US 20060161973A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2006/0161973 A1**

Royer et al. (43) **Pub. Date: Jul. 20, 2006**

(54) **SYSTEM AND USER INTERFACE SUPPORTING CONCURRENT APPLICATION INITIATION AND INTEROPERABILITY**

(76) Inventors: **Barry Lynn Royer**, Blue Bell, PA (US); **John Andrew Heil**, Malvern, PA (US)

Correspondence Address:
Siemens Corporation
Intellectual Property Department
170 Wood Avenue South
Iselin, NJ 08830 (US)

(21) Appl. No.: **11/377,984**

(22) Filed: **Mar. 17, 2006**

Related U.S. Application Data

(62) Division of application No. 09/817,311, filed on Mar. 26, 2001, now Pat. No. 7,043,752.

(60) Provisional application No. 60/261,148, filed on Jan. 12, 2001.

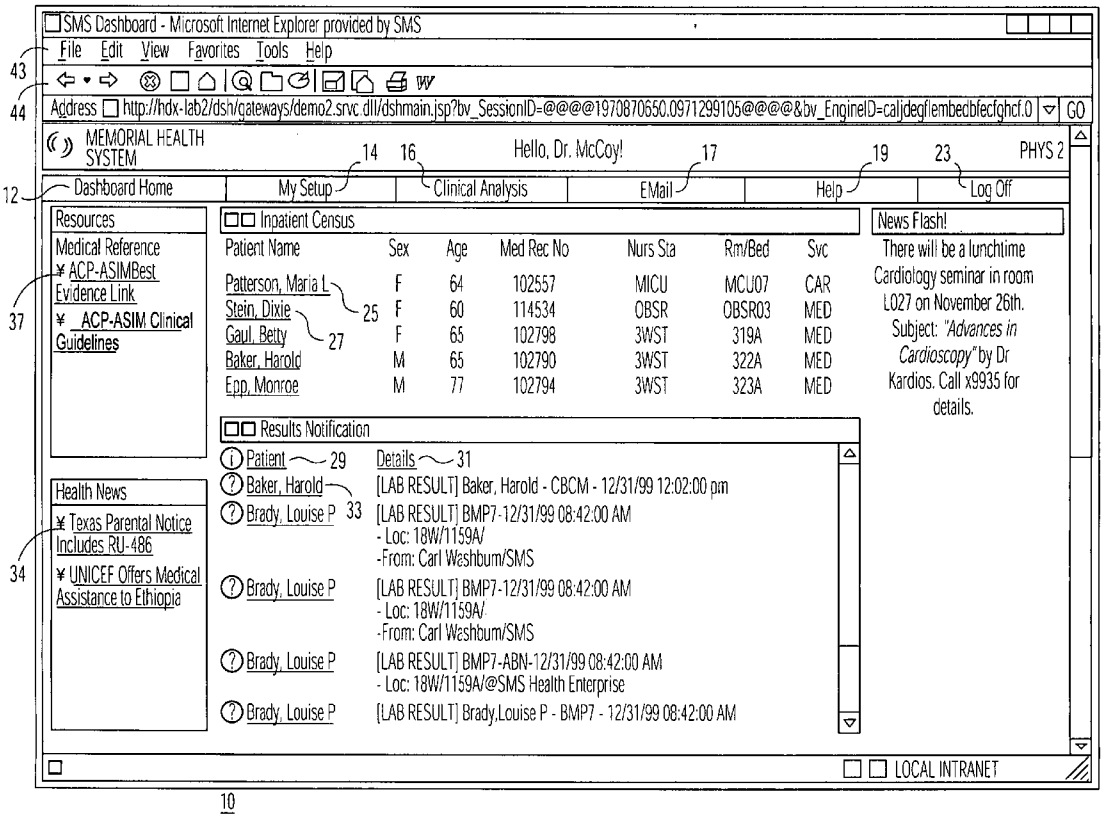
Publication Classification

(51) **Int. Cl.**
G06F 17/30 (2006.01)

(52) **U.S. Cl.** **726/8**

(57) **ABSTRACT**

A system and associated communication protocol enables Internet compatible applications to be integrated into any process involving concurrent operation of applications. A system employed by a first application for supporting concurrent operation of a plurality of network compatible applications involves an entitlement processor for authorizing user access to the first application in response to validation of user identification information. The system also involves a communication processor for communicating a session initiation request to a managing application to initiate generation of a session identifier particular to a user initiated session and for use by a plurality of concurrently operating applications to uniquely identify the user initiated session in response to validation of user identification information. The managing application also initiates generation of an encryption key particular to a user initiated session for common use by a plurality of concurrently operating applications in encrypting data associated with a personal record.



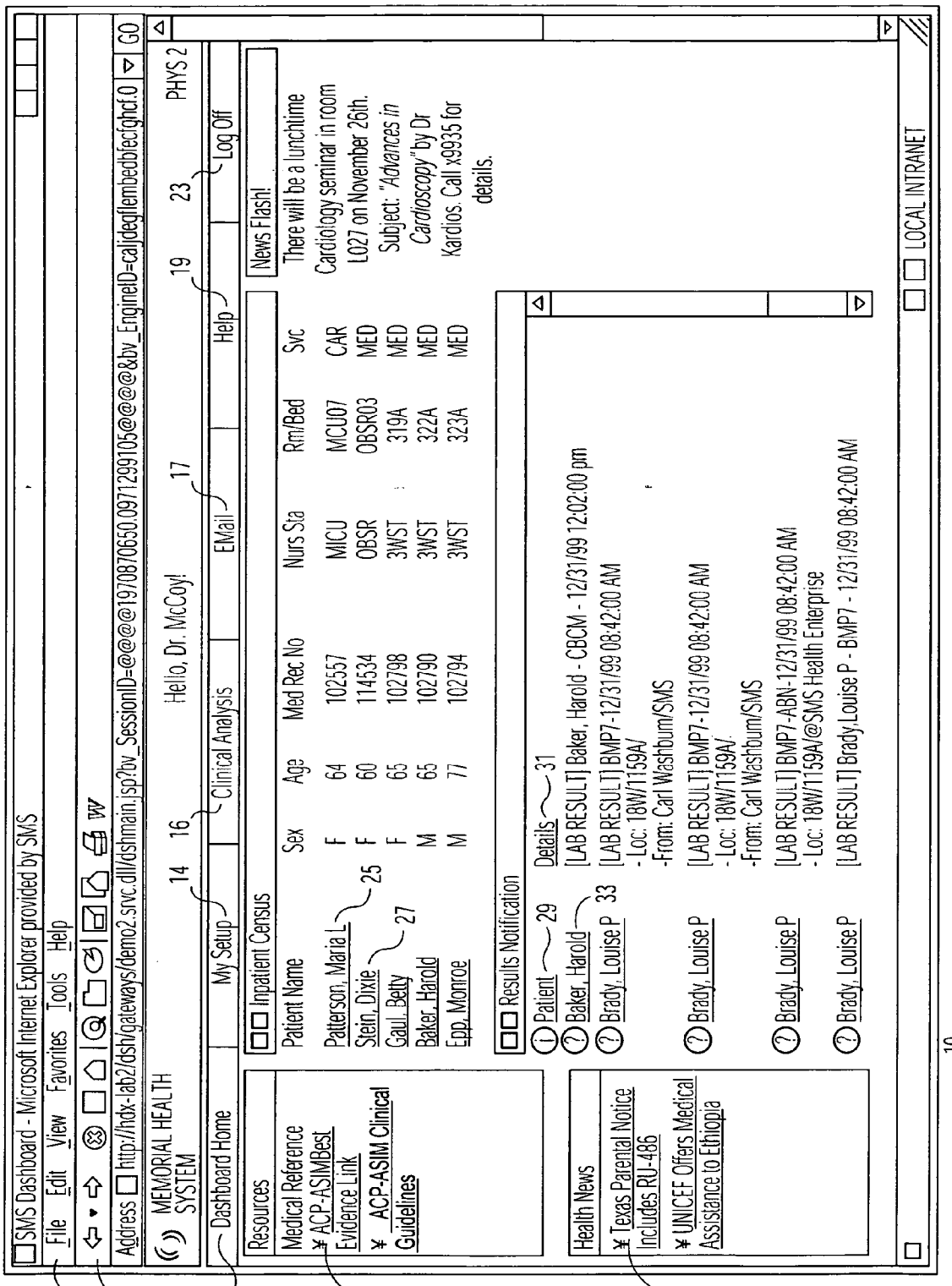

SMS Dashboard - Microsoft Internet Explorer provided by SMS
File Edit View Favorites Tools Help
Address http://hdx-lab2/dsh/gateways/demo2.srvc.dll/dshmain.jsp?bv_SessionID=@@@@1970870650.0971299105@@@@&bv_EngineID=caljdegflembedbfecfghcf.0
GO
MEMORIAL HEALTH SYSTEM
Hello, Dr. McCoy!
PHYS 2
Dashboard Home
My Setup
Clinical Analysis
EMail
Help
Log Off
Resources
Medical Reference
ACP-ASIMBest Evidence Link
ACP-ASIM Clinical Guidelines
Inpatient Census
Patient Name
Sex
Age
Med Rec No
Nurs Sta
Rm/Bed
Svc
Patterson, Maria L
F
64
102557
MICU
MCU07
CAR
Stein, Dixie
F
60
114534
OBSR
OBSR03
MED
Gaul, Betty
F
65
102798
3WST
319A
MED
Baker, Harold
M
65
102790
3WST
322A
MED
Epp, Monroe
M
77
102794
3WST
323A
MED
News Flash!
There will be a lunchtime Cardiology seminar in room L027 on November 26th. Subject: "Advances in Cardioscopy" by Dr Kardios. Call x9935 for details.
Results Notification
Patient
Details
Baker, Harold
[LAB RESULT] Baker, Harold - CBCM - 12/31/99 12:02:00 pm
Brady, Louise P
[LAB RESULT] BMP7-12/31/99 08:42:00 AM - Loc: 18W/1159A/ -From: Carl Washburn/SMS
Brady, Louise P
[LAB RESULT] BMP7-12/31/99 08:42:00 AM - Loc: 18W/1159A/ -From: Carl Washburn/SMS
Brady, Louise P
[LAB RESULT] BMP7-ABN-12/31/99 08:42:00 AM - Loc: 18W/1159A/@SMS Health Enterprise
Brady, Louise P
[LAB RESULT] Brady,Louise P - BMP7 - 12/31/99 08:42:00 AM
Health News
Texas Parental Notice Includes RU-486
UNICEF Offers Medical Assistance to Ethiopia
LOCAL INTRANET
10
12
14
16
17
19
23
25
27
29
31
33
34
37
43
44

FIG. 1

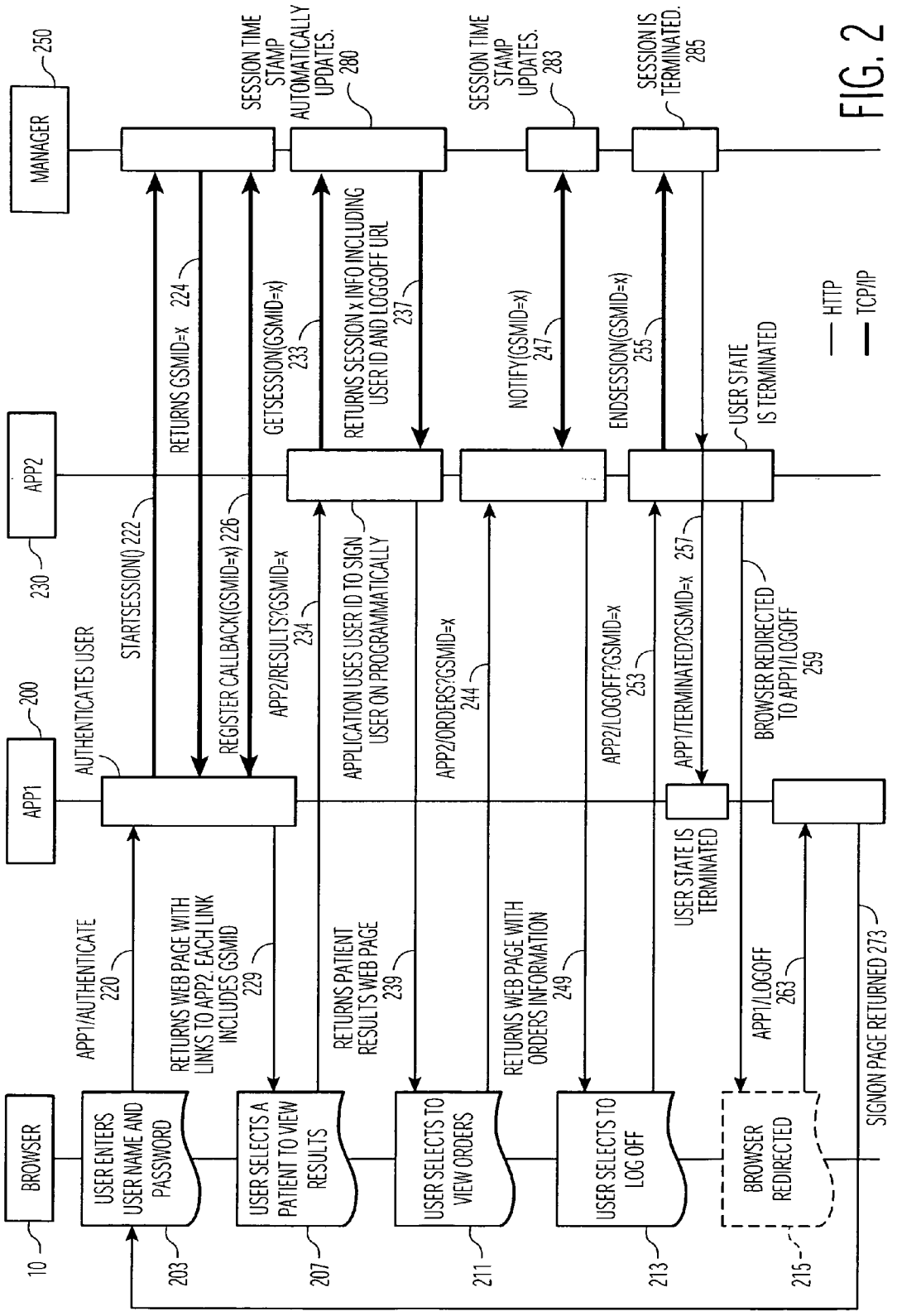

BROWSER
APP1
APP2
MANAGER
10
200
230
250
USER ENTERS USER NAME AND PASSWORD
203
APP1/AUTHENTICATE 220
AUTHENTICATES USER
STARTSESSION() 222
RETURNS GSMID=x 224
RETURNS WEB PAGE WITH LINKS TO APP2. EACH LINK INCLUDES GSMID 229
USER SELECTS A PATIENT TO VIEW RESULTS
207
REGISTER CALLBACK(GSMID=x) 226
APP2/RESULTS?GSMID=x 234
GETSESSION(GSMID=x) 233
SESSION TIME STAMP AUTOMATICALLY UPDATES. 280
APPLICATION USES USER ID TO SIGN USER ON PROGRAMMATICALLY
RETURNS PATIENT RESULTS WEB PAGE 239
RETURNS SESSION x INFO INCLUDING USER ID AND LOGGOFF URL 237
USER SELECTS TO VIEW ORDERS
211
APP2/ORDERS?GSMID=x 244
RETURNS WEB PAGE WITH ORDERS INFORMATION 249
NOTIFY(GSMID=x) 247
SESSION TIME STAMP UPDATES. 283
USER SELECTS TO LOG OFF
213
APP2/LOGOFF?GSMID=x 253
ENDSESSION(GSMID=x) 255
SESSION IS TERMINATED. 285
USER STATE IS TERMINATED
APP1/TERMINATED?GSMID=x 257
USER STATE IS TERMINATED
BROWSER REDIRECTED
215
APP1/LOGOFF 263
BROWSER REDIRECTED TO APP1/LOGOFF 259
SIGNON PAGE RETURNED 273
HTTP
TCP/IP

FIG. 2

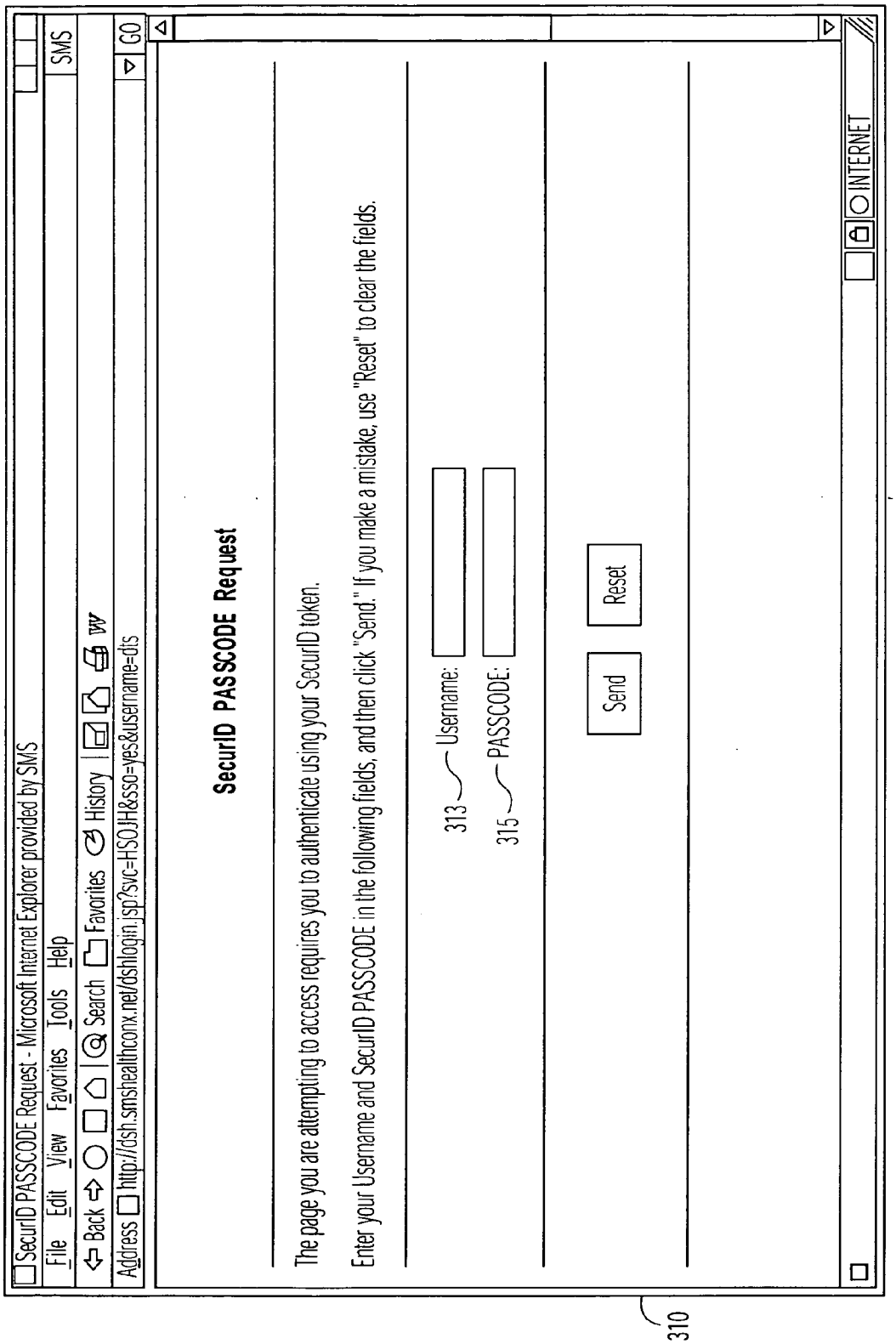

SecurID PASSCODE Request - Microsoft Internet Explorer provided by SMS
File Edit View Favorites Tools Help
SMS
Back Search Favorites History
Address http://dsh.smshealthconx.net/dshlogin.jsp?svc=HSOJH&sso=yes&username=dts
GO
SecurID PASSCODE Request
The page you are attempting to access requires you to authenticate using your SecurID token.
Enter your Username and SecurID PASSCODE in the following fields, and then click "Send." If you make a mistake, use "Reset" to clear the fields.
313
Username:
315
PASSCODE:
Send
Reset
INTERNET
310

FIG. 3

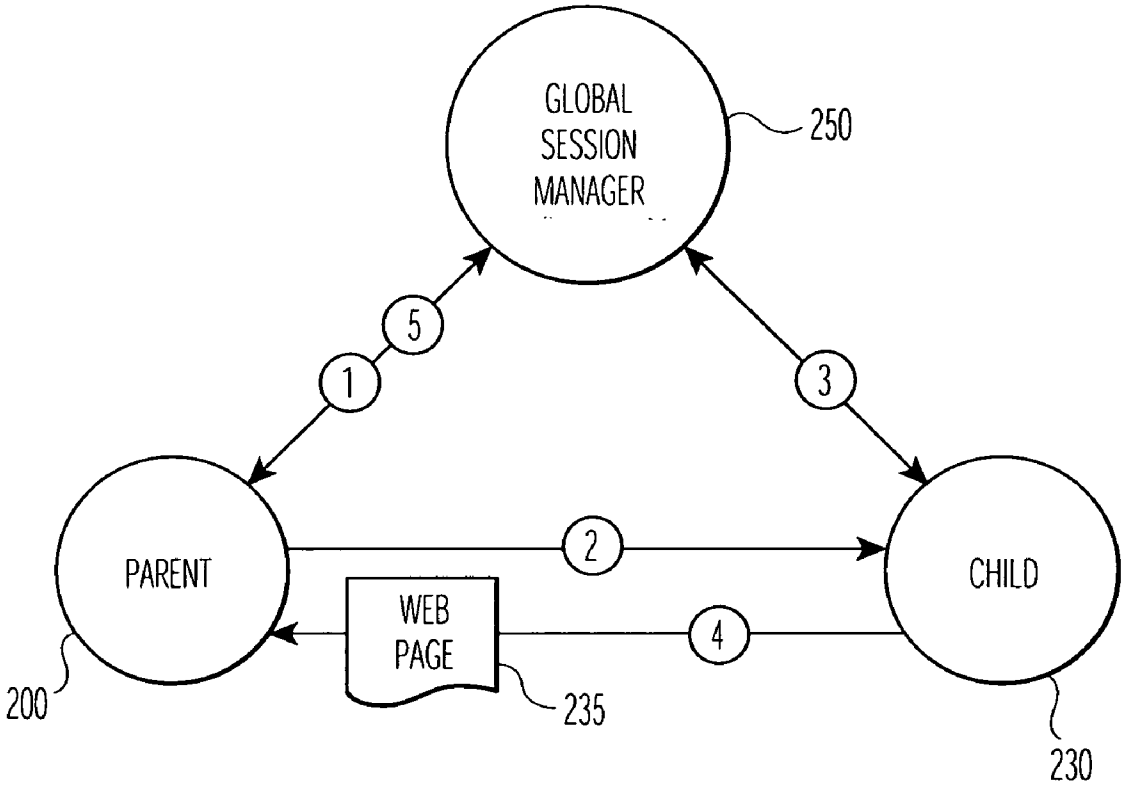

GLOBAL
SESSION
MANAGER
250
5
1
3
PARENT
2
CHILD
WEB
PAGE
4
200
235
230

FIG. 4

	NAME	IN/OUT	TYPE	DESCRIPTION
500	AuthServer	IN	STRING	AUTHENTICATING SERVICE NAME. THIS IDENTIFIES THE AUTHENTICATION SYSTEM (DATABASE) USED TO AUTHENTICATE THE USER. IT, ALONG WITH UserID MAY BE USED AS A KEY INTO A USER-MAPPING TOOL FOR MAPPING userIDs (FOR A GIVEN USER) ACROSS MULTIPLE SYSTEMS.
503	Language	IN	STRING	LANGUAGE AND COUNTRY IDENTIFICATION. THE INTERNET RFC 1766 (STANDARD FOR LANGUAGE CODES) IS TO BE USED.
505	LogoffURL	IN	STRING	URL USED TO REDIRECT THE USER TO A LOG-ON SCREEN WHEN A SESSION TIME-OUT HAS BEEN DETECTED. THE URL MUST BE FULLY QUALIFIED (NO RELATIVE ADDRESSING).
507	LogoffURL Target	IN	STRING	NAME OF FRAME TO BE TARGETED WHEN THE BROWSER IS REDIRECTED TO LogoffURL.
513	TIMEOUT	IN	INTEGER	SESSION YTIME-OUT IN SECONDS. IF OMITTED, TIMEOUT DEFAULTS TO 10 MINUTES
517	UserID	IN	STRING	USER IDENTIFICATION. THIS (ALONG WITH THE AuthServer PROPERTY) IS USED TO HELP IDENTIFY THE USER. THIS SHOULD BE THE UserID USED TO AUTHENTICATE THE USER. (SEE AuthServer).
520	SessID	OUT	STRING	UNIQUE SESSION IDENTIFIER ASSIGNED BY MANAGER 250
530	SessionKey	OUT	STRING	KEY TO BE USED TO ENCRYPT AND DECRYPT URL DATA.
525	SMResult	OUT	INTEGER	RESULT OF REQUEST. □ SUCCESS □ FAILURE

FIG. 5

	NAME	IN/OUT	TYPE	DESCRIPTION
600	SessID	IN	STRING	SESSION IDENTIFIER.
603	SMResult	OUT	INTEGER	RESULT OF REQUEST. □ SUCCESS □ FAILURE □ NOT FOUND

FIG. 6

	NAME	IN/OUT	TYPE	DESCRIPTION
700	SessID	IN	STRING	SESSION IDENTIFIER.
703	AuthServer	IN	STRING	STRING USED TO IDENTIFY THE USER DATABASE TO WHICH THE UserID IS ASSOCIATED.
705	UserID	IN	STRING	THE USER IDENTIFIER.
707	SMResult	OUT	INTEGER	RESULT OF REQUEST. □ SUCCESS □ FAILURE □ NOT FOUND □ TIME-OUT

FIG. 7

	NAME	IN/OUT	TYPE	DESCRIPTION
800	SessID	IN	STRING	SESSION IDENTIFIER.
803	AuthServer	IN	STRING	STRING USED TO IDENTIFY THE USER DATABASE TO WHICH THE UserID IS ASSOCIATED.
806	UserID	OUT	STRING	THE USER IDENTIFIER.
809	SMResult	OUT	INTEGER	RESULT OF REQUEST. □ SUCCESS □ FAILURE □ NOT FOUND □ TIME-OUT

FIG. 8

	NAME	IN/OUT	TYPE	DESCRIPTION
900	SessID	IN	STRING	SESSION IDENTIFIER.
903	AuthServer	OUT	STRING	THIS IDENTIFIES THE AUTHENTICATION SYSTEM (DATABASE) USED TO AUTHENTICATE THE USER.
907	LANGUAGE	OUT	STRING	LANGUAGE AND COUNTRY IDENTIFICATION. THE INTERNET RFC 1766 (STANDARD FOR LANGUAGE CODES) IS TO BE USED.
911	LogoffURL	OUT	STRING	URL USED TO REDIRECT THE USER TO A LOG-ON SCREEN WHEN A SESSION TIME-OUT HAS BEEN DETECTED. THE URL IS FULLY QUALIFIED (NO RELATIVE ADDRESSING). IF BLANK, THE APPLICATION SHOULD RETURN A BLANK PAGE TO THE BROWSER.
913	LogoffURL Target	OUT	STRING	NAME OF FRAME TO BE TARGETED WHEN THE BROWSER IS REDIRECTED TO LogoffURL. IF BLANK, NO TARGET FRAME SHOULD BE SPECIFIED
915	TIMEOUT	OUT	INTEGER	SESSION TIME-OUT IN SECONDS.
917	UserID	OUT	STRING	USER IDENTIFICATION USED TO AUTHENTICATE THE USER.
921	SessionKey	OUT	STRING	KEY TO BE USED TO ENCRYPT AND DECRYPT URL DATA.
923	SMResult	OUT	INTEGER	RESULT OF REQUEST. □ SUCCESS □ FAILURE □ NOT FOUND □ TIME-OUT

FIG. 9

	NAME	IN/OUT	TYPE	DESCRIPTION
449	SessID	IN	STRING	SESSION IDENTIFIER.
450	CallbackURL	IN	STRING	URL TO BE CONTACTED WHEN A SPECIFIED SESSION EVENT OCCURS. THIS CAN BE ANY VALID URL COMPLETE WITH PATH INFORMATION AND QUERY DATA. THE URL IS REFERENCED Y AS IT IS ENTERED IN THIS PROPERTY. NOTE: THE URL SPECIFIED HERE IS TO BE VALID IN THE NETWORK SEGMENT WHERE THE MANAGER THAT INVOKES THE CALLBACK OPERATES (NETWORK ADDRESS TRANSLATION MAY BE NECESSARY).
453	CallbackType	IN	INTEGER	TYPE OF EVENT THAT WILL TRIGGER THIS CALLBACK. ONLY SESSION TERMINATION EVENT IS CURRENTLY SUPPORTED. POSSIBLE VALUE ARE: 1 - END SESSION EVENT
445	SMResult	OUT	INTEGER	RESULT OF REQUEST □ SUCCESS □ FAILURE □ NOT FOUND □ TIME-OUT

FIG. 10

	NAME	IN/OUT	TYPE	DESCRIPTION
460	SessID	IN	STRING	SESSION IDENTIFICATION
463	SMResult	OUT	INTEGER	RESULT OF REQUEST □ SUCCESS □ FAILURE □ NOT FOUND □ TIME-OUT

FIG. 11

	NAME	IN/OUT	TYPE	DESCRIPTION
573	SessID	IN	STRING	SESSION IDENTIFIER.
577	ActivityInterval	OUT	INTEGER	SECONDS SINCE LAST ACTIVITY.
583	TIMEOUT	OUT	INTEGER	SESSION TIME-OUT IN SECONDS
589	SMResult	OUT	INTEGER	RESULT OF REQUEST. □ SUCCESS □ FAILURE □ NOT FOUND □ TIME-OUT

FIG. 12

	NAME	IN/OUT	TYPE	DESCRIPTION
653	ClearText	IN	STRING	THE STRING TO BE ENCRYPTED. EXAMPLE STRING: GSMHASH=4326&pid=17388902&patname=smith
656	SessionKey	IN	STRING	THE SHARED ENCRYPTION KEY TO BE USED. THIS IS THE *SessionKey* VALUE RETRIEVED THROUGH THE MANAGER 250 *GetSession* METHOD.
659	CipherText	Out	STRING	THE ENCRYPTED STRING. THIS IS USED TO VALUE THE DATA ELEMENT *GSM*. THE STRING IS ENCODED IN MIME 64 ENCODING FORMAT.
664	SMResult	Out	INTEGER	RESULT OF REQUEST. □ SUCCESS □ FAILURE

FIG. 13

	NAME	IN/OUT	TYPE	DESCRIPTION
751	CipherText	IN	STRING	THE ENCRYPTED STRING. THIS TYPICALLY COMES FROM THE DATA ELEMENT *GSM* AND IS IN MIME 64 ENCODING FORMAT.
753	SessionKey	IN	STRING	THE SHARED ENCRYPTION KEY TO BE USED. THIS IS THE *SessionKey* VALUE RETRIEVED THROUGH THE MANAGER 250 *GetSession* METHOD.
755	ClearText	OUT	STRING	THE RESULTING DECIPHERED STRING.
757	SMResult	OUT	INTEGER	RESULT OF REQUEST. □ SUCCESS □ FAILURE

FIG. 14

	NAME	IN/OUT	TYPE	DESCRIPTION
783	ClearText	IN	STRING	THE STRING TO BE HASHED.
787	HASH	OUT	STRING	THE RESULTANT HASH VALUE.
789	SMResult	Out	INTEGER	RESULT OF REQUEST. □ SUCCESS □ FAILURE

FIG. 15

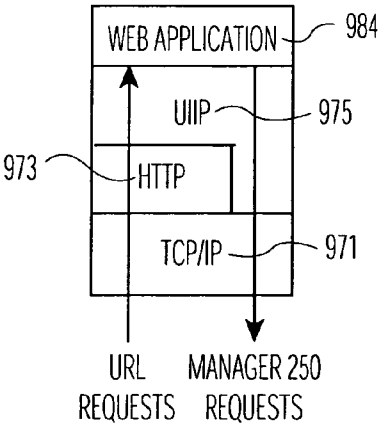


FIG. 16

SYSTEM AND USER INTERFACE SUPPORTING CONCURRENT APPLICATION INITIATION AND INTEROPERABILITY

[0001] This is a divisional application of non-provisional application Ser. No. 09/817,311 filed Mar. 26, 2001 of provisional application Ser. No. 60/261,148 by B. Royer filed Jan. 12, 2001.

BACKGROUND OF THE INVENTION

[0002] The management of information for medical purposes for use by physicians, hospital staff and other workers in the health care field poses a number of challenges. The information required by a physician, to optimize health care, is both varied in nature and in the sources from which it must be derived. A physician may typically need to have access to patient medical records, diagnostic images, diagnostic and dietary information systems, an appointment schedule, patient test results, medical literature, a prescription and drug interaction management system, insurance and billing information as well as a staff management system, for example. Access to such information and related services necessitate the use of a system including a communication platform supporting Internet operation and possibly local intra-net operation. Further, it is desirable that such a system for providing access to such an array of comprehensive information sources and related services should also provide a user interface that is suitable for use by a layman in the field and should not require extensive operator training.

[0003] There are a number of difficulties in providing such a comprehensive system. Specifically, it is necessary that such a system should support multiple different concurrent Internet based applications with the capability of conveying information between individual applications. These difficulties are compounded by the fact that individual applications may employ a unique data format or other operational feature limiting concurrent operation and interoperability. A system according to invention principles addresses these difficulties and derivative problems.

SUMMARY OF THE INVENTION

[0004] A system and associated communication protocol enables network (including Internet) compatible applications to be integrated into any process involving concurrent operation of applications. It does this by specifying the rules for conveying URL data and other data between applications and by employing a managing application and services to facilitate application session initiation and to provide common, essential session properties and by a variety of other mechanisms. A system employed by a first application for supporting concurrent operation of a plurality of network compatible applications involves an entitlement processor for authorizing user access to the first application in response to validation of user identification information. The system also involves a communication processor for communicating a session initiation request to a managing application to initiate generation of a session identifier particular to a user initiated session and for use by a plurality of concurrently operating applications to uniquely identify the user initiated session in response to validation of user identification information.

[0005] In another feature of the invention the managing application initiates generation of an encryption key particular to a user initiated session for common use by a plurality of concurrently operating applications in encrypting data associated with a personal record.

BREF DESCRIPTION OF THE DRAWING

[0006] **FIG. 1** shows a web browser window including multiple links to a plurality of medical related applications, according to invention principles.

[0007] **FIG. 2** is a system command flow diagram showing system protocol operation involving a managing application (GSM—Global Session Manager) two applications and a web browser, according to the present invention.

[0008] **FIG. 3** shows a common logon menu used for multiple applications, according to invention principles.

[0009] **FIG. 4** shows command interaction between multiple concurrently operating applications, according to invention principles.

[0010] **FIG. 5** shows the bidirectional command and response data communicated between an application and the managing application for initiating a session of user operation, according to invention principles.

[0011] **FIG. 6** shows the bidirectional command and response data communicated between an application and the managing application for terminating a session of user operation, according to invention principles.

[0012] **FIG. 7** shows the bidirectional command and response data used by an application for informing the managing application of a valid userid and associated authentication database, according to invention principles.

[0013] **FIG. 8** shows the bidirectional command and response data used by an application to retrieve a valid userid for a particular authentication database from the managing application, according to invention principles.

[0014] **FIG. 9** shows the bidirectional command and response data used by an application for retrieving session context data from the managing application, according to invention principles.

[0015] **FIG. 10** shows the bidirectional command and response data used by an application to provide the managing application with a URL to be accessed upon an event terminating the current user operation session, according to invention principles.

[0016] **FIG. 11** shows the bidirectional command and response data used by an application to inform the managing application of activity, according to invention principles.

[0017] **FIG. 12** shows the bidirectional command and response data used by an application to retrieve its timeout status data held by the managing application, according to invention principles.

[0018] **FIG. 13** shows the bidirectional command and response data used by an application to have a URL data string encrypted by the managing application, according to invention principles.

[0019] **FIG. 14** shows the bidirectional command and response data used by an application to have a URL data string decrypted by the managing application, according to invention principles.

[0020] **FIG. 15** shows the bidirectional command and response data used by an application to have a string hashed by the managing application, according to invention principles.

[0021] **FIG. 16** is a system hierarchical protocol layer diagram including an interoperability protocol, according to the invention principles.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

[0022] A system and associated protocol enables Internet compatible applications comprising any grouping of software to be integrated into a workflow capable of supporting a browser. Workflow, as used in this document, refers to a task sequence typically involving initiation, intermediate command operation and termination of Internet compatible applications via a displayed user interface occurring between a user logon and a user logoff command. The system involves a centralized session manager and protocol for passing URL data between applications and other functions. These include providing services to coordinate user inactivity timeouts and provide common, essential session properties for facilitating concurrent application operation for providing access to an array of comprehensive (medical and other) information sources and related services. Internet compatible applications employing this system may be dynamically re-organized to implement different workflows or task sequences involving different operational constraints and limitations. The system advantageously facilitates reuse and interoperability of web based applications in multiple different sequences and concurrent operation configurations.

[0023] The system addresses a variety of problems involved in supporting concurrent operation of Internet compatible applications for accessing multiple information sources and related services for medical and other purposes. As such, the system addresses the problems involved in maintaining concurrent operation of applications in a framework providing a common web browser-like user interface. The system specifically addresses problems involved in managing different inactivity timeout periods and in facilitating user initiation (e.g., logon), operation and termination (e.g., logoff) of multiple Internet applications and in securely passing URL, patient (and user) identification and other information between applications. A managing application is employed to coordinate user operation sessions. Specifically the managing application coordinates inactivity timeout operation and maintains and conveys properties between concurrent applications in order to create a smooth user operation session. For this purpose, the managing application also coordinates the use of a single logon screen common to multiple concurrent applications.

[0024] The principles of the invention may be applied to any system involving concurrent operation of different software applications. Further, although the disclosed system is described in the context of communicating and processing web page data and associated URLs (Universal Resource Locators), this is exemplary only. The system may process any form of data that may be communicated via Internet Protocol (IP) or HyperText Transmission Protocol (HTTP) from an Internet source and includes any form of packetized data including streamed video or audio data, telephone messages, computer programs, Emails or other communications, for example.

[0025] **FIG. 1** shows a web browser composite window **10** including multiple links to a plurality of medically related applications. The web browser provides typical command toolbars **43** and **44** as well as an application initiation bar (items **12-23**). The web browser interface permits a user to initiate multiple concurrent applications including, for example, an application providing an inpatient census window (e.g. for patients **25** and **27**) together with a laboratory test results application providing a results notification window including displayed items **29**, **31** and **33**. Other concurrent applications permit access to health care information and resources such as via reference link **37** and news item link **34**.

[0026] **FIG. 2** is a system command flow diagram showing system protocol operation involving a managing application **250** (GSM—Global Session Manager), two applications **200** and **230** (App1 and App2) and a web browser **10** (e.g. as described in connection with **FIG. 1**). The system protocol employed by manager **250** supports coherent harmonized and concurrent operation of multiple applications (e.g., applications **200** and **230**) in implementing a task sequence or workflow. Manager **250** is advantageously used by the applications **200** and **230** to reference global data that is essential to a workflow. Such global data includes, for example, user identification information, a shared key used for the encryption of URL data, and a common URL to be used for handling a logoff and logon function. The system protocol involves applications **200** and **230** intermittently notifying manager **250** of activity to prevent an inactivity timeout while a user is active in another concurrent application.

[0027] Manager **250** employs a system protocol for passing session context information to applications **200** and **230** via URL query or form data. The session context information comprises a session identifier, a hash value, and application specific data. The session identifier is used by applications **200** and **230** to identify a user initiated session in communicating with manager **250**. The hash value is used by applications **200** and **230** to validate that a received URL has not been corrupted, intentionally or otherwise. The application data portion of the session context information may or may not be encrypted as determined by the application communicating the URL. The application specific data is tailored to meet the intended function of a target application. The protocol employed by manager **250** supports applications that use the generated session context information and do not alter it. In alternative embodiments, applications **200** and **230** may employ internal managers using other protocols to support a global context concept, either as an alternative to manager **250**, or in addition to manager **250**. Such other protocols comprise, for example, HL7 (Health Level Seven) protocol or CCOW (Clinical Context Object Workgroup V1.2 Ratified May 2000) protocol. The described system supports use of alternative protocols as well as the communication of data between applications, other than just session context information.

[0028] Manager **250** maintains security by operating in a secure environment that prevents unauthorized access to the manager application itself. Security is also provided by ensuring applications **200** and **230** (that communicate with manager **250**) also operate in a secure environment. Manager **250** also maintains security by detecting and ignoring

received URLs that have been intentionally or otherwise corrupted and by preventing replay and display of received URLs.

[0029] **FIG. 4** shows command interaction between concurrently operating applications **200** and **230**, web browser **235** and manager **250** using a system interoperability protocol in accordance with invention principles. **FIG. 4** shows a simplified version of the command interaction of **FIG. 2**. In an exemplary user operation session, parent application **200** starts a session and notifies manager **250** of activity (1). Subsequently, parent application **200** references a child application **230** (2). A child application typically provide web pages to other applications. Specifically, child application **230** notifies manager **250** of activity (3) and returns a web page **235** to parent application **200** (4). Eventually, parent application **200** terminates the session via a command to manager **250** (5). The application that establishes a session with manager **250** is said to be the parent application. All additional applications that participate in that session are referred to as child applications. The collection of the parent and all child applications together are said to be the participants. Manager **250** provides centralized services to the parent and child applications in order to coordinate them. A parent application creates a session after the user is authenticated and before a child application is referenced. A parent application may delay establishing a session until a specific event, e.g., until the parent downloads (to a browser) a web page containing links to child applications. Typically, a session is ended when the user signs off or when the user times out due to inactivity.

[0030] Returning to **FIG. 2**, a command **220** to initiate parent application **200** and to authenticate user identification information (password and user name entered via browser command **203**) is passed via browser **10** to parent application **200**. Parent application **200** authenticates the received user information using either an internal or external database of authentication information for verification. Upon successful authentication, parent **200** communicates startsession **222** data (as shown in **FIG. 5**) to manager **250** to establish a new session. The startsession command is invoked prior to generating links to another application. **FIG. 5** shows the startsession bidirectional command and response data communicated between parent application **200** and manager **250** for initiating a session of user operation. Parent **200** determines the **FIG. 5** properties comprising authserver **500**, language **503**, logoffurl **505**, logoffurltarget **507**, timeout **513** and userid **517**. In response, manager **250** returns properties (in command **224** of **FIG. 2**) to parent **200** comprising, a unique session identifier **520** to be used when referencing child application URLs and manager **250**, session key **530** (used to encrypt and decrypt URL data) and SMResult **525** indicating command status. A failure indication by SMResult **525** indicates that the service is unavailable possibly due to a temporary condition (e.g. network problems) or to a permanent condition (e.g. a configuration error). None of the **FIG. 5** properties are mandatory. In an alternative embodiment, manager **250** returns session identifier **520** in encrypted form for decryption and use by application **200** using a predetermined encryption key available to applications including applications **200** and **230** and manager **250**.

[0031] Through the startsession command, parent **200** sets static session properties that are to be used by all participants of the session. This, in affect, means that parent application **200** sets the tone and control of the user experience. For example, the parent determines the logon/logoff web page to be used and how the user is known to the participants as well as the user inactivity limits. It is the responsibility of parent application **200** to ensure that proper authentication has been carried out before it creates a session.

[0032] In the command interaction diagram of **FIG. 2**, parent application **200** registers a URL with manager **250** using a register callback command (indicated as item **226** in **FIG. 2** and detailed in **FIG. 10**). Specifically, parent **200** provides manager **250** with a callback URL (Universal Resource Locator **450** in **FIG. 10**) defining a web page to be accessed upon an event terminating the current user operation session. The callback URL is provided to manager **250** together with session identifier **449** and callback type indicator **453** (**FIG. 10**). The register callback command is also employed by manager **250** to update an activity status time indicator used to monitor parent application **200** activity status. In response to the register callback command, manager **250** returns a command status indicator (SMResult **445** of **FIG. 10** indicated by item **226** of **FIG. 2**) to parent **200**. Indicator **445** indicates status such as command success, failure, time out or a not found condition. A failure result indicates that the service is unavailable possibly due to a temporary condition (e.g. network problems) or to a permanent condition (e.g. a configuration error). A time out condition indicates that the session has timed out and parent application **200** redirects browser **10** to the URL found in the logoffurl property targeted to the frame found in the logoffurltarget property (items **505** and **507** of **FIG. 5**). A not found condition indicates that manager **250** has no record of the requested session ID. In response, parent application **200** displays a message indicating that the session is no longer active and that the user should navigate to the logon screen to restart.

[0033] In the command interaction diagram of **FIG. 2**, parent application **200**, in command **229**, returns a web page to browser application **10** following register callback command (**226**). The web page includes embedded URL links to other (child) applications including a URL link to (child) application **230**. The embedded URL links are processed, to include within the URL data itself, the session identifier and additional context information if required (e.g., a patient identifier). Application **230** uses the session identifier in communicating with managing application **250** in order to obtain information about the session for facilitating user operation and task workflow. Manager **250** identifies an authenticated user of child application **230** from previously stored identification data eliminating the need for a user to logon again to access child application **230**. This constitutes a silent logon process. A child application providing access to other child applications generates URL links (to the other child applications) incorporating the session identifier and additional context information as required. A parent or child application providing access to its own web pages may optionally pass the session identifier or context via a URL. An application creates a session via a startsession command when, for example, it create a URL reference to a different application and a session identifier has not yet been established. If a session identifier has already been established and provided to the application, it is used and propagated when referencing other applications.

[0034] The maintenance of security is an important consideration in processing the embedded URL links for incorporation in the web page returned to browser **10** in command **229** of **FIG. 2**. Also, different security concerns are involved in processing URL links to include the session identifier and additional context information. One concern is the prevention of unauthorized access to manager **250'**. Other concerns include the prevention of URL replay once a session has ended and provision of protection against attacks involving corruption of URL data.

[0035] The prevention of unauthorized access to manager **250** is facilitated by ensuring that applications **200** and **230** that access manager **250** operate in a protected and trusted environment. This is attained by operating the system on proprietary networks (e.g. Wide Area Networks—WANs or Local Area Networks—LANs) and intra-nets. Applications that are accessed through public networks such as the Internet are designed so that components accessing manager **250** operate in a trusted environment. Additional levels of security are attainable by employing additional protocol layers, e.g., involving digitally signed software, user certificates, or SSL (Secure Sockets Layer V3.0) protocol of November 1996 by Internet Engineering Task Force (IETF), etc.

[0036] The prevention of URL replay and protection against URL corruption (deliberate or otherwise) is advantageously achieved by the use of strong encryption, as well as limited duration sessions and randomly generated shared encryption keys. These security measures ensure that URL query or form data is unaltered and that a link generated by a participant application (e.g., applications **200** and **230**) is not redirected. The security measures also ensure that a valid URL is not replayed after a session has ended and that a session identifier is difficult for an unauthorized party to determine.

[0037] The creation of a session, e.g., via startsession command **226** of **FIG. 2**, initiates the generation of an encryption key (**530** of **FIG. 5**) for that session by manager **250** that is returned to application **200** by command **224** of **FIG. 2**. This key is generated on a random basis and is shared for the duration of that session with those applications that are participants of the session (applications **200** and **230** in the example of **FIG. 2**). The key is used by an encryption algorithm employed by application **200** to encrypt the query or form data portions of a generated URL link (e.g. a URL link to child application **230**) that is incorporated in a web page provided by command **229** for display using browser **10**. The encryption method is designed so that if any of the encrypted string is changed, it prevents the string from being successfully decrypted. In addition, manager **250** uses a unique session identifier (SID—item **520** of **FIG. 5**) for each new session. This procedure protects against the corruption and replay of a URL (once the session has ended) and prevents the generation of valid URL data from anywhere but participant applications.

[0038] In addition, application **200** ensures that a URL link (e.g. a URL link to child application **230**) embedded in a web page provided for display using browser **10** is not redirected. For this purpose, application **200** generates a hash value from the domain, path, program, and program data portion of the URL. Application **200** (as the sending application) generates a hash value from the fully qualified URL link. Specifically, application **200** generates the hash value from the URL data either lying between the "http://" and the question mark "?" or from the data lying between the "http://" and the pound/number sign "#"—whichever comes first. The data used to create the hash value is the fully qualified URL even if relative addressing is used in the actual reference (e.g. in the HREF or ACTION attributes). In addition, the string to be hashed is advantageously converted to lower case before being hashed. This is done since it is recognized by the inventors that some web servers and browsers may not preserve case or may be sensitive to case. Further, the hash algorithm used in both the sending application **200** and receiving application **230** in this described embodiment is the RSA MD5 128 bit hashing algorithm. The RSA MD5 algorithm is the Rivest Shamir Adleman public key cryptosystem used with MD5 hash function and is described in publications available on the Internet. However, this is exemplary only and other algorithms or other mechanisms performing a function similar to that of the hashing function may also be employed. The created hash value is conveyed as part of the embedded URL link referencing child application **230** in the web page data provided by application **200** to browser **10**.

[0039] In another embodiment, application **200** does not create the hash value itself but uses manager **250** to create the hashed value. Specifically, application **200** communicates with manager **250** using the bidirectional command and response data of **FIG. 15** to have a string hashed. Application **200** sends the string to be hashed **783** (**FIG. 15**) to manager **250** which responds with the hashed value **787** and the status of this command (SMResult **789**) indicating command success or failure.

[0040] Upon user selection of the embedded URL link via browser **10**, the URL data and hash value are passed to application **230** via command **234**. A user selects the embedded URL link (browser command **207**) on browser **10** to view a patient laboratory results, for example. Application **230** decrypts the received hash value for comparison with a corresponding hash value independently generated from corresponding URL data retrieved from a web server. The corresponding URL data indicates the true URL link data and is therefore usable for verification. Application **230** independently generates a hash value from corresponding URL data comprising a concatenation of the following HTTP server-side attributes (or equivalents). For ASP applications (ASP means Active Server Page—a Microsoft proprietary style of processing web requests) the SERVER_NAME and SCRIPT_NAME are used. For non-ASP applications, e.g. for CGI (Common Gateway Interface) compatible applications the SERVER_NAME, SCRIPT_NAME, and PATH_INFO are used. The independently generated hash value and the hash value received by application **230** from application **200** via browser **10** are compared and if they are not equal, the request to initiate application **230** is rejected.

[0041] Applications are vulnerable to the corruption of URL data and the context information conveyed within the URL data. The URL data conveyed from application **200** to application **230** includes context information comprising a session identifier and optionally a user or patient identifier. This URL data is potentially vulnerable to corruption to cause URL replay or redirection of an application to a

substitute address or to gain access to application functions and parameters for unauthorized purposes. In order to protect against such corruption and to ensure that the entity being accessed is the one originally targeted, portions of the URL data conveyed between applications are advantageously encrypted.

[0042] Application **200** processes a URL for incorporation within web page data for communication to browser **10** and for communication to application **230** by concatenating the URL fields into one string prior to encryption using a key previously received from manager **250**. The processed URL data includes a session identifier and encrypted data. A processed URL includes a field labeled as a GSM data field. The format of the field is as follows:

GSM=x:y

Where:

[0043] GSM—is the key name of the "key=value" pair

[0044] x—is the manager **250** session identifier

[0045] :—is the field separator

[0046] y—is the encrypted string

The session identifier (x) is the session identifier (item **520** of **FIG. 5**) provided by manager **250** in the startsession command-response **224** (**FIG. 2**): In its unencrypted form, the encrypted string (y) is itself made up of "key=value" pairs. A URL hash value (identified in a URL by the label GSH) is incorporated in a processed URL and is generated by hashing on the addressable portion of a fully qualified URL. The addressable portion of the URL is hashed to compress and reduce the quantity of data to be processed. Other x:y data field pairs may be included to accommodate requirements of particular applications and other application data may also be conveyed with a URL. However this other data is not encrypted within the GSM data field. Application data that is to be encrypted with the encryption key from manager **250** is placed into the encrypted portion of the GSM data field. The GSM data field is compatible with the Uniform Resource Identifiers (URI) syntax authored by the Internet Engineering Task Force (IETF) in Request for Comment document RFC 2396. The RFC documents are available via the Internet and are prepared by Internet standards working groups.

[0047] The URL processing performed by application **200** (and performed by other applications passing context data) is illustrated below. An exemplary URL string that is to be processed is,

[0048] www.smed.com/altoona/prd/results.exe/1?GSM=16253384937&GSH=24017 &Pid=1772693&Frgclr=blue

The data field GSM is the session identifier provided by manager **250**. The data field GSH is the hash value derived by application **200**. The derived hash value is advantageously encrypted by application **200** prior to its communication within processed URL data to other applications (e.g. application **230**). Additionally, in this example application **200** encrypts a patient identifier (Pid) for communication to application **230**. The system protocol employed by manager **250** (and applications **200** and **230**) determines that data to be encrypted is collated into an individual MIME (Multipurpose Internet Mail Extension) format data field for encryption into one string. Therefore, as an example, the string

[0049] GSH=24017&Pid=1772693

is encrypted into the string

[0050] 16sfdjwhejeyw7rh3hekw

[0051] Application **200** encrypts the string using a two-key triple DES (Data Encryption Standard) algorithm in cipher block chaining mode employing a 64 bit block and an effective 112 bit key length. The resulting cipher text complies with the URL query data encoding format and represents a single value of a "key=value" pair. Although in this embodiment application **200** performs the encryption, in an alternative embodiment application **200** (or a child application) instructs manager **250** to perform the encryption. For this purpose, application **200** uses the bidirectional command and response data of **FIG. 13** to have a URL data string encrypted into a MIME formatted string by manager **250**.

[0052] In the embodiment in which the string is encrypted by manager **250**, application **200** sends the string to be encrypted cleartext **653** together with the encryption key to be used, sessionkey **656** (**FIG. 13**), to manager **250**. Manager **250** encrypts string **653** using key **656** and responds to application **200** with the encrypted string ciphertext **659** and command status indicator SMResult **664**. SMResult **664** indicates command success or failure in a similar fashion to that previously described in connection with command **226**, for example.

[0053] The session identifier and the encrypted text are concatenated into one field (per the manager **250** protocol) and the resulting processed URL data is:

[0054] www.smed.com/altoona/prd/results.exe/1?GSM=16253384937:16sfdjwhejeyw7rh3hekw&Frgclr=blue

[0055] Application **230** decrypts the encrypted portion of the processed URL data to provide the hash value for use in validating that application **230** is the intended recipient of the URL request. Application **230** compares the hash value received in the URL data with a corresponding hash value independently generated from corresponding URL data retrieved from a web server. If the two hash values do not match, the addressable portion of the URL has been altered and the request is rejected. In addition, application **230** rejects received URL data that does not contain at least an encrypted GSH portion. In other embodiments the context data (session identifier and patient identifier) may not be encrypted or may be conveyed within URL data in both encrypted and non-encrypted form. In the latter case, an application automatically parses received URL data to identify encrypted data elements and uses the encrypted data elements in preference to corresponding non-encrypted versions of the data elements. Further, although this example shows context and security data being passed as URL query data, this is exemplary only. The context and security (and other) data may be passed in another format, for example, the "GSM=value" data may be conveyed as form data via a Form and POST method. The Form and POST method is known in the art. Further, the URL processing functions

described herein may also be employed by a browser application or a managing application in other embodiments.

[0056] Although in this embodiment application **230** decrypts the encrypted portion of the processed URL data to provide the hash value, in an alternative embodiment application **230** (or another parent or child application) instructs manager **250** to perform the decryption. For this purpose, application **230** uses the bidirectional command and response data of **FIG. 14** to have an encrypted URL (MIME formatted) data string decrypted by manager **250**. For this purpose, application **230** sends the string to be decrypted ciphertext **751** together with the encryption key to be used, sessionkey **753** (**FIG. 14**), to manager **250**. Manager **250** decrypts encrypted string **751** using key **753** and responds to application **230** with the decrypted string cleartext **755** and command status indicator SMResult **757**. SMResult **757** indicates command success or failure in a similar fashion to that previously described in connection with command **226**, for example.

[0057] Upon successful validation of the received hash value following command **234** (**FIG. 2**) to initiate application **230**, application **230** communicates with manager **250** via command **233** to obtain session context information. Specifically, application **230** employs the bidirectional command and response data of **FIG. 9** to retrieve session context data from manager **250**. Application **230** sends session identifier **900** (**FIG. 9**) to manager **250** in command **233**. Manager **250** responds in command **237** with **FIG. 9** properties comprising authserver **903**, language **907**, logoffurl **911**, logoffurltarget **913**, timeout **915** and userid **917**, session key **921** and SMResult **923**. These properties have a similar function as the corresponding properties previously described in connection with **FIG. 5**. Further, manager **250** (in step **280** of **FIG. 2**) updates a session activity time stamp maintained for application **230** and used to monitor application **230** activity in response to successful execution of commands **233** and **237** to retrieve session context information.

[0058] SMResult **923** is a status indicator that indicates command success, failure, time out or a not found condition. A failure result indicates that the service is unavailable possibly due to a temporary condition (e.g. network problems) or to a permanent condition (e.g. a configuration error). A time out condition indicates that the session has timed out resulting in manager **250** returning to application **230** the logoffurl property targeted to the frame found in the logoffurltarget property (items **911** and **913** of **FIG. 9**). In this condition the other property values are not valued and application **230** redirects browser **10** to the URL found in the logoffurl property targeted to the frame found in the logoffurltarget property. A not found condition indicates that manager **250** has no record of the requested session identifier. In response, application **230** displays a message indicating that the session is no longer active and that the user should navigate to the logon screen to restart.

[0059] Application **230**, in command **239**, returns a web page including patient laboratory results (previously requested via browser command **207**) for display via browser application **10**. The laboratory results web page incorporates an embedded link to view a web page provided by the same application (application **230**) displaying a list of orders for laboratory tests. Therefore, the user selection of the embedded link to view laboratory test orders in browser command **211** represents an intra-application command request. Further, because the link to the test results page is an intra-application link there is no requirement for this particular embedded link to be processed in the manner previously described to incorporate the session identifier and other context information. Application **230** may employ its own mechanism for maintaining state between intra-application URL link selection and the provision of associated web pages. Alternatively, application **230** may optionally process the embedded intra-application URL link, in the manner previously described, in order to convey session identification and other context information between the different intra-application web page access states.

[0060] Application **230**, in command **247**, notifies manager **250** of activity using the bidirectional command and response data of **FIG. 11** and in response (in step **283** of **FIG. 2**) manager **250** updates an inactivity timing status indicator maintained for application **230**. Command **247** in **FIG. 2** is representative only and in practice notification commands are advantageously repeated until the session is ended in order to prevent an application timing out due to inactivity. In other embodiments application **230** may use different schemes for notification. However, it is the responsibility of application **230** (or other participant applications including parent application **200**) to notify manager **250** of activity in such a manner as to keep a session from timing out provided that the application is active. Application **230** sends session identifier **460** (**FIG. 11**) to manager **250** in command **247**. Manager **250** responds in command **247** with a command status indicator SMResult **463** indicating success, failure, not found or time-out in a similar fashion to that previously described in connection with command **237**, for example.

[0061] Application **230** may perform notifications using the command of **FIG. 11** when it is called upon (e.g. when a web page is provided) or application **230** may notify manager **250** of activity using batched notification commands. Such a batched notification may be advantageously used in the case of a PC application that monitors key and mouse movements and periodically notifies manager **250** with a single notification, for example. The command of **FIG. 11** is used by either a parent application or a child application in order to update activity status and, in response, manager **250** records the time at which it was notified. In addition, other commands may be used to notify manager **250** of activity. Specifically in this embodiment the commands depicted in **FIGS. 9 and 10** (corresponding to commands **233** and **226** of **FIG. 2** respectively) are used to notify manager **250** of activity.

[0062] In the described system, manager **250** acts as a central coordinator for monitoring activity of participant applications by requiring that participant applications notify manager **250** of activity. Participant applications may also query manager **250** in order to determine if a session is still active. This coordination and management function advantageously addresses problems involved in managing disparate inactivity time-out limits when different applications are combined into a single user task sequence and workflow. For example, such a problem arises if a user works in one application for a period of sufficient duration to trigger a time-out in another concurrent application. This situation leads to a loss of context and a confusing workflow.

[0063] Manager **250** is essentially passive in this embodiment and does not automatically generate a timeout event. Manager **250** maintains a time stamp for a session indicating when the session last had activity reported and initiates termination of a session if a session having an expired time stamp is referenced (e.g., via command **233**) or if a session termination command is received. Upon termination of a session, manager **250** informs those participant applications of the termination that have previously requested a notification of session termination. In other embodiments, manager **250** may be configured to automatically generate a timeout event upon one or more predetermined conditions or combination of conditions.

[0064] In addition, an application (e.g. applications **200** or **230**) uses the bidirectional command and response data format of **FIG. 12** to retrieve its timeout status data maintained by manager **250**. An application sends session identifier **573** (**FIG. 12**) to manager **250** and manager **250** responds with activityinterval **577**, timeout indicator **583** and SMResult **589**. The activityinterval **577** indicates the number of seconds since the last activity update, timeout indicator **583** identifies the time-out limit in seconds and status indicator SMResult **589** indicates command status. SMResult **589** indicates success, failure, not found or time-out in a similar fashion to that previously described in connection with command **237**, for example.

[0065] Application **230**, in command **249**, returns a web page including laboratory test orders (previously requested via browser command **211**) for display via browser application **10**. Subsequently, the user elects to logoff via browser command **213** and in response browser **10** issues a logoff command **253** to application **230**. Application **230**, in command **255**, instructs manager **250** that the session of user operation is to be terminated using the bidirectional command and response data of **FIG. 6**. Specifically, application **230** sends session identifier **600** (**FIG. 6**) to manager **250** in command **255** and in response (in step **285** of **FIG. 2**), manager **250** updates status indicators it maintains to reflect that session **600** is terminated. Manager **250** also responds to application **200** and to application **230** in command **257** with a command status indicator SMResult **603** identifying that the session is terminated and indicating status of the session termination command request. Specifically, SMResult **603** indicates success, failure, not found or time-out in a similar fashion to that previously described in connection with command **237**, for example. However, applications receiving SMResult **603** assume the session is terminated irrespective of the nominal SMResult **603** status.

[0066] Upon receipt of SMResult **603** from manager **250**, application **230** in command **259** redirects browser **10** (command **215**) to access a logon web page available via the logoffurl property targeted to the frame found in the logoffurltarget property previously obtained from manager **250** in command **237**. Specifically, browser **10** accesses a logon page at the logoffurl provided by parent application **200** via command **263** and this logon page is returned to browser **10** in command **273**.

[0067] Manager **250** may also be instructed to terminate a session under conditions other than in response to a user initiated logoff command. Such conditions occur, for example, when (a) an inactivity time out limit is exceeded and (b) an application becomes non-responsive to manager **250** initiated polling. Such polling is initiated by manager **250** periodically to clean up a session operation and to eliminate non-responsive applications that have avoided inactivity time out through a software error or other occurrence.

[0068] **FIG. 3** shows a common logon menu web page **310**, supporting entry of username **313** and password **315**, and enabling a single logon to provide access to multiple applications facilitating smooth workflow operation for a user. This is achieved by providing a logon menu web page as a common starting place to which a user is directed upon initial logon or upon logoff from a session of activity or upon a termination condition such as an error condition or upon time-out due to inactivity. For this purpose, manager **250** maintains a logoffurl property (e.g. item **505** of **FIG. 5**) provided by a parent application (e.g. application **200**) that is used by participant applications to get back to a common starting web page. Upon a participant application receiving a time-out status from manager **250** or when manager **250** explicitly ends a session, a browser (e.g., browser **10** of **FIG. 2**) is redirected to the logon page at the URL specified in the logoffurl property maintained by manager **250**.

[0069] A number of problems involved in providing a common logon web page are advantageously recognized. Specifically, problems are involved in enabling a user to logon once to access different applications with different user identifiers and different authentication systems. These problems are addressed in the present system by providing participant applications of a session with a list of AuthenticatingSystem-user ID pairs. A parent application (application **200**) provides an authenticating service identifier and user identifier used by the parent application to authenticate the user via the authserver and userid properties (items **500** and **517** respectively) of the session initiation command of **FIG. 5**. This information is provided for use by participant applications to identify the parent application and to identify how the user is known by the parent application if necessary.

[0070] A participant application (parent or child) may also use the bidirectional command and response data of **FIG. 7** to inform manager **250** of a valid userid and associated authentication database in a user operation session. Thereby, manager **250** compiles a database for mapping a userid of a participant application to an authenticated and different userid of a second application. This enables the second application to be accessed transparently to a user without the need for a user to re-login. An application (e.g., application **200** of **FIG. 2**) sends session identifier **700** (**FIG. 7**), authserver **703** and userid **705** to manager **250**. Authserver **703** identifies the user authentication database associated with userid **705**. Thereby, manager **250** compiles a database mapping userid to authentication service identifier that supports user authentication by subsequently accessed applications (e.g., application **230**). Manager **250** responds with command status indicator SMResult **707**. SMResult **707** indicates command success, failure, not found or time-out in a similar fashion to that previously described in connection with command **237**, for example.

[0071] A participant child application uses the bidirectional command and response data of **FIG. 8** to retrieve a valid userid for this particular child application and for an associated authentication database from the manager **250**. A child application (e.g., application **230** of **FIG. 2**) sends

session identifier **800** and authserver (authentication service identifier) **803** (**FIG. 8**) to manager **250**. Manager **250** responds to application **230** with userid **806** valid for the corresponding Authserver **803** used by application **230** and also returns command status indicator SMResult **809**. SMResult **809** indicates command success, failure, not found or timeout in a similar fashion to that previously described in connection with command **237**, for example. Child application **230** may also employ the authenticating system ID and user ID provided by parent application **200** for validation purposes since these properties are stored by manager **250** in its database. Application **230** obtains the appropriate userid **806** to be used with its authentication service (identified by item **803**) to enable access to a user without the need for a user to re-login. This involves participant application **230** relying on previous authentication of the user by parent application **200**. In an alternative embodiment, application **230** authenticates the user using a separate authentication process which may be the same or different to the service employed by application **200**.

[**0072**] The data format employed for the authserver identifier **803** is configurable within a participant application but ideally conforms to a standard among the participant applications in order to minimize proliferation of different identifiers for a common user within the manager **250** database mapping.

[**0073**] **FIG. 16** is a system protocol diagram indicating the hierarchical organization of communication protocol layers used by applications **200** and **230** for communication with browser **10** and manager **250** (**FIG. 2**). Applications **200** and **230** together with browser **10** and manager **250** provide access to medical information and related services in a system including a communication platform supporting Internet operation and local intra-net operation. The system may also involve other networks including Local Area Networks (LANs), Wide Area Networks (WANs) and other dedicated hospital networks or other medical (or other) systems and communication networks.

[**0074**] An application (e.g., applications **200** and **230**) residing in web application layer **984** communicates with manager **250** using a User Interface Interoperability Protocol (UIIP) data format **975** comprising command data structures presented in **FIGS. 5-15**. The UIIP command and response data **975** involves the TCP/IP (Transmission Control Protocol/Internet Protocol) layer **971**. Applications **200** and **230** use the UIIP **975** and TCP/IP **971** layers in communicating with manager **250** in commands **222**, **224**, **226**, **233**, **237**, **247** and **255** as illustrated in **FIG. 2**. Manager **250** also communicates with applications **200** and **230** using HTTP and TCP/IP protocol as exemplified in command **257** of **FIG. 2**. Browser **10** and applications **200** and **230** communicate using (TCP/IP and) HTTP format URL data strings processed in accordance with the UIIP as previously explained and indicated on **FIG. 2**.

[**0075**] The architectures and processes presented in **FIG. 2**, **FIG. 4** and **FIG. 16** are not exclusive and the data formats of **FIG. 5-15** are also adaptable to accommodate different elements and properties. Other architectures and processes may also be derived in accordance with the principles of the invention to accomplish the same objectives. Further, the communication processes and steps of **FIG. 2** and data formats of **FIG. 5-15** may be implemented on different platforms for different functions and may be applied within the applications internal to a processing device such as a PC or other processing device or system. The communication processes and data formats may also be applied for Internet or intra-net (or any other network) based work flow or task implementation. The inventive principles may be employed in any system involving the concurrent operation of different applications.

What is claimed:

1. A system employed by a first application for supporting concurrent operation of a plurality of network compatible applications, comprising:

an entitlement processor for authorizing user access to said first application in response to validation of user identification information; and

a communication processor for initiating generation of,

a session identifier particular to a user initiated session and for use by a plurality of concurrently operating applications to uniquely identify said user initiated session and

an encryption key for use by said first application in encrypting personal record parameters conveyed in URL data,

in response to validation of user identification information wherein

said communication processor includes said personal record parameters in different types of command messages communicated to applications of said plurality of concurrently operating applications.

2. A system employed by a first application for supporting concurrent operation of a plurality of network compatible applications, comprising:

an entitlement processor for authorizing user access to said first application in response to validation of user identification information; and

a communication processor for initiating generation of,

a session identifier particular to a user initiated session and for use by a plurality of concurrently operating applications to uniquely identify said user initiated session and

an encryption key for use by said first application in encrypting personal record parameters conveyed in URL data and said communication processor includes said includes encrypted personal record parameters in different types of command messages communicated to applications of said plurality of concurrently operating applications.

3. A system according to claim 2, wherein

said encryption key is for common use by said plurality of concurrently operating applications in encrypting data associated with a personal record in data fields of a URL string.

4. A system according to claim 1, wherein

said communication processor also communicates additional parameters to a managing application for storage, said additional parameters including one or more of, (a) an authentication service identifier, (b) a language identifier, (c) a URL to direct a browser to a starting

application upon termination of a session, (d) a URL for use in acquiring a web page providing a logon menu to support user initiation of another session, (e) a URL to be contacted upon a predetermined event and (f) an identification of a type of said predetermined event.

5. A system according to claim 1, including

an input processor for receiving said session identifier and an associated encryption key from said managing application.

6. A system according to claim 5, including

an encryption processor for use in encrypting medical data associated with a patient medical record.

7. A system employed by a managing application for supporting concurrent operation of a plurality of network compatible applications, comprising:

an input processor for receiving from a first application a session initiation request to initiate generation of a session identifier;

a session identifier generator for generating a session identifier particular to a user initiated session and for use by a plurality of concurrently operating applications to uniquely identify said user initiated session;

an encryption key generator for generating an encryption key for use by said first application in encrypting personal record parameters conveyed in URL data, said encryption key generator randomly generating an encryption key particular to said user initiated session, in response to said session initiation request, said encryption key being for common use by said plurality of concurrently operating applications in encrypting data associated with a personal record; and

a communication processor for including at least one of, (a) encrypted personal record parameters and (b) said session identifier, in different types of command messages communicated to applications of said plurality of concurrently operating applications.

8. A system according to claim 7, wherein

said encryption key generator randomly generates an encryption key particular to said user initiated session, in response to said session initiation request.

9. A system according to claim 8, wherein

said encryption key is for common use by said plurality of concurrently operating applications in encrypting data associated with a personal record.

10. A system employed by a managing application for supporting concurrent operation of a plurality of network compatible applications, comprising:

an input processor for receiving from a first application a session initiation request to initiate generation of a session identifier;

a session identifier generator for generating a session identifier particular to a user initiated session and for use by a plurality of concurrently operating applications to uniquely identify said user initiated session; and

a communication processor for,

communicating said session identifier to said first application and

including said generated session identifier in different types of command messages communicated to applications of said plurality of concurrently operating applications.

11. A system supporting concurrent operation of a plurality of Internet compatible applications, comprising:

a browser application providing a user interface display permitting user entry of identification information and commands for a plurality of Internet compatible applications and for providing user identification information to a first application for validation;

a managing application for generating,

a session identifier particular to a user initiated session and

an encryption key for use by said first application in encrypting personal record parameters conveyed in URL data, in response to receiving a session initiation request from a first application and for communicating said session identifier to said first application; and

a communication processor for including at least one of, (a) encrypted personal record parameters and (b) said session identifier, in different types of command messages communicated to applications of said plurality of concurrently operating applications.

12. A system according to claim 11, wherein

said encryption key is also to be used in encrypting and decrypting a session identifier conveyed in URL data.

13. A system according to claim 11, wherein said managing application includes

said encryption key generator randomly generates an encryption key particular to said user initiated session, in response to said session initiation request.

14. A system according to claim 13, wherein

said encryption key is for common use by said plurality of concurrently operating applications in encrypting data associated with a personal record.

15. A system supporting concurrent operation of a plurality of Internet compatible applications, comprising:

a browser application providing a user interface display permitting user entry of identification information and commands for a plurality of Internet compatible applications and for providing personal record parameters to a first application;

a managing application for generating a session identifier particular to a user initiated session in response to receiving a session initiation request from a first application and for communicating said session identifier to said first application; and

a communication processor for including at least one of, (a) personal record parameters and (b) said session identifier, in different types of command messages communicated to said plurality of Internet compatible applications.

* * * * *