TCP port. It supports multiple output formats (including Flash, WML and SVG) and good chart interaction such as pop-up text and drill-down. The charts can be visually designed by a graphic designer and later rendered programmatically.

[0191] Reporting (Printing)

[0192] The reporting technology/product should support the following requirements:

[0193] It must be able to produce charts similar in nature to those produced on-screen for the user.

[0194] It must be able to export the user's data in Excel format for download.

[0195] It must be able to provide the user with the facility to print a report on his/her local printer.

[0196] It must be able to provide the user report in PDF format for use as an e-mail attachment.

[0197] The final choice of product is still pending. Among the products evaluated include Formula One from Tidestone Technologies and Style Report from Inetsoft Corp.

Development Tools

User Interface

[0198] Visual Design

[0199] Dreamweaver software may be used as a tool for developing the front-end HTML. Flash and (optionally) Fireworks software can be used along with Dreamweaver. MS Frontpage software, although popular, is not preferred since it introduces irrelevant HTML into each document.

[0200] Chart Design

[0201] Popchart software (see section Charting above) includes a graphics chart builder that allows charts to be interactively designed. The designer may use this builder to create aesthetically pleasing charts.

[0202] HTML Validation

[0203] All HTML should pass validation for acceptance. Tidy is one utility that can check HTML for validity.

Web Server

[0204] An Apache web server may be used for development. The deployment web server is provided by the hosting service.

[0205] Servlet Engine

[0206] Tomcat software may be used for development due to it providing integrated debugging with many IDEs (Integrated Development Environments) and its smaller memory footprint when compared with other similar commercial products.

Framework

[0207] J2EE Blueprints (JPS)

[0208] The current application architecture is modeled after the J2EE Blueprints reference application (the Java Pet Store version 1.1.1). Although EJBs are not used at this stage, the tier separation is maintained to allow them to be introduced at a later stage.

[0209] com.oreilly.servlet

[0210] This is a collection of utility classes available at www.servlets.com that may be useful in servlet development. Developed and maintained by Jason Hunter, author of the book Java Servlet Programming, 2nd Ed. (must read).

Java Code

[0211] Integrated Development Environment

[0212] Forte for Java, Community Edition may be used because it provides a rich feature set for free. JRun Studio has also been evaluated and may be used if we switch to JRun in a later development phase.

[0213] JSP Tag Libraries

[0214] Tag libraries from the Apache-Jakarta (and the Struts framework) project may be used if needed. Proprietary taglibs will not be used. Taglibs should be used if their use results in significant effort saving.

[0215] Code Documentation

[0216] Javadoc.

[0217] Build Tool

[0218] Any industry standard build tool (such as Ant or make) can be used as the build tool.

[0219] Unit Testing

[0220] The JUnit unit-testing framework (wwwjunit.org) may be used to unit test all developed code. Extensions to JUnit for server-side testing and J2EE testing (e.g. HttpUnit, Cactus) may also be used.

Database

[0221] Either Oracle or MS SQL Server 2000 may be used as the database. It is important to be careful to not have dependencies on the database to allow for switching if required by hosting providers.

Process Control

[0222] Configuration Management

[0223] Visual SourceSafe or CVS may be used. CVS is accessible over the Internet.

[0224] Bug Tracking

[0225] EbugStomp software may be used for web-based bug tracking.

Application Architecture

Request and Response Flow

[0226] This section describes the architecture of the application: exploring the partitioning of functionality into mod-

ules, the assignment of functionality to tiers, and object decomposition within the tiers.

Application Structure

[0227] Modules

[0228] This is a list of the functional modules of the system. Some of the modules are relatively simple but have been broken out to provide an idea of the broad functional areas of the application.

[0229] Control Module

[0230] This module provides the basic framework for the other functional modules. It provides key features such as navigation, security etc.

[0231] User Account Module

[0232] This module maintains information about a person's account (contact information, payment and credit card information) as well as the devices associated with a particular account.

[0233] User Profile Module

[0234] This module maintains information about a person's profile such as demographic information, medical conditions, medications and allergies.

[0235] Device Information Module

[0236] This module maintains information about the various types of devices supported by the system.

[0237] Data Synchronization Module

[0238] This module provides the functionality to upload the newly acquired readings data from a user's PC to the server. It flags potentially erroneous readings and provides a preview of the data so that the user may optionally reject some readings. Please refer to the section Data Synchronization between User's PC and Server for more details.

[0239] Data Management Module

[0240] This module manages the data obtained for each user. It also provides functionality to associate annotations with data points.

[0241] Data View: Charts Module

[0242] This module provides the functionality for the user to view his/her readings data as charts. The user may choose to view either preset ranges of data or choose an arbitrary range.

[0243] Data View: Tables Module

[0244] This module provides the user with a tabular view of their data. There are no complex table-editing functions.

[0245] Trends Module

[0246] This module contains the logic to provide the user indication of certain trends in their data.

[0247] Alerts Module

[0248] This module provides the functionality where the user can select from preset patterns to look for in the data to be alerted about. The user may choose certain parameters of the triggering events e.g. if their blood pressure is above X for Y days, then trigger an alert, where X and Y are chosen by the user.

[0249] Reminders Module

[0250] The reminders module generates preset reminders to a user about certain events that were supposed to have occurred but have not.

[0251] Groups Module

[0252] This module will provide functionality in which the user can choose to invite another registered user to be a part of his/her group. The invited user can then have access to the inviting user's data.

[0253] E-Mail Module

[0254] This module provides the functionality to send e-mail arising from a number of places in the application: when an alert is generated, for sending a user report etc.

[0255] Report Module

[0256] The report module will provide functionality to generate simple reports for the user incorporating charts.

[0257] Print Module

[0258] This module will provide the functionality to print on the client PC.

[0259] Export Module

[0260] This module provides the capability to export the user's data in Excel format.

[0261] Interfaces

[0262] The web application will have the following interfaces with external systems. These interfaces may be either synchronous (i.e. online, in real time) or asynchronous (i.e. as a batch operation, deferred in time).

[0263] Interface with Credit Card System

[0264] The application will transmit the user's credit card number and obtain online verification. It will also make periodic charges to the credit card as the user's account is automatically renewed on expiry.

[0265] Interface with E-Store

[0266] An outsourced e-store will provide the capability for a user to buy a device online. The application should be able to transfer information such as credit card information etc. to this system. Additionally, the front-end for such a system needs to be created and integrated with the rest of the application.

[0267] Interface with Refund Generation System

[0268] This system will generate pro-rated refund checks for users canceling their accounts. The application must transfer the required information to this system.

[0269] Interface with Customer Support System

[0270] A customer support system providing (at least) e-mail based support needs to be integrated with the application/site.

[0271] Interface with Device Control Software on the User's PC

[0272] At this time, the interface with the device control software running on the user's PC is asynchronous and is handled by uploading and downloading files to a known location on the user's disk. This will eventually be enhanced to obtain the file location from the Windows registry.

Key Mechanisms

Business Object Framework

[0273] The Business Object Framework (BOF) is designed as a lightweight stand-in for Enterprise Java Beans (EJBs). The idea is to create structural parallels with the EJB component structure (such as the home and remote interfaces, session and entity beans etc.) in a simple object framework that allows for easy migration to EJBs at a later time. The key differences between EJBs and BOs are:

[0274] There is no container that the BOF objects (and their derived objects/components) run in. Because of that, BOF-derived objects must manage their own lifecycle such as creation and destruction. This is however minimal and is abstracted behind utility (factory) methods.

[0275] BOF-derived objects are generically referred to as BOs (for business objects) and are packaged in the bo package within the application structure. The scope of BOs is the

same as for regular Java objects. This is different from EJBs, whose scope is controlled by the container and managed by methods on the home interface. The BO implementation objects must directly implement the BOF equivalent of the remote and home interfaces i.e. the BOInterface and BOHome. This is a key distinction from EJBs, in which the implementation object does not directly implement its home and remote interfaces.

[0276] The methods in the BOHome that are implemented by the BO have the same names as defined in the interfaces. This is different from EJBs which have a naming convention of prefixing ejb to the method name e.g. ejbCreate for a method named create in the home interface. BOs extend either the SessionBO or EntityBO base class. This is different from EJBs, which instead implement either the SessionBean or EntityBean interfaces. BOs are not remotely referenced and execute in the local JVM instead.

[0277] Components

[0278] The BOF has the following main types of components as set forth below:

[0279] Business Object Interface

[0280] This is the equivalent to the EJB remote interface. All business process methods are defined in this interface. Interfaces of this type extend com.carematix.bof.BOInterface and are named per the EJB convention i.e. using the domain class name. The BO must implement this interface directly.

[0281] Business Object Home (Interface)

[0282] This interface is the equivalent of the EJB home interface. At the very least, it must define a create(. . .) method to return a concrete BO object implementing the BOInterface. Interfaces of this type extend com.carematix.bof.BOHome and are named per the EJB convention i.e. by appending "Home" to the domain class name. The BO must implement this interface directly.

[0283] Business Object

[0284] This is the abstract base class (com.carematix.bof.bo) for Session Business Objects (SBOs) and Entity Business Objects (EBOs) described below.

[0285] Session Business Object (SBO)

[0286] An SBO is the equivalent of a session EJB. One key distinction is that SBOs can only be stateless, not stateful beyond the scope of the calling method. It is best to assume them as stateless. Typically, SBOs define most of their methods in the BOInterface and have just a simple create( ) method in the BOHome interface. A session BO extends the base class com.carematix.bof.SessionBO.

[0287] Entity Business Object (EBO)

[0288] An EBO is the equivalent of an entity EJB and encapsulates some persistent data representation. Typically, EBOs have a simple BOInterface and have most of their methods in the BOHome interface such as create (. . .), load (. . .), store (. . .), remove (. . .), findByPrimaryKey (. . .) etc. Not all of these methods are required in every EBO and only those needed by the application must be defined. An entity BO extends the base class com.carematix.bof.EntityBO.

EXAMPLES

[0289] Source code for the placeholder classes comprising the BOF is available. Accompanying that are two example packages: boexample1 and boexample2. These provide example implementations for a session BO and entity BO respectively. They illustrate the patterns and conventions required to create BOs and should be thoroughly studied.

[0290] Usage

[0291] In order to access a BO from calling code, the following example of a handler method for handling account events is illustrative:

```
        public void perform(PWMAppEvent event) throws
PWMAppEventException
  {
    AccountEvent ae = (AccountEvent)event;
    ...
    switch (ae.getActionType( ))
      {
        case AccountEvent.CREATE_ACCOUNT:
        {
          try
          {
            UserHome userHome = BOUtil.getUserHome( );
            User user = userHome.create( );
            user.createAccount(ae.getUserId( ), . . .);
            ...
            }
            catch (DuplicateKeyException dke)
            {
            ...
            }
            // Catch more exceptions
        }
    }
 }
```

[0292] Deployment

[0293] Entries need to be created in the web.xml file for each BO in order to enable JNDI lookup. Entries for the BO examples are as follows:

```
        <?xml version="1.0" encoding="ISO-8859-1"?>
        <!DOCTYPE web-app PUBLIC '-//Sun Microsystems,
Inc.//DTD Web Application 2.2//EN' 'http://java.sun.com/j2ee/dtds/
web-app.sub.--2.2.dtd- '>
        <web-app>
        . . .
        <env-entry>
        <env-entry-name>bo/boexample1/Example1</env-entry-name>
        <env-entry-type>java.lang.String</env-entry-type><env-entry-
value>com.carematix.boexample1.bo.Example1SBO</env-entry-value>
        </env-entry>
        <env-entry>
        <env-entry-name>bo/boexample2/Example2</env-entry-name>
        <env-entry-type>java.lang.String</env-entry-type>
        <env-entry-
value>com.carematix.boexample2.bo.Example2EBO</env-entry-value>
        </env-entry>
        <env-entry>
        <env-entry-name>bo/boexample2/
Example2DAOClass</env-entry--name>
        <env-entry-type>java.lang.String</env-entry-
type><env-entry-value>
        com.carematix.boexample2.dao.
Example2DAOImpl</env-entry-value>
        </env-entry>
        . . .
        </web-app>
```

[0294] Template Mechanism

[0295] The XML-based templating solution provides many benefits. XML is a useful, consistent way of representing application parameters. The XML format is easy to use for both presentation specialists and third-party tools. The fact

that XML is standardized means we can use off-the-shelf software to handle all the details of parsing and writing our screen definitions, configuration files, and other forms of structured data. Finally, XML can be validated against a DTD (Document Type Definition), so the parser can handle errors relating to ill-formed input, instead of in the application code.

[0296] Three files control the templating system:

[0297] The template file (template.jsp), which defines the basic layout of all of the pages generated from the template. There is one template file per supported language.

[0298] The request mappings file (requestmappings.xml), which maps virtual URLs to screen names (or request handler classes). There is one request mappings file in the application.

[0299] The screen definitions file (screendefinitions.xml), which defines the parameters for each screen. There is one screen definitions file per supported language.

[0300] The main controller servlet MainServlet receives incoming HTTP requests. MainServlet matches the incoming URLs to the URLs defined in the request mappings file, obtaining a screen name. The template parameters corresponding to the screen name are retrieved from the screen definitions file, and the template is then served using those parameters. The MainServlet does not do all of the work. Most of the work is delegated to other classes that specialize in screen flow management, template instantiation, and the like.

[0301] The XML files in the templating system are parsed by a standard XML parser, created using the JAXP (Java API for XML Parsers) optional package. The template is served as a JSP page, and the inclusion of the dynamic content occurs because the JSP engine calls the custom tag <j2ee:insert> each time it occurs.

[0302] In a Web application, each screen presented to the user can be considered as a different view. However, unlike the classic MVC architecture, all these views share the same controller. There needs to be a mechanism that allows the controller to choose a particular view to render in response to a user request. In the sample application, the controller makes this selection by specifying the screen ID of the screen to present as the response. This screen ID is mapped to a screen definition, then the template is instantiated. Recall that the file template.jsp defines the template for the sample application. This file includes another file, ScreenDefinitions.jsp, which defines all the screens of the sample application. When the controller invokes the template file at request time, it sets the appropriate screen definition in the request scope. The template file passes this information to the screen definitions file which then returns the appropriate screen definition for the request. One goal in structuring template and screen definition files is to facilitate internationalization (discussed in Section 4.5). This is achieved by separating text content from Java code. Since screen definitions that contain direct and indirect parameters are candidates for internationalization, we want to keep ScreenDefinitions.jsp devoid of Java technology code. We achieve this through the use of JSP custom tags.

[0303] Data Access Mechanism

[0304] The application uses a bimodal data access mechanism to increase efficiency of data access. Since it is anticipated that the majority of data access will be read-only (50% or more, as a conservative estimate), the data access mechanism will not utilize a transactional component for reading, relying instead on direct JDBC access to reduce overhead. Conversely, data updates will still continue to rely on a transaction component. When a transactional update does occur, the model will be refreshed from the data store and a model update event propagated to all registered views to update themselves, ensuring data display consistency.

[0305] Data Synchronization Between User's PC and Server

[0306] The functional specifications for the data synchronization mechanism are listed below, followed by details on each of the components that implement the mechanism. In cases where the implementation details of a component is not yet decided (such as whether to use an applet or not on the client PC), it is referred to generically (e.g. data sync client).

[0307] Functional Specifications

[0308] When a user logs on to the home page (FIG. **17**) using the PC the data file is uploaded to server. The server filters duplicate readings records. The records are decrypted on the server, assigned to each user (by determining which user the device is registered with) and put into a staging area. This staging area is so that the user may optionally preview the data before accepting the values. The device control file is uploaded to the server. The server downloads updated device control file with last updated timestamps for each device in list. If device has been removed from the system, then its entry in the file is deleted. The old device list file is backed up. The data file is cleared/deleted. If a user has logged onto the site for the first time from this PC, then a new control file is downloaded. This control file will contain entries for the registered devices of the current user only and will be built up as other users log into the PC. Similarly, when a user registers a device, its entry is added to the control file on the PC if not already there.

[0309] For this phase of implementation, the locations of the control file and the data file on the user's PC is fixed. Note the following:

[0310] No association information between the user and the device is kept on the user's PC. This determination is made after the data is uploaded to the server by matching the device id against the RegisteredDevice table.

[0311] Data for all users (i.e. from all devices in the control file) is uploaded to the server, regardless of the user logged on.

[0312] Device Control File

[0313] This file contains records in the format

[0314] <device id>, <data update timestamp>

[0315] where <device id> is all the information required to uniquely identify a device to the system (such as manufacturer number, model number, serial number and user number) and the <data update timestamp> is the timestamp (including date) that data for this device was last received by the server. Note that this is not the timestamp for which data was last uploaded from this particular PC (on which the control file resides) but is the timestamp for which the server last received data (from any user PC) for the particular (registered) device.

[0316] User Data File

[0317] This file contains records in the format

[0318] <device id>, <reading timestamp>, <reading data>

[0319] where <device id> is as described in the section Device Control File. The <reading timestamp> is the time of the reading and the <reading data> is a data record in a format specific to the type of reading. Each record in the file is individually encrypted (encryption to

be implemented in a subsequent phase) and is appended to the end of the file. Note that data for all users is aggregated in the same file.

[0320] Data Synchronization Client

[0321] This client runs on the user's PC to handle the file operations to and from the server such as uploading and downloading, checking for file existence, creating backup files etc. Note that these are all file operations and do not require any business logic in this client. All of the business logic decisions are made at the server and the appropriate files are downloaded. This allows for the client to be implemented as a signed applet.

[0322] A standalone Data Synchronization Client that does not require a PC and is directly able to connect to the server via a network connection is developed by embedding the synchronization logic in the firmware of a microcontroller and using it to connect to the server through a connection established either via a phone line (which requires an embedded modem) or via a connection established by another network terminal such as a web-phone or video-phone to which the Data Synchronization Client is connected.

[0323] Data Synchronization Manager

[0324] The Data Synchronization Manager component runs on the server and is the primary interface between the Data Synchronization Client and the rest of the application. Its primary operations include: receiving upload of the data file from Data Synchronization Client; decrypting the data records; determining which user each record belongs to (from RegisteredDevice mapping); putting each user's respective records in staging area (table) for user preview if so specified. This function is actually delegated to the ReadingsManager; putting each user's respective records in data tables if user does not want preview. This function is actually delegated to the ReadingsManager; signaling Data Synchronization Client to delete data file; receiving upload of control file; if control file does not exist, then creating it initially; updating control file with last uploaded timestamp for each registered device; downloading updated control file to Data Synchronization Client.

[0325] The Data Synchronization Manager has the following invocation points:

[0326] On user login.

[0327] On user registering one or more device(s).

[0328] On user previewing uploaded data prior to accepting it.

[0329] On an embedded Data Synchronization Client connecting to the server.

[0330] The Data Synchronization Manager is implemented as a Session BO as well as a standalone server application known as the iModem server.

[0331] IModem Server and Protocol Specification

[0332] The key attributes and mechanisms of the iModem server (i.e. Data Synchronization Manager for Standalone Data Synchronization Clients) are:

[0333] Accept multiple connections over the same server port at the same time: This will happen (in production) since more than one iModem could be connecting to the server and will require that the server be multi-threaded, using an independent thread for handling each connection. This is somewhat analogous to how web servers handle HTTP, which is a stateless protocol wherein several clients connect over the same port (**80**) at the same time. This also implies that the port number to which the iModem connects will remain constant.

[0334] Build in a versioning scheme into the protocol: This allows for upgrades over time without breaking backward compatibility.

[0335] Provide a status value in each message response. Zero indicates OK, non-zero values indicate various error conditions.

[0336] Error handling: There must be enough error handling and recoverability that a user's data is never lost and that each end (iModem or server) can signal to the other that data transfer either occurred or not. There must also be timeouts to handle conditions when no response is achieved. The value of the timeout (in seconds) must be a parameter read from a configuration XML file.

[0337] Break down current messages into multi-step messages: This is to handle two basic limitations: 1) each message to and from the iModem must be less than 256 bytes and 2) the iModem embedded code is more efficient in parsing fixed-length responses rather than variable length responses.

[0338] Encryption and keys: A basic mechanism (using placeholders as necessary) needs to be put in place such that in case the data stream is intercepted by a third party, all information that is private to the patient (such as reading values) and all information that can be correlated to deduce the message format (such as device identification information) is scrambled or encrypted.

[0339] This servlet is a "listener" for an embedded internet-modem (iModem) that has established a dial-up connection with a PPP server and is then communicating over TCP/IP to a specific server and port. The primary purpose of this servlet is to accept TCP connections over the specified port, parse the data coming over the connection and return responses. It also does basic connection management of connections and sockets as necessary. The incoming data stream comprises a series of messages in a simple protocol detailed below. This servlet parses the data stream, decodes the protocol message and responds based on logic detailed below. In summary, this servlet serves as a protocol handler for the iModem hub data transmission, which is done over TCP/IP sockets.

[0340] IModem Protocol.

[0341] The data protocol is a variant of the data exchange (XML) protocol that is used by the main web application for data interchange (defined by the files devicecontrol.xml and readingsdata.xml, hereafter referred to as the Datasync protocol). It differs from the Datasync protocol in the following manner to allow for the limited processing requirements of the iModem hub:

[0342] Numeric values are specified as hexadecimal text with each byte value comprised of 2 ASCII characters i.e. a decimal value 1085, which is 43D hex (or 043D hex, more accurately), will be specified as the characters '0', '4', '3' and 'D'. The most significant byte (MSB) is first when decoding multi-byte values.

[0343] Descriptive tags in the Datasync protocol are replaced with 4-digit hexadecimal numbers referred to as tokens. E.g. a <control-entry> tag may be denoted as <04AD>.sup.1. The mapping of the tags in the Datasync protocol to the iModem protocol must be configurable via another XML file (tagmappings.xml) so that it may be changed as the protocols evolve. End tags are denoted by a 4-digit hex number that is calculated by setting the most-significant-bit (MSB) of the number denoting the start tag. This defines the start tag and end tag relationship of the iModem protocol..sup.1 The angle braces < > will still be used to surround tags

[0344] Readings are sent serially as a series of Readings Data messages (see below). This is given the current limitation of the iModem restricting the message length to be less than 256 bytes. The server response is either a success or error indicator.

[0345] All timestamps are in seconds elapsed since Jan. 1, 1970, denoted as a 4-byte number and encoded as the ASCII representation of the hex digits, MSB first. There is an additional command for getting the current server time that is not part of the Datasync protocol. The details of this message are defined below, with a descriptive tag being used in place of a token for illustration purposes.

[0346] Initiate Session

[0347] Command

```
<initiate-session><hub-id>H3H2H1H0</hub-id><protocol-
version>P1P0</protocol-version></initate-session>
```

[0348] Response

```
<initiate-session><status>S1S0</status><current-
time>T3T2T1T0</current-time><update-flag>F0</
updateflag><key>K1K0</key><initate-session>
```

[0349] Details

[0350] The HubID is an identifier that is encoded in the iModem hub and is transmitted as a means of identifying itself. It is a long i.e. 4 byte number. This ID will be populated in the database through the device registration UI (which will be enhanced to support this later) and is used to associate a user (and his registered devices) with a specific hub.

[0351] On receiving this message, the server should set up a "session" with this connection and Hub ID and return a status code of 0000 in the response. All subsequent messages coming over this connection are considered in the context of the established session. If an unrecognized Hub ID is received, a status code of 0001 is returned with other message data being same. The <protocol-version> value is two byes, with the MSB being the major version and the LSB being the minor version. So currently, this value would be 0101 (current version is 1.1=>1<<8+1=257 decimal=0101 hex, formatted as 2 characters for each byte).

[0352] The response returned is the current server time, in seconds elapsed since Jan. 1, 1970. It is a 4-byte number and is encoded as the ASCII representation of the hex digits, MSB first.

[0353] There are two additional fields added for version 1.1. These are the update-flag and key. <update-flag> is a byte value that is used to signal to the iModem that there are parameter updates (see message Parameter Updates) that it needs to query for. The intent is to provide a way for parameters such as dial out numbers, dialing prefixes etc. to be provisioned via a web interface and indicated to the iModem so that it can update its own memory-based parameters. For version 1.1, the value returned is always 00 (zero).

[0354] The <key> is a placeholder value for now. It is a 16 bit number and its value is always 0000 (zero).

```
Parameter Updates
Command
<param-updates></param-updates>
Response
<param-updates><status>S1S0</status></param-updates&- gt;
```

[0355] Details

[0356] This is a placeholder message for now and will be expanded in future protocol revisions. The status code is always 0000.

[0357] Readings Data

[0358] Command

```
<readings-data><reading-record><device-id><manufacturer-
no>M2M1M0</manufacturer-no><device-model>D2D1D0</device-
model><serial-no>N3N2N1N0</serial-no><user-no>U0</user-no>
</device-id><timestamp>T3T2T1T0</timestamp><reading><reading-
type>BP-HR</reading-type><units>mm Hg</units><values>
<bp-systolic>V0</bp-systolic><bp-diastolic>V0</bp-diastolic>
<pulse-rate>V0</pulse-rate></values></reading><-/reading-record>
</readings-data>
Response
<readings-data><status>S1S0</status></readings-data>
Details
```

[0359] This message is a fixed-format equivalent to the readingsdata.xml file containing a single reading. The appropriate token values (as set up in tagmappings.xml file) are used for <reading-type> and <units> tag values.

[0360] The <status> value is a 16-bit integer. The value 0000 (zero) indicates no error whereas non-zero values indicate error conditions. Currently, only the value 0001 is assigned as a general error indicator. This list of values will be expanded over time and the code should be written to allow for it without significant change.

[0361] Because the iModem requires a fixed-length, fixed-format data command, the server must do some conversion/transformation to derive the application-specific XML <readings-data> command. The iModem will always send all data (BP, weight or blood sugar) in the format described above (for BP). The only difference for weight and blood sugar will be that the <reading-type> and <units> will correspond to the type of reading, so that for weight, <reading-type> value is WEIGHT and <units> value is POUNDS and for blood sugar, <reading-type> is BLOODSUGAR and <units> value is mg/dL. The value of the reading is derived from the byte values contained in the <bp-systolic>, <bp-diastolic> and <pulse-rate> tags. Because of this, the <value> tag is no longer required.

[0362] The server must use the following algorithm/logic to convert these fixed format iModem <readings-data> commands to the Datasync protocol's <readings-data> messages:

```
if (reading-type == BP-HR)
    then translate iModem <readings-data> message
directly into Datasync <readings-data> message
else if (reading-type == WEIGHT)
    then
        value = ((<bp-diastolic> << 8 + <bp-systolic>) >> 6) * 2.2
        // this value is in POUNDS, << & >> are left-shift & right-
shift operations
```

-continued

```
else if (reading-type == BLOODSUGAR)
    then
        value = <bp-systolic>
```

[0363] Number of Devices

[0364] Command

[0365] <device-control-list></device-control-list>

[0366] Response

```
<device-control-list><status>S1S0</status><num-
devices>N0</num-devices></device-control-list>
```

[0367] Details

[0368] This command is used to obtain the number of devices currently registered by the user. The status is generally expected to be 0000, non-zero indicates an unspecified (as of yet) error.

[0369] Device Information

[0370] Command

[0371] <device-info><device-num>N0</device-num></device-info>

[0372] Response

```
<device-info><status>S1S0</status><control-entry><device-
id><manufacturer-no>M2M1M0</manufacturer-no><device-
model>D2D1D0 </device-model><serial-no>N3N2N1N0</serial-
no><user-no>U0</user-no></device-id><reminders></reminders><first-
association-timestamp> T3T2T1T0</first-association-timestamp><last-
update-timestamp> T3T2T1T0</last-update-timestamp></control-
entry></device-info>
```

[0373] Details

[0374] The iModem sequentially requests device details for each device registered by the user, passing in a device number (actually, device index) as part of the command. The list is zero-indexed i.e. the first device is 0 (zero) and the index increments sequentially from there. The information in the response is the same as a single-entry device control file. Currently, nothing is returned for reminders (even if there are some set) since the iModem is not set up to handle them. These will be handled in a subsequent protocol revision.

[0375] Terminate Session

[0376] Command

[0377] <terminate-session></terminate-session>

[0378] Response

[0379] This command does not return a response.

[0380] Details

[0381] This command is for the iModem to indicate to the server that it is done with the session prior to terminating the connection so that the server may release resources and perform other cleanup. The server does not return a response to this command since the client's state is undefined after this. It releases resources and terminates the server's socket connection.

[0382] Reminders

[0383] Reminders are in a slightly different format from the Datasync protocol. The structure is as follows:

```
<reminders><reminder><start>T3T2T1T0</start>-<repe at>
nn</repeat><for>nn</for></reminder>; . . . </reminders>
```

[0384] where . . . denotes more reminder records, each delineated by a <reminder></reminder> sequence. Each reminder block is part of the <control-entry> block. The <repeat> value is the number of seconds to repeat the reminder after the start. It is zero if there is no repetition. The <for> value is the number of times this repetition must be carried out. It too is zero if there is no repetition.

[0385] Chart Creation and Display

[0386] The chart creation and display mechanism is comprised of collaborating objects and components across all four application tiers (view, web, business object and data persistence tiers). As previously mentioned, Popchart is the charting server being used to render the data for display in the Flash format. This requires the Flash plug-in to be installed in the client browser, which the majority of the browsers (over 90%) already do.

[0387] The approach to chart creation and display is to isolate dependency on Popchart in the view tier only. This is to allow change over to another charting product if required later. Doing this requires encapsulating all code to create a Popchart chart in presentation components in the view tier. All the information (data) required to produce the chart is kept as generic objects or components and supplied to the view components via the standard application framework. This mechanism is elaborated upon below.

[0388] The View Tier

[0389] This tier is comprised of the JSP page containing the chart as well as an embedded JavaBean that converts the chart model objects into Popchart specific instructions (there may be the need to do specific JavaBeans for each chart type but the approach will remain the same for each). This JavaBean encapsulates the logic to create Popchart instructions from the chart model objects and isolates the dependency on Popchart.

[0390] The Popchart JSPExample demonstrates such a JSP/JavaBean combination. The key points of difference are:

[0391] The example code has a lot of hard-coded elements such as label information etc. All this information is being supplied in the chart model object(s) instead.

[0392] The example code has a lot of direct database access for retrieving data. All data required to create the chart will be supplied through the chart model object(s) or in objects directly associated with them. Some of these data objects would be the domain model objects themselves e.g. the Range object would contain the range of reading values to be charted.

[0393] The Web Tier

[0394] This tier contains the standard web handler for processing user input to the chart (such as when a user displays the chart, changes the date range, clicks on a point for annotation or drill-down etc.). The handler creates a chart event object (class ChartEvent) encapsulating the user request and forwards it to the Business Object tier for handling.

[0395] Note that there may be requests that can be handled in the web tier itself instead of having to go to the BO tier. For

example, the user may choose to view a data range that is a subset of the range already displayed. In this case, there is no need to go to the BO tier to get data for display since the data has already been sent to the web tier in the chart model object(s). The web tier handlers need to be intelligent enough to check for these cases in order to improve application performance.

[0396] The Business Object Tier

[0397] This tier contains the components required to handle chart events generated from the web tier. The main facade component is a session BO named ChartManagerSBO. This BO interprets the chart event and creates/retrieves the chart model objects that are required to generate the requested chart. It invokes the ReadingsManager to get the actual readings model objects. This BO also provides the defaults for chart presentation.

[0398] The chart model objects also belong in this tier. These objects encapsulate the complete information required to render the chart. This includes information about the various chart display parameters as well references to the model objects that hold the actual data for display. The data used to populate the chart model is obtained from the chartconfig.xml file using a DAO implementing the ChartDAO interface.

[0399] The chart model object is mutable i.e. it can be changed when a user adds or changes annotations on data points. The annotation actually changes a particular reading referenced by the range, which is in turn referenced by the chart model object.

[0400] Implementation Notes

[0401] Popchart servlet redirector may be used.

[0402] The implementation DAO object actually is ChartXmlDAOImpl, meaning that it reads the XML configuration file.

[0403] The elements in the chart model object that are specific to the charting product are kept as properties so as to not make the structure of the model object dependent on product-specific attributes.

Patterns

[0404] Following are the design patterns utilized in the architecture. Further details can be found in publicly available J2EE Blueprints documents as well as in a book, entitled "Design Patterns", by Vlissides, Gamma, et al and are thus elaborated upon further here.

[0405] Model-View-Controller (MVC)

[0406] Data Access Object (DAO)

[0407] Bimodal Data Access

[0408] Front Component

[0409] Value Object

File Formats

[0410] Device Control File

[0411] This file is currently in plain text XML format. It may later be stored encrypted if necessary. It does not currently have a DTD and therefore requires a non-validating parser. For now, the URI for this file is file://c:.backslash.carematix.backslash.config.backslash.devicecontrol.xml.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<device-control-list>
  <control-entry>
```

-continued

```
    <device-id>
      <manufacturer-no>12345</manufacturer-no>
      <device-model>67890</device-model>
      <serial-no>555555555</serial-no>
      <user-no>1<user-no>
    </device-id>
  <last-update-timestamp>06-05-2001 11:21:33</last-update-
timestamp>
  </control-entry>
 <!-- More <control-entry> records for all other devices that
  users of this PC have registered with Carematix.
 -->
 </device-control-list>
```

[0412] User Data File

[0413] This file is currently in plain text XML format. It may later be converted into a format in which each XML <reading-record> entry is encrypted and stored as a fixed length record, with new records being appended to the end. It does not currently have a DTD and therefore requires a non-validating parser. For now, the URI for this file is file://c:.backslash.carematix.backslash.data.backslash.readingsdata.xml.

[0414] Note: This file may contain duplicate <reading-record> entries. That is allowed.

```
<readings-data>
  <!-- Reading record for blood pressure and heart rate reading -->
  <reading-record>
    <device-id>
      <manufacturer-no>12345</manufacturer-no>
      <device-model>67890</device-model>
      <serial-no>555555555</serial-no>
      <user-no>1<user-no>
    </device-id>
    <timestamp 06-05-2001 11:21:33 PM/>
    <reading>
    <!-- The reading-type tag may be eliminated later -->
    <reading-type>BP-HR</reading-type>
    <values>
    <bp-systolic 120 />
    <bp-diastolic 80 />
    <pulse-rate 72 />
    </values>
  </reading>
</reading-record>
<!-- Reading record for temperature reading -->
<reading-record>
  <device-id>
    <manufacturer-no>29933</manufacturer-no>
    <device-model>68909</device-model>
    <serial-no>123456789</serial-no>
    </device-id>
<timestamp 06-05-2001 11:21:42 PM/>
<reading>
<!-- The reading-type tag may be eliminated later -->
 <reading-type>TEMP</reading-type>
  <!-- The units for temperature are FARENHEIGHT
  or CELCIUS
   <units FARENHEIGHT />
  <value 98.7 />
 </reading>
</reading-record>
<!-- Reading record for weight reading -->
<reading-record>
  <device-id>
    <manufacturer-no>29890</manufacturer-no>
    <device-model>68309</device-model>
    <serial-no>123456789</serial-no>
    </device-id>
```

-continued

```
<timestamp 06-05-2001 11:25:42 PM/>
<reading>
  <!-- The reading-type tag may be eliminated later -->
<reading-type>WEIGHT</reading-- type>
  <!-- The units for weight are KILOGRAMS or POUNDS -->
  <units KILOGRAMS />
  <value 65 />
```

-continued

```
  </reading>
 </reading-record>
... more reading records from all devices (and all users)
</readings-data>
```

XML to iModem Tag Mappings

```
<tag-mapping type="request" xml-tag="readings-data" imodem-tag="0000"/>
<tag-mapping type= "request" xml-tag="reading- record" imodem-tag="0001"/>
<tag-mapping type= "request" xml-tag="reading" imodem -tag="0002"/>
<tag-mapping type="request" xml-tag="reading-type" imodem- tag="0003" byte-length="1">
<tag-token xml-token="BP-HR" imodem-token="04"/>
<tag-token xml-token="TEMP" imodem-token="05"/>
<tag-token xml-token="WEIGHT" imodem-token="06"/>
<tag-token xml-token="BLOODSUGAR" imodem-token="07"/>
</tag-mapping>
<tag-mapping type="request" xml-tag="values" imodem-tag="0004"/>
<tag-mapping type="request" xml-tag="bp-systolic" imodem- tag="0005" byte-length="1"/>
<tag-mapping type="request" xml-tag="bp-diastolic" imodem- tag="0006" byte-length="1"/>
<tag-mapping type="request" xml-tag="pulse-rate" imodem- tag="0007" byte-length="1"/>
<tag-mapping type="request" xml-tag="units" imodem- tag="0008" byte-length="1">
<tag-token xml-token="mm Hg" imodem-token="00"/>
<tag-token xml-token="Kelvin" imodem-token="01"/>
<tag-token xml-token="POUNDS" imodem-token="02"/>
<tag-token xml-token="mg/dL" imodem-token="03"/>
</tag-mapping>
<tag-mapping type="response" xml-tag="current- time" imodem-tag="0009" byte-length="4"/>
<tag-mapping type="common" xml-tag="device-id" imodem- tag="000A"/>
<tag-mapping type="common" xml-tag="manufacturer- no" imodem-tag="000B" byte-length="3"/>
tag-mapping type="common" xml-tag="device- model" imodem-tag="000C" byte-length="3"/>
tag-mapping type="common" xml-tag="serial-no" imodem- tag="000D" byte-length="4"/>
tag-mapping type="common" xml-tag="user-no" imodem- tag="000E" byte-length="1"/>
tag-mapping type="common" xml-tag="device-control- list" imodem-tag="000F"/>
tag-mapping type="response" xml-tag="control- entry" imodem-tag="0091"/>
tag-mapping type="response" xml-tag="reminders" imodem- tag="0092"/>
tag-mapping type="response" xml-tag="reminder" imodem- tag="0093"/>
tag-mapping type="response" xml-tag="start" imodem- tag="0094" byte-length="4"/>
tag-mapping type="response" xml-tag="repeat" imodem- tag="0095"byte-length="4"/>
tag-mapping type="response" xml-tag="for" imodem- tag="0096" byte-length="1"/>
tag-mapping type="request" xml-tag="timestamp" imodem- tag="0097"byte-length="4"/>
<tag-mapping type="response" xml-tag="first-association-timestamp" imodem-tag="0098" byte-length="4"/>
<tag-mapping type="response" xml-tag="last-update-timestamp" imodem-tag="0099" byte-length="4"/>
<tag-mapping type="response" xml-tag="status" imodem- tag="009A"byte-length="2"/>
<tag-mapping type="common" xml-tag="device-info" imodem-tag="009B">
<tag-mapping type="response" xml-tag="num-devices" imodem-tag="009C" byte-length="1"/>
<tag-mapping type="request" xml-tag="device-num" imodem-tag="009D" byte-length="1"/>
<tag-mapping type="common" xml-tag="initiate-session" imodem-
```

-continued

```
tag="0030"/>
    <tag-mapping type="request" xml-tag="hub-id" imodem-tag="0031" byte-
length="4"/>
    <tag-mapping type="request" xml-tag="protocol-version" imodem-tag="0032"
byte-length="2"/>
    <tag-mapping type="response" xml-tag="update-flag" imodem-tag="0033"
byte-length="1"/>
    <tag-mapping type="response" xml-tag="key" imodem-tag="0034" byte-
length="2"/>
    <tag-mapping type="common" xml-tag="terminate-session" imodem-
tag="0035"/>
    <tag-mapping type="common" xml-tag="param-updates" imodem-
tag="0036"/>
```

[0415] Chart Configuration File

[0416] This file contains the necessary information for formatting and displaying data charts. This information is kept in XML format neutral of the specific charting product so as to allow changeover to another charting application if necessary later. It does not currently have a DTD and therefore requires a non-validating parser. This file is stored on the server, not the client and is called chartconfig.xml

[0417] Please note the following: this format is an initial version and will likely be enhanced over time. The method for altering and enhancing chart display is to add tags and/or properties in the sections for each chart and use the values assigned to each to drive the behavior of the code. This is preferred over hard-coding the way a chart is displayed in the code itself.

[0418] In subsequent phases, this file may have an XSL transform applied to generate the product specific charting instructions e.g. if using Popchart, an XSLT would be provided to generate PCScript from this XML file.

```
<chart-config>
    <bp-chart>
        <type>TIMELINE</type>
        <title>Blood Pressure</title>
        <height>xxx</height>
        <width>yyy</width>
        <property name="appearance-file" value=
        "URL of appearance file" />
        <property name="date-format" value="%m/%d/%y %H:%M" />
    </bp-chart>
    <hr-chart>
        <type>TIMESCATTER</type>
        <title>Heart Rate</title>
        <height>xxx</height>
        <width>yyy</width>
        <property name="appearance-file" value=
        "URL of appearance file" />
        <property name="date-format" value="%m/%d/%y %H:%M" />
    </hr-chart>
    <temperature-chart>
        <type>TIMELINE</type>
        <title>Temperature</title>
        <height>xxx</height>
        <width>yyy</width>
        <property name="appearance-file" value=
        "URL of appearance file" />
        <property name="date-format" value="%m/%d/%y %H:%M" />
    </temperature-chart>
    <weight-chart>
        <type>TIMELINE</type>
        <title>Blood Pressure</title>
        <height>xxx</height>
```

-continued

```
        <width>yyy</width>
        <property name="appearance-file" value=
        "URL of appearance file" />
        <property name="date-format" value="%m/%d/%y %H:%M" />
    </weight-chart>
</chart-config>
```

[0419] Obviously, many modifications and variations of the present invention are possible in light of the above teachings. Thus, it is to be understood that, within the scope of the appended claims, the invention may be practiced otherwise than as specifically described above.

What is claimed and desired to be covered by a Letters Patent is as follows:

1. A patient monitoring system including:

a patient monitoring device determining a plurality of physiological readings of a patient and wirelessly transmitting said plurality of patient readings to a wireless hub;

a wireless hub wirelessly receiving said plurality of physiological readings; and

a remote server, wherein said remote server receives said plurality of physiological readings from said wireless hub and compares said plurality of physiological readings to a predetermined alert condition based on a plurality of physiological readings,

wherein an alert indication is provided to said patient when said plurality of patient readings violate said alert condition.

2. The system of claim **1** wherein said alert condition is established by said patient.

3. The system of claim **1** wherein said alert condition is established by an entity other than said patient.

4. The system of claim **3** wherein said alert condition is established by a physician.

5. The system of claim **1** wherein said alert condition is established using a web portal to said remote server.

6. The system of claim **1** further including a second predetermined alert condition, wherein at least one of said plurality of physiological readings is compared to said second predetermined alert condition.

7. The system of claim **1** wherein said predetermined alert condition is established using a trend in a plurality of previous physiological readings from said patient.

8. The system of claim **1** wherein an alert indication is also provided to an entity other than said patient.

9. The system of claim **8** wherein said entity is a physician.

10. The system of claim **1** wherein a plurality of alerts are provided to said patient.

11. The system of claim **10** wherein a first of said plurality of alerts is provided to said patient using a first communication channel and a second of said plurality of alerts is provided to said patient using a communication channel other than said first communication channel.

12. The system of claim **11** wherein one of said communication channels is a pager and another of said communication channels is e-mail.

14. A patient monitoring system including:

a patient monitoring device determining a first physiological reading of a patient and a second physiological reading of a patient and wirelessly transmitting said first physiological reading and said second physiological reading to a wireless hub;

a wireless hub wirelessly receiving said first physiological reading and said second physiological reading; and

a remote server, wherein said remote server receives said first physiological reading and said second physiological reading from said wireless hub, wherein said remote server includes a predetermined alert condition that requires a plurality of physiological readings from a patient to determine whether said alert condition has been triggered,

wherein said remote server uses both said first physiological reading and said second physiological reading to determine whether said alert condition has been triggered,

wherein an alert indication is provided to said patient when said alert condition has been triggered.

15. The system of claim **14** wherein said remote server also includes a second predetermined alert condition that requires only a single physiological reading from a patient to determine whether said second predetermined alert condition has been triggered,

wherein at least one of said first physiological reading and said second physiological reading are compared to said second predetermined alert condition to determining whether said second predetermined alert condition has been triggered,

wherein an alert indication is provided to said patient when said second alert condition has been triggered.

16. The system of claim **15** wherein both said first physiological reading and said second physiological reading are compared to said second predetermined alert condition.

17. The system of claim **14** wherein said alert condition is established using a web portal to said remote server.

18. The system of claim **14** wherein said predetermined alert condition is established using a trend in a plurality of previous physiological readings from said patient.

19. The system of claim **14** wherein an alert indication is also provided to an entity other than said patient.

20. The system of claim **14** wherein a plurality of alerts are provided to said patient.

21. The system of claim **20** wherein a first of said plurality of alerts is provided to said patient using a first communication channel and a second of said plurality of alerts is provided to said patient using a communication channel other than said first communication channel.

22. The system of claim **21** wherein one of said communication channels is a pager and another of said communication channels is e-mail.

23. The system of claim **15** wherein a plurality of communication channels to said patient are provided and the communication channel selected to communicate an alert indication varies depending upon whether said alert condition or said second alert condition has been triggered.

24. A method for monitoring a patient, said method including:

determining a plurality of physiological readings of a patient at a patient monitoring device;

wirelessly transmitting said plurality of patient readings to a wireless hub;

wirelessly receiving said plurality of physiological readings at said hub;

transmitting said plurality of physiological readings to a remote server;

receiving said plurality of physiological readings from said wireless hub at said remote server;

comparing said plurality of physiological readings to a predetermined alert condition based on a plurality of physiological readings; and

providing an alert indication to said patient when said plurality of patient readings violate said alert condition.

25. A method for monitoring a patient, said method including:

determining a first physiological reading of a patient at a patient monitoring device;

determining a second physiological reading of said patient at said patient monitoring device;

wirelessly transmitting said first physiological reading and said second physiological reading to a wireless hub;

wirelessly receiving said first physiological reading and said second physiological reading at said hub;

transmitting said first physiological reading and said second physiological reading to a remote server;

receiving said first physiological reading and said second physiological reading from said wireless hub at said remote server, wherein said remote server includes a predetermined alert condition that requires a plurality of physiological readings from a patient to determine whether said alert condition has been triggered;

using both said first physiological reading and said second physiological reading to determine whether said alert condition has been triggered; and

providing an alert indication to said patient when said alert condition has been triggered.

* * * * *