Operand ↓	PM	Preferred embodiment Administrate processing
205	C	Administrating an email causes searching a MS email system with search criteria of the email parameter string. The email parameter string can specify searching any email fields for any values including wildcarding (patterns for matching). Each field is referenced with a predefined name and then associated with a search criteria. For example, the email string of "subj:'personnel';recip:'george@alltell.com';body:'reduction in force'" causes searching all emails with a subject containing "personnel" and was sent to "george@alltell.com" and has a message body containing the string "reduction in force". To search for certain email containers/folders, a sub-search criteria of "folders" is used (e.g. "folders:sent,inbox,company;" indicates to only search the email folders of sent, inbox, and company (no specification preferably indicates to search all folders). Those skilled in the art recognize many useful syntaxes for searching any characteristics of email. Wildcarding (pattern matching) is preferably inherent by searching for substrings. All occurrences found in history are presented with at least their date/time stamps, subject line, sender and recipient, and perhaps other information, of the email and when it took place. In another embodiment, the most recent occurrence from searched folders is presented, and perhaps in an interface which enables appropriate MS email system processing from that point forward (e.g. when processed at local MS). The search takes place as though the user manually launched the search, entered the criteria for the search, and then was presented with the result(s) in an appropriate administration interface. Appropriate MS storage is updated and subsequently processed as though the user had manually performed the search. Preferably, the administration command data is maintained to LBX History, a historical log, or other useful storage for subsequent use. Subsequent administration includes modifying the email(s) as desired, and perhaps having an option to send/resend.

Fig. 73B-2

Operand ↓	PM	Preferred embodiment Administrate processing
207	C	Administrating an sms message causes searching a MS sms messaging system with search criteria of the sms message parameter string. The message parameter string can specify searching any message fields for any values including wildcarding. Each field is referenced with a predefined name and then associated with a search criteria. For example, the sms message string of "recip:'2144034071@nextel.com,9725397137@lbxsrv.com';" causes searching all messages to the sought recipients. To search for certain messaging containers/folders, a sub-search criteria of "folders" is used (e.g. "folders:outgoing" indicates to only search the outgoing folder (no specification preferably indicates to search all folders). Those skilled in the art recognize many useful syntaxes for searching any characteristics of messages. Wildcarding (pattern matching) is preferably inherent by searching for substrings. In the preferred embodiment, all occurrences found in history are presented with at least their date/time stamps, message, sender and recipient, and perhaps other information, of the message and when it took place. In another embodiment, the most recent occurrence from searched folders is presented, and perhaps in an interface which enables appropriate MS messaging system processing from that point forward (e.g. when processed at local MS). The search takes place as though the user manually launched the search, entered the criteria for the search, and then was presented with the result(s) in an appropriate administration interface. Appropriate MS storage is updated and subsequently processed as though the user had manually performed the search. Preferably, the administration command data is maintained to LBX History, a historical log, or other useful storage for subsequent use. Subsequent administration includes modifying the sms message(s) as desired, and perhaps having an option to send/resend.

Fig. 73B-3

Operand ↓	PM	Preferred embodiment Administrate processing
209	C	Administrating a broadcast email causes searching a MS email system with search criteria of the email parameter string. The email parameter string can specify searching any email fields for any values including wildcarding. Each field is referenced with a predefined name and then associated with a search criteria. For example, the email string of "subj:'personnel';recip:'george@alltell.com';body:'reduction in force'" causes searching all emails with a subject containing "personnel" and was sent to "george@alltell.com" and has a message body containing the string "reduction in force". To search for certain email containers/folders, a sub-search criteria of "folders" is used (e.g. "folders:sent,inbox,company;" indicates to only search the email folders of sent, inbox, and company (no specification preferably indicates to search all folders). Those skilled in the art recognize many useful syntaxes for searching any characteristics of email. Wildcarding (pattern matching) is preferably inherent by searching for substrings. In the preferred embodiment, all occurrences found in history are presented with at least their date/time stamps, subject line, sender and recipient, and perhaps other information, of the email and when it took place. In another embodiment, the most recent occurrence from searched folders is presented, and perhaps in an interface which enables appropriate MS email system processing from that point forward (e.g. when processed at local MS). The search takes place as though the user manually launched the search, entered the criteria for the search, and then was presented with the result(s) in an appropriate administration interface. Appropriate MS storage is updated and subsequently processed as though the user had manually performed the search. Preferably, the administration command data is maintained to LBX History, a historical log, or other useful storage for subsequent use. Subsequent administration includes modifying the email(s) as desired, and perhaps having an option to send/resend.

Fig. 73B-4

Operand ↓	PM	Preferred embodiment Administrate processing
211	C	Administrating a broadcast sms message causes searching a MS sms messaging system with search criteria of the sms message parameter string. The message parameter string can specify searching any message fields for any values including wildcarding. Each field is referenced with a predefined name and then associated with a search criteria. For example, the sms message string of "recip:'2144034071@nextel.com,9725397137@lbxsrv.com';" causes searching all messages to the sought recipients. To search for certain messaging containers/folders, a sub-search criteria of "folders" is used (e.g. "folders:outgoing" indicates to only search the outgoing folder (no specification preferably indicates to search all folders). Those skilled in the art recognize many useful syntaxes for searching any characteristics of messages. Wildcarding (pattern matching) is preferably inherent by searching for substrings. In the preferred embodiment, all occurrences found in history are presented with at least their date/time stamps, message, sender and recipient, and perhaps other information, of the message and when it took place. In another embodiment, the most recent occurrence from searched folders is presented, and perhaps in an interface which enables appropriate MS messaging system processing from that point forward (e.g. when processed at local MS). The search takes place as though the user manually launched the search, entered the criteria for the search, and then was presented with the result(s) in an appropriate administration interface. Appropriate MS storage is updated and subsequently processed as though the user had manually performed the search. Preferably, the administration command data is maintained to LBX History, a historical log, or other useful storage for subsequent use. Subsequent administration includes modifying the message(s) as desired, and perhaps having an option to send/resend.
213	O	See Find Command for identical processing this LBX release.

Fig. 73B-5

Operand i	PM	Preferred embodiment Administrate processing
215	C	Administrating an application causes searching the MS for application (and with the params parameter if specified). The app parameter is preferably an executable name. Providing a more defined partial or full path to the application parameter will validate that it is found there. The app parameter string preferably supports wildcarding. In the preferred embodiment, all occurrences found on the MS and their paths are presented to the user with at least their date/time stamps, size, and perhaps attributes information. In another embodiment, all parts which are linked to the executable are identified with their paths, date/time stamps, size, and perhaps attributes when a symbol file is specified with a new parameter. The symbol file is output from a link process and can be used to identify all executable parts such as dynamic link libraries, linked binaries, and any other executable binary file involved with the application. The search takes place as though the user manually launched the search, entered the criteria for the search, and then was presented with the result(s) in an appropriate administration interface (e.g. a properties edit user interface). Appropriate MS storage is updated and subsequently processed as though the user had manually performed the search. Preferably, the administration command data is maintained to LBX History, a historical log, or other useful storage for subsequent use. Subsequent administration includes modifying the configuration, startup parameters, or any other environmental variables of the application.
217	S	See Compose Command for identical processing, except processing may take place locally and/or at privilege-providing remote MS(s) (system(s) parameter).
219	S	See Compose Command for identical processing, except processing may take place locally and/or at privilege-providing remote MS(s) (system(s) parameter).
221	O	See Compose Command for identical processing, except processing may take place locally and/or at privilege-providing remote MS(s) (system(s) parameter).
223	C	See Compose Command for identical processing, except processing may take place locally and/or at privilege-providing remote MS(s) (system(s) parameter).
225	O	See Invoke Command for identical processing this LBX release.
227	O	See Invoke Command for identical processing this LBX release.
229	S	See Invoke Command for identical processing this LBX release.
231	C	See Invoke Command for identical processing this LBX release.
233	S	See Invoke Command for identical processing this LBX release.
235	O	See Invoke Command for identical processing this LBX release.
237	O	See Invoke Command for identical processing this LBX release.
239	O	See Invoke Command for identical processing this LBX release.

Fig. 73B-6

Operand ↓	PM	Preferred embodiment Administrate processing
241	S	See Compose Command for identical processing, except processing may take place locally and/or at privilege-providing remote MS(s) (system(s) parameter).
243	O	See Invoke Command for identical processing this LBX release.
245	S	See Compose Command for identical processing, except processing may take place locally and/or at privilege-providing remote MS(s) (system(s) parameter).
247	O	See Compose Command for identical processing, except processing may take place locally and/or at privilege-providing remote MS(s) (system(s) parameter).
249	O	See Compose Command for identical processing, except processing may take place locally and/or at privilege-providing remote MS(s) (system(s) parameter).
251	C	See Compose Command for identical processing, except processing may take place locally and/or at privilege-providing remote MS(s) (system(s) parameter).
253	C	See Compose Command for identical processing, except processing may take place locally and/or at privilege-providing remote MS(s) (system(s) parameter).
...		

Fig. 73B-7

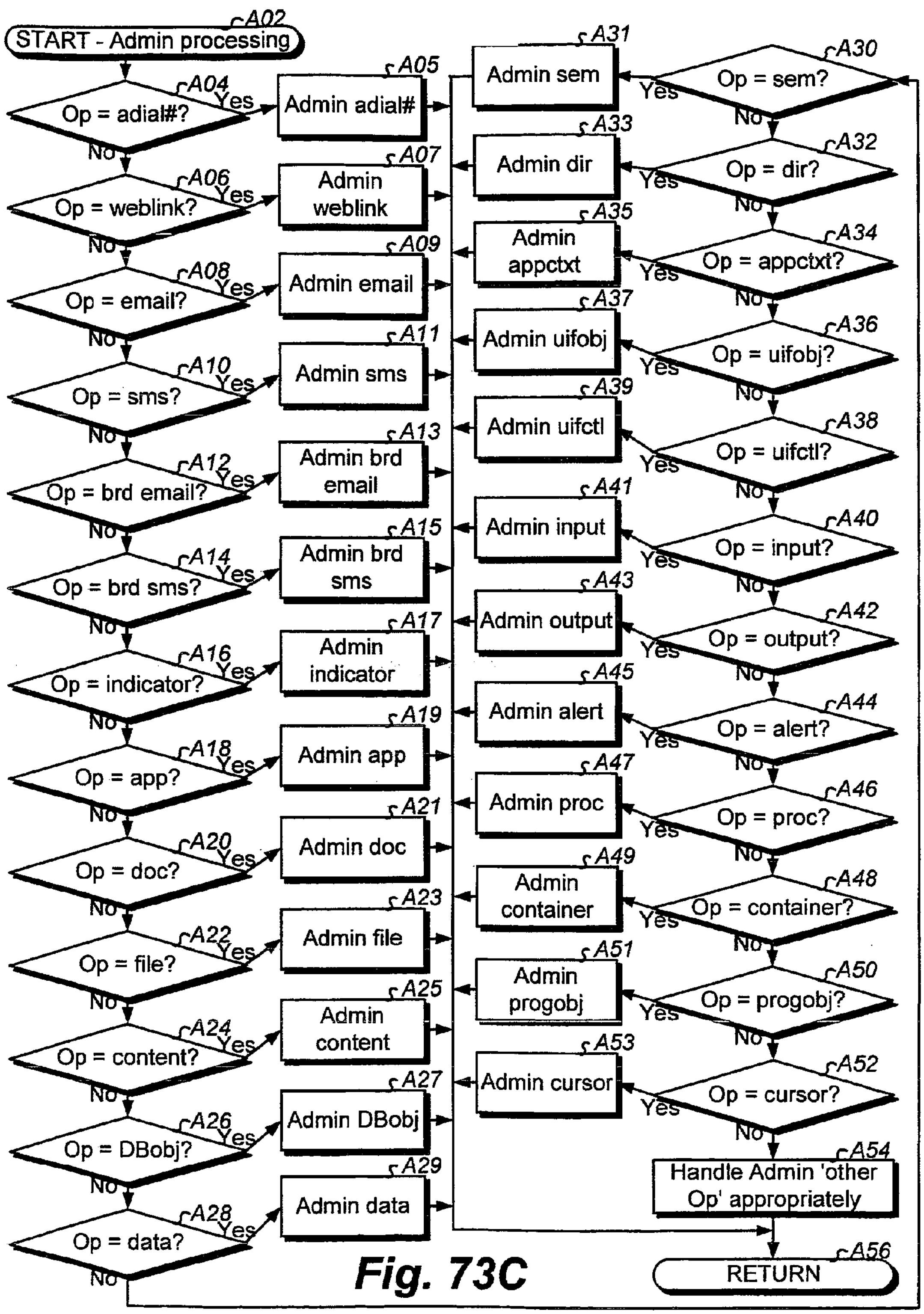
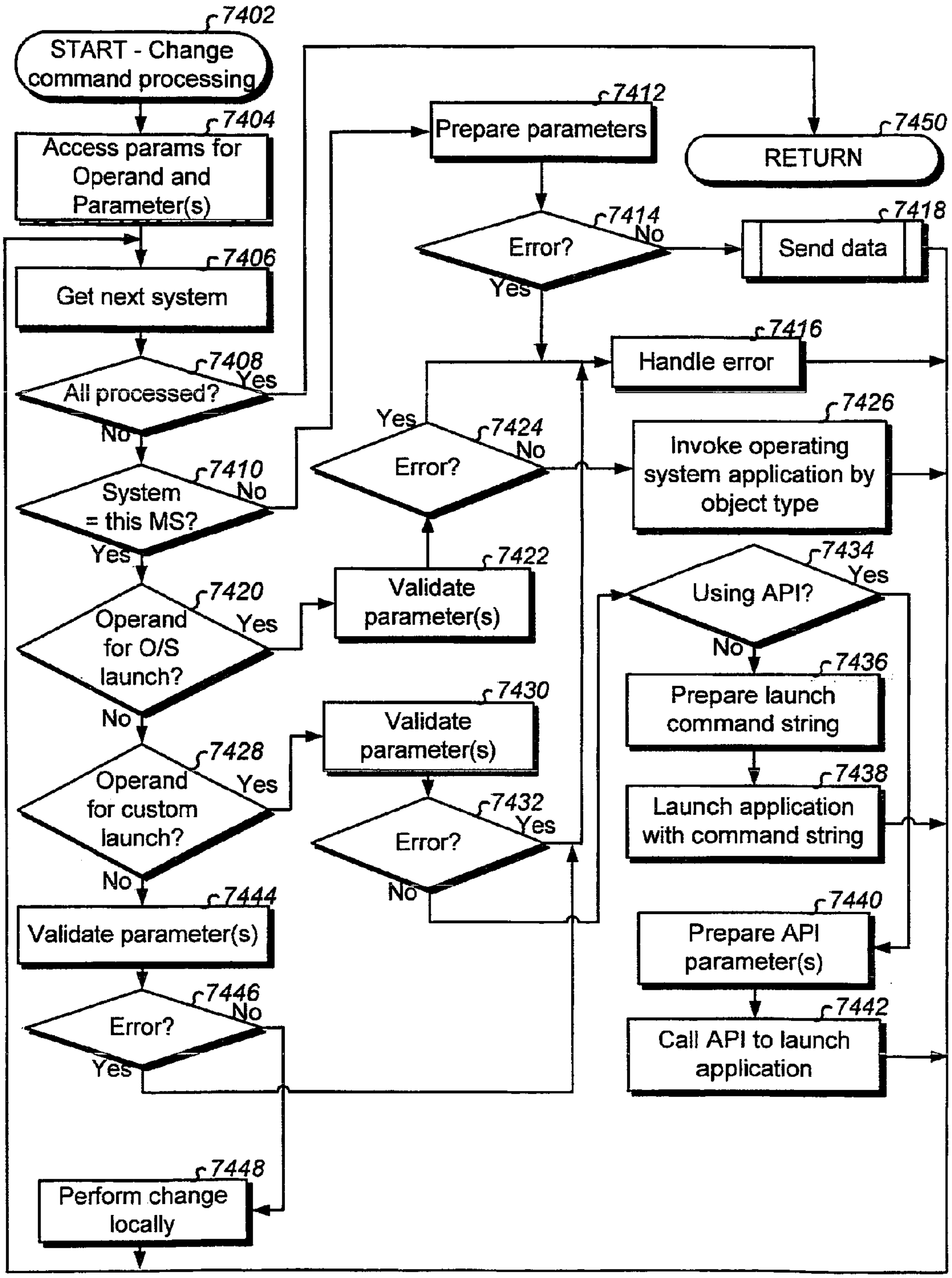

START - Admin processing
A02
Op = adial#?
A04
Admin adial#
A05
Op = weblink?
A06
Admin weblink
A07
Op = email?
A08
Admin email
A09
Op = sms?
A10
Admin sms
A11
Op = brd email?
A12
Admin brd email
A13
Op = brd sms?
A14
Admin brd sms
A15
Op = indicator?
A16
Admin indicator
A17
Op = app?
A18
Admin app
A19
Op = doc?
A20
Admin doc
A21
Op = file?
A22
Admin file
A23
Op = content?
A24
Admin content
A25
Op = DBobj?
A26
Admin DBobj
A27
Op = data?
A28
Admin data
A29
Op = sem?
A30
Admin sem
A31
Op = dir?
A32
Admin dir
A33
Op = appctxt?
A34
Admin appctxt
A35
Op = uifobj?
A36
Admin uifobj
A37
Op = uifctl?
A38
Admin uifctl
A39
Op = input?
A40
Admin input
A41
Op = output?
A42
Admin output
A43
Op = alert?
A44
Admin alert
A45
Op = proc?
A46
Admin proc
A47
Op = container?
A48
Admin container
A49
Op = progobj?
A50
Admin progobj
A51
Op = cursor?
A52
Admin cursor
A53
Handle Admin 'other Op' appropriately
A54
RETURN
A56
Yes
No

Fig. 73C



7402
START - Change command processing
7404
Access params for Operand and Parameter(s)
7406
Get next system
7408
All processed?
Yes
No
7410
System = this MS?
No
Yes
7420
Operand for O/S launch?
Yes
No
7428
Operand for custom launch?
Yes
No
7444
Validate parameter(s)
7446
Error?
No
Yes
7448
Perform change locally
7412
Prepare parameters
7414
Error?
No
Yes
7422
Validate parameter(s)
7424
Error?
Yes
No
7430
Validate parameter(s)
7432
Error?
Yes
No
7450
RETURN
7418
Send data
7416
Handle error
7426
Invoke operating system application by object type
7434
Using API?
Yes
No
7436
Prepare launch command string
7438
Launch application with command string
7440
Prepare API parameter(s)
7442
Call API to launch application

Fig. 74A

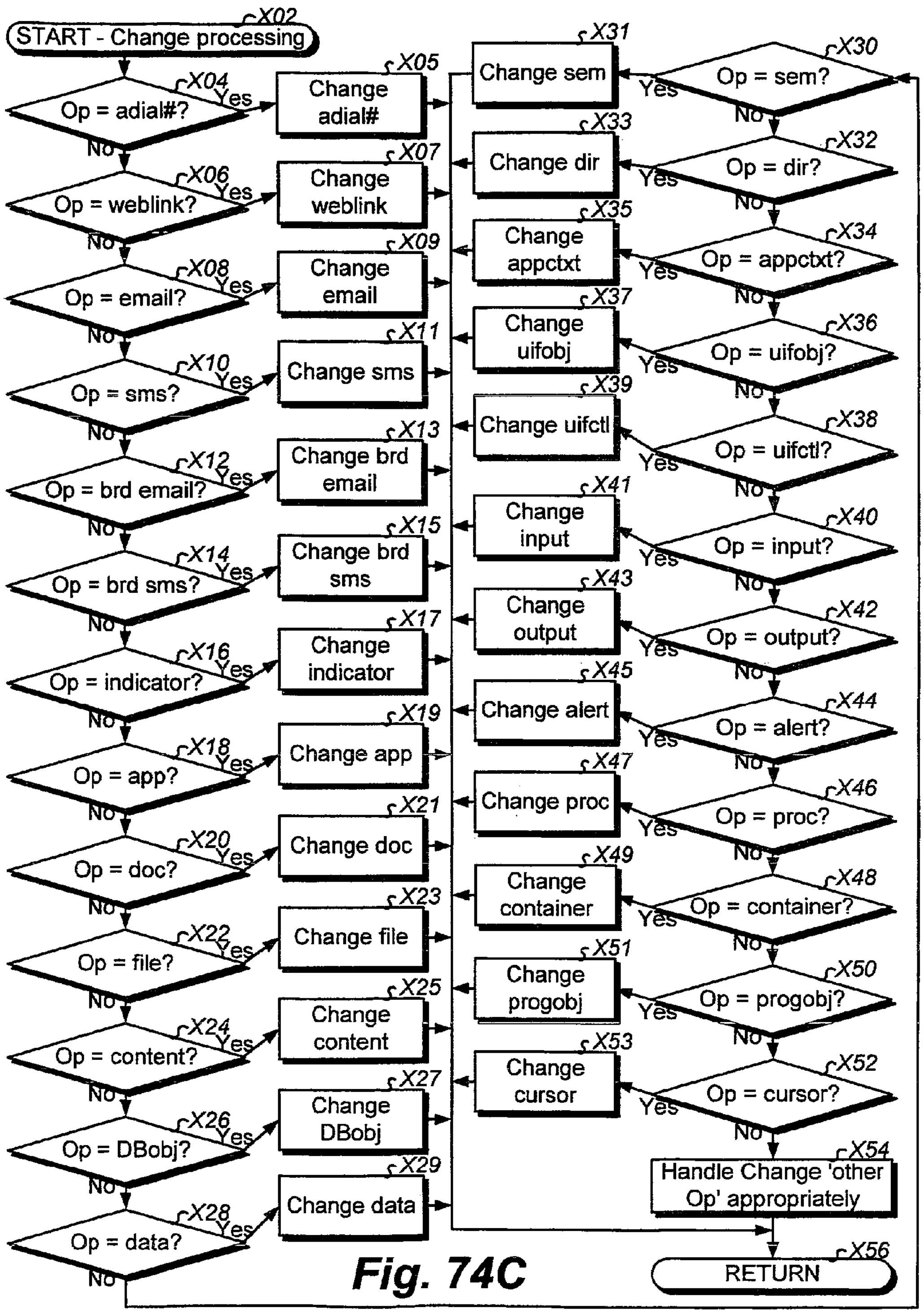

START - Change processing
X02
Op = adial#?
X04
Change adial#
X05
Op = weblink?
X06
Change weblink
X07
Op = email?
X08
Change email
X09
Op = sms?
X10
Change sms
X11
Op = brd email?
X12
Change brd email
X13
Op = brd sms?
X14
Change brd sms
X15
Op = indicator?
X16
Change indicator
X17
Op = app?
X18
Change app
X19
Op = doc?
X20
Change doc
X21
Op = file?
X22
Change file
X23
Op = content?
X24
Change content
X25
Op = DBobj?
X26
Change DBobj
X27
Op = data?
X28
Change data
X29
Op = sem?
X30
Change sem
X31
Op = dir?
X32
Change dir
X33
Op = appctxt?
X34
Change appctxt
X35
Op = uifobj?
X36
Change uifobj
X37
Op = uifctl?
X38
Change uifctl
X39
Op = input?
X40
Change input
X41
Op = output?
X42
Change output
X43
Op = alert?
X44
Change alert
X45
Op = proc?
X46
Change proc
X47
Op = container?
X48
Change container
X49
Op = progobj?
X50
Change progobj
X51
Op = cursor?
X52
Change cursor
X53
Handle Change 'other Op' appropriately
X54
RETURN
X56
Yes
No

Fig. 74C

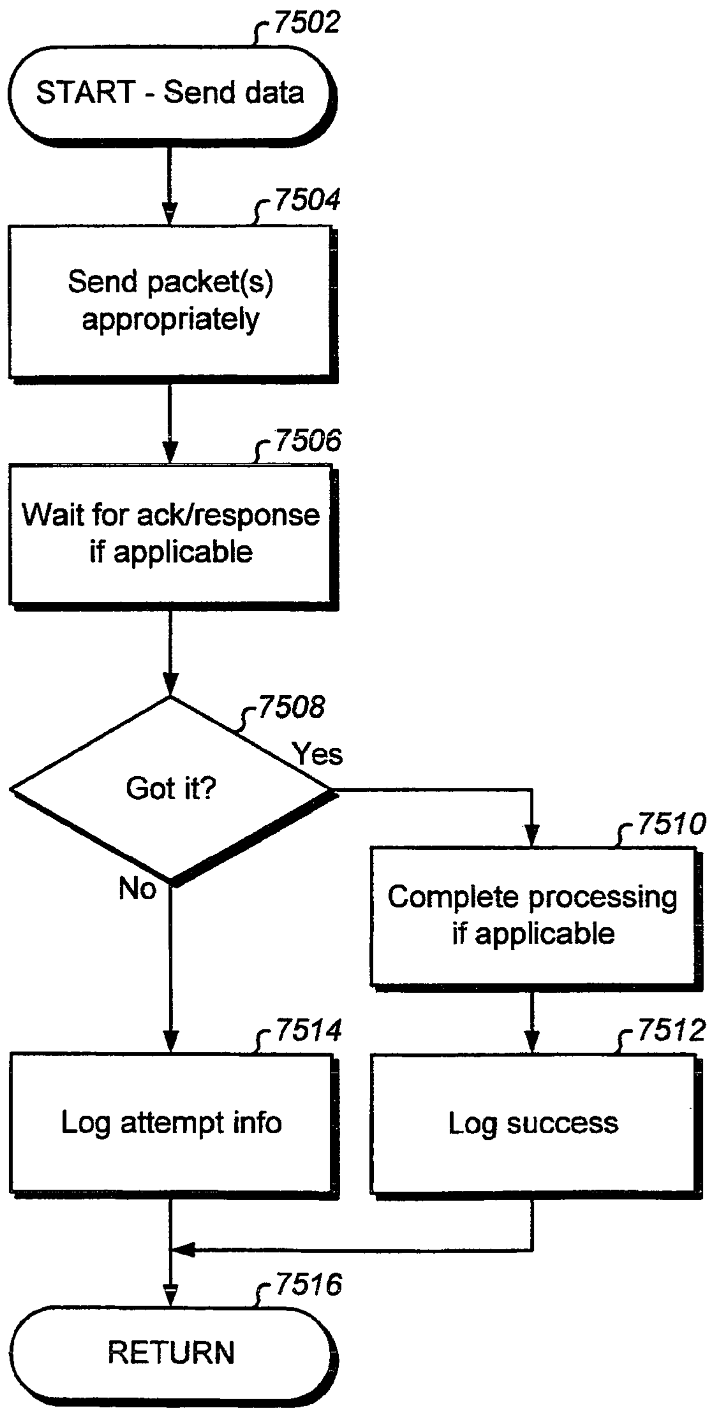
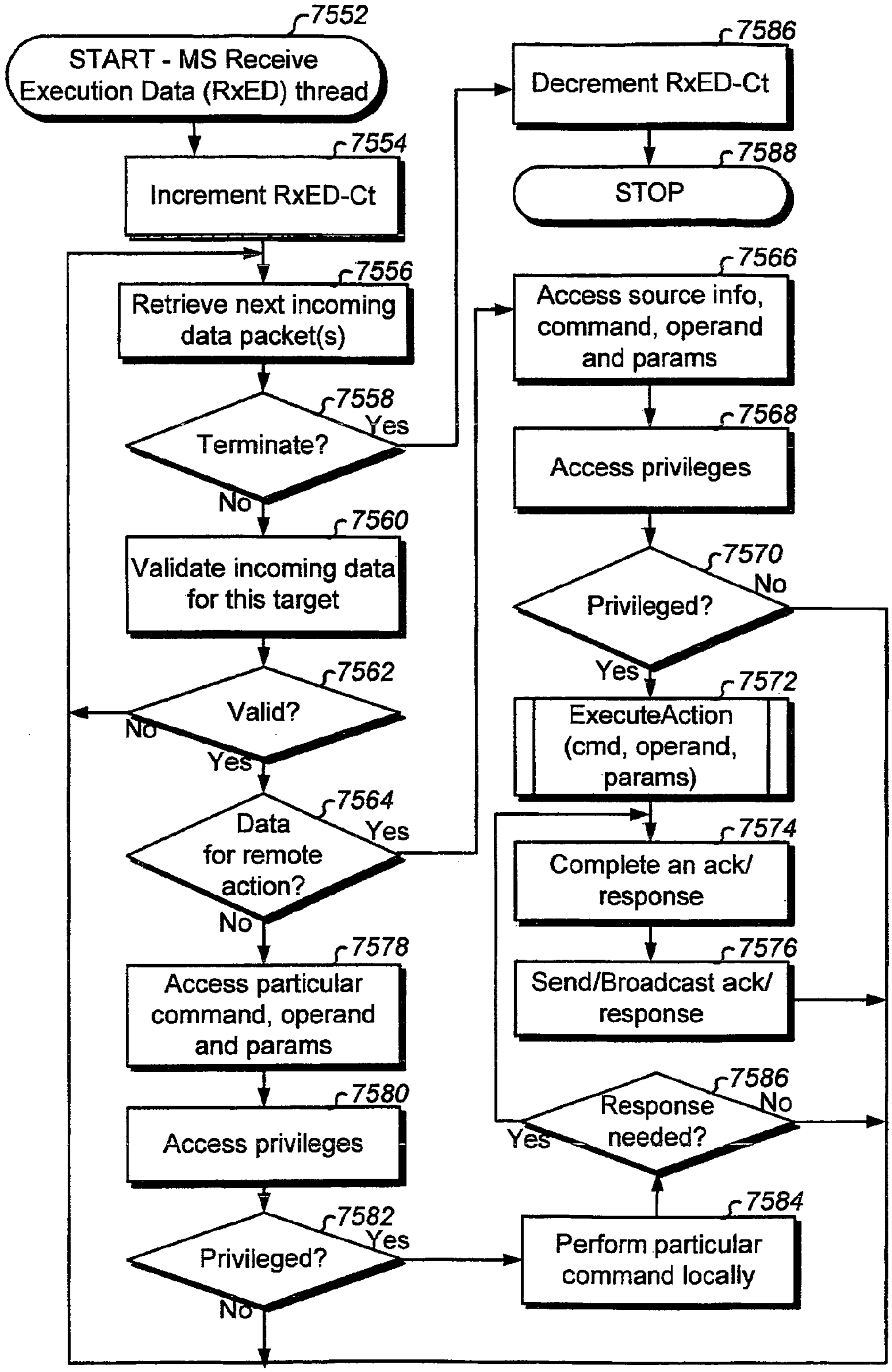

7502
START - Send data
7504
Send packet(s) appropriately
7506
Wait for ack/response if applicable
7508
Got it?
Yes
No
7510
Complete processing if applicable
7514
Log attempt info
7512
Log success
7516
RETURN

Fig. 75A



7552
START - MS Receive Execution Data (RxED) thread
7554
Increment RxED-Ct
7556
Retrieve next incoming data packet(s)
7558
Terminate?
Yes
No
7560
Validate incoming data for this target
7562
Valid?
No
Yes
7564
Data for remote action?
Yes
No
7578
Access particular command, operand and params
7580
Access privileges
7582
Privileged?
Yes
No
7586
Decrement RxED-Ct
7588
STOP
7566
Access source info, command, operand and params
7568
Access privileges
7570
Privileged?
No
Yes
7572
ExecuteAction (cmd, operand, params)
7574
Complete an ack/ response
7576
Send/Broadcast ack/ response
7586
Response needed?
No
Yes
7584
Perform particular command locally

Fig. 75B

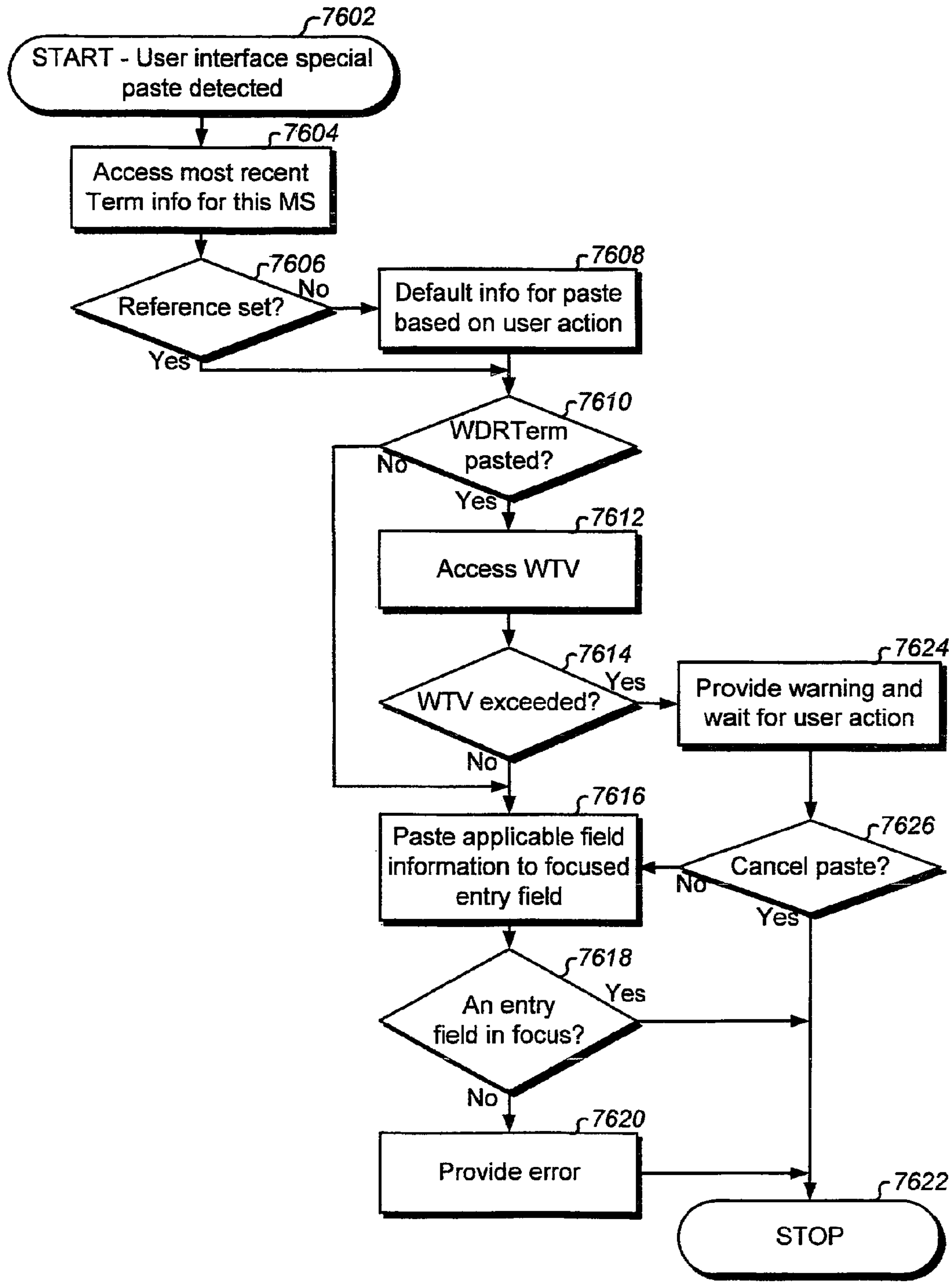

7602
START - User interface special paste detected
7604
Access most recent Term info for this MS
7606
Reference set?
No
Yes
7608
Default info for paste based on user action
7610
WDRTerm pasted?
No
Yes
7612
Access WTV
7614
WTV exceeded?
Yes
No
7624
Provide warning and wait for user action
7616
Paste applicable field information to focused entry field
7626
Cancel paste?
No
Yes
7618
An entry field in focus?
Yes
No
7620
Provide error
7622
STOP

Fig. 76

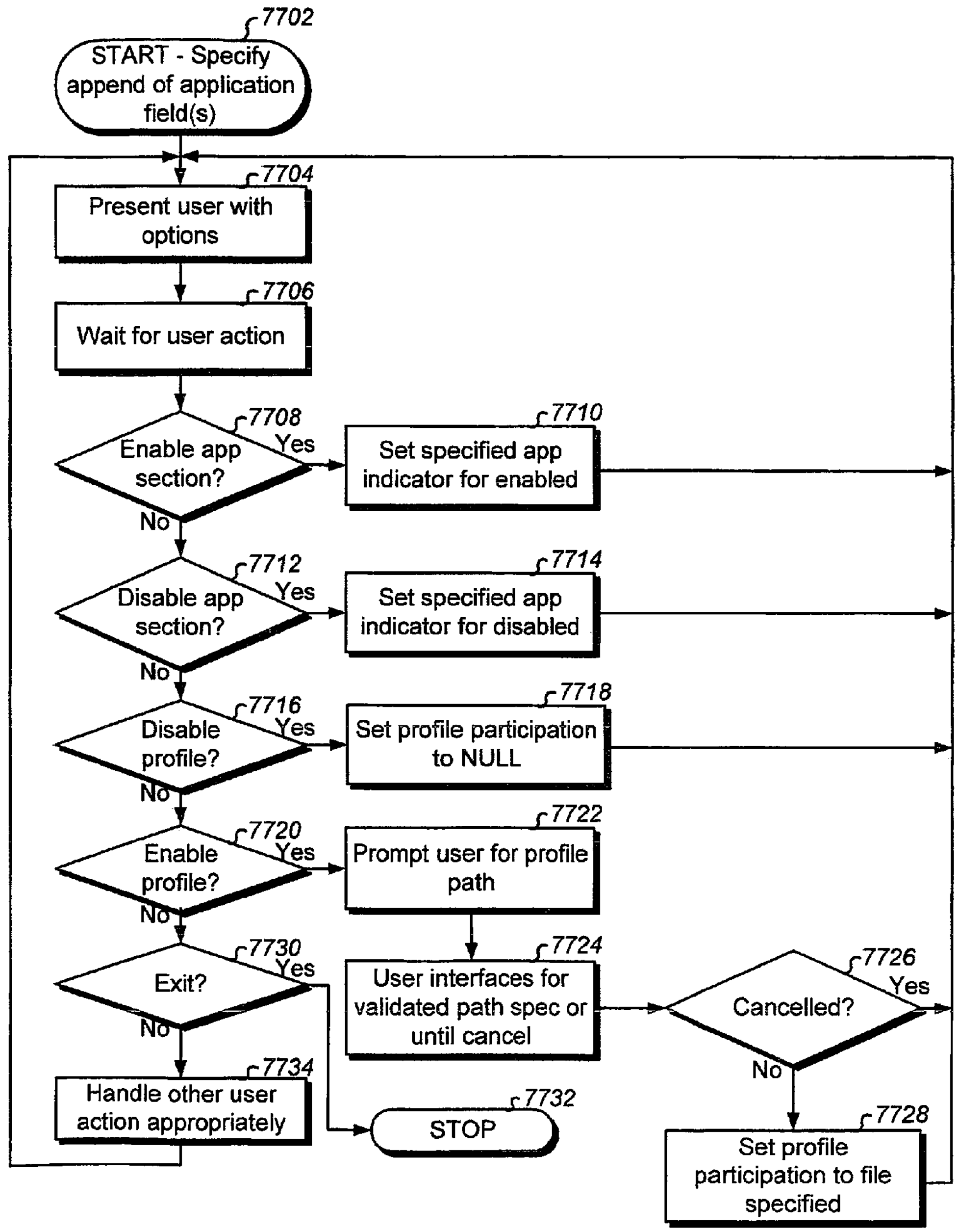

7702
START - Specify append of application field(s)
7704
Present user with options
7706
Wait for user action
7708
Enable app section?
Yes
No
7710
Set specified app indicator for enabled
7712
Disable app section?
Yes
No
7714
Set specified app indicator for disabled
7716
Disable profile?
Yes
No
7718
Set profile participation to NULL
7720
Enable profile?
Yes
No
7722
Prompt user for profile path
7730
Exit?
Yes
No
7724
User interfaces for validated path spec or until cancel
7726
Cancelled?
Yes
No
7734
Handle other user action appropriately
7732
STOP
7728
Set profile participation to file specified

Fig. 77

```
...
<home>
        ...
        <city>Moorestown</city>
        <state>New Jersey</state>
        ...
</home>
...
<interests >
basketball;programming;  running;  football
</interests>
...
<hangouts>
        ...
        <morning></morning>
        <lunch>Jammin's;Mongolian Barbeque</lunch>
        <evening></evening>
        ...
</hangouts>
...
```


Fig. 78

LOCATION BASED EXCHANGE PERMISSIONS

CROSS-REFERENCES TO RELATED APPLICATIONS

This application is a continuation of application Ser. No. 12/287,064 filed Oct. 3, 2008 and entitled "System and Method for Location Based Exchanges of Data Facilitating Distributed Locational Applications" which is a continuation in part of application Ser. No. 12/077,041 filed Mar. 14, 2008 and entitled "System and Method for Location Based Exchanges of Data Facilitating Distributed Locational Applications". This application contains an identical specification to Ser. No. 12/287,064 except for the title, abstract, and claims.

FIELD OF THE INVENTION

The present disclosure relates generally to location based services for mobile data processing systems, and more particularly to location based exchanges of data between distributed mobile data processing systems for locational applications. A common connected service is not required for location based functionality and features. Location based exchanges of data between distributed mobile data processing systems enable location based features and functionality in a peer to peer manner.

BACKGROUND OF THE INVENTION

The internet has exploded with new service offerings. Websites yahoo.com, google.com, ebay.com, amazon.com, and iTunes.com have demonstrated well the ability to provide valuable services to a large dispersed geographic audience through the internet (ebay, yahoo, google, amazon and iTunes (Apple) are trademarks of the respective companies). Thousands of different types of web services are available for many kinds of functionality. Advantages of having a service as the intermediary point between clients, users, and systems, and their associated services, includes centralized processing, centralized maintaining of data, for example to have an all knowing database for scope of services provided, having a supervisory point of control, providing an administrator with access to data maintained by users of the web service, and other advantages associated with centralized control. The advantages are analogous to those provided by the traditional mainframe computer to its clients wherein the mainframe owns all resources, data, processing, and centralized control for all users and systems (clients) that access its services. However, as computers declined in price and adequate processing power was brought to more distributed systems, such as Open Systems (i.e. Windows, UNIX, Linux, and Mac environments), the mainframe was no longer necessary for many of the daily computing tasks. In fact, adequate processing power is incorporated in highly mobile devices, various handheld mobile data processing systems, and other mobile data processing systems. Technology continues to drive improved processing power and data storage capabilities in less physical space of a device. Just as Open Systems took much of the load of computing off of mainframe computers, so to can mobile data processing systems offload tasks usually performed by connected web services. As mobile data processing systems are more capable, there is no need for a service to middleman interactions possible between them.

While a centralized service has its advantages, there are also disadvantages. A service becomes a clearinghouse for all

web service transactions. Regardless of the number of threads of processing spread out over hardware and processor platforms, the web service itself can become a bottleneck causing poor performance for timely response, and can cause a large amount of data that must be kept for all connected users and/or systems. Even large web services mentioned above suffer from performance and maintenance overhead. A web service response will likely never be fast enough. Additionally, archives must be kept to ensure recovery in the event of a disaster because the service houses all data for its operations. Archives also require storage, processing power, planning, and maintenance. A significantly large and costly data center is necessary to accommodate millions of users and/or systems to connect to the service. There is a tremendous amount of overhead in providing such a service. Data center processing power, data capacity, data transmission bandwidth and speed, infrastructure entities, and various performance considerations are quite costly. Costs include real estate required, utility bills for electricity and cooling, system maintenance, personnel to operate a successful business with service(s), etc. A method is needed to prevent large data center costs while eliminating performance issues for features sought. It is inevitable that as users are hungry for more features and functionality on their mobile data processing systems, processing will be moved closer to the device for optimal performance and infrastructure cost savings.

Service delivered location dependent content was disclosed in U.S. Pat. Nos. 6,456,234; 6,731,238; 7,187,997 (Johnson). Anonymous location based services was disclosed in U.S. PTO Publication 2006/0022048 (Johnson). The Johnson patents and published application operate as most web services do in that the clients connecting to the service benefit from the service by having some connectivity to the service. U.S. Publication 2006/0022048 (Johnson) could cause large numbers of users to inundate the service with device heartbeats and data to maintain, depending on the configurations made. While this may be of little concern to a company that has successfully deployed substantially large web service resources, it may be of great concern to other more frugal companies. A method is needed for enabling location dependent features and functionality without the burden of requiring a service.

Users are skeptical about their privacy as internet services proliferate. A service by its very nature typically holds information for a user maintained in a centralized service database. The user's preferences, credential information, permissions, customizations, billing information, surfing habits, and other conceivable user configurations and activity monitoring, can be housed by the service at the service. Company insiders, as well as outside attackers, may get access. Most people are concerned with preventing personal information of any type being kept in a centralized database which may potentially become compromised from a security standpoint. Location based services are of even more concern, in particular when the locations of the user are to be known to a centralized service. A method and system is needed for making users comfortable with knowing that their personal information is at less risk of being compromised.

A reasonable requirement is to push intelligence out to the mobile data processing systems themselves, for example, in knowing their own locations and perhaps the locations of other nearby mobile data processing systems. Mobile data processing systems can intelligently handle many of their own application requirements without depending on some remote service. Just as two people in a business organization should not need a manager to speak to each other, no two mobile data processing systems should require a service middleman for useful location dependent features and functionality. The knowing of its own location should not be the end of social interaction implementation local to the mobile data processing systems, but rather the starting place for a large number of useful distributed local applications that do not require a service.

Different users use different types of Mobile data processing Systems (MSs) which are also called mobile devices: laptops, tablet computers, Personal Computers (PCs), Personal Digital Assistants (PDAs), cell phones, automobile dashboard mounted data processing systems, shopping cart mounted data processing systems, mobile vehicle or apparatus mounted data processing systems, Personal Navigational Devices (PNDs), iPhones (iPhone is a trademark of Apple, Inc.), various handheld mobile data processing systems, etc. MSs move freely in the environment, and are unpredictably moveable (i.e. can be moved anywhere, anytime). Many of these Mobile data processing Systems (MSs) do not have capability of being automatically located, or are not using a service for being automatically located. Conventional methods use directly relative stationary references such as satellites, antennas, etc. to locate MSs. Stationary references are expensive to deploy, and risk obsolescence as new technologies are introduced to the marketplace. Stationary references have finite scope of support for locating MSs.

While the United States E911 mandate for cellular devices documents requirements for automatic location of a Mobile data processing System (MS) such as a cell phone, the mandate does not necessarily promote real time location and tracking of the MSs, nor does it define architecture for exploiting Location Based Services (LBS). We are in an era where Location Based Services (LBS), and location dependent features and functionality, are among the most promising technologies in the world. Automatic locating of every Mobile data processing System (MS) is an evolutionary trend. A method is needed to shorten the length of time for automatically locating every MS. Such a goal can be costly using prior art technologies such as GPS (Global Positioning System), radio wave triangulation, coming within range to a known located sensor, or the like. Complex system infrastructure, or added hardware costs to the MSs themselves, make such ventures costly and time constrained by schedules and costs involved in engineering, construction, and deployment.

A method is needed for enabling users to get location dependent features and functionality through having their mobile locations known, regardless of whether or not their MS is equipped for being located. Also, new and modern location dependent features and functionality can be provided to a MS unencumbered by a connected service.

BRIEF SUMMARY OF THE INVENTION

LBS (Location Based Services) is a term which has gained in popularity over the years as MSs incorporate various location capability. The word "Services" in that terminology plays a major role in location based features and functionality involving interaction between two or more users. This disclosure introduces a new terminology, system, and method referred to as Location Based eXchanges (LBX). LBX is an acronym used interchangeably/contextually throughout this disclosure for the singular term "Location Based Exchange" and for the plural term "Location Based Exchanges", much the same way LBS is used interchangeably/contextually for the single term "Location Based Service" and for the plural term "Location Based Services". LBX describes leveraging the distributed nature of connectivity between MSs in lieu of leveraging a common centralized service nature of connectivity between MSs. The line can become blurred between LBS and LBX since the same or similar features and functionality are provided, and in some cases strengths from both may be used. The underlying architectural shift differentiates LBX from LBS for depending less on centralized services, and more on distributed interactions between MSs. LBX provide server-free and server-less location dependent features and functionality.

Disclosed are many different aspects to LBX, starting with the foundation requirement for each participating MS to know, at some point in time, their own whereabouts. LBX is enabled when an MS knows its own whereabouts. It is therefore a goal to first make as many MSs know their own whereabouts as possible. When two or more MSs know their own whereabouts, LBX enables distributed locational applications whereby a server is not required to middleman social interactions between the MSs. The MSs interact as peers. LBX disclosed include purely peer to peer interactions, peer to peer interactions for routing services, peer to peer interactions for delivering distributed services, and peer to peer interactions for location dependent features and functionality. One embodiment of an LBX enabled MS is referred to as an lbxPhone™.

It is an advantage herein to have no centralized service governing location based features and functionality among MSs. Avoiding a centralized service prevents performance issues, infrastructure costs, and solves many of the issues described above. No centralized service also prevents a user's information from being kept in one accessible place. LBS contain centralized data that is personal in nature to its users. This is a security concern. Having information for all users in one place increases the likelihood that a disaster to the data will affect more than a single user. LBX spreads data out across participating systems so that a disaster affecting one user does not affect any other user.

It is an advantage herein for enabling useful distributed applications without the necessity of having a service, and without the necessity of users and/or systems registering with a service. MSs interact as peers in preferred embodiments, rather than as clients to a common service (e.g. internet connected web service).

It is an advantage herein for locating as many MSs as possible in a wireless network, and without additional deployment costs on the MSs or the network. Conventional locating capability includes GPS (Global Positioning System) using stationary orbiting satellites, improved forms of GPS, for example AGPS (Adjusted GPS) and DGPS (Differential GPS) using stationary located ground stations, wireless communications to stationary located cell tower base stations, TDOA (Time Difference of Arrival) or AOA (Angle of Arrival) triangulation using stationary located antennas, presence detection in vicinity of a stationary located antenna, presence detection at a wired connectivity stationary network location, or other conventional locating systems and methods. Mobile data processing systems, referred to as Indirectly Located Mobile data processing systems (ILMs), are automatically located using automatically detected locations of Directly Located Mobile data processing systems (DLMs) and/or automatically detected locations of other ILMs. ILMs are provided with the ability to participate in the same LBS, or LBX, as a DLM (Directly Located Mobile data processing system). DLMs are located using conventional locating capability mentioned above. DLMs provide reference locations for automatically locating ILMs, regardless of where any one is currently located. DLMs and ILMs can be highly mobile, for example when in use by a user. There are a variety of novel methods for automatically locating ILMs, for example triangulating an ILM (Indirectly Located Mobile data processing system) location using a plurality of DLMs, detecting the ILM being within the vicinity of at least one DLM, triangulating an ILM location using a plurality of other ILMs, detecting the ILM being within the vicinity of at least one other ILM, triangulating an ILM location using a mixed set of DLM(s) and ILM(s), determining the ILM location from heterogeneously located DLMs and/or ILMs, and other novel methods.

MSs are automatically located without using direct conventional means for being automatically located. The conventional locating capability (i.e. conventional locating methods) described above is also referred to as direct methods. Conventional methods are direct methods, but not all direct methods are conventional. There are new direct techniques disclosed below. Provided herein is an architecture, as well as systems and methods, for immediately bringing automatic location detection to every MS in the world, regardless of whether that MS is equipped for being directly located. MSs without capability of being directly located are located by leveraging the automatically detected locations of MSs that are directly located. This is referred to as being indirectly located. An MS which is directly located is hereinafter referred to as a Directly Located Mobile data processing system (DLM). For a plural acronym, MSs which are directly located are hereinafter referred to as Directly Located Mobile data processing systems (DLMs). MSs without capability of being directly located are located using the automatically detected locations of MSs that have already been located. An MS which is indirectly located is hereinafter referred to as an Indirectly Located Mobile data processing system (ILM). For a plural acronym, MSs which are indirectly located are hereinafter referred to as Indirectly Located Mobile data processing systems (ILMs). A DLM can be located in the following ways:

A) New triangulated wave forms;
B) Missing Part Triangulation (MPT) as disclosed below;
C) Heterogeneous direct locating methods;
D) Assisted Direct Location Technology (ADLT) using a combination of direct and indirect methods;
E) Manually specified; and/or
F) Any combinations of A) through E);

DLMs provide reference locations for automatically locating ILMs, regardless of where the DLMs are currently located. It is preferable to assure an accurate location of every DLM, or at least provide a confidence value of the accuracy. A confidence value of the accuracy is used by relative ILMs to determine which are the best set (e.g. which are of highest priority for use to determine ILM whereabouts) of relative DLMs (and/or ILMs) to use for automatically determining the location of the ILM.

In one example, the mobile locations of several MSs are automatically detected using their local GPS chips. Each is referred to as a DLM. The mobile location of a non-locatable MS is triangulated using radio waves between it and three (3) of the GPS equipped DLMs. The MS becomes an ILM upon having its location determined relative the DLMs. ILMs are automatically located using DLMs, or other already located ILMs. An ILM can be located in the following ways:

G) Triangulating an ILM location using a plurality of DLMs with wave forms of any variety (e.g. AOA, TDOA, MPT (a heterogeneous location method));
H) Detecting the ILM being within the reasonably close vicinity of at least one DLM;
I) Triangulating an ILM location using a plurality of other ILMs with wave forms of any variety;
J) Detecting the ILM being within the reasonable close vicinity of at least one other ILM;
K) Triangulating an ILM location using a mixed set of DLM(s) and ILM(s) with wave forms of any variety (referred to as ADLT);
L) Determining the ILM location from heterogeneously located DLMs and/or ILMs (i.e. heterogeneously located, as used here, implies having been located relative different location methodologies);
M) A) through F) Above; and/or
N) Any combinations of A) through M).

Locating functionality may leverage GPS functionality, including but not limited to GPS, AGPS (Adjusted GPS), DGPS, (Differential GPS), or any improved GPS embodiment to achieve higher accuracy using known locations, for example ground based reference locations. The NexTel GPS enabled iSeries cell phones provide excellent examples for use as DLMs (Nextel is a trademark of Sprint/Nextel). Locating functionality may incorporate triangulated locating of the MS, for example using a class of Radio Frequency (RF) wave spectrum (cellular, WiFi (some WiFi embodiments referred to as WiMax), bluetooth, etc), and may use measurements from different wave spectrums for a single location determination (depends on communications interface(s) **70** available). A MS may have its whereabouts determined using a plurality of wave spectrum classes available to it (cellular, WiFi, bluetooth, etc). The term “WiFi” used throughout this disclosure also refers to the industry term “WiMax”. Locating functionality may include in-range proximity detection for detecting the presence of the MS. Wave forms for triangulated locating also include microwaves, infrared wave spectrum relative infrared sensors, visible light wave spectrum relative light visible light wave sensors, ultraviolet wave spectrum relative ultraviolet wave sensors, X-ray wave spectrum relative X-ray wave sensors, gamma ray wave spectrum relative gamma ray wave sensors, and longwave spectrum (below AM) relative longwave sensors. While there are certainly more common methods for automatically locating a MS (e.g. radio wave triangulation, GPS, in range proximity detection), those skilled in the art recognize there are methods for different wave spectrums being detected, measured, and used for carrying information between data processing systems.

Kubler et al (U.S. PTO publications 2004/0264442, 2004/0246940, 2004/0228330, 2004/0151151) disclosed methods for detecting presence of mobile entities as they come within range of a sensor. In Kubler et al, accuracy of the location of the detected MS is not well known, so an estimated area of the whereabouts of the MS is enough to accomplish intended functionality, for example in warehouse installations. A confidence value of this disclosure associated with Kubler et al tends to be low (i.e. not confident), with lower values for long range sensors and higher values for short range sensors.

GPS and the abundance of methods for improving GPS accuracy has led to many successful systems for located MSs with high accuracy. Triangulation provides high accuracies for locating MSs. A confidence value of this disclosure associated with GPS and triangulating location methods tends to be high (i.e. confident). It is preferred that DLMs use the highest possible accuracy method available so that relative ILMs are well located. Not all DLMs need to use the same location methods. An ILM can be located relative DLMs, or other ILMs, that each has different locating methodologies utilized.

Another advantage herein is to generically locate MSs using varieties and combinations of different technologies. MSs can be automatically located using direct conventional methods for accuracy to base on the locating of other MSs.

MSs can be automatically located using indirect methods. Further, it is an advantage to indirectly locate a MS relative heterogeneously located MSs. For example, one DLM may be automatically located using GPS. Another DLM may be automatically located using cell tower triangulation. A third DLM may be automatically located using within range proximity. An ILM can be automatically located at a single location, or different locations over time, relative these three differently located DLMs. The automatically detected location of the ILM may be determined using a form of triangulation relative the three DLMs just discussed, even though each DLM had a different direct location method used. In a preferred embodiment, industry standard IEEE 802.11 WiFi is used to locate (triangulate) an ILM relative a plurality of DLMs (e.g. TDOA in one embodiment). This standard is prolific among more compute trended MSs. Any of the family of 802.11 wave forms such as 802.11a, 802.11b, 802.11g, or any other similar class of wave spectrum can be used, and the same spectrum need not be used between a single ILM and multiple DLMs. 802.x used herein generally refers to the many 802.whatever variations.

Another advantage herein is to make use of existing marketplace communications hardware, communications software interfaces, and communications methods and location methods where possible to accomplish locating an MS relative one or more other MSs. While 802.x is widespread for WiFi communications, other RF wave forms can be used (e.g. cell phone to cell tower communications). In fact, any wave spectrum for carrying data applies herein.

Still another advantage is for support of heterogeneous locatable devices. Different people like different types of devices as described above. Complete automation of locating functionality can be provided to a device through local automatic location detection means, or by automatic location detection means remote to the device. Also, an ILM can be located relative a laptop, a cell phone, and a PDA (i.e. different device types).

Yet another advantage is to prevent the unnecessary storing of large amounts of positioning data for a network of MSs. Keeping positioning data for knowing the whereabouts of all devices can be expensive in terms of storage, infrastructure, performance, backup, and disaster recovery. A preferred embodiment simply uses a distributed approach to determining locations of MSs without the overhead of an all-knowing database maintained somewhere. Positions of MSs can be determined "on the fly" without storing information in a master database. However, there are embodiments for storing a master database, or a subset thereof, to configurable storage destinations, when it makes sense. A subset can be stored at a MS.

Another advantage includes making use of existing location equipped MSs to expand the network of locatable devices by locating non-equipped MSs relative the location of equipped MSs. MSs themselves help increase dimensions of the locatable network of MSs. The locatable network of MSs is referred to as an LN-Expanse (i.e. Location-Network Expanse). An LN-Expanse dynamically grows and shrinks based on where MSs are located at a particular time. For example, as users travel with their personal MSs, the personal MSs themselves define the LN-Expanse since the personal MSs are used to locate other MSs. An ILM simply needs location awareness relative located MSs (DLMs and/or ILMs).

Yet another advantage is a MS interchangeably taking on the role of a DLM or ILM as it travels. MSs are chameleons in this regard, in response to location technologies that happen to be available. A MS may be equipped for DLM capability, but may be in a location at some time where the capability is inoperable. In these situations the DLM takes on the role of an ILM. When the MS again enters a location where it can be a DLM, it automatically takes on the role of the DLM. This is very important, in particular for emergency situations. A hiker has a serious accident in the mountains which prevents GPS equipped DLM capability from working. Fortunately, the MS automatically takes on the role of an ILM and is located within the vicinity of neighboring (nearby) MSs. This allows the hiker to communicate his location, operate useful locational application functions and features at his MS, and enable emergency help that can find him.

It is a further advantage that MS locations be triangulated using any wave forms (e.g. RF, microwaves, infrared, visible light, ultraviolet, X-ray, gamma ray). X-ray and gamma ray applications are special in that such waves are harmful to humans in short periods of times, and such applications should be well warranted to use such wave forms. In some medical embodiments, micro-machines may be deployed within a human body. Such micro-machines can be equipped as MSs. Wave spectrums available at the time of deployment can be used by the MSs for determining exact positions when traveling through a body.

It is another advantage to use TDOA (Time Difference Of Arrival), AOA (Angle Of Arrival), and Missing Part Triangulation (MPT) when locating a MS. TDOA uses time information to determine locations, for example for distances of sides of a triangle. AOA uses angles of arrival to antennas to geometrically assess where a MS is located by intersecting lines drawn from the antennas with detected angles. MPT is disclosed herein as using combinations of AOA and TDOA to determine a location. Exclusively using all AOA or exclusively using all TDOA is not necessary. MPT can be a direct method for locating MSs.

Yet another advantage is to locate MSs using Assisted Direct Location Technology (ADLT). ADLT is disclosed herein as using direct (conventional) location capability together with indirect location capability to confidently determine the location of a MS.

Still another advantage is to permit manual specification for identifying the location of a MS (a DLM). The manual location can then in turn be used to facilitate locating other MSs. A user interface may be used for specification of a DLM location. The user interface can be local, or remote, to the DLM. Various manual specification methods are disclosed. Manual specification is preferably used with less mobile MSs, or existing MSs such as those that use dodgeball.com (trademark of Google). The confidence value depends on how the location is specified, whether or not it was validated, and how it changes when the MS moves after being manually set. Manual specification should have limited scope in an LN-expanse unless inaccuracies can be avoided.

Another advantage herein is locating a MS using any of the methodologies above, any combinations of the methodologies above, and any combinations of direct and/or indirect location methods described.

Another advantage is providing synergy between different locating technologies for smooth operations as an MS travels. There are large numbers of methods and combinations of those methods for keeping an MS informed of its whereabouts. Keeping an MS informed of its whereabouts in a timely manner is critical in ensuring LBX operate optimally, and for ensuring nearby MSs without certain locating technologies can in turn be located.

It is another advantage for locating an MS with multiple location technologies during its travels, and in using the best of breed data from multiple location technologies to infer a

MS location confidently. Confidence values are associated with reference location information to ensure an MS using the location information can assess accuracy. A DLM is usually an "affirmifier". An affirmifier is an MS with its whereabouts information having high confidence of accuracy and can serve as a reference for other MSs. An ILM can also be an affirmifier provided there is high confidence that the ILM location is known. An MS (e.g. ILM) may be a "pacifier". A pacifier is an MS having location information for its whereabouts with a low confidence for accuracy. While it can serve as a reference to other ILMs, it can only do so by contributing a low confidence of accuracy.

It is an advantage to synergistically make use of the large number of locating technologies available to prevent one particular type of technology to dominate others while using the best features of each to assess accurate mobile locations of MSs.

A further advantage is to leverage a data processing system with capability of being located for co-locating another data processing system without any capability of being located. For example, a driver owns an older model automobile, has a useful second data processing system in the automobile without means for being automatically located. The driver also own a cell phone, called a first data processing system, which does have means for being automatically located. The location of the first data processing system can be shared with the second data processing system for locating the second data processing system. Further still, the second data processing system without means for being automatically located is located relative a first set (plurality) of data processing systems which are not at the same location as the second data processing system. So, data processing systems are automatically located relative at least one other data processing which can be automatically located.

Another advantage is a LBX enabled MS includes a service informant component for keeping a supervisory service informed. This prevents an MS from operating in total isolation, and prevents an MS from operating in isolation with those MSs that are within its vicinity (e.g. within maximum range **1306**) at some point in time, but to also participate when the same MSs are great distances from each other. There are LBX which would fit well into an LBS model, but a preferred embodiment chooses to use the LBX model. For example, multiple MS users are seeking to carpool to and from a common destination. The service informant component can perform timely updates to a supervisory service for route comparisons between MSs, even though periods of information are maintained only at the MSs. For example, users find out that they go to the same church with similar schedules, or coworkers find out they live nearby and have identical work schedules. The service informant component can keep a service informed of MS whereabouts to facilitate novel LBX applications.

It is a further advantage in leveraging the vast amount of MS WiFi/WiMax deployment underway in the United States. More widespread WiFi/WiMax availability enhances the ability for well performing peer to peer types of features and functionality disclosed.

It is a further advantage to prevent unnecessary established connections from interfering with successfully triangulating a MS position. As the MS roams and encounters various wave spectrum signals, that is all that is required for determining the MS location. Broadcast signaling contains the necessary location information for automatically locating the MS.

Yet another advantage is to leverage Network Time Protocol (NTP) for eliminating bidirectional communications in determining Time of Arrival (TOA) and TDOA (Time Difference Of Arrival) measurements (TDOA as used in the disclosure generally refers to both TOA and TDOA). NTP enables a single unidirectional transmission of data to carry all that is necessary in determining TDOA, provided the sending data processing system and the receiving data processing system are NTP synchronized to an adequate granulation of time.

It is an advantage of this disclosure to provide a competing superior alternative to server based mobile technologies such as that of U.S. Pat. Nos. 6,456,234; 6,731,238; 7,187,997; and U.S. PTO Publication 2006/0022048 (Johnson). It is also an advantage to leverage both LBX technology and LBS technology in the same MS in order to improve the user experience. The different technologies can be used to complement each other in certain embodiments.

A further advantage herein is to leverage existing "usual communications" data transmissions for carrying new data that is ignored by existing MS processing, but observed by new MS processing, for carrying out processing maximizing location functions and features across a large geography. Alternatively, new data can be transmitted between systems for the same functionality.

It is an advantage herein in providing peer to peer service propagation. ILMs are provided with the ability to participate in the same Location Based Services (LBS) or other services as DLM(s) in the vicinity. An MS may have access to services which are unavailable to other MSs. Any MS can share its accessible services for being accessible to any other MS, preferably in accordance with permissions. For example, an MS without internet access can get internet access via an MS in the vicinity with internet access. In a preferred embodiment, permissions are maintained in a peer to peer manner prior to lookup for proper service sharing. In another embodiment, permissions are specified and used at the time of granting access to the shared services. Once granted for sharing, services can be used in a mode as if the sharing user is using the services, or in a mode as if the user accepting the share is a new user to the service. Routing paths are dynamically reconfigured and transparently used as MSs travel. Hop counts dynamically change to strive for a minimal number of hops for an MS getting access to a desirable service. Route communications depend on where the MS needing the service is located relative a minimal number of hops through other MSs to get to the service. Services can be propagated from DLMs to DLMS, DLMs to ILMs, or ILMs to ILMs.

It is another advantage herein for providing peer to peer permissions, authentication, and access control. A service is not necessary for maintaining credentials and permissions between MSs. Permissions are maintained locally to a MS. In a centralized services model, a database can become massive in size when searching for needed permissions. Permission searching and validation of U.S. PTO Publication 2006/0022048 (Johnson) was costly in terms of database size and performance. There was overhead in maintaining who owned the permission configuration for every permission granted. Maintaining permissions locally, as described below, reduces the amount of data to represent the permission because the owner is understood to be the personal user of the MS. Additionally, permission searching is very fast because the MS only has to search its local data for permissions that apply to only its MS.

Yet another advantage is to provide a nearby, or nearness, status using a peer to peer system and method, rather than intelligence maintained in a centralized database for all participating MSs. There is lots of overhead in maintaining a large database containing locations of all known MSs. This disclosure removes such overhead through using nearby detection means of one MS when in the vicinity of another

MS. There are varieties of controls for governing how to generate the nearby status. In one aspect, a MS automatically calls the nearby MS thereby automatically connecting the parties to a conversation without user interaction to initiate the call. In another aspect, locally maintained configurations govern functionality when MSs are newly nearby, or are newly departing being nearby. Nearby status, alerts, and queries are achieved in a LBX manner.

It is yet another advantage for automatic call forwarding, call handling, and call processing based on the whereabouts of a MS, or whereabouts of a MS relative other MSs. The nearness condition of one MS to another MS can also affect the automatic call forwarding functionality.

Yet another advantage herein is for peer to peer content delivery and local MS configuration of that content. Users need no connectivity to a service. Users make local configurations to enjoy location based content delivery to other MSs. Content is delivered under a variety of circumstances for a variety of configurable reasons. Content maintained local to an MS is delivered asynchronously to other MSs for nearby alerts, arrival or departure to and from geofenced areas, and other predicated conditions of nearby MSs. While it may appear there are LBS made available to users of MSs, there are in fact LBX being made available to those users.

Another advantage herein is a LBX enabled MS can operate in a peer to peer manner to data processing systems which control environmental conditions. For example, automobile equipped (or driver kept) MSs encounter an intersection having a traffic light. Interactions between the MSs at the intersection and a data processing system in the vicinity for controlling the traffic light can automatically override light color changing for optimal traffic flow. In another embodiment, a parking lot search by a user with an MS is facilitated as he enters the parking lot, and in accordance with parking spaces currently occupied. In general, other nearby data processing systems can have their control logic processed for a user's preferences (as defined in the MS), a group of nearby user's preferences, and/or situational locations (see U.S. Pat. Nos. 6,456,234; 6,731,238; 7,187,997 (Johnson) for "situational location" terminology) of nearby MSs.

Another advantage herein is an MS maintains history of hotspot locations detected for providing graphical indication of hotspot whereabouts. This information can be used by the MS user in guiding where a user should travel in the future for access to services at the hotspot. Hotspot growth prevents a database in being timely configured with new locations. The MS can learn where hotspots are located, as relevant to the particular MS. The hotspot information is instantly available to the MS.

A further advantage is for peer to peer proximity detection for identifying a peer service target within the MS vicinity. A peer service target can be acted upon by an MS within range, using an application at the MS. The complementary whereabouts of the peer service target and MS automatically notify the user of service availability. The user can then use the MS application for making a payment, or for performing an account transfer, account deposit, account deduction, or any other transaction associated with the peer service target.

Yet another advantage is for a MS to provide new self management capability such as automatically marking photographs taken with location information, a date/time stamp, and who was with the person taking the picture.

Yet another advantage is being alerted to nearby people needing assistance and nearby fire engines or police cars that need access to roads.

A further advantage is providing a MS platform for which new LBX features and functionality can be brought quickly to the marketplace. The platform caters to a full spectrum of users including highly technical software developers, novice users, and users between those ranges. A rich programming environment is provided wherein whereabouts (WDR) information interchanged with other MSs in the vicinity causes triggering of privileged actions configured by users. The programming environment can be embedded in, or "plugged into", an existing software development environment, or provided on its own. A syntax may be specified with source code statements, XML, SQL database definitions, a datastream, or any other derivative of a well defined BNF grammar. A user friendly configuration environment is provided wherein whereabouts information interchanged with other MSs in the vicinity causes triggering of privileged actions configured by users. The platform is an event based environment wherein WDRs containing certain configured sought information are recognized at strategic processing paths for causing novel processing of actions. Events can be defined with complex expressions, and actions can be defined using homegrown executables, APIs, scripts, applications, a set of commands provided with the LBX platform, or any other executable processing. The LBX platform includes a variety of embodiments for charter and permission definitions including an internalized programmatic form, a SQL database form, a data record form, a datastream form, and a well defined BNF grammar for deriving other useful implementations (e.g. lex and yacc).

It is another advantage to support a countless number of privileges that can be configured, managed, and processed in a peer to peer manner between MSs. Any peer to peer feature or set of functionality can have a privilege associated to it for being granted from one user to another. It is also an advantage for providing a variety of embodiments for how to manage and maintain privileges in a network of MSs.

It is another advantage to support a complete set of options for charters that can be configured, managed, and processed in a peer to peer manner between MSs. Charters can become effective under a comprehensive set of conditions, expressions, terms, and operators. It is also an advantage for providing a variety of embodiments for how to manage and maintain charters in a network of MSs.

It is a further advantage for providing multithreaded communications of permission and charter information and transactions between MSs for well performing peer to peer interactions. Any signal spectrum for carrying out transmission and reception is candidate, depending on the variety of MS. In fact, different signaling wave spectrums, types, and protocols may be used in interoperating communications, or even for a single transaction, between MSs.

It is yet another advantage for increasing the range of the LN-expanse from a wireless vicinity to potentially infinite vicinity through other data processing (e.g. routing) equipment. While wireless proximity is used for governing automatic location determination, whereabouts information may be communicated between MSs great distances from each other provided there are privileges and/or charters in place making such whereabouts information relevant for the MS. Whereabouts information of others will not be maintained unless there are privileges in place to maintain it. Whereabouts information may not be shared with others if there have been no privileges granted to a potential receiving MS. Privileges can provide relevance to what whereabouts (WDR) information is of use, or should be processed, maintained, or acted upon.

Further features and advantages of the disclosure, as well as the structure and operation of various embodiments of the disclosure, are described in detail below with reference to the accompanying drawings. In the drawings, like reference numbers generally indicate identical, functionally similar, and/or structurally similar elements. The drawing in which an element first appears is indicated by the leftmost digit(s) in the corresponding reference number, except that reference numbers **1** through **99** may be found on the first 4 drawings of FIGS. **1**A through **1**D. None of the drawings, discussions, or materials herein is to be interpreted as limiting to a particular embodiment. The broadest interpretation is intended. Other embodiments accomplishing same functionality are within the spirit and scope of this disclosure. It should be understood that information is presented by example and many embodiments exist without departing from the spirit and scope of this disclosure.

BRIEF DESCRIPTION OF THE DRAWINGS

There is no guarantee that there are descriptions in this specification for explaining every novel feature found in the drawings. The present disclosure will be described with reference to the accompanying drawings, wherein:

FIG. **1**A depicts a preferred embodiment high level example componentization of a MS in accordance with the present disclosure;

FIG. **1**B depicts a Location Based eXchanges (LBX) architectural illustration for discussing the present disclosure;

FIG. **1**C depicts a Location Based Services (LBS) architectural illustration for discussing prior art of the present disclosure;

FIG. **1**D depicts a block diagram of a data processing system useful for implementing a MS, ILM, DLM, centralized server, or any other data processing system disclosed herein;

FIG. **1**E depicts a network illustration for discussing various deployments of whereabouts processing aspects of the present disclosure;

FIG. **2**A depicts an illustration for describing automatic location of a MS through the MS coming into range of a stationary cellular tower;

FIG. **2**B depicts an illustration for describing automatic location of a MS through the MS coming into range of some stationary antenna;

FIG. **2**C depicts an illustration for discussing an example of automatically locating a MS through the MS coming into range of some stationary antenna;

FIG. **2**D depicts a flowchart for describing a preferred embodiment of a service whereabouts update event of an antenna in-range detected MS when MS location awareness is monitored by a stationary antenna or cell tower;

FIG. **2**E depicts a flowchart for describing a preferred embodiment of an MS whereabouts update event of an antenna in-range detected MS when MS location awareness is monitored by the MS;

FIG. **2**F depicts a flowchart for describing a preferred embodiment of a procedure for inserting a Whereabouts Data Record (WDR) to an MS whereabouts data queue;

FIG. **3**A depicts a locating by triangulation illustration for discussing automatic location of a MS;

FIG. **3**B depicts a flowchart for describing a preferred embodiment of the whereabouts update event of a triangulated MS when MS location awareness is monitored by some remote service;

FIG. **3**C depicts a flowchart for describing a preferred embodiment of the whereabouts update event of a triangulated MS when MS location awareness is monitored by the MS;

FIG. **4**A depicts a locating by GPS triangulation illustration for discussing automatic location of a MS;

FIG. **4**B depicts a flowchart for describing a preferred embodiment of the whereabouts update event of a GPS triangulated MS;

FIG. **5**A depicts a locating by stationary antenna triangulation illustration for discussing automatic location of a MS;

FIG. **5**B depicts a flowchart for describing a preferred embodiment of the whereabouts update event of a stationary antenna triangulated MS;

FIG. **6**A depicts a flowchart for describing a preferred embodiment of a service whereabouts update event of a physically or logically connected MS;

FIG. **6**B depicts a flowchart for describing a preferred embodiment of a MS whereabouts update event of a physically or logically connected MS;

FIGS. **7**A, **7**B and **7**C depict a locating by image sensory illustration for discussing automatic location of a MS;

FIG. **7**D depicts a flowchart for describing a preferred embodiment of graphically locating a MS, for example as illustrated by FIGS. **7**A through **7**C;

FIG. **8**A heterogeneously depicts a locating by arbitrary wave spectrum illustration for discussing automatic location of a MS;

FIG. **8**B depicts a flowchart for describing a preferred embodiment of locating a MS through physically contacting the MS;

FIG. **8**C depicts a flowchart for describing a preferred embodiment of locating a MS through a manually entered whereabouts of the MS;

FIG. **9**A depicts a table for illustrating heterogeneously locating a MS;

FIG. **9**B depicts a flowchart for describing a preferred embodiment of heterogeneously locating a MS;

FIGS. **10**A and **10**B depict an illustration of a Locatable Network expanse (LN-Expanse) for describing locating of an ILM with all DLMs;

FIG. **10**C depicts an illustration of a Locatable Network expanse (LN-Expanse) for describing locating of an ILM with an ILM and DLM;

FIGS. **10**D, **10**E, and **10**F depict an illustration of a Locatable Network expanse (LN-Expanse) for describing locating of an ILM with all ILMs;

FIGS. **10**G and **10**H depict an illustration for describing the infinite reach of a Locatable Network expanse (LN-Expanse) according to MSs;

FIG. **10**I depicts an illustration of a Locatable Network expanse (LN-Expanse) for describing a supervisory service;

FIG. **11**A depicts a preferred embodiment of a Whereabouts Data Record (WDR) **1100** for discussing operations of the present disclosure;

FIGS. **11**B, **11**C and **11**D depict an illustration for describing various embodiments for determining the whereabouts of an MS;

FIG. **11**E depicts an illustration for describing various embodiments for automatically determining the whereabouts of an MS;

FIG. **12** depicts a flowchart for describing an embodiment of MS initialization processing;

FIGS. **13**A through **13**C depict an illustration of data processing system wireless data transmissions over some wave spectrum;

FIG. **14**A depicts a flowchart for describing a preferred embodiment of MS LBX configuration processing;

FIG. **14**B depicts a continued portion flowchart of FIG. **14**A for describing a preferred embodiment of MS LBX configuration processing;

FIG. **15**A depicts a flowchart for describing a preferred embodiment of DLM role configuration processing;

FIG. **15**B depicts a flowchart for describing a preferred embodiment of ILM role configuration processing;

FIG. **15**C depicts a flowchart for describing a preferred embodiment of a procedure for Manage List processing;

FIG. **16** depicts a flowchart for describing a preferred embodiment of NTP use configuration processing;

FIG. **17** depicts a flowchart for describing a preferred embodiment of WDR maintenance processing;

FIG. **18** depicts a flowchart for describing a preferred embodiment of a procedure for variable configuration processing;

FIG. **19** depicts an illustration for describing a preferred embodiment multithreaded architecture of peer interaction processing of a MS in accordance with the present disclosure;

FIG. **20** depicts a flowchart for describing a preferred embodiment of MS whereabouts broadcast processing;

FIG. **21** depicts a flowchart for describing a preferred embodiment of MS whereabouts collection processing;

FIG. **22** depicts a flowchart for describing a preferred embodiment of MS whereabouts supervisor processing;

FIG. **23** depicts a flowchart for describing a preferred embodiment of MS timing determination processing;

FIG. **24**A depicts an illustration for describing a preferred embodiment of a thread request queue record;

FIG. **24**B depicts an illustration for describing a preferred embodiment of a correlation response queue record;

FIG. **24**C depicts an illustration for describing a preferred embodiment of a WDR request record;

FIG. **25** depicts a flowchart for describing a preferred embodiment of MS WDR request processing;

FIG. **26**A depicts a flowchart for describing a preferred embodiment of MS whereabouts determination processing;

FIG. **26**B depicts a flowchart for describing a preferred embodiment of processing for determining a highest possible confidence whereabouts;

FIG. **27** depicts a flowchart for describing a preferred embodiment of queue prune processing;

FIG. **28** depicts a flowchart for describing a preferred embodiment of MS termination processing;

FIG. **29**A depicts a flowchart for describing a preferred embodiment of a process for starting a specified number of threads in a specified thread pool;

FIG. **29**B depicts a flowchart for describing a preferred embodiment of a procedure for terminating the process started by FIG. **29**A;

FIGS. **30**A through **30**B depict a preferred embodiment BNF grammar for variables, variable instantiations and common grammar for BNF grammars of permissions, groups and charters;

FIG. **30**C depicts a preferred embodiment BNF grammar for permissions and groups;

FIGS. **30**D through **30**E depict a preferred embodiment BNF grammar for charters;

FIGS. **31**A through **31**E depict a preferred embodiment set of command and operand candidates for Action Data Records (ADRs) facilitating discussing associated parameters of the ADRs of the present disclosure;

FIG. **32**A depicts a preferred embodiment of a National Language Support (NLS) directive command cross reference;

FIG. **32**B depicts a preferred embodiment of a NLS directive operand cross reference;

FIG. **33**A depicts a preferred embodiment American National Standards Institute (ANSI) X.409 encoding of the BNF grammar of FIGS. **30**A through **30**B for variables, variable instantiations and common grammar for BNF grammars of permissions and charters;

FIG. **33**B depicts a preferred embodiment ANSI X.409 encoding of the BNF grammar of FIG. **30**C for permissions and groups;

FIG. **33**C depicts a preferred embodiment ANSI X.409 encoding of the BNF grammar of FIGS. **30**D through **30**E for charters;

FIGS. **34**A through **34**G depict preferred embodiment C programming source code header file contents, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **35**A depicts a preferred embodiment of a Granting Data Record (GDR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **35**B depicts a preferred embodiment of a Grant Data Record (GRTDR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **35**C depicts a preferred embodiment of a Generic Assignment Data Record (GADR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **35**D depicts a preferred embodiment of a Privilege Data Record (PDR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **35**E depicts a preferred embodiment of a Group Data Record (GRPDR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **36**A depicts a preferred embodiment of a Description Data Record (DDR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **36**B depicts a preferred embodiment of a History Data Record (HDR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **36**C depicts a preferred embodiment of a Time specification Data Record (TDR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **36**D depicts a preferred embodiment of a Variable Data Record (VDR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **37**A depicts a preferred embodiment of a Charter Data Record (CDR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **37**B depicts a preferred embodiment of an Action Data Record (ADR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **37**C depicts a preferred embodiment of a Parameter Data Record (PARMDR) for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **38** depicts a flowchart for describing a preferred embodiment of MS permissions configuration processing;

FIGS. **39**A through **39**B depict flowcharts for describing a preferred embodiment of MS user interface processing for permissions configuration;

FIGS. **40**A through **40**B depict flowcharts for describing a preferred embodiment of MS user interface processing for grants configuration;

FIGS. **41**A through **41**B depict flowcharts for describing a preferred embodiment of MS user interface processing for groups configuration;

FIG. **42** depicts a flowchart for describing a preferred embodiment of a procedure for viewing MS configuration information of others;

FIG. **43** depicts a flowchart for describing a preferred embodiment of a procedure for configuring MS acceptance of data from other MSs;

FIG. **44**A depicts a flowchart for describing a preferred embodiment of a procedure for sending MS data to another MS;

FIG. **44**B depicts a flowchart for describing a preferred embodiment of receiving MS configuration data from another MS;

FIG. **45** depicts a flowchart for describing a preferred embodiment of MS charters configuration processing;

FIGS. **46**A through **46**B depict flowcharts for describing a preferred embodiment of MS user interface processing for charters configuration;

FIGS. **47**A through **47**B depict flowcharts for describing a preferred embodiment of MS user interface processing for actions configuration;

FIGS. **48**A through **48**B depict flowcharts for describing a preferred embodiment of MS user interface processing for parameter information configuration;

FIG. **49**A depicts an illustration for preferred permission data characteristics in the present disclosure LBX architecture;

FIG. **49**B depicts an illustration for preferred charter data characteristics in the present disclosure LBX architecture;

FIGS. **50**A through **50**C depict an illustration of data processing system wireless data transmissions over some wave spectrum;

FIG. **51**A depicts an example of a source code syntactical encoding embodiment of permissions, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **51**B depicts an example of a source code syntactical encoding embodiment of charters, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **52** depicts another preferred embodiment C programming source code header file contents, derived from the grammar of FIGS. **30**A through **30**E;

FIG. **53** depicts a preferred embodiment of a Prefix Registry Record (PRR) for discussing operations of the present disclosure;

FIG. **54** depicts an example of an XML syntactical encoding embodiment of permissions and charters, derived from the BNF grammar of FIGS. **30**A through **30**E;

FIG. **55**A depicts a flowchart for describing a preferred embodiment of MS user interface processing for Prefix Registry Record (PRR) configuration;

FIG. **55**B depicts a flowchart for describing a preferred embodiment of Application Term (AppTerm) data modification;

FIG. **56** depicts a flowchart for appropriately processing an encoding embodiment of the BNF grammar of FIGS. **30**A through **30**E, in context for a variety of parser processing embodiments;

FIG. **57** depicts a flowchart for describing a preferred embodiment of WDR In-process Triggering Smarts (WITS) processing;

FIG. **58** depicts an illustration for granted data characteristics in the present disclosure LBX architecture;

FIG. **59** depicts a flowchart for describing a preferred embodiment of a procedure for enabling LBX features and functionality in accordance with a certain type of permissions;

FIG. **60** depicts a flowchart for describing a preferred embodiment of a procedure for performing LBX actions in accordance with a certain type of permissions;

FIG. **61** depicts a flowchart for describing a preferred embodiment of performing processing in accordance with configured charters;

FIG. **62** depicts a flowchart for describing a preferred embodiment of a procedure for performing an action corresponding to a configured command;

FIG. **63**A depicts a flowchart for describing a preferred embodiment of a procedure for Send command action processing;

FIGS. **63**B-**1** through **63**B-**7** depicts a matrix describing how to process some varieties of the Send command;

FIG. **63**C depicts a flowchart for describing one embodiment of a procedure for Send command action processing, as derived from the processing of FIG. **63**A;

FIG. **64**A depicts a flowchart for describing a preferred embodiment of a procedure for Notify command action processing;

FIGS. **64**B-**1** through **64**B-**4** depicts a matrix describing how to process some varieties of the Notify command;

FIG. **64**C depicts a flowchart for describing one embodiment of a procedure for Notify command action processing, as derived from the processing of FIG. **64**A;

FIG. **65**A depicts a flowchart for describing a preferred embodiment of a procedure for Compose command action processing;

FIGS. **65**B-**1** through **65**B-**7** depicts a matrix describing how to process some varieties of the Compose command;

FIG. **65**C depicts a flowchart for describing one embodiment of a procedure for Compose command action processing, as derived from the processing of FIG. **65**A;

FIG. **66**A depicts a flowchart for describing a preferred embodiment of a procedure for Connect command action processing;

FIGS. **66**B-**1** through **66**B-**2** depicts a matrix describing how to process some varieties of the Connect command;

FIG. **66**C depicts a flowchart for describing one embodiment of a procedure for Connect command action processing, as derived from the processing of FIG. **66**A;

FIG. **67**A depicts a flowchart for describing a preferred embodiment of a procedure for Find command action processing;

FIGS. **67**B-**1** through **67**B-**13** depicts a matrix describing how to process some varieties of the Find command;

FIG. **67**C depicts a flowchart for describing one embodiment of a procedure for Find command action processing, as derived from the processing of FIG. **67**A;

FIG. **68**A depicts a flowchart for describing a preferred embodiment of a procedure for Invoke command action processing;

FIGS. **68**B-**1** through **68**B-**5** depicts a matrix describing how to process some varieties of the Invoke command;

FIG. **68**C depicts a flowchart for describing one embodiment of a procedure for Invoke command action processing, as derived from the processing of FIG. **68**A;

FIG. **69**A depicts a flowchart for describing a preferred embodiment of a procedure for Copy command action processing;

FIGS. **69**B-**1** through **69**B-**14** depicts a matrix describing how to process some varieties of the Copy command;

FIG. **69**C depicts a flowchart for describing one embodiment of a procedure for Copy command action processing, as derived from the processing of FIG. **69**A;

FIG. **70**A depicts a flowchart for describing a preferred embodiment of a procedure for Discard command action processing;

FIGS. **70**B-**1** through **70**B-**11** depicts a matrix describing how to process some varieties of the Discard command;

FIG. **70**C depicts a flowchart for describing one embodiment of a procedure for Discard command action processing, as derived from the processing of FIG. **70**A;

FIG. **71**A depicts a flowchart for describing a preferred embodiment of a procedure for Move command action processing;

FIGS. **71**B-**1** through **71**B-**14** depicts a matrix describing how to process some varieties of the Move command;

FIG. **71**C depicts a flowchart for describing one embodiment of a procedure for Move command action processing, as derived from the processing of FIG. **71**A;

FIG. **72**A depicts a flowchart for describing a preferred embodiment of a procedure for Store command action processing;

FIGS. **72**B-**1** through **72**B-**5** depicts a matrix describing how to process some varieties of the Store command;

FIG. **72**C depicts a flowchart for describing one embodiment of a procedure for Store command action processing, as derived from the processing of FIG. **72**A;

FIG. **73**A depicts a flowchart for describing a preferred embodiment of a procedure for Administration command action processing;

FIGS. **73**B-**1** through **73**B-**7** depicts a matrix describing how to process some varieties of the Administration command;

FIG. **73**C depicts a flowchart for describing one embodiment of a procedure for Administration command action processing, as derived from the processing of FIG. **73**A;

FIG. **74**A depicts a flowchart for describing a preferred embodiment of a procedure for Change command action processing;

FIG. **74**C depicts a flowchart for describing one embodiment of a procedure for Change command action processing, as derived from the processing of FIG. **74**A;

FIG. **75**A depicts a flowchart for describing a preferred embodiment of a procedure for sending data to a remote MS;

FIG. **75**B depicts a flowchart for describing a preferred embodiment of processing for receiving execution data from another MS;

FIG. **76** depicts a flowchart for describing a preferred embodiment of processing a special Term information paste action at a MS;

FIG. **77** depicts a flowchart for describing a preferred embodiment of configuring data to be maintained to WDR Application Fields; and

FIG. **78** depicts a simplified example of an XML syntactical encoding embodiment of a profile for the profile section of WDR Application Fields.

DETAILED DESCRIPTION OF THE INVENTION

With reference now to detail of the drawings, the present disclosure is described. Obvious error handling is omitted from the flowcharts in order to focus on the key aspects of the present disclosure. Obvious error handling includes database I/O errors, field validation errors, errors as the result of database table/data constraints or unique keys, data access errors, communications interface errors or packet collision, hardware failures, checksum validations, bit error detections/corrections, and any other error handling as well known to those skilled in the relevant art in context of this disclosure. A semicolon may be used in flowchart blocks to represent, and separate, multiple blocks of processing within a single physical block. This allows simpler flowcharts with less blocks in the drawings by placing multiple blocks of processing description in a single physical block of the flowchart. Flowchart processing is intended to be interpreted in the broadest sense by example, and not for limiting methods of accomplishing the same functionality. Preferably, field validation in the flowcharts checks for SQL injection attacks, communications protocol sniff and hack attacks, preventing of spoofing MS addresses, syntactical appropriateness, and semantics errors where appropriate. Disclosed user interface processing and/or screenshots are also preferred embodiment examples that can be implemented in other ways without departing from the spirit and scope of this disclosure. Alternative user interfaces (since this disclosure is not to be limiting) will use similar mechanisms, but may use different mechanisms without departing from the spirit and scope of this disclosure.

Locational terms such as whereabouts, location, position, area, destination, perimeter, radius, geofence, situational location, or any other related two or three dimensional locational term used herein to described position(s) and/or locations and/or whereabouts is to be interpreted in the broadest sense. Location field **1100***c* may include an area (e.g. on earth), a point (e.g. on earth), or a three dimensional bounds in space. In another example, a radius may define a sphere in space, rather than a circle in a plane. In some embodiments, a planet field forms part of the location (e.g. Earth, Mars, etc as part of field **1100***c*) for which other location information (e.g. latitude and longitude on Mars also part of field **1100***c*) is relative. In some embodiments, elevations (or altitudes) from known locatable point(s), distances from origin(s) in the universe, etc. can denote where exactly is a point of three dimensional space, or three dimensional sphere, area, or solid, is located. That same point can provide a mathematical reference to other points of the solid area/region in space. Descriptions for angles, pitches, rotations, etc from some reference point(s) may be further provided. Three dimensional areas/regions include a conical shape, cubical shape, spherical shape, pyramidal shape, irregular shapes, or any other shape either manipulated with a three dimensional graphic interface, or with mathematical model descriptions. Areas/regions in space can be occupied by a MS, passed through (e.g. by a traveler) by a MS, or referenced through configuration by a MS. In a three dimensional embodiment, nearby/nearness is determined in terms of three dimensional information, for example, a spherical radius around one MS intersecting a spherical radius around another MS. In a two dimensional embodiment, nearby/nearness is determined in terms of two dimensional information, for example, a circular radius around one MS intersecting a circular radius around another MS. Points can be specified as a point in a x-y-z plane, a point in polar coordinates, or the like, perhaps the center of a planet (e.g. Earth) or the Sun, some origin in the Universe, or any other origin for distinctly locating three dimensional location(s), positions, or whereabouts in space. Elevation (e.g. for earth, or some other planet, etc) may be useful to the three dimensional point of origin, and/or for the three dimensional region in space. A region in space may also be specified with connecting x-y-z coordinates together to bound the three dimensional region in space. There are many methods for representing a location (field **1100***c*) without departing from the spirit and scope of this disclosure. MSs, for example as carried by users, can travel by airplane through three dimensional areas/regions in space, or travel under the sea through three dimensional regions in space.

Various embodiments of communications between MSs, or an MS and service(s), will share channels (e.g. frequencies) to communicate, depending on when in effect. Sharing a channel will involve carrying recognizable and processable signature to distinguish transmissions for carrying data. Other embodiments of communications between MSs, or an MS and service(s), will use distinct channels to communicate, depending on when in effect. The number of channels that can be concurrently listened on and/or concurrently transmitted on by a data processing system will affect which embodiments are preferred. The number of usable channels will also affect which embodiments are preferred. This disclosure avoids unnecessary detail in different communication channel embodiments so as to not obfuscate novel material. Independent of various channel embodiments within the scope and spirit of the present disclosure, MSs communicate with other MSs in a peer to peer manner, in some aspects like automated walkie-talkies.

Novel features disclosed herein need not be provided as all or none. Certain features may be isolated in some MS embodiments, or may appear as any subset of features and functionality in other embodiments.

Location Based eXchanges (LBX) Architecture

FIG. **1**A depicts a preferred embodiment high level example componentization of a MS in accordance with the present disclosure. A MS **2** includes processing behavior referred to as LBX Character **4** and Other Character **32**. LBX character **4** provides processing behavior causing MS **2** to take on the character of a Location Based Exchange (LBX) MS according to the present disclosure. Other Character **32** provides processing behavior causing MS to take on character of prior art MSs in context of the type of MS. Other character **32** includes at least other processing code **34**, other processing data **36**, and other resources **38**, all of which are well known to those skilled in the art for prior art MSs. In some embodiments, LBX character **4** components may, or may not, make use of other character **32** components **34**, **36**, and **38**. Other character **32** components may, or may not, make use of LBX character **4** components **6** through **30**.

LBX character **4** preferably includes at least Peer Interaction Processing (PIP) code **6**, Peer Interaction Processing (PIP) data **8**, self management processing code **18**, self management processing data **20**, WDR queue **22**, send queue **24**, receive queue **26**, service informant code **28**, and LBX history **30**. Peer interaction processing (PIP) code **6** comprises executable code in software, firmware, or hardware form for carrying out LBX processing logic of the present disclosure when interacting with another MS. Peer interaction processing (PIP) data **8** comprises data maintained in any sort of memory of MS **2**, for example hardware memory, flash memory, hard disk memory, a removable memory device, or any other memory means accessible to MS **2**. PIP data **8** contains intelligence data for driving LBX processing logic of the present disclosure when interacting with other MSs. Self management processing code **18** comprises executable code in software, firmware, or hardware form for carrying out the local user interface LBX processing logic of the present disclosure. Self management processing data **20** contains intelligence data for driving processing logic of the present disclosure as disclosed for locally maintained LBX features. WDR queue **22** contains Whereabouts Data Records (WDRs) **1100**, and is a First-In-First-Out (FIFO) queue when considering housekeeping for pruning the queue to a reasonable trailing history of inserted entries (i.e. remove stale entries). WDR queue **22** is preferably designed with the ability of queue entry retrieval processing similar to Standard Query Language (SQL) querying, wherein one or more entries can be retrieved by querying with a conditional match on any data field(s) of WDR **1100** and returning lists of entries in order by an ascending or descending key on one or any ascending/descending ordered list of key fields.

All disclosed queues (e.g. **22**, **24**, **26**, **1980** and **1990** (See FIG. **19**)) are implemented with an appropriate thread-safe means of queue entry peeking (makes copy of sought queue entry without removing), discarding, retrieval, insertion, and queue entry field sorted search processing. Queues are understood to have an associated implicit semaphore to ensure appropriate synchronous access to queue data in a multi-threaded environment to prevent data corruption and misuse. Such queue interfaces are well known in popular operating systems. In MS operating system environments which do not have an implicit semaphore protected queue scheme, queue accesses in the present disclosure flowcharts are to be understood to have a previous request to a queue-assigned semaphore lock prior to queue access, and a following release of the semaphore lock after queue access. Operating systems without semaphore control may use methods to achieve similar thread-safe synchronization functionality. Queue functionality may be accomplished with lists, arrays, databases (e.g. SQL) and other methodologies without departing from the spirit and scope of queue descriptions herein.

Queue **22** alternate embodiments may maintain a plurality of WDR queues which segregate WDRs **1100** by field(s) values to facilitate timely processing. WDR queue **22** may be at least two (2) separate queues: one for maintaining the MS **2** whereabouts, and one for maintaining whereabouts of other MSs. WDR queue **22** may be a single instance WDR **1100** in some embodiments which always contains the most current MS **2** whereabouts for use by MS **2** applications (may use a sister queue **22** for maintaining WDRs from remote MSs). At least one entry is to be maintained to WDR queue **22** at all times for MS **2** whereabouts.

Send queue **24** (Transmit (Tx) queue) is used to send communications data, for example as intended for a peer MS within the vicinity (e.g. nearby as indicated by maximum range **1306**) of the MS **2**. Receive queue **26** (Receive (Rx) queue) is used to receive communications data, for example from peer MSs within the vicinity (e.g. nearby as indicated by maximum range **1306**) of the MS **2**. Queues **24** and **26** may also each comprise a plurality of queues for segregating data thereon to facilitate performance in interfacing to the queues, in particular when different queue entry types and/or sizes are placed on the queue. A queue interface for sending/receiving data to/from the MS is optimal in a multi-threaded implementation to isolate communications transport layers to processing behind the send/receive queue interfaces, but alternate embodiments may send/receive data directly from a processing thread disclosed herein. Queues **22**, **24**, and/or **26** may be embodied as a purely data form, or SQL database, maintained at MS **2** in persistent storage, memory, or any other storage means. In some embodiments, queues **24** and **26** are not necessary since other character **32** will already have accessible resources for carrying out some LBX character **4** processing.

Queue embodiments may contain fixed length records, varying length records, pointers to fixed length records, or pointers to varying length records. If pointers are used, it is assumed that pointers may be dynamically allocated for record storage on insertions and freed upon record use after discards or retrievals.

As well known to those skilled in the art, when a thread sends on a queue **24** in anticipation of a corresponding response, there is correlation data in the data sent which is sought in a response received by a thread at queue **26** so the sent data is correlated with the received data. In a preferred embodiment, correlation is built using a round-robin generated sequence number placed in data for sending along with a unique MS identifier (MS ID). If data is not already encrypted in communications, the correlation can be encrypted. While the unique MS identifier (MS ID) may help the MS identify which (e.g. wireless) data is destined for it, correlation helps identify which data at the MS caused the response. Upon receipt of data from a responder at queue **26**, correlation processing uses the returned correlation (e.g. field **1100***m*) to correlate the sent and received data. In preferred embodiments, the sequence number is incremented each time prior to use to ensure a unique number, otherwise it may be difficult to know which data received is a response to which data was sent, in particular when many data packets are sent within seconds. When the sequence number reaches a maximum value (e.g. 2**32−1), then it is round-robinned to 0 and is incremented from there all over again. This assures proper correlation of data between the MS and responders over time. There are other correlation schemes (e.g. signatures, random number generation, checksum counting, bit patterns, date/time stamp derivatives) to accomplish correlation functionality. If send and receive queues of Other Character **32** are used, then correlation can be used in a similar manner to correlate a response with a request (i.e. a send with a receipt).

There may be good reason to conceal the MS ID when transmitting it wirelessly. In this embodiment, the MS ID is a dependable and recognizable derivative (e.g. a pseudo MS ID) that can be detected in communications traffic by the MS having the pseudo MS ID, while concealing the true MS ID. This would conceal the true MS ID from would-be hackers sniffing wireless protocol. The derivative can always be reliably the same for simplicity of being recognized by the MS while being difficult to associate to a particular MS. Further still, a more protected MS ID (from would-be hackers that take time to deduce how an MS ID is scrambled) can itself be a dynamically changing correlation anticipated in forthcoming communications traffic, thereby concealing the real MS ID (e.g. phone number or serial number), in particular when anticipating traffic in a response, yet still useful for directing responses back to the originating MS (with the pseudo MS ID (e.g. correlation)). A MS would know which correlation is anticipated in a response by saving it to local storage for use until it becomes used (i.e. correlated in a matching response), or becomes stale. In another embodiment, a correlation response queue (like CR queue **1990**) can be deployed to correlate responses with requests that contain different correlations for pseudo MS IDs. In all embodiments, the MS ID (or pseudo MS ID) of the present disclosure should enable targeting communications traffic to the MS.

Service informant code **28** comprises executable code in software, firmware, or hardware form for carrying out of informing a supervisory service. The present disclosure does not require a connected web service, but there are features for keeping a service informed with activities of MS LBX. Service informant code **28** can communicate as requested any data **8**, **20**, **22**, **24**, **26**, **30**, **36**, **38**, or any other data processed at MS **2**.

LBX history **30** contains historical data useful in maintaining at MS **2**, and possibly useful for informing a supervisory service through service informant code **28**. LBX History **30** preferably has an associated thread of processing for keeping it pruned to the satisfaction of a user of MS **2** (e.g. prefers to keep last 15 days of specified history data, and 30 days of another specified history data, etc). With a suitable user interface to MS **2**, a user may browse, manage, alter, delete, or add to LBX History **30** as is relevant to processing described herein. Service informant code **28** may be used to cause sending of an outbound email, SMS message, outbound data packet, or any other outbound communication in accordance with LBX of the MS.

PIP data **8** preferably includes at least permissions **10**, charters **12**, statistics **14**, and a service directory **16**. Permissions **10** are configured to grant permissions to other MS users for interacting the way the user of MS **2** desires for them to interact. Therefore, permissions **10** contain permissions granted from the MS **2** user to other MS users. In another embodiment, permissions **10** additionally, or alternatively, contain permissions granted from other MS users to the MS **2** user. Permissions are maintained completely local to the MS **2**. Charters **12** provide LBX behavior conditional expressions for how MSs should interact with MS **2**. Charters **12** are configured by the MS **2** user for other MS users. In another embodiment, charters **12** additionally, or alternatively, are configured by other MS users for the MS **2** user. Some charters expressions depend on permissions **10**. Statistics **14** are maintained at MS **2** for reflecting peer (MS) to peer (MS) interactions of interest that occurred at MS **2**. In another embodiment, statistics **14** additionally, or alternatively, reflect peer (MS) to peer (MS) interactions that occurred at other MSs, preferably depending on permissions **10**. Service informant code **28** may, or may not, inform a service of statistics **14** maintained. Service directory **16** includes routing entries for how MS **2** will find a sought service, or how another MS can find a sought service through MS **2**.

In some embodiments, any code (e.g. **6**, **18**, **28**, **34**, **38**) can access, manage, use, alter, or discard any data (e.g. **8**, **20**, **22**, **24**, **26**, **30**, **36**, **38**) of any other component in MS **2**. Other embodiments may choose to keep processing of LBX character **4** and other character **32** disjoint from each other. Rectangular component boundaries are logical component representations and do not have to delineate who has access to what. MS (also MSs) references discussed herein in context for the new and useful features and functionality disclosed is understood to be an MS **2** (MSs **2**).

FIG. **1**B depicts a Location Based eXchanges (LBX) architectural illustration for discussing the present disclosure. LBX MSs are peers to each other for locational features and functionality. An MS **2** communicates with other MSs without requiring a service for interaction. For example, FIG. **1**B depicts a wireless network **40** of five (5) MSs. Each is able to directly communicate with others that are in the vicinity (e.g. nearby as indicated by maximum range **1306**). In a preferred embodiment, communications are limited reliability wireless broadcast datagrams having recognizable data packet identifiers. In another embodiment, wireless communications are reliable transport protocols carried out by the MSs, such as TCP/IP. In other embodiments, usual communications data associated with other character **32** include new data (e.g. Communications Key **1304**) in transmissions for being recognized by MSs within the vicinity. For example, as an MS conventionally communicates, LBX data is added to the protocol so that other MSs in the vicinity can detect, access, and use the data. The advantage to this is that as MSs use wireless communications to carry out conventional behavior, new LBX behavior is provided by simply incorporating additional information (e.g. Communications Key **1304**) to existing communications.

Regardless of the embodiment, an MS **2** can communicate with any of its peers in the vicinity using methods described below. Regardless of the embodiment, a communication path **42** between any two MSs is understood to be potentially bidirectional, but certainly at least unidirectional. The bidirectional path **42** may use one communications method for one direction and a completely different communications method for the other, but ultimately each can communicate to each other. When considering that a path **42** comprises two unidirectional communications paths, there are N*(N−1) unidirectional paths for N MSs in a network **40**. For example, 10 MSs results in 90 (i.e. 10*9) one way paths of communications between all 10 MSs for enabling them to talk to each other. Sharing of the same signaling channels is preferred to minimize the number of MS threads listening on distinct channels. Flowcharts are understood to process at incredibly high processing speeds, in particular for timely communications processing. While the MSs are communicating wirelessly to each other, path **42** embodiments may involve any number of intermediary systems or communications methods, for example as discussed below with FIG. **1**E.

FIG. **1**C depicts a Location Based Services (LBS) architectural illustration for discussing prior art of the present disclosure. In order for a MS to interact for LBS with another MS, there is service architecture **44** for accomplishing the interaction. For example, to detect that MS **1** is nearby MS N, the service is indispensably involved in maintaining data and carrying out processing. For example, to detect that MS **1** is arriving to, or departing from, a geofenced perimeter area configured by MS N, the service was indispensably involved in maintaining data and carrying out processing. For example, for MS N to locate MS **1** on a live map, the service was indispensably involved in maintaining data and carrying out processing. In another example, to grant and revoke permissions from MS1 to MS N, the service was indispensably involved in maintaining data and carrying out processing. While it is advantageous to require a single bidirectional path **46** for each MS (i.e. two unidirectional communications paths; (2*N) unidirectional paths for N MSs), there are severe requirements for service(s) when there are lots of MSs (i.e. when N is large). Wireless MSs have advanced beyond cell phones, and are capable of housing significant parallel processing, processing speed, increased wireless transmission speeds and distances, increased memory, and richer features.

FIG. **1**D depicts a block diagram of a data processing system useful for implementing a MS, ILM, DLM, centralized server, or any other data processing system described herein. An MS **2** is a data processing system **50**. Data processing system **50** includes at least one processor **52** (e.g. Central Processing Unit (CPU)) coupled to a bus **54**. Bus **54** may include a switch, or may in fact be a switch **54** to provide dedicated connectivity between components of data processing system **50**. Bus (and/or switch) **54** is a preferred embodiment coupling interface between data processing system **50** components. The data processing system **50** also includes main memory **56**, for example, random access memory (RAM). Memory **56** may include multiple memory cards, types, interfaces, and/or technologies. The data processing system **50** may include secondary storage devices **58** such as persistent storage **60**, and/or removable storage device **62**, for example as a compact disk, floppy diskette, USB flash, or the like, also connected to bus (or switch) **54**. In some embodiments, persistent storage devices could be remote to the data processing system **50** and coupled through an appropriate communications interface. Persistent storage **60** may include flash memory, disk drive memory, magnetic, charged, or bubble storage, and/or multiple interfaces and/or technologies, perhaps in software interface form of variables, a database, shared memory, etc.

The data processing system **50** may also include a display device interface **64** for driving a connected display device (not shown). The data processing system **50** may further include one or more input peripheral interface(s) **66** to input devices such as a keyboard, keypad, Personal Digital Assistant (PDA) writing implements, touch interfaces, mouse, voice interface, or the like. User input ("user input", "user events" and "user actions" used interchangeably) to the data processing system are inputs accepted by the input peripheral interface(s) **66**. The data processing system **50** may still further include one or more output peripheral interface(s) **68** to output devices such as a printer, facsimile device, or the like. Output peripherals may also be available via an appropriate interface.

Data processing system **50** will include a communications interface(s) **70** for communicating to another data processing system **72** via analog signal waves, digital signal waves, infrared proximity, copper wire, optical fiber, or other wave spectrums described herein. A MS may have multiple communications interfaces **70** (e.g. cellular connectivity, 802.x, etc). Other data processing system **72** may be an MS. Other data processing system **72** may be a service. Other data processing system **72** is a service data processing system when MS **50** communicates to other data processing system **72** by way of service informant code **28**. In any case, the MS and other data processing system are said to be interoperating when communicating.

Data processing system programs (also called control logic) may be completely inherent in the processor(s) **52** being a customized semiconductor, or may be stored in main memory **56** for execution by processor(s) **52** as the result of a read-only memory (ROM) load (not shown), or may be loaded from a secondary storage device into main memory **56** for execution by processor(s) **52**. Such programs, when executed, enable the data processing system **50** to perform features of the present disclosure as discussed herein. Accordingly, such data processing system programs represent controllers of the data processing system.

In some embodiments, the disclosure is directed to a control logic program product comprising at least one processor **52** having control logic (software, firmware, hardware microcode) stored therein. The control logic, when executed by processor(s) **52**, causes the processor(s) **52** to provide functions of the disclosure as described herein. In another embodiment, this disclosure is implemented primarily in hardware, for example, using a prefabricated component state machine (or multiple state machines) in a semiconductor element such as a processor **52**.

Those skilled in the art will appreciate various modifications to the data processing system **50** without departing from the spirit and scope of this disclosure. A data processing system, and more particularly a MS, preferably has capability for many threads of simultaneous processing which provide control logic and/or processing. These threads can be embodied as time sliced threads of processing on a single hardware processor, multiple processors, multi-core processors, Digital Signal Processors (DSPs), or the like, or combinations thereof. Such multi-threaded processing can concurrently serve large numbers of concurrent MS tasks. Concurrent processing may be provided with distinct hardware processing and/or as appropriate software driven time-sliced thread processing. Those skilled in the art recognize that having multiple threads of execution on an MS is accomplished in many different ways without departing from the spirit and scope of this disclosure. This disclosure strives to deploy software to existing MS hardware configurations, but the disclosed software can be deployed as burned-in microcode to new hardware of MSs.

Data processing aspects of drawings/flowcharts are preferably multi-threaded so that many MSs and applicable data processing systems are interfaced with in a timely and optimal manner. Data processing system **50** may also include its own clock mechanism (not shown), if not an interface to an atomic clock or other clock mechanism, to ensure an appropriately accurate measurement of time in order to appropriately carry out processing described below. In some embodiments, Network Time Protocol (NTP) is used to keep a consistent universal time for MSs and other data processing systems in communications with MSs. This is most advantageous to prevent unnecessary round-tripping of data between data processing systems to determine timing (e.g. Time Difference of Arrival (TDOA)) measurements. A NTP synchronized date/time stamp maintained in communications is compared by a receiving data processing system for comparing with its own NTP date/time stamp to measure TOA (time of arrival (i.e. time taken to arrive)). Of course, in the absence of NTP used by the sender and receiver, TOA is also calculated in a bidirectional transmission using correlation. In this disclosure, TOA measurements from one location technology are used for triangulating with TOA measurements from another location technology, not just for determining “how close”. Therefore, TDOA terminology is generally used herein to refer to the most basic TOA measurement of a wave spectrum signal being the difference between when it was sent and when it was received. TDOA is also used to describe using the difference of such measurements to locate (triangulate). NTP use among participating systems has the advantage of a single unidirectional broadcast data packet containing all a receiving system requires to measure TDOA, by knowing when the data was sent (date/time stamp in packet) and when the data was received (signal detected and processed by receiving system). A NTP clock source (e.g. atomic clock) used in a network is to be reasonably granular to carry out measurements, and ensures participating MSs are updated timely according to anticipated time drifts of their own clocks. There are many well known methods for accomplishing NTP, some which require dedicated thread(s) for NTP processing, and some which use certain data transmitted to and from a source to keep time in synch.

Those skilled in the art recognize that NTP accuracy depends on participating MS clocks and processing timing, as well as time server source(s). Radio wave connected NTP time server(s) is typically accurate to as granular as 1 millisecond. Global Positioning System (GPS) time servers provide accuracy as granular as 50 microseconds. GPS timing receivers provide accuracy to around 100 nanoseconds, but this may be reduced by timing latencies in time server operating systems. With advancements in hardware, microcode, and software, obvious improvements are being made to NTP. In NTP use embodiments of this disclosure, an appropriate synchronization of time is used for functional interoperability between MSs and other data processing systems using NTP. NTP is not required in this disclosure, but it is an advantage when in use.

LBX Directly Located Mobile Data Processing Systems (DLMs)

FIG. **1**E depicts a network illustration for discussing various deployments of whereabouts processing aspects of the present disclosure. In some embodiments, a cellular network cluster **102** and cellular network cluster **104** are parts of a larger cellular network. Cellular network cluster **102** contains a controller **106** and a plurality of base stations, shown generally as base stations **108**. Each base station covers a single cell of the cellular network cluster, and each base station **108** communicates through a wireless connection with the controller **106** for call processing, as is well known in the art. Wireless devices communicate via the nearest base station (i.e. the cell the device currently resides in), for example base station **108***b*. Roaming functionality is provided when a wireless device roams from one cell to another so that a session is properly maintained with proper signal strength. Controller **106** acts like a telephony switch when a wireless device roams across cells, and it communicates with controller **110** via a wireless connection so that a wireless device can also roam to other clusters over a larger geographical area. Controller **110** may be connected to a controller **112** in a cellular cluster through a physical connection, for example, copper wire, optical fiber, or the like. This enables cellular clusters to be great distances from each other. Controller **112** may in fact be connected with a physical connection to its base stations, shown generally as base stations **114**. Base stations may communicate directly with the controller **112**, for example, base station **114***e*. Base stations may communicate indirectly to the controller **112**, for example base station **114***a* by way of base station **114***d*. It is well known in the art that many options exist for enabling interoperating communications between controllers and base stations for the purpose of managing a cellular network. A cellular network cluster **116** may be located in a different country. Base controller **118** may communicate with controller **110** through a Public Service Telephone Network (PSTN) by way of a telephony switch **120**, PSTN **122**, and telephony switch **124**, respectively. Telephony switch **120** and telephony switch **124** may be private or public. In one cellular network embodiment of the present disclosure, the services execute at controllers, for example controller **110**. In some embodiments, the MS includes processing that executes at a wireless device, for example mobile laptop computer **126**, wireless telephone **128**, a personal digital assistant (PDA) **130**, an iPhone **170**, or the like. As the MS moves about, positional attributes are monitored for determining location. The MS may be handheld, or installed in a moving vehicle. Locating a wireless device using wireless techniques such as Time Difference of Arrival (TDOA) and Angle Of Arrival (AOA) are well known in the art. The service may also execute on a server computer accessible to controllers, for example server computer **132**, provided an appropriate timely connection exists between cellular network controller(s) and the server computer **132**. Wireless devices (i.e. MSs) are preferably known by a unique identifier, for example a phone number, caller id, device identifier, or like appropriate unique handle.

In another embodiment of the present disclosure, GPS satellites such as satellite **134**, satellite **136**, and satellite **138** provide information, as is well known in the art, to GPS devices on earth for triangulation locating of the GPS device. In this embodiment, a MS has integrated GPS functionality so that the MS monitors its positions. The MS is preferably known by a unique identifier, for example a phone number, caller id, device identifier, or like appropriate unique handle.

In yet another embodiment of the present disclosure, a physically connected device, for example, telephone **140**, computer **142**, PDA **144**, telephone **146**, and fax machine **148**, may be newly physically connected to a network. Each is a MS, although the mobility is limited. Physical connections include copper wire, optical fiber, USB, or any other physical connection, by any communications protocol thereon. Devices are preferably known by a unique identifier, for example a phone number, caller id, device identifier, physical or logical network address, or like appropriate unique handle. The MS is detected for being newly located when physically connected. A service can be communicated to upon detecting connectivity. The service may execute at an Automatic Response Unit (ARU) **150**, a telephony switch, for example telephony switch **120**, a web server **152** (for example, connected through a gateway **154**), or a like data processing system that communicates with the MS in any of a variety of ways as well known to those skilled the art. MS detection may be a result of the MS initiating a communication with the service directly or indirectly. Thus, a user may connect his laptop to a hotel network, initiate a communication with the service, and the service determines that the user is in a different location than the previous communication. A local area network (LAN) **156** may contain a variety of connected devices, each an MS that later becomes connected to a local area network **158** at a different location, such as a PDA **160**, a server computer **162**, a printer **164**, an internet protocol telephone **166**, a computer **168**, or the like. Hard copy presentation could be made to printer **164** and fax **148**.

Current technology enables devices to communicate with each other, and other systems, through a variety of heterogeneous system and communication methods. Current technology allows executable processing to run on diverse devices and systems. Current technology allows communications between the devices and/or systems over a plethora of methodologies at close or long distance. Many technologies also exist for automatic locating of devices. It is well known how to have an interoperating communications system that comprises a plurality of individual systems communicating with each other with one or more protocols. As is further known in the art of developing software, executable processing of the present disclosure may be developed to run on a particular target data processing system in a particular manner, or customized at install time to execute on a particular data processing system in a particular manner.

FIG. **2**A depicts an illustration for describing automatic location of a MS, for example a DLM **200**, through the MS coming into range of a stationary cellular tower. A DLM **200**, or any of a variety of MSs, travels within range of a cell tower, for example cell tower **108***b*. The known cell tower location is used to automatically detect the location of the DLM **200**. In fact, any DLM that travels within the cell served by cell tower **108***b* is identified as the location of cell tower **108***b*. The confidence of a location of a DLM **200** is low when the cell coverage of cell tower **108***b* is large. In contrast, the confidence of a location of a DLM **200** is higher when the cell coverage of cell tower **108***b* is smaller. However, depending on the applications locating DLMs using this method, the locating can be quite acceptable. Location confidence is improved with a TDOA measurement for the elapsed time of communication between DLM **200** and cell tower to determine how close the MS is to the cell tower. Cell tower **108***b* can process all locating by itself, or with interoperability to other services as connected to cell tower **108***b* in FIG. **1**E. Cell tower **108***b* can communicate the location of DLM **200** to a service, to the DLM **200**, to other MSs within its coverage area, any combination thereof, or to any connected data processing system, or MS, of FIG. **1**E.

FIG. **2**B depicts an illustration for describing automatic location of a MS, for example a DLM **200**, through the MS coming into range of some stationary antenna. DLM **200**, or any of a variety of MSs, travels within range of a stationary antenna **202** that may be mounted to a stationary object **204**. The known antenna location is used to automatically detect the location of the DLM **200**. In fact, any DLM that travels within the coverage area served by antenna **202** is identified as the location of antenna **202**. The confidence of a location of a DLM **200** is low when the antenna coverage area of antenna **202** is large. In contrast, the confidence of a location of a DLM **200** is higher when the antenna coverage area of antenna **202** is smaller. However, depending on the applications locating DLMs using this method, the locating can be quite acceptable. Location confidence is improved with a TDOA measurement for the elapsed time of communication between DLM **200** and a particular antenna to determine how close the MS is to the antenna. Antenna **202** can process all locating by itself (with connected data processing system (not shown) as well known to those skilled in the art), or with interoperability to other services as connected to antenna **202**, for example with connectivity described in FIG. **1**E. Antenna **202** can be used to communicate the location of DLM **200** to a service, to the DLM **200**, to other MSs within its coverage area, any combination thereof, or to any connected data processing system, or MS, of FIG. **1**E.

FIG. **2**C depicts an illustration for discussing an example of automatically locating a MS, for example a DLM **200**, through the MS coming into range of some stationary antenna. DLM **200**, or any of a variety of MSs, travels within range of a stationary antenna **212** that may be mounted to a stationary object, such as building **210**. The known antenna location is used to automatically detect the location of the DLM **200**. In fact, any DLM that travels within the coverage area served by antenna **212** is identified as the location of antenna **212**. The confidence of a location of a DLM **200** is low when the antenna coverage area of antenna **212** is large. In contrast, the confidence of a location of a DLM **200** is higher when the antenna coverage area of antenna **212** is smaller. However, depending on the applications locating DLMs using this method, the locating can be quite acceptable. Location confidence is improved with a TDOA measurement as described above. Antenna **212** can process all locating by itself (with connected data processing system (not shown) as well known to those skilled in the art), or with interoperability to other services as connected to antenna **212**, for example with connectivity described in FIG. **1**E. Antenna **212** can be used to communicate the location of DLM **200** to a service, to the DLM **200**, to other MSs within its coverage area, any combination thereof, or to any connected data processing system, or MS, of FIG. **1**E.

Once DLM **200** is within the building **210**, a strategically placed antenna **216** with a desired detection range within the building is used to detect the DLM **200** coming into its proximity. Wall breakout **214** is used to see the antenna **216** through the building **210**. The known antenna **216** location is used to automatically detect the location of the DLM **200**. In fact, any DLM that travels within the coverage area served by antenna **216** is identified as the location of antenna **216**. The confidence of a location of a DLM **200** is low when the antenna coverage area of antenna **216** is large. In contrast, the confidence of a location of a DLM **200** is higher when the antenna coverage area of antenna **216** is smaller. Travels of DLM **200** can be limited by objects, pathways, or other limiting circumstances of traffic, to provide a higher confidence of location of DLM **200** when located by antenna **216**, or when located by any locating antenna described herein which detects MSs coming within range of its location. Location confidence is improved with a TDOA measurement as described above. Antenna **216** can process all locating by itself (with connected data processing system (not shown) as well known to those skilled in the art), or with interoperability to other services as connected to antenna **216**, for example with connectivity described in FIG. **1**E. Antenna **216** can be used to communicate the location of DLM **200** to a service, to the DLM **200**, to other MSs within its coverage area, any combination thereof, or to any connected data processing system, or MS, of FIG. **1**E. Other in-range detection antennas of a FIG. **2**C embodiment may be strategically placed to facilitate warehouse operations such as in Kubler et al.

FIG. **2**D depicts a flowchart for describing a preferred embodiment of a service whereabouts update event of an antenna in-range detected MS, for example a DLM **200**, when MS location awareness is monitored by a stationary antenna, or cell tower (i.e. the service thereof). FIGS. **2**A through **2**C location detection processing are well known in the art. FIG. **2**D describes relevant processing for informing MSs of their own whereabouts. Processing begins at block **230** when a MS signal deserving a response has been received and continues to block **232** where the antenna or cell tower service has authenticated the MS signal. A MS signal can be received for processing by blocks **230** through **242** as the result of a continuous, or pulsed, broadcast or beaconing by the MS (FIG. **13**A), perhaps as part of usual communication protocol in progress for the MS (FIG. **13**A usual data **1302** with embedded Communications Key (CK) **1304**), or an MS response to continuous, or pulsed, broadcast or beaconing via the service connected antenna (FIG. **13**C). MS and/or service transmission can be appropriately correlated for a response (as described above) which additionally facilitates embodiments using TDOA measurements (time of communications between the MS and antenna, or cell tower) to determine at least how close is the MS in range (or use in conjunction with other data to triangulate the MS location). The MS is preferably authenticated by a unique MS identifier such as a phone number, address, name, serial number, or any other unique handle to the MS. In this, and any other embodiments disclosed, an MS may be authenticated using a group identifier handle indicating membership to a supported/known group deserving further processing. Authentication will preferably consult a database for authenticating that the MS is known. Block **232** continues to block **234** where the signal received is immediately responded back to the MS, via the antenna, containing at least correlation along with whereabouts information for a Whereabouts Data Record (WDR) **1100** associated with the antenna (or cell tower). Thereafter, the MS receives the correlated response containing new data at block **236** and completes a local whereabouts data record **1100** (i.e. WDR **1100**) using data received along with other data determined by the MS.

In another embodiment, blocks **232** through **234** are not required. A service connected antenna (or cell tower) periodically broadcasts its whereabouts (WDR info (e.g. FIG. **13**C)) and MSs in the vicinity use that directly at block **236**. The MS can choose to use only the confidence and location provided, or may determine a TDOA measurement for determining how close it is. If the date/time stamp field **1100***b* indicates NTP is in use by the service, and the MS is also using NTP, then a TDOA measurement can be determined using the one unidirectional broadcast via the antenna by using the date/time stamp field **1100***b* received with when the WDR information was received by the MS (subtract time difference and use known wave spectrum for distance). If either the service or MS is not NTP enabled, then a bidirectional correlated data flow between the service and MS is used to assess a TDOA measurement in terms of time of the MS. One embodiment provides the TDOA measurement from the service to the MS. Another embodiment calculates the TDOA measurement at the MS.

Network Time protocol (NTP) can ensure MSs have the same atomic clock time as the data processing systems driving antennas (or cell towers) they will encounter. Then, date/time stamps can be used in a single direction (unidirectional) broadcast packet to determine how long it took to arrive to/from the MS. In an NTP embodiment, the MS (FIG. **13**A) and/or the antenna (FIG. **13**C) sends a date/time stamp in the pulse, beacon, or protocol. Upon receipt, the antenna (or cell tower) service data processing system communicates how long the packet took from an MS to the antenna (or cell tower) by comparing the date/time stamp in the packet and a date/time stamp of when it was received. The service may also set the confidence value, before sending WDR information to the MS. Similarly, an MS can compare a date/time stamp in the unidirectional broadcast packet sent from a locating service (FIG. **13**C) with when received by the MS. So, NTP facilitates TDOA measurements in a single broadcast communication between systems through incorporation to usual communications data **1302** with a date/time stamp in Communications Key (CK) **1304**, or alternatively in new data **1302**. Similarly, NTP facilitates TDOA measurement in a single broadcast communication between systems through incorporation to usual communications data **1312** with a date/time stamp in Communications Key (CK) **1314**, or alternatively in new data **1312**.

The following template is used in this disclosure to highlight field settings. See FIG. **11**A descriptions. Fields are set to the following upon exit from block **236**:

MS ID field **1100***a* is preferably set with: Unique MS identifier of the MS invoking block **240**. This field is used to uniquely distinguish this MS WDRs on queue **22** from other originated WDRs.

DATE/TIME STAMP field **1100***b* is preferably set with: Date/time stamp for WDR completion at block **236** to the finest granulation of time achievable by the MS. The NTP use indicator is set appropriately.

LOCATION field **1100***c* is preferably set with: Location of stationary antenna (or cell tower) as communicated by the service to the MS.

CONFIDENCE field **1100***d* is preferably set with: The same value (e.g. 76) for any range within the antenna (or cell tower), or may be adjusted using the TDOA measurement (e.g. amount of time detected by the MS for the response at block **234**). The longer time it takes between the MS sending a signal detected at block **232** and the response with data back received by the MS (block **234**), the less confidence there is for being located because the MS must be a larger distance from the antenna or cell tower. The less time it takes between the MS sending a signal detected at block **232** and the response with data back, the more confidence there is for being located because the MS must be a closer distance to the antenna or cell tower. Confidence values are standardized for all location technologies. In some embodiments of FIG. **2**D processing, a confidence value can be set for 1 through 100 (1 being lowest confidence and 100 being highest confidence) wherein a unit of measurement between the MS and antenna (or cell tower) is used directly for the confidence value. For example, 20 meters is used as the unit of measurement. For each unit of 20 meters distance determined by the TDOA measurement, assign a value of 1, up to a worst case of 100 (i.e. 2000 meters). Round the 20 meter unit of distance such that 0 meters to <25 meters is 20 meters (i.e. 1 unit of measurement), 26 meters to <45 meters is 40 meters (i.e. 2 units of measurement), and so on. Once the number of units is determined, subtract that number from 101 for the confidence value (i.e. 1 unit=confidence value 100, 20 units=confidence value 81; 100 units or greater=confidence value of 1). Yet another embodiment will use a standard confidence value for this “coming in range” technology such as 76 and then further increase or decrease the confidence using the TDOA measurement. Many embodiments exist for quantifying a higher versus lower confidence. In any case, a confidence value (e.g. 76) is determined by the MS, service, or both (e.g. MS uses TDOA measurement to modify confidence sent by service).

LOCATION TECHNOLOGY field **1100***e* is preferably set with: "Server Antenna Range" for an antenna detecting the MS, and is set to "Server Cell Range" for a cell tower detecting the MS. The originator indicator is set to DLM.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: The period of time for communications between the antenna and the MS (a TDOA measurement), if known; a communications signal strength, if available; wave spectrum used (e.g. from MS receive processing), if available; particular communications interface **70**, if available. The TDOA measurement may be converted to a distance using wave spectrum information. The values populated here should have already been factored into the confidence value at block **236**.

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: Parameters uniquely identifying a/the service (e.g. antenna (or cell tower)) and how to best communicate with it again, if available. May not be set, regardless if received from the service.

SPEED field **1100***h* is preferably set with: Data received by MS at block **234**, if available.

HEADING field **1100***i* is preferably set with: Data received by MS at block **234**, if available.

ELEVATION field **1100***j* is preferably set with: data received by MS at block **234**, if available. Elevation field **1100***j* is preferably associated with the antenna (or cell tower) by the elevation/altitude of the antenna (or cell tower).

APPLICATION FIELDS field **1100***k* is preferably set with: Data received at block **234** by the MS, or set by data available to the MS, or set by both the locating service for the antenna (or cell tower) and the MS itself. Application fields include, and are not limited to, MS navigation APIs in use, social web site identifying information, application information for applications used, accessed, or in use by the MS, or any other information complementing whereabouts of the MS.

CORRELATION FIELD **1100***m* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

SENT DATE/TIME STAMP field **1100***n* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

RECEIVED DATE/TIME STAMP field **1100***p* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

A service connected to the antenna (or cell tower) preferably uses historical information and artificial intelligence interrogation of MS travels to determine fields **1100***h* and **1100***i*. Block **236** continues to block **238** where parameters are prepared for passing to FIG. **2**F processing invoked at block **240**. Parameters are set for: WDRREF=a reference or pointer to the WDR; DELETEQ=FIG. **2**D location queue discard processing; and SUPER=FIG. **2**D supervisory notification processing. Thereafter, block **240** invokes FIG. **2**F processing and FIG. **2**D processing terminates at block **242**. FIG. **2**F processing will insert to queue **22** so this MS knows at least its own whereabouts whenever possible. A single data instance embodiment of WDR queue **22** will cause FIG. **2**F to update the single record of WDR information for being current upon exit from block **240** (this is true for all flowchart blocks invoking FIG. **2**F processing).

With reference now to FIG. **2**F, depicted is a flowchart for describing a preferred embodiment of a procedure for inserting a Whereabouts Data Record (WDR) **1100** to MS WDR queue **22**. Appropriate semaphores are used for variables which can be accessed simultaneously by another thread other than the caller. With reference now to FIG. **2**F, procedure processing starts at block **270** and continues to block **272** where parameters passed from the invoking block of processing, for example block **240**, are determined. The variable WDRREF is set by the caller to a reference or pointer to the WDR so subsequent blocks of FIG. **2**F can access the WDR. The variable DELETEQ is set by the caller so that block **292** knows how to discard obsolete location queue entries. The DELETEQ variable can be a multi-field record (or reference thereof) for how to prune. The variable SUPER is set by the caller so that block **294** knows under what condition(s), and which data, to contact a supervisory service. The SUPER variable can be a multi-field record (or reference thereof) for instruction.

Block **272** continues to block **274** where the DLMV (see FIG. **12** and later discussions for DLMV (DLM role(s) List Variable)), or ILMV (see FIG. **12** and later discussions for ILMV (ILM role(s) List Variable)), is checked for an enabled role matching the WDR for insertion (e.g. DLM: location technology field **1100***e* (technology and originator indicator) when MS ID=this MS; ILM: DLM or ILM indicator when MS ID not this MS). If no corresponding DLMV/ILMV role is enabled for the WDR to insert, then processing continues to block **294** (the WDR is not inserted to queue **22**). If the ILMV/DLMV role for the WDR is enabled, then processing continues to block **276** where the confidence of the WDR **1100** is validated prior to insertion. An alternate embodiment to FIG. **2**F will not have block **274** (i.e. block **272** continues directly to block **276**) since appropriate DLM and/or ILM processing may be terminated anyway when DLM/ILM role(s) are disabled (see FIG. **14**A/B).

If block **276** determines the data to be inserted is not of acceptable confidence (e.g. field **1100***d*<confidence floor value (see FIG. **14**A/B)), then processing continues to block **294** described below. If block **276** determines the data to be inserted is of acceptable confidence (e.g. field **1100***d*>70), then processing continues to block **278** for checking the intent of the WDR insertion.

If block **278** determines the WDR for insert is a WDR describing whereabouts for this MS (i.e. MS ID matching MS of FIG. **2**F processing (DLM: FIGS. **2**A through **9**B, or ILM: FIG. **26**A/B)), then processing continues to block **280**. If block **278** determines the WDR for insert is from a remote ILM or DLM (i.e. MS ID does not match MS of FIG. **2**F processing), then processing continues to block **290**. Block **280** peeks the WDR queue **22** for the most recent highest confidence entry for this MS whereabouts by searching queue **22** for: the MS ID field **1100***a* matching the MS ID of FIG. **2**F processing, and a confidence field **1100***d* greater than or equal to the confidence floor value, and a most recent date/time stamp field **1100***b*. Thereafter, if block **282** determines one was found, then processing continues to block **284**, otherwise processing continues to block **286** where a Last Whereabouts date/Time stamp (LWT) variable is set to field **1100***b* of the WDR for insert (e.g. first MS whereabouts WDR), and processing continues to block **288**.

If block **284** determines the WDR for insertion has significantly moved (i.e. using a movement tolerance configuration (e.g. 3 meters) with fields **1100***c* of the WDR for insert and the WDR peeked at block **280**), then block **286** sets the LWT (Last Whereabouts date/Time stamp) variable (with appropriate semaphore) to field **1100***b* of the WDR for insert, and processing continues to block **288**, otherwise processing continues directly to block **288** (thereby keeping the LWT as its last setting). The LWT is to hold the most recent date/time stamp of when the MS significantly moved as defined by a movement tolerance. The movement tolerance can be system defined or configured, or user configured in FIG. **14** by an option for configuration detected at block **1408**, and then using the Configure Value procedure of FIG. **18** (like confidence floor value configuration).

Block **288** accesses the DLMV and updates it with a new DLM role if there is not one present for it. This ensures a correct list of DLMV roles are available for configuration by FIG. **14**. Preferably, by default an unanticipated DLMV role is enabled (helps inform the user of its availability). Likewise in another embodiment, ILMV roles can be similarly updated, in particular if a more granulated list embodiment is maintained to the ILMV, or if unanticipated results help to identify another configurable role. By default, block **274** should allow unanticipated roles to continue with WDR insertion processing, and then block **288** can add the role, enable it, and a user can decide what to do with it in configuration (FIG. **14**A/B).

Thereafter, the WDR **1100** is inserted to the WDR queue **22** at block **290**, block **292** discards any obsolete records from the queue as directed by the caller (invoker), and processing continues to block **294**. The WDR queue **22** preferably contains a list of historically MS maintained Whereabouts Data Records (WDRs) as the MS travels. When the MS needs its own location, for example from an application access, or to help locate an ILM, the queue is accessed for returning the WDR with the highest confidence value (field **1100***d*) in the most recent time (field **1100***b*) for the MS (field **1100***a*). Block **292** preferably discards by using fields **1100***b* and **1100***d* relative to other WDRs. The queue should not be allowed to get too large. This will affect memory (or storage) utilization at the MS as well as timeliness in accessing a sought queue entry. Block **292** also preferably discards WDRs from queue **22** by moving selected WDRs to LBX History **30**.

As described above, queue interfaces assume an implicit semaphore for properly accessing queue **22**. There may be ILMs requesting to be located, or local applications of the MS may request to access the MS whereabouts. Executable thread(s) at the MS can accesses the queue in a thread-safe manner for responding to those requests. The MS may also have multiple threads of processing for managing whereabouts information from DLMs, ILMs, or stationary location services. The more concurrently executable threads available to the MS, the better the MS is able to locate itself and respond to others (e.g. MSs). There can be many location systems and methods used to keeping a MS informed of its own whereabouts during travel. While the preferred embodiment is to maximize thread availability, the obvious minimum requirement is to have at least 1 executable thread available to the MS. As described above, in operating system environments without proper queue interfaces, queue access blocks are first preceded by an explicit request for a semaphore lock to access queue **22** (waits until obtained), and then followed by a block for releasing the semaphore lock to another thread for use. Also, in the present disclosure it is assumed in blocks which access data accessible to more than 1 concurrent thread (e.g. shared memory access to DLMV or ILMV at block **274**) that an appropriate semaphore (created at block **1220**) protect synchronous access.

If block **294** determines information (e.g. whereabouts) should be communicated by service informant code **28** to a supervisory service, for example a service **1050**, then block **296** communicates specified data to the service and processing terminates at block **298** by returning to the invoker (caller). If block **294** determines a supervisory service is not to be informed, then processing terminates with an appropriate return to the caller at block **298**. Service informant code **28**, at block **296**, can send information as data that is reliably acknowledged on receipt, or as a datagram which most likely (but unreliably) is received.

Depending on the SUPER variable, block **294** may opt to communicate every time a WDR is placed to the queue, or when a reasonable amount of time has passed since last communicating to the supervisory service, or when a WDR confidence reaches a certain sought value, or when any WDR field or fields contain certain sought information, or when a reasonably large number of entries exist in WDR queue **22**, or for any processing condition encountered by blocks **270** through **298**, or for any processing condition encountered by caller processing up to the invocation of FIG. **2**F processing. Different embodiments will send a single WDR **1100** at block **296**, a plurality of WDRs **1100**, or any other data. Various SUPER parameter(s) embodiments for FIG. **2**F caller parameters can indicate what, when, where and how to send certain data. Block **296** may send an email, an SMS message, or use other means for conveying data. Service informant code **28** may send LBX history **30**, statistics **14** and/or any other data **8**, data **20**, queue data, data **36** or resources **38**. Service informant code **28** may update data in history **30**, statistics **14** or any other data **8**, data **20**, queue data, data **36** and/or resources **38**, possibly using conditions of this data to determine what is updated. Blocks **294** and **296** may be omitted in some embodiments.

If a single WDR is sent at block **296** as passed to FIG. **2**F processing, then the WDR parameter determined at block **272** is accessed. If a plurality of WDRs is sent at block **296**, then block **296** appropriately interfaces in a thread-safe manner to queue **22**, and sends the WDRs.

Some preferred embodiments do not incorporate blocks **278** through **286**. (i.e. block **276** continues to block **288** if confidence ok). Blocks **278** through **286** are for the purpose of implementing maintaining a date/time stamp of last MS significant movement (using a movement tolerance). Architecture **1900** uses FIG. **2**F, as does DLM processing. FIG. **2**F must perform well for the preferred multithreaded architecture **1900**. Block **280** performs a peek, and block **284** can be quite timely depending on embodiments used for location field **1100***c*. A movement tolerance incorporated at the MS is not necessary, but may be nice to have. Therefore, blocks **278** through **286** are optional blocks of processing.

FIG. **2**F may also maintain (with appropriate semaphore) the most recent WDR describing whereabouts of the MS of FIG. **2**F processing to a single data record every time a new one is to be inserted. This allows applications needing current whereabouts to simply access a current WDR, rather than interface to a plurality of WDRs at queue **22**. For example, there could be a new block **289** for updating the single WDR **1100** (just prior to block **290** such that incoming blocks to block **290** go to new block **289**, and new block **289** continues to block **290**).

With reference now to FIG. **2**E, depicted is a flowchart for describing a preferred embodiment of an MS whereabouts update event of an antenna in-range detected MS, for example a DLM **200**, when MS location awareness is monitored by the MS. FIG. **2**E describes relevant processing for MSs to maintain their own whereabouts. Processing begins at block **250** when the MS receives a signal from an antenna (or cell tower) deserving a response and continues to block **252** where the antenna or cell tower signal is authenticated by the MS as being a legitimate signal for processing. The signal can be received for processing by blocks **250** through **264** as the result of a continuous, or pulsed, broadcast or beaconing by the antenna, or cell tower (FIG. **13**C), or as part of usual communication protocol in progress with at least one MS

(FIG. **13**C usual data **1312** with embedded Communications Key **1314**), or as a response via antenna to a previous MS signal (FIG. **13**A). The signal is preferably authenticated by a data parsed signature deserving further processing. Block **252** continues to block **254** where the MS sends an outbound request for soliciting an immediate response from the antenna (or cell tower) service. The request by the MS is appropriately correlated (e.g. as described above) for a response, which additionally facilitates embodiments using TDOA measurements (time of communications between the MS and antenna, or cell tower) to determine how close is the MS in range. Block **254** waits for a response, or waits until a reasonable timeout, whichever occurs first. There are also multithreaded embodiments to breaking up FIG. **2**E where block **254** does not wait, but rather terminates FIG. **2**E processing and depends on another thread to correlate the response and then continue processing blocks **256** through **260** (like architecture **1900**).

Thereafter, if block **256** determines the request timed out, then processing terminates at block **264**. If block **256** determines the response was received, then processing continues to block **258**. Block **258** completes a WDR **1100** with appropriate response data received along with data set by the MS. See FIG. **11**A descriptions. Fields are set to the following upon exit from block **258**:

MS ID field **1100***a* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

DATE/TIME STAMP field **1100***b* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION field **1100***c* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

CONFIDENCE field **1100***d* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION TECHNOLOGY field **1100***e* is preferably set with: "Client Antenna Range" for an antenna detecting the MS, and is set to "Client Cell Range" for a cell tower detecting the MS. The originator indicator is set to DLM.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

SPEED field **1100***h* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

HEADING field **1100***i* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

ELEVATION field **1100***j* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

APPLICATION FIELDS field **1100***k* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

CORRELATION FIELD **1100***m* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

SENT DATE/TIME STAMP field **1100***n* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

RECEIVED DATE/TIME STAMP field **1100***p* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

The longer time it takes between sending a request and getting a response at block **254**, the less confidence there is for being located because the MS must be a larger distance from the antenna or cell tower. The less time it takes, the more confidence there is for being located because the MS must be a closer distance to the antenna or cell tower. Confidence values are analogously determined as described for FIG. **2**D. FIG. **2**D NTP embodiments also apply here. NTP can be used so no bidirectional communications is required for TDOA measurement. In this embodiment, the antenna (or cell tower) sets a NTP date/time stamp in the pulse, beacon, or protocol. Upon receipt, the MS instantly knows how long the packet took to be received by comparing the NTP date/time stamp in the packet and a MS NTP date/time stamp of when it was received (i.e. no request/response pair required). If location information is also present with the NTP date/time stamp in data received at block **252**, then block **252** can continue directly to block **258**.

An alternate MS embodiment determines its own (direction) heading and/or speed for WDR completion based on historical records maintained to the WDR queue **22** and/or LBX history **30**.

Block **258** continues to block **260** for preparing parameters for: WDRREF=a reference or pointer to the WDR; DELETEQ=FIG. **2**E location queue discard processing; and SUPER=FIG. **2**E supervisory notification processing. Thereafter, block **262** invokes the procedure (FIG. **2**F processing) to insert the WDR to queue **22**. After FIG. **2**F processing of block **262**, FIG. **2**E processing terminates at block **264**.

In alternative "coming within range" (same as "in range", "in-range", "within range") embodiments, a unique MS identifier, or MS group identifier, for authenticating an MS for locating the MS is not necessary. An antenna emitting signals (FIG. **13**C) will broadcast (in CK **1314** of data **1312**) not only its own location information (e.g. location field **1100***c*), but also an NTP indicated date/time stamp field **1100***b*, which the receiving MS (also having NTP for time synchronization) uses to perform a TDOA measurement upon receipt. This will enable a MS to determine at least how close (e.g. radius **1318** range, radius **1320** range, radius **1322** range, or radius **1316** range) it is located to the location of the antenna by listening for and receiving the broadcast (e.g. of FIG. **13**C). Similarly, in another embodiment, an NTP synchronized MS emits signals (FIG. **13**A) and an NTP synchronized data processing system associated with a receiving antenna can make a TDOA measurement upon signal receipt. In other embodiments, more than a single unidirectional signal may be used while still preventing the requirement to recognize the MS to locate it. For example, an antenna emitting signals (e.g. FIG. **13**C hotspot WiFi 802.x) will contain enough information for a MS to respond with correlation for being located, and visa-versa. In any case, there can be multi-directional exchanged signals for determining a TDOA measurement.

FIG. **3**A depicts a locating by triangulation illustration for discussing automatic location of a MS, for example DLM **200**. DLM **200** is located through triangulation, as is well known in the art. At least three base towers, for example, base tower **108***b*, base tower **108***d*, and base tower **108***f*, are used for locating the MS. A fourth base tower may be used if elevation (or altitude) was configured for use in locating DLM **200**. There are cases where only two base towers are necessary given routes of travel are limited and known, for example, in spread out roadways or limited configured locations. Base towers may also be antennas **108***b*, **108***d*, and **108***f* in similar triangulation embodiments.

FIG. **3**B depicts a flowchart for describing a preferred embodiment of the whereabouts update event of a triangulated MS, for example DLM **200**, when MS location awareness is monitored by some remote service. While FIG. **3**A location determination with TDOA and AOA is well known in the art, FIGS. **3**B and **3**C include relevant processing for MSs to maintain their own whereabouts. Processing begins at block **310** and continues to block **312** where base stations able to communicate to any degree with a MS continue reporting to their controller the MS signal strength with an MS identifier (i.e. a unique handle) and Time Difference of Arrival (TDOA) information, Angle of Arrival (AOA) information, or heterogeneously both TDOA and AOA (i.e. MPT), depending on the embodiment. The MS can pick signals from base stations. In some embodiments, the MS monitors a paging channel, called a forward channel. There can be multiple forward channels. A forward channel is the transmission frequency from the base tower to the MS. Either the MS provides broadcast heartbeats (FIG. **13**A) for base stations, or the base stations provide heartbeats (FIG. **13**C) for a response from the MS, or usual MS use protocol signals are detected and used (incorporating CK **1304** in usual data **1302** by MS, or CK **1314** in "usual data" **1312** by service). Usual data is the usual communications traffic data in carrying out other character **32** processing. Communication from the MS to the base tower is on what is called the reverse channel. Forward channels and reverse channel are used to perform call setup for a created session channel.

TDOA is calculated from the time it takes for a communication to occur from the MS back to the MS via the base tower, or alternatively, from a base tower back to that base tower via the MS. NTP may also be used for time calculations in a unidirectional broadcast from a base tower (FIG. **13**C) to the MS, or from the MS (FIG. **13**A) to a base tower (as described above). AOA is performed through calculations of the angle by which a signal from the MS encounters the antenna. Triangle geometry is then used to calculate a location. The AOA antenna is typically of a phased array type.

See "Missing Part Triangulation (MPT)" section below with discussions for FIGS. **11**A through **11**E for details on heterogeneously locating the MS using both TDOA and AOA (i.e. Missing Part Triangulation (MPT)). Just as high school taught geometry for solving missing parts of a triangle, so to does MPT triangulate an MS location. Think of the length of a side of a triangle as a TDOA measurement—i.e. length of time, translatable to a distance. Think of the AOA of a signal to an antenna as one of the angles of a triangle vertice. Solving with MPT analogously uses geometric and trigonometric formulas to solve the triangulation, albeit at fast processing speeds.

Thereafter, if the MS is determined to be legitimate and deserving of processing (similar to above), then block **314** continues to block **316**. If block **314** determines the MS is not participating with the service, in which case block **312** did little to process it, then processing continues back to block **312** to continue working on behalf of legitimate participating MSs. The controller at block **316** may communicate with other controllers when base stations in other cellular clusters are picking up a signal, for example, when the MS roams. In any case, at block **316**, the controller(s) determines the strongest signal base stations needed for locating the MS, at block **316**. The strongest signals that can accomplish whereabouts information of the MS are used. Thereafter, block **318** accesses base station location information for base stations determined at block **316**. The base station provides stationary references used to (relatively) determine the location of the MS. Then, block **320** uses the TDOA, or AOA, or MPT (i.e. heterogeneously both AOA and TDOA) information together with known base station locations to calculate the MS location.

Thereafter, block **322** accesses historical MS location information, and block **324** performs housekeeping by pruning location history data for the MS by time, number of entries, or other criteria. Block **326** then determines a heading (direction) of the MS based on previous location information. Block **326** may perform Artificial Intelligence (AI) to determine where the MS may be going by consulting many or all of the location history data. Thereafter, block **328** completes a service side WDR **1100**, block **330** appends the WDR information to location history data and notifies a supervisory service if there is one outside of the service processing of FIG. **3**B. Processing continues to block **332** where the service communicates the WDR to the located MS.

Thereafter, the MS completes its own WDR at block **334** for adding to WDR queue **22** to know its own whereabouts whenever possible, and block **336** prepares parameters for invoking WDR insertion processing at block **338**. Parameters are set for: WDRREF=a reference or pointer to the MS WDR; DELETEQ=FIG. **3**B location queue discard processing; and SUPER=FIG. **3**B supervisory notification processing (e.g. no supervisory notification processing because it was already handled at block **330**, or by being in context of the FIG. **3**B service processing). At block **338**, the MS invokes FIG. **2**F processing already described. After block **338**, processing continues back to block **312**. Of course, block **332** continues directly to block **312** at the service(s) since there is no need to wait for MS(s) processing in blocks **334** through **338**. FIG. **3**B processing is continuous for every MS in the wireless network 7 days a week, 24 hours a day.

See FIG. **11**A descriptions. Fields are set to the following upon exit from block **334**:

MS ID field **1100***a* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

DATE/TIME STAMP field **1100***b* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION field **1100***c* is preferably set with: The triangulated location of the MS as communicated by the service.

CONFIDENCE field **1100***d* is preferably set with: Confidence of triangulation determined by the service which is passed to the MS at block **332**. The confidence value may be set with the same value (e.g. 85) regardless of how the MS was triangulated. In other embodiments, field **1100***d* will be determined (completely, or adjusting the value of 85) by the service for TDOA measurements used, AOA measurements, signal strengths, wave spectrum involved, and/or the abundance of particular MS signals available for processing by blocks **312** through **320**. Higher confidences are assigned for smaller TDOA measurements (shorter distances), strong signal strengths, and numerous additional data points beyond what is necessary to locate the MS. Lower confidences are assigned for larger TDOA measurements, weak signal strengths, and minimal data points necessary to locate the MS. A reasonable confidence can be assigned using this information as guidelines where 1 is the lowest confidence and 100 is the highest confidence.

LOCATION TECHNOLOGY field **1100***e* is preferably set with: "Server Cell TDOA", "Server Cell AOA", "Server Cell MPT", "Server Antenna TDOA", "Server Antenna AOA", or "Server Antenna MPT", depending on how the MS was located and what flavor of service was used. The originator indicator is set to DLM.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: null (not set) for indicating that all triangulation data was factored into determining confidence, and none is relevant for a single TDOA or AOA measurement in subsequent processing (i.e. service did all the work).

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

SPEED field **1100***h* is preferably set with: Service WDR information at block **332**, wherein the service used historical information and artificial intelligence interrogation of MS travels to determine, if available.

HEADING field **1100***i* is preferably set with: Service WDR information at block **332**, wherein the service used historical information and artificial intelligence interrogation of MS travels to determine, if available.

ELEVATION field **1100*j*** is preferably set with: Elevation/altitude, if available.

APPLICATION FIELDS field **1100*k*** is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

CORRELATION FIELD **1100*m*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

SENT DATE/TIME STAMP field **1100*n*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

RECEIVED DATE/TIME STAMP field **1100*p*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

FIG. **3**C depicts a flowchart for describing a preferred embodiment of the whereabouts update event of a triangulated MS, for example a DLM **200**, when MS location awareness is monitored by the MS. Communications between the base stations and MS is similar to FIG. **3**B processing except the MS receives information (FIG. **13**C) for performing calculations and related processing. Processing begins at block **350** and continues to block **352** where the MS continues receiving (FIG. **13**C) pulse reporting from base stations (or antennas). AOA, TDOA, and MPT (See "Missing Part Triangulation (MPT)" section below with discussions for FIGS. **11**A through **11**E for details on heterogeneously locating the MS using both TDOA and AOA) can be used to locate the MS, so there are many possible signal types received at block **352**. Then, block **354** determines the strongest signals which can accomplish a completed WDR, or at least a location, of the MS. Thereafter, block **356** parses base station location information from the pulse messages that are received by the MS. Block **358** communicates with base stations to perform TDOA and/or AOA measurements and calculations. The time it takes for a communication to occur from the MS back to the MS for TDOA, or alternatively, from a base tower back to that base tower can be used. NTP may also be used, as described above, so that base towers (or antennas) broadcast signals (FIG. **13**C) picked up by the MS which already contain the base tower locations and NTP date/time stamps for TDOA calculations. Block **358** uses the TDOA and/or AOA information with the known base station information to determine the MS location. While AOA information from the base stations (or antennas) is used by the MS, various MS embodiments can use AOA information detected at an MS antenna provided the heading, yaw, pitch, and roll is known at the MS during the same time as signal reception by the MS. A 3-axis accelerometer (e.g. in iPhone) may also provide yaw, pitch and roll means for proper AOA calculation.

Thereafter, block **360** accesses historical MS location information (e.g. WDR queue **22** and/or LBX history **30**) to prevent redundant information kept at the MS, and block **362** performs housekeeping by pruning the LBX history **30** for the MS by time, number of entries, or other criteria. Block **364** then determines a heading (direction) of the MS based on previous location information (unless already known from block **358** for AOA determination). Block **364** may perform Artificial Intelligence (AI) to determine where the MS may be going by consulting queue **22** and/or history **30**. Thereafter, block **366** completes a WDR **1100**, and block **368** prepares parameters for FIG. **2**F processing: WDRREF=a reference or pointer to the MS WDR; DELETEQ=FIG. **3**C location queue discard processing; and SUPER=FIG. **3**B supervisory notification processing. Block **368** continues to block **370** for invoking FIG. **2**F processing already described above. After block **370**, processing continues back to block **352**. FIG. **3**C processing is continuous for the MS as long as the MS is enabled. In various multithreaded embodiments, many threads at the MS work together for high speed processing at blocks **352** through **358** for concurrently communicating to many stationary references.

See FIG. **11**A descriptions. Fields are set to the following upon exit from block **366**:

MS ID field **1100*a*** is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

DATE/TIME STAMP field **1100*b*** is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION field **1100*c*** is preferably set with: The triangulated location of the MS as determined by the MS.

CONFIDENCE field **1100*d*** is preferably set with: The confidence of triangulation as determined by the MS. Confidence may be set with the same value (e.g. 80 since MS may be moving during triangulation) regardless of how the MS was triangulated. In other embodiments, field **1100*d*** will be determined (completely, or adjusting the value of 80) by the MS for TDOA measurements used, AOA measurements, signal strengths, wave spectrum involved, and/or the abundance of particular service signals available for processing. Higher confidences are assigned for smaller TDOA measurements (shorter distances), strong signal strengths, and numerous additional data points beyond what is necessary to locate the MS. Lower confidences are assigned for larger TDOA measurements, weak signal strengths, and minimal data points necessary to locate the MS. A reasonable confidence can be assigned using this information as guidelines where 1 is the lowest confidence and 100 is the highest confidence.

LOCATION TECHNOLOGY field **1100*e*** is preferably set with: "Client Cell TDOA", "Client Cell AOA", "Client Cell MPT", "Client Antenna TDOA", "Client Antenna AOA", or "Client Antenna MPT", depending on how the MS located itself. The originator indicator is set to DLM.

LOCATION REFERENCE INFO field **1100*f*** is preferably set with: Data associated with selected best stationary reference(s) used by the MS: the selection location/whereabouts, TDOA measurement to it, and wave spectrum (and/or particular communications interface **70**) used, if reasonable. The TDOA measurement may be converted to a distance using wave spectrum information. Also, preferably set herein is data associated with a selected best stationary reference used by the MS (may be same or different than for TDOA measurement): the selection location, AOA measurement to it, and heading, yaw, pitch, and roll values (or accelerometer readings), if reasonable. Values that may be populated here should have already been factored into the confidence value. There may be one or more stationary reference whereabouts with useful measurements maintained here for FIG. **26**B processing of block **2652**.

COMMUNICATIONS REFERENCE INFO field **1100*g*** is preferably set with: Parameters referencing MS internals, if desired.

SPEED field **1100*h*** is preferably set with: Speed determined by the MS using historical information (queue **22** and/or history **30**) and artificial intelligence interrogation of MS travels to determine, if reasonable.

HEADING field **1100*i*** is preferably set with: Heading determined by the MS using historical information (queue **22** and/or history **30**) and artificial intelligence interrogation of MS travels to determine, if reasonable.

ELEVATION field **1100*j*** is preferably set with: Elevation/altitude, if available.

APPLICATION FIELDS field **1100*k*** is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

CORRELATION FIELD **1100*m*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

SENT DATE/TIME STAMP field **1100*n*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

RECEIVED DATE/TIME STAMP field **1100*p*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

In alternative triangulation embodiments, a unique MS identifier, or MS group identifier, for authenticating an MS for locating the MS is not necessary. An antenna emitting signals (FIG. **13**C) will broadcast (CK **1314** of data **1312**) not only its own location information, but also an NTP date/time stamp, which the receiving MS (also having NTP for time synchronization) uses to perform TDOA measurements upon receipt. This will enable a MS to determine how close (e.g. radius **1318** range, radius **1320** range, radius **1322** range, or radius **1316** range) it is located to the location of the antenna by listening for and receiving the broadcast (e.g. of FIG. **13**C). Similarly, in another embodiment, an NTP synchronized MS emits signals (FIG. **13**A) and an NTP synchronized data processing system associated with a receiving antenna can determine a TDOA measurement upon signal receipt. In other embodiments, more than a single unidirectional signal may be used while still preventing the requirement to recognize the MS to locate it. For example, an antenna emitting signals will contain enough information for a MS to respond with correlation for being located. Alternatively, an MS emitting signals will contain enough information for a service to respond with correlation for being located. In any case, there can be multi-directional exchanged signals for determining TDOA. Similarly, a service side data processing system can interact with a MS for AOA information without requiring a known identifier of the MS (use request/response correlation).

FIG. **4**A depicts a locating by GPS triangulation illustration for discussing automatic location of a MS, for example a DLM **200**. A MS, for example DLM **200**, is located through GPS triangulation as is well known in the art. At least three satellites, for example, satellite **134**, satellite **136**, and satellite **138**, are necessary for locating the MS. A fourth satellite would be used if elevation, or altitude, was configured for use by the present disclosure. Ground based stationary references can further enhance whereabouts determination.

FIG. **4**B depicts a flowchart for describing a preferred embodiment of the whereabouts update event of a GPS triangulated MS, for example a DLM **200**. Repeated continuous GPS location processing begins at block **410** and continues to block **412** where the MS initializes to the GPS interface, then to block **414** for performing the conventional locating of the GPS enabled MS, and then to block **416** for calculating location information. In some embodiments, block **412** may only be necessary a first time prior to repeated invocations of FIG. **4**B processing. Block **414** may be an implicit wait for pulses from satellites, or an event driven mechanism when GPS satellite pulses are received for synchronized collection, or a multithreaded implementation concurrently listening for, and processing collaboratively, the signals. Block **414** and block **416** processing is well known in the art. Thereafter, the MS completes a WDR **1100** at block **418**, block **420** prepares parameters for FIG. **2**F invocation, and block **422** invokes, with the WDR, the FIG. **2**F processing (described above). Processing then terminates at block **424**. Parameters prepared at block **420** are: WDRREF=a reference or pointer to the WDR; DELETEQ=FIG. **4**B location queue discard processing; and SUPER=FIG. **4**B supervisory notification processing. GPS location processing is preferably continuous for the MS as long as the MS is enabled.

See FIG. **11**A descriptions. Fields are set to the following upon exit from block **418**:

MS ID field **1100*a*** is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

DATE/TIME STAMP field **1100*b*** is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION field **1100*c*** is preferably set with: The GPS location of the MS.

CONFIDENCE field **1100*d*** is preferably set with: Confidence of GPS variety (usually high) which may be set with the same value (e.g. 95 for DGPS, 93 for AGPS, and 90 for GPS). In other embodiments, field **1100*d*** will be determined (completely, or amending the defaulted value) by the MS for timing measurements, signal strengths, and/or the abundance of particular signals available for processing, similarly to as described above. An MS may not be aware of the variety of GPS, in which case straight GPS is assumed.

LOCATION TECHNOLOGY field **1100*e*** is preferably set with: "GPS", "A-GPS", or "D-GPS", depending on (if known) flavor of GPS. The originator indicator is set to DLM.

LOCATION REFERENCE INFO field **1100*f*** is preferably set with: null (not set) for indicating that data was factored into determining confidence, and none is relevant for a single TDOA or AOA measurement in subsequent processing.

COMMUNICATIONS REFERENCE INFO field **1100*g*** is preferably set with: Parameters referencing MS internals, if desired.

SPEED field **1100*h*** is preferably set with: Speed determined by the MS using a suitable GPS interface, or historical information (queue **22** and/or history **30**) and artificial intelligence interrogation of MS travels to determine, if reasonable.

HEADING field **1100*i*** is preferably set with: Heading determined by the MS using a suitable GPS interface, or historical information (queue **22** and/or history **30**) and artificial intelligence interrogation of MS travels to determine, if reasonable.

ELEVATION field **1100*j*** is preferably set with: Elevation/altitude, if available.

APPLICATION FIELDS field **1100*k*** is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

CORRELATION FIELD **1100*m*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

SENT DATE/TIME STAMP field **1100*n*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

RECEIVED DATE/TIME STAMP field **1100*p*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

FIG. **5**A depicts a locating by stationary antenna triangulation illustration for discussing automatic location of a MS, for example DLM **200**. There may be communication/transmission issues when an MS is taken indoors. Shown is a top view of an indoor floor plan **502**. Antenna stations **504** (shown generally as **504**) are strategically placed over the area so that an MS can be located. Triangulation techniques again apply. At least three antenna stations, for example, station **504*f***, station **504*h***, and station **504*i*** are used to locate the MS, for example DLM **200**. In floor plan embodiments where aisles delimit travel, only two antenna stations may be necessary, for example at either end of the particular aisle. While most stations **504** may receive signals from the MS, only the strongest stations are used. FIG. **5**A and associated discussions can also be used for an outside triangulation embodiment using a similar strategic antenna placement scheme. Processing described for FIGS. **3**A to **3**C can also be used for an indoor embodiment as described by FIG. **5**A.

FIG. **5**B depicts a flowchart for describing a preferred embodiment of the whereabouts update event of a stationary antenna triangulated MS, for example a DLM **200**. In one embodiment, indoor location technology of Pinpoint corporation (Pinpoint is a trademark of Pinpoint Corporation) is utilized to locate any MS that moves about the indoor location. The Pinpoint corporation methodology begins at block **510** and continues to block **512**. A cell controller drives antenna stations to emit a broadcast signal from every station. Any MS within range (i.e. indoors) will phase modulate its unique identifier onto a return signal it transmits, at block **514**. Stations at block **516** receive the transmission and strength of signal. The cell controller that drives stations sorts out and selects the strongest (e.g. 3) signals. The cell controller, at block **518**, also extracts the unique MS identifier from the return signal, and TDOA is used to calculate distances from the stations receiving the strongest signals from the MS at block **520**. Alternative embodiments can use AOA or MPT to determine locations. The locations of the controller selected stations are registered in an overlay map in an appropriate coordinate system, landmark system, or grid of cells. Block **522** locates the MS using the overlay map, locations of the (e.g. 3) selected stations, and the calculated distances triangulated from the selected stations, using TDOA, AOA, or MPT in various embodiments. Thereafter, block **524** calculates location information of the MS. Processing continues with repeated broadcast at block **512** and subsequent processing for every MS within range.

Thereafter, block **526** accesses historical MS location information, performs housekeeping by pruning location history data for the MS by time, number of entries, or other criteria, and determines a heading (direction) of the MS based on previous location information. Block **526** may perform Artificial Intelligence (AI) to determine where the MS may be going by consulting many or all of the location history data. Thereafter, block **528** completes a service side WDR **1100**, block **530** appends the WDR information to location history data and notifies a supervisory service if there is one outside of the service processing of FIG. **5**B. Processing continues to block **532** where the service communicates the WDR to the located MS.

Thereafter, the MS completes the WDR at block **534** for adding to WDR queue **22**. Thereafter, block **536** prepares parameters passed to FIG. **2**F processing for: WDRREF=a reference or pointer to the MS WDR; DELETEQ=FIG. **5**B location queue discard processing; and SUPER=FIG. **5**B supervisory notification processing (e.g. no supervisory notification processing because it was already handled at block **530**, or by being in context of the FIG. **5**B service processing). Block **536** continues to block **538** where the MS invokes FIG. **2**F processing already described above. After block **538**, processing continues back to block **514**. Of course, block **532** continues directly to block **514** at the service(s) since there is no need to wait for MS(s) processing in blocks **534** through **538**. FIG. **5**B processing is continuous for every MS in the wireless network 7 days a week, 24 hours a day.

See FIG. **11**A descriptions. Fields are set to the following upon exit from block **534**:

MS ID field **1100*a*** is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

DATE/TIME STAMP field **1100*b*** is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION field **1100*c*** is preferably set with: The triangulated location of the MS as communicated by the service.

CONFIDENCE field **1100*d*** is preferably set with: Confidence of triangulation determined by the service which is passed to the MS at block **532**. The confidence value may be set with the same value (e.g. 95 (normally high for triangulation using densely positioned antennas)) regardless of how the MS was triangulated. In other embodiments, field **1100*d*** will be determined (completely, or adjusting the value of 95) by the service for TDOA measurements used, AOA measurements, signal strengths, wave spectrum involved, and/or the abundance of particular MS signals available for processing. Higher confidences are assigned for smaller TDOA measurements (shorter distances), strong signal strengths, and numerous additional data points beyond what is necessary to locate the MS. Lower confidences are assigned for larger TDOA measurements, weak signal strengths, and minimal data points necessary to locate the MS. A reasonable confidence can be assigned using this information as guidelines where 1 is the lowest confidence and 100 is the highest confidence.

LOCATION TECHNOLOGY field **1100*e*** is preferably set with: “Server Antenna TDOA”, “Server Antenna AOA”, or “Server Antenna MPT”, depending on how the MS was located and what flavor of service was used. The originator indicator is set to DLM.

LOCATION REFERENCE INFO field **1100*f*** is preferably set with: null (not set) for indicating that all triangulation data was factored into determining confidence, and none is relevant for a single TDOA or AOA measurement in subsequent processing (i.e. service did all the work).

COMMUNICATIONS REFERENCE INFO field **1100*g*** is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

SPEED field **1100*h*** is preferably set with: Service WDR information at block **532**, wherein the service used historical information and artificial intelligence interrogation of MS travels to determine, if available.

HEADING field **1100*i*** is preferably set with: Service WDR information at block **532**, wherein the service used historical information and artificial intelligence interrogation of MS travels to determine, if available.

ELEVATION field **1100*j*** is preferably set with: Elevation/altitude, if available.

APPLICATION FIELDS field **1100*k*** is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

CORRELATION FIELD **1100*m*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

SENT DATE/TIME STAMP field **1100*n*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

RECEIVED DATE/TIME STAMP field **1100*p*** is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

FIG. **6**A depicts a flowchart for describing a preferred embodiment of a service whereabouts update event of a physically, or logically, connected MS, for example a DLM **200**. A MS may be newly located and physically, or logically, connected, whereby communications between the MS and service is over a physical/logical connection. Physical connections may occur by connecting a conduit for communications to the MS, or from the MS to a connection point. Conduits include ethernet cables, optical fiber, firewire, USB, or any other means for conduit for communications through a physical medium. Conduits also include wireless mediums (air) for transporting communications, such as when an MS comes into physical wireless range eligible for sending and receiving communications. Logical connections may occur, after a physical connection already exists, for example through a successful communication, or authenticated, bind between a MS and other MS, or MS and service. Logical connections also include the result of: successfully logging into an application, successfully authenticated for access to some resource, successfully identified by an application, or any other logical status upon a MS being certified, registered, signed in, authenticated, bound, recognized, affirmed, or the like.

Relevant processing begins at block **602** and continues to block **604** where an MS device is physically/logically connected to a network. Thereafter, the MS accesses a service at block **606**. Then, at block **608**, the service accesses historical MS location history, and block **610** performs housekeeping by pruning the location history data maintained for the MS by time, number of entries, or other criteria. Block **610** may perform Artificial Intelligence (AI) to determine where the MS may be going (e.g. using heading based on previous locations) by consulting much or all of the location history data. Thereafter, service processing at block **612** completes a service side WDR **1100**, then the service appends WDR information to location history data at block **614**, and may notify a supervisory service if there is one outside of the service processing of FIG. **6**A. Processing continues to block **616** where the service communicates WDR information to the newly physically/logically connected MS. There are many embodiments for determining a newly connected MS location using a physical or logical address, for example consulting a database which maps locations to network addresses (e.g. location to logical ip address; location to physical wall jack/port; etc). Then, at block **618** the MS completes its own WDR using some information from block **616**, FIG. **2**F parameters are prepared at block **620**, block **622** invokes FIG. **2**F processing already described above, and processing terminates at block **624**. Parameters are set at block **620** for: WDRREF=a reference or pointer to the MS WDR; DELETEQ=FIG. **6**A location queue discard processing; and SUPER=FIG. **6**A supervisory notification processing (e.g. no supervisory notification processing because it was already handled at block **614**, or by being in context of the FIG. **6**A service processing). Of course, block **616** continues directly to block **624** at the service(s) since there is no need to wait for MS processing in blocks **618** through **622**. FIG. **6**A processing is available at any appropriate time in accordance with the underlying service.

See FIG. **11**A descriptions. Fields are set to the following upon exit from block **618**:

MS ID field **1100***a* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

DATE/TIME STAMP field **1100***b* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION field **1100***c* is preferably set with: The location of the MS as communicated by the service.

CONFIDENCE field **1100***d* is preferably set with: Confidence (determined by the service) according to how the MS was connected, or may be set with the same value (e.g. 100 for physical connect, 77 for logical connect (e.g. short range wireless)) regardless of how the MS was located. In other embodiments, field **1100***d* will be determined by the service for anticipated physical conduit range, wireless logical connect range, etc. The resulting confidence value can be adjusted based on other parameters analogously to as described above.

LOCATION TECHNOLOGY field **1100***e* is preferably set with "Service Physical Connect" or "Service Logical Connect", depending on how the MS connected. The originator indicator is set to DLM.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: null (not set), but if a TDOA measurement can be made (e.g. short range logical connect, and using methodologies described above), then a TDOA measurement, a communications signal strength, if available; and wave spectrum (and/or particular communications interface **70**) used, if available. The TDOA measurement may be converted to a distance using wave spectrum information. Possible values populated here should have already been factored into the confidence value.

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

SPEED field **1100***h* is preferably set with: null (not set), but can be set with speed required to arrive to the current location from a previously known location, assuming same time scale is used.

HEADING field **1100***i* is preferably set with: null (not set), but can be set to heading determined when arriving to the current location from a previously known location.

ELEVATION field **1100***j* is preferably set with: Elevation/altitude (e.g. of physical connection, or place of logical connection detection), if available.

APPLICATION FIELDS field **1100***k* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

CORRELATION FIELD **1100***m* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

SENT DATE/TIME STAMP field **1100***n* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

RECEIVED DATE/TIME STAMP field **1100***p* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

FIG. **6**B depicts a flowchart for describing a preferred embodiment of a MS whereabouts update event of a physically, or logically, connected MS, for example a DLM **200**. A MS may be newly located and physically/logically connected, whereby communications between the MS and service is over a physical/logical connection as described in FIG. **6**A above. Relevant processing begins at block **640** and continues to block **642** where an MS device is physically/logically connected. Thereafter, at block **644** the MS accesses the connectivity service and waits for an acknowledgement indicating a successful connection. Upon acknowledgement receipt, processing continues to block **646** where the MS requests WDR information via the connectivity service and waits for the data (i.e. connectivity service may be different than the location service, or may be one in the same). As part of connectivity, location service pointer(s) (e.g. ip address for http://112.34.323.18 referencing or a Domain Name Service (DNS) name like http://www.servicename.com) are provided with the connectivity acknowledgement from the connectivity service at block **644**, so the MS knows how to proceed at block **646** for retrieving location information. There are various embodiments for the location service determining a MS location as described above for FIG. **6**A. In an alternative embodiment, the MS already knows how to locate itself wherein block **644** continues directly to block **648** (no block **646**) because the MS maintains information for determining its own whereabouts using the physical or logical address received in the acknowledgement at block **644**. Similar mapping of a network address to the MS location can be in MS data, for example data **36**, data **8**, or data **20**. At block **648**, the MS completes its WDR **1100**. Thereafter, block **650** prepares FIG. **2**F parameters, block **652** invokes FIG. **2**F processing already described above, and processing terminates at block **654**. Parameters set at block **650** are: WDRREF=a reference or pointer to the MS WDR; DELETEQ=FIG. **6**B location queue discard processing; and SUPER=FIG. **6**B supervisory notification processing. FIG. **6**B processing is available at any appropriate time to the MS.

See FIG. **11**A descriptions. Fields are set to the following upon exit from block **648**:

MS ID field **1100***a* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

DATE/TIME STAMP field **1100***b* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION field **1100***c* is preferably set with: The location determined for the MS.

CONFIDENCE field **1100***d* is preferably set with: Confidence (determined by the service) according to how the MS was connected, or may be set with the same value (e.g. 100 for physical connect, 77 for logical connect (e.g. short range wireless)) regardless of how the MS was located. In other embodiments, field **1100***d* will be determined by the service for anticipated physical conduit range, wireless logical connect range, etc. The resulting confidence value can be adjusted based on other parameters analogously to as described above.

LOCATION TECHNOLOGY field **1100***e* is preferably set with "Client Physical Connect" or "Client Logical Connect", depending on how the MS connected. The originator indicator is set to DLM.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: null (not set), but if a TDOA measurement can be made (e.g. short range logical connect, and using methodologies described above), then a TDOA measurement, a communications signal strength, if available; and wave spectrum (and/or particular communications interface **70**) used, if available. The TDOA measurement may be converted to a distance using wave spectrum information. Possible values populated here should have already been factored into the confidence value.

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

SPEED field **1100***h* is preferably set with: null (not set), but can be set with speed required to arrive to the current location from a previously known location using, assuming same time scale is used.

HEADING field **1100***i* is preferably set with: null (not set), but can be set to heading determined when arriving to the current location from a previously known location.

ELEVATION field **1100***j* is preferably set with: Elevation/altitude (e.g. of physical connection, or place of logical connection detection), if available.

APPLICATION FIELDS field **1100***k* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

CORRELATION FIELD **1100***m* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

SENT DATE/TIME STAMP field **1100***n* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

RECEIVED DATE/TIME STAMP field **1100***p* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

FIGS. **7**A, **7**B and **7**C depict a locating by image sensory illustration for discussing automatic location of a MS, for example a DLM **200**. With reference now to FIG. **7**A, an image capture device **702** is positioned for monitoring MSs that come into the field of view **704** of device **702**. Device **702** may be a camcorder, video camera, image camera that takes at least one snapshot, timely snapshots, or motion/presence detection snapshots, or any other device capable of producing at least a snapshot image at some point in time containing objects in the field of view **704**. In one preferred embodiment, DLM **200** is sensed within the vicinity of device **702**, perhaps by antenna (or cell tower) **701**, prior to being photographed by device **702**. In another embodiment, DLM **200** is sensed by movement within the vicinity of device **702** with well know motion detection means. In yet another embodiment, device **702** periodically or continually records. Device **702** is connected to a locating service **700** for processing as described by FIG. **7**D. Locating service **700** has means for communicating wirelessly to DLM **200**, for example through a connected antenna (or cell tower) **701**. FIG. **7**A illustrates that device **702** participates in pattern recognition for identifying the location of a MS. The MS can have on its exterior a string of characters, serial number, barcode, license plate, graphic symbol(s), textual symbols, combinations thereof, or any other visually perceptible, or graphical, identification **708** that can be recognized optically, or in a photograph. Device **702** is to have graphical/pixel resolution capability matching the requirements for identifying a MS with the sought graphical identification. Graphical identification **708** can be formed on the perceptible exterior of DLM **200**, or can be formed as part of a housing/apparatus **706** which hosts DLM **200**. Graphical identification **708** can be automatically read from an image using well known barcode reader technology, an Optical Character Recognition (OCR) process, a license tag scanner, general pattern recognition software, or the like. Housing **706** is generally shown for representing an automobile (license plate recognition, for example used in prior art toll tag lanes), a shopping cart, a package, or any other hosting article of manufacture which has a DLM **200** as part of it. Upon recognition, DLM **200** is associated with the location of device **702**. Error in locating an MS will depend on the distance within the field of view **704** from device **702**. A distance may be estimated based on the anticipated size of identification **708**, relative its size determined within the field of view **704**.

With reference now to FIG. **7**B, image capture device **702** is positioned for monitoring MSs that come into the field of view **704** of device **702**. MSs are preferably distinguishable by appearance (e.g. color, shape, markings, labels, tags, etc), or as attached (e.g. recognized mount to host) or carried (e.g. recognized by its recognized user). Such techniques are well known to those skilled in the art. Device **702** is as described above with connectivity to locating service **700** and antenna (or cell tower) **701**. FIG. **7**B illustrates that device **702** uses known measurements within its field of view for determining how large, and where located, are objects that come into the field of view **704**. For example, a well placed and recognizable vertical line **710***a* and horizontal line **710***b*, which are preferably perpendicular to each other, have known lengths and positions. The objects which come into the field of view are measured based on the known lengths and positions of the lines **710***a* and **710***b* which may be landscape markings (e.g. parking lot lines) for additional purpose. Field of view **704** may contain many lines and/or objects of known dimensions strategically placed or recognized within the field of view **704** to facilitate image processing by service **700**. Building **714** may serve as a reference point having known dimension and position in measuring objects such as a person **716** or DLM **200**. A moving object such as a shopping cart **712** can have known dimensions, but not a specific position, to facilitate service **700** in locating an MS coming into the field of view **704**. Those skilled in the art recognize that known dimensions and/or locations of anticipated objects in field of view **704** have measurements facilitating discovering positions and measurements of new objects that may travel into the field of view **704**. Using FIG. **7**B techniques with FIG. **7**A techniques provides additional locating accuracy. A distance may be estimated based on the anticipated sizes of references in the field of view, relative size of the recognized MS.

With reference now to FIG. **7**C, image capture device **702** is positioned for monitoring MSs that come into the field of view **704** of device **702**. Device **702** is as described above with connectivity to locating service **700** and antenna (or cell tower) **701**. MSs are preferably distinguishable by appearance (e.g. color, shape, markings, labels, tags, etc), or as attached (e.g. recognized mount to host) or carried (e.g. recognized by its user), or as identified by FIG. **7**A and/or FIG. **7**B methodologies. FIG. **7**C illustrates that device **702** uses known locations within its field of view for determining how large, and where located, are objects that come into the field of view **704**. For example, building **714**, tree **720**, and traffic sign **722** have its locations known in field of view **704** by service **700**. Solving locations of objects that move into the field of view is accomplished with graphical triangulation measurements between known object reference locations (e.g. building **714**, tree **720**, and sign **722**) and the object to be located. Timely snapshots by device **702** provide an ongoing locating of an MS, for example DLM **200**. Line segment distances **724** (a, b, c) can be measured using references such as those of FIG. **7**B. Whereabouts are determined by providing known coordinates to anticipated objects such as building **714**, tree **720**, and sign **722**. Similarly, graphical AOA measurements (i.e. graphical angle measurements) and graphical MPT measurements can be used in relation to anticipated locations of objects within the field of view **704**. There may be many anticipated (known) object locations within field of view **704** to further facilitate locating an MS. Being nearby an object may also be enough to locate the MS by using the object's location for the location of the MS. Using FIG. **7**C techniques with FIG. **7**A and/or FIG. **7**B techniques provides additional locating accuracy.

The system and methodologies illustrated by FIGS. **7**A through **7**C are preferably used in optimal combination by locating service **700** to provide a best location of an MS. In some embodiments, MS whereabouts is determined as the location of a device **702** by simply being recognized by the device **702**. In other embodiments, multiple devices **702** can be strategically placed within a geographic area for being used in combination to a common locating service **700** for providing a most accurate whereabouts of an MS. Multiple field of views **704** from difference angles of different devices **702** enable more precise locating within three dimensional space, including precise elevations.

FIG. **7**D depicts a flowchart for describing a preferred embodiment of graphically locating a MS in accordance with locating service **700** described above, for example as illustrated by FIGS. **7**A through **7**C. Locating service **700** may be a single capable data processing system, or many connected data processing systems for enhanced parallel processing. Locating service **700** may be connected to services involved with any other locating technology described in this application for synergistic services as an MS is mobile. Locating service **700** begins at block **732** and continues to block **734** where the service **700** is initialized in preparation of MS whereabouts analysis. Block **734** initializes its table(s) of sought identifying criteria which can be pattern recognized. In one preferred embodiment, color/shade, shape, appearance and applicable sought information is initialized for each sought identifying criteria. Pattern recognition is well known in the art and initialization is specific for each technology discussed above for FIGS. **7**A through **7**C. For FIGS. **7**B and **7**C discussions, positions, measurements, and reference points of known landmarks are additionally accounted. Thereafter, block **736** gets the next snapshot from device(s) **702**. If there is none waiting to get, block **736** waits for one. If there is one queued up for processing, then block **736** continues to block **738**. FIG. **7**D is processing of a service, and is preferably multi-threaded. For example, blocks **736** through **754** can occur concurrently in many threads for processing a common queue of snapshots received from a device **702**, or many devices **702**. Each thread may process all sought criteria, or may specialize in a subset of sought criteria wherein if nothing is found, the thread can place the snapshot back on a queue for thread processing for another sought criteria after marking the queue entry as having been processed for one particular subset. So, threads may be specialized and work together in seeking all criteria, or may each work in parallel seeking the same criteria. In preferred embodiments, there is at least one queue of snapshots received by block(s) **736**. Block **736** continues to block **738** which attempts to detect an MS having sought criteria using pattern recognition techniques of FIGS. **7**A through **7**C, in particular, or in combination. In one example embodiment, as device **702** provides service **700** with at least one timely snapshot to block **736**, the snapshot graphic is scanned at block **738** for identifying characters/symbols/appearance of sought criteria. Block **738** continues with its search result to block **740**. If block **740** determines no MS was detected, then processing continues back to block **736**. If block **738** detected at least one MS (as determined at block **740**), then block **742** calculates WDR information for the MS(s) detected, block **744** notifies a supervisory service of MS whereabouts if applicable, block **746** communicates the WDR information to MS(s) detected (for example via antenna **701**), and processing continues to block **748**.

There may be a plurality of MSs in the field of view, so communications at block **746** targets each MS recognized. A MS should not rely on the service to have done its job correctly. At a MS, block **748** checks the MS ID communicated for validation. If block **748** determines the MS ID is incorrect, then processing continues back to block **736** (for the particular MS). If block **748** determines the MS ID is correct, then processing continues to block **750** where the particular MS completes its WDR **1100** received from service **700**. Thereafter, MS(s) prepare parameters at block **752**, invoke local FIG. **2**F processing already described above (at block **754**), and processing continues for service **700** back to block **736**. Of course, block **746** continues directly to block **736** at the service(s) since there is no need to wait for MS(s) processing in blocks **748** through **754**. Parameters set at block **752** are: WDRREF=a reference or pointer to the MS WDR; DELETEQ=FIG. **7**D location queue discard processing; and SUPER=FIG. **7**D supervisory notification (e.g. no supervisory notification processing because it was already handled at block **744**, or by being in context of the FIG. **7**D service processing). No snapshots from device **702** are to be missed at block **736**.

See FIG. **11**A descriptions. Fields are set to the following upon exit from block **750**:

MS ID field **1100***a* is preferably set with: Unique MS identifier of the MS, after validating at the MS that the service **700** has correctly identified it. This field is used to uniquely distinguish this MS WDRs on queue **22** from other originated WDRs. The service **700** may determine a MS ID from a database lookup using above appearance criteria. Field **1100***a* may also be determined using the transmission methods as described for FIGS. **2**A through **2**E, for example by way of antenna **701**. For example, when the MS comes within range of antenna **701**, FIG. **7**D processing commences. Another embodiment prevents recognizing more than one MS within the field of view **704** at any time (e.g. a single file entryway), in which case the service can solicit a "who are you" transmission to identify the MS and then send back its whereabouts (in which case the MS sets its own MS ID here).

DATE/TIME STAMP field **1100***b* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION field **1100***c* is preferably set with: The location determined for the MS by the service.

CONFIDENCE field **1100***d* is preferably set with: same value (e.g. 76) regardless of how the MS location was determined. In other embodiments, field **1100***d* will be determined by the number of distance measurements and/or the abundance of particular objects used in the field of view **704**. The resulting confidence value can be adjusted based on other graphical parameters involved, analogously to as described above.

LOCATION TECHNOLOGY field **1100***e* is preferably set with: "Server Graphic-Patterns" "Server Graphic-Distances", "Server Graphic Triangulate", or a combination field value depending on how the MS was located and what flavor of service was used. The originator indicator is set to DLM.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: null (not set) for indicating that all whereabouts determination data was factored into the confidence, and none is relevant for a single TDOA or AOA measurement in subsequent processing (i.e. service did all the work).

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

SPEED field **1100***h* is preferably set with: null (not set), but can be set with speed required to arrive to the current location from a previously known time at a location (e.g. using previous snapshots processed), assuming the same time scale is used.

HEADING field **1100***i* is preferably set with: null (not set), but can be set to heading determined when arriving to the current location from a previously known location (e.g. using previous snapshots processed).

ELEVATION field **1100***j* is preferably set with: Elevation/altitude, if available, if available.

APPLICATION FIELDS field **1100***k* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

CORRELATION FIELD **1100***m* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

SENT DATE/TIME STAMP field **1100***n* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

RECEIVED DATE/TIME STAMP field **1100***p* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

In an alternative embodiment, MS **2** may be equipped (e.g. as part of resources **38**) with its own device **702** and field of view **704** for graphically identifying recognizable environmental objects or places to determine its own whereabouts. In this embodiment, the MS would have access to anticipated objects, locations and dimensions much the same way described for FIGS. **7**A through **7**D, either locally maintained or verifiable with a connected service. Upon a successful recognition of an object, place, or other graphically perceptible image which can be mapped to a location, the MS would complete a WDR similarly to above. The MS may recognize addresses, buildings, landmarks, of other pictorial data. Thus, the MS may graphically determine its own location. The MS would then complete a WDR **1100** for FIG. **2**F processing exactly as described for FIG. **7**D with the exceptions of fields that follow:

MS ID field **1100***a* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION field **1100***c* is preferably set with: The location determined for the MS by the MS.

LOCATION TECHNOLOGY field **1100***e* is preferably set with: "Client Graphic-Patterns" "Client Graphic-Distances", "Client Graphic Triangulate", or a combination field value depending on how the MS located itself. The originator indicator is set to DLM.

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: null (not set).

FIG. **8**A heterogeneously depicts a locating by arbitrary wave spectrum illustration for discussing automatic location of a MS. In the case of acoustics or sound, prior art has shown that a noise emitting animal or object can be located by triangulating the sound received using TDOA by strategically placed microphones. It is known that by figuring out time delay between a few strategically spaced microphones, one can infer the location of the sound. In a preferred embodiment, an MS, for example DLM **200**, emits a pulsed or constant sound (preferably beyond the human hearing range) which can be sensed by microphones **802** though **806**. Data is superimposed on the sound wave spectrum with variations in pitch or tone, or data occurs in patterned breaks in sound transmission. Data may contain a unique identifier of the MS so service(s) attached to microphones **802** through **806** can communicate uniquely to an MS. In some embodiments, sound used by the MS is known to repel certain pests such as unwanted animals, rodents, or bugs in order to prevent the person carrying the MS from encountering such pests during travel, for example during outdoor hiking or mountain climbing. In submarine acoustics, AOA is a method to locate certain objects. The FIGS. **3**B and **3**C flowcharts occur analogously for sound signals received by microphones **802** through **806** which are connected to service processing of FIGS. **3**B and **3**C. The only difference is wave spectrum used.

It has been shown that light can be used to triangulate position or location information (e.g. U.S. Pat. No. 6,549,288 (Migdal et al) and U.S. Pat. No. 6,549,289 (Ellis)). Optical sensors **802** through **806** detect a light source of, or illumination of, an MS, for example DLM **200**. Data is superimposed on the light wave spectrum with specified frequency/wavelength and/or periodicity, or data occurs in patterned breaks in light transmission. Data may contain a unique identifier of the MS so service(s) attached to sensors **802** through **806** can communicate uniquely to an MS. Mirrors positioned at optical sensors **802** through **806** may be used to determine an AOA of light at the sensor, or alternatively TDOA of recognizable light spectrum is used to position an MS. The FIGS. **3**B and **3**C flowcharts occur analogously for light signals received by sensors **802** through **806** which are connected to service processing of FIGS. **3**B and **3**C. The only difference is wave spectrum used.

Heterogeneously speaking, FIG. **8**A illustrates having strategically placed sensors **802** through **806** for detecting a wave spectrum and using TDOA, AOA, or MPT. Those skilled in the art appreciate that a wave is analogously dealt with by FIGS. **3**B and **3**C regardless of the wave type, albeit with different sensor types **802** through **806** and different sensor interface to service(s) of FIGS. **3**B and **3**C. Wave signal spectrums for triangulation by analogous processing to FIGS. **3**B and **3**C include microwaves, infrared, visible light, ultraviolet light, X-rays, gamma rays, longwaves, magnetic spectrum, or any other invisible, visible, audible, or inaudible wave spectrum. Sensors **802** through **806** are appropriately matched according to the requirements. Alternatively, a MS may be sensing wave spectrums emitted by transmitters **802** through **806**.

Those skilled in the relevant arts appreciate that the point in all this discussion is all the wave forms provide methods for triangulating whereabouts information of an MS. Different types of wave forms that are available for an MS can be used solely, or in conjunction with each other, to determine MS whereabouts. MSs may be informed of their location using the identical wave spectrum used for whereabouts determination, or may use any other spectrum available for communicating WDR information back to the MS. Alternatively, the MS itself can determine WDR information relative applicable sensors/transmitters. In any case, a WDR **1100** is completed analogously to FIGS. **3**B and **3**C.

FIG. **8**B depicts a flowchart for describing a preferred embodiment of locating a MS through physically sensing a MS, for example a DLM **200**. Processing begins at block **810** upon contact with a candidate MS and continues to block **812** where initialization takes place. Initialization includes determining when, where, and how the contact was made. Then, block **814** takes the contact sample and sets it as input containing a unique identifier or handle of the MS which was sensed. There are various known embodiments of how the MS is sensed:

a) Touching sensors contact the MS (or host/housing having MS) to interpret physical characteristics of the MS in order to uniquely identify it (e.g. Braille, embossed/raised/depressed symbols or markings, shape, temperature, depressions, size, combinations thereof, etc);

b) Purchase is made with MS while in vicinity of device accepting purchase, and as part of that transaction, the MS is sensed as being at the same location as the device accepting purchase, for example using a cell phone to purchase a soft drink from a soft drink dispensing machine;

c) Barcode reader is used by person to scan the MS (or host/housing having MS), for example as part of shipping, receiving, or transporting;

d) The MS, or housing with MS, is sensed by its odor (or host/housing having MS), perhaps an odor indicating where it had been, where it should not be, or where it should be. Various odor detection techniques may be used;

e) Optical sensing wherein the MS is scanned with optical sensory means, for example to read a serial number; and/or

f) Any sensing means which can identify the MS through physical contact, or by nearby/close physical contact with some wave spectrum.

Block **814** continues to block **816** where a database is accessed for recognizing the MS identifier (handle) by mapping sensed information with an associated MS handle. If a match is found at block **818**, then block **822** determines WDR **1100** information using the location of where sensing took place. If block **818** determines no match was found, then data is saved at block **820** for an unrecognized entity such as is useful when an MS should have been recognized, but was not. In another embodiment, the MS handle is directly sensed so block **814** continues directly to block **818** (no block **816**). Block **820** continues to block **834** where processing terminates. Block **816** may not use the entire MS identifier for search, but some portion of it to make sure it is a supported MS for being located by sensing. The MS identifier is useful when communicating wirelessly the WDR information to the MS (at block **826**).

Referring now back to block **822**, processing continues to block **824** where a supervisory service may be updated with the MS whereabouts (if applicable), and block **826** communicates the WDR information to the MS. Any available communication method can be used for communicating the WDR information to the MS, as described above. Thereafter, the MS completes the WDR at block **828**, block **830** prepares FIG. **2**F parameters, and block **832** invokes FIG. **2**F processing already described above. Processing terminates thereafter at block **834**. Parameters set at block **830** are: WDRREF=a reference or pointer to the MS WDR; DELETEQ=FIG. **8**B location queue discard processing; and SUPER=FIG. **8**B supervisory notification (e.g. no supervisory notification processing because it was already handled at block **824**, or by being in context of the FIG. **8**B service processing). FIG. **8**B processing is available at any appropriate time for the MS. In an alternate embodiment, the MS senses its environment to determine whereabouts.

See FIG. **11**A descriptions. Fields are set to the following upon exit from block **828**:

MS ID field **1100***a* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

DATE/TIME STAMP field **1100***b* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION field **1100***c* is preferably set with: Location of the sensor sensing the MS.

CONFIDENCE field **1100***d* is preferably set with: Should be high confidence (e.g. 98) for indisputable contact sensing and is typically set with the same value.

LOCATION TECHNOLOGY field **1100***e* is preferably set with: "Contact", or a specific type of Contact. The originator indicator is set to DLM.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: null (not set).

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

SPEED field **1100***h* is preferably set with: null (not set), but can be set with speed required to arrive to the current location from a previously known time at a location, assuming the same time scale is used.

HEADING field **1100***i* is preferably set with: null (not set), but can be set to heading determined when arriving to the current location from a previously known location.

ELEVATION field **1100***j* is preferably set with: Elevation/altitude, if available.

APPLICATION FIELDS field **1100***k* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

CORRELATION FIELD **1100***m* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

SENT DATE/TIME STAMP field **1100***n* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

RECEIVED DATE/TIME STAMP field **1100***p* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

FIG. **8**C depicts a flowchart for describing a preferred embodiment of locating a MS, for example a DLM **200**, through a manually entered location of the MS. MS user interface processing begins at block **850** when a user starts the user interface from code **18** and continues to block **852**. Any of a variety of user interfaces, dependent on the type of MS, is used for manually entering the location of the MS. A user interfaces with the MS at block **852** until one of the monitored actions relevant to this disclosure are detected. Thereafter, if block **854** determines the user has selected to set his location manually, then processing continues to block **860**. If block **854** determines the user did not select to manually set his location, then block **856** determines if the user selected to force the MS to determine its location. If the user did select to force the MS to get its own location, then block **856** continues to block **862**. If the user did not select to force the MS to get its own location as determined by block **856**, then processing continues to block **858**. If block **858** determines the user wanted to exit the user interface, then block **880** terminates the interface and processing terminates at block **882**. If block **858** determines the user did not want to exit the user interface, then block **884** handles any user interface actions which caused exit from block **852** yet were not handled by any action processing relevant to this disclosure.

With reference back to block **860**, the user interfaces with the MS user interface to manually specify WDR information. The user can specify:

1) An address or any address subset such as a zip code;
2) Latitude, longitude, and elevation;
3) MAPSCO identifier;
4) FEMA map identifier;
5) USDA map identifier;
6) Direct data entry to a WDR **1100**; or
7) Any other method for user specified whereabouts of the MS.

The user can specify a relevant confidence value for the manually entered location, however, processing at block **860** preferably automatically defaults a confidence value for the data entered. For example, a complete address, validated at block **860**, will have a high confidence. A partial address such as city and state, or a zip code will have a low confidence value. The confidence value will reflect how large an area is candidate for where the MS is actually located. To prevent completely relying on the user at block **860** for accurate WDR information, validation embodiments may be deployed. Some examples:

Upon specification (e.g. FEMA), the MS will access connected service(s) to determine accuracy (FEMA conversion tables);

Upon specification (e.g. MAPSCO), the MS will access local resources to help validate the specification (e.g. MAPSCO conversion tables); and/or

Upon specification (e.g. address), the MS can access queue **22** and/or history **30** for evidence proving likelihood of accuracy. The MS may also access services, or local resources, for converting location information for proper comparisons.

In any case, a confidence field **1100***d* value can be automatically set based on the validation results, and the confidence may, or may not, be enabled for override by the user.

After WDR information is specified at block **860**, the MS completes the WDR at block **874**, block **876** prepares parameters for FIG. **2**F processing, and (at block **878**) the MS invokes FIG. **2**F processing already described above before returning back to block **852**. Parameters set at block **876** are: WDRREF=a reference or pointer to the MS WDR; DELETEQ=FIG. **8**C location queue discard processing; and SUPER=FIG. **8**C supervisory notification processing. Various embodiments permit override of the confidence floor value by the user, or by FIG. **8**C processing. Block **874** may convert the user specified information into a standardized more usable form in an LN-expanse (e.g. convert to latitude and longitude if possible, truncated precision for more area coverage). WDR **1100** fields (see FIG. **11**A) are set analogously in light of the many variations already described above.

With reference back to block **862**, if it is determined that the MS is equipped with capability (e.g. in range, or in readiness) to locate itself, then processing continues to block **864** where the MS locates itself using MS driven capability described by FIGS. **2**E, **3**C, **4**B, **6**B, and **8**A or MS driven alternative embodiments to FIGS. **2**D, **3**B, **5**B, **6**A, **7**D, **8**A, and **8**B, or any other MS capability for determining its own whereabouts with or without help from other data processing systems or services. Interfacing to locating capability preferably involves a timeout in case there is no, or slow, response, therefore block **864** continues to block **868** where it determined whether or not block **864** timed out prior to determining a location. If block **868** determines a timeout was encountered, then block **872** provides the user with an error to the user interface, and processing continues back to block **852**. Block **872** preferably requires use acknowledgement prior to continuing to block **852**.

If block **868** determines there was no timeout (i.e. whereabouts successfully determined), then block **870** interfaces to the locating interface to get WDR information, block **874** completes a WDR, and blocks **876** and **878** do as described above. If block **862** determines the MS cannot locate itself and needs help, then block **866** emits at least one broadcast request to any listening service which can provide the MS its location. Appropriate correlation is used for an anticipated response. Example services listening are service driven capability described by FIGS. **2**D, **3**B, **5**B, **6**A, **7**D, **8**A, and **8**B, or service side alternative embodiments of FIGS. **2**E, **3**C, **4**B, **6**B, and **8**A, or any other service capability for determining MS whereabouts with or without help from the MS or other data processing systems or services. Block **866** then continues to block **868**.

If block **868** determines a timeout was encountered from the service broadcast request, then block **872** provides the user with an error to the user interface, and processing continues back to block **852**. If block **868** determines there was no timeout (i.e. whereabouts successfully determined), then block **870** receives WDR information from the locating interface of the responding service, block **874** completes a WDR, and blocks **876** and **878** do as already described above.

See FIG. **11**A descriptions. Depending how the MS was located via processing started at block **856** to block **862**, a WDR is completed analogous to as described in Figures above. If the user manually specified whereabouts at block **860**, fields are set to the following upon exit from block **874**:

MS ID field **1100***a* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

DATE/TIME STAMP field **1100***b* is preferably set with: Same as was described for FIG. **2**D (block **236**) above.

LOCATION field **1100***c* is preferably set with: Location entered by the user, or converted from entry by the user; preferably validated.

CONFIDENCE field **1100***d* is preferably set with: User specified confidence value, or a system assigned value per a validated manual specification. Confidence should reflect confidence of location precision (e.g. validated full address high; city and zip code low, etc). Manually specified confidences are preferably lower than other location technologies since users may abuse or set incorrectly, unless validated. Specifying lower confidence values than technologies above, for completely manual WDR specifications (i.e. no validation), ensures that manual specifications are only used by the MS in absence of other technologies. There are many validation embodiments that can be deployed (as described above) for a manually entered address wherein the resulting confidence may be based on validation(s) performed (e.g. compare recent history for plausible current address, use current latitude and longitude for database lookup to compare with address information entered, etc). The system and/or user may or may not be able to override the confidence value determined.

LOCATION TECHNOLOGY field **1100***e* is preferably set with: "Manual", or "Manual Validated". Types of validations may further be elaborated. The originator indicator is set to DLM.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: null (not set).

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: null (not set).

SPEED field **1100***h* is preferably set with: null (not set).

HEADING field **1100***i* is preferably set with: null (not set).

ELEVATION field **1100***j* is preferably set with: null (not set).

APPLICATION FIELDS field **1100***k* is preferably set with: Same as was described for FIG. **2**D (block **236**) above; or as decided by the user.
CORRELATION FIELD **1100***m* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).
SENT DATE/TIME STAMP field **1100***n* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).
RECEIVED DATE/TIME STAMP field **1100***p* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

FIG. **9**A depicts a table for illustrating heterogeneously locating a MS, for example a DLM **200**. While many location methods and systems have been exhausted above, there may be other system and methods for locating an MS which apply to the present disclosure. The requirement for LBX is that the MS be located, regardless of how that occurs. MSs disclosed herein can be located by one or many location technologies discussed. As MS prices move lower, and capabilities increase, an affordable MS will contain multiple abilities for being located. GPS, triangulation, in-range detection, and contact sensory may all be used in locating a particular MS as it travels. Equipping the MS with all techniques is straightforward and is compelling when there are competing, or complementary, technologies that the MS should participate in.

The FIG. **9**A table has DLM location methods for rows and a single column for the MS (e.g. DLM **200**). Each location technology can be driven by the client (i.e. the MS), or a service (i.e. the location server(s)) as denoted by a row qualifier "C" for client or "S" for service. An MS may be located by many technologies. The table illustrated shows that the MS with unique identifier 0A12:43 EF:985B:012F is able to be heterogeneously located, specifically with local MS GPS capability, service side cell tower in-range detection, service side cell tower TDOA, service side cell tower MPT (combination of TDOA and AOA), service side antenna in-range detection, service side antenna AOA, service side antenna TDOA, service side antenna MPT, service side contact/sensory, and general service side MPT. The unique identifier in this example is a universal product identifier (like Host Bus Adapter (HBA) World Wide Name (WWN) identifiers are generated), but could be in other form as described above (e.g. phone #214-403-4071). An MS can have any subset of technologies used to locate it, or all of the technologies used to locate it at some time during its travels. An MS is heterogeneously located when two or more location technologies are used to locate the MS during MS travels and/or when two or more location technologies with incomplete results are used in conjunction with each other to locate the MS during MS travels, such as MPT. MPT is a heterogeneous location technology because it uses at least two different methods to accomplish a single location determination. Using combinations of different location technologies can be used, for example a TDOA measurement from an in-range antenna with a TDOA measurement relative a cell tower (e.g. as accomplished in MS processing of FIG. **26**B), using completely different services that have no knowledge of each other. Another combination is to use a synergy of whereabouts data from one technology with whereabouts data from another technology. For example, in-range detection is used in combination with graphical identification to provide better whereabouts of a MS. In another example, a GPS equipped MS travels to an area where GPS does not work well (e.g. downtown amidst large and tall buildings). The DLM becomes an ILM, and is triangulated relative other MSs. So, an MS is heterogeneously located using two or more technologies to determine a single whereabouts, or different whereabouts of the MS during travel.

FIG. **9**B depicts a flowchart for describing a preferred embodiment of heterogeneously locating a MS, for example DLM **200**. While heterogeneously locating an MS can occur by locating the MS at different times using different location technologies, flowchart **9**B is shown to discuss a generalization of using different location technologies with each other at the same time to locate an MS. Processing begins at block **950** and continues to block **952** where a plurality of parameters from more than one location technology are examined for locating an MS. Processing begins at block **950** by a service (or the MS) when a location technology by itself cannot be used to confidently locate the MS. Data deemed useful at block **952**, when used in conjunction with data from a different location technology to confidently locate the MS, is passed for processing to block **954**. Block **954** heterogeneously locates the MS using data from at least two location technologies to complement each other and to be used in conjunction with each other in order to confidently locate the MS. Once the MS whereabouts are determined at block **954**, WDR information is communicated to the MS for further processing at block **956**. In some embodiments where a service is heterogeneously locating the MS, block **956** communicates WDR information wirelessly to the MS before processing begins at block **958**. In another embodiment where the MS is heterogeneously locating itself, block **956** communicates WDR information internally to WDR completion processing at block **958**. In preferred embodiments, the MS completes its WDR information at block **958**, FIG. **2**F parameters are prepared at block **960**, and the MS invokes FIG. **2**F processing already described above (at block **962**), before processing terminates at block **964**. Parameters set at block **960** are: WDRREF=a reference or pointer to the MS WDR; DELETEQ=FIG. **9**B location queue discard processing; and SUPER=FIG. **9**B supervisory notification processing. WDR **1100** fields (see FIG. **11**A) are set analogously in light of many variations already described above.

In some embodiments of FIG. **9**B processing, Missing Part Triangulation (MPT) is used to heterogeneously locate an MS. For a service side embodiment example, block **950** begins service processing when TDOA information itself cannot be used to confidently locate the MS, or AOA information itself cannot be used to confidently locate the MS, however using angles and distances from each in conjunction with each other enables solving whereabouts confidently. See "Missing Part Triangulation (MPT)" section below with discussions for FIGS. **11**A through **11**E for MPT processing of blocks **952** and **954**. Data discovered at block **952** and processed by block **954** depends on the embodiment, what stationary reference point locations are known at the time of blocks **952** and **954** processing, and which parts are missing for triangulating the MS. Having three (3) sides (all TDOA) with known stationary vertices location(s) solves the triangle for locating the MS. Three (3) angles (all AOA) with known stationary vertices location(s) solves the triangle for locating the MS. Those skilled in the art appreciate that solving triangulation can make complementary use of different distances (time used to determine length in TDOA) and angles (from AOA) for deducing a MS location confidently (e.g. MPT). Those skilled in the art recognize that having stationary reference locations facilitates requiring less triangular information for deducing a MS location confidently.

While MPT has been discussed by example, flowchart **9**B is not to be interpreted in a limiting sense. Any location technologies, for example as shown in FIG. **9**A, can be used in conjunction with each other when not all information required is available in a single location technology to confidently deduce an MS location. Data available from the different location technologies available will be examined on its own merits, and optionally used in conjunction to deduce a confident location. For example, a TDOA (difference between when signal sent and when received) measurement from "coming within range" technology can be used to distinguish how close, or how far, is an MS in the vicinity. That measurement may be used to more confidently locate the MS using other TDOA measurements from other unrelated "coming within range" whereabouts information.

With the many DLM examples above, it should be clear now to the reader how to set the WDR **1100** for DLM invoked FIG. **2**F processing. There can be other location technologies that will set WDR **1100** fields analogously. Locating methodologies of FIGS. **2**A through **9**B can be used in any combination, for example for more timely or accurate locating. Furthermore, a MS automatically takes on a role of a DLM or ILM depending on what capability is available at the time, regardless of whether or not the MS is equipped for being directly located. As a DLM roams to unsupported areas, it can remain a DLM using different DLM technologies, and it can become an ILM to depend on other MSs (ILMs or DLMs) in the vicinity to locate it.

LBX Indirectly Located Mobile Data Processing Systems (ILMs)

FIGS. **10**A and **10**B depict an illustration of a Locatable Network expanse (LN-Expanse) **1002** for describing locating of an ILM with all DLMs. With reference now to FIG. **10**A, DLM **200***a*, DLM **200***b*, DLM **200***c*, DLM **200***d*, and DLM **200***e* (referred to generally in FIGS. **10**A and **10**B discussions as DLMs **200**) are each automatically and directly located, for example using any of the automatic location technologies heretofore described. ILM **1000***b* is automatically located using the reference locations of DLM **200***b*, DLM **200***c*, and DLM **200***e*. DLMs **200** can be mobile while providing reference locations for automatically determining the location of ILM **1000***b*. Timely communications between MSs is all that is required for indirectly locating MSs. In some embodiments, DLMs **200** are used to triangulate the position of ILM **1000***b* using aforementioned wave spectrum(s) reasonable for the MSs. Different triangulation embodiments can triangulate the location of ILM **1000***b* using TDOA, AOA, or MPT, preferably by the ILM **1000***b* seeking to be located. In other embodiments, TDOA information is used to determine how close ILM **1000***b* is to a DLM for associating the ILM at the same location of a DLM, but with how close nearby. In other embodiments, an ILM is located by simply being in communications range to another MS. DLMs **200** can be referenced for determining elevation of an ILM. The same automatic location technologies used to locate a DLM can be used to automatically locate an ILM, except the DLMs are mobile and serve as the reference points. It is therefore important that DLM locations be timely known when references are needed for locating ILMs. Timely ILM interactions with other MSs, and protocol considerations are discussed in architecture **1900** below. DLMs **200***b*, **200***c*, and **200***e* are preferably selected for locating ILM **1000***b* by their WDR high confidence values, however any other WDR data may be used whereby wave spectrum, channel signal strength, time information, nearness, surrounded-ness, etc is considered for generating a confidence field **1100***d* of the WDR **1100** for the located ILM. Preferably, those considerations are factored into a confidence value, so that confidence values can be completely relied upon.

With reference now to FIG. **10**B, ILM **1000***c* has been located relative a plurality of DLMs, namely DLM **200***b*, DLM **200***d*, and DLM **200***e*. ILM **1000***c* is located analogously to ILM **1000***b* as described for FIG. **10**A, except there are different DLMs involved with doing the locating of ILM **1000***c* because of a different location of ILM **1000***c*. FIGS. **10**A and **10**B illustrate that MSs can be located using other MSs, rather than fixed stationary references described for FIGS. **2**A through **9**B. ILM **1000***b* and ILM **1000***c* are indirectly located using DLMs **200**.

FIG. **10**C depicts an illustration of a Locatable Network expanse (LN-Expanse) **1002** for describing locating of an ILM with an ILM and DLM. ILM **1000***a* is automatically located using the reference locations of DLM **200***c*, DLM **200***b*, and ILM **1000***b*. DLM **200***b*, DLM **200***c* and ILM **1000***b* can be mobile while providing reference locations for automatically determining the location of ILM **1000***a*. In some embodiments, MSs are used to triangulate the position of ILM **1000***a* using any of the aforementioned wave spectrum(s) (e.g. WiFi, cellular radio, etc) reasonable for the MSs. Different triangulation embodiments can triangulate the location of ILM **1000***a* using TDOA, AOA, or MPT, preferably by the ILM **1000***a* seeking to be located. In other embodiments, TDOA information is used to determine how close ILM **1000***a* is to a MS (DLM or ILM) for associating the ILM at the same location of a MS, but with how close nearby. In other embodiments, an ILM is located by simply being in communications range to another MS. DLMs or ILMs can be referenced for determining elevation of ILM **1000***a*. The same automatic location technologies used to locate a MS (DLM or ILM) are used to automatically locate an ILM, except the MSs are mobile and serve as the reference points. It is therefore important that MS (ILM and/or DLM) locations be timely known when references are needed for locating ILMs. Timely ILM interactions with other MSs, and protocol considerations are discussed in architecture **1900** below. DLM **200***b*, DLM **200***c*, and ILM **1000***b* are preferably selected for locating ILM **1000***a* by their WDR high confidence values, however any other WDR data may be used whereby wave spectrum, channel signal strength, time information, nearness, surrounded-ness, etc is considered for generating a confidence field **1100***d* of the WDR **1100** for the located ILM. Preferably, those considerations were already factored into a confidence value so that confidence values can be completely relied upon. ILM **1000***a* is indirectly located using DLM(s) and ILM(s).

FIGS. **10**D, **10**E, and **10**F depict an illustration of a Locatable Network expanse (LN-Expanse) **1002** describing locating of an ILM with all ILMs. With reference now to FIG. **10**D, ILM **1000***e* is automatically located using the reference locations of ILM **1000***a*, ILM **1000***b*, and ILM **1000***c*. ILM **1000***a*, ILM **1000***b* and ILM **1000***c* can be mobile while providing reference locations for automatically determining the location of ILM **1000***e*. Timely communications between MSs is all that is required. In some embodiments, MSs are used to triangulate the position of ILM **1000***e* using any of the aforementioned wave spectrum(s) reasonable for the MSs. Different triangulation embodiments can triangulate the location of ILM **1000***e* using TDOA, AOA, or MPT processing (relative ILMs **1000***a* through **1000***c*), preferably by the ILM **1000***e* seeking to be located. ILMs can be referenced for determining elevation of ILM **1000***e*. The same automatic location technologies used to locate a MS (DLM or ILM) are used to automatically locate an ILM, except the MSs are mobile and serve as the reference points. It is therefore important that ILM locations be timely known when references are needed for locating ILMs. Timely ILM interactions with other MSs, and protocol considerations are discussed in architecture **1900** below. ILM **1000***a*, ILM **1000***b*, and ILM **1000***c* are preferably selected for locating ILM **1000**e by their WDR high confidence values, however any other WDR data may be used whereby wave spectrum, channel signal strength, time information, nearness, surrounded-ness, etc is considered for generating a confidence field **1100**d of the WDR **1100** for the located ILM. Preferably, those considerations were already factored into a confidence value so that confidence values can be completely relied upon. ILM **1000**e is indirectly located using ILM **1000**a, ILM **1000**b, and ILM **1000**c.

With reference now to FIG. **10**E, ILM **1000**g is automatically located using the reference locations of ILM **1000**a, ILM **1000**c, and ILM **1000**e. ILM **1000**a, ILM **1000**c and ILM **1000**e can be mobile while providing reference locations for automatically determining the location of ILM **1000**g. ILM **1000**g is located analogously to ILM **1000**e as described for FIG. **10**D, except there are different ILMs involved with doing the locating of ILM **1000**g because of a different location of ILM **1000**g. Note that as ILMs are located in the LN-expanse **1002**, the LN-expanse expands with additionally located MSs.

With reference now to FIG. **10**F, ILM **1000**i is automatically located using the reference locations of ILM **1000**f, ILM **1000**g, and ILM **1000**h. ILM **1000**f, ILM **1000**g and ILM **1000**h can be mobile while providing reference locations for automatically determining the location of ILM **1000**i. ILM **1000**i is located analogously to ILM **1000**e as described for FIG. **10**D, except there are different ILMs involved with doing the locating of ILM **1000**i because of a different location of ILM **1000**i. FIGS. **10**D through **10**F illustrate that an MS can be located using all ILMs, rather than all DLMs (FIGS. **10**A and **10**B), a mixed set of DLMs and ILMs (FIG. **10**C), or fixed stationary references (FIGS. **2**A through **9**B). ILMs **1000**e, **1000**g, and **1000**i are indirectly located using ILMs. Note that in the FIG. **10** illustrations the LN-expanse **1002** has expanded down and to the right from DLMs directly located up and to the left. It should also be noted that locating any MS can be done with at least one other MS. Three are not required as illustrated. It is preferable that triangulation references used surround an MS.

FIGS. **10**G and **10**H depict an illustration for describing the reach of a Locatable Network expanse (LN-Expanse) according to MSs. Location confidence will be dependent on the closest DLMs, how stale an MS location becomes for serving as a reference point, and how timely an MS refreshes itself with a determined location. An MS preferably has highest available processing speed with multithreaded capability in a plurality of hardware processors and/or processor cores. A substantially large number of high speed concurrent threads of processing that can occur within an MS provides for an optimal capability for being located quickly among its peer MSs, and for serving as a reference to its peer MSs. MS processing described in flowcharts herein assumes multiple threads of processing with adequate speed to accomplish an optimal range in expanding the LN-Expanse **1002**.

With reference now to FIG. **10**G, an analysis of an LN-Expanse **1002** will contain at least one DLM region **1022** containing a plurality of DLMs, and at least one DLM indirectly located region **1024** containing at least one ILM that has been located with all DLMs. Depending on the range, or scope, of an LN-Expanse **1002**, there may be a mixed region **1026** containing at least one ILM that has been indirectly located by both an ILM and DLM, and there may be an exclusive ILM region **1028** containing at least one ILM that has been indirectly located by all ILMs. The further in distance the LN-Expanse has expanded from DLM region **1022** with a substantial number of MSs, the more likely there will an exclusive ILM region **1028**. NTP may be available for use in some regions, or some subset of a region, yet not available for use in others. NTP is preferably used where available to minimize communications between MSs, and an MS and service(s). An MS has the ability to make use of NTP when available.

With reference now to FIG. **10**H, all MSs depicted know their own locations. The upper left-hand portion of the illustration consists of region **1022**. As the reader glances more toward the rightmost bottom portion of the illustration, there can be regions **1024** and regions **1026** in the middle of the illustration. At the very rightmost bottom portion of the illustration, remaining ILMs fall in region **1028**. An ILM is indirectly located relative all DLMs, DLMs and ILMs, or all ILMs. An "Affirmifier" in a LN-expanse confidently knows its own location and can serve as a reference MS for other MSs. An affirmifier is said to "affirmify" when in the act of serving as a reference point to other MSs. A "Pacifier" can contribute to locating other systems, but with a low confidence of its own whereabouts. The LN-Expanse is a network of located/locatable MSs, and is preferably expanded by a substantial number of affirmifiers.

FIG. **10**I depicts an illustration of a Locatable Network expanse (LN-Expanse) for describing a supervisory service, for example supervisory service **1050**. References in flowcharts for communicating information to a supervisory service can refer to communicating information to supervisory service **1050** (e.g. blocks **294** and **296** from parameters passed to block **272** for many processing flows). The only requirement is that supervisory service **1050** be contactable from an MS (DLM or ILM) that reports to it. An MS reporting to service **1050** can communicate directly to it, through another MS (i.e. a single hop), or through a plurality of MSs (i.e. a plurality of hops). Networks of MSs can be preconfigured, or dynamically reconfigured as MSs travel to minimize the number of hops between a reporting MS and service **1050**. A purely peer to peer preferred embodiment includes a peer to peer network of located/locatable MSs that interact with each other as described herein. The purely peer to peer preferred embodiment may have no need to include a service **1050**. Nevertheless, a supervisory service may be warranted to provide certain processing centralization, or for keeping information associated with MSs. In some embodiments, supervisory service **1050** includes at least one database to house data (e.g. data **8**; data **20**; data **36**; data **38**, queue data **22**, **24**, **26**; and/or history **30**) for any subset of MSs which communicate with it, for example to house MS whereabouts information.

FIG. **11**A depicts a preferred embodiment of a Whereabouts Data Record (WDR) **1100** for discussing operations of the present disclosure. A Whereabouts Data Record (WDR) **1100** may also be referred to as a Wireless Data Record (WDR) **1100**. A WDR takes on a variety of formats depending on the context of use. There are several parts to a WDR depending on use. There is an identity section which contains a MS ID field **1100**a for identifying the WDR. Field **1100**a can contain a null value if the WDR is for whereabouts information received from a remote source which has not identified itself. MSs do not require identities of remote data processing systems in order to be located. There is a core section which is required in WDR uses. The core section includes date/time stamp field **1100**b, location field **1100**c, and confidence field **1100**d. There is a transport section of fields wherein any one the fields may be used when communicating WDR information between data processing systems. Transport fields include correlation field **1100**m, sent date/time stamp field **1100**n, and received date/time stamp field **1100**p. Transport fields may also be communicated to send processing (e.g. queue **24**), or received from receive processing (e.g.

queue **26**). Other fields are of use depending on the MS or applications thereof, however location technology field **1100***e* and location reference info field **1100***f* are of particular interest in carrying out additional novel functionality of the present disclosure. Communications reference information field **1100***g* may be valuable, depending on communications embodiments in the LN-expanse.

Some fields are multi-part fields (i.e. have sub-fields). Whereabouts Data Records (WDRs) **1100** may be fixed length records, varying length records, or a combination with field(s) in one form or the other. Some WDR embodiments will use anticipated fixed length record positions for subfields that can contain useful data, or a null value (e.g. −1). Other WDR embodiments may use varying length fields depending on the number of sub-fields to be populated. Other WDR embodiments will use varying length fields and/or sub-fields which have tags indicating their presence. Other WDR embodiments will define additional fields to prevent putting more than one accessible data item in one field. In any case, processing will have means for knowing whether a value is present or not, and for which field (or sub-field) it is present. Absence in data may be indicated with a null indicator (−1), or indicated with its lack of being there (e.g. varying length record embodiments).

When a WDR is referenced in this disclosure, it is referenced in a general sense so that the contextually reasonable subset of the WDR of FIG. **11**A is used. For example, when communicating WDRs (sending/receiving data **1302** or **1312**) between data processing systems, a reasonable subset of WDR **1100** is communicated in preferred embodiments as described with flowcharts. When a WDR is maintained to queue **22**, preferably most (if not all) fields are set for a complete record, regardless if useful data is found in a particular field (e.g. some fields may be null (e.g. −1)). Most importantly, Whereabouts Data Records (WDRs) are maintained to queue **22** for maintaining whereabouts of the MS which owns queue **22**. LBX is most effective the more timely (and continuous) a MS has valid whereabouts locally maintained. WDRs are designed for maintaining whereabouts information independent of any location technology applied. Over time, a MS may encounter a plurality of location technologies used to locate it. WDRs maintained to a first MS queue **22** have the following purpose:

1) Maintain timely DLM whereabouts information of the first MS independent of any location technology applied;
2) Maintain whereabouts information of nearby MSs independent of any location technology applied;
3) Provide DLM whereabouts information to nearby MSs for determining their own locations (e.g. provide whereabouts information to at least a second MS for determining its own location);
4) Maintain timely ILM whereabouts information of the first MS independent of any location technology applied; and
5) Provide ILM whereabouts information to nearby MSs so they can determine their own locations (e.g. first MS providing whereabouts information to at least a second MS for the second MS determining its own whereabouts).

A MS may go in and out of DLM or ILM roles as it is mobile. Direct location methods are not always available to the MS as it roams, therefore the MS preferably does all of 1 through 5 above. When the WDR **1100** contains a MS ID field **1100***a* matching the MS which owns queue **22**, that WDR contains the location (location field **1100***c*) with a specified confidence (field **1100***d*) at a particular time (date/time stamp field **1100***b*) for that MS. Preferably the MS ID field **1100***a*, date/time stamp field **1100***b* and confidence field **1100***d* is all that is required for searching from the queue **22** the best possible, and most timely, MS whereabouts at the time of searching queue **22**. Other embodiments may consult any other fields to facilitate the best possible MS location at the time of searching and/or processing queue **22**. The WDR queue **22** also maintains affirmifier WDRs, and acceptable confidence pacifier WDRs (block **276**), which are used to calculate a WDR having matching MS field **1100***a* so the MS knows its whereabouts via indirect location methods. Affirmifier and pacifier WDRs have MS ID field **1100***a* values which do not match the MS owning queue **22**. This distinguishes WDRs of queue **22** for A) accessing the current MS location; from B) the WDRs from other MSs. All WDR fields of affirmifier and pacifier originated WDRs are of importance for determining a best location of the MS which owns queue **22**, and in providing LBX functionality.

MS ID field **1100***a* is a unique handle to an MS as previously described. Depending on the installation, MS ID field **1100***a* may be a phone #, physical or logical address, name, machine identifier, serial number, encrypted identifier, concealable derivative of a MS identifier, correlation, pseudo MS ID, or some other unique handle to the MS. An MS must be able to distinguish its own unique handle from other MS handles in field **1100***a*. For indirect location functionality disclosed herein, affirmifier and pacifier WDRs do not need to have a correct originating MS ID field **1100***a*. The MS ID may be null, or anything to distinguish WDRs for MS locations. However, to accomplish other LBX features and functionality, MS Identifiers (MS IDs) of nearby MSs (or unique correlations thereof) maintained in queue **22** are to be known for processing by an MS. MS ID field **1100***a* may contain a group identifier of MSs in some embodiments for distinguishing between types of MSs (e.g. to be treated the same, or targeted with communications, as a group), as long as the MS containing queue **22** can distinguish its own originated WDRs **1100**. A defaulted value may also be set for a "do not care" setting (e.g. null).

Date/Time stamp field **1100***b* contains a date/time stamp of when the WDR record **1100** was completed by an MS for its own whereabouts prior to WDR queue insertion. It is in terms of the date/time scale of the MS inserting the local WDR (NTP derived or not). Date/Time stamp field **1100***b* may also contain a date/time stamp of when the WDR record **1100** was determined for the whereabouts of an affirmifier or pacifier originating record **1100** to help an MS determine its own whereabouts, but it should still be in terms of the date/time scale of the MS inserting the local WDR (NTP derived or not) to prevent time conversions when needed, and to promote consistent queue **22** searches/sorts/etc. The date/time stamp field **1100***b* should use the best possible granulation of time, and may be in synch with other MSs and data processing systems according to NTP. A time zone, day/light savings time, and NTP indicator is preferably maintained as part of field **1100***b*. The NTP indicator (e.g. bit) is for whether or not the date/time stamp is NTP derived (e.g. the NTP use setting is checked for setting this bit when completing the WDR for queue **22** insertion). In some embodiments, date/time stamp field **1100***b* is measured in the same granulation of time units to an atomic clock available to MSs of an LN-Expanse **1002**. When NTP is used in a LN-Expanse, identical time server sources are not a requirement provided NTP derived date/time stamps have similar accuracy and dependability.

Location field **1100***c* depends on the installation of the present disclosure, but can include a latitude and longitude, cellular network cell identifier, geocentric coordinates, geodetic coordinates, three dimensional space coordinates, area described by GPS coordinates, overlay grid region identifier or coordinates, GPS descriptors, altitude/elevation (e.g. in lieu of using field **1100***j*), MAPSCO reference, physical or logical network address (including a wildcard (e.g. ip addresses 145.32.*.*)), particular address, polar coordinates, or any other two/three dimensional location methods/means used in identifying the MS location. Data of field **1100***c* is preferably a consistent measure (e.g. all latitude and longitude) for all location technologies that populate WDR queue **22**. Some embodiments will permit using different measures to location field **1100***c* (e.g. latitude and longitude for one, address for another; polar coordinates for another, etc) which will be translated to a consistent measure at appropriate processing times.

Confidence field **1100***d* contains a value for the confidence that location field **1100***c* accurately describes the location of the MS when the WDR is originated by the MS for its own whereabouts. Confidence field **1100***d* contains a value for the confidence that location field **1100***c* accurately describes the location of an affirmifier or pacifier that originated the WDR. A confidence value can be set according to known timeliness of processing, communications and known mobile variables (e.g. MS speed, heading, yaw, pitch, roll, etc) at the time of transmission. Confidence values should be standardized for all location technologies used to determine which location information is of a higher/lower confidence when using multiple location technologies (as determined by fields **1100***e* and **1100***f*) for enabling determination of which data is of a higher priority to use in determining whereabouts. Confidence value ranges depend on the implementation. In a preferred embodiment, confidence values range from 1 to 100 (as discussed previously) for denoting a percentage of confidence. 100% confidence indicates the location field **1100***c* is guaranteed to describe the MS location. 0% confidence indicates the location field **1100***c* is guaranteed to not describe the MS location. Therefore, the lowest conceivable value of a queue **22** for field **1100***d* should be 1. Preferably, there is a lowest acceptable confidence floor value configured (by system, administrator, or user) as used at points of queue entry insertion—see block **276** to prevent frivolous data to queue **22**. In most cases, WDRs **1100** contain a confidence field **1100***d* up to 100. In confidence value preferred embodiments, pacifiers know their location with a confidence of less than 75, and affirmifiers know their location with a confidence value 75 or greater. The confidence field is skewed to lower values as the LN-expanse **1002** is expanded further from region **1022**. Confidence values are typically lower when ILMs are used to locate a first set of ILMs (i.e. first tier), and are then lower when the first set of ILMs are used to locate a second set of ILMs (second tier), and then lower again when the second set of ILMs are used to locate a third set of ILMs (third tier), and so on. Often, examination of a confidence value in a WDR **1100** can indicate whether the MS is a DLM, or an ILM far away from DLMs, or an MS which has been located using accurate (high confidence) or inaccurate (low confidence) locating techniques.

Location Technology field **1100***e* contains the location technology used to determine the location of location field **1100***c*. An MS can be located by many technologies. Location Technology field **1100***e* can contain a value from a row of FIG. **9**A or any other location technology used to locate a MS. WDRs inserted to queue **22** for MS whereabouts set field **1100***e* to the technology used to locate the MS. WDRs inserted to queue **22** for facilitating a MS in determining whereabouts set field **1100***e* to the technology used to locate the affirmifier or pacifier. Field **1100***e* also contains an originator indicator (e.g. bit) for whether the originator of the WDR **1100** was a DLM or ILM. When received from a service that has not provided confidence, this field may be used by a DLM to determine confidence field **1100***d*.

Location Reference Info field **1100***f* preferably contains one or more fields useful to locate a MS in processing subsequent of having been inserted to queue **22**. In other embodiments, it contains data that contributed to confidence determination. Location Reference Info field **1100***f* may contain information (TDOA measurement and/or AOA measurement—see inserted field **1100***f* for FIGS. **2**D, **2**E and **3**C) useful to locate a MS in the future when the WDR originated from the MS for its own whereabouts. Field **1100***f* will contain selected triangulation measurements, wave spectrum used and/or particular communications interfaces **70**, signal strength(s), TDOA information, AOA information, or any other data useful for location determination. Field **1100***f* can also contain reference whereabouts information (FIG. **3**C) to use relative a TDOA or AOA (otherwise WDR location field assumed as reference). In one embodiment, field **1100***f* contains the number of DLMs and ILMs which contributed to calculating the MS location to break a tie between using WDRs with the same confidence values. In another embodiment, a tier of ILMs used to locate the MS is maintained so there is an accounting for the number of ILMs in the LN-expanse between the currently located MS and a DLM. In other embodiments, MS heading, yaw, pitch and roll, or accelerometer values are maintained therein, for example for antenna AOA positioning. When wave spectrum frequencies or other wave characteristics have changed in a transmission used for calculating a TDOA measurement, appropriate information may be carried along, for example to properly convert a time into a distance. Field **1100***f* should be used to facilitate correct measurements and uses, if needed conversions have not already taken place.

Communications reference information field **1100***g* is a multipart record describing the communications session, channel, and bind criteria between the MS and MSs, or service(s), that helped determine its location. In some embodiments, field **1100***g* contains unique MS identifiers, protocol used, logon/access parameters, and useful statistics of the MSs which contributed to data of the location field **1100***c*. An MS may use field **1100***g* for WDRs originated from affirmifiers and pacifiers for subsequent LBX processing.

Speed field **1100***h* contains a value for the MS speed when the WDR is originated by the MS for its own whereabouts. Speed field **1100***d* may contain a value for speed of an affirmifier or pacifier when the WDR was originated elsewhere. Speed is maintained in any suitable units.

Heading field **1100***i* contains a value for the MS heading when the WDR is originated by the MS for its own whereabouts. Heading field **1100***i* may contain a value for heading of an affirmifier or pacifier when the WDR was originated elsewhere. Heading values are preferably maintained in degrees up to 360 from due North, but is maintained in any suitable directional form.

Elevation field **1100***j* contains a value for the MS elevation (or altitude) when the WDR is originated by the MS for its own whereabouts. Elevation field **1100***j* may contain a value for elevation (altitude) of an affirmifier or pacifier when the WDR was originated elsewhere. Elevation (or altitude) is maintained in any suitable units.

Application fields **1100***k* contains one or more fields for describing application(s) at the time of completing, or originating, the WDR **1100**. Application fields **1100***k* may include field(s) for:

a) MS Application(s) in use at time;
b) MS Application(s) context(s) in use at time;
c) MS Application(s) data for state information of MS Application(s) in use at time;
d) MS Application which caused WDR **1100**;
e) MS Application context which caused WDR **1100**;
f) MS Application data for state information of MS Application which caused WDR **1100**;
g) Application(s) in use at time of remote MS(s) involved with WDR;
h) Application(s) context(s) in use at time of remote MS(s) involved with WDR;
i) MS Application(s) data for state information of remote MS(s) involved with WDR;
j) Remote MS(s) criteria which caused WDR **1100**;
k) Remote MS(s) context criteria which caused WDR **1100**;
l) Remote MS(s) data criteria which caused WDR **1100**;
m) Application(s) in use at time of service(s) involved with WDR;
n) Application(s) context(s) in use at time of service(s) involved with WDR;
o) MS Application(s) data for state information of service (s) involved with WDR;
p) Service(s) criteria which caused WDR **1100**;
q) Service(s) context criteria which caused WDR **1100**;
r) Service(s) data criteria which caused WDR **1100**;
s) MS navigation APIs in use;
t) Web site identifying information;
u) Physical or logical address identifying information;
v) Situational location information as described in U.S. Pat. Nos. 6,456,234; 6,731,238; 7,187,997 (Johnson);
w) Transactions completed at a MS;
x) User configurations made at a MS;
y) Environmental conditions of a MS;
z) Application(s) conditions of a MS;
aa) Service(s) conditions of a MS;
bb) Date/time stamps (like field **1100***b*) with, or for, any item of a) through aa); and/or
cc) Any combinations of a) through bb).

Correlation field **1100***m* is optionally present in a WDR when the WDR is in a transmission between systems (e.g. wireless communications) such as in data **1302** or **1312**. Field **1100***m* provides means for correlating a response to an earlier request, or to correlate a response to an earlier broadcast. Correlation field **1100***m* contains a unique handle. In a LN-expanse which globally uses NTP, there is no need for correlation in data **1302** or **1312**. Correlation field **1100***m* may be present in WDRs of queues **24** or **26**. Alternatively, a MS ID is used for correlation.

Sent date/time stamp field **1100***n* is optionally present in a WDR when the WDR is in transmission between systems (e.g. wireless communications) such as in data **1302** or **1312**. Field **1100***n* contains when the WDR was transmitted. A time zone, day/light savings time, and NTP indicator is preferably maintained as part of field **1100***n*. Field **1100***n* is preferably not present in WDRs of queue **22** (but can be if TDOA measurement calculation is delayed to a later time). In some embodiments, there is no need for field **1100***n*. Whereabouts determined for MSs of an LN-Expanse may be reasonably timely, facilitating simplicity of setting outbound field **1100***b* to the transmission date/time stamp at the sending data processing system, rather than when the WDR was originally completed for whereabouts (e.g. when substantially the same time anyway). Sent date/time field **1100***n* may be present in WDRs of queues **24** or **26**.

Received date/time stamp field **1100***p* is preferably present in a WDR when inserted to queue **26** by receiving thread(s) upon received data **1302** or **1312**. Field **1100***p* contains when the WDR was received by the MS. A time zone, day/light savings time, and NTP indicator is preferably maintained as part of field **1100***p*. Field **1100***p* is preferably not present in WDRs of queue **22** (but can be if TDOA measurement calculation is delayed to a later time). In some embodiments, there is no need for field **1100***p*. For example, thread(s) **1912** may be listening directly on applicable channel(s) and can determine when the data is received. In another embodiment, thread(s) **1912** process fast enough to determine the date/time stamp of when data **1302** or **1312** is received since minimal time has elapsed between receiving the signal and determining when received. In fact, known processing duration between when received and when determined to be received can be used to correctly alter a received date/time stamp. Received date/time stamp field **1100***p* is preferably added to records placed to queue **26** by receiving thread(s) feeding queue **26**.

Any fields of WDR **1100** which contain an unpredictable number of subordinate fields of data preferably use a tagged data scheme, for example an X.409 encoding for a Token, Length, and Value (called a TLV encoding). Therefore, a WDR **1100**, or field therein, can be a variable sized record. For example, Location Reference info field **1100***f* may contain TTA, 8, 0.1456 where the Token="TTA" for Time Till Arrival (TDOA measurement between when sent and when received), Length=8 for 8 bytes to follow, and Value=0.1456 in time units contained within the 8 bytes; also SS, 4, 50 where Token="Signal Strength", 4=4 for 4 bytes to follow, and Value=50 dBu for the signal strength measurement. This allows on-the-fly parsing of unpredictable, but interpretable, multipart fields. The TLV encoding also enables-on-the-fly configuration for parsing new subordinate fields to any WDR **1100** field in a generic implementation, for example in providing parse rules to a Lex and Yacc implementation, or providing parse rules to a generic top down recursive TLV encoding parser and processor.

Any field of WDR **1100** may be converted: a) prior to insertion to queue **22**; or b) after access to queue **22**; or c) by queue **22** interface processing; for standardized processing. Any field of WDR **1100** may be converted when sending/receiving/broadcasting, or related processing, to ensure a standard format. Other embodiments will store and access values of WDR **1100** field(s) which are already in a standardized format. WDR **1100** fields can be in any order, and a different order when comparing what is in data transmitted versus data maintained to queue **22**.

An alternate embodiment to WDRs maintained to queue **22** preserves transport fields **1100***m*, **1100***n* and/or **1100***p*, for example for use on queue **22**. This would enable **1952** thread(s) to perform TDOA measurements that are otherwise calculated in advance and kept in field **1100***f*. However, queue **22** size should be minimized and the preferred embodiment uses transport fields when appropriate to avoid carrying them along to other processing.

FIGS. **11**B, **11**C and **11**D depict an illustration for describing various embodiments for determining the whereabouts of an MS, for example an ILM **1000***e*. With reference now to FIG. **11**B, a MS **1000***e* location is located by using locations of three (3) other MSs: MS_4, MS_5, and MS_6 (referred to generally as MS_j). MS_j are preferably located with a reasonably high level of confidence. In some embodiments, MS_j are all DLMs. In some embodiments, MS_j are all ILMs. In some embodiments, MS_j are mixed DLMs and ILMs. Any of the MSs may be mobile during locating of MS **1000***e*. Wave spectrums in use, rates of data communications and MS processing speed, along with timeliness of processing described below, provide timely calculations for providing whereabouts of ILM **1000***e* with a high level of confidence. The most confident MSs (MS_j) were used to determine the MS **1000***e* whereabouts. For example, MS_j were all located using a form of GPS, which in turn was used to triangulate the whereabouts of MS **1000***e*. In another example, MS_4 was located by a form of triangulation technology, MS_5 was located by a form of "coming into range" technology, and MS_6 was located by either of the previous two, or some other location technology. It is not important how an MS is located. It is important that each MS know its own whereabouts and maintain a reasonable confidence to it, so that other MSs seeking to be located can be located relative highest confidence locations available. The WDR queue **22** should always contain at least one entry indicating the location of the MS **2** which owns WDR queue **22**. If there are no entries contained on WDR queue **22**, the MS **2** does not know its own location.

With reference now to FIG. **11**C, a triangulation of MS **1000***e* at location **1102** is explained using location (whereabouts) **1106** of MS_4, location (whereabouts) **1110** of MS_5, and location (whereabouts) **1114** of MS_6. Signal transmission distance from MS_j locations are represented by the radiuses, with r_1 the TDOA measurement (time difference between when sent and when received) between MS_4 and MS **1000***e*, with r_2 the TDOA measurement (time difference between when sent and when received) between MS_5 and MS **1000***e*, with r_3 the TDOA measurement (time difference between when sent and when received) between MS_6 and MS **1000***e*. In this example, the known locations of MS_S which are used to determine the location of MS **1000***e* allow triangulating the MS **1000***e* whereabouts using the TDOA measurements. In fact, less triangular data in the illustration can be necessary for determining a highly confident whereabouts of MS **1000***e*.

With reference now to FIG. **11**D, a triangulation of MS **1000***e* at location **1102** is explained using location (whereabouts) **1106** of MS_4, location (whereabouts) **1110** of MS_5, and location (whereabouts) **1114** of MS_6. In some embodiments, AOA measurements taken at a positioned antenna of MS **1000***e* at location **1102** are used relative the whereabouts **1106**, whereabouts **1110**, whereabouts **1114** (AOA **1140**, AOA **1144** and AOA **1142**), wherein AOA measurements are detected for incoming signals during known values for MS heading **1138** with MS yaw, pitch, and roll (or accelerometer readings). AOA triangulation is well known in the art. Line segment **1132** represents the direction of signal arrival to the antenna at whereabouts **1102** from MS_4 at whereabouts **1106**. Line segment **1134** represents the direction of signal arrival to the antenna at whereabouts **1102** from MS_5 at whereabouts **1110**. Line segment **1136** represents the direction of signal arrival to the antenna at whereabouts **1102** from MS_6 at whereabouts **1114**. In this example, the known locations of MS_S which are used to determine the location of MS **1000***e* allow triangulating the MS **1000***e* whereabouts using the AOA measurements. In fact, less triangular data in the illustration can be necessary for determining a highly confident whereabouts of MS **1000***e*. Alternative embodiments will use AOA measurements of outbound signals from the MS at whereabouts **1102** detected at antennas of whereabouts **1106** and/or **1110** and/or **1114**.

Missing Part Triangulation (MPT)

FIGS. **11**C and **11**D illustrations can be used in a complementary manner when only one or two TDOA measurements are available and/or not all stationary locations, or MS reference locations, are known at the time of calculation. Another example is when only one or two AOA angles is available and/or not all stationary locations, or MS reference locations, are known at the time of calculation. However, using what is available from each technology in conjunction with each other allows solving the MS whereabouts (e.g. blocks **952**/**954** processing above). MPT is one example of solving for missing parts using more than one location technology. Condition of data known for locating a MS (e.g. whereabouts **1106**, **1110** and **1114**) may be the following:

1) AAS=two angles and a side;
2) ASA=two angles and a common side;
3) SAS=two sides and the included angle; or
4) SSA=two sides and a non-included angle.

TDOA measurements are distances (e.g. time difference between when sent and when received), and AOA measurements are angles. Each of the four conditions are recognized (e.g. block **952** above), and data is passed for each of the four conditions for processing (e.g. block **954** above). For AAS (#1) and ASA (#2), processing (e.g. block **954**) finds the third angle by subtracting the sum of the two known angles from 180 degrees (i.e. using mathematical law that triangles' interior angles add up to 180 degrees), and uses the mathematical law of Sines (i.e. a/sin A=b/sin B=c/sin C) twice to find the second and third sides after plugging in the knowns and solving for the unknowns. For SAS (#3), processing (e.g. block **954**) uses the mathematical law of Cosines (i.e. $a^2=b^2+c^2-2bc\cos A$) to find the third side, and uses the mathematical law of Sines (sin A/a=sin B/b=sin C/c (derived from law of Sines above)) to find the second angle. For SSA (#4), processing (e.g. block **954**) uses the mathematical law of Sines (i.e. (sin A/a=sin B/b=sin C/c) twice to get the second angle, and mathematical law of Sines (a/sin A=b/sin B=c/sin C) to get the third side. Those skilled in the art recognize other useful trigonometric functions and formulas, and similar uses of the same trigonometric functions, for MPT depending on what data is known. The data discovered and processed depends on an embodiment, what reference locations are available, and which parts are missing for MPT. MPT uses different distances (time used to determine length in TDOA) and/or angles (from AOA or TDOA technologies) for deducing a MS location confidently (e.g. MPT). Even a single AOA measurement from a known reference location (stationary or MS) with a single TDOA measurement relative that reference location can be used to confidently locate a MS, and triangulation measurements used to deduce a MS location need not be from the same location technologies or wave spectrums. Those skilled in the art recognize that having known reference locations facilitates requiring less triangular information for deducing a MS location confidently. MPT embodiments may exist for any aforementioned wave spectrums.

FIG. **11**E depicts an illustration for describing various embodiments for automatically determining the location of an MS. An MS can be located relative other MSs which were located using any of a variety of location technologies, for example any of those of FIG. **9**A. An MS is heterogeneously located when one of the following conditions are met:

- More than one location technology is used during travel of the MS;
- More than one location technology is used to determine a single whereabouts of the MS;
- MPT is used to locate the MS; and/or
- ADLT is used to locate the MS.

The WDR queue **22** and interactions between MSs as described below cause the MS to be heterogeneously located without special consideration to any particular location technology. While WDR **1100** contains field **1100***e*, field **1100***d* provides a standard and generic measurement for evaluating WDRs from different location technologies, without concern for the location technology used. The highest confidence entries to a WDR queue **22** are used regardless of which location technology contributed to the WDR queue **22**.

LBX Configuration

FIG. **12** depicts a flowchart for describing an embodiment of MS initialization processing. Depending on the MS, there are many embodiments of processing when the MS is powered on, started, restarted, rebooted, activated, enabled, or the like. FIG. **12** describes the blocks of processing relevant to the present disclosure as part of that initialization processing. It is recommended to first understand discussions of FIG. **19** for knowing threads involved, and variables thereof. Initialization processing starts at block **1202** and continues to block **1204** where the MS Basic Input Output System (BIOS) is initialized appropriately, then to block **1206** where other character **32** processing is initialized, and then to block **1208** to check if NTP is enabled for this MS. Block **1206** may start the preferred number of listen/receive threads for feeding queue **26** and the preferred number of send threads for sending data inserted to queue **24**, in particular when transmitting CK **1304** embedded in usual data **1302** and receiving CK **1304** or **1314** embedded in usual data **1302** or **1312**, respectively. The number of threads started should be optimal for parallel processing across applicable channel(s). In this case, other character **32** threads are appropriately altered for embedded CK processing (sending at first opportune outbound transmission; receiving in usual inbound transmission).

If block **1208** determines NTP is enabled (as defaulted or last set by a user (i.e. persistent variable)), then block **1210** initializes NTP appropriately and processing continues to block **1212**. If block **1208** determines NTP was not enabled, then processing continues to block **1212**. Block **1210** embodiments are well known in the art of NTP implementations (also see block **1626**). Block **1210** may cause the starting of thread(s) associated with NTP. In some embodiments, NTP use is assumed in the MS. In other embodiments, appropriate NTP use is not available to the MS. Depending on the NTP embodiment, thread(s) may pull time synchronization information, or may listen for and receive pushed time information. Resources **38** (or other MS local resource) provides interface to an MS clock for referencing, maintaining, and generating date/time stamps at the MS. After block **1210** processing, the MS clock is synchronized to NTP. Because of initialization of the MS in FIG. **12**, block **1210** may rely on a connected service to initially get the startup synchronized NTP date/time. MS NTP processing will ensure the NTP enabled/disabled variable is dynamically set as is appropriate (using semaphore access) because an MS may not have continuous clock source access during travel when needed for resynchronization. If the MS does not have access to a clock source when needed, the NTP use variable is disabled. When the MS has (or again gets) access to a needed clock source, then the NTP use variable is enabled.

Thereafter, block **1212** creates shared memory to maintain data shared between processes/threads, block **1214** initializes persistent data to shared memory, block **1216** initializes any non-persistent data to shared memory (e.g. some statistics **14**), block **1218** creates system queues, and block **1220** creates semaphore(s) used to ensure synchronous access by concurrent threads to data in shared memory, before continuing to block **1222**. Shared memory data accesses appropriately utilize semaphore lock windows (semaphore(s) created at block **1220**) for proper access. In one embodiment, block **1220** creates a single semaphore for all shared memory accesses, but this can deteriorate performance of threads accessing unrelated data. In the preferred embodiment, there is a semaphore for each reasonable set of data of shared memory so all threads are fully executing whenever possible. Persistent data is that data which maintains values during no power, for example as stored to persistent storage **60**. This may include data **8** (including permissions **10**, charters **12**, statistics **14**, service directory **16**), data **20**, LBX history **30**, data **36**, resources **38**, and/or other data. Persistent data preferably includes at least the DLMV (see DLM role(s) list Variable below), ILMV (see ILM role(s) list Variable below), process variables 19xx-Max values (19xx=**1902**, **1912**, **1922**, **1932**, **1942** and **1952** (see FIG. **19** discussions below)) for the last configured maximum number of threads to run in the respective process, process variables 19xx-PID values (19xx=**1902**, **1912**, **1922**, **1932**, **1942** and **1952** (see FIG. **19** discussions below)) for multi-purpose of: a) holding an Operating System Process Identifier (i.e. O/S PID) for a process started; and b) whether or not the respective process was last enabled (i.e. PID>0) or disabled (i.e. PID<=0), the confidence floor value (see FIG. **14**A), the WTV (see Whereabouts Timeliness Variable (see FIG. **14**A)), the NTP use variable (see FIG. **14**A) for whether or not NTP was last set to disabled or enabled (used at block **1208**), and the Source Periodicity Time Period (SPTP) value (see FIG. **14**B). There are reasonable defaults for each of the persistent data prior to the first use of MS **2** (e.g. NTP use is disabled, and only becomes enabled upon a successful enabling of NTP at least one time). Non-persistent data may include data involved in some regard to data **8** (and subsets of permissions **10**, charters **12**, statistics **14**, service directory **16**), data **20**, LBX history **30**, data **36**, resources **38**, queues, semaphores, etc. Block **1218** creates queues **22**, **24**, and **26**. Queues **1980** and **1990** are also created there if required. Queues **1980** and **1990** are not required when NTP is in use globally by participating data processing systems. Alternate embodiments may use less queues by threads sharing a queue and having a queue entry type field for directing the queue entry to the correct thread. Alternate embodiments may have additional queues for segregating entries of a queue disclosed for best possible performance. Other embodiments incorporate queues figuratively to facilitate explanation of interfaces between processing.

All queues disclosed herein are understood to have their own internally maintained semaphore for queue accesses so that queue insertion, peeking, accessing, etc uses the internally maintained semaphore to ensure two or more concurrently executing threads do not corrupt or misuse data to any queue. This is consistent with most operating system queue interfaces wherein a thread stays blocked (preempted) after requesting a queue entry until a queue entry appears in the queue. Also, no threads will collide with another thread when inserting, peeking, or otherwise accessing the same queue. Therefore, queues are implicitly semaphore protected. Other embodiments may use an explicit semaphore protected window around queue data accessing, in which case those semaphore(s) are created at block **1220**.

Thereafter, block **1222** checks for any ILM roles currently enabled for the MS (for example as determined from persistent storage of an ILM role(s) list Variable (ILMV) preferably preconfigured for the MS at first use, or configured as last configured by a user of the MS). ILM roles are maintained to the ILM role(s) list Variable (ILMV). The ILMV contains one or more entries for an ILM capability (role), each entry with a flag indicating whether it is enabled or disabled (marked=enabled, unmarked=disabled). If block **1222** determines there is at least one ILM role enabled (i.e. as marked by associated flag), then block **1224** artificially sets the corresponding 19xx-PID variables to a value greater than 0 for indicating the process(es) are enabled, and are to be started by subsequent FIG. **12** initialization processing. The 19xx-PID will be replaced with the correct Process Identifier (PID) upon exit from block **1232** after the process is started. Preferably, every MS can have ILM capability. However, a user may want to (configure) ensure a DLM has no ILM capability enabled (e.g. or having no list present). In some embodiments, by default, every MS has an unmarked list of ILM capability maintained to the ILMV for 1) USE DLM REFERENCES and 2) USE ILM REFERENCES. USE DLM REFERENCES, when enabled (marked) in the ILMV, indicates to allow the MS of FIG. **12** processing to determine its whereabouts relative remote DLMs. USE ILM REFERENCES, when enabled (marked) in the ILMV, indicates to allow the MS of FIG. **12** processing to determine its whereabouts relative remote ILMs. Having both list items marked indicates to allow determining MS whereabouts relative mixed DLMs and ILMs. An alternative embodiment may include a USE MIXED REFERENCES option for controlling the MS of FIG. **12** processing to determine its whereabouts relative mixed DLMs and/or ILMs. Alternative embodiments will enforce any subset of these options without exposing user configurations, for example on a MS without any means for being directly located.

For any of the ILMV roles of USE DLM REFERENCES, USE ILM REFERENCES, or both, all processes **1902**, **1912**, **1922**, **1932**, **1942** and **1952** are preferably started (i.e. **1902**-PID, **1912**-PID, **1922**-PID, **1932**-PID, **1942**-PID and **1952**-PID are artificially set at block **1224** to cause subsequent process startup at block **1232**). Characteristics of an anticipated LN-expanse (e.g. anticipated location technologies of participating MSs, MS capabilities, etc) will start a reasonable subset of those processes with at least process **1912** started. Block **1224** continues to block **1226**. If block **1222** determines there are no ILMV role(s) enabled, then block processing continues to block **1226**.

Block **1226** initializes an enumerated process name array for convenient processing reference of associated process specific variables described in FIG. **19**, and continues to block **1228** where the first member of the set is accessed for subsequent processing. The enumerated set of process names has a prescribed start order for MS architecture **1900**. Thereafter, if block **1230** determines the process identifier (i.e. 19xx-PID such that 19xx is **1902**, **1912**, **1922**, **1932**, **1942**, **1952** in a loop iteration of blocks **1228** through **1234**) is greater than **0** (e.g. this first iteration of **1952**-PID>0 implies it is to be started here; also implies process **1952** is enabled as used in FIGS. **14**A, **28**, **29**A and **29**B), then block **1232** spawns (starts) the process (e.g. **1952**) of FIG. **29**A to start execution of subordinate worker thread(s) (e.g. process **1952** thread(s)) and saves the real PID (Process Identifier) to the PID variable (e.g. **1952**-PID) returned by the operating system process spawn interface. Block **1232** passes as a parameter to the process of FIG. **29**A which process name to start (e.g. **1952**), and continues to block **1234**. If block **1230** determines the current process PID variable (e.g. **1952**-PID) is not greater than 0 (i.e. not to be started; also implies is disabled as used in FIGS. **14**A, **28**, **29**A and **29**B), then processing continues to block **1234**. Block **1234** checks if all process names of the enumerated set (pattern of 19xx) have been processed (iterated) by blocks **1228** through **1234**. If block **1234** determines that not all process names in the set have been processed (iterated), then processing continues back to block **1228** for handling the next process name in the set. If block **1234** determines that all process names of the enumerated set were processed, then block **1236** checks the DLMV (DLM role(s) list Variable). Blocks **1228** through **1234** iterate every process name of FIG. **19** to make sure that each is started in accordance with non-zero 19xx-PID variable values at FIG. **12** initialization.

Block **1236** checks for any DLM roles currently enabled for the MS (for example as determined from persistent storage of a DLM role(s) list Variable (DLMV) preferably preconfigured for the MS at first use if the MS contains DLM capability). DLM capability (roles), whether on-board at the MS, or determined during MS travels (see block **288**), is maintained to the DLM role(s) list Variable (DLMV). The DLMV contains one or more entries for a DLM capability (role), each (role) entry with a flag indicating whether it is enabled or disabled (marked=enabled, unmarked=disabled). If block **1236** determines there is at least one DLM role enabled (i.e. as marked by associated flag), then block **1238** initializes enabled role(s) appropriately and processing continues to block **1240**. Block **1238** may cause the starting of thread(s) associated with enabled DLM role(s), for DLM processing above (e.g. FIGS. **2**A through **9**B). Block **1238** may invoke API(s), enable flag(s), or initialize as is appropriate for DLM processing described above. Such initializations are well known in the art of prior art DLM capabilities described above. If block **1236** determines there are no DLM roles to initialize at the MS, then processing continues to block **1240**. Any of the FIG. **9**A technologies are eligible in the DLMV as determined to be present at the MS and/or as determined by historical contents of the WDR queue **22** (e.g. location technology field **1100*e*** with MS ID field **1100*a*** for this MS) and/or determined by LBX history **30**. Application Programming Interfaces (APIs) may also be used to determine MS DLM capability (role(s)) for entry(s) to the DLMV.

Block **1240** completes LBX character initialization, and FIG. **12** initialization processing terminates thereafter at block **1242**. Depending on what threads were started as part of block **1206**, Block **1240** may startup the preferred number of listen/receive threads for feeding queue **26** and the preferred number of send threads for sending data inserted to queue **24**, in particular when transmitting new data **1302** and receiving new data **1302** or **1312**. The number of threads started should be optimal for parallel processing across applicable channel(s). Upon encounter of block **1242**, the MS is appropriately operational, and a user at the MS of FIG. **12** processing will have the ability to use the MS and applicable user interfaces thereof.

With reference now to FIG. **29**A, depicted is a flowchart for describing a preferred embodiment of a process for starting a specified number of threads in a specified thread pool. FIG. **29**A is in itself an O/S process, has a process identifier (PID) after being started, will contain at least two threads of processing after being started, and is generic in being able to take on the identity of any process name passed to it (e.g. 19xx) with a parameter (e.g. from block **1232**). FIG. **29**A represents the parent thread of a 19xx process. The FIG. **29**A process is generic for executing any of processes 19xx (i.e. **1902**, **1912**, **1922**, **1932**, **1942** and **1952**) with the prescribed number of worker threads using the 19xx-Max configuration (i.e. **1902**-Max, **1912**-Max, **1922**-Max, **1932**-Max, **1942**-Max and **1952**-Max). FIG. **29**A will stay running until it (first all of its worker thread(s)) is terminated. FIG. **29**A consists of an O/S Process 19xx with at least a parent thread (main thread) and one worker thread (or number of worker threads for FIG. **19** processing as determined by 19xx-Max). The parent thread has purpose to stay running while all worker threads are running, and to own intelligence for starting worker threads and terminating the process when all worker threads are terminated. The worker threads are started subordinate to the FIG. **29**A process at block **2912** using an O/S start thread interface.

A 19xx (i.e. **1902**, **1912**, **1922**, **1932**, **1942** and **1952**) process starts at block **2902** and continues to block **2904** where the parameter passed for which process name to start (i.e. take on identity of) is determined (e.g. **1952**). Thereafter, block **2906** creates a RAM semaphore (i.e. operating system term for a well performing Random Access Memory (RAM) semaphore with scope only within the process (i.e. to all threads of the process)). The local semaphore name preferably uses the process name prefix (e.g. **1952**-Sem), and is used to synchronize threads within the process. RAM semaphores perform significantly better than global system semaphores. Alternate embodiments will have process semaphore(s) created at block **1220** in advance. Thereafter, block **2908** initializes a thread counter (e.g. **1952**-Ct) to 0 for counting the number of worker threads actually started within the 19xx process (e.g. **1952**), block **2910** initializes a loop variable J to 0, and block **2912** starts a worker thread (the first one upon first encounter of block **2912** for a process) in this process (e.g. process **1902** starts worker thread FIG. **20**, . . . , process **1952** starts worker thread FIG. **26**A—see architecture **1900** description below).

Thereafter, block **2914** increments the loop variable by 1 and block **2916** checks if all prescribed worker threads have been started. Block **2916** accesses the 19xx-Max (e.g. **1952**-Max) variable from shared memory using a semaphore for determining the maximum number of threads to start in the process worker thread pool. If block **2916** determines all worker threads have been started, then processing continues to block **2918**. If block **2916** determines that not all worker threads have been started for the process of FIG. **29**A, then processing continues back to block **2912** for starting the next worker thread. Blocks **2912** through **2916** ensure the 19xx-Max (e.g. **1952**-Max) number of worker threads are started within the process of FIG. **29**A.

Block **2918** waits until all worker threads of blocks **2912** through **2916** have been started, as indicated by the worker threads themselves. Block **2918** waits until the process 19xx-Ct variable has been updated to the prescribed 19xx-Max value by the started worker threads, thereby indicating they are all up and running. When all worker threads are started (e.g. **1952**-Ct=**1952**-Max), thereafter block **2920** waits (perhaps a very long time) until the worker thread count (e.g. **1952**-Ct) has been reduced back down to 0 for indicating that all worker threads have been terminated, for example when the user gracefully powers off the MS. Block **2920** continues to block **2922** when all worker threads have been terminated. Block **2922** sets the shared memory variable for the 19xx process (e.g. **1952**-PID) to 0 using a semaphore for indicating that the 19xx (e.g. **1952**) process is disabled and no longer running. Thereafter, the 19xx process terminates at block **2924**. Waiting at blocks **2918** and **2920** are accomplished in a variety of well known methods:

- Detect signal sent to process by last started (or terminated) worker thread that thread count is now MAX (or 0); or
- Loop on checking the thread count with sleep time between checks, wherein within the loop there is a check of the current count (use RAM semaphore to access), and processing exits the loop (and block) when the count has reached the sought value; or
- Use of a semaphore for a count variable which causes the parent thread of FIG. **29**A to stay blocked prior to the count reaching its value, and causes the parent thread to become cleared (will leave wait block) when the count reaches its sought value.

Starting threads of processing in FIG. **29**A has been presented from a software perspective, but there are hardware/firmware thread embodiments which may be started appropriately to accomplish the same functionality. If the MS operating system does not have an interface for returning the PID at block **1232**, then FIG. **29**A can have a block (e.g. **2905**) used to determine its own PID for setting the 19xx-PID variable.

FIGS. **13**A through **13**C depict an illustration of data processing system wireless data transmissions over some wave spectrum. Embodiments may exist for any of the aforementioned wave spectrums, and data carried thereon may or may not be encrypted (e.g. encrypted WDR information). With reference now to FIG. **13**A, a MS, for example a DLM **200***a*, sends/broadcasts data such as a data **1302** in a manner well known to those skilled in the art, for example other character **32** processing data. When a Communications Key (CK) **1304** is embedded within data **1302**, data **1302** is considered usual communications data (e.g. protocol, voice, or any other data over conventional forward channel, reverse channel, voice data channel, data transmission channel, or any other prior art use channel) which has been altered to contain CK **1304**. Data **1302** contains a CK **1304** which can be detected, parsed, and processed when received by another MS or other data processing system in the vicinity of the MS (e.g. DLM **200***a*) as determined by the maximum range of transmission **1306**. CK **1304** permits "piggy-backing" on current transmissions to accomplish new functionality as disclosed herein. Transmission from the MS radiate out from it in all directions in a manner consistent with the wave spectrum used. The radius **1308** represents a first range of signal reception from the MS **200***a*, perhaps by another MS (not shown). The radius **1310** represents a second range of signal reception from the MS **200***a*, perhaps by another MS (not shown). The radius **1311** represents a third range of signal reception from the MS **200***a*, perhaps by another MS (not shown). The radius **1306** represents a last and maximum range of signal reception from the MS **200***a*, perhaps by another MS (not shown). MS design for maximum radius **1306** may take into account the desired maximum range versus acceptable wave spectrum exposure health risks for the user of the MS. The time of transmission from MS **200***a* to radius **1308** is less than times of transmission from MS **200***a* to radiuses **1310**, **1311**, or **1306**. The time of transmission from MS **200***a* to radius **1310** is less than times of transmission from MS **200***a* to radiuses **1311** or **1306**. The time of transmission from MS **200***a* to radius **1311** is less than time of transmission from MS **200***a* to radius **1306**.

In another embodiment, data **1302** contains a Communications Key (CK) **1304** because data **1302** is new transmitted data in accordance with the present disclosure. Data **1302** purpose is for carrying CK **1304** information for being detected, parsed, and processed when received by another MS or other data processing system in the vicinity of the MS (e.g. DLM **200***a*) as determined by the maximum range of transmission **1306**.

With reference now to FIG. **13**B, a MS, for example an ILM **1000***k*, sends/broadcasts data such as a data **1302** in a manner well known to those skilled in the art. Data **1302** and CK **1304** are as described above for FIG. **13**A. Data **1302** or CK **1304** can be detected, parsed, and processed when received by another MS or other data processing system in the vicinity of the MS (e.g. ILM **1000***k*) as determined by the maximum range of transmission **1306**. Transmission from the MS radiate out from it in all directions in a manner consistent with the wave spectrum used, and as described above for FIG. **13**A.

With reference now to FIG. **13**C, a service or set of services sends/broadcasts data such as a data packet **1312** in a manner well known to those skilled in the art, for example to service other character **32** processing. When a Communications Key (CK) **1314** is embedded within data **1312**, data **1312** is considered usual communications data (e.g. protocol, voice, or any other data over conventional forward channel, reverse channel, voice data channel, data transmission channel, or any other prior art use channel) which has been altered to contain CK **1314**. Data **1312** contains a CK **1314** which can be detected, parsed, and processed when received by an MS or other data processing system in the vicinity of the service(s) as determined by the maximum range of transmission **1316**. CK **1314** permits "piggy-backing" on current transmissions to accomplish new functionality as disclosed herein. Transmissions radiate out in all directions in a manner consistent with the wave spectrum used, and data carried thereon may or may not be encrypted (e.g. encrypted WDR information). The radius **1318** represents a first range of signal reception from the service (e.g. antenna thereof), perhaps by a MS (not shown). The radius **1320** represents a second range of signal reception from the service (e.g. antenna thereof), perhaps by a MS (not shown). The radius **1322** represents a third range of signal reception from the service (e.g. antenna thereof), perhaps by a MS (not shown). The radius **1316** represents a last and maximum range of signal reception from the service (e.g. antenna thereof), perhaps by a MS (not shown). The time of transmission from service to radius **1318** is less than times of transmission from service to radiuses **1320**, **1322**, or **1316**. The time of transmission from service to radius **1320** is less than times of transmission from service to radiuses **1322** or **1316**. The time of transmission from service to radius **1322** is less than time of transmission from service to radius **1316**. In another embodiment, data **1312** contains a Communications Key (CK) **1314** because data **1312** is new transmitted data in accordance with the present disclosure. Data **1312** purpose is for carrying CK **1314** information for being detected, parsed, and processed when received by another MS or data processing system in the vicinity of the service(s) as determined by the maximum range of transmission.

In some embodiments, data **1302** and **1312** are prior art wireless data transmission packets with the exception of embedding a detectable CK **1304** and/or CK **1314**, respectively. Usual data communications of MSs are altered to additionally contain the CK so data processing systems in the vicinity can detect, parse, and process the CK. Appropriate send and/or broadcast channel processing is used. In other embodiments, data **1302** and **1312** are new broadcast wireless data transmission packets for containing CK **1304** and CK **1314**, respectively. A MS may use send queue **24** for sending/broadcasting packets to data processing systems in the vicinity, and may use the receive queue **26** for receiving packets from other data processing systems in the vicinity. Contents of CKs (Communications Keys) depend on which LBX features are in use and the functionality intended.

In the case of "piggybacking" on usual communications, receive queue **26** insertion processing simply listens for the usual data and when detecting CK presence, inserts CK information appropriately to queue **26** for subsequent processing. Also in the case of "piggybacking" on usual communications, send queue **24** retrieval processing simply retrieves CK information from the queue and embeds it in an outgoing data **1302** at first opportunity. In the case of new data communications, receive queue **26** insertion processing simply listens for the new data containing CK information, and inserts CK information appropriately to queue **26** for subsequent processing. Also in the case of new data communications, send queue **24** retrieval processing simply retrieves CK information from the queue and transmits CK information as new data.

LBX: LN-EXPANSE Configuration

FIG. **14**A depicts a flowchart for describing a preferred embodiment of MS LBX configuration processing. FIG. **14** is of Self Management Processing code **18**. MS LBX configuration begins at block **1402** upon user action to start the user interface and continues to block **1404** where user interface objects are initialized for configurations described below with current settings that are reasonable for display to available user interface real estate. Thereafter, applicable settings are presented to the user at block **1406** with options. Block **1406** preferably presents to the user at least whether or not DLM capability is enabled (i.e. MS to behave as a DLM=at least one role of DLMV enabled), whether or not ILM capability is enabled (i.e. MS to behave as an ILM=at least one role of ILMV enabled), and/or whether or not this MS should participate in the LN-expanse as a source location for other MSs (e.g. process **1902** and/or **1942** enabled). Alternative embodiments will further present more or less information for each of the settings, or present information associated with other FIG. **14** blocks of processing. Other embodiments will not configure DLM settings for an MS lacking DLM capability (or when all DLMV roles disabled). Other embodiments will not configure ILM settings when DLM capability is present. Block **1406** continues to block **1408** where processing waits for user action in response to options. Block **1408** continues to block **1410** when a user action is detected. If block **1410** determines the user selected to configure DLM capability (i.e. DLMV role(s)), then the user configures DLM role(s) at block **1412** and processing continues back to block **1406**. Block **1412** processing is described by FIG. **15**A. If block **1410** determines the user did not select to configure DLM capability (i.e. DLMV role(s)), then processing continues to block **1414**. If block **1414** determines the user selected to configure ILM capability (i.e. ILMV role(s)), then the user configures ILM role(s) at block **1416** and processing continues back to block **1406**. Block **1416** processing is described by FIG. **15**B. If block **1414** determines the user did not select to configure ILM capability (i.e. ILMV role(s)), then processing continues to block **1418**. If block **1418** determines the user selected to configure NTP use, then the user configures NTP use at block **1420** and processing continues back to block **1406**. Block **1420** processing is described by FIG. **16**. If block **1418** determines the user did not select to configure NTP use, then processing continues to block **1422**.

If block **1422** determines the user selected to maintain the WDR queue, then the user maintains WDRs at block **1424** and processing continues back to block **1406**. Block **1424** processing is described by FIG. **17**. Blocks **1412**, **1416**, **1420** and **1424** are understood to be delimited by appropriate semaphore control to avoid multi-threaded access problems. If block **1422** determines the user did not select to maintain the WDR queue, then processing continues to block **1426**. If block **1426** determines the user selected to configure the confidence floor value, then block **1428** prepares parameters for invoking a Configure Value procedure (parameters for reference (address) of value to configure; and validity criteria of value to configure), and the Configure Value procedure of FIG. **18** is invoked at block **1430** with the two (2) parameters. Thereafter, processing continues back to block **1406**. Blocks **1428** and **1430** are understood to be delimited by appropriate semaphore control when modifying the confidence floor value since other threads can access the floor value.

The confidence floor value is the minimum acceptable confidence value of any field **1100*d*** (for example as checked by block **276**). No WDR with a field **1100*d*** less than the confidence floor value should be used to describe MS whereabouts. In an alternative embodiment, the confidence floor value is enforced as the same value across an LN-expanse with no user control to modify it. One embodiment of FIG. **14** does not permit user control over a minimum acceptable confidence floor value. Various embodiments will default the floor value. Block **1812** enforces an appropriate value in accordance with the confidence value range implemented (e.g. value from 1 to 100). Since the confidence of whereabouts is likely dependent on applications in use at the MS, the preferred embodiment is to permit user configuration of the acceptable whereabouts confidence for the MS. A new confidence floor value can be put to use at next thread(s) startup, or can be used instantly with the modification made, depending on the embodiment. The confidence floor value can be used to filter out WDRs prior to inserting to queue **22**, filter out WDRs when retrieving from queue **22**, filter out WDR information when listening on channel(s) prior to inserting to queue **26**, and/or used in accessing queue **22** for any reason (depending on embodiments). While confidence is validated on both inserts and queries (retrievals/peeks), one or the other validation is fine (preferably on inserts). It is preferred that executable code incorporate checks where applicable since the confidence floor value can be changed after queue **22** is in use. Also, various present disclosure embodiments may maintain all confidences to queue **22**, or a particular set of acceptable confidences.

If block **1426** determines the user did not select to configure the confidence floor value, then processing continues to block **1432**. If block **1432** determines the user selected to configure the Whereabouts Timeliness Variable (WTV), then block **1434** prepares parameters for invoking the Configure Value procedure (parameters for reference (address) of value to configure; and validity criteria of value to configure), and the Configure Value procedure of FIG. **18** is invoked at block **1430** with the two (2) parameters. Thereafter, processing continues back to block **1406**. Blocks **1434** and **1430** are understood to be delimited by appropriate semaphore control when modifying the WTV since other threads can access the WTV.

A critical configuration for MS whereabouts processing is whereabouts timeliness. Whereabouts timeliness is how often (how timely) an MS should have accurate whereabouts. Whereabouts timeliness is dependent on how often the MS is updated with whereabouts information, what technologies are available or are in the vicinity, how capable the MS is of maintaining whereabouts, processing speed(s), transmission speed(s), known MS or LN-expanse design constraints, and perhaps other factors. In some embodiments, whereabouts timeliness is as soon as possible. That is, MS whereabouts is updated whenever possible as often as possible. In fact, the present disclosure provides an excellent system and methodology to accomplish that by leveraging location technologies whenever and wherever possible. However, there should be balance when considering less capable processing of a MS to prevent hogging CPU cycles from other applications at the MS. In other embodiments, a hard-coded or preconfigured time interval is used for keeping an MS informed of its whereabouts in a timely manner. For example, the MS should know its own whereabouts at least every second, or at least every 5 seconds, or at least every minute, etc. Whereabouts timeliness is critical depending on the applications in use at the MS. For example, if MS whereabouts is updated once at the MS every 5 minutes during high speeds of travel when using navigation, the user has a high risk of missing a turn during travel in downtown cities where timely decisions for turns are required. On the other hand, if MS whereabouts is updated every 5 seconds, and an application only requires an update accuracy to once per minute, then the MS may be excessively processing.

In some embodiments, there is a Whereabouts Timeliness Variable (WTV) configured at the MS (blocks **1432**, **1434**, **1430**). Whether it is user configured, system configured, or preset in a system, the WTV is used to:

- Define the maximum period of time for MS whereabouts to become stale at any particular time;
- Cause the MS to seek its whereabouts if whereabouts information is not up to date in accordance with the WTV; and
- Prevent keeping the MS too busy with keeping abreast of its own whereabouts.

In another embodiment, the WTV is automatically adjusted based on successes or failures of automatically locating the MS. As the MS successfully maintains timely whereabouts, the WTV is maintained consistent with the user configured, system configured, or preset value, or in accordance with active applications in use at the time. However, as the MS fails in maintaining timely whereabouts, the WTV is automatically adjusted (e.g. to longer periods of time to prevent unnecessary wasting of power and/or CPU resources). Later, as whereabouts become readily available, the WTV can be automatically adjusted back to the optimal value. In an emergency situation, the user always has the ability to force the MS to determine its own whereabouts anyway (Blocks **856** and **862** through **878**, in light of a WDR request and WDR response described for architecture **1900**). In embodiments where the WTV is adjusted in accordance with applications in use at the time, the most demanding requirement of any application started is maintained to the WTV. Preferably, each application of the MS initializes to an API of the MS with a parameter of its WTV requirements. If the requirement is more timely than the current value, then the more timely value is used. The WTV can be put to use at next thread(s) startup, or can be used instantly with the modification made, depending on the embodiment.

If block **1432** determines the user did not select to configure the WTV, then processing continues to block **1436**. If block **1436** determines the user selected to configure the maximum number of threads in a 19xx process (see 19xx-Max variable in FIG. **19** discussions), then block **1438** interfaces with the user until a valid 19xx-max variable is selected, and processing continues to block **1440**. If block **1440** determines the 19xx process is already running (i.e. 19xx-PID>0 implies it is enabled), then an error is provided to the user at block **1442**, and processing continues back to block **1406**. Preferably, block **1442** does not continue back to block **1406** until the user acknowledges the error (e.g. with a user action). If block **1440** determines the user selected 19xx process (process **1902**, process **1912**, process **1922**, process **1932**, process **1942**, or process **1952**) is not already running (i.e. 19xx-PID=0 implies it is disabled), then block **1444** prepares parameters for invoking the Configure Value procedure (parameters for reference (address) of 19xx-Max value to configure; and validity criteria of value to configure), and the Configure Value procedure of FIG. **18** is invoked at block **1430** with the two (2) parameters. Thereafter, processing continues back to block **1406**. Blocks **1438**, **1440**, **1444** and **1430** are understood to be delimited by appropriate semaphore control when modifying the 19xx-Max value since other threads can access it. The 19xx-Max value should not be modified while the 19xx process is running because the number of threads to terminate may be changed prior to terminating. An alternate embodiment of modifying a process number of threads will dynamically modify the number of threads in anticipation of required processing.

If block **1436** determines the user did not select to configure a process thread maximum (19xx-Max), then block **1446** checks if the user selected to (toggle) disable or enable a particular process (i.e. a 19xx process of FIG. **19**). If block **1446** determines the user did select to toggle enabling/disabling a particular FIG. **19** process, then block **1448** interfaces with the user until a valid 19xx process name is selected, and processing continues to block **1450**. If block **1450** determines the 19xx process is already running (i.e. 19xx-PID>0 implies it is enabled), then block **1454** prepares parameters (just as does block **2812**). Thereafter, block **1456** invokes FIG. **29**B processing (just as does block **2814**). Processing then continues back to block **1406**. If block **1450** determines the 19xx process is not running (i.e. 19xx-PID=0 implies it is disabled), then block **1452** invokes FIG. **29**A processing (just as does block **1232**). Processing then continues back to block **1406**. Block **1456** does not continue back to block **1406** until the process is completely terminated. Blocks **1448**, **1450**, **1452**, **1454** and **1456** are understood to be delimited by appropriate semaphore control.

Preferred embodiments of blocks **1446** and **1448** use convenient names of processes being started or terminated, rather than convenient brief process names such as **1902**, **1912**, **1922**, **1932**, **1942**, or **1952** used in flowcharts. In some embodiments, the long readable name is used, such as whereabouts broadcast process (**1902**), whereabouts collection process (**1912**), whereabouts supervisor process (**1922**), timing determination process (**1932**), WDR request process (**1942**), and whereabouts determination process (**1952**). For example, the user may know that the whereabouts supervisor process enabled/disabled indicates whether or not to have whereabouts timeliness monitored in real time. Enabling the whereabouts supervisor process enables monitoring for the WTV in real time, and disabling the whereabouts supervisor process disables monitoring the WTV in real time.

In another embodiment of blocks **1446** and **1448**, a completely new name or description may be provided to any of the processes to facilitate user interface usability. For example, a new name Peer Location Source Variable (PLSV) can be associated to the whereabouts broadcast process **1902** and/or **1942**. PLSV may be easier to remember. If the PLSV was toggled to disabled, the whereabouts broadcast process **1902** and/or **1942** terminates. If the PLSV was toggled to enabled, the whereabouts broadcast process **1902** and/or **1942** is started. It may be easier to remember that the PLSV enables/disables whether or not to allow this MS to be a location source for other MSs in an LN-expanse.

In other embodiments, a useful name (e.g. PLSV) represents starting and terminating any subset of 19xx processes (a plurality (e.g. **1902** and **1942**)) for simplicity. In yet other embodiments, FIGS. **14**A/**14**B can be used to start or terminate worker thread(s) in any process, for example to throttle up more worker threads in a process, or to throttle down for less worker threads in a process, perhaps modifying thread instances to accommodate the number of channels for communications, or for the desired performance. There are many embodiments for fine tuning the architecture **1900** for optimal peer to peer interaction. In yet other embodiments, toggling may not be used. There may be individual options available at block **1408** for setting any data of this disclosure. Similarly, the 19xx-Max variables may be modified via individual user friendly names and/or as a group of 19xx-Max variables.

Referring back to block **1446**, if it is determined the user did not select to toggle for enabling/disabling process(es), then processing continues to block **1458**. If block **1458** determines the user selected to exit FIGS. **14**A/**14**B configuration processing, then block **1460** terminates the user interface appropriately and processing terminates at block **1462**. If block **1458** determines the user did not select to exit the user interface, then processing continues to block **1466** of FIG. **14**B by way of off page connector **1464**.

With reference now to FIG. **14**B, depicted is a continued portion flowchart of FIG. **14**A for describing a preferred embodiment of MS LBX configuration processing. If block **1466** determines the user selected to configure the Source Periodicity Time Period (SPTP) value, then block **1468** prepares parameters for invoking the Configure Value procedure (parameters for reference (address) of value to configure; and validity criteria of value to configure), and the Configure Value procedure of FIG. **18** is invoked at block **1470** with the two (2) parameters. Thereafter, processing continues back to block **1406** by way of off page connector **1498**. Blocks **1468** and **1470** are understood to be delimited by appropriate semaphore control when modifying the SPTP value since other threads can access it. The SPTP configures the time period between broadcasts by thread(s) **1902**, for example 5 seconds. Some embodiments do not permit configuration of the SPTP.

If block **1466** determines the user did not select to configure the SPTP value, then processing continues to block **1472**. If block **1472** determines the user selected to configure service propagation, then the user configures service propagation at block **1474** and processing continues back to block **1406** by way of off page connector **1498**. If block **1472** determines the user did not select to configure service propagation, then processing continues to block **1476**.

If block **1476** determines the user selected to configure permissions **10**, then the user configures permissions at block **1478** and processing continues back to block **1406** by way of off page connector **1498**. If block **1476** determines the user did not select to configure permissions **10**, then processing continues to block **1480**. If block **1480** determines the user selected to configure charters **12**, then the user configures charters **12** at block **1482** and processing continues back to block **1406** by way of off page connector **1498**. If block **1480** determines the user did not select to configure charters **12**, then processing continues to block **1484**. If block **1484** determines the user selected to configure statistics **14**, then the user configures statistics **14** at block **1486** and processing continues back to block **1406** by way of off page connector **1498**. If block **1484** determines the user did not select to configure statistics **14**, then processing continues to block **1488**. If block **1488** determines the user selected to configure service informant code **28**, then the user configures code **28** at block **1490** and processing continues back to block **1406** by way of off page connector **1498**. If block **1488** determines the user did not select to configure code **28**, then processing continues to block **1492**. If block **1492** determines the user selected to maintain LBX history **30**, then the user maintains LBX history at block **1494** and processing continues back to block **1406** by way of off page connector **1498**. If block **1492** determines the user did not select to maintain LBX history **30**, then processing continues to block **1496**.

Block **1496** handles other user interface actions leaving block **1408**, and processing continues back to block **1406** by way of off page connector **1498**.

Details of blocks **1474**, **1478**, **1482**, **1486**, **1490**, **1494**, and perhaps more detail to block **1496**, are described with other flowcharts. Appropriate semaphores are requested at the beginning of block processing, and released at the end of block processing, for thread safe access to applicable data at risk of being accessed by another thread of processing at the same time of configuration. In some embodiments, a user/administrator with secure privileges to the MS has ability to perform any subset of configurations of FIGS. **14**A and **14**B processing, while a general user may not. Any subset of FIG. **14** configuration may appear in alternative embodiments, with or without authenticated administrator access to perform configuration.

FIG. **15**A depicts a flowchart for describing a preferred embodiment of DLM role configuration processing of block **1412**. Processing begins at block **1502** and continues to block **1504** which accesses current DLMV settings before continuing to block **1506**. If there were no DLMV entries (list empty) as determined by block **1506**, then block **1508** provides an error to the user and processing terminates at block **1518**. The DLMV may be empty when the MS has no local DLM capability and there hasn't yet been any detected DLM capability, for example as evidenced by WDRs inserted to queue **22**. Preferably, the error presented at block **1508** requires the user to acknowledge the error (e.g. with a user action) before block **1508** continues to block **1518**. If block **1506** determines at least one entry (role) is present in the DLMV, then the current DLMV setting(s) are saved at block **1510**, the manage list processing procedure of FIG. **15**C is invoked at block **1512** with the DLMV as a reference (address) parameter, and processing continues to block **1514**.

Block **1514** determines if there were any changes to the DLMV from FIG. **15**C processing by comparing the DLMV after block **1512** with the DLMV saved at block **1510**. If there were changes via FIG. **15**C processing, such as a role which was enabled prior to block **1512** which is now disabled, or such as a role which was disabled prior to block **1512** which is now enabled, then block **1514** continues to block **1516** which handles the DLMV changes appropriately. Block **1516** continues to block **1518** which terminates FIG. **15**A processing. If block **1514** determines there were no changes via block **1512**, then processing terminates at block **1518**.

Block **1516** enables newly enabled role(s) as does block **1238** described for FIG. **12**. Block **1516** disables newly disabled role(s) as does block **2804** described for FIG. **28**.

FIG. **15**B depicts a flowchart for describing a preferred embodiment of ILM role configuration processing of block **1416**. Processing begins at block **1522** and continues to block **1524** which accesses current ILMV settings before continuing to block **1526**. If there were no ILMV entries (list empty) as determined by block **1526**, then block **1528** provides an error to the user and processing terminates at block **1538**. The ILMV may be empty when the MS is not meant to have ILM capability. Preferably, the error presented at block **1528** requires the user to acknowledge the error before block **1528** continues to block **1538**. If block **1526** determines at least one entry (role) is present in the ILMV, then the current ILMV setting(s) are saved at block **1530**, the manage list processing procedure of FIG. **15**C is invoked with a reference (address) parameter of the ILMV at block **1532**, and processing continues to block **1534**.

Block **1534** determines if there were any changes to the ILMV from FIG. **15**C processing by comparing the ILMV after block **1532** with the ILMV saved at block **1530**. If there were changes via FIG. **15**C processing, such as a role which was enabled prior to block **1532** which is now disabled, or such as a role which was disabled prior to block **1532** which is now enabled, then block **1534** continues to block **1536** which handles the ILMV changes appropriately. Block **1536** continues to block **1538** which terminates FIG. **15**B processing. If block **1534** determines there were no changes via block **1532**, then processing terminates at block **1538**.

Block **1536** enables newly enabled role(s) as does blocks **1224** through **1234** described for FIG. **12**. Block **1536** disables newly disabled role(s) as does blocks **2806** through **2816** described for FIG. **28**.

FIG. **15**C depicts a flowchart for describing a preferred embodiment of a procedure for Manage List processing. Processing starts at block **1552** and continues to block **1554**. Block **1554** presents the list (DLM capability if arrived to by way of FIG. **15**A; ILM capability if arrived to by way of FIG. **15**B) to the user, as passed to FIG. **15**C processing with the reference parameter by the invoker, with which list items are marked (enabled) and which are unmarked (disabled) along with options, before continuing to block **1556** for awaiting user action. Block **1554** highlights currently enabled roles, and ensures disabled roles are not highlighted in the presented list. When a user action is detected at block **1556**, thereafter, block **1558** checks if a list entry was enabled (marked) by the user, in which case block **1560** marks the list item as enabled, saves it to the list (e.g. DLMV or ILMV), and processing continues back to block **1554** to refresh the list interface. If block **1558** determines the user did not respond with an enable action, then block **1562** checks for a disable action. If block **1562** determines the user wanted to disable a list entry, then block **1564** marks (actually unmarks it) the list item as disabled, saves it to the list (e.g. DLMV or ILMV), and processing continues back to block **1554**. If block **1562** determines the user did not want to disable a list item, then block **1566** checks if the user wanted to exit FIG. **15**C processing. If block **1566** determines the user did not select to exit list processing, then processing continues to block **1568** where other user interface actions are appropriately handled and then processing continues back to block **1554**. If block **1566** determines the user did select to exit manage list processing, then FIG. **15**C processing appropriately returns to the caller at block **1570**.

FIG. **15**C interfaces with the user for desired DLMV (via FIG. **15**A) or ILMV (via FIG. **15**B) configurations. In some embodiments, it makes sense to have user control over enabling or disabling DLM and/or ILM capability (roles) to the MS, for example for software or hardware testing.

FIG. **16** depicts a flowchart for describing a preferred embodiment of NTP use configuration processing of block **1420**. Processing starts at block **1602** and continues to block **1604** where the current NTP use setting is accessed. Thereafter, block **1606** presents the current NTP use setting to its value of enabled or disabled along with options, before continuing to block **1608** for awaiting user action. When a user action is detected at block **1608**, block **1610** checks if the NTP use setting was disabled at block **1608**, in which case block **1612** terminates NTP use appropriately, block **1614** sets (and saves) the NTP use setting to disabled, and processing continues back to block **1606** to refresh the interface. Block **1612** disables NTP as does block **2828**.

If block **1610** determines the user did not respond for disabling NTP, then block **1616** checks for a toggle to being enabled. If block **1616** determines the user wanted to enable NTP use, then block **1618** accesses known NTP server address(es) (e.g. ip addresses preconfigured to the MS, or set with another user interface at the MS), and pings each one, if necessary, at block **1620** with a timeout. As soon as one NTP server is determined to be reachable, block **1620** continues to block **1622**. If no NTP server was reachable, then the timeout will have expired for each one tried at block **1620** for continuing to block **1622**. Block **1622** determines if at least one NTP server was reachable at block **1620**. If block **1622** determines no NTP server was reachable, then an error is presented to the user at block **1624** and processing continues back to block **1606**. Preferably, the error presented at block **1624** requires the user to acknowledge the error before block **1624** continues to block **1606**. If block **1622** determines that at least one NTP server was reachable, then block **1626** initializes NTP use appropriately, block **1628** sets the NTP use setting to enabled (and saves), and processing continues back to block **1606**. Block **1626** enables NTP as does block **1210**.

Referring back to block **1616**, if it is determined the user did not want to enable NTP use, then processing continues to block **1630** where it is checked if the user wanted to exit FIG. **16** processing. If block **1630** determines the user did not select to exit FIG. **16** processing, then processing continues to block **1632** where other user interface actions leaving block **1608** are appropriately handled, and then processing continues back to block **1606**. If block **1630** determines the user did select to exit processing, then FIG. **16** processing terminates at block **1634**.

FIG. **17** depicts a flowchart for describing a preferred embodiment of WDR maintenance processing of block **1424**. Processing starts at block **1702** and continues to block **1704** where it is determined if there are any WDRs of queue **22**. If block **1704** determines there are no WDRs for processing, then block **1706** presents an error to the user and processing continues to block **1732** where FIG. **17** processing terminates. Preferably, the error presented at block **1706** requires the user to acknowledge the error before block **1706** continues to block **1732**. If block **1704** determines there is at least one WDR, then processing continues to block **1708** where the current contents of WDR queue **22** is appropriately presented to the user (in a scrollable list if necessary). Thereafter, block **1710** awaits user action. When a user action is detected at block **1710**, block **1712** checks if the user selected to delete a WDR from queue **22**, in which case block **1714** discards the selected WDR, and processing continues back to block **1708** for a refreshed presentation of queue **22**. If block **1712** determines the user did not select to delete a WDR, then block **1716** checks if the user selected to modify a WDR. If block **1716** determines the user wanted to modify a WDR of queue **22**, then block **1718** interfaces with the user for validated WDR changes before continuing back to block **1708**. If block **1716** determines the user did not select to modify a WDR, then block **1720** checks if the user selected to add a WDR to queue **22**. If block **1720** determines the user selected to add a WDR (for example, to manually configure MS whereabouts), then block **1722** interfaces with the user for a validated WDR to add to queue **22** before continuing back to block **1708**. If block **1720** determines the user did not select to add a WDR, then block **1724** checks if the user selected to view detailed contents of a WDR, perhaps because WDRs are presented in an abbreviated form at block **1708**. If it is determined at block **1724** the user did select to view details of a WDR, then block **1726** formats the WDR in detail form, presents it to the user, and waits for the user to exit the view of the WDR before continuing back to block **1708**. If block **1724** determines the user did not select to view a WDR in detail, then block **1728** checks if the user wanted to exit FIG. **17** processing. If block **1728** determines the user did not select to exit FIG. **17** processing, then processing continues to block **1730** where other user interface actions leaving block **1710** are appropriately handled, and then processing continues back to block **1708**. If block **1728** determines the user did select to exit processing, then FIG. **17** processing terminates at block **1732**.

There are many embodiments for maintaining WDRs of queue **22**. In some embodiments, FIG. **17** (i.e. block **1424**) processing is only provided for debug of an MS. In a single instance WDR embodiment, block **1708** presents the one and only WDR which is used to keep current MS whereabouts whenever possible. Other embodiments incorporate any subset of FIG. **17** processing.

FIG. **18** depicts a flowchart for describing a preferred embodiment of a procedure for variable configuration processing, namely the Configure Value procedure, for example for processing of block **1430**. Processing starts at block **1802** and continues to block **1804** where parameters passed by the invoker of FIG. **18** are determined, namely the reference (address) of the value for configuration to be modified, and the validity criteria for what makes the value valid. Passing the value by reference simply means that FIG. **18** has the ability to directly change the value, regardless of where it is located. In some embodiments, the parameter is an address to a memory location for the value. In another embodiment, the value is maintained in a database or some persistent storage, and FIG. **18** is passed enough information to know how to permanently affect/change the value.

Block **1804** continues to block **1806** where the current value passed is presented to the user (e.g. confidence floor value), and then to block **1808** for awaiting user action. When a user action is detected at block **1808**, block **1810** checks if the user selected to modify the value, in which case block **1812** interfaces with the user for a validated value using the validity criteria parameter before continuing back to block **1806**. Validity criteria may take the form of a value range, value type, set of allowable values, or any other criteria for what makes the value a valid one.

If block **1810** determines the user did not select to modify the value, then block **1814** checks if the user wanted to exit FIG. **18** processing. If block **1814** determines the user did not select to exit FIG. **18** processing, then processing continues to block **1816** where other user interface actions leaving block **1808** are appropriately handled, and then processing continues back to block **1806**. If block **1814** determines the user did select to exit processing, then FIG. **18** processing appropriately returns to the caller at block **1818**.

LBX: LN-EXPANSE Interoperability

FIG. **19** depicts an illustration for describing a preferred embodiment multithreaded architecture of peer interaction processing of a MS in accordance with the present disclosure. MS architecture **1900** preferably includes a set of Operating System (O/S) processes (i.e. O/S terminology "process" with O/S terminology "thread" or "threads (i.e. thread(s))), including a whereabouts broadcast process **1902**, a whereabouts collection process **1912**, a whereabouts supervisor process **1922**, a timing determination process **1932**, a WDR request process **1942**, and a whereabouts determination process **1952**. Further included are queues for interaction of processing, and process associated variables to facilitate processing. All of the FIG. **19** processes are of PIP code **6**. There is preferably a plurality (pool) of worker threads within each of said 19xx processes (i.e. **1902**, **1912**, **1922**, **1932**, **1942** and **1952**) for high performance asynchronous processing. Each 19xx process (i.e. **1902**, **1912**, **1922**, **1932**, **1942** and **1952**) preferably has at least two (2) threads:

1) "parent thread"; and
2) "worker thread".

A parent thread (FIG. **29**A) is the main process thread for:

- starting the particular process;
- starting the correct number of worker thread(s) of that particular process;
- staying alive while all worker threads are busy processing; and

properly terminating the process when worker threads are terminated.

The parent thread is indeed the parent for governing behavior of threads at the process whole level. Every process has a name for convenient reference, such as the names **1902**, **1912**, **1922**, **1932**, **1942** and **1952**. Of course, these names may take on the associated human readable forms of whereabouts broadcast process, whereabouts collection process, whereabouts supervisor process, timing determination process, WDR request process, and whereabouts determination process, respectively. For brevity, the names used herein are by the process label of FIG. **19** in a form 19xx. There must be at least one worker thread in a process. Worker thread(s) are described with a flowchart as follows:

1902—FIG. **20**;
1912—FIG. **21**;
1922—FIG. **22**;
1932—FIG. **23**;
1942—FIG. **25**; and
1952—FIG. **26**A.

Threads of architecture MS are presented from a software perspective, but there are applicable hardware/firmware process thread embodiments accomplished for the same functionality. In fact, hardware/firmware embodiments are preferred when it is known that processing is mature (i.e. stable) to provide the fastest possible performance. Architecture **1900** processing is best achieved at the highest possible performance speeds for optimal wireless communications processing. There are two (2) types of processes for describing the types of worker threads:

1) "Slave to Queue"; and
2) "Slave to Timer".

A 19xx process is a slave to queue process when its worker thread(s) are driven by feeding from a queue of architecture **1900**. A slave to queue process stays "blocked" (O/S terminology "blocked"=preempted) on a queue entry retrieval interface until the sought queue item is inserted to the queue. The queue entry retrieval interface becomes "cleared" (O/S terminology "cleared"=clear to run) when the sought queue entry is retrieved from the queue by a thread. These terms (blocked and cleared) are analogous to a semaphore causing a thread to be blocked, and a thread to be cleared, as is well known in the art. Queues have semaphore control to ensure no more than one thread becomes clear at a time for a single queue entry retrieved (as done in an O/S). One thread sees a particular queue entry, but many threads can feed off the same queue to do the same work concurrently. Slave to queue type of processes are **1912**, **1932**, **1942** and **1952**. A slave to queue process is properly terminated by inserting a special termination queue entry for each worker thread to terminate itself after queue entry retrieval.

A 19xx process is a slave to timer process when its worker thread(s) are driven by a timer for peeking a queue of architecture **1900**. A timer provides the period of time for a worker thread to sleep during a looped iteration of checking a queue for a sought entry (without removing the entry from the queue). Slave to timer threads periodically peek a queue, and based on what is found, will process appropriately. A queue peek does not alter the peeked queue. The queue peek interface is semaphore protected for preventing peeking at an un-opportune time (e.g. while thread inserting or retrieving from queue). Queue interfaces ensure one thread is acting on a queue with a queue interface at any particular time. Slave to timer type of processes are **1902** and **1922**. A slave to timer process is properly terminated by inserting a special termination queue entry for each worker thread to terminate itself by queue entry peek.

Block **2812** knows the type of 19xx process for preparing the process type parameter for invocation of FIG. **29**B at block **2814**. The type of process has slightly different termination requirements because of the worker thread(s) processing type. Alternate embodiments of slave to timer processes will make them slave to queue processes by simply feeding off Thread Request (TR) queue **1980** for driving a worker thread when to execute (and when to terminate). New timer(s) would insert timely queue entries to queue **1980**, and processes **1902** and **1922** would retrieve from the queue (FIG. **24**A record **2400**). The queue entries would become available to queue **1980** when it is time for a particular worker thread to execute. Worker threads of processes **1902** and **1922** could retrieve, and stay blocked on, queue **1980** until an entry was inserted by a timer for enabling a worker thread (field **2400***a* set to **1902** or **1912**). TR queue **1980** is useful for starting any threads of architecture **1900** in a slave to queue manner. This may be a cleaner architecture for all thread pools to operate the same way (slave to queue). Nevertheless, the two thread pool methods are implemented.

Each 19xx process has at least four (4) variables for describing present disclosure processing:

19xx-PID=The O/S terminology "Process Identifier (PID)" for the O/S PID of the 19xx process. This variable is also used to determine if the process is enabled (PID>0), or is disabled (PID=0 (i.e. <=0));

19xx-Max=The configured number of worker thread(s) for the 19xx process;

19xx-Sem=A process local semaphore for synchronizing 19xx worker threads, for example in properly starting up worker threads in process 19xx, and for properly terminating worker threads in process 19xx; and

19xx-Ct=A process local count of the number of worker thread(s) currently running in the 19xx process.

19xx-PID and 19xx-Max are variables of PIP data **8**. 19xx-Sem and 19xx-Ct are preferably process 19xx stack variables within the context of PIP code **6**. 19xx-PID is a semaphore protected global variable in architecture **1900** so that it can be used to determine whether or not a particular 19xx process is enabled (i.e. running) or disabled (not running). 19xx-Max is a semaphore protected global variable in architecture **1900** so that user configuration processing outside of architecture **1900** can be used to administrate a desired number of worker threads for a 19xx process. Alternate embodiments will not provide user configuration of 19xx-Max variables (e.g. hard coded maximum number of threads), in which case no 19xx-Max global variable is necessary. "Thread(s) 19xx" is a brief form of stating "worker thread(s) of the 19xx process".

Receive (Rx) queue **26** is for receiving CK **1304** or CK **1314** data (e.g. WDR or WDR requests), for example from wireless transmissions. Queue **26** will receive at least WDR information (destined for threads **1912**) and WDR requests (FIG. **24**C records **2490** destined for threads **1942**). At least one thread (not shown) is responsible for listening on appropriate channel(s) and immediately depositing appropriate records to queue **26** so that they can be processed by architecture **1900**. Preferably, there is a plurality (pool) of threads for feeding queue **26** based on channel(s) being listened on, and data **1302** or **1312** anticipated for being received. Alternative embodiments of thread(s) **1912** may themselves directly be listening on appropriate channels and immediately processing packets identified, in lieu of a queue **26**. Alternative embodiments of thread(s) **1942** may themselves directly be listening on appropriate channels and immediately processing packets identified, in lieu of a queue **26**. Queue **26** is preferred to isolate channel(s) (e.g. frequency(s)) and transmission reception processing in well known modular (e.g.

Radio Frequency (RF)) componentry, while providing a high performance queue interface to other asynchronous threads of architecture **1900** (e.g. thread(s) of process **1912**). Wave spectrums (via particular communications interface **70**) are appropriately processed for feeding queue **26**. As soon as a record is received by an MS, it is assumed ready for processing at queue **26**. All queue **26** accesses are assumed to have appropriate semaphore control to ensure synchronous access by any thread at any particular time to prevent data corruption and misuse. Queue entries inserted to queue **26** may have arrived on different channel(s), and in such embodiments a channel qualifier may further direct queue entries from queue **26** to a particular thread **1912** or **1942** (e.g. thread(s) dedicated to channel(s)). In other embodiments, receive processing feeds queue **26** independent of any particular channel(s) monitored, or received on (the preferred embodiment described). Regardless of how data is received and then immediately placed on queue **26**, a received date/time stamp (e.g. fields **1100***p* or **2490***c*) is added to the applicable record for communicating the received date/time stamp to a thread (e.g. thread(s) **1912** or **1942**) of when the data was received. Therefore, the queue **26** insert interface tells the waiting thread(s) when the data was actually received. This ensures a most accurate received date/time stamp as close to receive processing as possible (e.g. enabling most accurate TDOA measurements). An alternate embodiment could determine applicable received date/time stamps in thread(s) **1912** or thread(s) **1942**. Other data placed into received WDRs are: wave spectrum and/or particular communications interface **70** of the channel received on, and heading/yaw/pitch/roll (or accelerometer readings) with AOA measurements, signal strength, and other field **1100***f* eligible data of the receiving MS. Depending on alternative embodiments, queue **26** may be viewed metaphorically for providing convenient grounds of explanation.

Send (Tx) queue **24** is for sending/communicating CK **1304** data, for example for wireless transmissions. At least one thread (not shown) is responsible for immediately transmitting (e.g. wirelessly) anything deposited to queue **24**. Preferably, there is a plurality (pool) of threads for feeding off of queue **24** based on channel(s) being transmitted on, and data **1302** anticipated for being sent. Alternative embodiments of thread(s) of processes **1902**, **1922**, **1932** and **1942** may themselves directly transmit (send/broadcast) on appropriate channels anything deposited to queue **24**, in lieu of a queue **24**. Queue **24** is preferred to isolate channel(s) (e.g. frequency(s)) and transmission processing in well known modular (e.g. RF) componentry, while providing a high performance queue interface to other asynchronous threads of architecture **1900** (e.g. thread(s) **1942**). Wave spectrums and/or particular communications interface **70** are appropriately processed for sending from queue **24**. All queue **24** accesses are assumed to have appropriate semaphore control to ensure synchronous access by any thread at any particular time to prevent data corruption and misuse. As soon as a record is inserted to queue **24**, it is assumed sent immediately. Preferably, fields sent depend on fields set. Queue entries inserted to queue **24** may contain specification for which channel(s) to send on in some embodiments. In other embodiments, send processing feeding from queue **24** has intelligence for which channel(s) to send on (the preferred embodiment described). Depending on alternative embodiments, queue **24** may be viewed metaphorically for providing convenient grounds of explanation.

When interfacing to queue **24**, the term "broadcast" refers to sending outgoing data in a manner for reaching as many MSs as possible (e.g. use all participating communications interfaces **70**), whereas the term "send" refers to targeting a particular MS or group of MSs.

WDR queue **22** preferably contains at least one WDR **1100** at any point in time, for at least describing whereabouts of the MS of architecture **1900**. Queue **22** accesses are assumed to have appropriate semaphore control to ensure synchronous access by any thread at any particular time to prevent data corruption and misuse. A single instance of data embodiment of queue **22** may require an explicit semaphore control for access. In a WDR plurality maintained to queue **22**, appropriate queue interfaces are again provided to ensure synchronous thread access (e.g. implicit semaphore control). Regardless, there is still a need for a queue **22** to maintain a plurality of WDRs from remote MSs. The preferred embodiment of all queue interfaces uses queue interface maintained semaphore(s) invisible to code making use of queue (e.g. API) interfaces. Depending on alternative embodiments, queue **22** may be viewed metaphorically for providing convenient grounds of explanation.

Thread Request (TR) queue **1980** is for requesting processing by either a timing determination (worker) thread of process **1932** (i.e. thread **1932**) or whereabouts determination (worker) thread of process **1952** (i.e. thread **1952**). When requesting processing by a thread **1932**, TR queue **1980** has requests (retrieved via processing **1934** after insertion processing **1918**) from a thread **1912** to initiate TDOA measurement. When requesting processing by a thread **1952**, TR queue **1980** has requests (retrieved via processing **1958** after insertion processing **1918** or **1930**) from a thread **1912** or **1922** so that thread **1952** performs whereabouts determination of the MS of architecture **1900**. Requests of queue **1980** comprise records **2400**. Preferably, there is a plurality (pool) of threads **1912** for feeding queue **1980** (i.e. feeding from queue **26**), and for feeding a plurality each of threads **1932** and **1952** from queue **1980**. All queue **1980** accesses are assumed to have appropriate semaphore control to ensure synchronous access by any thread at any particular time to prevent data corruption and misuse. Depending on alternative embodiments, queue **1980** may be viewed metaphorically for providing convenient grounds of explanation.

With reference now to FIG. **24**A, depicted is an illustration for describing a preferred embodiment of a thread request queue record, as maintained to Thread Request (TR) queue **1980**. TR queue **1980** is not required when a LN-expanse globally uses NTP, as found in thread 19xx processing described for architecture **1900**, however it may be required at a MS which does not have NTP, or a MS which interacts with another data processing system (e.g. MS) that does not have NTP. Therefore, TR queue record **2400** (i.e. queue entry **2400**) may, or may not, be required. This is the reason FIG. **1**A does not depict queue **1980**. When NTP is in use globally (in LN-expanse), TDOA measurements can be made using a single unidirectional data (**1302** or **1312**) packet containing a sent date/time stamp (of when the data was sent). Upon receipt, that sent date/time stamp received is compared with the date/time of receipt to determine the difference. The difference is a TDOA measurement. Knowing transmission speeds with a TDOA measurement allows calculating a distance. In this NTP scenario, no thread(s) **1932** are required.

Threads **1912** and/or DLM processing may always insert the MS whereabouts without requirement for thread(s) **1952** by incorporating thread **1952** logic into thread **1912**, or by directly starting (without queue **1980**) a thread **1952** from a thread **1912**. Therefore, threads **1952** may not be required. If threads **1952** are not required, queue **1980** may not be required by incorporating thread **1932** logic into thread **1912**, or by directly starting (without queue **1980**) a thread **1932** from a thread **1912**. Therefore, queue **1980** may not be required, and threads **1932** may not be required.

Records **2400** (i.e. queue entries **2400**) contain a request type field **2400***a* and data field **2400***b*. Request type field **2400***a* simply routes the queue entry to destined thread(s) (e.g. thread(s) **1932** or thread(s) **1952**). A thread **1932** remains blocked on queue **1980** until a record **2400** is inserted which has a field **2400***a* containing the value **1932**. A thread **1952** remains blocked on queue **1980** until a record **2400** is inserted which has a field **2400***a* containing the value **1952**. Data field **2400***b* is set to zero (0) when type field **2400***a* contains **1952** (i.e. not relevant). Data field **2400***b* contains an MS ID (field **1100***a*) value, and possibly a targeted communications interface **70** (or wave spectrum if one to one), when type field contains **1932**. Field **2400***b* will contain information for appropriately targeting the MS ID with data (e.g. communications interface to use if MS has multiple of them). An MS with only one communications interface can store only a MS ID in field **2400***b*.

Records **2400** are used to cause appropriate processing by 19xx threads (e.g. **1932** or **1952**) as invoked when needed (e.g. by thread(s) **1912**). Process **1932** is a slave to queue type of process, and there are no queue **1980** entries **2400** which will not get timely processed by a thread **1932**. No interim pruning is necessary to queue **1980**.

With reference now back to FIG. **19**, Correlation Response (CR) queue **1990** is for receiving correlation data for correlating requests transmitted in data **1302** with responses received in data **1302** or **1312**. Records **2450** are inserted to queue **1990** (via processing **1928**) from thread(s) **1922** so that thread(s) **1912** (after processing **1920**) correlate data **1302** or **1312** with requests sent by thread(s) **1922** (e.g. over interface **1926**), for the purpose of calculating a TDOA measurement. Additionally, records **2450** are inserted to queue **1990** (via processing **1936**) from thread(s) **1932** so that thread(s) **1912** (after processing **1920**) correlate data **1302** or **1312** with requests sent by thread(s) **1932** (e.g. over interface **1938**), for the purpose of calculating a TDOA measurement. Preferably, there is a plurality (pool) of threads for feeding queue **1990** and for feeding from queue **1990** (feeding from queue **1990** with thread(s) **1912**). All queue **1990** accesses are assumed to have appropriate semaphore control to ensure synchronous access by any thread at any particular time to prevent data corruption and misuse. Depending on alternative embodiments, queue **1990** may be viewed metaphorically for providing convenient grounds of explanation.

With reference now to FIG. **24**B, depicted is an illustration for describing a preferred embodiment of a correlation response queue record, as maintained to Correlation Response (CR) queue **1990**. CR queue **1990** is not required when a LN-expanse globally uses NTP, as found in thread 19xx processing described for architecture **1900**, however it may be required at a MS which does not have NTP, or a MS which interacts with another data processing system (e.g. MS) that does not have NTP. Therefore, CR record **2450** (i.e. queue entry **2450**) may, or may not, be required. This is the reason FIG. **1**A does not depict queue **1990**. The purpose of CR queue **1990** is to enable calculation of TDOA measurements using correlation data to match a request with a response. When NTP is used globally in the LN-expanse, no such correlations between a request and response is required, as described above. In the NTP scenario, thread(s) **1912** can deduce TDOA measurements directly from responses (see FIG. **21**), and there is no requirement for threads **1932**.

TDOA measurements are best taken using date/time stamps as close to the processing points of sending and receiving as possible, otherwise critical regions of code may be required for enabling process time adjustments to the measurements when processing is "further out" from said points. This is the reason MS receive processing provides received date/time stamps with data inserted to queue **26** (field **1100***p* or **2490***c*). In a preferred embodiment, send queue **24** processing inserts to queue **1990** so the date/time stamp field **2450***a* for when sent is as close to just prior to having been sent as possible. However, there is still the requirement for processing time spent inserting to queue **1990** prior to sending anyway. Anticipated processing speeds of architecture **1900** allow reasonably moving sent date/time stamp setting just a little "further out" from actually sending to keep modular send processing isolated. A preferred embodiment (as presented) assumes the send queue **24** interface minimizes processing instructions from when data is placed onto queue **24** and when it is actually sent, so that the sending thread(s) 19xx (**1902**, **1922**, **1932** and **1942**) insert to queue **1990** with a reasonably accurate sent/date stamp field **2450***a*. This ensures a most accurate sent date/time stamp (e.g. enabling most accurate TDOA measurements). An alternate embodiment makes appropriate adjustments for more accurate time to consider processing instructions up to the point of sending after queue **1990** insertion.

Records **2450** (i.e. queue entries **2450**) contain a date/time stamp field **2450***a* and a correlation data field **2450***b*. Date/time stamp field **2450***a* contains a date/time stamp of when a request (data **1302**) was sent as set by the thread inserting the queue entry **2450**. Correlation data field **2450***b* contains unique correlation data (e.g. MS id with suffix of unique number) used to provide correlation for matching sent requests (data **1302**) with received responses (data **1302** or **1312**), regardless of the particular communications interface(s) used (e.g. different wave spectrums supported by MS). Upon a correlation match, a TDOA measurement is calculated using the time difference between field **2450***a* and a date/time stamp of when the response was received (e.g. field **1100***p*). A thread **1912** accesses queue **1990** for a record **2450** using correlation field **2450***b* to match, when data **1302** or **1312** contains correlation data for matching. A thread **1912** then uses the field **2450***a* to calculate a TDOA measurement. Process **1912** is not a slave to queue **1990** (but is to queue **26**). A thread **1912** peeks queue **1990** for a matching entry when appropriate. Queue **1990** may contain obsolete queue entries **2450** until pruning is performed. Some WDR requests may be broadcasts, therefore records **2450** may be used for correlating a plurality of responses. In another record **2450** embodiment, an additional field **2450***c* is provided for specification of which communication interface(s) and/or channel(s) to listen on for a response.

With reference now back to FIG. **19**, any reasonable subset of architecture **1900** processing may be incorporated in a MS. For example in one minimal subset embodiment, a DLM which has excellent direct locating means only needs a single instance WDR (queue **22**) and a single thread **1902** for broadcasting whereabouts data to facilitate whereabouts determination by other MSs. In a near superset embodiment, process **1942** processing may be incorporated completely into process **1912**, thereby eliminating processing **1942** by having threads **1912** feed from queue **26** for WDR requests as well as WDR information. In another subset embodiment, process **1922** may only send requests to queue **24** for responses, or may only start a thread **1952** for determining whereabouts of the MS. There are many viable subset embodiments depending on the MS being a DLM or ILM, capabilities of the MS, LN-expanse deployment design choices, etc. A reference to FIG. **19** accompanies thread 19xx flowcharts (FIGS. **20**, **21**, **22**, **23**, **25** and **26**A). The user, preferably an administrator type (e.g. for lbxPhone™ debug) selectively configures whether or not to start or terminate a process (thread pool), and perhaps the number of threads to start in the pool (see FIG. **14**A). Starting a process (and threads) and terminating processes (and threads) is shown in flowcharts **29**A and **29**B. There are other embodiments for properly starting and terminating threads without departing from the spirit and scope of this disclosure.

LBX of data may also be viewed as LBX of objects, for example a WDR, WDR request, TDOA request, AOA request, charters, permissions, data record(s), or any other data may be viewed as an object. An subset of an object or data may also be viewed as an object.

FIG. **20** depicts a flowchart for describing a preferred embodiment of MS whereabouts broadcast processing, for example to facilitate other MSs in locating themselves in an LN-expanse. FIG. **20** processing describes a process **1902** worker thread, and is of PIP code **6**. Thread(s) **1902** purpose is for the MS of FIG. **20** processing (e.g. a first, or sending, MS) to periodically transmit whereabouts information to other MSs (e.g. at least a second, or receiving, MS) to use in locating themselves. It is recommended that validity criteria set at block **1444** for **1902**-Max be fixed at one (1) in the preferred embodiment. Multiple channels for broadcast at block **2016** should be isolated to modular send processing (feeding from a queue **24**).

In an alternative embodiment having multiple transmission channels visible to process **1902**, there can be a worker thread **1902** per channel to handle broadcasting on multiple channels. If thread(s) **1902** (block **2016**) do not transmit directly over the channel themselves, this embodiment would provide means for communicating the channel for broadcast to send processing when interfacing to queue **24** (e.g. incorporate a channel qualifier field with WDR inserted to queue **24**). This embodiment could allow specification of at least one (1) worker thread per channel, however multiple worker threads configurable for process **1902** as appropriated for the number of channels configurable for broadcast.

Processing begins at block **2002**, continues to block **2004** where the process worker thread count **1902**-Ct is accessed and incremented by 1 (using appropriate semaphore access (e.g. **1902**-Sem)), and continues to block **2006** for peeking WDR queue **22** for a special termination request entry. Block **2004** may also check the **1902**-Ct value, and signal the process **1902** parent thread that all worker threads are running when **1902**-Ct reaches **1902**-Max. Thereafter, if block **2008** determines that a worker thread termination request was not found in queue **22**, processing continues to block **2010**. Block **2010** peeks the WDR queue **22** (using interface **1904**) for the most recent highest confidence entry for this MS whereabouts by searching queue **22** for: the MS ID field **1100***a* matching the MS ID of FIG. **20** processing, and a confidence field **1100***d* greater than or equal to the confidence floor value, and a most recent NTP enabled date/time stamp field **1100***b* within a prescribed trailing period of time (e.g. preferably less than or equal to 2 seconds). For example, block **2010** peeks the queue (i.e. makes a copy for use if an entry found for subsequent processing, but does not remove the entry from queue) for a WDR of this MS (i.e. MS of FIG. **20** processing) which has the greatest confidence over 75 and has been most recently inserted to queue **22** with an NTP date/time stamp in the last 2 seconds. Date/time stamps for MS whereabouts which are not NTP derived have little use in the overall palette of process 19xx choices of architecture **1900** because receiving data processing systems (e.g. MSs) will have no means of determining an accurate TDOA measurement in the unidirectional transmission from an NTP disabled MS. A receiving data processing system will still require a bidirectional correlated exchange with the MS of FIG. **20** processing to determine an accurate TDOA measurement in its own time scale (which is accomplished with thread(s) **1922** pulling WDR information anyway). An alternate embodiment to block **2010** will not use the NTP indicator as a search criteria so that receiving data processing systems can receive to a thread **1912**, and then continue for appropriate correlation processing, or can at least maintain whereabouts to queue **22** to know who is nearby.

Thread **1902** is of less value to the LN-expanse when it broadcasts outdated/invalid whereabouts of the MS to facilitate locating other MSs. In an alternate embodiment, a movement tolerance (e.g. user configured or system set (e.g. 3 meters)) is incorporated at the MS, or at service(s) used to locate the MS, for knowing when the MS has significantly moved (e.g. more than 3 meters) and how long it has been (e.g. 45 seconds) since last significantly moving. In this embodiment, the MS is aware of the period of time since last significantly moving and the search time criteria is set using the amount of time since the MS significantly moved (whichever is greater). This way a large number of (perhaps more confident candidates) WDRs are searched in the time period when the MS has not significantly moved. Optional blocks **278** through **284** may have been incorporated to FIG. **2**F for movement tolerance processing just described, in which case the LWT is compared to the current date/time of block **2010** processing to adjust block **2010** search time criteria for the correct trailing period. In any case, a WDR is sought at block **2010** which will help other MSs in the LN-expanse locate themselves, and to let other MSs know who is nearby.

Thereafter, if block **2012** determines a useful WDR was found, then block **2014** prepares the WDR for send processing, block **2016** broadcasts the WDR information (using send interface **1906**) by inserting to queue **24** so that send processing broadcasts data **1302** (e.g. on all available communications interface(s) **70**), for example as far as radius **1306**, and processing continues to block **2018**. The broadcast is for reception by data processing systems (e.g. MSs) in the vicinity. At least fields **1100***b*, **1100***c*, **1100***d*, and **1100***n* are broadcast. See FIG. **11**A descriptions. Fields are set to the following upon exit from block **2014**:

MS ID field **1100***a* is preferably set with: Field **1100***a* from queue **22**, or transformed (if not already) into a pseudo MS ID (possibly for future correlation) if desired. This field may also be set to null (not set) because it is not required when the NTP indicator of field **1100***b* is enabled and the broadcast is sent with an NTP enabled field **1100***n*.

DATE/TIME STAMP field **1100***b* is preferably set with: Field **1100***b* from queue **22**.

LOCATION field **1100***c* is preferably set with: Field **1100***c* from queue **22**.

CONFIDENCE field **1100***d* is preferably set with: Field **1100***d* from queue **22**.

LOCATION TECHNOLOGY field **1100***e* is preferably set with: Field **1100***e* from queue **22**.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: null (not set). Null indicates to send processing feeding from queue **24** to use all available comm. interfaces **70** (i.e. Broadcast). Specifying a comm. interface targets the specified interface (i.e. send).

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: null (not set). If MS ID (or pseudo MS ID) is sent, this is all that is required to target this MS.

SPEED field **1100***h* is preferably set with: Field **1100***h* from queue **22**.

HEADING field **1100***i* is preferably set with: Field **1100***i* from queue **22**.

ELEVATION field **1100***j* is preferably set with: Field **1100***j* from queue **22**.

APPLICATION FIELDS field **1100***k* is preferably set with: Field **1100***k* from queue **22**. An alternate embodiment will add, alter, or discard data (with or without date/time stamps) here at the time of block **2014** processing.

CORRELATION FIELD **1100***m* is preferably set with: null (not set).

SENT DATE/TIME STAMP field **1100***n* is preferably set with: Sent date/time stamp as close in processing the broadcast of block **2016** as possible.

RECEIVED DATE/TIME STAMP field **1100***p* is preferably set with: Not Applicable (i.e. N/A for sending).

Block **2018** causes thread **1902** to sleep according to the SPTP setting (e.g. a few seconds). When the sleep time has elapsed, processing continues back to block **2006** for another loop iteration of blocks **2006** through **2016**. Referring back to block **2012**, if a useful WDR was not found (e.g. candidates too old), then processing continues to block **2018**. Referring back to block **2008**, if a worker thread termination request entry was found at queue **22**, then block **2020** decrements the worker thread count by 1 (using appropriate semaphore access (e.g. **1902**-Sem)), and thread **1902** processing terminates at block **2022**. Block **2020** may also check the **1902**-Ct value, and signal the process **1902** parent thread that all worker threads are terminated when **1902**-Ct equals zero (0).

Block **2016** causes broadcasting data **1302** containing CK **1304** wherein CK **1304** contains WDR information prepared as described above for block **2014**. Alternative embodiments of block **2010** may not search a specified confidence value, and broadcast the best entry available anyway so that listeners in the vicinity will decide what to do with it. A semaphore protected data access (instead of a queue peek) may be used in embodiments where there is always one WDR current entry maintained for the MS.

In the embodiment wherein usual MS communications data **1302** of the MS is altered to contain CK **1304** for listening MSs in the vicinity, send processing feeding from queue **24**, caused by block **2016** processing, will place WDR information as CK **1304** embedded in usual data **1302** at the next opportune time of sending usual data **1302**. If an opportune time is not timely, send processing should discard the send request of block **2016** to avoid broadcasting outdated whereabouts information (unless using a movement tolerance and time since last significant movement). As the MS conducts its normal communications, transmitted data **1302** contains new data CK **1304** to be ignored by receiving MS other character **32** processing, but to be found by listening MSs within the vicinity which anticipate presence of CK **1304**. Otherwise, when LN-Expanse deployments have not introduced CK **1304** to usual data **1302** communicated on a receivable signal by MSs in the vicinity, FIG. **20** sends repeated timely pulsed broadcasts of new data **1302** (per SPTP) for MSs in the vicinity of the first MS to receive. In any case, appropriate implementation should ensure field **1100***n* is as accurate as possible for when data **1302** is actually sent.

An alternate embodiment to architecture **1900** for elimination of process **1902** incorporates a trigger implementation for broadcasting MS whereabouts at the best possible time—i.e. when the MS whereabouts is inserted to queue **22**. As soon as a new (preferably NTP enabled) WDR candidate becomes available, it can be broadcast at a new block **279** of FIG. **2**F. (e.g. new block **279** continued to from block **278** and then continuing to block **280**). Fields are set as described above for FIG. **20**. Preferably, the new block **279** starts an asynchronous thread consisting of blocks **2014** and **2016** so that FIG. **2**F processing performance is not impacted. In a further embodiment, block **279** can be further enhanced using the SPTP value to make sure that too many broadcasts are not made. The SPTP (Source Periodicity Time Period) could be observed for getting as close as possible to broadcasting whereabouts in accordance with SPTP (e.g. worst case there are not enough broadcasts).

FIG. **21** depicts a flowchart for describing a preferred embodiment of MS whereabouts collection processing. FIG. **21** processing describes a process **1912** worker thread, and is of PIP code **6**. Thread(s) **1912** purpose is for the MS of FIG. **21** processing (e.g. a second, or receiving, MS) to collect potentially useful WDR information from other MSs (e.g. at least a first, or sending, MS) in the vicinity for determining whereabouts of the receiving (second) MS. It is recommended that validity criteria set at block **1444** for **1912**-Max be set as high as possible (e.g. 10) relative performance considerations of architecture **1900**, with at least one thread per channel that WDR information may be received on by the receiving MS. Multiple channels for receiving data fed to queue **26** should be isolated to modular receive processing (feeding a queue **26**).

In an alternative embodiment having multiple receiving transmission channels visible to process **1912** (e.g. thread(s) **1912** receiving directly), there can be a worker thread **1912** per channel to handle receiving on multiple channels simultaneously. If thread(s) **1912** do not receive directly from the channel, the preferred embodiment of FIG. **21** would not need to convey channel information to thread(s) **1912** waiting on queue **26** anyway. Embodiments could allow specification/configuration of many thread(s) **1912** per channel.

Processing begins at block **2102**, continues to block **2104** where the process worker thread count **1912**-Ct is accessed and incremented by 1 (using appropriate semaphore access (e.g. **1912**-Sem)), and continues to block **2106** for interim housekeeping of pruning the WDR queue by invoking a Prune Queues procedure of FIG. **27**. Block **2104** may also check the **1912**-Ct value, and signal the process **1912** parent thread that all worker threads are running when **1912**-Ct reaches **1912**-Max. Block **2106** may not be required since block **2130** can cause queue **22** pruning (block **292**).

Thereafter, block **2108** retrieves from queue **26** a WDR (using interface **1914**), perhaps a special termination request entry, or a WDR received in data **1302** (CK **1304**) or data **1312** (CK **1314**), and only continues to block **2110** when a WDR has been retrieved. Block **2108** stays blocked on retrieving from queue **26** until any WDR is retrieved. If block **2110** determines that a special WDR indicating to terminate was not found in queue **26**, processing continues to block **2112**. Block **2112** adjusts date/time stamp field **1100***b* if necessary depending on NTP use in the LN-expanse and adjusts the confidence field **1100***d* accordingly. In a preferred embodiment, fields **1100***b* and **1100***d* for the WDR in process is set as follows for certain conditions:

Fields **1100***b*, **1100***n* and **1100***p* all NTP indicated: keep fields **1100***b* and **1100***d* as is; or

Fields **1100***b* and **1100***n* are NTP indicated, **1100***p* is not: Is correlation (field **1100***m*) present?: No, then set confidence (field **1100***d*) to 0 (for filtering out at block **2114**)/Yes, then set field **1100***b* to **1100***p* (in time terms of this MS) and adjust confidence lower based on differences between fields **1100***b*, **1100***n* and **1100***p*; or

Fields **1100***b* and **1100***p* are NTP indicated, **1100***n* is not: Is correlation present?: No, then set confidence to 0 (for filtering out at block **2114**)/Yes, then set field **1100***b* to

1100*p* (in time terms of this MS) and adjust confidence lower based on differences between fields **1100***b*, **1100***n* and **1100***p*; or

Fields **1100***b* NTP indicated, **1100***n* and **1100***p* not: Is correlation present?: No, then set confidence to 0 (for filtering out at block **2114**)/Yes, then set field **1100***b* to **1100***p* (in time terms of this MS) and adjust confidence lower based on differences between fields **1100***b*, **1100***n* and **1100***p*; or

Field **1100***b* not NTP indicated, **1100***n* and **1100***p* are: Is correlation present?: No, then set confidence to 0 (for filtering out at block **2114**)/Yes, then set field **1100***b* to **1100***p* (in time terms of this MS) and adjust confidence lower based on differences between fields **1100***b*, **1100***n* and **1100***p*; or

Fields **1100***b* and **1100***p* are not NTP indicated, **1100***n* is: Is correlation present?: No, then set confidence to 0 (for filtering out at block **2114**)/Yes, then set field **1100***b* to **1100***p* (in time terms of this MS) and adjust confidence lower based on differences between fields **1100***b*, **1100***n* and **1100***p*; or

Fields **1100***b* and **1100***n* are not NTP indicated, **1100***p* is: Is correlation present?: No, then set confidence to 0 (for filtering out at block **2114**)/Yes, then set field **1100***b* to **1100***p* (in time terms of this MS) and adjust confidence lower based on differences between fields **1100***b*, **1100***n* and **1100***p*; or

Fields **1100***b*, **1100***n* and **1100***p* not NTP indicated: Is correlation present?: No, then set confidence to 0 (for filtering out at block **2114**)/Yes, then set field **1100***b* to **1100***p* (in time terms of this MS) and adjust confidence lower based on differences between fields **1100***b*, **1100***n* and **1100***p*.

NTP ensures maintaining a high confidence in the LN-expanse, but absence of NTP is still useful. Confidence values should be adjusted with the knowledge of the trailing time periods used for searches when sharing whereabouts (e.g. thread(s) **1942** searches). Block **2112** continues to block **2114**.

If at block **2114**, the WDR confidence field **1100***d* is not greater than the confidence floor value, then processing continues back to block **2106**. If block **2114** determines that the WDR field **1100***d* is satisfactory, then block **2116** initializes a TDOA_FINAL variable to False, and block **2118** checks if the WDR from block **2108** contains correlation (field **1100***m*).

If block **2118** determines the WDR does not contain correlation, then block **2120** accesses the ILMV, block **2122** determines the source (ILM or DLM) of the WDR using the originator indicator of field **1100***e*, and block **2124** checks suitability for collection of the WDR. While processes 19xx running are generally reflective of the ILMV roles configured, it is possible that the more descriptive nature of ILMV role(s) not be one to one in relationship to 19xx processes, in particular depending on the subset of architecture **1900** in use. Block **2124** is redundant anyway because of block **274**. If block **2124** determines the ILMV role is disabled for collecting this WDR, then processing continues back to block **2106**. If block **2124** determines the ILMV role is enabled for collecting this WDR, then processing continues to block **2126**.

If block **2126** determines both the first (sending) and second (receiving) MS are NTP enabled (i.e. Fields **1100***b*, **1100***n* and **1100***p* are NTP indicated) OR if TDOA_FINAL is set to True (as arrived to via block **2150**), then block **2128** completes the WDR for queue **22** insertion, block **2130** prepares parameters for FIG. **2**F processing and block **2132** invokes FIG. **2**F processing (interface **1916**). Parameters set at block **2130** are: WDRREF=a reference or pointer to the WDR completed at block **2128**; DELETEQ=FIG. **21** location queue discard processing; and SUPER=FIG. **21** supervisory notification processing. Block **2128** calculates a TDOA measurement whenever possible and inserts to field **1100***f*. See FIG. **11**A descriptions. Fields are set to the following upon exit from block **2128**:

MS ID field **1100***a* is preferably set with: Field **1100***a* from queue **26**.

DATE/TIME STAMP field **1100***b* is preferably set with: Preferred embodiment discussed for block **2112**.

LOCATION field **1100***c* is preferably set with: Field **1100***c* from queue **26**.

CONFIDENCE field **1100***d* is preferably set with: Confidence at equal to or less than field **1100***d* received from queue **26** (see preferred embodiment for block **2112**).

LOCATION TECHNOLOGY field **1100***e* is preferably set with: Field **1100***e* from queue **26**.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: All available measurements from receive processing (e.g. AOA, heading, yaw, pitch, roll, signal strength, wave spectrum, particular communications interface **70**, etc), and TDOA measurement(s) as determined in FIG. **21** (blocks **2128** and **2148**).

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: Field **1100***g* from queue **26**.

SPEED field **1100***h* is preferably set with: Field **1100***h* from queue **26**.

HEADING field **1100***i* is preferably set with: Field **1100***i* from queue **26**.

ELEVATION field **1100***j* is preferably set with: Field **1100***j* from queue **26**.

APPLICATION FIELDS field **1100***k* is preferably set with: Field **1100***k* from queue **26**. An alternate embodiment will add, alter, or discard data (with or without date/time stamps) here at the time of block **2128** processing.

CORRELATION FIELD **1100***m* is preferably set with: Not Applicable (i.e. not maintained to queue **22**). Was used by FIG. **21** processing.

SENT DATE/TIME STAMP field **1100***n* is preferably set with: Not Applicable (i.e. not maintained to queue **22**). Was used by FIG. **21** processing.

RECEIVED DATE/TIME STAMP field **1100***p* is preferably set with: Not Applicable (i.e. not maintained to queue **22**). Was used by FIG. **21** processing.

Block **2132** continues to block **2134** where a record **2400** is built (i.e. field **2400***a*=**1952** and field **2400***b* is set to null (e.g. −1)) and then block **2136** inserts the record **2400** to TR queue **1980** (using interface **1918**) so that a thread **1952** will perform processing. Blocks **2134** and **2136** may be replaced with an alternative embodiment for starting a thread **1952**. Block **2136** continues back to block **2106**.

Referring now back to block **2126**, if it is determined that a TDOA measurement cannot be made (i.e. (field **1100***n* or **1100***p* not NTP indicated) OR if TDOA_FINAL is set to False), then block **2138** checks if the WDR contains a MS ID (or pseudo MS ID). If block **2138** determines there is none, then processing continues back to block **2106** because there is no way to distinguish one MS from another with respect to the WDR retrieved at block **2108** for directing bidirectional correlation. An alternate embodiment will use a provided correlation field **1100***m* received at block **2108**, instead of a field **1100***a*, for knowing how to target the originating MS for TDOA measurement processing initiated by a thread **1932**. If block **2138** determines there is a usable MS ID (or correlation field), then block **2140** builds a record **2400** (field **2400***a*=**1932**, field **2400***b*=the MS ID (or pseudo MS ID, or correlation) and particular communications interface from field **1100***f* (if available) of the WDR of block **2108**, and block **2142** inserts the record **2400** to queue **1980** (interface **1918**) for starting a thread **1932**. Block **2142** continues back to block **2106**. An alternate embodiment causes block **2126** to continue directly to block **2140** (no block **2138**) for a No condition from block **2126**. Regardless of whether the originating MS ID can be targeted, a correlation (in lieu of an MS ID) may be used when the MS responds with a broadcast. The WDR request made by thread **1932** can be a broadcast rather than a targeted request. Thread(s) **1932** can handle sending targeted WDR requests (to a known MS ID) and broadcast WDR requests.

Referring back to block **2118**, if it is determined the WDR does contain correlation (field **1100***m*), block **2144** peeks the CR queue **1990** (using interface **1920**) for a record **2450** containing a match (i.e. field **1100***m* matched to field **2450***b*). Thereafter, if block **2146** determines no correlation was found on queue **1990** (e.g. response took too long and entry was pruned), then processing continues to block **2120** already described. If block **2146** determines the correlation entry was found (i.e. thread **1912** received a response from an earlier request (e.g. from a thread **1922** or **1932**), then block **2148** uses date/time stamp field **2450***a* (from block **2144**) with field **1100***p* (e.g. from block **2108**) to calculate a TDOA measurement in time scale of the MS of FIG. **21** processing, and sets field **1100***f* appropriately in the WDR. Note that correlation field **2450***b* is valid across all available MS communications interfaces (e.g. all supported active wave spectrums). The TDOA measurement considers duration of time between the earlier sent date/time of record **2450** and the later time of received date/time field **1100***p*. The TDOA measurement may further be altered at block **2148** processing time to a distance knowing the velocity of the wave spectrum used as received to queue **26**. Block **2148** continues to block **2150** where the TDOA_FINAL variable is set to True, then to block **2120** for processing already described.

Referring back to block **2110**, if a WDR for a worker thread termination request was found at queue **26**, then block **2152** decrements the worker thread count by 1 (using appropriate semaphore access (e.g. **1912**-Sem)), and thread **1912** processing terminates at block **2154**. Block **2152** may also check the **1912**-Ct value, and signal the process **1912** parent thread that all worker threads are terminated when **1912**-Ct equals zero (0).

In the embodiment wherein usual MS communications data **1302** of the MS is altered to contain CK **1304** or **1314** for listening MSs in the vicinity, receive processing feeding queue **26** will place WDR information to queue **26** as CK **1304** or **1314** is detected for being present in usual communication data **1302** or **1304**. As normal communications are conducted, transmitted data **1302** or **1312** contains new data CK **1304** or **1314** to be ignored by receiving MS other character **32** processing, but to be found by listening MSs within the vicinity which anticipate presence of CK **1304** or **1314**. Otherwise, when LN-Expanse deployments have not introduced CK **1304** (or **1314**) to usual data **1302** (or **1312**) communicated on a receivable signal by MSs in the vicinity, FIG. **21** receives new data **1302** (or **1312**) sent. In any case, field **1100***p* should be as accurate as possible for when data **1302** (or **1312**) was actually received. Critical regions of code and/or anticipated execution timing may be used to affect a best setting of field **1100***p*.

So, FIG. **21** is responsible for maintaining whereabouts of others to queue **22** with data useful for triangulating itself.

FIG. **22** depicts a flowchart for describing a preferred embodiment of MS whereabouts supervisor processing, for example to ensure the MS of FIG. **22** processing (e.g. first MS) is maintaining timely whereabouts information for itself. FIG. **22** processing describes a process **1922** worker thread, and is of PIP code **6**. Thread(s) **1922** purpose is for the MS of FIG. **22** processing (e.g. a first, or sending, MS), after determining its whereabouts are stale, to periodically transmit requests for whereabouts information from MSs in the vicinity (e.g. from at least a second, or receiving, MS), and/or to start a thread **1952** for immediately determining whereabouts. Alternative embodiments to FIG. **22** will implement processing of blocks **2218** through **2224**, or processing of blocks **2226** through **2228**, or both as depicted in FIG. **22**. It is recommended that validity criteria set at block **1444** for **1922**-Max be fixed at one (1) in the preferred embodiment. Multiple channels for broadcast at block **2224** should be isolated to modular send processing feeding from a queue **24**.

In an alternative embodiment having multiple transmission channels visible to process **1922**, there can be a worker thread **1922** per channel to handle broadcasting on multiple channels. If thread(s) **1922** (block **2224**) do not transmit directly over the channel, this embodiment would provide means for communicating the channel for broadcast to send processing when interfacing to queue **24** (e.g. incorporate a channel qualifier field with WDR request inserted to queue **24**). This embodiment could allow specification of one (1) thread per channel, however multiple worker threads configurable for process **1922** as determined by the number of channels configurable for broadcast.

Processing begins at block **2202**, continues to block **2204** where the process worker thread count **1922**-Ct is accessed and incremented by 1 (using appropriate semaphore access (e.g. **1922**-Sem)), and continues to block **2206** for interim housekeeping of pruning the CR queue by invoking a Prune Queues procedure of FIG. **27**. Block **2204** may also check the **1922**-Ct value, and signal the process **1922** parent thread that all worker threads are running when **1922**-Ct reaches **1922**-Max. Block **2206** continues to block **2208** for peeking WDR queue **22** (using interface **1924**) for a special termination request entry. Thereafter, if block **2210** determines that a worker thread termination request was not found in queue **22**, processing continues to block **2212**. Block **2212** peeks the WDR queue **22** (using interface **1924**) for the most recent highest confidence entry for this MS whereabouts by searching queue **22** for: the MS ID field **1100***a* matching the MS ID of FIG. **22** processing, and a confidence field **1100***d* greater than or equal to the confidence floor value, and a most recent date/time stamp field **1100***b* within a prescribed trailing period of time of block **2212** search processing using a function of the WTV (i.e. f(WTV)=short-hand for "function of WTV") for the period. For example, block **2212** peeks the queue (i.e. makes a copy for use if an entry found for subsequent processing, but does not remove the entry from queue) for a WDR of the first MS which has the greatest confidence over 75 and has been most recently inserted to queue **22** in the last 3 seconds. Since the MS whereabouts accuracy may be dependent on timeliness of the WTV, it is recommended that the f(WTV) be some value less than or equal to WTV, but preferably not greater than the WTV. Thread **1922** is of less value to the MS when not making sure in a timely manner the MS is maintaining timely whereabouts for itself. In an alternate embodiment, a movement tolerance (e.g. user configured or system set (e.g. 3 meters)) is incorporated at the MS, or at service(s) used to locate the MS, for knowing when the MS has significantly moved (e.g. more than 3 meters) and how long it has been (e.g. 45 seconds) since last significantly moving. In this embodiment, the MS is aware of the period of time since last significantly moving and the f(WTV) is set using the amount of time since the MS significantly moved (i.e. f(WTV)=as described above, or the amount of time since significantly moving, whichever is greater). This way a large number of (perhaps more confident candidates) WDRs are searched in the time period when the MS has not significantly moved. Optional blocks **278** through **284** may have been incorporated to FIG. **2**F for movement tolerance processing just described, in which case the LWT is compared to the current date/time to adjust the WTV for the correct trailing period. In any case, a WDR is sought at block **2212** which will verify whether or not MS whereabouts are current.

Thereafter, if block **2214** determines a satisfactory WDR was found, then processing continues to block **2216**. Block **2216** causes thread **1922** to sleep according to a f(WTV) (preferably a value less than or equal to the WTV (e.g. 95% of WTV)). When the sleep time has elapsed, processing continues back to block **2206** for another loop iteration of blocks **2206** through **2214**.

If block **2214** determines a current WDR was not found, then block **2218** builds a WDR request (e.g. containing record **2490** with field **2490***a* for the MS of FIG. **22** processing (MS ID or pseudo MS ID) so receiving MSs in the LN-expanse know who to respond to, and field **2490***b* with appropriate correlation for response), block **2220** builds a record **2450** (using correlation generated for the request at block **2218**), block **2222** inserts the record **2450** to queue **1990** (using interface **1928**), and block **2224** broadcasts the WDR request (record **2490**) for responses. Absence of field **2490***d* indicates to send processing feeding from queue **24** to broadcast on all available comm. interfaces **70**.

With reference now to FIG. **24**C, depicted is an illustration for describing a preferred embodiment of a WDR request record, as communicated to queue **24** or **26**. When a LN-expanse globally uses NTP, as found in thread 19xx processing described for architecture **1900**, a WDR request record **2490** may, or may not, be required. TDOA calculations can be made using a single unidirectional data (**1302** or **1312**) packet containing a sent date/time stamp (of when the data was sent) as described above.

Records **2490** contain a MS ID field **2490***a* and correlation field **2490***b*. MS ID field **2490***a* contains an MS ID (e.g. a value of field **1100***a*). An alternate embodiment will contain a pseudo MS ID (for correlation), perhaps made by a derivative of the MS ID with a unique (suffix) portion, so that receiving MSs can directly address the MS sending the request without actually knowing the MS ID (i.e. they know the pseudo MS ID which enables the MS to recognize originated transmissions). Correlation data field **2490***b* contains unique correlation data (e.g. MS id with suffix of unique number) used to provide correlation for matching sent requests (data **1302**) with received WDR responses (data **1302** or **1312**). Upon a correlation match, a TDOA measurement is calculated using the time difference between field **2450***a* and a date/time stamp of when the response was received (e.g. field **1100***p*). Received date/time stamp field **2490***c* is added by receive processing feeding queue **26** when an MS received the request from another MS. Comm interface field **2490***d* is added by receive processing inserting to queue **26** for how to respond and target the originator. Many MSs do not have choices of communications interfaces, so field **2490***d* may not be required. If available it is used, otherwise a response can be a broadcast. Field **2490***d* may contain a wave spectrum identifier for uniquely identifying how to respond (e.g. one to one with communications interface), or any other value for indicating how to send given how the request was received.

With reference back to FIG. **22**, block **2218** builds a request that receiving MSs will know is for soliciting a response with WDR information. Block **2218** generates correlation for field **2450***b* to be returned in responses to the WDR request broadcast at block **2224**. Block **2220** also sets field **2450***a* to when the request was sent. Preferably, field **2450***a* is set as close to the broadcast as possible. In an alternative embodiment, broadcast processing feeding from queue **24** makes the record **2450** and inserts it to queue **1990** with a most accurate time of when the request was actually sent. Fields **2450***a* are to be as accurate as possible. Block **2224** broadcasts the WDR request data **1302** (using send interface **1926**) by inserting to queue **24** so that send processing broadcasts data **1302**, for example as far as radius **1306**. Broadcasting preferably uses all available communications interface(s) **70** (e.g. all available wave spectrums). Therefore, the comm interface field **2490***d* is not set (which implies to send processing to do a broadcast).

Block **2224** continues to block **2226** where a record **2400** is built (i.e. field **2400***a*=**1952** and field **2400***b* is set to null (e.g. −1)) and then block **2228** inserts the record **2400** to TR queue **1980** (using interface **1930**) so that a thread **1952** will perform processing. Blocks **2226** and **2228** may be replaced with an alternative embodiment for starting a thread **1952**. Block **2228** continues back to block **2216**.

Referring back to block **2210**, if a worker thread termination request entry was found at queue **22**, then block **2230** decrements the worker thread count by 1 (using appropriate semaphore access (e.g. **1922**-Sem)), and thread **1922** processing terminates at block **2232**. Block **2230** may also check the **1922**-Ct value, and signal the process **1922** parent thread that all worker threads are terminated when **1922**-Ct equals zero (0).

In the embodiment wherein usual MS communications data **1302** of the MS is altered to contain CK **1304** for listening MSs in the vicinity, send processing feeding from queue **24**, caused by block **2224** processing, will place the request as CK **1304** embedded in usual data **1302** at the next opportune time of sending usual data **1302**. This may require the alternative embodiment of adding the entry to queue **1990** being part of send processing. As the MS conducts its normal communications, transmitted data **1302** contains new data CK **1304** to be ignored by receiving MS other character **32** processing, but to be found by listening MSs within the vicinity which anticipate presence of CK **1304**. Otherwise, when LN-Expanse deployments have not introduced CK **1304** to usual data **1302** communicated on a receivable signal by MSs in the vicinity, FIG. **22** sends new WDR request data **1302**.

FIG. **23** depicts a flowchart for describing a preferred embodiment of MS timing determination processing. FIG. **23** processing describes a process **1932** worker thread, and is of PIP code **6**. Thread(s) **1932** purpose is for the MS of FIG. **23** processing to determine TDOA measurements when needed for WDR information received. It is recommended that validity criteria set at block **1444** for **1932**-Max be set as high as possible (e.g. 12) relative performance considerations of architecture **1900**, to service multiple threads **1912**.

Processing begins at block **2302**, continues to block **2304** where the process worker thread count **1932**-Ct is accessed and incremented by 1 (using appropriate semaphore access (e.g. **1932**-Sem)), and continues to block **2306** for interim housekeeping of pruning the CR queue by invoking a Prune Queues procedure of FIG. **27**. Block **2304** may also check the **1932**-Ct value, and signal the process **1932** parent thread that all worker threads are running when **1932**-Ct reaches **1932**-Max.

Thereafter, block **2308** retrieves from queue **1980** a record **2400** (using interface **1934**), perhaps a special termination request entry, or a record **2400** received from thread(s) **1912**, and only continues to block **2310** when a record **2400** containing field **2400***a* set to **1932** has been retrieved. Block **2308** stays blocked on retrieving from queue **1980** until a record **2400** with field **2400***a*=**1932** is retrieved. If block **2310** determines a special entry indicating to terminate was not found in queue **1980**, processing continues to block **2312**.

If at block **2312**, the record **2400** does not contain a MS ID (or pseudo MS ID) in field **2400***b*, processing continues to block **2314** for building a WDR request (record **2490**) to be broadcast, and then to block **2318**. Broadcasting preferably uses all available communications interface(s) **70** (e.g. all available wave spectrums). If block **2312** determines the field **2400***b* is a valid MS ID (not null), block **2316** builds a WDR request targeted for the MS ID, and processing continues to block **2318**. A targeted request is built for targeting the MS ID (and communications interface, if available) from field **2400***b*. Send processing is told which communications interface to use, if available (e.g. MS has multiple), otherwise send processing will target each available interface. In the unlikely case a MS ID is present in field **2400***b* without the communications interface applicable, then all communications interfaces **70** are used with the targeted MS ID. In MS embodiments with multiple communications interfaces **70**, then **2400***b* is to contain the applicable communication interface for sending. Block **2318** generates appropriate correlation for a field **2450***b* (e.g. to be compared with a response WDR at block **2144**), block **2320** sets field **2450***a* to the current MS date/time stamp, block **2322** inserts the record **2450** to queue **1990** (using interface **1936**), and block **2324** sends/broadcasts (using interface **1938**) a WDR request (record **2490**). Thereafter, processing continues back to block **2306** for another loop iteration. An alternative embodiment will only target a WDR request to a known MS ID. For example, block **2312** would continue back to block **2306** if no MS ID is found (=null), otherwise it will continue to block **2316** (i.e. no use for block **2314**).

Block **2318** sets field **2450***b* to correlation to be returned in responses to the WDR request sent/broadcast at block **2324**. Block **2320** sets field **2450***a* to when the request is sent. Preferably, field **2450***a* is set as close as possible to when a send occurred. In an alternative embodiment, send processing feeding from queue **24** makes the record **2450** and inserts it to queue **1990** with a most accurate time of when the request was actually sent. Fields **2450***a* are to be as accurate as possible. Block **2324** sends/broadcasts the WDR request data **1302** (using send interface **1938**) by inserting to queue **24** a record **2490** (**2490***a*=the targeted MS ID (or pseudo MS ID) OR null if arrived to from block **2314**, field **2490***b*=correlation generated at block **2318**) so that send processing sends data **1302**, for example as far as radius **1306**. A null MS ID may be responded to by all MSs in the vicinity. A non-null MS ID is to be responded to by a particular MS. Presence of field **2490***d* indicates to send processing feeding from queue **24** to target the MS ID over the specified comm. interface (e.g. when MS has a plurality of comm. interfaces **70** (e.g. cellular, WiFi, Bluetooth, etc; i.e. MS supports multiple classes of wave spectrum)).

Referring back to block **2310**, if a worker thread termination request was found at queue **1980**, then block **2326** decrements the worker thread count by 1 (using appropriate semaphore access (e.g. **1932**-Sem)), and thread **1932** processing terminates at block **2328**. Block **2326** may also check the **1932**-Ct value, and signal the process **1932** parent thread that all worker threads are terminated when **1932**-Ct equals zero (0).

In the embodiment wherein usual MS communications data **1302** of the MS is altered to contain CK **1304** for listening MSs in the vicinity, send processing feeding from queue **24**, caused by block **2324** processing, will place the WDR request as CK **1304** embedded in usual data **1302** at the next opportune time of sending usual data **1302**. As the MS conducts its normal communications, transmitted data **1302** contains new data CK **1304** to be ignored by receiving MS other character **32** processing, but to be found by listening MSs within the vicinity which anticipate presence of CK **1304**. This may require the alternative embodiment of adding the entry to queue **1990** being part of send processing. Otherwise, when LN-Expanse deployments have not introduced CK **1304** to usual data **1302** communicated on a receivable signal by MSs in the vicinity, FIG. **22** sends/broadcasts new WDR request data **1302**.

An alternate embodiment to block **2324** can wait for a response with a reasonable timeout, thereby eliminating the need for blocks **2318** through **2322** which is used to correlate the subsequent response (to thread **1912**) with the request sent at block **2324**. However, this will cause a potentially unpredictable number of simultaneously executing thread(s) **1932** when many MSs are in the vicinity.

Thread(s) **1932** are useful when one or both parties to WDR transmission (sending and receiving MS) do not have NTP enabled. TDOA measurements are taken to triangulate the MS relative other MSs in real time.

FIG. **25** depicts a flowchart for describing a preferred embodiment of MS WDR request processing, for example when a remote MS requests (e.g. from FIG. **22** or **23**) a WDR. Receive processing identifies targeted requests destined (e.g. FIG. **23**) for the MS of FIG. **25** processing, and identifies general broadcasts (e.g. FIG. **22**) for processing as well. FIG. **25** processing describes a process **1942** worker thread, and is of PIP code **6**. Thread(s) **1942** purpose is for the MS of FIG. **25** processing to respond to incoming WDR requests. It is recommended that validity criteria set at block **1444** for **1942**-Max be set as high as possible (e.g. 10) relative performance considerations of architecture **1900**, to service multiple WDR requests simultaneously. Multiple channels for receiving data fed to queue **26** should be isolated to modular receive processing.

In an alternative embodiment having multiple receiving transmission channels visible to process **1942**, there can be a worker thread **1942** per channel to handle receiving on multiple channels simultaneously. If thread(s) **1942** do not receive directly from the channel, the preferred embodiment of FIG. **25** would not need to convey channel information to thread(s) **1942** waiting on queue **24** anyway. Embodiments could allow specification/configuration of many thread(s) **1942** per channel.

Processing begins at block **2502**, continues to block **2504** where the process worker thread count **1942**-Ct is accessed and incremented by 1 (using appropriate semaphore access (e.g. **1942**-Sem)), and continues to block **2506** for retrieving from queue **26** a record **2490** (using interface **1948**), perhaps a special termination request entry, and only continues to block **2508** when a record **2490** is retrieved. Block **2506** stays blocked on retrieving from queue **26** until any record **2490** is retrieved. If block **2508** determines a special entry indicating to terminate was not found in queue **26**, processing continues to block **2510**. There are various embodiments for thread(s) **1912** and thread(s) **1942** to feed off a queue **26** for different record types, for example, separate queues **26**A and **26**B, or a thread target field with either record found at queue **26** (e.g. like field **2400***a*). In another embodiment, thread(s) **1912** are modified with logic of thread(s) **1942** to handle all records described for a queue **26**, since thread(s) **1912** are listening for queue **26** data anyway.

Block **2510** peeks the WDR queue **22** (using interface **1944**) for the most recent highest confidence entry for this MS whereabouts by searching queue **22** for: the MS ID field **1100***a* matching the MS ID of FIG. **25** processing, and a confidence field **1100***d* greater than or equal to the confidence floor value, and a most recent date/time stamp field **1100***b* within a prescribed trailing period of time of block **2510** search processing (e.g. 2 seconds). For example, block **2510** peeks the queue (i.e. makes a copy for use if an entry found for subsequent processing, but does not remove the entry from queue) for a WDR of the MS (of FIG. **25** processing) which has the greatest confidence over 75 and has been most recently inserted to queue **22** in the last 2 seconds. It is recommended that the trailing period of time used by block **2510** be never greater than a few seconds. Thread **1942** is of less value to the LN-expanse when it responds with outdated/invalid whereabouts of the MS to facilitate locating other MSs. In an alternate embodiment, a movement tolerance (e.g. user configured or system set (e.g. 3 meters)) is incorporated at the MS, or at service(s) used to locate the MS, for knowing when the MS has significantly moved (e.g. more than 3 meters) and how long it has been (e.g. 45 seconds) since last significantly moving. In this embodiment, the MS is aware of the period of time since last significantly moving and the trailing period of time used by block **2510** is set using the amount of time since the MS significantly moved, or the amount of time since significantly moving, whichever is greater. This way a large number of (perhaps more confident candidate) WDRs are searched in the time period when the MS has not significantly moved. Optional blocks **278** through **284** may have been incorporated to FIG. **2**F for movement tolerance processing just described, in which case the LWT is compared to the current date/time to adjust the trailing period of time used by block **2510** for the correct trailing period. In any case, a WDR is sought at block **2510** to satisfy a request helping another MS in the LN-expanse locate itself.

Thereafter, if block **2512** determines a useful WDR was not found, then processing continues back to block **2506** for another loop iteration of processing an inbound WDR request. If block **2512** determines a useful WDR was found, then block **2514** prepares the WDR for send processing with correlation field **1100***m* set from correlation field **2490***b* retrieved at block **2506**, and block **2516** sends/broadcasts (per field **2490***a*) the WDR information (using send interface **1946**) by inserting to queue **24** so that send processing transmits data **1302**, for example as far as radius **1306**, and processing continues back to block **2506**. At least fields **1100***b*, **1100***c*, **1100***d*, **1100***m* and **1100***n* are sent/broadcast. See FIG. **11**A descriptions. Fields are set to the following upon exit from block **2514**:

MS ID field **1100***a* is preferably set with: Field **2490***a* from queue **26**.

DATE/TIME STAMP field **1100***b* is preferably set with: Field **1100***b* from queue **22**.

LOCATION field **1100***c* is preferably set with: Field **1100***c* from queue **22**.

CONFIDENCE field **1100***d* is preferably set with: Field **1100***d* from queue **22**.

LOCATION TECHNOLOGY field **1100***e* is preferably set with: Field **1100***e* from queue **22**.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: null (not set) for Broadcast by send processing, otherwise set to field **2490***d* for Send by send processing.

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: null (not set).

SPEED field **1100***h* is preferably set with: Field **1100***h* from queue **22**.

HEADING field **1100***i* is preferably set with: Field **1100***i* from queue **22**.

ELEVATION field **1100***j* is preferably set with: Field **1100***j* from queue **22**.

APPLICATION FIELDS field **1100***k* is preferably set with: Field **1100***k* from queue **22**. An alternate embodiment will add, alter, or discard data (with or without date/time stamps) here at the time of block **2514** processing.

CORRELATION FIELD **1100***m* is preferably set with: Field **2490***b* from queue **26**.

SENT DATE/TIME STAMP field **1100***n* is preferably set with: Sent date/time stamp as close in processing the send/broadcast of block **2516** as possible.

RECEIVED DATE/TIME STAMP field **1100***p* is preferably set with: Not Applicable (i.e. N/A for sending).

Embodiments may rely completely on the correlation field **2490***b* with no need for field **2490***a*. Referring back to block **2508**, if a worker thread termination request was found at queue **26**, then block **2518** decrements the worker thread count by 1 (using appropriate semaphore access (e.g. **1942**-Sem)), and thread **1942** processing terminates at block **2520**. Block **2518** may also check the **1942**-Ct value, and signal the process **1942** parent thread that all worker threads are terminated when **1942**-Ct equals zero (0).

Block **2516** causes sending/broadcasting data **1302** containing CK **1304**, depending on the type of MS, wherein CK **1304** contains WDR information prepared as described above for block **2514**. Alternative embodiments of block **2510** may not search a specified confidence value, and broadcast the best entry available anyway so that listeners in the vicinity will decide what to do with it. A semaphore protected data access (instead of a queue peek) may be used in embodiments where there is always one WDR current entry maintained for the MS.

In the embodiment wherein usual MS communications data **1302** of the MS is altered to contain CK **1304** for listening MSs in the vicinity, send processing feeding from queue **24**, caused by block **2516** processing, will place WDR information as CK **1304** embedded in usual data **1302** at the next opportune time of sending usual data **1302**. If an opportune time is not timely, send processing should discard the send request of block **2516** to avoid broadcasting outdated whereabouts information (unless using a movement tolerance and time since last significant movement). As the MS conducts its normal communications, transmitted data **1302** contains new data CK **1304** to be ignored by receiving MS other character **32** processing, but to be found by listening MSs within the vicinity which anticipate presence of CK **1304**. Otherwise, when LN-Expanse deployments have not introduced CK **1304** to usual data **1302** communicated on a receivable signal by MSs in the vicinity, FIG. **25** sends/broadcasts new WDR response data **1302**. In any case, field **1100***n* should be as accurate as possible for when data **1302** is actually sent. Critical regions of code (i.e. prevent thread preemption) and/or anticipated execution timing may be used to affect a best setting of field **1100***n*.

In an alternate embodiment, records **2490** contain a sent date/time stamp field **2490***e* of when the request was sent by a remote MS, and the received date/time stamp field **2490***c* is processed at the MS in FIG. **25** processing. This would enable block **2514** to calculate a TDOA measurement for returning in field **1100***f* of the WDR sent/broadcast at block **2516**.

FIG. **26**A depicts a flowchart for describing a preferred embodiment of MS whereabouts determination processing. FIG. **26**A processing describes a process **1952** worker thread, and is of PIP code **6**. Thread(s) **1952** purpose is for the MS of FIG. **26**A processing to determine its own whereabouts with useful WDRs from other MSs. It is recommended that validity criteria set at block **1444** for **1952**-Max be set as high as possible (e.g. 10) relative performance considerations of architecture **1900**, to service multiple threads **1912**. **1952**-Max may also be set depending on what DLM capability exists for the MS of FIG. **26**A processing. In an alternate embodiment, thread(s) 19xx are automatically throttled up or down (e.g. **1952**-Max) per unique requirements of the MS as it travels.

Processing begins at block **2602**, continues to block **2604** where the process worker thread count **1952**-Ct is accessed and incremented by 1 (using appropriate semaphore access (e.g. **1952**-Sem)), and continues to block **2606** for interim housekeeping of pruning the WDR queue by invoking a Prune Queues procedure of FIG. **27**. Block **2604** may also check the **1952**-Ct value, and signal the process **1952** parent thread that all worker threads are running when **1952**-Ct reaches **1952**-Max. Block **2606** may not be necessary since pruning may be accomplished at block **2620** when invoking FIG. **2**F (block **292**).

Thereafter, block **2608** retrieves from queue **1980** a record **2400** (using interface **1958**), perhaps a special termination request entry, or a record **2400** received from thread(s) **1912**, and only continues to block **2610** when a record **2400** containing field **2400***a* set to **1952** has been retrieved. Block **2608** stays blocked on retrieving from queue **1980** until a record **2400** with field **2400***a*=**1952** is retrieved. If block **2610** determines a special entry indicating to terminate was not found in queue **1980**, processing continues to block **2612**.

Block **2612** peeks the WDR queue **22** (using interface **1954**) for the most recent highest confidence entry for this MS whereabouts by searching queue **22** for: the MS ID field **1100***a* matching the MS ID of FIG. **26**A processing, and a confidence field **1100***d* greater than or equal to the confidence floor value, and a most recent date/time stamp field **1100***b* within a prescribed trailing period of time of block **2612** search processing using a f(WTV) for the period. For example, block **2612** peeks the queue (i.e. makes a copy for use if an entry found for subsequent processing, but does not remove the entry from queue) for a WDR of the MS (of FIG. **26**A processing) which has the greatest confidence over 75 and has been most recently inserted to queue **22** in the last 2 seconds. Since MS whereabouts accuracy may be dependent on timeliness of the WTV, it is recommended that the f(WTV) be some value less than or equal to WTV. In an alternate embodiment, a movement tolerance (e.g. user configured or system set (e.g. 3 meters)) is incorporated at the MS, or at service(s) used to locate the MS, for knowing when the MS has significantly moved (e.g. more than 3 meters) and how long it has been (e.g. 45 seconds) since last significantly moving. In this embodiment, the MS is aware of the period of time since last significantly moving and the f(WTV) is set using the amount of time since the MS significantly moved (i.e. f(WTV)=as described above, or the amount of time since significantly moving, whichever is greater). This way a large number of (perhaps more confident candidate) WDRs are searched in the time period when the MS has not significantly moved. Optional blocks **278** through **284** may have been incorporated to FIG. **2**F for movement tolerance processing just described, in which case the LWT is compared to the current date/time to adjust the WTV for the correct trailing period.

Thereafter, if block **2614** determines a timely whereabouts for this MS already exists to queue **22** (current WDR found), then processing continues back to block **2606** for another loop iteration of processing. If **2614** determines a satisfactory WDR does not already exist in queue **22**, then block **2600** determines a new highest confidence WDR for this MS (FIG. **26**B processing) using queue **22**.

Thereafter, if block **2616** determines a WDR was not created (BESTWDR variable=null) for the MS of FIG. **26**A processing (by block **2600**), then processing continues back to block **2606**. If block **2616** determines a WDR was created (BESTWDR=WDR created by FIG. **26**B) for the MS of FIG. **26**A processing by block **2600**, then processing continues to block **2618** for preparing FIG. **2**F parameters and FIG. **2**F processing is invoked with the new WDR at block **2620** (for interface **1956**) before continuing back to block **2606**. Parameters set at block **2618** are: WDRREF=a reference or pointer to the WDR completed at block **2600**; DELETEQ=FIG. **26**A location queue discard processing; and SUPER=FIG. **26**A supervisory notification processing.

Referring back to block **2610**, if a worker thread termination request was found at queue **1980**, then block **2622** decrements the worker thread count by 1 (using appropriate semaphore access (e.g. **1952**-Sem)), and thread **1952** processing terminates at block **2624**. Block **2622** may also check the **1952**-Ct value, and signal the process **1952** parent thread that all worker threads are terminated when **1952**-Ct equals zero (0).

Alternate embodiments to FIG. **26**A will have a pool of thread(s) **1952** per location technology (WDR field **1100***e*) for specific WDR field(s) selective processing. FIG. **26**A processing is shown to be generic with handling all WDRs at block **2600**.

FIG. **26**B depicts a flowchart for describing a preferred embodiment of processing for determining a highest possible confidence whereabouts, for example in ILM processing, such as processing of FIG. **26**A block **2600**. Processing starts at block **2630**, and continues to block **2632** where variables are initialized (BESTWDR=null, THIS_MS=null, REMOTE_MS=null). BESTWDR will reference the highest confidence WDR for whereabouts of the MS of FIG. **26**B processing (i.e. this MS) upon return to FIG. **26**A when whereabouts determination is successful, otherwise BESTWDR is set to null (none found). THIS_MS points to an appropriately sorted list of WDRs which were originated by this MS and are DLM originated (i.e. inserted by the DLM of FIG. **26**B processing). REMOTE_MS points to an appropriately sorted list of WDRs which were originated by other MSs (i.e. from DLMs and/or ILMs and collected by the ILM of FIG. **26**B processing).

Thereafter, block **2634** peeks the WDR queue **22** (using interface **1954**) for most recent WDRs by searching queue **22** for: confidence field **1100***d* greater than or equal to the confidence floor value, and a most recent date/time stamp field **1100***b* within a prescribed trailing period of time of block **2634** search processing using a f(WTV) for the period. For example, block **2634** peeks the queue (i.e. makes a copy of all WDRs to a result list for use if any found for subsequent processing, but does not remove the entry(s) from queue) for all WDRs which have confidence over 75 and has been most recently inserted to queue **22** in the last 2 seconds. It is recommended that the f(WTV) used here be some value less than or equal to the WTV (want to be ahead of curve, so may use a percentage (e.g. 90%)), but preferably not greater than a couple/few seconds (depends on MS, MS applications, MS environment, whereabouts determination related variables, etc).

In an alternative embodiment, thread(s) **1952** coordinate with each other to know successes, failures or progress of their sister threads for automatically adjusting the trailing f(WTV) period of time appropriately. See "Alternative IPC Embodiments" below.

Thread **1952** is of less value to the MS when whereabouts are calculated using stale WDRs, or when not enough useful

WDRs are considered. In an alternate embodiment, a movement tolerance (e.g. user configured or system set (e.g. 3 meters)) is incorporated at the MS, or at service(s) used to locate the MS, for knowing when the MS has significantly moved (e.g. more than 3 meters) and how long it has been (e.g. 45 seconds) since last significantly moving. In this embodiment, the MS is aware of the period of time since last significantly moving and the f(WTV) is set using the amount of time since the MS significantly moved (i.e. f(WTV)=as described above, or the amount of time since significantly moving, whichever is greater). This way a large number of (perhaps more confident candidates) WDRs are searched in the time period when the MS has not significantly moved. Optional blocks **278** through **284** may have been incorporated to FIG. **2**F for movement tolerance processing just described, in which case the LWT is compared to the current date/time to adjust the WTV for the correct trailing period. In any case, all useful WDRs are sought at block **2634** and placed into a list upon exit from block **2634**.

Thereafter, block **2636** sets THIS_MS list and REMOTE_MS list sort keys to be used at blocks **2644** and **2654**. Blocks **2638** through **2654** will prioritize WDRs found at block **2634** depending on the sort keys made at block **2636**. A number of variables may be used to determine the best sort keys, such as the time period used to peek at block **2634** and/or the number of entries in the WDR list returned by block **2634**, and/or other variables. When the time period of search is small (e.g. less than a couple seconds), lists (THIS_MS and REMOTE_MS) should be prioritized primarily by confidence (fields **1100***d*) since any WDRs are valuable for determining whereabouts. This is the preferred embodiment.

When the time period is great, careful measure must be taken to ensure stale WDRs are not used (e.g. >few seconds, and not considering movement tolerance). Depending on decision embodiments, there will be preferred priority order sort keys created at exit from block **2636**, for example "key1/key2/key3" implies that "key1" is a primary key, "key2" is a second order key, and "key3" is a third order key. A key such as "field-**1100***b*/field-**1100***d*/field-**1100***f*:signal-strength" would sort WDRs first by using date/time stamp fields **1100***b*, then by confidence value fields **1100***d* (sorted within matching date/time stamp WDRs), then by signal-strength field **1100***f* sub-field values (sorted within matching WDR confidences; no signal strength present=lowest priority). Another sort key may be "field-**1100***d*/field-**1100***b*" for sorting WDRs first by using confidence values, then by date/time stamps (sorted within matching WDR confidences). The same or different sort keys can be used for lists THIS_MS and REMOTE_MS. Any WDR data (fields or subfields) can be sorted with a key, and sort keys can be of N order dimension such that "key1/key2/ . . . /keyN". Whatever sort keys are used, block **2686** will have to consider confidence versus being stale, relative to the WTV. In the preferred embodiment, the REMOTE_MS and THIS_MS lists are set with the same sort keys of "field-**1100***d*/field-**1100***b*" (i.e. peek time period used at block **2634** is less than 2 seconds) so that confidence is primary.

Thereafter, block **2638** gets the first (if any) WDR in the list returned at block **2634** (also processes next WDR in list when encountered again in loop of blocks **2638** through **2654**), and block **2640** checks if all WDRs have already been processed. If block **2640** finds that all WDRs have not been processed, then block **2642** checks the WDR origination. If block **2642** determines the WDR is one that originated from a remote MS (i.e. MS ID does not match the MS of FIG. **26**B processing), then block **2644** inserts the WDR into the REMOTE_MS list using the desired sort key (confidence primary, time secondary) from block **2636**, and processing continues to block **2638** for another loop iteration. If block **2642** determines the WDR is one that originated from this MS (MS ID field **1100***a* matches the MS of FIG. **26**B processing (e.g. this MS being a DLM at the time of WDR creation (this MS ID=field **1100***a*) or this MS being an ILM at the time of WDR creation (previous processing of FIG. **26**A)), then processing continues to block **2646** to determine how to process the WDR which was inserted by "this MS" for its own whereabouts.

Block **2646** accesses field **1100***f* for data found there (e.g. FIGS. **2**D and **2**E may have inserted useful TDOA measurements, even though DLM processing occurred; or FIG. **3**C may have inserted useful TDOA and/or AOA measurements with reference station(s) whereabouts; or receive processing may have inserted AOA and related measurements). Thereafter, if block **2648** determines presence of TDOA and/or AOA data, block **2650** checks if reference whereabouts (e.g. FIG. **3**C selected stationary reference location(s)) is also stored in field **1100***f*. If block **2650** determines whereabouts information is also stored to field **1100***f*, then block **2652** makes new WDR(s) from the whereabouts information containing at least the WDR Core and field **1100***f* containing the AOA and/or TDOA information as though it were from a remote DLM or ILM. Block **2652** also performs the expected result of inserting the WDR of loop processing into the THIS_MS list using the desired sort key from block **2636**. Processing then continues to block **2644** where the newly made WDR(s) is inserted into the REMOTE_MS list using the desired sort key (confidence primary, time secondary) from block **2636**. Block **2644** continues back to block **2638**.

Block **2646** through **2652** show that DLM stationary references may contribute to determining whereabouts of the MS of FIG. **26**B processing by making such references appear to processing like remote MSs with known whereabouts. Any DLM location technology processing discussed above can facilitate FIG. **26**B whereabouts processing when reference whereabouts can be maintained to field **1100***f* along with relative AOA, TDOA, MPT, confidence, and/or other useful information for locating the MS. Various embodiments will populate field **1100***f* wherever possible with any useful locating fields (see data discussed for field **1100***f* with FIG. **11**A discussions above) for carrying plenty of information to facilitate FIG. **26**B processing.

Referring back to block **2650**, if it is determined that whereabouts information was not present with the AOA and/or TDOA information of field **1100***f*, then processing continues to block **2644** for inserting into the REMOTE_MS list (appropriately with sort key from block **2636**) the currently looped WDR from block **2634**. In-range location technology associates the MS with the antenna (or cell tower) location, so that field **1100***c* already contains the antenna (or cell tower) whereabouts, and the TDOA information was stored to determine how close the MS was to the antenna (or cell tower) at the time. The WDR will be more useful in the REMOTE_MS list, then if added to the THIS_MS list (see loop of blocks **2660** through **2680**). Referring back to block **2648**, if it is determined that no AOA and/or TDOA information was in field **1100***f*, then processing continues to block **2654** for inserting the WDR into the THIS_MS list (appropriately with sort key (confidence primary, time secondary) from block **2636**).

Block **2654** handles WDRs that originated from the MS of FIG. **26**B (this MS), such as described in FIGS. **2**A through **9**B, or results from previous FIG. **26**A processing. Block **2644** maintains remote DLMs and/or ILMs (their whereabouts) to the REMOTE_MS list in hope WDRs contain useful field **1100*f*** information for determining the whereabouts of the MS of FIG. **26**B processing. Block **2652** handles WDRs that originated from the MS of FIG. **26**B processing (this MS), but also processes fields from stationary references used (e.g. FIG. **3**C) by this MS which can be helpful as though the WDR was originated by a remote ILM or DLM. Thus, block **2652** causes inserting to both lists (THIS_MS and REMOTE_MS) when the WDR contains useful information for both. Blocks **2652**, **2654** and **2644** cause the iterative loop of blocks **2660** through **2680** to perform ADLT using DLMs and/or ILMs. Alternate embodiments of blocks **2638** through **2654** may use peek methodologies to sort from queue **22** for the REMOTE_MS and THIS_MS lists.

Referring back to block **2640**, if it is determined that all WDRs in the list from block **2634** have been processed, then block **2656** initializes a DISTANCE list and ANGLE list each to null, block **2658** sets a loop iteration pointer to the first entry of the prioritized REMOTE_MS list (e.g. first entry higher priority then last entry in accordance with sort key used), and block **2660** starts the loop for working with ordered WDRs of the REMOTE_MS list. Exit from block **2640** to block **2656** occurs when the REMOTE_MS and THIS_MS lists are in the desired priority order for subsequent processing. Block **2660** gets the next (or first) REMOTE_MS list entry for processing before continuing to block **2662**. If block **2662** determines all WDRs have not yet been processed from the REMOTE_MS list, then processing continues to block **2664**.

Blocks **2664** and **2670** direct collection of all useful ILM triangulation measurements for TDOA, AOA, and/or MPT triangulation of this MS relative known whereabouts (e.g. other MSs). It is interesting to note that TDOA and AOA measurements (field **1100*f***) may have been made from different communications interfaces **70** (e.g. different wave spectrums), depending on interfaces the MS has available (i.e. all can participate). For example, a MS with blue-tooth, WiFi and cellular phone connectivity (different class wave spectrums supported) can be triangulated using the best available information (i.e. heterogeneous location technique). Examination of fields **1100*f*** in FIG. **17** can show wave spectrums (and/or particular communications interfaces **70**) inserted by receive processing for what the MS supports. If block **2664** determines an AOA measurement is present (field **1100*f*** sub-field), then block **2666** appends the WDR to the ANGLE list, and processing continues to block **2668**. If block **2664** determines an AOA measurement is not present, then processing continues to block **2670**. If block **2670** determines a TDOA measurement is present (field **1100*f*** sub-field), then block **2672** appends the WDR to the DISTANCE list, and processing continues to block **2674**. Block **2674** uses WDRs for providing at least an in-range whereabouts of this MS by inserting to the THIS_MS list in sorted confidence priority order (e.g. highest confidence first in list, lowest confidence at end of list). Block **2674** continues to block **2668**. Block **2674** may cause duplicate WDR(s) inserted to the THIS_MS list, but this will have no negative effect on selected outcome.

Block **2668** compares the ANGLE and DISTANCE lists constructed thus far from loop processing (blocks **2660** through **2680**) with minimum triangulation requirements (e.g. see "Missing Part Triangulation (MPT)" above). Three (3) sides, three (3) angles and a side, and other known triangular solution guides will also be compared. Thereafter, if block **2676** determines there is still not enough data to triangulate whereabouts of this MS, then processing continues back to block **2660** for the next REMOTE_MS list entry, otherwise block **2678** maximizes diversity of WDRs to use for triangulating. Thereafter, block **2680** uses the diversified DISTANCE and ANGLE lists to perform triangulation of this MS, block **2682** inserts the newly determined WDR into the THIS_MS list in sort key order, and continues back to block **2660**. Block **2680** will use heterogeneous (MPT), TDOA and/or AOA triangulation on ANGLE and DISTANCE lists for determining whereabouts.

Block **2682** preferably keeps track of (or checks THIS_MS for) what it has thus far determined whereabouts for in this FIG. **26**B thread processing to prevent inserting the same WDR to THIS_MS using the same REMOTE_MS data. Repeated iterations of blocks **2676** through **2682** will see the same data from previous iterations and will use the best of breed data in conjunction with each other at each iteration (in current thread context). While inserting duplicates to THIS_MS at block **2682** does not cause failure, it may be avoided for performance reasons. Duplicate insertions are preferably avoided at block **2674** for performance reasons as well, but they are again not harmful. Block **2678** preferably keeps track of previous diversity order in this FIG. **26**B thread processing to promote using new ANGLE and DISTANCE data in whereabouts determination at block **2680** (since each iteration is a superset of a previous iteration (in current thread context). Block **2678** promotes using WDRs from different MSs (different MS IDs), and from MSs located at significantly different whereabouts (e.g. to maximize surrounded-ness), preferably around the MS of FIG. **26**B processing. Block **2678** preferably uses sorted diversity pointer lists so as to not affect actual ANGLE and DISTANCE list order. The sorted pointer lists provide pointers to entries in the ANGLE and DISTANCE lists for a unique sorted order governing optimal processing at block **2680** to maximize unique MSs and surrounded-ness, without affecting the lists themselves (like a SQL database index). Different embodiments of blocks **2678** through **2682** should minimize inserting duplicate WDRs (for performance reasons) to THIS_MS which were determined using identical REMOTE_MS list data. Block **2682** causes using ADLT at blocks **2684** through **2688** which uses the best of breed whereabouts, either as originated by this MS maintained in THIS_MS list up to the thread processing point of block **2686**, or as originated by remote MSs (DLMs and/or ILMs) processed by blocks **2656** through the start of block **2684**.

Referring back to block **2662**, if it is determined that all WDRs in the REMOTE_MS list have been processed, then block **2684** sets the BESTWDR reference to the head of THIS_MS (i.e. BESTWDR references first WDR in THIS_MS list which is so far the best candidate WDR (highest confidence) for this MS whereabouts, or null if the list is empty). It is possible that there are other WDRs with matching confidence adjacent to the highest confidence entry in the THIS_MS list. Block **2684** continues to block **2686** for comparing matching confidence WDRs, and if there are matches, then breaking a tie between WDRs with matching confidence by consulting any other WDR field(s) (e.g. field **1100*f*** signal strength, or location technology field **1100*e***, etc). If there is still a tie between a plurality of WDRs, then block **2686** may average whereabouts to the BESTWDR WDR using the matching WDRs. Thereafter processing continues to block **2688** where the BESTWDR is completed, and processing terminates at block **2690**. Block **2688** also frees resources (if any) allocated by FIG. **26**B processing (e.g. lists). Blocks **2686** through **2688** result in setting BESTWDR to the highest priority WDR (i.e. the best possible whereabouts determined). It is possible that FIG. **26**B processing causes a duplicate WDR inserted to queue **22** (at block **2620**) for this MS whereabouts determination, but that is no issue except for impacting performance to queue **22**. An alternate embodiment to queue **22** may define a unique index for erring out when inserting a duplicate to prevent frivolous duplicate entries, or block **2688** will incorporate processing to eliminate the chance of inserting a WDR of less use than what is already contained at queue **22**. Therefore, block **2688** may include processing for ensuring a duplicate will not be inserted (e.g. null the BESTWDR reference) prior to returning to FIG. **26**A at block **2690**.

Averaging whereabouts at block **2686** occurs only when there are WDRs at the head of the list with a matching highest confidence value and still tie in other WDR fields consulted, yet whereabouts information is different. In this case, all matching highest confidence whereabouts are averaged to the BESTWDR to come up with whereabouts in light of all matching WDRs. Block **2686** performs ADLT when finalizing a single whereabouts (WDR) using any of the whereabouts found in THIS_MS (which may contain at this point DLM whereabouts originated by this MS and/or whereabouts originated by remote DLMs and/or ILMs). Block **2686** must be cognizant of sort keys used at blocks **2652** and **2654** in case confidence is not the primary key (time may be primary).

If no WDRs were found at block **2634**, or no THIS_MS list WDRs were found at blocks **2652** and **2654**, and no REMOTE_MS list entries were found at block **2644**; or no THIS_MS list WDRs were found at blocks **2652** and **2654**, and no REMOTE_MS list entries were found useful at blocks **2664** and/or **2670**; then block **2684** may be setting BESTWDR to a null reference (i.e. none in list) in which case block **2686** does nothing. Hopefully, at least one good WDR is determined for MS whereabouts and a new WDR is inserted for this MS to queue **22**, otherwise a null BESTWDR reference will be returned (checked at block **2616**). See FIG. **11**A descriptions. If BESTWDR is not null, then fields are set to the following upon exit from block **2688**:

MS ID field **1100***a* is preferably set with: MS ID of MS of FIG. **26**B processing.

DATE/TIME STAMP field **1100***b* is preferably set with: Date/time stamp of block **2688** processing.

LOCATION field **1100***c* is preferably set with: Resulting whereabouts after block **2688** completion.

CONFIDENCE field **1100***d* is preferably set with: WDR Confidence at THIS_MS list head.

LOCATION TECHNOLOGY field **1100***e* is preferably set with: "ILM TDOA Triangulation", "ILM AOA Triangulation", "ILM MPT Triangulation" or "ILM in-range", as determined by the WDRs inserted to MS_LIST at blocks **2674** and **2682**. The originator indicator is set to ILM.

LOCATION REFERENCE INFO field **1100***f* is preferably set with: null (not set), but may be set with contributing data for analysis of queue **22** provided it is marked for being overlooked by future processing of blocks **2646** and **2648** (e.g. for debug purpose).

COMMUNICATIONS REFERENCE INFO field **1100***g* is preferably set with: null (not set).

SPEED field **1100***h* is preferably set with: Block **2688** may compare prioritized entries and their order of time (field **1100***b*) in THIS_MS list for properly setting this field, if possible.

HEADING field **1100***i* is preferably set with: null (not set). Block **2688** may compare prioritized entries and their order of time (field **1100***b*) in THIS_MS list for properly setting this field, if possible.

ELEVATION field **1100***j* is preferably set with: Field **1100***j* of BESTWDR (may be averaged if WDR tie(s)), if available.

APPLICATION FIELDS field **1100***k* is preferably set with: Field(s) **1100***k* from BESTWDR or tie(s) thereof from THIS_MS. An alternate embodiment will add, alter, or discard data (with or without date/time stamps) here at the time of block **2688** processing.

CORRELATION FIELD **1100***m* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

SENT DATE/TIME STAMP field **1100***n* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

RECEIVED DATE/TIME STAMP field **1100***p* is preferably set with: Not Applicable (i.e. not maintained to queue **22**).

Block **2680** determines whereabouts using preferred guidelines, such as whereabouts determined never results in a confidence value exceeding any confidence value used to determine whereabouts. Some embodiments will use the mean (average) of confidence values used, some will use the highest, and some the lowest of the WDRs used. Preferred embodiments tend to properly skew confidence values to lower values as the LN-Expanse grows away from region **1022**. Blocks **2668** through **2680** may consult any of the WDR fields (e.g. field **1100***f* sub-fields yaw, pitch, roll; speed, heading, etc) to deduce the most useful WDR inputs for determining an optimal WDR for this MS whereabouts.

Alternative IPC Embodiments

Thread(s) **1952** are started for every WDR collected from remote MSs. Therefore, it is possible that identical new WDRs are inserted to queue **22** using the same WDR information at blocks **2634** of simultaneously executing threads **1952**, but this will not cause a problem since at least one will be found when needed, and duplicates will be pruned together when appropriate. Alternative embodiments provide IPC (Interprocess Communications Processing) coordination between **1952** threads for higher performance processing, for example:

- As mentioned above, thread(s) **1952** can coordinate with each other to know successes, failures or progress of their sister **1952** thread(s) for automatically adjusting the trailing f(WTV) period of time appropriately. The f(WTV) period of time used at block **2634** would be semaphore accessed and modified (e.g. increased) for another **1952** thread when a previous **1952** thread was unsuccessful in determining whereabouts (via semaphore accessed thread outcome indicator). After a successful determination, the f(WTV) period of time could be reset back to the smaller window. One embodiment of increasing may start with 10% of the WTV, then 20% at the next thread, 30% at the next thread, up to 90%, until a successful whereabouts is determined. After successful whereabouts determination, a reset to its original starting value is made.
- A semaphore accessed thread **1952** busy flag is used for indicating a certain thread is busy to prevent another **1952** thread from doing the same or similar work. Furthermore, other semaphore protected data for what work is actually being performed by a thread can be informative to ensure that no thread **1952** starts for doing duplicated effort.
- Useful data of statistics **14** may be appropriately accessed by thread(s) **1952** for dynamically controlling key variables of FIG. **26**B processing, such as the search f(WTV) time period, sort keys used, when to quit loop processing (e.g. on first successful whereabouts determination at block **2680**), surrounded-ness preferences, etc. This can dynamically change the FIG. **26**B logic from one thread to another for desired results.

FIG. **26**B continues processing through every WDR retrieved at block **2634**. An alternative embodiment will terminate processing after finding the first (which is highest priority data supported) successful triangulation at block **2682**.

FIG. **27** depicts a flowchart for describing a preferred embodiment of queue prune processing. Queue pruning is best done on an interim basis by threads which may insert to the queue being pruned. In an alternate embodiment, a background asynchronous thread will invoke FIG. **27** for periodic queue pruning to ensure no queue which can grow becomes too large. The Prune Queues procedure starts at block **2702** and continues to block **2704** where parameters passed by a caller for which queue(s) (WDR and/or CR) to prune are determined. Thereafter, if block **2706** determines that the caller wanted to prune the WDR queue **22**, block **2708** appropriately prunes the queue, for example discarding old entries using field **1100***b*, and processing continues to block **2710**. If block **2706** determines that the caller did not want to prune the WDR queue **22**, then processing continues to block **2710**. If block **2710** determines that the caller wanted to prune the CR queue **1990**, block **2712** appropriately prunes the queue, for example discarding old entries using field **2450***a*, and processing continues to block **2714**. If block **2710** determines that the caller did not want to prune the CR queue **1990**, then processing continues to block **2714**. Block **2714** appropriately returns to the caller.

The current design for queue **1980** does not require FIG. **27** to prune it. Alternative embodiments may add additional queues for similar processing. Alternate embodiments may use FIG. **27** like processing to prune queues **24**, **26**, or any other queue under certain system circumstances. Parameters received at block **2704** may also include how to prune the queue, for example when using different constraints for what indicates entry(s) for discard.

FIG. **28** depicts a flowchart for describing a preferred embodiment of MS termination processing. Depending on the MS, there are many embodiments of processing when the MS is powered off, restarted, rebooted, reactivated, disabled, or the like. FIG. **28** describes the blocks of processing relevant to the present disclosure as part of that termination processing. Termination processing starts at block **2802** and continues to block **2804** for checking any DLM roles enabled and appropriately terminating if any are found (for example as determined from persistent storage variable DLMV). Block **2804** may cause the termination of thread(s) associated with enabled DLM role(s) for DLM processing above (e.g. FIGS. **2**A through **9**B). Block **2804** may invoke API(s), disable flag(s), or terminate as is appropriate for DLM processing described above. Such terminations are well known in the art of prior art DLM capabilities described above. Block **2804** continues to block **2806**.

Blocks **2806** through **2816** handle termination of all processes/threads associated with the ILMV roles so there is no explicit ILMV check required. Block **2806** initializes an enumerated process name array for convenient processing reference of associated process specific variables described in FIG. **19**, and continues to block **2808** where the first member of the set is accessed for subsequent processing. The enumerated set of process names has a prescribed termination order for MS architecture **1900**. Thereafter, if block **2810** determines the process identifier (i.e. 19xx-PID such that 19xx is **1902**, **1912**, **1922**, **1932**, **1942**, **1952** in a loop iteration of blocks **2808** through **2816**) is greater than 0 (e.g. this first iteration of **1912**-PID>0 implies it is to be terminated here; also implies process **1912** is enabled as used in FIGS. **14**A, **28**, **29**A and **29**B), then block **2812** prepares parameters for FIG. **29**B invocation, and block **2814** invokes (calls) the procedure of FIG. **29**B to terminate the process (of this current loop iteration (19xx)). Block **2812** prepares the second parameter in accordance with the type of 19xx process. If the process (19xx) is one that is slave to a queue for dictating its processing (i.e. blocked on queue until queue entry present), then the second parameter (process type) is set to 0 (directing FIG. **29**A processing to insert a special termination queue entry to be seen by worker thread(s) for terminating). If the process (19xx) is one that is slave to a timer for dictating its processing (i.e. sleeps until it is time to process), then the second parameter (process type) is set to the associated 19xx-PID value (directing FIG. **29**B to use in killing/terminating the PID in case the worker thread(s) are currently sleeping). Block **2814** passes the process name and process type as parameters to FIG. **29**B processing. Upon return from FIG. **29**B, block **2814** continues to block **2816**. If block **2810** determines that the 19xx process is not enabled, then processing continues to block **2816**. Upon return from FIG. **29**B processing, the process is terminated and the associated 19xx-PID variable is already set to 0 (see blocks **2966**, **2970**, **2976** and **2922**).

Block **2816** checks if all process names of the enumerated set (19xx) have been processed (iterated) by blocks **2808** through **2816**. If block **2816** determines that not all process names in the set have been processed (iterated), then processing continues back to block **2808** for handling the next process name in the set. If block **2816** determines that all process names of the enumerated set were processed, then block **2816** continues to block **2818**.

Block **2818** destroys semaphore(s) created at block **1220**. Thereafter, block **2820** destroys queue(s) created at block **1218** (may have to remove all entries first in some embodiments), block **2822** saves persistent variables to persistent storage (for example to persistent storage **60**), block **2824** destroys shared memory created at block **1212**, and block **2826** checks the NTP use variable (saved prior to destroying shared memory at block **2824**).

If block **2826** determines NTP is enabled, then block **2828** terminates NTP appropriately (also see block **1612**) and processing continues to block **2830**. If block **2826** determines NTP was not enabled, then processing continues to block **2830**. Block **2828** embodiments are well known in the art of NTP implementations. Block **2828** may cause terminating of thread(s) associated with NTP use.

Block **2830** completes LBX character termination, then block **2832** completes other character **32** termination processing, and FIG. **28** processing terminates thereafter at block **2834**. Depending on what threads were started at block **1240**, block **2830** may terminate the listen/receive threads for feeding queue **26** and the send threads for sending data inserted to queue **24**. Depending on what threads were started at block **1206**, block **2832** may terminate the listen/receive threads for feeding queue **26** and the send threads for sending data inserted to queue **24** (i.e. other character **32** threads altered to cause embedded CK processing). Upon encounter of block **2834**, the MS is appropriately terminated for reasons at set forth above for invoking FIG. **28**.

With reference now to FIG. **29**B, depicted is a flowchart for describing a preferred embodiment of a procedure for terminating a process started by FIG. **29**A. When invoked by a caller, the procedure starts at block **2952** and continues to block **2954** where parameters passed are determined. There are two parameters: the process name to terminate, and the type of process to terminate. The type of process is set to 0 for a process which has worker threads which are a slave to a queue. The type of process is set to a valid O/S PID when the process worker threads are slave to a timer.

Thereafter, if block **2956** determines the process type is 0, then block **2958** initializes a loop variable J to 0, and block **2960** inserts a special termination request queue entry to the appropriate queue for the process worker thread to terminate. See FIG. **19** discussions for the queue inserted for which 19xx process name.

Thereafter, block **2962** increments the loop variable by 1 and block **2964** checks if all process prescribed worker threads have been terminated. Block **2964** accesses the 19xx-Max (e.g. **1952**-Max) variable from shared memory using a semaphore for determining the maximum number of threads to terminate in the process worker thread pool. If block **2964** determines all worker threads have been terminated, processing continues to block **2966** for waiting until the 19xx-PID variable is set to disabled (e.g. set to 0 by block **2922**), and then to block **2978** which causes return to the caller. Block **2966** uses a preferred choice of waiting described for blocks **2918** and **2920**. The 19xx process (e.g. **1952**) will have its 19xx-PID (e.g. **1952**-PID) variable set at 0 (block **2922**) when the process terminates. In some embodiments, the waiting methodology used at block **2966** may use the 19xx-PID variable, or may be signaled by the last terminating worker thread, or by block **2922**.

If block **2964** determines that not all worker threads have been terminated yet, then processing continues back to block **2960** to insert another special termination request queue entry to the appropriate queue for the next process worker thread to terminate. Blocks **2960** through **2964** insert the proper number of termination queue entries to the same queue so that all of the 19xx process worker threads terminate.

Referring back to block **2956**, if it is determined the process type is not 0 (i.e. is a valid O/S PID), then block **2968** inserts a special WDR queue **22** entry enabling a queue peek for worker thread termination. The reader will notice that the process termination order of block **2806** ensures processes which were slaves to the WDR queue **22** have already been terminated. This allows processes which are slaves to a timer to see the special termination queue entry inserted at block **2968** since no threads (which are slaves to queue) will remove it from queue **22**. Thereafter, block **2970** waits until the 19xx process name (parameter) worker threads have been terminated using a preferred choice of waiting described for blocks **2918** and **2920**. The 19xx process (e.g. **1902**) will have its 19xx-PID (e.g. **1902**-PID) variable set at 0 (block **2922**) when the process terminates. In some embodiments, the waiting methodology used at block **2970** may use the 19xx-PID variable, or may be signaled by the last terminating worker thread, or by block **2922**. Block **2970** also preferably waits for a reasonable timeout period in anticipation of known sleep time of the 19xx process being terminated, for cases where anticipated sleep times are excessive and the user should not have to wait for lengthy FIG. **28** termination processing. If the timeout occurs before the process is indicated to be terminated, then block **2970** will continue to block **2972**. Block **2970** also continues to block **2972** when the process has successfully terminated.

If block **2972** determines the 19xx process did terminate, the caller is returned to at block **2978** (i.e. 19xx-PID already set to disabled (0)). If block **2972** determines the 19xx process termination timed out, then block **2974** forces an appropriate O/S kill to the PID thereby forcing process termination, and block **2976** sets the 19xx-PID variable for disabled (i.e. process 19xx was terminated). Thereafter, block **2978** causes return to the caller.

There are many embodiments for setting certain queue entry field(s) identifying a special queue termination entry inserted at blocks **2960** and **2968**. Some suggestions: In the case of terminating thread(s) **1912**, queue **26** insertion of a WDR preferably sets the MS ID field with a value that will never appear in any other case except a termination request (e.g. −100). In the case of terminating thread(s) **1902**, **1922** and **1952**, queue **22** insertion of a WDR preferably sets the MS ID field with a value that will never appear in any other case except a termination request (e.g. −100). In the case of terminating thread(s) **1942**, queue **26** insertion of a WDR request preferably sets the MS ID field with a value that will never appear in any other case except a termination request (e.g. −100). In the case of terminating thread(s) **1932**, queue **1980** insertion of a thread request queue record **2400** preferably sets field **2400***a* with a value that will never appear in any other case except a termination request (e.g. −100). Of course, any available field(s) can be used to indicate termination to particular thread(s)).

Terminating threads of processing in FIG. **29**B has been presented from a software perspective, but there are hardware/firmware thread embodiments which may be terminated appropriately to accomplish the same functionality. If the MS operating system does not have an interface for killing the PID at block **2974**, then blocks **2972** through **2976** can be eliminated for relying on a FIG. **28** invocation timeout (incorporated for block **2814**) to appropriately rob power from remaining thread(s) of processing.

An ILM has many methods and systems for knowing its own location. LBX depends on MSs maintaining their own whereabouts. No service is required to maintain the whereabouts of MSs in order to accomplish novel functionality.

LBX: Permissions and Charters—Configuration

Armed with its own whereabouts, as well as whereabouts of others and others nearby, a MS uses charters for governing many of the peer to peer interactions. A user is preferably unaware of specificities of the layer(s) providing WDR interoperability and communications. Permissions **10** and charters **12** surface desired functionality to the MS user(s) without fully revealing the depth of features that could be made available. Permissions provide authentication for novel features and functionality, and to which context to apply the charters. However, some permissions can provide action(s), features, and functionality by themselves without a charter. It is preferred that LBX features and functionality be provided in the most elegant manner across heterogeneous MSs.

User configured permissions are maintained at a MS and their relevance (applicability) to WDRs that are being processed is determined. WDR processing events are recognized through being placed in strategic LBX processing paths of WDRs. For example, permissions govern processing of newly processed WDRs at a MS, regardless of where the WDR originated. A permission can provide at least one privilege, and may provide a plurality of privileges. A permission is granted from a grantor identity to a grantee identity. Depending on what permissions are determined relevant to (i.e. applicable to) a WDR being processed (e.g. by accessing at least one field in the WDR), an action or plurality of actions which are associated with the permission can automatically occur. Actions may be as simple as modifying a setting which is monitored/used by an LBX application, or as complex as causing many executable application actions for processing. User configured charters are maintained at a MS and their relevance applicability) to WDRs that are being processed is determined, preferably in context of the same recognized events (i.e. strategic processing paths) which are used for determining relevance of permissions to WDRs. A charter consists of a conditional expression and can have an action or plurality of actions which are associated with the expression. Upon evaluating the expression to an actionable condition (e.g. evaluates to a Boolean true result), the associated action(s) are invoked. Charters can be created for a MS by a user of that MS, or by a user of another MS. Charters are granted similarly to permissions in using a grantor and grantee identity, therefore granting a charter is equivalent to granting a permission to execute the charter.

While some embodiments will provide disclosed features as one at a time implementations, a comprehensive architecture is disclosed for providing a platform that will survive LBX maturity. FIGS. **30**A through **30**E depict a preferred embodiment BNF (Backus Naur Form) grammar for permissions **10** and charters **12**. A BNF grammar is an elegant method for describing the many applicable derived subset embodiments of syntax and semantics in carrying out processing behavior. The BNF grammar of FIGS. **30**A through **30**E specifically describes:

- Prescribed command languages, such as a programming language, for encoding/representing permissions **10** and charters **12** (e.g. a Whereabouts Programming Language (WPL));
- Prescribed configuration in a Lex & Yacc processing of a suitable encoding;
- Prescribed XML encodings/representations of permissions **10** and charters **12**;
- Prescribed communications datastream encodings/representations of permissions **10** and charters **12**, such as in an ANSI encoding standard (e.g. X.409);
- Prescribed internalized encodings/representations of permissions **10** and charters **12**, for example in a data processing memory;
- Prescribed internalized encodings/representations of permissions **10** and charters **12**, for example in a data processing storage means;
- Prescribed database schemas for encoding/representing permissions **10** and charters **12**;
- Prescribed semantics of constructs to carry out permissions **10** and charters **12**;
- A delimited set of constructs for defining different representative syntaxes for carrying out permissions **10** and charters **12**; and
- Prescribed data processing of interpreters and/or compilers for internalizing a syntax for useful semantics as disclosed herein.

There are many embodiments (e.g. BNF grammar subsets) of carrying out permissions **10** and charters **12** without departing from the spirit and scope of the present disclosure. A particular implementation will choose which derivative method and system to implement, and/or which subset of the BNF grammars shall apply. Atomic elements of the BNF grammar (leaf nodes of the grammar tree) are identified within double quotes (e.g. "text string" implies the value is an atomic element in text string form). Atomic elements are not constructs which elaborate to other things and/or types of data.

FIGS. **30**A through **30**B depict a preferred embodiment BNF grammar **3002*a*** through **3002*b*** for variables, variable instantiations and common grammar for BNF grammars of permissions **10**, groups (e.g. data **8**) and charters **12**. Variables are convenient for holding values that become instantiated where appropriate. This provides a rich programming language and/or macro nature to the BNF grammar. Variables can be set with: a) a typed value (i.e. value of a particular data type (may be a list)); b) another variable for indirect referencing; c) a plurality of typed values; d) a plurality of variable references; or e) any combinations of a) through d). Variables can appear anywhere in the permissions or charters encodings. When variables are referenced by name, they preferably resolve to the name of the variable (not the value). When variables are referenced by their name with an instantiation operator (e.g. *), the variable is instantiated (i.e. elaborated/resolved) to assigned value(s). Instantiation also provides a macro (or function) ability to optionally replace subset(s) (preferably string replacements) of the variable's instantiated value with parameter substitutions. This enables customizably instantiating values (i.e. optionally, string occurrences in the value are replaced with specified matching parameters). An alternate embodiment to string substitution is for supporting numbers to be incremented, decremented, or kept as is, depending on the substitution syntax. For example:

*myVar(555++,23−=4,888−−,200+=100)

This instantiation specifies that all occurrences of the string "555" should be incremented by 1 such that the first occurrence of "555" becomes "556", next occurrence of "555" becomes "557", and so on. Changing all occurrences of "555" to "556" is accomplished with the string substitution. This instantiation also specifies that all occurrences of the string "23" should be decremented by 4 such that the first occurrence of "23" becomes "19", next occurrence of "23" becomes "15", and so on. Changing all occurrences of "23" to "19" is accomplished with the string substitution. This instantiation also specifies that all occurrences of the string "888" should be decremented by 1 such that the first occurrence of "888" becomes "887", next occurrence of "888" becomes "886", and so on. Changing all occurrences of "888" to "887" is accomplished with the string substitution. This instantiation also specifies that all occurrences of the string "200" should be incremented by 100 such that the first occurrence of "200" becomes "300", next occurrence of "200" becomes "400", and so on. Changing all occurrences of "200" to "300" is accomplished with the string substitution.

Preferably, when a variable is set to another variable (e.g. a=b), an instantiation of the variable (i.e. *a) equals the variable b, not b's value (i.e. *(*a)=b's value). If the variable b is set to a variable c (e.g. b=c) in the example, and the variable a is set to the variable b as already described (past or future, prior to instantiation), and c was set (i.e. c=2) to the value 2 (past or future, prior to instantiation), then the preferred embodiment requires three (3) instantiations of variable a to get to the value assigned to variable c (e.g. *(*(*a)))=2). Instantiation of variable a (e.g. *a) preferably corresponds to a level of "peeling back" through the hierarchy of variable assignments if one exists. Alternative embodiments will allow a single instantiation of a variable to get through any number of indirect variable assignments for the first encountered value in the indirect chain value (e.g. *a=2) at the time of instantiation. Either semantic may have useful features from a programming standpoint. Over-instantiating (e.g. *(*c)=error) should cause an error. An assigned value is the leaf node in peeling back with instantiations.

The BNF Grammar "null" is an atomic element for no value. In a syntactic embodiment, a null value may be a special null character (e.g. Ø). The History construct is preferably used to track when certain constructs were created and last modified. An alternative embodiment will track all construct changes to LBX history **30** for later human, or automated, processing audit.

Grammar **3002*b*** "system type" is an atomic element (atomic elements are not constructs which elaborate to other things; atomic elements are shown delimited in double quotes) generalized for the type of MS (e.g. PDA, cell phone, laptop, etc). Other embodiments will provide more detail to the type of MS (e.g. iPhone, Blackberry Pearl, Nextel i845, Nokia 741, etc). ID is an identity construct of the present disclosure for identifying a MS, a user, a group, or any other entity for which to associate data and/or processing. IDType provides the type of ID to support a heterogeneous identifying grammar. An identity (i.e. ID [IDType]) can be directly associated to a MS (e.g. MS ID), or may be indirectly associated to a MS (e.g. user ID or group ID of the MS). Indirect identity embodiments may assume an appropriate lookup for mapping between identities is performed to get one identity by looking up another identity. There may be multiple identities for a MS. Identities, by definition, provide a collective handle to data. For example, an email sender or recipient is an example of an identity ("logical handle") which can be associated to a user identity and/or MS identity and/or group identity. A sender, source, recipient, and system parameter in some atomic commands presented below is any of the variety of types of identities.

Address elements of "ip address" and "SNA address" are examples of logical addresses, but are mentioned specifically anyway. ID, IDType and Address construct atomic elements (as elaborated on Right Hand Side (RHS)) are self explanatory. The TimeSpec construct is one of various kinds of "date/time stamp" or "date/time period" atomic elements. In a syntactic embodiment, date/time stamps are specified with prefixed character(s) and a time format such as xYYYYMMDDHHMMSS.12.J (J=# places to right of decimal point, such that 1=is the one tenth (1/10) second place, two=the one hundredth (1/100) second place, etc). The first character(s) (i.e. x) clarify the date/time stamp information.

- >20080314 indicates "in effect if current date/time after Mar. 14, 2008;
- >=20080314 indicates "in effect if current date/time on or after Mar. 14, 2008;
- <200803142315 indicates "in effect if current date/time prior to Mar. 14, 2008 at 11:15 PM;
- <=200803142315 indicates "in effect if current date/time on or prior to Mar. 14, 2008 at 11:15 PM; and
- =20080314231503 indicates "in effect if current date/time matches Mar. 14, 2008 at 11:15:03 PM.

Date/time periods may have special leading characters, just as described above (which are also periods). When using the date/time format, the granulation of the date/time stamp is a period of time.

- 20080314 indicates "in effect if current date/time during Mar. 14, 2008;
- 200803142315 indicates "in effect if current date/time during Mar. 14, 2008 at 11:15 PM (any time during that minute); and
- 20080314231503 indicates "in effect if current date/time during Mar. 14, 2008 at 11:15:03 PM (any time during that second).

Date/time periods can also be specified with a range using a colon such as 20080314:20080315 (Mar. 14, 2008 through Mar. 15, 2008). A date/time period can be plural such as 20080314:20080315, 2008031712:2008031823 (i.e. multiple periods) by using a comma.

FIG. **30**C depicts a preferred embodiment BNF grammar **3034** for permissions **10** and groups (of data **8**). The terminology "permissions" and "privileges" are used interchangeably in this disclosure. However, the BNF grammar shows a permission can provide one privilege, or a plurality of privileges. There are a massive number (e.g. thousands) of values for "atomic privilege for assignment" (i.e. privileges that can be assigned from a grantor to a grantee) in grammar **3034**. Few examples are discussed below. This disclosure would be extremely lengthy to describe every privilege. The reader can determine a minimum set of LBX privileges (permissions) disclosed as: Any configurable privilege granted by one identity to another identity that can limit, enable, disable, delegate, or govern actions, feature(s), functionality, behavior(s), or any subset(s) thereof which are disclosed herein. Every feature disclosed herein, or feature subset thereof, can be managed (granted and enforced) with an associated privilege. Privileges may be used to "turn on" a feature or "turn off" a feature, depending on various embodiments.

There are two (2) main types of permissions (privileges): semantic privileges which on their own enable LBX features and functionality; and grammar specification privileges which enable BNF grammar specifications. Semantic privileges are named, anticipated by applications, and have a semantic meaning to an application. Semantic privileges are variables to applications whereby values at the time of an application checking the variable(s) determine how the application will behave. Semantic privileges can also have implicit associated action(s). Grammar specification privileges are named, anticipated by charter parser implementation, and indicate what is, and what is not, permitted when specifying a charter. Grammar specification privileges are variables to charter parsing whereby values at the time of charter parse logic checking the variable(s) determine whether or not the charter is valid (i.e. privileged) for execution. Impersonation is not directly defined in the BNF grammar of charters, and is therefore considered a semantic privilege.

The "MS relevance descriptor" atomic element is preferably a binary bit-mask accommodating all anticipated MS types (see "system type"). Each system type is represented by a bit-mask bit position wherein a bit set to 1 indicates the MS type does participate with the privilege assigned, and a bit set to 0 indicates the MS type does not participate with the privilege assigned. This is useful when MSs do not have equivalent capabilities thereby limiting interoperability for a particular feature governed by a privilege. When the optional MSRelevance construct is not specified with a privilege, the preferred default is assumed relevance for all MSs (i.e. =all bits set to 1). An alternate embodiment will make the default relevant for no MSs (i.e. =all bits set to 0). Privilege codes (i.e. syntactical constants equated to an "atomic privilege for assignment" description) are preferably long lived and never changing so that as new LBX privileges are introduced (i.e. new privileges supported), the old ones retain their values and assigned function, and operate properly with new software releases (i.e. backwards compatible). Thus, new constants (e.g. \lbxall=privilege for allowing all LBX interoperable features) for "atomic privilege for assignment" should be chosen carefully.

Grants are used to organize privileges in desired categories and/or sub-categories (e.g. organization name, team name, person name, etc and then privileges for that particular grant name). A grant can be used like a folder. Grants provide an hierarchy of tree branch nodes while privileges are leaf nodes of the grant privilege tree. There are many types of privileges. Many are categorized for configuring charter conditions and charter actions, and some can be subsets of others, for example to have an overall category of privileges as well as many subordinate privileges within that category. This facilitates enabling/disabling an entire set with a single configuration, or enabling/disabling certain privileges within the set. This also prevents forcing a user to define Grants to define privilege categories. BNF grammar **3034** does not clarify the Privilege construct with a parameter for further interpretation, however some embodiments will incorporate an optional Parameters specification:

Privilege="atomic privilege for assignment"[Parameters] [MSRelevance] [TimeSpec] [Description] [History] |VarInstantiations

In such embodiments, Parameters preferably resolves to the Parameters construct of FIG. **30**E for clarifying how to apply a particular privilege. Parameters, if used for privileges, have meaning within the context of a particular privilege. Some examples of semantic privileges (i.e. "atomic privilege for assignment") that can be granted from a grantor identity (ID/IDType) to a grantee identity (ID/IDType) include:

- Impersonate: allows the grantee to perform MS administration of grantor (alternate embodiments will further granulate to a plurality of impersonate privileges for each possible type, or target, of administration);
- LBX interoperable: allows overall LBX interoperability (all or none);
- View nearby status: enables determining if nearby each other;
- View whereabouts status: enables determining whereabouts (e.g. on a map);
- View Reports: enables viewing statistics and/or reports; This privilege is preferably set with a parameter for which statistics and/or which reports; An alternate embodiment will have individual privileges for each type of statistic and/or report;
- View Historical Report: enables viewing history information (e.g. routes); This privilege is preferably set with a parameter for which history information; An alternate embodiment will have individual privileges for each type of history information;
- Set Geofence arrival alert: allows an action for alerting based on arrival to a geofenced area; This privilege may be set with parameter(s) for which eligible area(s) to define geofences; An alternate embodiment will have individual privileges for each area(s);
- Set Geofence departure alert: allows an action for alerting based on departure from a geofenced area; This privilege may be set with parameter(s) for which eligible area(s) to define geofences; An alternate embodiment will have individual privileges for each area(s);
- Set nearby arrival alert: allows an action for alerting based on arrival to being nearby; This privilege may be set with a parameter for quantifying amount nearby;
- Set nearby departure alert: allows an action for alerting based on departure from being nearby; This privilege may be set with a parameter for quantifying amount nearby;
- Set Geofence group arrival alert: allows an action for alerting based on a group's arrival to a geofenced area; This privilege may be set with parameter(s) for which groups or MSs apply;
- Set Geofence group departure alert: allows an action for alerting based on a group's departure from a geofenced area; This privilege may be set with parameter(s) for which groups or MSs apply;
- Set nearby group arrival alert: allows an action for alerting based on a group's arrival to being nearby; This privilege may be set with parameter(s) for quantifying amount nearby, and/or which groups or MSs apply;
- Set nearby group departure alert: allows an action for alerting based on a group's departure from being nearby; This privilege may be set with parameter(s) for quantifying amount nearby, and/or which groups or MSs apply;
- Set Situational Location (as defined in U.S. Pat. Nos. 6,456,234; 6,731,238; 7,187,997; U.S. PTO Publication 2006/0022048 (Johnson)) arrival alert: allows an action for alerting based on arrival to a situational location; This privilege may be set with parameter(s) for one or more situational location(s) defined;
- Set Situational Location (as defined in U.S. Pat. Nos. 6,456,234; 6,731,238; 7,187,997; U.S. PTO Publication 2006/0022048 (Johnson)) departure alert: allows an action for alerting based on departure from a situational location; This privilege may be set with a parameter(s) for one or more situational location(s) defined;
- Set Situational Location (as defined in U.S. Pat. Nos. 6,456,234; 6,731,238; 7,187,997; U.S. PTO Publication 2006/0022048 (Johnson)) group arrival alert: allows an action for alerting based on a group's arrival to a situational location; This privilege may be set with parameter(s) for one or more situational location(s) defined, and/or which groups or MSs apply;
- Set Situational Location (as defined in U.S. Pat. Nos. 6,456,234; 6,731,238; 7,187,997; U.S. PTO Publication 2006/0022048 (Johnson)) group departure alert: allows an action for alerting based on a group's departure from a situational location; This privilege may be set with parameter(s) for one or more situational location(s) defined, and/or which groups or MSs apply;
- Allow action monitoring: allows condition for the monitoring of certain action(s); This privilege may be set with parameter(s) for which action(s) to be monitored;
- Accept service routing: enables being a service routing system; This privilege may be set with parameter(s) for which service(s) to route;
- Allow whereabouts monitoring (i.e. any WDR **1100** fields): allows condition for the monitoring of certain whereabouts; This privilege may be set with parameter(s) for which area(s) where whereabouts can be monitored; Another embodiment will define a specific privilege for each field and/or subfield of a WDR **1100** (e.g. speed monitoring (e.g. field **1100***h*));
- Service informant utilization (includes derived subsets for how to be used; e.g. log for me all successful detections (or particular types) by the remote MS of interest);
- Strip out WDR information inbound, outbound, and/or prior to be inserting to queue **22**: these types of privileges may also affect what charters can and cannot do;
- Support certain types of service informant code processing, for example for carpool collaboration;
- Participate in parking lot search functionality; this privilege may be set with parameter(s) for which parking lots apply;
- Be a candidate peer service target for any particular application, types of applications, or all applications, or for certain MSs, certain groups, or combinations of any of these (parameter(s) may be specified);
- Participate in LN-expanse as a master MS, for example to maintain a database of historical MSs in the vicinity, or a database of identity mappings (e.g. users to MSs; (parameter(s) may be specified);
- Keep track of hotspot history;
- Provide service propagation for any particular application, types of applications, or all applications, or for certain MSs, certain groups, or combinations of any of these (parameter(s) may be specified);
- Enable automatic call forwarding functionality when within proximity to a certain phone, for example to route a wireless call to a nearby wired line phone; this privilege may be set with parameter(s) for which phones or phone numbers participate;
- Enable configuration of deliverable content that can be delivered in a peer to peer manner to a MS in the vicinity,

using any data type, size, location, or other characteristic to be a unique privilege; parameter(s) may be specified to qualify this;

A privilege for any functionality or feature disclosed herein;

Any subordinate privilege of above, or of any functionality or feature disclosed herein;

Any parent privilege of above, or of any functionality or feature disclosed herein; and/or

Any privilege combination of above, or of any functionality or feature disclosed herein.

Grammar specification privileges can enable/disable permitted specifications of certain charter terms, conditions, actions, or any other charter aspect. Some examples of grammar specification privileges (i.e. "atomic privilege for assignment") that can be granted from a grantor identity (ID/IDType) to a grantee identity (ID/IDType) include:

Accept autodial #: allows an action for sending a speed dial number;

Accept web link: allows an action for sending a hyper link;

Accept email: allows an action for sending an email;

Accept SMS msg: allows an action for sending an SMS message;

Accept content: allows an action for sending a content of any type;

Accept broadcast email: allows an action for sending a broadcast email;

Accept broadcast SMS msg: allows an action for sending a broadcast SMS message;

Accept indicator: allows an action for sending an indicator;

Accept invocation: allows an action for invoking (optionally with parameters for which executable and parameters to it) an executable (application, script, command file, or any other executable); Alternate embodiments will have specific privileges for each type of executable that may be invoked);

Accept file: allows an action for sending a file or directory;

Accept semaphore control: allows an action for setting or clearing a semaphore; This privilege is preferably set with a parameter for which semaphore and what to do (set or clear);

Accept data control: allows an action for access, storing, alerting, or discarding data (alternate embodiments will further granulate to a plurality of data control privileges for each data control type (access, store, alter, discard, etc); This privilege may be set with parameter(s) for which data and what to do;

Accept database control: allows an action for access, storing, alerting, or discarding database data (alternate embodiments will further granulate to a plurality of data control privileges for each data control type (access, store, alter, discard, etc); This privilege may be set with parameter(s) for which database data and what to do;

Accept file control: allows an action for access, storing, alerting, or discarding file/directory path data (alternate embodiments will further granulate to a plurality of data control privileges for each data control type (access, store, alter, discard, etc); This privilege may be set with parameter(s) for which directory or file path(s) and what to do;

Allow profile match comparison: allows condition for the monitoring of certain profile(s); This privilege may be set with a parameter(s) for which profile(s) can be monitored/compared; An alternate embodiment will define a specific privilege for each ProfileMatch type;

Allow interest match comparison: allows condition for the monitoring of interests; This privilege may be set with parameter(s) for which interests can be monitored/compared; An alternate embodiment will define a specific privilege for each interest candidate;

Allow filters match comparison: allows condition for the monitoring of filters; This privilege may be set with parameter(s) for which filters can be monitored/compared; An alternate embodiment will define a specific privilege for each filter candidate;

Allow movement monitoring: allows condition for the monitoring of movement; This privilege may be set with parameter(s) for quantifying how much movement, and/or how long for lack of movement (an alternate embodiment will define distinct privileges for each movement monitoring type);

Allow application use monitoring: allows condition for the monitoring of application usage; This privilege may be set with parameter(s) for specifying which application(s) to monitor, and/or how long for usage of the application(s); Another embodiment specifies which aspect of the application is to be monitored (e.g. data, DB data, semaphore, thread/process invoke or terminate, file/directory data, etc);

Allow invocation monitoring: allows an action for monitoring application(s) used (optionally with parameter(s) for which application/executable); Alternate embodiments will have specific privileges for each application or executable of interest;

Allow application termination monitoring: allows condition for monitoring application(s) terminated (optionally with parameter(s) for which application/executable); Alternate embodiments will have specific privileges for each application or executable of interest;

Allow file system monitoring: allows condition for monitoring a file or directory; This privilege may be set with parameter(s) for specifying which path(s) to monitor, and/or what to monitor for, and how long for absence or removal of the path(s);

Allow semaphore monitoring: allows condition for monitoring a semaphore; This privilege may be set with parameter(s) for specifying which semaphore(s) to monitor, and/or what to monitor for (clear or set);

Allow data monitoring (file or directory): allows condition for monitoring data; This privilege may be set with parameter(s) for specifying which data to monitor, and/or what value to monitor for (charter condition like a debugger watch);

Allow data attribute monitoring (file or directory): allows condition for monitoring data attribute(s); This privilege may be set with parameter(s) for specifying which data attributes (e.g. chmod or attrib or extended attributes) to monitor, and/or what value to monitor for (charter condition like a debugger watch);

Allow database monitoring: allows condition for monitoring database data; This privilege may be set with parameter(s) for specifying which database data to monitor, and/or what value to monitor for (like a database trigger);

Allow sender monitor: allows condition for monitoring sender information; This privilege may be set with parameter(s) for specifying which sender address(es) to monitor email or SMS messages from (may have separate privileges for each type of distribution);

Allow recipient monitor: allows condition for monitoring recipient information; This privilege may be set with parameter(s) for specifying which recipient address(es) to monitor email or SMS messages to (may have separate privileges for each type of distribution);

Allow "modification" instead of "monitor"/"monitoring" for each monitor/monitoring privilege described above;

Allow focused title bar use: allows using the focused title bar for alerting;

A privilege for any BNF grammar atomic command, atomic operand, parameter(s), parameter type, atomic operator, or underlying action performed in a charter herein;

Any subordinate privilege of above, or of any functionality or feature disclosed herein;

Any parent privilege of above, or of any functionality or feature disclosed herein; and/or

Any privilege combination of above, or of any functionality or feature disclosed herein.

While the Grantor construct translates to the owner of the permission configuration according to grammar **3034**, impersonation permits a user to take on the identity of a Grantor for making a configuration. For example, a group by its very nature is a form of impersonation when a single user of the group grants permissions from the group to another identity. A user may also impersonate another user (if has the privilege to do so) for making configurations. In an alternative embodiment, grammar **3034** may include means for identifying the owner of the permission(s) granted. Group constructs provide means for collections of ID constructs, for example for teams, departments, family, whatever is selected for grouping by a name (atomic element "group name"). The impersonation privilege should be delegated very carefully in the preferred embodiment since the BNF grammar does not carry owner information except through a History construct use.

The Grantor of a privilege is the identity wanting to convey a privilege to another identity (the Grantee). The Grantee is the identity becoming privileged by administration of another identity (the Grantor). There are various embodiments for maintaining privileges, some embodiments having the side affect of increasing, or decreasing, the palette of available privileges for assignment. Privilege/Permission embodiments include:

1) Administrated privileges are maintained and enforced at the Grantor's MS. As privileged Grantee WDR information is detected at the Grantor's MS, or as Grantor WDR information is detected at the Grantor's MS: the appropriately privileged Grantee is provided with LBX application features at their (Grantee) MS in accordance with the privileges granted;

2) Administrated privileges are maintained and enforced at the Grantor's MS, but are also communicated to the Grantee's MS for being used by the Grantee for informative purposes. As privileged Grantee WDR information is detected at the Grantor's MS, or as Grantor WDR information is detected at the Grantor's MS: the appropriately privileged Grantee is provided with LBX application features at their (Grantee) MS in accordance with the privileges granted;

3) Administrated privileges are maintained at the Grantor's MS for administration purpose, but are used for governing features/processing at a Grantee MS. Privileges are appropriately communicated to a Grantee MS for WDR information processing, such that as Grantor WDR information is detected at the Grantee MS, the Grantee is provided with LBX application features at their (Grantee) MS in accordance with the privileges granted; and/or

4) Privileges are stored at both the Grantor's MS and the Grantee's MS for WDR information processing including any combination of #1 through #3 above (i.e. WDR information processing at each MS provides LBX features benefiting the Grantor and/or Grantee).

5) See FIG. **49**A discussions for some of the permission/privilege assignment considerations between a Grantor identity and a Grantee identity.

FIGS. **30**D through **30**E depict a preferred embodiment BNF grammar **3068***a* through **3068***b* for charters. Charters embody conditional events to be monitored and the actions to cause when those events occur. Notice there is still a Grantee and Grantor construct in charters, even in the face of having privileges for governing the charters. Grantor and Grantee constructs used in charters have to do with granting the permission/privilege to enable charters at a particular MS. Once they are enabled at a MS, permissions/privileges of grammar **3034** may be used to govern how the charters process.

It is important to note the context of terminology use "Grantor" and "Grantee" appears in, since they are similarly used in context of charters versus permissions. In both cases there is an acceptance/authentication/configuration granted by a Grantor to a Grantee. A permission Grantor grants a privilege to a Grantee. A charter Grantor grants a privilege to enable a Grantee's charters (may be at the mercy of privileges in the preferred embodiment). The Grantee construct in charters translates to the owner/creator/maintainer identity of the charter configuration according to grammar **3068***a* and **3068***b*, and the Grantor construct translates to an identity the Grantee has created the charter for, but does not necessarily have the privilege to do so, or does not necessarily have the privilege for any subset of processing of the charter. Privileges preferably govern whether charters are in effect, and how they are in effect. An alternative embodiment will activate (make in effect) a charter by granting it from one identity to another as shown in grammar **3068***a*. A charter consists of a conditional expression and can have an action or plurality of actions which are associated with the conditional expression. Upon evaluating the expression to an actionable condition (e.g. evaluates to a Boolean true result), the associated action(s) are invoked.

Impersonation permits a user to take on the identity of a Grantee for making a configuration. For example, a group by its very nature is a form of impersonation when a single user of the group administrates charters for the group. A user may also impersonate another user (if has the privilege to do so) for making configurations. In an alternative embodiment, grammar **3068***a* and **3068***b* may include means for identifying the owner of the charters administrated. The impersonation privilege should be delegated very carefully in the preferred embodiment since the BNF grammar does not carry owner information except through a History construct use.

The Grantee of a charter is the identity (e.g. creates and owns the charter) wanting to have its charters processed for another identity (the Grantor). The Grantor is the identity targeted for processing the administrated charter(s) created by the Grantee. The terminology "Grantor" and "Grantee" will become reversed (to match privilege assignments) in an embodiment which grants charters like privileges. There are various embodiments for maintaining charters, some embodiments having the side affect of increasing, or decreasing, the palette of available charter processing deployed. Charter embodiments include:

6) Administrated charters are stored at the Grantee's (the administrator's) MS. As privilege providing Grantor WDR information is detected at the Grantee's MS, the Grantee is provided with LBX application charter processing at his (Grantee) MS, preferably in accordance with privileges defined as described in #1 through #5 above;

7) Administrated charters are maintained at the Grantee's (the administrator's) MS, but are communicated to the Grantor's MS for being used for informative purposes. As privilege providing Grantor WDR information is detected at the Grantee's MS, the Grantee is provided with LBX application charter processing at his (Grantee) MS, preferably in accordance with privileges defined as described in #1 through #5 above;

8) Administrated charters are maintained at the Grantee's MS for administration purpose, but are used for processing at the Grantor MS. Charters are appropriately communicated to the Grantor MS for WDR information processing, such that as Grantor WDR information is detected at the Grantor MS, the Grantee is provided with LBX application features for processing at the Grantor's MS, preferably in accordance with privileges defined as described in #1 through #5 above. Also, as Grantee WDR information is detected at the Grantor's MS, the Grantee is provided with LBX application charter processing at his (Grantee) MS, preferably in accordance with privileges defined as described in #1 through #5 above; and/or

9) Charters are maintained at both the Grantor's MS and the Grantee's MS for WDR information processing, including any combination of #6 through #8 above (i.e. WDR information processing at each MS provides LBX features benefiting the Grantor and/or the Grantee).

10) See FIG. **49**B discussions for some of the charter assignment considerations between a Grantee identity and a Grantor identity.

Grammar **3068***a* "and" and "or" are atomic elements for CondOp operators. In a syntactic embodiment, "and" and "or" may be special characters (e.g. &, I, respectively). Grammar **3068***a* Value elaboration "atomic term" (RHS) is an atomic element for a special type of term that can be used in a condition specification, such as:

My MS location (e.g. \loc_my): preferred embodiment resolves to field **1100***c* from the most recent WDR which describes this MS (i.e. the MS of atomic term evaluation processing); WTV may be used to determine if this is of use (if not, may return a null, cause a failure in a conditional match, or generate an error);

A specified MS, or group, mobile location (e.g. \locByL_–30.21, –97.2=location at the specified latitude and longitude (ensure no intervening blanks): preferred embodiment resolves to a specified location comparable to a WDR field **1100***c*, not necessarily in the same format or units used as field **1100***c* (i.e. converted appropriately for a valid comparison when used). There are many different formats and units that can be specified here with a unique syntax;

A specified MS, or group, situational location (e.g. \slByL_–30.21, –97.2; 1050F=situational location at the specified latitude, longitude and elevation in feet (ensure no intervening blanks): preferred embodiment resolves to specified situational location comparable to applicable WDR fields, not necessarily in the same format or units used (i.e. converted appropriately for valid comparison(s) when used). See U.S. Pat. No. 6,456,234 (Johnson) for the definition of a situational location that can be specified. A reasonable syntax following the leading escape character and "sl" prefix should be used; this example assumes an anticipated order (lat, long, elevation); One embodiment also assumes an order for other situational location criteria wherein a semicolon (;) delimits data (i.e. use ";" to show lack of data at anticipated position (e.g. \slByL_–30.21, –97.2;;;;56); Another embodiment uses descriptors to indicate which data is being described so any order can be specified (e.g. \slByL_lat=–30.21, lon=–97.2;elev=1050F). There are many different formats, fields and units that can be specified here with a unique syntax;

My current MS mobile location (e.g. \loc_my): same as described above;

A current MS, or group, mobile location (e.g. \locByID_Larry=location of MS with id Larry, \locG_dept78=location of members of the group dept78): preferred embodiment resolves to a location associated with an identifier. Preferably, queue **22** is accessed first for the most recent occurrence of a WDR matching the identifier(s). An alternate embodiment additionally searches LBX history **30** if not found elsewhere. In one embodiment, an averaged location is made for a group identifier using locations of the identifiers belonging to the group, otherwise a group containing MSs with different locations causes a false condition when used in an expression, or alternatively cause an error. This is preferably used to compare locations of WDRs from a plurality of different MSs without requiring a value to be surfaced back to the expression reference;

A current MS, or group, situational location (e.g. \slByID_Larry=situational location of MS with id Larry, \slG_dept78=situational location of members of the group dept78): preferred embodiment resolves to a situational location associated with an identifier. Preferably, queue **22** is accessed first for the most recent occurrence of a WDR matching the identifier(s). An alternate embodiment additionally searches LBX history **30** if not found elsewhere. In one embodiment, an averaged situational location is made for a group identifier using locations of the identifiers belonging to the group, otherwise a group containing MSs with different locations causes a false condition when used in an expression, or alternatively cause an error. This is preferably used to compare situational locations of WDRs from a plurality of different MSs without requiring a value to be surfaced back to the expression reference;

Last application used (e.g. \appLast): preferably resolves to an application reference (e.g. name) which can be successfully compared to a MS operating system maintained reference for the application (e.g. as maintained to LBX history) that was last used by the MS user (e.g. embodiments for last focused, or last used that had user input directed to it). One embodiment implements only known PRR applications using field **5300***a* and/or **5300***b* for the reference (See FIGS. **53** and **55**A);

Last application context used (e.g. \appLastCtxt): preferably resolves to an application context reference which can be successfully compared to a MS operating system context maintained for comparison to LBX history. One embodiment implements only known PRR applications using field **5300***a* and/or **5300***b* for the application reference (See FIGS. **53** and **55**A), and saved user input for the context of when the application was focused. Another embodiment incorporates the system and methods of U.S. Pat. No. 5,692,143 ("Method and system for recalling desktop states in a data processing system", Johnson et al) to maintain application contexts to history;

Application in use (e.g. \appLive): preferably resolves to an application reference (e.g. name) which can be successfully compared to a MS operating system maintained reference for the application (e.g. as maintained to LBX history) that may or may not be running (active) on the MS. One embodiment implements only known PRR applications using field **5300***a* and/or **5300***b* for the reference (See FIGS. **53** and **55**A);

Application context in use (e.g. \appLiveCtxt): preferably resolves to an application context reference which can be successfully compared to a MS operating system context maintained for comparison. One embodiment implements only known PRR applications using field **5300***a* and/or **5300***b* for the application reference (See FIGS. **53** and **55**A), and saved user input for the current context of the application (e.g. maintained to LBX history). Another embodiment incorporates the system and methods of U.S. Pat. No. 5,692,143 ("Method and system for recalling desktop states in a data processing system", Johnson et al) to maintain application contexts;

Application active (e.g. \appLive): same as application in use above;

Application context active (e.g. \appLiveCtxt): same as application context in use above;

Current MS system date/time (e.g. \timestamp); preferably resolves to the MS date/time from the MS system clock interface for a current date/time stamp;

Particular LBX maintained statistical value (e.g. \st_statisticName wherein statisticName is the name of the statistic): preferably resolves to the referenced statistic name of statistics **14**. There are potentially hundreds of statistics maintained for the MS;

MS ID of MS hosting atomic term (e.g. \thisms; alternate embodiments support ID and IDType grammar rules): preferably resolves to the identifier of the MS where the atomic term is being resolved; and/or

Most current WDR field of \thisMS (e.g. \fldname); fldname is identical to WDR in-process field names which can reference any field, subfield, set, subset, or derived data/information of a WDR in process (i.e. _fldname, _I_fldname, _O_fldname). The difference here is that the most recent WDR (e.g. of queue **22**) for \thisMS is accessed, rather than an in-process WDR. The leading backslash indicates to reference the most recent WDR for \thisMS. In some embodiments, the WTV is accessed and an error is produced for \fldname references that reference stale WDR information.

Preferably, a convenient syntax using a leading escape character refers to an atomic term (e.g. \loc_my=My MS location). When used in conjunction with other conditions, an "atomic term" provides extraordinary location based expressions. Other Grammar **3068***a* atomic elements are described here: "Any WDR **1100** field, or any subset thereof" is self explanatory; "Any Application data field, or any subset thereof" is an atomic element for any semaphore, data, database data, file/directory data, or any other reference-able data of a specified application; "number" is any number; "text string" is any text string; "True" is a Boolean representing true; "False" is a Boolean representing false; "typed memory pointer" is a pointer to memory location (of any memory or storage described for FIG. **1**D) containing a known type of data and length; "typed memory value" is a memory location (of any memory or storage described for FIG. **1**D) containing a known type of data and length; "typed file path" is a file path location (of any memory or storage described for FIG. **1**D) containing a known type of data and length; "typed file path and offset" is a file path location (of any memory or storage described for FIG. **1**D) and an offset therein (e.g. byte offset) for pointing to a known type of data and length; "typed DB qualifier" is a database data path (of any memory or storage described for FIG. **1**D) for qualifying data in a database (e.g. with a query, with a identity/table/row/column qualifier, or other reasonable database qualifying method).

WDRTerm provides means for setting up conditions on any WDR **1100** field or subfield that is detected for WDR(s):

Inserted by FIG. **2**F processing (e.g. received from other MSs, or created by the hosting MS); and/or

Sent/communicated outbound from a MS; and/or

Received/communicated inbound to a MS.

An alternate BNF grammar embodiment qualifies the "Any WDR **1100** field, or any subset thereof" atomic element with an operator for which of the three MS code paths to check WDR field conditions (e.g. Operators of "OUTBOUND" and "INBOUND", denoted by perhaps a syntactical O and I, respectively). Absence of an operator can be assumed for checking WDRs on FIG. **2**F insert processing. Such embodiments result in a BNF grammar WDRTerm definition of:

WDRTerm=[WDRTermOp] "Any WDR **1100** field, or any subset thereof" [Description] [History]|VarInstantiate

WDRTermOp="inbound"|"outbound"

Yet another embodiment will allow combination operators for qualifying a combination of any three MS code paths to check.

AppTerm provides means for setting up conditions on data of any application of an MS, for example to trigger an action based on a particular active call during whereabouts processing. A few AppTerm examples are any of the following:

Any phone application data record data (e.g. incoming call(s), outgoing call(s), active call(s), caller id, call attributes, etc)

Any email/SMS message application data record data (e.g. mailbox attributes, message last sent, message last received, message being composed, last type of message sent, last type of message received, attribute(s) of any message(s), etc)

Any address book application data record data (e.g. group(s) defined, friend(s) defined, entry(s) defined and any data associated with those, etc)

Any calendar application data record data (e.g. last scheduled entry, most recently removed entry, number of entries per time period(s), last scheduled event attendee(s), number of scheduled events for specified qualifier, next forthcoming appointment, etc)

Any map application data record data; and/or

Any other application data record data of a MS.

Grammar **3068***b* completes definition of grammar rules for charters. The Invocation construct elaborates to any of a variety of executables, with or without parameters, including Dynamic Link Library (DLL) interfaces (e.g. function), post-compile linked interfaces (e.g. function), scripts, batch files, command files, or any other executable. The invoked interface should return a value, preferably a Boolean (true or false), otherwise one will preferably be determined or defaulted for it. The Op construct contains atomic elements (called atomic operators) for certain operators used for terms to specify conditions. In syntactical embodiments, each atomic operator may be clarified with a not modifier (i.e. !). For example, "equal to" is "=" and "not equal to" is "!=". Those skilled in the art recognize which atomic operator is contextually appropriate for which applicable terms (see BNF grammar **3068***a*). There are many reasonable syntactical embodiments for atomic operators, with at least:

=: equal to;

!=: not equal to;

>: greater than;

!>: not greater than;

>=: greater than or equal to;

!>=: not greater than or equal to;

- <: less than;
- !<: not less than;
- <=: less than or equal to;
- !<=: not less than or equal to;
- ^: in;
- !^: not in;
- ^^: was in;
- !^^: was not in;
- @: at;
- !@: not at;
- @@: was at;
- !@@: was not at;
- $(range): in vicinity of (range=distance (e.g. 10 F=10 Feet));
- !$(range): not in vicinity of (range=distance (e.g. 1 L=1 Mile));
- >$(range): newly in vicinity of;
- !>$(range): not newly in vicinity of;
- $>(range): departed from vicinity of;
- !$>(range): not departed from vicinity of;
- (spec)$(range): recently in vicinity of (spec=time period (e.g. 8 H=in last 8 hours));
- (spec)!$(range): not recently in vicinity of (spec=time period (e.g. 8 H=in last 8 hours));
- (spec)$$(range): recently departed from vicinity of (spec=time period (e.g. 5 M=in last 5 minutes)); and
- (spec)!$$(range): not recently departed from vicinity of (spec=time period (e.g. 5 M=in last 5 minutes)).

Values for "range" above can be any reasonable units such as 3K implies 3 Kilometers, 3 M implies 3 Meters, 1 L implies 3 Miles, 3 F implies 3 Feet, etc. Values for "spec" above can be any reasonable time specification as described for TimeSpec (FIG. **30**B) and/or using qualifiers like "range" such as 3 W implies 3 Weeks, 3 D implies 3 Days, 3H implies 3 Hours, 3 M implies 3 Minutes, etc.

Resolving of conditions using atomic operators involves evaluating conditions (BNF grammar constructs) and additionally accessing similar data of LBX history **30** in some preferred embodiments. Atomic operator validation errors should result when inappropriately used.

Example syntactical embodiments of the "atomic profile match operator" atomic element include:

- #: number of profile matches;
- %: percentage of profile matches;
- #(tag(s)): number of profile tag section matches (e.g. #(interests) compares one profile tag "interests"); and
- % (tag(s)): percentage of profile tag section matches (e.g. #(interest, activities) compares a plurality of profile tags ("interests" and "activities").

In one embodiment of profiles maintained at MSs, a LBX singles/dating application maintains a MS profile for user's interests, tastes, likes, dislikes, etc. The ProfileMatch operators enable comparing user profiles under a variety of conditions, for example to cause an action of alerting a user that a person of interest is nearby. See FIGS. **77** and **78** for other profile information.

Atomic operators are context sensitive and take on their meaning in context to terms (i.e. BNF Grammar Term) they are used with. An alternate embodiment incorporates new appropriate atomic operators for use as CondOp operators, provided the result of the condition is a Boolean (e.g. term>=term results in a true or false). Also, while a syntactical form of parenthesis is not explicitly shown in the BNF grammar, the Conditions constructs explicitly defines how to make complex expressions with multiple conditions. Using parenthesis is one preferred syntactical embodiment for carrying out the Conditions construct. The intention of the BNF grammar is to end up with any reasonable conditional expression for evaluating to a Boolean True or False. Complex expression embodiments involving any conceivable operators, terms, order of evaluation (e.g. as syntactically represented with parentheses), and other arithmetic similarities, are certainly within the spirit and scope of this disclosure.

BNF grammar terms are to cover expressions containing conditions involving WDR fields (WDRTerm), situational locations, geofences (i.e. a geographic boundary identifying an area or space), two dimensional and three dimensional areas, two dimensional and three dimensional space, point in an area, point in space, movement amounts, movement distances, movement activity, MS IDs, MS group IDs, current mobile locations, past mobile locations, future mobile locations, nearness, distantness, newly near, newly afar, activities at locations (past, present, future), applications and context thereof in use at locations (past, present, future), etc. There are many various embodiments for specific supported operators used to provide interpretation to the terms. Certain operators, terms, and processing is presented for explanation and is in no way meant to limit the many other expression (BNF Grammar Expression) embodiments carrying the spirit of the disclosure.

The Command construct elaborates to atomic commands. The "atomic command" atomic element is a list of supported commands such as those found in the column headings of FIGS. **31**A through **31**E table (see discussions for FIGS. **31**A through **31**E). There are many commands, some popular commands being shown. The Operand construct elaborates to atomic operands. The "atomic operand" atomic element is a list of supported operands (data processing system objects) such as those found in the row headings of FIGS. **31**A through **31**E table (see discussions for FIGS. **31**A through **31**E). There are many operands, some popular operands being shown. For each command and operand combination, there may be anticipated parameters. The command and operand pair indicates how to interpret and process the parameters.

The constructs of Parameter, WDRTerm, AppTerm, Value and Data are appropriately interpreted within context of their usage. An optional time specification is made available when specifying charters (i.e. when charter is in effect), expressions (i.e. a plurality of conditions (e.g. with Conditions within Expressions construct)), a particular condition (e.g. with Condition elaborations within Condition construct), and actions (e.g. with Action elaborations within Action construct). One embodiment supports multiple Host specifications for a particular action. Some embodiments allow an Invocation to include invocations as parameters in a recursive manner so as to "bubble up" a resulting Boolean (e.g. fcn1(2, fcn2(p1, x, 45), 10) such that fcn2 may also have invocations for parameters. The conventional inside out evaluation order is implemented. Other embodiments support various types of invocations which contribute to the overall invocation result returned.

In alternate embodiments, an action can return a return code, for example to convey success, failure, or some other value(s) back to the point of performing the action. Such embodiments may support nesting of returned values in BNF grammar Parameters so as to affect the overall processing of actions. For example: action1(parameter(s), . . . , action2(. . . parameters . . .), . . . parameter(s)), and action2 may include returning value(s) from its parameters (which are actions).

Wildcarding is of value for broader specifications in a single specification. Wildcards may be used for BNF grammar specification wherever possible to broaden the scope of a particular specification (e.g. Condition, TimeSpec, etc).

FIGS. **31**A through **31**E depict a preferred embodiment set of command and operand candidates for Action Data Records (ADRs) (e.g. FIG. **37**B) facilitating the discussing of associated parameters (e.g. FIG. **37**C) of the ADRs of the present disclosure. Preferably, there are grammar specification privileges for governing every aspect of charters. Commands (atomic commands), operands (atomic operands), operators (atomic operators and CondOp), parameters (Parameter), associated conditions (Condition and CondOp), terms (Term), actions thereof (Action), associated data types thereof (Data), affected identities thereof (ID/IDType), and any other charter specification aspect, can be controlled by grammar specification privileges.

An "atomic command" is an enumeration shown in column headings (i.e. **101**, **103**, . . . etc) with an implied command meaning. FIG. **32**A shows what meaning is provided to some of the "atomic command" enumerations shown (also see FIG. **34**D). A plurality of commands can map to a single command meaning. This supports different words/phrases (e.g. spoken in a voice command interface) to produce the same resulting command so that different people specify commands with terminology, language, or (written) form they prefer. An "atomic operand" is an enumeration shown in row headings (i.e. **201**, **203**, . . . etc) with an implied operand meaning. FIG. **32**B shows what meaning is provided to some of the "atomic operand" enumerations shown (also see FIG. **34**D). A plurality of operands can map to a single operand meaning. This supports different words/phrases (e.g. spoken in a voice command interface) to produce the same resulting operand so that different people specify operands with terminology, language, or (written) form they prefer. Operands are also referred to as data processing system objects because they are common objects associated with data processing systems. FIGS. **31**A through **31**E demonstrate anticipated parameters for each combination of a command with an operand. There are potentially hundreds (or more) of commands and operands. This disclosure would be extremely large to cover all the different commands, operands, and parameters that may be reasonable. Only some examples with a small number of parameters are demonstrated in FIGS. **31**A through **31**E to facilitate discussions. There can be a large number of parameters for a command and operand pair. Each parameter, as shown by the BNF grammar, may be in many forms. In one preferred embodiment (not shown in BNF grammar), the Parameter construct of FIG. **30**E may also elaborate to a ParameterExpression which is any valid arithmetic expression that elaborates to one of the Parameter constructs (RHS) shown in the BNF Grammar. This allows specifying expressions which can be evaluated at run time for dynamically evaluating to a parameter for processing.

The combination of a command with an operand, and its set of associated parameters, form an action in the present disclosure, relative the BNF grammar discussed above. Some of the command/operand combinations overlap, or intersect, in functionality and/or parameters. In general, if parameters are not found (null specified) for an anticipated parameter position, a default is assumed (e.g. parameters of 5, 7 indicates three (3) parameters of 5, use default or ignore, and 7). Operands and parameters are preferably determined at executable code run time when referenced/accessed so that the underlying values may dynamically change as needed at executable code run time in the same references. For example, a variable set with constructs which elaborates to a command, operand, and parameters, can be instantiated in different contexts for completely different results. Also, a programming language enhanced with new syntax (e.g. as described in FIG. **51**) may include a loop for processing a single construct which causes completely different results at each loop iteration. The operand or parameter specification itself may be for a static value or dynamic value as determined by the reference used. An alternate embodiment elaborates values like a preprocessed macro ahead of time prior to processing for static command, operand, and parameter values. Combinations described by FIGS. **31**A through **31**E are discussed with flowcharts. In another embodiment, substitution (like parameter substitution discussed above for FIG. **30**A) can be used for replacing parameters at the time of invocation. In any case, Parameters can contain values which are static or dynamically changing up to the time of reference.

Parameters of atomic command processing will evaluate/resolve/elaborate to an appropriate data type and form for processing which is described by the #B matrices below (e.g. FIG. **63**B is the matrix for describing atomic send command processing). The #B descriptions provide the guide for the data types and forms supportable for the parameters. For example, an email body parameter may be a string, a file containing text, a variable which resolves to a string or file, etc. The BNF grammar is intended to be fully exploited in the many possible embodiments used for each parameter.

FIG. **32**A depicts a preferred embodiment of a National Language Support (NLS) directive command cross reference. Each "atomic command" has at least one associated directive, and in many cases a plurality of directives. Depending on an MS embodiment, a user may interact with the MS with typed text, voice control, selected graphical user interface text, symbols, or objects, or some other form of communication between the user and the MS. A directive (FIG. **32**A command and FIG. **32**B operand) embodies the MS recognized communication by the user. Directives can be a word, a phrase, a symbol, a set of symbols, something spoken, something displayed, or any other form of communications between a user and the MS. It is advantageous for a plurality of command directives mapped to an "atomic command" so a MS user is not limited with having to know the one command to operate the MS. The MS should cater to everyone with all anticipated user input from a diverse set of users which may be used to specify a command. This maximizes MS usability. The command directive is input to the MS for translating to the "atomic command". One preferred embodiment of a directive command cross reference **3202** maps a textual directive (Directive column) to a command ("atomic command" of Command column). In this embodiment, a user types a directive or speaks a directive to a voice control interface (ultimately converted to text). Cross reference **3204-1** demonstrates an English language cross reference. Preferably, there is a cross reference for every language supported by the MS, for example, a Spanish cross reference **3204-2**, a Russian cross reference, a Chinese cross reference, and a cross reference for the L languages supported by the MS (i.e. **3204**-L being the final cross referenced language). Single byte character (e.g. Latin-1) and double byte character (e.g. Asian Pacific) encodings are supported. Commands disclosed are intended to be user friendly through support of native language, slang, or preferred command annunciation (e.g. in a voice control interface). FIG. **34**D enumerates some commands which may appear in a command cross reference **3202**.

FIG. **32**B depicts a preferred embodiment of a NLS directive operand cross reference. Each "atomic operand" has at least one associated directive, and in many cases a plurality of directives. It is advantageous for a plurality of operand directives mapped to an "atomic operand" so a MS user is not limited with having to know the one operand to operate the MS. The MS should cater to everyone with all anticipated user input from a diverse set of users which may be used to specify an operand. The directive is input to the MS for translating to the "atomic operand". One preferred embodiment of a directive operand cross reference **3252** maps a textual directive (Directive column) to an operand ("atomic operand" of Operand column). In this embodiment, a user types a directive or speaks a directive to a voice control interface (ultimately converted to text). Cross reference **3254-1** demonstrates an English language cross reference. Preferably, there is a cross reference for every language supported by the MS, for example, a Spanish cross reference **3254-2**, a Russian cross reference, a Chinese cross reference, and a cross reference for the L languages supported by the MS (i.e. **3254**-L being the final cross referenced language). Operands disclosed are intended to be user friendly through support of native language, slang, or preferred command annunciation (e.g. in a voice control interface). FIG. **34**D enumerates some operands which may appear in an operand cross reference **3252**.

In the preferred embodiment, Parameters are contextually determined upon the MS recognizing user directives, depending on the context in use at the time. In another embodiment, Parameters will also have directive mappings for being interpreted for MS processing, analogously to FIGS. **32**A and **32**B.

FIG. **33**A depicts a preferred embodiment American National Standards Institute (ANSI) X.409 encoding of the BNF grammar of FIGS. **30**A through **30**B for variables, variable instantiations and common grammar for BNF grammars of permissions and charters. A one superscript (1) is shown in constructs which may not be necessary in implementations since the next subordinate token can be parsed and deciphered on its own merit relative the overall length of the datastream containing the subordinate tokens. For example, a plural Variables construct and token is not necessary since an overall datastream length can be provided which contains sibling Variable constructs that can be parsed. Preferably, Variable assignments include the X.409 datastreams for the constructs or atomic elements as described in FIGS. **33**A through **33**C. FIG. **33**B depicts a preferred embodiment ANSI X.409 encoding of the BNF grammar of FIG. **30**C for permissions **10** and groups, and FIG. **33**C depicts a preferred embodiment ANSI X.409 encoding of the BNF grammar of FIGS. **30**D through **30**E for charters **12**. All of the X.409 encodings are preferably used to communicate information of permissions **10** and/or charters **12** (e.g. the BNF grammar constructs) between systems.

The preferred embodiment of a WDRTerm is a system well known WDR field/subfield variable name with two (2) leading underscore characters (e.g. source code references of: _confidence refers to a confidence value of a WDR confidence field **1100***d*; _msyaw refers to a yaw value of a WDR location reference field **1100***f* MS yaw subfield). Some useful examples using a WDRTerm include:

- A specified MS, or group, WDR **1100** field (e.g. condition using field **1100***a* of (_I_msid !=George) & (_I_msid ^ChurchGroup));
- specified MS, or group, WDR **1100** field or subfield value;
- A current MS, or group, WDR **1100** field (e.g. condition using field **1100***a* of (_msid !=George) & (_msid ^ChurchGroup)); and
- A current MS, or group, WDR **1100** field or subfield value;

The preferred embodiment of an AppTerm is a system well known application variable name with a registered prefix, followed by an underscore character, followed by the variable name in context for the particular application (e.g. source code references of: M_source refers to a source email address of a received email for the registered MS email application which was registered with a "M" prefix; B_srchcriteria refers to the most recently specified search criteria used in the MS internet browser application which was registered with a "B" prefix). The preferred WDRTerm and AppTerm syntaxes provide user specifiable programmatic variable references for expressions/conditions to cause certain actions. The double underscore variable references refer to a WDR in process (e.g. inserted to queue **22** (_fldname), inbound to MS (_I_fldname), outbound from MS (_O_fldname)) at the particular MS. There is a system well known double underscore variable name for every field and subfield of a WDR as disclosed herein. The registered prefix name variable references always refer to data applicable to an object in process (e.g. specific data for: email just sent, email just received, phone call underway, phone call last made, phone call just received, calendar entry last posted, etc) within an application of the particular MS. There is a system well known underscore variable name for each exposed application data, and registering the prefix correlates the variable name to a particular MS application (see FIG. **53**).

An "atomic term" is another special type of user specifiable programmatic variable reference for expressions/conditions to cause certain actions. The preferred embodiment of an atomic term is a system well known variable name with a leading backslash (\) escape character (e.g. source code references of: \loc_my refers to the most recent MS location; \timestamp refers to the current MS system date/time in a date/time stamp format). There can be atomic terms to facilitate expression/condition specifications, some of which were described above.

FIGS. **33**A through **33**C demonstrate using the BNF grammar of FIGS. **30**A through **30**E to define an unambiguous datastream encoding which can be communicated between systems (e.g. MSs, or service and MS). Similarly, those skilled in the art recognize how to define a set of XML tags and relationships from the BNF grammar of FIGS. **30**A through **30**E for communicating an unambiguous XML datastream encoding which can be communicated between systems. For example, X.409 encoded tokens are translatable to XML tags that have scope between delimiters, and have attributes for those tags. The XML author may improve efficiency by making some constructs, which are subordinate to other constructs, into attributes (e.g. ID and IDType constructs as attributes to a Grantor and/or Grantee XML tag). The XML author may also decide to have some XML tags self contained (e.g. <History creatordt=" . . . " creatorid=" . . . " . . . /> or provide nesting, for example to accommodate an unpredictable plurality of subordinate items (e.g. <Permission . . . > . . . <Grantor userid="joe"/> . . . <Grantee groupid="dept1"/> . . . <Grantee groupid="dept43"/> . . . <Grantee groupid="dept9870"/> . . . </Permission>). It is a straightforward matter for translating the BNF grammar of FIGS. **30**A through **30**E into an efficiently processed XML encoding for communications between MSs. An appropriate XML header will identify the datastream (and version) to the receiving system (like HTML, WML, etc) and the receiving system (e.g. MS) will process accordingly using the present disclosure guide for proper parsing to internalize to a suitable processable format (e.g. FIGS. **34**A through **34**G, FIGS. **35**A through **37**C, FIG. **52**, or another suitable format per disclosure). See FIG. **54** for one example of an XML encoding.

FIGS. **34**A through **34**G depict preferred embodiment C programming source code header file contents, derived from the grammar of FIGS. **30**A through **30**E. A C example was selected so that the embodiment was purely data in nature. Another preferred embodiment utilizes an object oriented programming source code (e.g. C++, C#, or Java), but those examples mix data and object code in defining relationships. A preferred object oriented architecture would create objects for BNF grammar constructs that contain applicable processing data and code. The object hierarchy would then equate to construct relationships. Nevertheless, a purely data form of source code is demonstrated by FIGS. **34**A through **34**G (and FIG. **52**) to facilitate understanding. Those skilled in the relevant arts know how to embody the BNF grammar of FIG. **30**A through **30**E in a particular programming source code. The C programming source code may be used for:

- Parsing, processing, and/or internalizing a derivative X.409 encoding of the BNF grammar of FIGS. **30**A through **30**E (e.g. FIGS. **33**A through **33**C);
- Parsing, processing, and/or internalizing a derivative XML encoding of the BNF grammar of FIGS. **30**A through **30**E;
- Compiler parsing, processing, and/or internalizing of a programming language processing form of the BNF grammar of FIGS. **30**A through **30**E;
- Interpreter parsing, processing, and/or internalizing of a programming language processing form of the BNF grammar of FIGS. **30**A through **30**E;
- Internalized representation of permissions **10**, groups (data **8**) and/or charters **12** to data processing system memory;
- Internalized representation of permissions **10**, groups (data **8**) and/or charters **12** to data processing system storage; and/or
- Parsing, processing, and/or internalizing any particular derivative form, or subset, of the BNF grammar of FIGS. **30**A through **30**E.

Source code header information is well understood by those skilled in the relevant art in light of the BNF grammar disclosed. The example does make certain assumptions which are easily altered depending on specificities of a derivative form, or subset, of the grammar of FIGS. **30**A through **30**E. Assumptions are easily modified for "good" implementations through modification of isolated constants in the header file:

- TLV tokens are assumed to occupy 2 bytes in length;
- TLV length bytes are assumed to occupy 4 bytes in length;
- Some of the header definitions may be used solely for processing X.409 encodings in which case they can be removed depending on the context of source code use;
- Data structure linkage;
- Data structure form without affecting objective semantics;
- Data structure field definitions;
- Unsigned character type is used for data that can be a typecast byte stream, and pointers to unsigned character is used for pointers to data that can be typecast;
- Source code syntax; or
- Other aspects of the source code which are adaptable to a particular derivative form, or subset, of the BNF grammar of FIGS. **30**A through **30**E.

The TIMESPEC structure of FIG. **34**E preferably utilizes a well performing Julian date/time format. Julian date/time formats allows using unambiguous floating point numbers for date/time stamps. This provides maximum performance for storage, database queries, and data manipulation. Open ended periods of time use an unspecified start, or end data/time stamp, as appropriate (i.e. DT_NOENDSPEC or DT_NOSTARTSPEC). A known implemented minimal time granulation used in Julian date/time stamps can be decrement or incremented by one (1) as appropriate to provide a non-inclusive date/time stamp period delimiter in a range specification (e.g. >date/time stamp).

The VAR structure provides a pointer to a datastream which can be typecast (if applicable in embodiments which elaborate the variable prior to being instantiated, or referenced), or later processed. Variables are preferably not elaborated/evaluated until instantiated or referenced. For example, the variable assigned value(s) which are parsed from an encoding remains unprocessed (e.g. stays in X.409 datastream encoded form) until instantiated. Enough space is dynamically allocated for the value(s) (e.g. per length of variable's value(s)) (e.g. X.409 encoding form), the variable's value (e.g. X.409 encoding) is copied to the allocated space, and the v.value pointer is set to the start of the allocated space. The v.value pointer will be used later when the variable is instantiated (to then parse and process the variable value(s) when at the context they are instantiated).

An alternate embodiment to the PERMISSION structure of FIG. **34**F may not require the grantor fields (e.g. grantor, gortype) since the data processing system owning the data may only maintain permissions for the grantor (e.g. the MS user). An alternate embodiment to the CHARTER structure of FIG. **34**G may not require the grantee fields (e.g. grantee, geetype) or the grantor fields (e.g. grantor, gortype) since the data processing system owning the data may only maintain charters for that user at his MS. Another embodiment to the CHARTER structure of FIG. **34**G may not require the grantor fields (e.g. grantor, gortype) since the data processing system owning the data may be self explanatory for the Grantor identity (e.g. charters used at MS of Grantor).

FIGS. **35**A through **37**C, and FIG. **53**, illustrate data records, for example maintained in an SQL database, or maintained in record form by a data processing system. Depending on the embodiment, some data record fields disclosed may be multi-part fields (i.e. have sub-fields), fixed length records, varying length records, or a combination with field(s) in one form or another. Some data record field embodiments will use anticipated fixed length record positions for subfields that can contain useful data, or a null value (e.g. −1). Other embodiments may use varying length fields depending on the number of sub-fields to be populated, or may use varying length fields and/or sub-fields which have tags indicating their presence. Other embodiments will define additional data record fields to prevent putting more than one accessible data item in one field. In any case, processing will have means for knowing whether a value is present or not, and for which field (or sub-field) it is present. Absence in data may be indicated with a null indicator (−1), or indicated with its lack of being there (e.g. varying length record embodiments). Fields described may be converted: a) prior to storing; or b) after accessing; or c) by storage interface processing; for standardized processing. Fields described may not be converted (i.e. used as is).

FIG. **35**A depicts a preferred embodiment of a Granting Data Record (GDR) **3500** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. A GDR **3500** is the main data record for defining a granting of permissions **10**, or charters **12**. A granting identifier (granting ID) field **3500***a* contains a unique number generated for the record **3500** to distinguish it from all other records **3500** maintained. For example, in a Microsoft SQL Server deployment, granting ID field **3500***a* is a primary key column. Another embodiment uses the correlation generation techniques described above to ensure a unique number is generated. Field **3500***a* facilitates well performing searches, updates, deletes, and other I/O (input/output) interfaces. Field **3500***a* may match (for joining) a field **3520***b* or **3700***a*, depending on the GDR type (GDR type field **3500***t* with value of Permission or Charter). A granting type field **3500***t* distinguishes the type of GDR (Permission or Charter) for: a

Grantor granting all privileges to a Grantee (i.e. Permission (e.g. ID field **3500***a* unique across GDRs but not used to join other data records)), a Grantor granting specific privilege(s) and/or grants of privileges (permission(s)) to a Grantee ((i.e. Permission (e.g. ascendant ID field **3520***b* value in ID field **3500***a*)), and a Grantor granting enablement of a charter to a Grantee ((i.e. Charter (e.g. charter ID field **3700***a* value in ID field **3500***a*)). An owner information (info) field **3500***b* provides who the owner (creator and/or maintainer) is of the GDR **3500**. Depending on embodiments, or how the GDR **3500** was created, owner info field **3500***b* may contain data like the ID and type pair as defined for fields **3500***c* and **3500***d*, or fields **3500***e* and **3500***f*. An alternate embodiment to owner info field **3500***b* is two (2) fields: owner info ID field **3500***b*-**1** and owner info type field **3500***b*-**2**. Yet another embodiment removes field **3500***b* because MS user (e.g. the grantor) information is understood to be the owner of the GDR **3500**. The owner field **3500***b* may become important in user impersonation. A grantor ID field **3500***c* provides an identifier of the granting grantor and a grantor type field **3500***d* provides the type of the grantor ID field **3500***c*. A grantee ID field **3500***e* provides an identifier of the granting grantee and a grantee type field **3500***f* provides the type of the grantee ID field **3500***e*.

FIG. **35**B depicts a preferred embodiment of a Grant Data Record (GRTDR) **3510** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. A GRTDR **3510** is the main data record for defining a grant. A grant identifier (grant ID) field **3510***a* contains a unique number generated for the record **3510** to distinguish it from all other records **3510** maintained. Field **3510***a* is to be maintained similarly to as described for field **3500***a* (e.g. primary key column, correlation generation, facilitates well performing I/O). An owner information (info) field **3510***b* provides who the owner (creator and/or maintainer) is of the GRTDR **3510**. Field **3510***b* is to be maintained similarly to as described for field **3500***b* (e.g. embodiments for like ID and type pair, two (2) fields, removal because MS user information understood to be owner). A grant name field **3510***c* provides the name of the grant.

FIG. **35**C depicts a preferred embodiment of a Generic Assignment Data Record (GADR) **3520** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. A GADR **3520** is the main data record for defining an assignment relationship between data records. The assignment relationship can be viewed as a container relationship, or a parent-child relationship such as in a tree structure. An ascendant type field **3520***a* contains the type of parent (or container) data record in the relationship. Values maintained to field **3520***a* include Permission, Grant, or Group. An ascendant ID field **3520***b* provides an identifier of the parent (or container) data record in the relationship (used for joining data records in queries in an SQL embodiment). Values maintained to field **3520***b* include values of granting ID field **3500***a*, grant ID field **3510***a*, or group ID field **3540***a*. A descendant type field **3520***c* contains the type of child (or contained) data record in the relationship. Values maintained to field **3520***c* include Grant, Privilege, Group, or ID Type (e.g. Grantor or Grantee ID type). A descendant ID field **3520***d* provides an identifier of the child (or contained) data record in the relationship (used in joining data records in queries in an SQL embodiment). Values maintained to field **3520***d* include values of grant ID field **3510***a*, privilege identifier (i.e. "atomic privilege for assignment"), group ID field **3540***a*, ID field **3500***c*, or ID field **3500***e*. Records **3520** (key for list below is descendant first; ascendant last (i.e. " . . . in a . . . ")) are used to represent:

Grant(s) (the descendants) in a permission (the ascendant);
Privilege(s) in a permission;
Grant(s) in a grant (e.g. tree structure of grant names);
Privilege(s) in a grant;
Groups(s) in a group (e.g. tree structure of group names);
IDs in a group (e.g. group of grantors and/or grantees); and/or
Other parent/child relationships of data records disclosed.

An alternate embodiment will define distinct record definitions (e.g. **3520**-*z*) for any subset of relationships described to prevent data access performance of one relationship from impacting performance accesses of another relationship maintained. For example, in an SQL embodiment, there may be two (2) tables: one for handling three (3) of the relationships described, and another for handling all other relationships described. In another SQL example, six (6) distinct tables could be defined when there are only six (6) relationships to maintain. Each of the distinct tables could have only two (2) fields defined for the relationship (i.e. ascendant ID and descendant ID). The type fields may not be required since it would be known that each table handles a single type of relationship (i.e. GADR-grant-to-permission, GADR-privilege-to-permission, GADR-grant-to-grant, GADR-privilege-to-grant, GADR-group-to-group and GADR-ID-to-group). Performance considerations may provide good reason to separate out relationships maintained to distinct tables (or records).

FIG. **35**D depicts a preferred embodiment of a Privilege Data Record (PDR) **3530** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. A privilege ID field **3530***a* contains a unique number associated to a supported privilege (i.e. "atomic privilege for assignment"). Field **3530***a* associates a MS relevance field **3530***b* to a particular privilege for indicating the MS types which apply to a privilege. There should not be more than one PDR **3530** at a MS with matching fields **3530***a* since the associated field **3530***b* defines the MS types which are relevant for that privilege. If there is no record **3530** for a particular privilege, then it is preferably assumed that all MSs participate with the privilege. MS relevance field **3530***b* is preferably a bit mask accommodating all anticipated MS types, such that a 1 in a predefined MS type bit position indicates the MS participates with the privilege, and a 0 in a predefined MS type bit position indicates the MS does not participate with the privilege. Optimally, there are no records **3530** at a MS which implies all supported privileges interoperate fully with other MSs according to the present disclosure.

FIG. **35**E depicts a preferred embodiment of a Group Data Record (GRPDR) **3540** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. A GRPDR **3540** is the main data record for defining a group. A group identifier (group ID) field **3540***a* contains a unique number generated for the record **3540** to distinguish it from all other records **3540** maintained. Field **3540***a* is to be maintained similarly to as described for field **3500***a* (e.g. primary key column, correlation generation, facilitates well performing I/O). An owner information (info) field **3540***b* provides who the owner (creator and/or maintainer) is of the GRPDR **3540**. Field **3540***b* is to be maintained similarly to as described for field **3500***b* (e.g. embodiments for like ID and type pair, two (2) fields, removal because MS user information understood to be owner). A group name field **3540***c* provides the name of the group.

FIG. **36**A depicts a preferred embodiment of a Description Data Record (DDR) **3600** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. A DDR **3600** is for maintaining description information for certain constructs. A description ID field **3600***a* provides an identifier of the data record associated to the description field **3600***c*. For example, values maintained to field **3600***a* are used for joining data records in queries in an SQL embodiment. Values maintained to field **3600***a* include values of granting ID field **3500***a*, grant ID field **3510***a*, a privilege ID (e.g. as candidate to field **3530***a*), ID field **3500***c*, ID field **3500***e*, charter ID field **3700***a*, action ID field **3750***a*, parameter ID field **3775***a*, group ID field **3540***a*, or any other ID field for associating a description. A description type field **3600***b* contains the type of data record to be associated (e.g. joined) to the description field **3600***c*. Values maintained to field **3600***b* include Permission, Grant, Privilege, ID, Charter, Action, Parameter, or Group in accordance with a value of field **3600***a*. Field **3600***c* contains a description, for example a user defined text string, to be associated to the data described by fields **3600***a* and **3600***b*. Alternate embodiments will move the description data to a new field of the data record being associated to, or distinct record definitions **3600**-*y* may be defined for any subset of relationship/association to prevent data access performance of one relationship/association from impacting performance accesses of another relationship/association maintained (analogous to distinct embodiments for GADR **3520**).

FIG. **36**B depicts a preferred embodiment of a History Data Record (HDR) **3620** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. A HDR **3620** is for maintaining history information for certain constructs. A history ID field **3620***a* provides an identifier of the data record associated to the history field **3620***c*. For example, values maintained to field **3620***a* are used for joining data records in queries in an SQL embodiment. Values maintained to field **3620***a* include values of granting ID field **3500***a*, grant ID field **3510***a*, a privilege ID (e.g. as candidate to field **3530***a*), ID field **3500***c*, ID field **3500***e*, charter ID field **3700***a*, action ID field **3750***a*, parameter ID field **3775***a*, group ID field **3540***a*, or any other ID field for associating a history. A history type field **3620***b* contains the type of data record to be associated (e.g. joined) to the history field **3620***c*. Values maintained to field **3620***b* include Permission, Grant, Privilege, ID, Charter, Action, Parameter, or Group in accordance with a value of field **3620***a*. Field **3620***c* contains a history, for example a collection of fields for describing the creation and/or maintenance of data associated to the data described by fields **3620***a* and **3620***b*. Alternate embodiments will move the history data to new field(s) of the data record being associated to, or distinct record definitions **3620**-*x* may be defined for any subset of relationship/association to prevent data access performance of one relationship/association from impacting performance accesses of another relationship/association maintained (analogous to distinct embodiments for GADR **3520**). Another embodiment may break out subfields of field **3620***c* to fields **3620***c*-**1**, **3620***c*-**2**, **3620***c*-**3**, etc. for individual fields accesses (e.g. see CreatorInfo and ModifierInfo sub-fields).

FIG. **36**C depicts a preferred embodiment of a Time specification Data Record (TDR) **3640** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. A TDR **3640** is for maintaining time spec information for certain constructs. A time spec ID field **3640***a* provides an identifier of the data record associated to the time spec field **3640***c*. For example, values maintained to field **3640***a* are used for joining data records in queries in an SQL embodiment. Values maintained to field **3640***a* include values of granting ID field **3500***a*, grant ID field **3510***a*, a privilege ID (e.g. as candidate to field **3530***a*), charter ID field **3700***a*, action ID field **3750***a*, or any other ID field for associating a time spec (specification). A time spec type field **3640***b* contains the type of data record to be associated (e.g. joined) to the time spec field **3640***c*. Values maintained to field **3640***b* include Permission, Grant, Privilege, Charter, or Action in accordance with a value of field **3640***a*. Field **3640***c* contains a time spec, for example one or more fields for describing the date/time(s) for which the data associated to the data described by fields **3640***a* and **3640***b* is applicable, enabled, or active. For example, permissions can be granted as enabled for particular time period(s). Alternate embodiments will move the time spec data to new field(s) of the data record being associated to, or distinct record definitions **3640**-*w* may be defined for any subset of relationship/association to prevent data access performance of one relationship/association from impacting performance accesses of another relationship/association maintained (analogous to distinct embodiments for GADR **3520**). Another embodiment may break out subfields of field **3640***c* to fields **3640***c*-**1**, **3640***c*-**2**, **3620***c*-**3**, etc. Field **3640***c* (and sub-fields if embodiment applicable) can describe specific date/time(s) or date/time period(s). Yet another embodiment, maintains plural TDRs for a data record of ID field **3640***a*. Field **3640***c* is intended to qualify the associated data of fields **3640***a* and **3640***b* for being applicable, enabled, or active at future time(s), past time(s), or current time(s). An alternate embodiment of field **3640***c* may include a special tense qualifier as defined below:

- Past ("P"): indicates that the associated data record (e.g. permission, charter, action, etc) applies to all WDR information maintained to LBX History **30**;
- Self Past ("SP"): indicates that the associated data record (e.g. permission, charter, action, etc) applies to only WDR information maintained to LBX History **30** for the MS owning history **30**;
- Other Past ("OP"): indicates that the associated data record (e.g. permission, charter, action, etc) applies to only WDR information maintained to LBX History **30** for all MSs other than the one owning history **30**;
- Future ("F"): indicates that the associated data record (e.g. permission, charter, action, etc) applies to all WDRs created/received (e.g. inserted to queue **22**) in the future by the MS (i.e. after configuration made);
- Self Future ("SF"): indicates that the associated data record (e.g. permission, charter, action, etc) applies to all WDRs created in the future (e.g. inserted to queue **22**) by the MS for its own whereabouts (i.e. after configuration made);
- Other Future ("OF"): indicates that the associated data record (e.g. permission, charter, action, etc) applies to all WDRs received (e.g. inserted to queue **22**) in the future by the MS for other MS whereabouts (i.e. after configuration made);
- All ("A"): indicates that the associated data record (e.g. permission, charter, action, etc) applies to all WDRs created/received in the future by the MS (i.e. after configuration made) and WDRs already contained by queue **22**;
- Self All ("SA"): indicates that the associated data record (e.g. permission, charter, action, etc) applies to all WDRs created in the future by the MS for its own whereabouts (i.e. after configuration made) and WDRs already contained by queue **22** for the MS;
- Other All ("OA"): indicates that the associated data record (e.g. permission, charter, action, etc) applies to all WDRs received in the future by the MS for other MS whereabouts (i.e. after configuration made) and WDRs already contained by queue **22** for other MSs; and/or
- Any combination of above (e.g. "SF, OA, OP")

A syntactical equivalent may be specified for subsequent internalization causing configurations to immediately take effect. Another embodiment qualifies which set of MSs to apply time specification for, but this is already accomplished below in the preferred embodiment through specifications of conditions. Yet another embodiment provides an additional qualifier specification for which WDRs to apply the time specification: WDRs maintained by the MS (e.g., to queue **22**), inbound WDRs as communicated to the MS, outbound WDRs as communicated from the MS; for enabling applying of time specifications before and/or after privileges/charters are applied to WDRs with respect to an MS. Blocks **3970**, **4670** and **4470** may be amended to include processing for immediately checking historical information maintained at the MS which privileges/charters have relevance, for example after specifying a historical time specification or special tense qualifier.

FIG. **36**D depicts a preferred embodiment of a Variable Data Record (VDR) **3660** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. A VDR **3660** contains variable information that may be instantiated. A record **3660** provide a single place to define an encoding that is instantiated in many places. One advantage is for saving on encoding sizes. An owner information (info) field **3660***a* provides who the owner (creator and/or maintainer) is of the VDR **3660**. Field **3660***a* is to be maintained similarly to as described for field **3500***b* (e.g. embodiments for like ID and type pair, two (2) fields, removal because grantor information understood to be owner). Variable name field **3660***b* contains the variable name string, variable type field **3660***c* contains the variable type, and variable value field **3660***d* contains the value(s) of the variable for instantiation. Preferably, field **3660***d* remains in its original form until the variable is instantiated. For example, in an X.409 embodiment, field **3660***d* contains the X.409 encoding datastream (including the overall length for starting bytes) of the variable value. In a programming source, XML, or other syntactical embodiment (of grammar of FIGS. **30**A through **30**F), field **3660***d* contains the unelaborated syntax in text form for later processing (e.g. stack processing). Thus, field **3660***d* may be a BLOB (Binary Large Object) or text. Preferably, field **3660***d* is not elaborated, or internalized, until instantiated. When a variable is set to another variable name, field **3660***d* preferably contains the variable name and the variable type field **3660***c* indicates Variable. Preferably, field **3660***d* handles varying length data well for performance, or an alternate embodiment will provide additional VDR field(s) to facilitate performance.

FIG. **37**A depicts a preferred embodiment of a Charter Data Record (CDR) **3700** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. A CDR **3700** is the main data record for defining a charter. A charter identifier (charter ID) field **3700***a* contains a unique number generated for the record **3700** to distinguish it from all other records **3700** maintained. Field **3700***a* is to be maintained similarly to as described for field **3500***a* (e.g. primary key column, correlation generation, facilitates well performing I/O). Grantee and Grantor information is linked to with a match of field **3700***a* with **3500***a*. An alternate embodiment will require no Grantee or Grantor specification for a charter (e.g. charters maintained and used at the user's MS). An owner information (info) field **3700***b* provides who the owner (creator and/or maintainer) is of the CDR **3700**. Field **3700***b* is to be maintained similarly to as described for field **3500***b* (e.g. embodiments for like ID and type pair, two (2) fields, removal because MS user information understood to be owner). An expression field **3700***c* contains the expression containing one or more conditions for when to perform action(s) of action field **3700***d*. Preferably, field **3700***c* remains in its original form until the conditions are to be elaborated, processed, or internalized. For example, in one X.409 embodiment, field **3700***c* contains the X.409 encoding datastream for the entire Expression TLV. In the preferred syntactical embodiment (programming source code, XML encoding, programming source code enhancement, or the like), field **3700***c* contains the unelaborated syntax in text form for later stack processing of conditions and terms and their subordinate constructs. Thus, field **3700***c* may be a BLOB (Binary Large Object) or (preferably) text. An alternate embodiment to field **3700***c* may use General Assignment Data Records (GADRs) **3520** to assign condition identifier fields of a new condition data record to charter identifier fields **3700***a* (to prevent a single field from holding an unpredictable number of conditions for the charter of record **3700**). Actions field **3700***d* contains an ordered list of one or more action identifiers **3750***a* of actions to be performed when the expression of field **3700***c* is evaluated to TRUE. For example, in the preferred syntactical embodiment, when actions field **3700***d* contains "45,2356,9738", the action identifier fields **3750***a* have been identified as an ordered list of actions 45, 2356 and 9738 which are each an action identifier contained in an ADR **3750** field **3750***a*. An alternate embodiment to field **3700***d* will use General Assignment Data Records (GADRs) **3520** to assign action identifier fields **3750***a* to charter identifier fields **3700***a* (to prevent a single field from holding an unpredictable number of actions for the charter of record **3700**). Another alternative embodiment may include Grantor and Grantee information as part of the CDR (e.g. new fields **3700***e* through **3700***h* like fields **3500***c* through **3500***f*).

FIG. **37**B depicts a preferred embodiment of an Action Data Record (ADR) **3750** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. An action identifier (action ID) field **3750***a* contains a unique number generated for the record **3750** to distinguish it from all other records **3750** maintained. Field **3750***a* is to be maintained similarly to as described for field **3500***a* (e.g. primary key column, correlation generation, facilitates well performing I/O). An owner information (info) field **3750***b* provides who the owner (creator and/or maintainer) is of the ADR **3750**. Field **3750***b* is to be maintained similarly to as described for field **3500***b* (e.g. embodiments for like ID and type pair, two (2) fields, removal because MS user information understood to be owner). Host field **3750***c* contains the host (if not null) for where the action is to take place. An alternate embodiment allows multiple host specification(s) for the action. Host type field **3750***d* qualifies the host field **3750***c* for the type of host(s) to perform the action (helps interpret field **3750***c*). An alternate embodiment allows multiple host type specifications for multiple host specifications for the action. Yet another embodiment uses a single host field **3750***c* to join to a new table for gathering all applicable hosts for the action. Command field **3750***e* contains an "atomic command" (such as those found at the top of FIG. **34**D), operand field **3750***f* contains an "atomic operand" (e.g. such as those found at the bottom of FIG. **34**D), and parameter IDs field **3750***g* contains a list of null, one or more parameter identifiers **3775***a* (an ordered list) for parameters in accordance with the combination of command field **3750***e* and operand field **3750***f* (see FIGS. **31**A through **31**E for example parameters). There is a list of supported commands, list of supported operands, and a set of appropriate parameters depending on the combination of a particular command with a particular operand. In the preferred syntactical embodiment, when parameter IDs field **3750***g* contains "234, 18790", the parameter IDs fields **3775***a* have been identified as an ordered list of parameters 234 and 18790 which are each a parameter identifier contained in a record **3775** field **3775***a*. An alternate embodiment to field **3750***g* will use General Assignment Data Records (GADRs) **3520** to assign parameter identifier fields **3775***a* to action identifier fields **3750***a* (to prevent a single field from holding an unpredictable number of parameters for the action of record **3750**).

FIG. **37**C depicts a preferred embodiment of a Parameter Data Record (PARMDR) **3775** for discussing operations of the present disclosure, derived from the grammar of FIGS. **30**A through **30**E. A parameter identifier (parameter ID) field **3775***a* contains a unique number generated for the record **3775** to distinguish it from all other records **3775** maintained. Field **3775***a* is to be maintained similarly to as described for field **3500***a* (e.g. primary key column, correlation generation, facilitates well performing I/O). An owner information (info) field **3775***b* provides who the owner (creator and/or maintainer) is of the record **3775**. Field **3775***b* is to be maintained similarly to as described for field **3500***b* (e.g. embodiments for like ID and type pair, two (2) fields, removal because MS user information understood to be owner). Parameters field **3775***c* contains one or more parameters pointed to by data of field **3750***g*, preferably in a conveniently parsed form. Field **3750***g* can point to a single record **3775** which contains a plurality of parameters in field **3775***c*, or field **3750***g* can specify a plurality of parameters pointing to plural records **3775**, each containing parameter information in fields **3775***c*.

In one embodiment, data can be maintained to data records of FIGS. **35**A through **37**C, and FIG. **53**, such that it is marked as enabled or disabled (e.g. additional column in SQL table for enabled/disabled). In another embodiment, a record is configured in disabled form and then subsequently enabled, for example with a user interface. Any subset of data records may be enabled or disabled as a related set. Privileges may be configured for which subsets can be enabled or disabled by a user. In another embodiment, privileges themselves enable or disable a data record, a subset of data records, a subset of data record types, or a subset of data of data records.

Data records were derived from the BNF grammar of FIGS. **30**A through **30**E. Other data record embodiments may exist. In a preferred embodiment, data records of FIGS. **35**A through **37**C are maintained to persistent storage of the MS. A MS used for the first time should be loaded with a default set of data (e.g. starter templates containing defaulted data) preloaded to the data records for user convenience. Loading may occur from local storage or from remotely loading, for example over a communications channel when first initializing the MS (e.g. enhanced block **1214** for additionally ensuring the data records are initialized, in particular for the first startup of an MS). Owner fields (e.g. field **3500***b*) for preloaded data are preferably set to a system identity for access and use by all users. Preferably, a user cannot delete any of the system preloaded data. While the data records themselves are enough to operate permissions **10** and charters **12** at the MS after startup, a better performing internalization may be preferred. For example, block **1216** can be enhanced for additionally using data records to internalize to a non-persistent well performing form such as compiled C encoding of FIGS. **34**A through **34**G (also see FIG. **52**), and block **2822** can be enhanced for additionally using the internalized data to write out to data records maintained in persistent storage. Any compiled/interpreted programming source code may be used without departing from the spirit and scope of the disclosure. FIGS. **34**A through **34**G (also see FIG. **52**) are an example, but may provide an internalized form for processing. In any case, many examples are provided for encoding permissions **10** and charters **12**. Continuing with the data record examples, for example a persistent storage form of data records in a MS local SQL database (e.g. a data record corresponds to a particular SQL table, and data record fields correspond to the SQL table columns), flowcharts **38** through **48**B are provided for configuration of permissions **10** and charters **12**. Data records are to be maintained in a suitable MS performance conscious form (may not be an SQL database). An "s" is added as a suffix to disclosed acronyms (e.g. GDR) to reference a plural version of the acronym (e.g. GDRs=Granting Data Records).

FIGS. **35**A through **37**C assume an unlimited number of records (e.g. objects) to accomplish a plurality of objects (e.g. BNF grammar constructs). In various embodiments, a high maximum number plurality of the BNF grammar derived objects is supported wherever possible. In various embodiments, any MS storage or memory means, local or remotely attached, can be used for storing information of an implemented derivative of the BNF grammar of this disclosure. Also, various embodiments may use a different model or schema to carry out functionality disclosed. Various embodiments may use an SQL database (e.g. Oracle, SQL Server, Informix, DB2, etc. for storing information, or a non-SQL database form (e.g. data or record retrieval system, or any interface for accessible record formatted data).

It is anticipated that management of permissions **10** and charters **12** be as simple and as lean as possible on an MS. Therefore, a reasonably small subset of the FIGS. **30**A through **30**E grammar is preferably implemented. While FIGS. **35**A through **48**B demonstrate a significantly large derivative of the BNF grammar, the reader should appreciate that this is to "cover all bases" of consideration, and is not necessarily a derivative to be incorporated on a MS of limited processing capability and resources. A preferred embodiment is discussed, but much smaller derivatives are even more preferred on many MSs. Appropriate semaphore lock windows are assumed incorporated when multiple asynchronous threads can access the same data concurrently.

FIG. **38** depicts a flowchart for describing a preferred embodiment of MS permissions configuration processing of block **1478**. FIG. **38** is of Self Management Processing code **18**. Processing starts at block **3802** and continues to block **3804** where a list of permissions configuration options are presented to the user. Thereafter, block **3806** waits for a user action in response to options presented. Block **3806** continues to block **3808** when a user action has been detected. If block **3808** determines the user selected to configure permissions data, then the user configures permissions data at block **3810** (see FIG. **39**A) and processing continues back to block **3804**. If block **3808** determines the user did not select to configure permissions data, then processing continues to block **3812**. If block **3812** determines the user selected to configure grants data, then the user configures grants data at block **3814** (see FIG. **40**A) and processing continues back to block **3804**. If block **3812** determines the user did not select to configure grants data, then processing continues to block **3816**. If block **3816** determines the user selected to configure groups data, then the user configures groups data at block **3818** (see FIG. **41**A) and processing continues back to block **3804**. If block **3816** determines the user did not select to configure groups data, then processing continues to block **3820**. If block **3820** determines the user selected to view other's groups data, then block **3822** invokes the view other's info processing of FIG. **42** with GROUP_INFO as a parameter (for viewing other's groups data information) and processing continues back to block **3804**. If block **3820** determines the user did not select to view other's groups data, then processing continues to block

3824. If block **3824** determines the user selected to view other's permissions data, then block **3826** invokes the view other's info processing of FIG. **42** with PERMISSION_INFO as a parameter (for viewing other's permissions data information) and processing continues back to block **3804**. If block **3824** determines the user did not select to view other's permissions data, then processing continues to block **3828**. If block **3828** determines the user selected to view other's grants data, then block **3830** invokes the view other's info processing of FIG. **42** with GRANT_INFO as a parameter (for viewing other's grants data information) and processing continues back to block **3804**. If block **3828** determines the user did not select to view other's grants data, then processing continues to block **3832**. If block **3832** determines the user selected to send permissions data, then block **3834** invokes the send data processing of FIG. **44**A with PERMISSION_INFO as a parameter (for sending permissions data) and processing continues back to block **3804**. If block **3832** determines the user did not select to send permissions data, then processing continues to block **3836**. If block **3836** determines the user selected to configure accepting permissions, then block **3838** invokes the configure acceptance processing of FIG. **43** with PERMISSION_INFO as a parameter (for configuring acceptance of permissions data) and processing continues back to block **3804**. If block **3836** determines the user did not select to configure accepting permissions, then processing continues to block **3840**. If block **3840** determines the user selected to exit block **1478** processing, then block **3842** completes block **1478** processing. If block **3840** determines the user did not select to exit, then processing continues to block **3844** where all other user actions detected at block **3806** are appropriately handled, and processing continues back to block **3804**.

In an alternate embodiment where the MS maintains GDRs **3500**, GRTDRs **3510**, GADRs **3520**, PDRs **3530** and GRPDRs **3540** (and their associated data records DDRs, HDRs and TDRs) at the MS where they were configured, FIG. **38** may not provide blocks **3820** through **3830**. The MS may be aware of its user permissions and need not share the data (i.e. self contained). In some embodiments, options **3820** through **3830** cause access to locally maintained data for others (other users, MSs, etc) or cause remote access to data when needed (e.g. from the remote MSs). In the embodiment where no data is maintained locally for others, blocks **3832** through **3838** may not be necessary. The preferred embodiment is to locally maintain permissions data for the MS user and others (e.g. MS users) which are relevant to provide the richest set of permissions governing MS processing at the MS.

FIGS. **39**A through **39**B depict flowcharts for describing a preferred embodiment of MS user interface processing for permissions configuration of block **3810**. With reference now to FIG. **39**A, processing starts at block **3902**, continues to block **3904** for initialization (e.g. a start using database command), and then to block **3906** where groups the user is a member of are accessed. Block **3906** retrieves all GRPDRs **3540** joined to GADRs **3520** such that the descendant type field **3520***c* and descendant ID field **3520***d* match the user information, and the ascendant type field **3520***a* is set to Group and the ascendant ID field **3520***b* matches the group ID field **3540***a*. While there may be different types of groups as defined for the BNF grammar, the GRPDR is a derivative embodiment which happens to not distinguish. Alternate embodiments may carry a group type field to select appropriate records by group type. Yet another embodiment may not have a block **3906** with processing at block **3908** for gathering data additionally by groups the user is a member of. Block **3906** continues to block **3908**.

Block **3908** accesses all GDRs (e.g. all rows from a GDR SQL table) for the user of FIG. **39**A matching field **3500***t* to Permission, and the owner information of the GDRs (e.g. user information matches field **3500***b*) to the user and to groups the user is a member of (e.g. group information matches field **3500***b* (e.g. owner type=group, owner id=a group ID field **3540***a* from block **3906**). The GDRs are additionally joined (e.g. SQL join) with DDRs and TDRs (e.g. fields **3600***b* and **3640***b*=Permission and by matching ID fields **3600***a* and **3640***a* with field **3500***a*). Description field **3600** may provide a useful description last saved by the user for the permission entry. Block **3908** may also retrieve system predefined data records for use and/or management. Thereafter, each joined entry returned at block **3908** is associated at block **3910** with the corresponding data IDs (at least fields **3500***a* and **3540***a*) for easy unique record accesses when the user acts on the data. Block **3910** also initializes a list cursor to point to the first list entry to be presented to the user. Thereafter, block **3912** sets user interface indication for where the list cursor is currently set (e.g. set to highlight the entry), and any list scrolling settings are set (the list is initially not set for being scrolled on first FIG. **39**A processing encounter to block **3912** from block **3910**). Block **3912** continues to block **3914** where the entry list is presented to the user in accordance with the list cursor and list scroll settings managed for presentation at block **3912**. Thereafter, block **3916** waits for user action to the presented list of permissions data and will continue to block **3918** when a user action has been detected. Presentation of the scrollable list preferably presents in an entry format such that an entry contains fields for: DDR **3600** description; GDR owner information, grantor information and grantee information; GRPDR owner information and group name if applicable; and TDR time spec information. Alternate embodiments will present less information, or more information (e.g. GRTDR(s) **3510** and/or PDR(s) **3530** via GADR(s) **3520** joining fields (e.g. **3500***a*, **3510***a*, **3520***b*)).

If block **3918** determines the user selected to set the list cursor to a different entry, then block **3920** sets the list cursor accordingly and processing continues back to block **3912**. Block **3912** always sets for indicating where the list cursor is currently pointed and sets for appropriately scrolling the list if necessary when subsequently presenting the list at block **3914**. If block **3918** determines the user did not select to set the list cursor, then processing continues to block **3922**. If block **3922** determines the user selected to add a permission, then block **3924** accesses a maximum number of permissions allowed (perhaps multiple maximum values accessed), and block **3926** checks the maximum(s) with the number of current permissions defined. There are many embodiments for what deems a maximum (for this user, for a group, for this MS, etc). If block **3926** determines a maximum number of permissions allowed already exists, then block **3928** provides an error to the user and processing continues back to block **3912**. Block **3928** preferably requires the user to acknowledge the error before continuing back to block **3912**. If block **3926** determines a maximum was not exceeded, then block **3930** interfaces with the user for entering validated permission data and block **3932** adds the data record(s), appropriately updates the list with the new entry, and sets the list cursor appropriately for the next list presentation refresh, before continuing back to block **3912**. If block **3922** determines the user did not want to add a permission, processing continues to block **3934**. Block **3932** will add a GDR **3500**, DDR **3600**, HDR **3620** (to set creator information) and TDR **3640**. The DDR and TDR are optionally added by the user, but the DDR may be strongly suggested (if not enforced on the add). This will provide a permission record assigning all privileges from the grantor to the grantee. Additionally, blocks **3930/3932** may support adding new GADR(s) **3520** for assigning certain grants and/or privileges (which are validated to exist prior to adding data at block **3932**).

If block **3934** determines the user selected to delete a permission, then block **3936** deletes the data record currently pointed to by the list cursor, modifies the list for the discarded entry, and sets the list cursor appropriately for the next list presentation refresh, before continuing back to block **3912**. Block **3936** will use the granting ID field **3500***a* (associated with the entry at block **3910**) to delete the permission. Associated GADR(s) **3520**, DDR **3600**, HDR **3620**, and TDR **3640** is also deleted (e.g. preferably with a cascade delete in a SQL embodiment). If block **3934** determines the user did not select to delete a permission, then processing continues to block **3952** of FIG. **39**B by way of off-page connector **3950**.

With reference now to FIG. **39**B, if block **3952** determines the user selected to modify a permission, then block **3954** interfaces with the user to modify permission data of the entry pointed to by the list cursor. The user may change information of the GDR and any associated records (e.g. DDR, TDR and GADR(s)). The user may also add the associated records at block **3954**. Block **3954** waits for a user action indicating completion. Block **3954** will continue to block **3956** when the complete action is detected at block **3954**. If block **3956** determines the user exited, then processing continues back to block **3912** by way of off-page connector **3998**. If block **3956** determines the user selected to save changes made at block **3954**, then block **3958** updates the data and the list is appropriately updated before continuing back to block **3912**. Block **3958** may update the GDR and/or any associated records (e.g. GADR(s), DDR, and/or TDR) using the permission id field **3500***a* (associated to the entry at block **3910**). Block **3958** will update an associated HDR as well. Block **3958** may add new GADR(s), a DDR and/or TDR as part of the permission change. If block **3952** determines the user did not select to modify a permission, then processing continues to block **3960**.

If block **3960** determines the user selected to get more details of the permission (e.g. show all joinable data to the GDR that is not already presented with the entry), then block **3962** gets additional details (may involve database queries in an SQL embodiment) for the permission pointed to by the list cursor, and block **3964** appropriately presents the information to the user. Block **3964** then waits for a user action that the user is complete reviewing details, in which case processing continues back to block **3912**. If block **3960** determines the user did not select to get more detail, then processing continues to block **3966**.

If block **3966** determines the user selected to internalize permissions data thus far being maintained, then block **3968** internalizes (e.g. as a compiler would) all applicable data records for well performing use by the MS, and block **3970** saves the internalized form, for example to MS high speed non-persistent memory. In one embodiment, blocks **3968** and **3970** internalize permission data to applicable C structures of FIGS. **34**A through **34**G (also see FIG. **52**). In various embodiments, block **3968** maintains statistics for exactly what was internalized, and updates any running totals or averages maintained for a plurality of internalizations up to this point, or over certain time periods. Statistics such as: number of active constructs; number of user construct edits of particular types; amount of associated storage used, freed, changed, etc with perhaps a graphical user interface to graph changes over time; number of privilege types specified, number of charters affected by permissions; and other permission dependent statistics. In other embodiments, statistical data is initialized at internalization time to prepare for subsequent gathering of useful statistics during permission processing. In embodiments where a tense qualifier is specified for TimeSpec information, saving the internalized form at block **3970** causes all past and current tense configurations to become effective for being processed.

Bock **3970** then continues back to block **3912**. If block **3966** determines the user did not select to internalize permission configurations, then processing continues to block **3972**. Alternate embodiments of processing permissions **10** in the present disclosure will rely upon the data records entirely, rather than requiring the user to redundantly internalize from persistent storage to non-persistent storage for use. Persistent storage may be of reasonably fast performance to not require an internalized version of permission **10**. Different embodiments may completely overwrite the internalized form, or update the current internalized form with any changes.

If block **3972** determines the user selected to exit block **3810** processing, then block **3974** cleans up processing thus far accomplished (e.g. issue a stop using database command), and block **3976** completes block **3810** processing. If block **3972** determines the user did not select to exit, then processing continues to block **3978** where all other user actions detected at block **3916** are appropriately handled, and processing continues back to block **3916** by way off off-page connector **3996**.

FIGS. **40**A through **40**B depict flowcharts for describing a preferred embodiment of MS user interface processing for grants configuration of block **3814**. With reference now to FIG. **40**A, processing starts at block **4002**, continues to block **4004** for initialization (e.g. a start using database command), and then to block **4006** where groups the user is a member of are accessed. Block **4006** retrieves all GRPDRs **3540** joined to GADRs **3520** such that the descendant type field **3520***c* and descendant ID field **3520***d* match the user information, and the ascendant type field **3520***a* is set to Group and the ascendant ID field **3520***b* matches the group ID field **3540***a*. While there may be different types of groups as defined for the BNF grammar, the GRPDR **3540** is a derivative embodiment which happens to not distinguish. Alternate embodiments may carry a group type field to select appropriate records by group type. Yet another embodiment may not have a block **4006** with processing at block **4008** for gathering data additionally by groups the user is a member of. Block **4006** continues to block **4008**.

Block **4008** accesses all GRTDRs **3510** (e.g. all rows from a GRTDR SQL table) for the user of FIG. **40**A matching the owner information of the GRTDRs (e.g. user information matches field **3510***b*) to the user and to groups the user is a member of (e.g. group information matches field **3510***b* (e.g. owner type=group, owner id=group ID field **3540***a* from block **4006**). The GRTDRs **3510** are additionally joined (e.g. SQL join) with DDRs **3600** and TDRs **3640** (e.g. fields **3600***b* and **3640***b*=Grant and by matching ID fields **3600***a* and **3640***a* with field **3510***a*). Description field **3600***c* can provide a useful description last saved by the user for the grant data, however the grant name itself is preferably self documenting. Block **4008** may also retrieve system predefined data records for use and/or management. Block **4008** will also retrieve grants within grants to present the entire tree structure for a grant entry. Block **4008** retrieves all GRTDRs **3510** joined to other GRTDRs **3510** through GADRs **3520** which will provide the grant tree structure hierarchy. Grants can be descendant to other grants in a grant hierarchy. Descendant type field **3520***c* set to Grant and descendant ID field **3520***d* for a particular grant will be a descending grant to an ascending grant of ascendant type field **3520***a* set to Grant and ascendant ID field **3520***b*. Therefore, each list entry is a grant entry that may be any node of a grant hierarchy tree. There may be grant information redundantly presented, for example when a grant is subordinate to more than one grant, but this helps the user know a grant tree structure if one has been configured. A visually presented embodiment may take the following form wherein a particular $Grant_i$ appears in the appropriate hierarchy form.

```
Grant Info_1
    Grant Info_11
    Grant Info_12
        Grant Info_121
        Grant Info_122
        ...
        Grant Info_12n
    ...
    Grant Info_1k
Grant Info_2
...
Grant Info_j
```

The list cursor can be pointing to any grant item within a single grant entry hierarchy. Thus, a single grant entry can be represented by a visual nesting, if applicable. Thereafter, each joined entry returned at block **4008** is associated at block **4010** with the corresponding data IDs (at least fields **3510***a* and **3540***a*) for easy unique record accesses when the user acts on the data. Block **4010** also initializes a list cursor to point to the first grant item to be presented to the user in the (possibly nested) list. Thereafter, block **4012** sets user interface indication for where the list cursor is currently set (e.g. set to highlight the entry) and any list scrolling settings are set (the list is initially not set for being scrolled on first FIG. **40**A processing encounter to block **4012** from block **4010**. Block **4012** continues to block **4014** where the entry list is presented to the user in accordance with the list cursor and list scroll settings managed for presentation at block **4012**. Thereafter, block **4016** waits for user action to the presented list of grant data and will continue to block **4018** when a user action has been detected. Presentation of the scrollable list preferably presents in an entry format with subordinate grants also reference-able by the list cursor. A grant entry of the grant tree presented preferably contains fields for: GRTDR name field **3510***c*; GRTDR owner information; GRPDR owner information and group name if applicable; TDR time spec information; and DDR information. Alternate embodiments will present less information, or more information (e.g. join PDR(s) **3530** via GADR(s) **3520** when applicable).

If block **4018** determines the user selected to set the list cursor to a different grant reference, then block **4020** sets the list cursor accordingly and processing continues back to block **4012**. Block **4012** always sets for indicating where the list cursor is currently pointed and sets for appropriately scrolling the list if necessary when subsequently presenting the list at block **4014**. If block **4018** determines the user did not select to set the list cursor, then processing continues to block **4022**. If block **4022** determines the user selected to add a grant, then block **4024** accesses a maximum number of grants allowed (perhaps multiple maximum values accessed), and block **4026** checks the maximum(s) with the number of current grants defined. There are many embodiments for what deems a maximum (for this user, for a group, for this MS, etc). If block **4026** determines a maximum number of grants allowed already exists, then block **4028** provides an error to the user and processing continues back to block **4012**. Block **4028** preferably requires the user to acknowledge the error before continuing back to block **4012**. If block **4026** determines a maximum was not exceeded, then block **4030** interfaces with the user for entering validated grant data and block **4032** adds the data record, appropriately updates the list with the new entry, and sets the list cursor appropriately for the next list presentation refresh, before continuing back to block **4012**. If block **4022** determines the user did not want to add a grant, processing continues to block **4034**. Block **4032** will add a GRTDR **3500**, DDR **3600**, HDR **3620** (to set creator information) and TDR **3640**. The DDR and TDR are optionally added by the user. Additionally, at block **4030** the user may add new GADR(s) **3520** for assigning certain grants to the added grant and/or privileges to the grant (which are validated to exist prior to adding data at block **4032**).

If block **4034** determines the user selected to modify a grant, then block **4036** interfaces with the user to modify grant data of the entry pointed to by the list cursor. The user may change information of the GRTDR and any associated records (e.g. DDR, TDR and GADR(s)). The user may also add the associated records at block **4036**. Block **4036** waits for a user action indicating completion. Block **4036** will continue to block **4038** when the action is detected at block **4036**. If block **4038** determines the user exited, then processing continues back to block **4012**. If block **4038** determines the user selected to save changes made at block **4036**, then block **4040** updates the data and the list is appropriately updated before continuing back to block **4012**. Block **4040** may update the GRTDR and/or any associated records (e.g. GADR(s), DDR, and/or TDR) using the grant id field **3510***a* (associated to the grant item at block **4010**). Block **4040** will update an associated HDR as well. Block **4036** may add new GADR(s), a DDR and/or TDR as part of the grant change. If block **4034** determines the user did not select to modify a grant, then processing continues to block **4052** by way of off-page connector **4050**.

With reference now to FIG. **40**B, if block **4052** determines the user selected to get more details of the grant (e.g. show all joinable data to the GRTDR that is not already presented with the entry), then block **4054** gets additional details (may involve database queries in an SQL embodiment) for the grant pointed to by the list cursor, and block **4056** appropriately presents the information to the user. Block **4056** then waits for a user action that the user is complete reviewing details, in which case processing continues back to block **4012** by way of off-page connector **4098**. If block **4052** determines the user did not select to get more detail, then processing continues to block **4058**.

If block **4058** determines the user selected to delete a grant, then block **4060** determines any data records (e.g. GADR(s) **3520**) that reference the grant data record to be deleted. Preferably, no ascending data records (e.g. GRTDRs) are joinable to the grant data record being deleted, otherwise the user may improperly delete a grant from a configured permission or other grant. In the case of descending grants, all may be cascaded deleted in one embodiment, provided no ascending grants exist for any of the grants to be deleted. The user should remove ascending references to a grant for deletion first. Block **4060** continues to block **4062**. If block **4062** determines there was at least one reference, block **4064** provides an appropriate error with the reference(s) found so the user can subsequently reconcile. Block **4064** preferably requires the user to acknowledge the error before continuing back to block **4012**. If no references were found as determined by block **4062**, then processing continues to block **4066** for deleting the data record currently pointed to by the list cursor, along with any other related records that can be deleted. Block **4066** also modifies the list for the discarded entry(s), and sets the list cursor appropriately for the next list presentation refresh, before continuing back to block **4012**. Block **4066** will use the grant ID field **3510***a* (associated with the entry at block **4010**) to delete a grant. Associated records (e.g. DDR **3600**, HDR **3620**, and TDR **3640**) are also deleted (e.g. preferably with a cascade delete in a SQL embodiment). If block **4058** determines the user did not select to delete a grant, then processing continues to block **4068**.

If block **4068** determines the user selected to exit block **3814** processing, then block **4070** cleans up processing thus far accomplished (e.g. issue a stop using database command), and block **4072** completes block **3814** processing. If block **4068** determines the user did not select to exit, then processing continues to block **4074** where all other user actions detected at block **4016** are appropriately handled, and processing continues back to block **4016** by way off off-page connector **4096**.

FIGS. **41**A through **41**B depict flowcharts for describing a preferred embodiment of MS user interface processing for groups configuration of block **3818**. With reference now to FIG. **41**A, processing starts at block **4102**, continues to block **4104** for initialization (e.g. a start using database command), and then to block **4106** where groups the user is a member of are accessed. Block **4106** retrieves all GRPDRs **3540** joined to GADRs **3520** such that the descendant type field **3520***c* and descendant ID field **3520***d* match the user information, and the ascendant type field **3520***a* is set to Group and the ascendant ID field **3520***b* matches the group ID field **3540***a*. While there may be different types of groups as defined for the BNF grammar, the GRPDR **3540** is a derivative embodiment which happens to not distinguish. Alternate embodiments may carry a group type field to select appropriate records by group type. Yet another embodiment may not have a block **4106** with processing at block **4108** for gathering data additionally by groups the user is a member of. Block **4106** continues to block **4108**.

Block **4108** accesses all GRPDRs **3540** (e.g. all rows from a GRPDR SQL table) for the user of FIG. **41**A matching the owner information of the GRPDRs (e.g. user information matches field **3540***b*) to the user and to groups the user is a member of (e.g. group information matches field **3540***b* (e.g. owner type=group, owner id=group ID field **3540***a* from block **4106**). The GRPDRs **3540** are additionally joined (e.g. SQL join) with DDRs **3600** and TDRs **3640** (e.g. fields **3600***b* and **3640***b*=Group and by matching ID fields **3600***a* and **3640***a* with field **3540***a*). Description field **3600***c* can provide a useful description last saved by the user for the group data, however the group name itself is preferably self documenting. Block **4108** may also retrieve system predefined data records for use and/or management. Block **4108** will also retrieve groups within groups to present the entire tree structure for a group entry. Block **4108** retrieves all GRPDRs **3540** joined to other GRPDRs **3540** through GADRs **3520** which will provide the group tree structure hierarchy. Groups can be descendant to other groups in a group hierarchy. Descendant type field **3520***c* set to Group and descendant ID field **3520***d* for a particular group will be a descending group to an ascending group of ascendant type field **3520***a* set to Group and ascendant ID field **3520***b*. Therefore, each list entry is a group entry that may be any node of a group hierarchy tree. There may be group information redundantly presented, for example when a group is subordinate to more than one group, but this helps the user know a group tree structure if one has been configured. A visually presented embodiment may take the following form wherein a particular $Group_i$ appears in the appropriate hierarchy form.

```
Group Info_1
        Group Info_11
        Group Info_12
                Group Info_121
                Group Info_122
                ...
             Group Info_12u
        ...
        Group Info_1t
Group Info_2
...
Group Info_s
```

The list cursor can be pointing to any group item within a single group entry hierarchy. Thus, a single group entry can be represented by a visual nesting, if applicable. Thereafter, each joined entry returned at block **4108** is associated at block **4110** with the corresponding data IDs (at least fields **3540***a*) for easy unique record accesses when the user acts on the data. Block **4110** also initializes a list cursor to point to the first group item to be presented to the user in the (possibly nested) list. Thereafter, block **4112** sets user interface indication for where the list cursor is currently set (e.g. set to highlight the entry) and any list scrolling settings are set (the list is initially not set for being scrolled on first FIG. **41**A processing encounter to block **4112** from block **4110**). Block **4112** continues to block **4114** where the entry list is presented to the user in accordance with the list cursor and list scroll settings managed for presentation at block **4112**. Thereafter, block **4116** waits for user action to the presented list of group data and will continue to block **4118** when a user action has been detected. Presentation of the scrollable list preferably presents in an entry format with subordinate groups also reference-able by the list cursor. A group entry of the group tree presented preferably contains fields for: GRPDR name field **3540***c*; GRPDR owner information; owning GRPDR owner information and group name if applicable; TDR time spec information; and DDR information. Alternate embodiments will present less information, or more information (e.g. join to specific identities via GADR(s) **3520** when applicable).

If block **4118** determines the user selected to set the list cursor to a different group entry, then block **4120** sets the list cursor accordingly and processing continues back to block **4112**. Block **4112** always sets for indicating where the list cursor is currently pointed and sets for appropriately scrolling the list if necessary when subsequently presenting the list at block **4114**. If block **4118** determines the user did not select to set the list cursor, then processing continues to block **4122**. If block **4122** determines the user selected to add a group, then block **4124** accesses a maximum number of groups allowed (perhaps multiple maximum values accessed), and block **4126** checks the maximum(s) with the number of current groups defined. There are many embodiments for what deems a maximum (for this user, for a group, for this MS, etc). If block **4126** determines a maximum number of groups allowed already exists, then block **4128** provides an error to the user and processing continues back to block **4112**. Block **4128** preferably requires the user to acknowledge the error before continuing back to block **4112**. If block **4126** determines a maximum was not exceeded, then block **4130** interfaces with the user for entering validated group data and block **4132** adds the data record, appropriately updates the list with the new entry, and sets the list cursor appropriately for the next list presentation refresh, before continuing back to block **4112**. If block **4122** determines the user did not want to add a group, processing continues to block **4134**. Block **4132** will add a GRTDR **3500**, DDR **3600**, HDR **3620** (to set creator information) and TDR **3640**. The DDR and TDR are optionally added by the user. Additionally, at block **4130** the user may add new GADR(s) **3520** for assigning certain groups to the added group and/or identities to the group (which are validated to exist prior to adding data at block **4132**).

If block **4134** determines the user selected to modify a group, then block **4136** interfaces with the user to modify group data of the entry pointed to by the list cursor. The user may change information of the GRPDR and any associated records (e.g. DDR, TDR and GADR(s)). The user may also add the associated records at block **4136**. Block **4136** waits for a user action indicating completion. Block **4136** will continue to block **4138** when the complete action is detected at block **4136**. If block **4138** determines the user exited, then processing continues back to block **4112**. If block **4138** determines the user selected to save changes made at block **4136**, then block **4140** updates the data and the list is appropriately updated before continuing back to block **4112**. Block **4140** may update the GRPDR and/or any associated GADR(s), DDR, and/or TDR using the group id field **3540***a* associated to the group item at block **4110**. Block **4140** will update an associated HDR as well. Blocks **4136**/**4140** may support adding new GADR(s), a DDR and/or TDR as part of the group change. If block **4134** determines the user did not select to modify a group, then processing continues to block **4152** by way of off-page connector **4150**.

With reference now to FIG. **41**B, if block **4152** determines the user selected to get more details of the group (e.g. show all joinable data to the GRPDR that is not already presented with the entry), then block **4154** gets additional details (may involve database queries in an SQL embodiment) for the group pointed to by the list cursor, and block **4156** appropriately presents the information to the user. Block **4156** then waits for a user action that the user is complete reviewing details, in which case processing continues back to block **4112** by way of off-page connector **4198**. If block **4152** determines the user did not select to get more detail, then processing continues to block **4158**.

If block **4158** determines the user selected to delete a group, then block **4160** determines any data records (e.g. GADR(s) **3520**) that reference the group data record to be deleted. Preferably, no ascending data records (e.g. GRPDRs) are joinable to the group data record being deleted, otherwise the user may improperly delete a group from a configured permission or other group. In the case of descending groups, all may be cascaded deleted in one embodiment, provided no ascending groups exist for any of the groups to be deleted. The user should remove ascending references to a group for deletion first. Block **4160** continues to block **4162**. If block **4162** determines there was at least one reference, block **4164** provides an appropriate error with the reference(s) found so the user can subsequently reconcile. Block **4164** preferably requires the user to acknowledge the error before continuing back to block **4112**. If no references were found as determined by block **4162**, then processing continues to block **4166** for deleting the data record currently pointed to by the list cursor, along with any other related records that can be deleted. Block **4166** also modifies the list for the discarded entry(s), and sets the list cursor appropriately for the next list presentation refresh, before continuing back to block **4112**. Block **4166** will use the group ID field **3540***a* (associated with the entry at block **4110**) to delete the group. Associated records (e.g. DDR **3600**, HDR **3620**, and TDR **3640**) are also deleted (e.g. preferably with a cascade delete in a SQL embodiment). If block **4158** determines the user did not select to delete a group, then processing continues to block **4168**.

If block **4168** determines the user selected to exit block **3818** processing, then block **4170** cleans up processing thus far accomplished (e.g. issue a stop using database command), and block **4172** completes block **3818** processing. If block **4168** determines the user did not select to exit, then processing continues to block **4174** where all other user actions detected at block **4116** are appropriately handled, and processing continues back to block **4116** by way off off-page connector **4196**.

FIG. **42** depicts a flowchart for describing a preferred embodiment of a procedure for viewing MS configuration information of others. Processing starts at block **4202** and continues to block **4204** where an object type parameter is determined for which information to present to the user as passed by the caller of FIG. **42** processing (e.g. GROUP_INFO, PERMISSION_INFO, GRANT_INFO, CHARTER_INFO, ACTION_INFO or PARAMETER_INFO). Thereafter, block **4206** performs initialization (e.g. a start using database command), and then the user specifies owner information (criteria), at block **4208**, for the object type data records to present. No privilege is assumed required for browsing other's information since it is preferably local to the MS of the user anyway. Block **4208** continues to block **4210**.

In an alternative embodiment, block **4208** appropriately accesses privileges granted from the owner criteria to the user of FIG. **42** to ensure the user has a privilege to browse the data records (per object type parameter) of the specified owner. Block **4208** will provide an error when there is no privilege, and will continue to block **4210** when there is a privilege. Block **4208** may also provide a user exit option for continuing to block **4216** for cases the user cannot successfully specify owner criteria. In similar embodiments, there may be a separate privilege required for each object type a user may browse.

Block **4210** gets (e.g. SQL selects) data according to the object type parameter (e.g. GRPDR(s), GDR(s), GRTDR(s), CDR(s), ADR(s) or PARMDR(s), along with any available associated joinable data (e.g. DDR(s), HDR(s), TDR(s) and data records via GADR(s) if applicable), per object type passed). There are various embodiments to block **4210** in accessing data: locally maintained data for the owner criteria specified at block **4208**, communicating with a remote MS for accessing the MS of the owner criteria to synchronously pull the data, or sending a request to a remote MS over an interface like interface **1926** for then asynchronously receiving by an interface like interface **1948** for processing. One preferred embodiment is to locally maintain relevant data. In privilege enforced embodiments, appropriate privileges are determined before allowing access to the other's data.

Thereafter, if block **4212** determines there were no data records according to the object type passed by the caller for the owner criteria specified at block **4208**, then block **4214** provides an error to the user, and processing continues to block **4216**. Block **4216** performs cleanup of processing thus far accomplished (e.g. perform a stop using database command), and then continues to block **4218** for returning to the caller of FIG. **42** processing. Block **4214** preferably requires the user to acknowledge the error before continuing to block **4216**.

If block **4212** determines at least one data record of object type was found, then block **4220** presents a browse-able scrollable list of entries to the user (i.e. similar to lists discussed for presentation by FIGS. **39**A&B, FIGS. **40**A&B, FIGS. **41**A&B, FIGS. **46**A&B, FIGS. **47**A&B or FIGS. **48**A&B, per object typed passed), and block **4222** waits for a user action in response to presenting the list. When a user action is detected at block **4222**, processing continues to block **4224**. If block

4224 determines the user selected to specify new owner criteria (e.g. for comparison to field **3500***b*, **3510***b*, **3540***b*, **3700***b*, **3750***b* or **3775***b*, per object type passed) for browse, then processing continues back to block **4208** for new specification and applicable processing already discussed for blocks thereafter. If block **4224** determines the user did not select to specify new owner criteria, processing continues to block **4226**.

If block **4226** determines the user selected to get more detail of a selected list entry, then processing continues to block **4228** for getting data details of the selected entry, and block **4230** presents the details to the user, and waits for user action. Detail presentation is similar to getting detail processing discussed for presentation by FIGS. **39**A&B, FIGS. **40**A&B, FIGS. **41**A&B, FIGS. **46**A&B, FIGS. **47**A&B or FIGS. **48**A&B, per object typed passed. Block **4230** continues to block **4232** upon a user action (complete/clone).

If block **4232** determines the user action from block **4230** was to exit browse, processing continues to block **4220**. If block **4232** determines the user action from block **4230** was to clone the data (e.g. to make a copy for user's own use), processing continues to block **4234** for accessing permissions. Thereafter, if block **4236** determines the user does not have permission to clone, processing continues to block **4238** for reporting an error (preferably requiring the user to acknowledge before leaving block **4238** processing), and then back to block **4220**. If block **4236** determines the user does have permission to clone, processing continues to block **4240** where the data item browsed is appropriately duplicated with defaulted fields as though the user of FIG. **42** processing had created new data himself. Processing then continues back to block **4220**. If block **4226** determines the user did not select to get more detail on a selected item, then processing continues to block **4242**.

If block **4242** determines the user selected to exit browse processing, then processing continues to block **4216** already described. If block **4242** determines the user did not select to exit, then processing continues to block **4244** where all other user actions detected at block **4222** are appropriately handled, and processing continues back to block **4222**.

In an alternate embodiment, FIG. **42** will support cloning multiple entries in one action so that a first user conveniently makes use of a second user's data (like starter template(s)) for the first user to create/configure new data without entering it from scratch in the other interfaces disclosed. Another embodiment will enforce unique privileges for which data can be cloned by which user(s).

FIG. **43** depicts a flowchart for describing a preferred embodiment of a procedure for configuring MS acceptance of data from other MSs, for example permissions **10** and charters **12**. In a preferred embodiment, permissions **10** and charters **12** contain data for not only the MS **2** but also other MSs which are relevant to the MS **2** (e.g. MS users are known to each other). Processing starts at block **4302** and continues to block **4304** where a parameter passed by a caller is determined. The parameter indicates which object type (data type) to configure delivery acceptance (e.g. PERMISSION_INFO, CHARTER_INFO). Thereafter, block **4306** displays acceptable methods for accepting data from other MSs, preferably in a radio button form in a visually perceptible user interface embodiment. A user is presented with two (2) main sets of options, the first set preferably being an exclusive selection:

- Accept no data (MS will not accept data from any source); or
- Accept all data (MS will accept data from any source); or
- Accept data according to permissions (MS will accept data according to those sources which have permission to send certain data (perhaps privilege also specifies by a certain method) to the MS).

And the second set being:

- Targeted data packet sent or broadcast data packet sent (preferably one or the other);
- Electronic Mail Application;
- SMS message; and/or
- Persistent Storage Update (e.g. file system).

Block **4306** continues to block **4308** where the user makes a selection in the first set, and any number of selections in the second set. Thereafter, processing at block **4310** saves the user's selections for the object type parameter passed, and processing returns to the caller at block **4312**. LBX processing may have intelligence for an hierarchy of attempts such as first trying to send or broadcast, if that fails send by email, if that fails send by SMS message, and if that fails alert the MS user for manually copying over the data at a future time (e.g. when MSs are in wireless vicinity of each other). Block **4306** may provide a user selectable order of the attempt types. Intelligence can be incorporated for knowing which data was sent, when it was sent, and whether or not all of the send succeeded, and a synchronous or asynchronous acknowledgement can be implemented to ensure it arrived safely to destination(s). Applicable information is preferably maintained to LBX history **30** for proper implementation.

In one embodiment, the second set of configurations is further governed by individual privileges (each send type), and/or privileges per a source identity. For example, while configurations of the second set may be enabled, the MS will only accept data in a form from a source in accordance with a privilege which is enabled (set for the source identity). Privilege examples (may also each have associated time specification) include:

- Grant Joe privilege to send all types of data (e.g. charters and privileges, or certain (e.g. types, contents, features, any characteristic(s)) charters and/or privileges);
- Grant Joe privilege to send certain type of data (e.g. charters or privileges, or certain (e.g. types, contents, features, any characteristic(s)) charters and/or privileges);
- Grant Joe privilege to send certain type of data using certain method (privilege for each data type and method combination); and/or
- Grant Joe privilege to send certain type of data using certain method(s) (privilege for each data type and method combination) at certain time(s).

In another embodiment, there may be other registered applications (e.g. specified other email applications) which are candidates in the second set. This allows more choices for a receiving application with an implied receiving method (or user may specify an explicit method given reasonable choices of the particular application). For example, multiple MS instant messaging and/or email applications may be selectable in the second set of choices, and appropriately interfaced to for accepting data from other MSs. This allows specifying preferred delivery methods for data (e.g. charters and/or permissions data), and an attempt order thereof.

In some embodiments, charter data that is received may be received by a MS in a deactivated form whereby the user of the receiving MS must activate the charters for use (e.g. define a new charter enabled field **3700***e* for indicating whether or not the charter is active (Y=Yes, N=No)). New field **3700***e* may also be used by the charter originator for disabling or enabling for a variety of reasons. This permits a user to examine charters, and perhaps put them to a test, prior to putting them into use. Other embodiments support activating charters (received and/or originated): one at a time, as selected sets by user specified criteria (any charter characteristic(s)), all or none, by certain originating user(s), by certain originating MS(s), or any other desirable criteria. Of course, privileges are defined for enabling accepting privileges or charters from a MS, but many privileges can be defined for accepting privileges or charters with certain desired characteristics from a MS.

FIG. **44**A depicts a flowchart for describing a preferred embodiment of a procedure for sending MS data to another MS. FIG. **44**A processing is preferably of linkable PIP code **6**. The purpose is for the MS of FIG. **44**A processing (e.g. a first, or sending, MS) to transmit information to other MSs (e.g. at least a second, or receiving, MS), for example permissions **10** or charters **12**. Multiple channels for sending, or broadcasting should be isolated to modular send processing (feeding from a queue **24**). In an alternative embodiment having multiple transmission channels visible to processing of FIG. **44**A (e.g. block **4430**), there can be intelligence to drive each channel for broadcasting on multiple channels, either by multiple send threads for FIG. **44**A processing, FIG. **44**A loop processing on a channel list, and/or passing channel information to send processing feeding from queue **24**. If FIG. **44**A does not transmit directly over the channel(s) (i.e. relies on send processing feeding from queue **24**), an embodiment may provide means for communicating the channel for broadcast/send processing when interfacing to queue **24** (e.g. incorporate a channel qualifier field with send packet inserted to queue **24**).

In any case, see detailed explanations of FIGS. **13**A through **13**C, as well as supporting exemplifications shown in FIGS. **50**A through **50**C, respectively. Processing begins at block **4402**, continues to block **4404** where the caller parameter passed to FIG. **44**A processing is determined (i.e. OBJ_TYPE), and processing continues to block **4406** for interfacing with the user to specify targets to send data to, in context of the object type parameter specified for sending (PERMISSION_INFO or CHARTER_INFO). An alternate embodiment will consult a configuration of data for validated target information. Depending on the present disclosure embodiment, a user may specify any reasonable supported (ID/ID-Type) combination of the BNF grammar ID construct (see FIG. **30**B) as valid targets. Validation will validate at least syntax of the specification. In another embodiment, block **4406** will access and enforce known permissions for validating which target(s) (e.g. grantor(s)) can be specified. Various embodiments will also support wildcarding the specifications for a group of ID targets (e.g. department* for all department groups). Additional target information is to be specified when required for sending, for example, if email or SMS message is to be used as a send method (i.e. applicable destination recipient addresses to be specified). An alternate embodiment to block **4406** accesses mapped delivery addresses from a database, or table, (referred to as a Recipient Address Book (RAB)) associating a recipient address to a target identity, thereby alleviating the user from manual specification, and perhaps allowing the user to save to the RAB for any new useful RAB data. In another embodiment, block **4428** (discussed below) accesses the RAB for a recipient address for the target when preparing the data for sending.

Upon validation at block **4406**, processing continues to block **4408**. It is possible the user was unsuccessful in specifying targets, or wanted to exit block **4406** processing. If block **4408** determines the user did not specify at least one validated target (equivalent to selecting to exit FIG. **44**A processing), then processing continues to block **4444** where processing returns to the caller. If block **4408** determines there is at least one target specified, then block **4410** accesses LBX history **30** to determine if any of the targets have been sent the specific data already. Thereafter, if block **4412** determines the most recently updated data for a target has already been sent, then block **4414** presents an informative error to the user, preferably requiring user action. Block **4414** continues to block **4416** when the user performs the action. If block **4416** determines the user selected to ignore the error, then processing continues to block **4418**, otherwise processing continues back to block **4406** for updating target specifications.

Block **4418** interfaces with the user to specify a delivery method. Preferably, there are defaulted setting(s) based on the last time the user encountered block **4418**. Any of the "second set" of options described with FIG. **43** can be made. Thereafter, block **4420** logs to LBX history **30** the forthcoming send attempt and gets the next target from block **4406** specifications before continuing to block **4422**. If block **4422** determines that all targets have not been processed, then block **4424** determines applicable OBJ_TYPE data for the target (e.g. check LBX history **30** for any new data that was not previously successfully sent), and block **4426** gets (e.g. preferably new data, or all, depending on embodiment) the applicable target's OBJ_TYPE data (permissions or charters) before continuing to block **4428**. Block **4428** formats the data for sending in accordance with the specified delivery method, along with necessary packet information (e.g. source identity, wrapper data, etc) of this loop iteration (from block **4418**), and block **4430** sends the data appropriately. For a broadcast send, block **4430** broadcasts the information (using a send interface like interface **1906**) by inserting to queue **24** so that send processing broadcasts data **1302** (e.g. on all available communications interface(s) **70**), for example as far as radius **1306**, and processing continues to block **4432**. The broadcast is for reception by data processing systems (e.g. MSs) in the vicinity (see FIGS. **13**A through **13**C, as further explained in detail by FIGS. **50**A through **50**C which includes potentially any distance). For a targeted send, block **4430** formats the data intended for recognition by the receiving target. Block **4430** causes sending/broadcasting data **1302** containing CK **1304**, depending on the type of MS, wherein CK **1304** contains information appropriately. In a send email embodiment, confirmation of delivery status may be used to confirm delivery with an email interface API to check the COD (Confirmation of Delivery) status, or the sending of the email (also SMS message) is assumed to have been delivered in one preferred embodiment.

In an embodiment wherein usual MS communications data **1302** of the MS is altered to contain CK **1304** for listening MSs in the vicinity, send processing feeding from queue **24**, caused by block **4430** processing, will place information as CK **1304** embedded in usual data **1302** at the next opportune time of sending usual data **1302**. This embodiment will replace synchronous sending success validation of blocks **4432** through **4440** and multiple delivery methods of **4418** (and subsequent loop processing) with status asynchronously updated by the receiving MS(s) for a single type of delivery method selected at block **4418**. An alternate embodiment will attempt the multiple send types in an appropriate asynchronous thread of processing depending on success of a previous attempt. As the MS conducts its normal communications, transmitted data **1302** contains new data CK **1304** to be ignored by receiving MS other character **32** processing, but to be found by listening MSs within the vicinity which anticipate presence of CK **1304**. Otherwise, when LN-Expanse deployments have not introduced CK **1304** to usual data **1302** communicated on a receivable signal by MSs in the vicinity, FIG. **44**A sends/broadcasts new data **1302**.

For sending an email, SMS message, or other application delivery method, block **4430** will use the additional target information (recipient address) specified via block **4406** for properly sending. Thereafter, block **4432** waits for a synchronous acknowledgement if applicable before either receiving one or timing out. If a broadcast was made, one (1) acknowledgement may be all that is necessary for validation, or all anticipated targets can be accounted for before deeming a successful ack. An email, SMS message, or other application send may be assumed reliable and that an ack was received. Thereafter, if block **4434** determines an applicable ack was received (i.e. data successfully sent/received), or none was anticipated (i.e. assume got it), then processing continues back to block **4420** for processing any next target(s). If block **4434** determines an anticipated ack was not received, then block **4436** logs the situation to LBX history **30** and the next specified delivery method is accessed. Thereafter, if block **4438** determines all delivery methods have already been processed for the current target, then processing continues to block **4440** for logging the overall status and providing an error to the user. Block **4440** may require a user acknowledgement before continuing back to block **4420**. If block **4438** determines there is another specified delivery method for sending, then processing continues back to block **4428** for sending using the next method.

Referring back to block **4422**, if all targets are determined to have been processed, then block **4442** maintains FIG. **44**A processing results to LBX history **30** and the caller is returned to at block **4444**. In an alternate embodiment to FIG. **44**A processing, a trigger implementation is used for sending/broadcasting data at the best possible time (e.g. when new/modified permissions or charters information is made for a target) as soon as possible, as soon as a target is detected to be nearby, or in the vicinity (vicinity is expanded as explained by FIGS. **50**A through **50**C), or as soon as the user is notified to send (e.g. in response to a modification) and then acknowledges to send. See FIGS. **50**A through **50**C for explanation of communicating data from a first MS to a second MS over greater distances. In another embodiment, background thread(s) timely poll (e.g. per user or system configurations) the permissions and/or charters data to determine which data should be sent, how to send it, who to send it to, what applicable permissions are appropriate, and when the best time is to send it. A time interval, or schedule, for sending data to others on a continual interim basis may also be configured. This may be particularly useful as a user starts using a MS for the first time and anticipates making many configuration changes. The user may start or terminate polling threads as part of FIGS. **14**A/**14**B processing, so that FIG. **44**A is relied on to make sure permissions and/or charters are communicated as needed. Appropriate blocks of FIGS. **44**A&B will also interface to statistics **14** for reporting successes, failures and status of FIGS. **44**A&B processing.

In sum, FIGS. **44**A and **44**B provide a LBX peer to peer method for ensuring permissions and charters are appropriately maintained at MSs, wherein FIG. **44**A sends in a peer to peer fashion and FIG. **44**B receives in a peer to peer to fashion. Thus, permissions **10** and charters **12** are sent from a first MS to a second MS for configuring maintaining, enforcing, and/or processing permissions **10** and charters **12** at an MS. There is no intermediary service required for permissions and charters for LBX interoperability. FIG. **44**A demonstrates a preferred push model. A pull model may be alternatively implemented. An alternative embodiment may make a request to a MS for its permissions and/or charters and then populate its local image of the data after receiving the response. Privileges would be appropriately validated at the sending MS(s) and/or receiving MS(s) in order to ensure appropriate data is sent/received to/from the requesting MS.

FIG. **44**B depicts a flowchart for describing a preferred embodiment of receiving MS configuration data from another MS. FIG. **44**B processing describes a Receive Configuration Data (RxCD) process worker thread, and is of PIP code **6**. There may be many worker threads for the RxCD process, just as described for a 19xx process. The receive configuration data (RxCD) process is to fit identically into the framework of architecture **1900** as other 19xx processes, with specific similarity to process **1942** in that there is data received from receive queue **26**, the RxCD thread(s) stay blocked on the receive queue until data is received, and a RxCD worker thread sends data as described (e.g. using send queue **24**). Blocks **1220** through **1240**, blocks **1436** through **1456** (and applicable invocation of FIG. **18**), block **1516**, block **1536**, blocks **2804** through **2818**, FIG. **29**A, FIG. **29**B, and any other applicable architecture **1900** process/thread framework processing is to adapt for the new RxCD process. For example, the RxCD process is initialized as part of the enumerated set at blocks **1226** (preferably last member of set) and **2806** (preferably first member of set) for similar architecture **1900** processing. Receive processing identifies targeted/broadcasted data (permissions and/or charter data) destined for the MS of FIG. **44**B processing. An appropriate data format is used, for example the X.409 encoding of FIGS. **33**A through **33**C wherein RxCD thread(s) purpose is for the MS of FIG. **44**B processing to respond to incoming data. It is recommended that validity criteria set at block **1444** for RxCD-Max be set as high as possible (e.g. 10) relative performance considerations of architecture **1900**, to service multiple data receptions simultaneously. Multiple channels for receiving data fed to queue **26** are preferably isolated to modular receive processing.

In an alternative embodiment having multiple receiving transmission channels visible to the RxCD process, there can be a RxCD worker thread per channel to handle receiving on multiple channels simultaneously. If RxCD thread(s) do not receive directly from the channel, the preferred embodiment of FIG. **44**B would not need to convey channel information to RxCD thread(s) waiting on queue **24** anyway. Embodiments could allow specification/configuration of many RxCD thread(s) per channel.

A RxCD thread processing begins at block **4452** upon the MS receiving permission data and/or charter data, continues to block **4454** where the process worker thread count RxCD-Ct is accessed and incremented by 1 (using appropriate semaphore access (e.g. RxCD-Sem)), and continues to block **4456** for retrieving from queue **26** sent data (using interface like interface **1948**), perhaps a special termination request entry, and only continues to block **4458** when a record of data (permission/charter data, or termination record) is retrieved. In one embodiment, receive processing deposits X.409 encoding data as record(s) to queue **26**, and may break up a datastream into individual records of data from an overall received (or ongoing) datastream. In another embodiment, XML is received and deposited to queue **26**, or some other suitable syntax is received as derived from the BNF grammar. In another embodiment, receive processing receives data in one format and deposits a more suitable format for FIG. **44**B processing. Receive processing embodiments may deposit "piece-meal" records of data as sent, "piece-meal" records broken up from data received, full charter or permission datastreams and/or subsets thereof to queue **26** for processing by FIG. **44**B.

Block **4456** stays blocked on retrieving from queue **26** until any record is retrieved, in which case processing continues to block **4458**. If block **4458** determines a special entry indicating to terminate was not found in queue **26**, processing continues to block **4460**. There are various embodiments for RxCD thread(s), thread(s) **1912** and thread(s) **1942** to feed off a queue **26** for different record types, for example, separate queues **26**A, **26**B and **26**C, or a thread target field with different record types found at queue **26** (e.g. like field **2400***a*). In another embodiment, there are separate queues **26**C and **26**D for separate processing of incoming charter and permission data. In another embodiment, thread(s) **1912** are modified with logic of RxCD thread(s) to handle permission and/or charter data records, since thread(s) **1912** are listening for queue **26** data anyway. In another embodiment, there are segregated RxCD threads RxCD-P and RxCD-C for separate permission and charter data processing.

Block **4460** validates incoming data for this targeted MS before continuing to block **4462**. A preferred embodiment of receive processing already validated the data is intended for this MS by having listened specifically for the data, or by having already validated it is at the intended MS destination (e.g. block **4458** can continue directly to block **4464** (no block **4460** and block **4462** required)). If block **4462** determines the data is valid for processing, then block **4464** accesses the data source identity information (e.g. owner information, sending MS information, grantor/grantee information, etc, as appropriate for an embodiment), block **4466** accesses acceptable delivery methods and/or permissions/privileges for the source identity to check if the data is eligible for being received, and block **4468** checks the result. Depending on an embodiment, block **4466** may enforce an all or none privilege for accepting the privilege or charter data, or may enforce specific privileges from the receiving MS (MS user) to the sending MS (MS user) for exactly which privileges or charters are acceptable to be received and locally maintained.

If block **4468** determines the delivery is acceptable (and perhaps privileged, or privileged per source), then block **4470** appropriately updates the MS locally with the data (depending on embodiment of **4466**, block **4470** may remove from existing data at the MS as well as per privilege(s)), block **4472** completes an acknowledgment, and block **4474** sends/broadcasts the acknowledgement (ack), before continuing back to block **4456** for more data. Block **4474** sends/broadcasts the ack (using a send interface like interface **1946**) by inserting to queue **24** so that send processing transmits data **1302**, for example as far as radius **1306**. Embodiments will use the different correlation methods already discussed above, to associate an ack with a send.

If block **4468** determines the data is not acceptable, then processing continues directly back to block **4456**. For security reasons, it is best not to respond with an error. It is best to ignore the data entirely. In another embodiment, an error may be returned to the sender for appropriate error processing and reporting. Referring back to block **4462**, if it is determined that the data is not valid, then processing continues back to block **4456**.

Referring back to block **4458**, if a worker thread termination request was found at queue **26**, then block **4476** decrements the RxCD worker thread count by 1 (using appropriate semaphore access (e.g. RxCD-Sem)), and RxCD thread processing terminates at block **4478**. Block **4476** may also check the RxCD-Ct value, and signal the RxCD process parent thread that all worker threads are terminated when RxCD-Ct equals zero (0).

Block **4474** causes sending/broadcasting data **1302** containing CK **1304**, depending on the type of MS, wherein CK **1304** contains ack information prepared. In the embodiment wherein usual MS communications data **1302** of the MS is altered to contain CK **1304** for listening MSs in the vicinity, send processing feeding from queue **24**, caused by block **4474** processing, will place ack information as CK **1304** embedded in usual data **1302** at the next opportune time of sending usual data **1302**. As the MS conducts its normal communications, transmitted data **1302** contains new data CK **1304** to be ignored by receiving MS other character **32** processing, but to be found by listening MSs within the vicinity which anticipate presence of CK **1304**. Otherwise, when LN-Expanse deployments have not introduced CK **1304** to usual data **1302** communicated on a receivable signal by MSs in the vicinity, FIG. **44**B sends/broadcasts new ack data **1302**.

In an alternate embodiment, permission and/or charter data records contain a sent date/time stamp field of when the data was sent by a remote MS, and a received date/time stamp field (like field **2490***c*) is processed at the MS in FIG. **44**B processing. This would enable calculating a TDOA measurement while receiving data (e.g. permissions and/or charter data) that can then be used for location determination processing as described above.

For other acceptable receive processing, methods are well known to those skilled in the art for "hooking" customized processing into application processing of sought data received. For example, in an email application, a callback function API is preferably made available to the present disclosure so that every time an applicable received email distribution is received with specified criteria (e.g. certain subject, certain attached file name, certain source, or any other identifiable email attribute(s) (provided by present disclosure processing to API) sent by block **4430**, the callback function (provided by present disclosure processing to the appropriate API) is invoked for custom processing. In this example, the present disclosure invokes the callback API for providing: the callback function to be invoked, and the email criteria for triggering invocation of the callback function; for processing of permissions or charter data. For example, a unique subject field indicates to the email application that the email item should be directed by the email application to the callback function for processing. The present disclosure callback function then parses permissions and/or charter information from the email item and updates local permissions **10** and/or charters **12**. Data received in the email item may be textual syntax derived from the BNF grammar in an email body or attached file form, XML syntax derived from the BNF grammar in email body or attached file form, an X.409 binary encoding in attached file form, or other appropriate format received with the email item (e.g. new Document Interchange Architecture (DIA) attribute data, etc). A process return status is preferably returned by the callback function, for example for appropriate email confirmation of delivery processing.

In another embodiment, the present disclosure provides at least one thread of processing for polling a known API, or email repository, for sought criteria (e.g. attributes) which identifies the email item as destined for present disclosure processing. Once the email item(s) are found, they are similarly parsed and processed for updating permissions **10** and/or charters **12**.

Thus, there are well known methods for processing data in context of this disclosure for receiving permissions **10** and/or charters **12** from an originating MS to a receiving MS, for example when using email. Similarly (callback function or polling), SMS messages can be used to communicate data **10** and/or **12** from one MS to another MS, albeit at smaller data exchange sizes. The sending MS may break up larger portions of data which can be sent as parse-able text (e.g. source syntax, XML, etc. derived from the BNF grammar) to the receiving MS. It may take multiple SMS messages to communicate the data in its entirety.

Regardless of the type of receiving application, those skilled in the art recognize many clever methods for receiving data in context of a MS application which communicates in a peer to peer fashion with another MS (e.g. callback function(s), API interfaces in an appropriate loop which can remain blocked until sought data is received for processing, polling known storage destinations of data received, or other applicable processing).

Permission data **10** and charter data **12** may be manually copied from one MS to another over any appropriate communications connection between the MSs. Permission data **10** and charter data **12** may also be manually copied from one MS to another MS using available file management system operations (move or copy file/data processing). For example, a special directory can be defined which upon deposit of a file to it, processing parses it, validates it, and uses it to update permissions **10** and/or charters **12**. Errors found may also be reported to the user, but preferably there are automated processes that create/maintain the file data to prevent errors in processing. Any of a variety of communications wave forms can be used depending on MS capability.

FIG. **45** depicts a flowchart for describing a preferred embodiment of MS charters configuration processing of block **1482**. FIG. **45** is of Self Management Processing code **18**. Processing starts at block **4502** and continues to block **4504** where a list of charters configuration options are presented to the user. Thereafter, block **4506** waits for a user action in response to options presented. Block **4506** continues to block **4508** when a user action has been detected. If block **4508** determines the user selected to configure charters data, then the user configures charters data at block **4510** (see FIG. **46**A) and processing continues back to block **4504**. If block **4508** determines the user did not select to configure charters data, then processing continues to block **4512**. If block **4512** determines the user selected to configure actions data, then the user configures actions data at block **4514** (see FIG. **47**A) and processing continues back to block **4504**. If block **4512** determines the user did not select to configure actions data, then processing continues to block **4516**. If block **4516** determines the user selected to configure parameter data, then the user configures parameter data at block **4518** (see FIG. **48**A) and processing continues back to block **4504**. If block **4516** determines the user did not select to configure parameter data, then processing continues to block **4520**. If block **4520** determines the user selected to view other's charter data, then block **4522** invokes the view other's info processing of FIG. **42** with CHARTER_INFO as a parameter (for viewing other's charter data) and processing continues back to block **4504**. If block **4520** determines the user did not select to view other's charter data, then processing continues to block **4524**. If block **4524** determines the user selected to view other's actions data, then block **4526** invokes the view other's info processing of FIG. **42** with ACTION_INFO as a parameter (for viewing other's action data) and processing continues back to block **4504**. If block **4524** determines the user did not select to view other's action data, then processing continues to block **4528**. If block **4528** determines the user selected to view other's parameter data, then block **4530** invokes the view other's info processing of FIG. **42** with PARAMETER_INFO as a parameter (for viewing other's parameter data information) and processing continues back to block **4504**. If block **4528** determines the user did not select to view other's parameter data, then processing continues to block **4532**. If block **4532** determines the user selected to send charters data, then block **4534** invokes the send data processing of FIG. **44**A with CHARTER_INFO as a parameter (for sending charters data) and processing continues back to block **4504**. If block **4532** determines the user did not select to send charters data, then processing continues to block **4536**. If block **4536** determines the user selected to configure accepting charters, then block **4538** invokes the configure acceptance processing of FIG. **43** with CHARTER_INFO as a parameter (for configuring acceptance of charters data) and processing continues back to block **4504**. If block **4536** determines the user did not select to configure accepting charters, then processing continues to block **4540**. If block **4540** determines the user selected to exit block **1482** processing, then block **4542** completes block **1482** processing. If block **4540** determines the user did not select to exit, then processing continues to block **4544** where all other user actions detected at block **4506** are appropriately handled, and processing continues back to block **4504**.

In an alternate embodiment where the MS maintains GDRs, GADRs, CDRs, ADRS, PARMDRs and GRPDRs (and their associated data records DDRs, HDRs and TDRs) at the MS where they were configured, FIG. **45** may not provide blocks **4520** through **4530**. The MS may be aware of its user charters and need not share the data (i.e. self contained). In some embodiments, options **4520** through **4530** cause access to locally maintained data for others (other users, MSs, etc) or cause remote access to data when needed (e.g. from the remote MSs). In the embodiment where no data is maintained locally for others, blocks **4532** through **4538** may not be necessary. In sum, the preferred embodiment is to locally maintain charters data for the MS user and others (e.g. MS users) which are relevant to provide the richest set of charters governing MS processing at the MS.

FIGS. **46**A through **46**B depict flowcharts for describing a preferred embodiment of MS user interface processing for charters configuration of block **4510**. With reference now to FIG. **46**A, processing starts at block **4602**, continues to block **4604** for initialization (e.g. a start using database command), and then to block **4606** where groups the user is a member of are accessed. Block **4606** retrieves all GRPDRs **3540** joined to GADRs **3520** such that the descendant type field **3520***c* and descendant ID field **3520***d* match the user information, and the ascendant type field **3520***a* is set to Group and the ascendant ID field **3520***b* matches the group ID field **3540***a*. While there may be different types of groups as defined for the BNF grammar, the GRPDR is a derivative embodiment which happens to not distinguish. Alternate embodiments may carry a group type field to select appropriate records by group type. Yet another embodiment may not have a block **4606** with processing at block **4608** for gathering data additionally by groups the user is a member of. Block **4606** continues to block **4608**.

Block **4608** accesses all CDRs (e.g. all rows from a CDR SQL table) for the user of FIG. **46**A (e.g. user information matches field **3700***b*), and for the groups the user is a member of (e.g. group information matches field **3700***b* (e.g. owner type=group, owner id=a group ID field **3540***a* from block **4606**)). The CDRs are additionally joined (e.g. SQL join) with GDRs, DDRs and TDRs (e.g. fields **3500***t*, **3600***b* and **3640***b*=Charter and by matching ID fields **3500***a*, **3600***a* and **3640***a* with field **3700***a*). Description field **3600** can provide a useful description last saved by the user for the charter entry. Block **4608** may also retrieve system predefined data records for use and/or management. Thereafter, each joined entry returned at block **4608** is associated at block **4610** with the corresponding data IDs (at least fields **3700***a*/**3500***a* and **3540***a*) for easy unique record accesses when the user acts on the data. Block **4610** also initializes a list cursor to point to the first list entry to be presented to the user. Thereafter, block **4612** sets user interface indication for where the list cursor is currently set (e.g. set to highlight the entry), and any list scrolling settings are set (the list is initially not set for being scrolled on first FIG. **46**A processing encounter to block **4612** from block **4610**). Block **4612** continues to block **4614** where the entry list is presented to the user in accordance with the list cursor and list scroll settings managed for presentation at block **4612**. Thereafter, block **4616** waits for user action to the presented list of charters data and will continue to block **4618** when a user action has been detected. Presentation of the scrollable list preferably presents in an entry format such that an entry contains fields for: DDR **3600** description; GDR owner information, grantor information and grantee information; GRPDR owner information and group name if applicable; CDR information; and TDR time spec information. Alternate embodiments will present less information, or more information (e.g. join to ADR and/or PARMDR information).

If block **4618** determines the user selected to set the list cursor to a different entry, then block **4620** sets the list cursor accordingly and processing continues back to block **4612**. Block **4612** always sets for indicating where the list cursor is currently pointed and sets for appropriately scrolling the list if necessary when subsequently presenting the list at block **4614**. If block **4618** determines the user did not select to set the list cursor, then processing continues to block **4622**. If block **4622** determines the user selected to add a charter, then block **4624** accesses a maximum number of charters allowed (perhaps multiple maximum values accessed), and block **4626** checks the maximum(s) with the number of current charters defined. There are many embodiments for what deems a maximum (for this user, for a group, for this MS, etc). If block **4626** determines a maximum number of charters allowed already exists, then block **4628** provides an error to the user and processing continues back to block **4612**. Block **4628** preferably requires the user to acknowledge the error before continuing back to block **4612**. If block **4626** determines a maximum was not exceeded, then block **4630** interfaces with the user for entering validated charter data and block **4632** adds the data record(s), appropriately updates the list with the new entry, and sets the list cursor appropriately for the next list presentation refresh, before continuing back to block **4612**. If block **4622** determines the user did not want to add a charter, processing continues to block **4634**. Block **4632** will add a CDR, GDR, DDR, HDR (to set creator information) and TDR. The DDR and TDR are optionally added by the user, but the DDR may be strongly suggested (if not enforced on the add). This will provide a charter record. Additionally, block **4630** may add new ADR(s) and/or PARMDR(s) (which are validated to exist prior to adding data at block **4632**). In one embodiment, a GDR associated to the CDR is not added; for indicating the user wants his charter made available to all other user MSs which are willing to accept it.

If block **4634** determines the user selected to delete a charter, then block **4636** deletes the data record currently pointed to by the list cursor, modifies the list for the discarded entry, and sets the list cursor appropriately for the next list presentation refresh, before continuing back to block **4612**. Block **4636** will use the Charter ID field **3700***a*/**3500***a* (associated with the entry at block **4610**) to delete the charter. Associated CDR, ADR(s), PARMDR(s), DDR **3600**, HDR **3620**, and TDR **3640** is also deleted (e.g. preferably with a cascade delete in a SQL embodiment). If block **4634** determines the user did not select to delete a charter, then processing continues to block **4652** of FIG. **46**B by way of off-page connector **4650**.

With reference now to FIG. **46**B, if block **4652** determines the user selected to modify a charter, then block **4654** interfaces with the user to modify charter data of the entry pointed to by the list cursor. The user may change information of the GDR, CDR, ADR and/or PARMDR and any associated records (e.g. DDR and TDR). The user may also add applicable records at block **4654**. Block **4654** waits for a user action indicating completion. Block **4654** will continue to block **4656** when the complete action is detected. If block **4656** determines the user exited, then processing continues back to block **4612** by way of off-page connector **4698**. If block **4656** determines the user selected to save changes made at block **4654**, then block **4658** updates the data and the list is appropriately updated before continuing back to block **4612**. Block **4658** may update the GDR, CDR, ADR, PARMDR and/or any associated records (e.g. DDR, and/or TDR) using the charter id field **3700***a*/**3500***a* (associated to the entry at block **4610**). Block **4658** will update an associated HDR as well. Block **4658** may add new CDR, ADR(s), PARMDR(s), a DDR and/or TDR as part of the charter change. If block **4652** determines the user did not select to modify a charter, then processing continues to block **4660**.

If block **4660** determines the user selected to get more details of the charter (e.g. show all joinable data to the GDR or CDR that is not already presented with the entry), then block **4662** gets additional details (may involve database queries in an SQL embodiment) for the charter pointed to by the list cursor, and block **4664** appropriately presents the information to the user. Block **4664** then waits for a user action that the user is complete reviewing details, in which case processing continues back to block **4612**. If block **4660** determines the user did not select to get more detail, then processing continues to block **4666**.

If block **4666** determines the user selected to internalize charters data thus far being maintained, then block **4668** internalizes (e.g. as a compiler would) all applicable data records for well performing use by the MS, and block **4670** saves the internalized form, for example to MS high speed non-persistent memory. In one embodiment, blocks **4668** and **4670** internalize charter data to applicable C structures of FIGS. **34**A through **34**G (also see FIG. **52**). In various embodiments, block **4668** maintains statistics for exactly what was internalized, and updates any running totals or averages maintained for a plurality of internalizations up to this point, or over certain time periods. Statistics such as: number of active constructs; number of user construct edits of particular types; amount of associated storage used, freed, changed, etc with perhaps a graphical user interface to graph changes over time; number of charter expressions, actions, term types, etc specified, number of charters affected and unaffected by permissions; and other charter dependent statistics. In other embodiments, statistical data is initialized at internalization time to prepare for subsequent gathering of useful statistics during charter processing. In embodiments where a tense qualifier is specified for TimeSpec information, saving the internalized form at block **4670** causes all past and current tense configurations to become effective for being processed.

Block **4670** then continues back to block **4612**. If block **4666** determines the user did not select to internalize charter configurations, then processing continues to block **4672**. Alternate embodiments of processing charters **12** in the present disclosure will rely upon the data records entirely, rather than requiring the user to redundantly internalize from persistent storage to non-persistent storage for use. Persistent storage may be of reasonably fast performance to not require an internalized version of charters **12**. Different embodiments may completely overwrite the internalized form, or update the current internalized form with any changes.