

EXHIBIT 3

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

U.S. PATENT NO.: 9,467,838

ART UNIT: 3992

CONTROL
NUMBER: 90/014,510

CONF. NO.: 7707

FILING DATE: May 15, 2020

EXAMINER: Tarae, Catherine
Michelle

TITLE: **METHOD TO PROVIDE AD HOC AND PASSWORD
PROTECTED DIGITAL AND VOICE NETWORKS**

FILED ELECTRONICALLY

Mail Stop *Ex Parte* Reexam
ATTN: Central Reexamination Unit
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

**DECLARATION OF MALCOLM K. BEYER JR. IN SUPPORT OF
REPLY TO OFFICE ACTION**

I, Malcolm K. Beyer, Jr., being duly sworn, hereby state as follows:

1. I am the Chief Executive Officer (“CEO”) of Plaintiff AGIS Software Development LLC (“AGIS”). I am also the first-named inventor on U.S. Patent No. 9,467,838 (the “’838 patent”). I submit this declaration based on my personal knowledge and in support of AGIS’s response to the non-final office action issued by the Office in its reexamination of the ’838 patent (Control No. 90/014,510).

Background

2. I graduated from the U.S. Naval Academy in 1962 and was commissioned as a Second Lieutenant in the U.S. Marine Corps. I later attended the U.S. Navy’s programming school and was the lead programmer for the first automated Marine Corps Tactical Operations Center Link-11 Navy Interface.

3. After leaving active service, I worked at a number of well-known technology companies, including System Development Corporation (considered the world's first computer software company) and Litton Industries. I then started several businesses which provided technology and engineering solutions supporting defense and military customers.

4. In 1987, I co-founded Advanced Programming Concepts, Inc. ("APC"), a Texas corporation based in Austin, Texas. APC operated primarily out of its main business location in Austin, Texas. APC specialized in designing, building, and supporting systems for enabling integration and sharing time-critical information across dissimilar military and defense applications. At APC, I was the majority shareholder and Chairman until we sold the business to Ultra Electronics in July 1999.

5. On June 30, 2004, I founded Advanced Ground Information Systems, Inc. ("AGIS Inc."). AGIS Inc.'s main business location is located at 92 Lighthouse Drive, Jupiter, Florida. AGIS Inc. also maintains business locations in Austin, TX and Kansas City, KS. AGIS Inc. employs approximately 15 people. In 2017, the Board of AGIS, Inc. approved a reorganization plan which organized AGIS and AGIS, Inc. under its current parent, AGIS Holdings, Inc. AGIS, Inc. continues to sell the AGIS LifeRing solutions while AGIS maintains a data center located at 1005 Stuart Lane, Marshall, Texas 75672. AGIS's Texas data center hosts servers, code, applications, and services necessary to run operations for AGIS Inc.'s LifeRing products and solutions and is used for research and development projects.

6. As the CEO of AGIS and AGIS Inc. and first-named inventor of the '838 patent, I have knowledge of all facets of the businesses, including the conception and reduction to practice of the '838 patent and the diligence involved in reducing to practice the claimed inventions.

7. Since 2004, AGIS Inc.'s primary business has revolved around offering the "LifeRing" products and solutions which include client-based applications and a server-based solutions for, generally, enabling smartphone, tablet, and PC users to easily and rapidly establish secure ad hoc digital networks. LifeRing 5.0 and its predecessor versions have been offered and sold to military, defense, and first-responder customers, as well as private industry customers.

8. AGIS LifeRing software was developed by United States military veterans as a response to the September 11th, 2001 terrorist bombings. Since that time, the company has continued its development efforts; leveraging the technological advances in modern PCs and Smartphones as well as communications methods. AGIS's LifeRing software enables PC, iPhone, and Android Tablet and Smartphone users to easily establish ad hoc COP networks where many people need to coordinate and collaborate with many others.

9. AGIS LifeRing provides users with the present location and status of others. LifeRing software uses GPS to provide location data. Users can seamlessly enter geo-locations of events or objects by selecting the desired map location and then the appropriate symbol. Information about other networked users and symbols can be obtained by touching symbols on the map.

10. AGIS LifeRing provides an interactive map display. Because accuracy is necessary to ensure the success of life and death missions, AGIS's maps are geo-referenced so that users and events are displayed at their correct locations in real-time. To interact and communicate with other users in a one or more groups, users simply touch the display at the desired map location and then select the appropriate or desired symbol, which appears at the correct map location, to initiate an exchange. LifeRing supports 80 different map types.

11. Over the years, LifeRing has been successfully deployed and tested in operational environments, including:

- New York Emergency Operations Center Test
- National Incident Management System (NIMS) Test
- Coalition Warrior Interoperability Demonstration (CWID)
- Army Network Integration Evaluation (NIE 12.1, 12.2 and 13.1)
- Numerous US Joint Chiefs of Staff Exercises
- The Defense Intelligence Agency (LifeRing won 4 Stars in the DIA's PLUGFEST)
- SOCOM TNT Exercises (2012 & 2013)
- US NATO Bold Quest (2012, 2013)
- Joint-Interagency Field Experimentation (JIFX) Exercises
- Army Expeditionary Warrior Experiment (AEWE 2013)
- Jolted Tactics
- US Navy
- International Partners (e.g., Australian, Dutch Defense Forces)

12. I am the first-named inventor of the '838 patent and its claimed inventions. I am very familiar with the claimed inventions of the '838 patent as I am the main point of contact for patent prosecution matters at AGIS (and formerly at AGIS, Inc.) and I personally reviewed and approved of the issued claims 1-84 of the '838 patent.

13. In addition to me, Mr. Christopher R. Rice is identified as a co-inventor on the '838 patent. With respect to the '838 patent, Mr. Rice's contributions included the client-server

communications. For example, Mr. Rice contributed to the portion of the claimed invention related to the following recited limitations: “sending, to a second server, a request for second georeferenced map data different from the first georeferenced map data” and “receiving, from the second server, the second georeferenced map data.”

14. In this reexamination proceeding, I understand that the Office has issued several rejections based on U.S. Patent No. 7,353,034 (“Haney”). I understand that Haney’s application filing date is April 4, 2005. However, I invented the claimed inventions prior to April 4, 2005.

15. The claimed inventions of the ’838 patent were conceived of by at least January 19, 2005 and were reduced to practice by October 22, 2005. The October 22, 2005 reduction to practice was the product of diligent work on the part of a team of engineers and programmers under my direction and supervision. Below, I identify and submit substantial evidence regarding the conception and reduction to practice of the claimed inventions of the ’838 patent.

16. The claimed inventions recited in the ’838 patent were conceived of by at least January 19, 2005. In August 2004, AGIS and its customer Raytheon engaged in discussions to develop a homeland security solution based on AGIS’s LifeRing solution. In Exhibit 1 to this declaration, I provide a copy of an email summarizing an outline of the planned phases of development. **Exhibit 1.** After these August 2004 discussions, AGIS submitted a proposal for a system of PDA/cell phone devices for use in a homeland defense system, and AGIS reduced its proposal to a statement of work (SOW) on or around August 29, 2004. The August 29, 2004 SOW outlined the system using PDA cell phone devices for transmitting and displaying tracked participants on maps, i.e., “correctly geographically superimposed on a map” and for interacting with the map to exchange data. **Exhibit 2.**

17. On January 19, 2005 I prepared a presentation for Raytheon attached hereto as **Exhibit 3**, and this presentation was presented to Raytheon personnel, including Kurt Winckler, on January 21, 2005. The January 19, 2005 presentation attached as **Exhibit 3** is evidence that I and Mr. Rice had conceived of the claimed inventions as early as January 19, 2005.

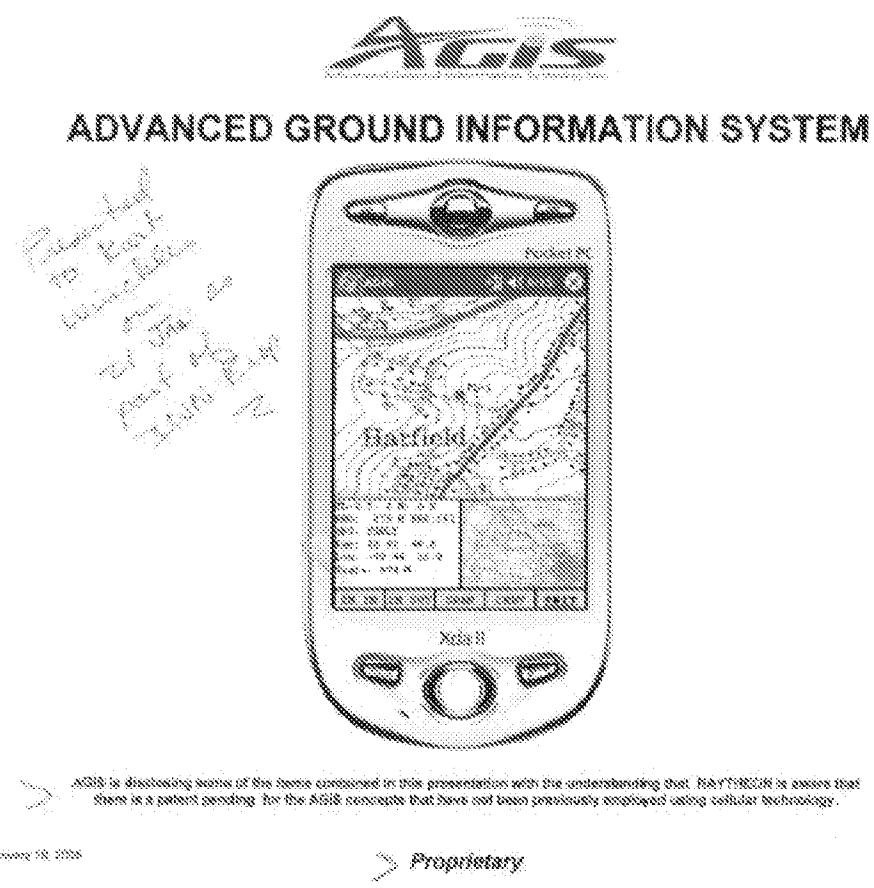


Exhibit 3 at 1.

18. The January 21, 2005 presentation describes cell phone/PDA devices programmed with software to permit users to display different maps and overlay maps, zoom and offset maps, enter other entities (track symbols), assign information associated with entities, obtain location and information of other AGIS units and tracking them, displaying tracking histories, and selecting different maps (e.g., fixed versus moving).

AGIS OPERATION

Display

The AGIS operator can:

- Display different maps and overlay maps
- Zoom and Offset maps
- Enter other entities (track symbols) such as accidents, fires, trucks, military units, etc., and assign information associated with them
- Obtain range and bearing readings to other locations
- Obtain location and information of other AGIS units and the tracks that they have entered
- Display the track histories
- Select a fixed map (his symbol moves across the map)
- Select a moving map (his symbol stays at the center of the display and the map moves underneath)
- Specify that the map and the symbols occupy most of the display

AGIS 2005

Proprietary

3

Exhibit 3 at 3.

19. The January 21, 2005 presentation describes that the AGIS software provides each user with the following features: display user's phone's and other users' phones' location, status, heading, speed, altitude, phone signal strength, and GPS status; displaying phones' locations, identities, and status information on different maps; establishing voice and data communications with other participants by interacting with the symbols; and transmitting messages and multimedia between participants.

AGIS OPERATION

Display

The AGIS operator can:

- Display different maps and overlay maps
- Zoom and Offset maps
- Enter other entities (track symbols) such as accidents, fires, trucks, military units, etc., and assign information associated with them
- Obtain range and bearing readings to other locations
- Obtain location and information of other AGIS units and the tracks that they have entered
- Display the track histories
- Select a fixed map (his symbol moves across the map)
- Select a moving map (his symbol stays at the center of the display and the map moves underneath)
- Specify that the map and the symbols occupy most of the display

January 18, 2005

Proprietary

3

Exhibit 3 at 3.

AGIS SOFTWARE

PROVIDES EACH USER:

- Display of his and other phones' location, status, heading, speed, altitude, phone signal strength and GPS status (on, off, 2D or 3D).
- His and other AGIS participants' location, identity and status information superimposed on a variety of maps, aerial photographs, and satellite images.
- Multiple, easily accessible, layered *SoftSwitches* organized by category to enable the AGIS operator to quickly operate the system.
- The ability quickly and easily to establish voice and data communications (*point to talk*) with any other AGIS net participant without losing the tactical picture by simply hooking other site's symbols.
- The ability quickly to transmit free text, fixed formatted messages, photographs and video clips between each user.

January 03, 2005

Proprietary

8

Exhibit 3 at 5.

AGIS SOFTWARE
Continued

AGIS FURTHER PROVIDES EACH USER:

- The ability to enter friendly and hostile forces (or other information such as fires, emergencies, accidents, pilot boats, fire boats, etc.)* locations on a map and to associate data with their symbol. This information is then transmitted along with the AGIS participants location on the AGIS cellular net to all other AGIS equipped users.
- The ability to receive and transmit free text, fixed formatted, photograph**, and video clip** messages by pointing at the AGIS user to whom they are to be sent.
- The ability to establish range and bearing data from the user to any point.
- The ability for each AGIS user to view his own and other tracks' history trail and last heading data.
- The ability to also operate from a PC and a Tablet*.

*Modification Part of the proposed contract. **Capability will be available in Spring of 2006

January 18, 2005

Proprietary

4

Exhibit 3 at 6.

20. The January 21, 2005 presentation describes the formation and use of groups or “nets” of participants.

AGIS OPERATION

The operator controls the AGIS with either a Stylus or using Finger On Glass

The operator interacts with the AGIS through the use of layered *SoftSwitches* which are located at the bottom of the display.

When initializing the system, the AGIS operator specifies from a list those AGIS users with whom he desires to net location data and with whom he desires to be able to conduct rapid voice and data communications.

When the AGIS is prompted to respond with its location, it sends its location and status to all that are part of the net.

MiStd 2525 symbols (or Homeland Defense defined symbols*) are associated with each user and appear at the correct location on each of the displays.

The AGIS operator obtains information concerning other symbols appearing on the display by touching the screen at their location thereby "hooking" them. Data concerning the other symbol then appears on the lower part of the AGIS display.

* Part of the proposed contract

January 19, 2000

Proprietary

8

Exhibit 3 at 8.

AGIS OPERATION

Cellular phone calls

- When the AGIS operator decides to call another AGIS user, he simply hooks the other user's symbol and selects the Call SoftSwitch. The phone of the AGIS receiving call then hears a ring and sees a box appear around the sender's symbol.
- The AGIS operator can setup pre-established nets of up to six AGIS net participants. When he desires to talk to that group of users he selects the appropriate net number and a conference call is automatically made.
- The AGIS operator can conference an almost unlimited number of AGIS participants by using the AGIS's 800 conferencing capability. Again the AGIS operator assigns AGIS participants to a net number. When he desires to make the conference call, he selects the net number. This action causes a message to be sent to all of the AGIS participants, causing their phones to automatically call an 800 number and automatically entering their participant code.

January 19, 2005

Proprietary

8

Exhibit 3 at 9.

21. AGIS's LifeRing solution was under development throughout 2004 and 2005. During this period, I conceived and reduced to practice both non-server and server-based implementations. As I discussed above, Mr. Rice's contributions included the client-server communications, including "sending, to a second server, a request for second georeferenced map data different from the first georeferenced map data" and "receiving, from the second server, the second georeferenced map data." In the non-server implementations, AGIS's maps were uploaded to and called from databases and the transmissions occurred over SMS only. However, in the server-based implementations, which were conceived of by January 19, 2005, the devices communicated with intermediary servers and the maps requested and received from servers. The

January 21, 2005 presentation describes an AGIS server implementation and the development and testing of preproduction server-based implementations were underway.

AGIS SERVER

- When AGIS is operating in a SMS (slow speed) mode of operation, communications are from the AGIS phone to the telephone company and then from the phone company to all other AGIS phones.
- When the AGIS is operating in a high speed GRPS / CDMA2000 1xEVDO mode, communications are from the AGIS phone to the telephone phone company, then to the TCP/IP Server and from the TCP/IP Server to all the other AGIS phones.
- The TCP/IP Server pushes the received data to the AGIS phones so that information is received within approximately 5 seconds.
- The TCP/IP Server provides interoperability between the GRPS / CDMA2000 1xEV-DO communications.
- The AGIS accounts for when the phones go into Voice mode and then transmits data in SMS until high speed communications are available.
- The TCP/IP Server is operating on a 24 hour basis. We are still in the process of resolving Server Trouble Reports.

January 18, 2005

Proprietary

18

Exhibit 3 at 14.

AGIS STATUS

THE PROTOTYPE AGIS HAS UNDERGONE A WEEK OF TESTING AT THE NAVAL RESEARCH LABORATORY'S (NRL) ANTITERRORISM GROUP. DURING THE TEST PERIOD, THE AGIS OPERATED WITHOUT FAILURE. THE RESULTS OF THIS TESTING WAS POSITIVE. IT WAS REQUESTED THAT:

AGIS FUNCTION SWITCHES BE MADE EASIER TO USE AND REMEMBER*

MULTIPLE MAP SETS CAPABILITY BE PROVIDED*

DIRECTION UP MAPS AND MULTIPLE LOCATION MAPS BE PROVIDED**

WE ARE UNDER CONTRACT TO ADD THE OTHER FEATURES (OTHER THAN SECURITY) DISCUSSED IN THIS PRESENTATION.

THE PREPRODUCTION AGIS IS DUE TO BE TESTED AT NRL IN FEB 2005.

ALL THE FEATURES (EXCEPT SECURITY) WILL BE INCORPORATED INTO THE SEATTLE DELIVERY

* Completed, ** Will be available in January or early February

January 18, 2005

Proprietary

15

Exhibit 3 at 15.

22. This is consistent with AGIS's plans to provide "over the air" georeferenced maps via servers. **Exhibit 4 at 8-9.**

and Figure 19 depicts selection of coastlines and geopolitical boundaries. As part of a

Proprietary

Advanced Ground Information Systems, Inc. Topic Number N05-069 Proposal Number N051-069-9
0725 Proprietary

separate non government funded effort, AGIS plans to incorporate "over the air" downloading of georeferenced maps.

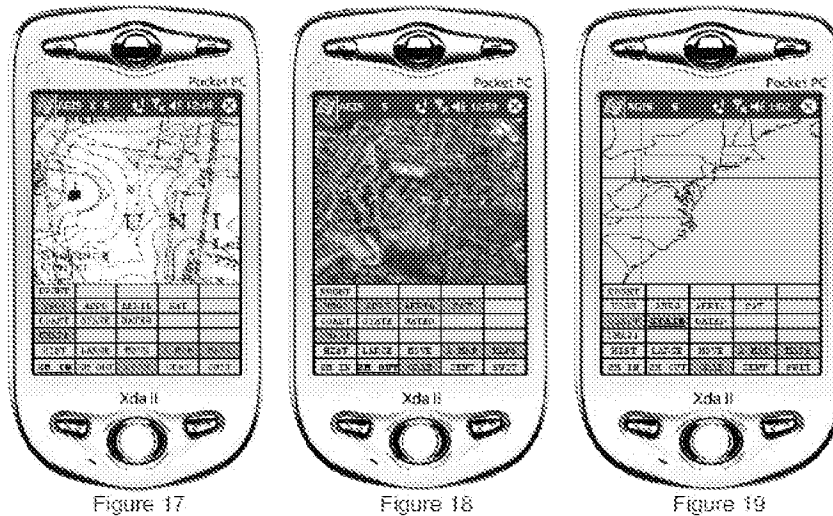


Figure 17

Figure 18

Figure 19

Exhibit 4 at 8-9.

23. Under my direction and supervision, AGIS was diligent in reducing to practice the claimed inventions of the '838 patent. Immediately after conception, I assembled a team to begin development and testing of various implementations. The AGIS project timeline consisted of multiple phases, and the project timeline was updated continuously over the course of development. Exhibits 5-8 describe multiple timeline updates based on changes to the server-based implementation. From January 2005 to October 2005, AGIS dedicated hundreds of hours to the development and testing of the server-based implementations of the AGIS LifeRing solution. The development team included myself, Christopher R. Rice, Sandel Blackwell,

Dennis Hoff, Scott Brown, and Jason Cardamone. **Exhibits 9-14** include timesheets from the individuals involved in the development and evaluation of the AGIS LifeRing solution during the period of April 2005 through October 2005. These timesheets demonstrate that I was diligent in reducing to practice the January 19, 2005 concepts for the server-based implementation of the AGIS LifeRing solution. Please note that these timesheets indicate billable hours only, which reflects a subset of the actual time dedicated to the research and development and testing of the LifeRing solution.

24. In April 2005, Mr. Rice dedicated at least 39 hours to fixing the SMS transmission and TCP protocol portions of LifeRing. In April 2005, Mr. Blackwell billed 89 hours for work including, for example, updating the change rate of the displayed information, grouping and selecting participant groups, and user location updates. Mr. Blackwell was also involved in testing TCP and SMS communications issues and server issues including fixing database scroll issues on the server-based implementation. Mr. Blackwell also worked on addressing memory leakage issues. Mr. Blackwell also worked on integrating and developing Global Mapper features and other different maps used in LifeRing. Mr. Blackwell communicated with and worked closely with Mr. Rice to address these problems. Mr. Blackwell also worked on finalizing several versions of code. In April 2005, Mr. Hoff billed 17 hours for work on testing portions of LifeRing and developing code for the client side of LifeRing. Mr. Hoff was involved in testing location tracking and the loading of the client side application. Prior to April, Mr. Hoff was involved in development and testing for the location tracking, mapping data, interaction with maps and symbols, displaying of locations, maps and symbols, data and multimedia transfers, memory issues, and other issues related to both client and server side issues and had billed over 190 hours for such work between January 2005's conception date

and April 2005. In April 2005, Mr. Brown billed 160 hours for work on testing the LifeRing system. Mr. Brown was responsible for testing all portions of the code and client side program to find bugs and errors and to track reporting on such items. Mr. Brown also assisted Mr. Blackwell, Mr. Rice and myself on documentation of the LifeRing reports, manuals, and versions.

25. In May 2005, Mr. Rice billed at least 60 hours for work related to fixing problems related to SMS and voice transitions, photo transmissions, and SMS transmissions. Mr. Rice also spent considerable time creating and testing UDP communications protocol for establishing communications between applications and servers, including testing the new implementation of the protocol with a device. Mr. Rice also worked on the development of the CSIF code. During this time, Mr. Rice communicated frequently with Mr. Blackwell, who was also working on updates to the code. In May 2005, Mr. Hoff billed 11 hours for work on testing portions of LifeRing and working with Mr. Rice and Mr. Blackwell on testing features. In May 2005, Mr. Cardamone billed 80 hours for work related to the code and configuration of an SVN code repository. Mr. Cardamone also worked on the communications and display issues for LifeRing. Mr. Brown was responsible for testing all portions of the code and client side program to find bugs and errors and to track reporting on such items. In May 2005, Mr. Brown billed 80 hours for work on testing the LifeRing system. Mr. Brown also assisted Mr. Blackwell, Mr. Rice and myself on documentation of the LifeRing reports, manuals, and versions.

26. In June 2005, Mr. Rice billed at least 60 hours for work related to transmission issues with SMS and phone/conference calls. He worked on enhancing the stability of our SMS transmissions and supported our efforts on data reductions, UDP communications protocol, and CSIF code. Mr. Rice also made changes to the code for location tracking and message handling.

Mr. Rice also worked on network configuration for our development server. During this time, Mr. Rice communicated frequently with Mr. Blackwell, who was also working on updates to the code. In June 2005, Mr. Hoff billed 10 hours for work on testing new versions of LifeRing. Mr. Hoff was involved in testing SMS features of LifeRing. In June 2005, Mr. Cardamone billed 160 hours for work related to the LifeRing code repository and data reductions/transmissions related to LifeRing data. Mr. Brown was responsible for testing all portions of the code and client side program to find bugs and errors and to track reporting on such items. In June 2005, Mr. Brown billed over 160 hours for work on testing the LifeRing system. Mr. Brown also assisted Mr. Blackwell, Mr. Rice and myself on documentation of the LifeRing reports, manuals, and versions.

27. In July 2005, Mr. Rice billed over 40 billable hours on communications infrastructure issues and network configuration and server migration issues. In July 2005, Mr. Blackwell billed over 80 billable hours on various issues related to LifeRing including communications issues, display issues, and location tracking issues. Mr. Blackwell performed research to solve some issues that we were having with our server-based implementation and he worked on developing the new maps that had been integrated and tested in April and May 2005. Again, during this period, Mr. Blackwell remained in constant communication with Mr. Rice in order to resolve issues with the development of LifeRing. In July 2005, Mr. Cardamone billed 120 hours for work on data processing and data reduction relating to the georeferenced maps and georeferenced map data for presentation and use in LifeRing. In April 2005, Mr. Brown billed 80 hours for work on testing the LifeRing system. Mr. Brown was responsible for testing all portions of the code and client side program to find bugs and errors and to track reporting on

such items. Mr. Brown also assisted Mr. Blackwell, Mr. Rice and myself on documentation of the LifeRing reports, manuals, and versions.

28. In August 2005, Mr. Rice billed over 80 hours for work on the LifeRing database systems, including databases used for location tracking data. Mr. Rice's August 2005 work also included work on the processing output (maps) for LifeRing. During August 2005, we ran into problems with the startup and shutdown of the LifeRing application. Mr. Rice worked on researching ways to reduce delays in starting the program and obtaining new maps and location data immediately upon startup. This included the retrieval of maps in both non-server and server-based implementations. In August 2005, Mr. Blackwell billed over 150 hours for work on the development of LifeRing. During this time, Mr. Blackwell was intimately involved in researching and development of the server-based implementation. He also spent considerable time on testing communications issues and updating the display of information on the client. In August 2005, Mr. Cardamone billed 200 hours for work on data processing and data reduction relating to the georeferenced maps and georeferenced map data for presentation and use in LifeRing. During this time, Mr. Cardamone also worked on code processing output for display (e.g., georeferenced maps) and various display filter issues in the LifeRing code. In August 2005, Mr. Brown billed 160 hours for work on testing the LifeRing system. Mr. Brown was responsible for testing all portions of the code and client side program to find bugs and errors and to track reporting on such items. Mr. Brown also assisted Mr. Blackwell, Mr. Rice and myself on documentation of the LifeRing reports, manuals, and versions. Mr. Brown also assisted on preparing the LifeRing product for presentation to customers in Seattle.

29. In September 2005, Mr. Rice billed over 70 hours for work on the LifeRing location tracking database and location history issues. Mr. Rice also began to develop code to

more efficiently manage and distribute processing tasks in order to improve the delay issues that we were having in retrieving and processing maps and location information while also maintaining a reliable data communications flow. These improvements were being developed to support our non-server and server-based implementations. In September 2005, Mr. Blackwell billed over 150 hours for work on the development of LifeRing. Considerable time was spent on the server-based implementation, including on display, mapping, and communications tests and integrating the former with our existing non-server implementation. Mr. Blackwell spent a considerable amount of time researching whether and how to migrate our existing data and programs to the server and the impact of “over the air” transfer on the performance of the client. Mr. Blackwell worked closely with Mr. Rice to press for an October 2005 release of the server-based implementation. In September 2005, Mr. Brown billed over 190 hours for work on testing the LifeRing system. Mr. Brown was responsible for testing all portions of the code and client side program to find bugs and errors and to track reporting on such items. Mr. Brown also assisted Mr. Blackwell, Mr. Rice and myself on documentation of the LifeRing reports, manuals, and versions.

30. In October 2005, Mr. Rice billed over 85 hours for work on the LifeRing development. In addition to working on the location tracking database issues identified in our bug reports, Mr. Rice worked on finalizing the development of our image server implementation, which was the project for requesting/receiving maps from servers. Mr. Rice’s work on this included writing and testing code for communicating with different servers for providing georeferenced maps to the LifeRing client applications on devices. In October 2005, Mr. Blackwell billed over 140 hours for work on the development of LifeRing. Again, Mr. Blackwell spent considerable time on the server-based implementation and working to finalize

the new release with Mr. Rice. This was the culmination of significant effort as the requested maps come in various formats and protocols from different servers. Mr. Rice and Mr. Blackwell were deeply involved in finalizing the October 22, 2005 code base for the server-based implementation and addressing issues found in bug reports and testing services. In October 2005, Mr. Brown billed 180 hours for work on testing the LifeRing system. Mr. Brown was responsible for testing all portions of the code and client side program to find bugs and errors and to track reporting on such items. Mr. Brown also assisted Mr. Blackwell, Mr. Rice and myself on documentation of the LifeRing reports, manuals, and versions. Mr. Brown spent considerable amount of time testing the October 22, 2005 server-based implementation.

31. The claimed inventions of the '838 patent were reduced to practice by at least October 22, 2005. Exhibit 15 consists of code files that were checked into and present in AGIS's code repository by October 22, 2005, and Exhibit 16 is a description of the map request function for the October 22, 2005 server-based implementation. The features presented in Exhibit 4 were also implemented in the October 22, 2005 server-based implementation. The October 22, 2005 server-based implementation included messaging protocols for joining group(s) and sending location information of its devices among the devices. The system's maps were georeferenced maps with the devices and other entities represented as symbols on the maps. The system's maps were real-time maps which included generating for presentation new maps with (1) updated locations, (2) different map types, and (3) different zoom levels and panned areas. By interacting with the system's maps, users could touch symbols to send communications and multimedia to other users. Prior to October 22, 2005, AGIS's LifeRing maps were generated from map data kept in databases resident on the device or on a connected peripheral device from

which maps were downloaded. Substantial and diligent efforts culminated in the claimed inventions as implemented in the October 22, 2005 version of LifeRing.

32. The below chart describes exemplary¹ correspondences between claims 1-84 of the '838 patent and the October 22, 2005 version of AGIS LifeRing. I have confirmed that the each of the code files in Exhibit 15, which are the code files cited in the below chart, were checked in to the AGIS code repository by October 22, 2005.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
1. A computer-implemented method comprising: performing, by a first device: joining a communication network corresponding to a group, wherein joining the communication network comprises transmitting a message including an identifier corresponding to the group;	<p>The AGIS LifeRing product practices the computer implemented method of claim 1. The AGIS LifeRing product practices performing “by a first device: joining a communication network corresponding to a group, wherein joining the communication network comprises transmitting a message including an identifier corresponding to the group.” The first device is programmed to receive a message from a second device related to joining a group. The process of joining a group is performed by sending messages to server and other users to join a group.</p> <p>The AGIS LifeRing product includes systems and methods for executing an application on PDA/cell phones, now referred to as a smartphone devices.</p> <p>See, e.g., “call.cpp” references to “phone” and “other phones.”</p>

¹ This is not intended to be an exhaustive identification of correspondence of the LifeRing product to the claims.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>The operator can then send voice conference or digital data to those AGIS equipped units by simply selecting their assigned NET SoftSwitch.</p> <p>The AGIS application software does not affect the operation of the PDA, the Cell phone or the GPS until it is activated. When the AGIS application software is selected, the AGIS PDA Cell phone is turned on, the GPS is automatically connected to the PDA / Cell phone through a Bluetooth interface and the AGIS Logo appears. Shortly thereafter the AGIS operation display appears. See Figures 1, 2, and 3</p> <div data-bbox="532 527 1230 934" style="text-align: center;"> </div> <p style="text-align: center;">Figure 1 Figure 2 Figure 3</p> <p>Exhibit 4 to the Beyer Declaration, depicting AGIS PDA/cell phones.</p> <p>The file “display/msgproc.cpp” implements the function “process_net_msg()”² where the logging function indicates “rcvd net msg” indicative of receiving a network message. The “nmsg” variable of type “NETUSERS” which is input to this function is tested against its member variable requestType which is specified in the file “csiftranscode.h”³ and states to have “(J)oining or R)request current participants”. The “nmsg.requestType” is then branched for cases of “I” for initial joining, “J” for joining, “R” for requesting and “X” for exiting the network. For the case of “J” for joining, the function makes a call to the “update_address_book()” function with sender information as the input parameters.</p> <p>The file “initfiles.cpp” implements the “update_address_book()”⁴ where the sender information is used to find its index in the address book by a call to “find_addr_number()”. The return from this call is used to update the array variable “csif_addr[]” by either adding the address if it does not exist indicative of a return value of -1 or by copying the “ipaddr” into a struct member of the indexed variable.</p>

² See, e.g., display/msgproc.cpp at L1234-1292..

³ See, e.g., csiftranscode/csiftranscode.h at L254-263.

⁴ See, e.g., display/initfiles.cpp at L701-721.

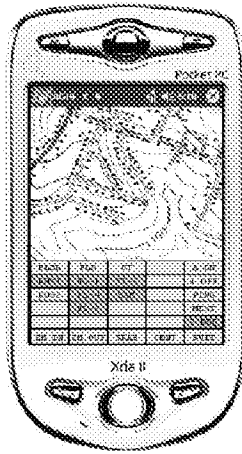
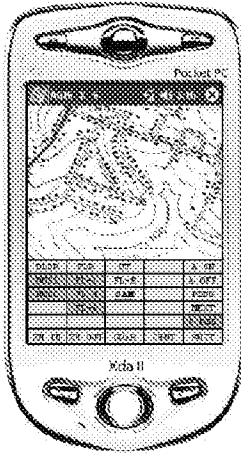
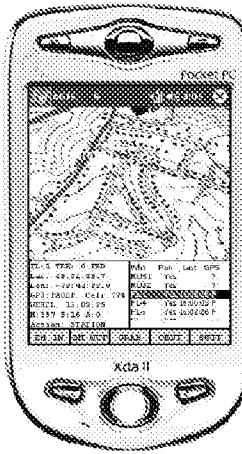
U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>The “csif_addr[]” array is an array of type “ADDRESS_ENTRY”⁵ which is specified in the file “csiftranscode.h”⁶ The members of this struct includes “name”, “number”, “group”, “type”, “level” and “ipaddr”. The marking of the “csif_addr[]” variable array constitutes the joining of a group.</p> <p>The server further provides additional functionality in file “server/msgproc.cpp” implements the function “process_net_msg()”⁷ where the logging function indicates “rcvd net msg” indicative of receiving a network message. The variables “from” of type character string and “nmsg” of type “NETUSERS” as described above is input to this function. Similar to described above, the “nmsg.requestType” is then branched for cases of “J” and not “J” described to mean of type “R” are handled. For the case of joining a group where “nmsg.requestType” equals “J”, the function “update_address_book()” is called with the sender and receiver information and returns with an index into the address book array “csif_addr” as discussed above. Then the logging command indicates that the “address book [address book index] updated with ipaddr [address book ip address]” and writes “user [name] joined network” to the display. Followed by two calls to “send_net_msg_participants()” using a wildcard designator “-1” indicating everything in the respective field is to be sent. The first call sends everything in the table to the currently added user. The second call sends to everyone the currently added user. In the same file the function “send_net_msg_participants()”⁸ documents that the user of “-1” in the first field indicates a “send to all” and the use of “-1” in the second field indicates a “send all net participants” condition.</p>

⁵ See, e.g., display/initfiles.h at L 35

⁶ See, e.g., csiftranscode/csiftranscode.h at L45-54.

⁷ See, e.g., server/msgproc.cpp at L235-293.

⁸ See, e.g., server/msgproc.cpp at L108-191.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p><i>2.5 AGIS Net Assignment and Ping</i></p> <p>The AGIS permits the person in charge of the network to specify which AGIS cell phones are to be part of the network. This is accomplished by the AGIS operator selecting the P REQ SoftSwitch which causes a matrix of the AGIS phones to appear. Note that this is a multilayered matrix and, if there are more than 28 phones, the operator can select the NEXT SoftSwitch which will cause yet another group of participants to appear. The operator can drop highlighted AGIS units from the net by selecting their SoftSwitch or add new ones to the net by selecting them. In Figure 22 note that AGIS Participant RUS 2 is not part of the AGIS communications net. In Figure 23 note that RUS 2 has been made part of the AGIS communications net and FL 5 and SAM have been dropped from the AGIS communications net. The AGIS operator can also select to immediately transmit his location and all his tracks to another AGIS unit and cause that unit to immediately report its location and all its tracks by selecting the PING SoftSwitch.</p> <p>To determine the status of the various AGISs in the communications net, the AGIS operator selects the PLIST SoftSwitch which causes display of the current AGIS communications net participants to be displayed in the lower left of the AGIS display. See Figure 24.</p> <div style="display: flex; justify-content: space-around; align-items: center;"> <div style="text-align: center;">  <p>Figure 22</p> </div> <div style="text-align: center;">  <p>Figure 23</p> </div> <div style="text-align: center;">  <p>Figure 24</p> </div> </div> <p>Exhibit 4 to the Beyer Declaration, at 10-11.</p> <p>Similarly, the server further provides functionality to support the server side function similar to the client side in the file “server/initfiles.cpp” by implementing the function “update__address__book()”⁹ where similar to the description above the call to “find__addr__number()” function retrieves an address index. If the address does not exist indicative of a return index value of “-1” calls the function “add__address__book()” otherwise updates the “csif__addr” array with the ip address input to the function. In the same file the function “add__address__book()”¹⁰ is implemented where the “csif__addr” array is populated with the “name”, “number”, “group”, “ipaddr” and a “isValid” member variable.</p>	

⁹ See, e.g., server/initfiles.cpp at L194-211.¹⁰ See, e.g., server/initfiles.cpp at L163-191.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p>participating in the group, wherein participating in the group includes sending first location information to a first server and receiving second location information from the first server, the first location information comprising a location of the first device, the second location information comprising one or more locations of one or more respective second devices included in the group;</p>	<p>The AGIS LifeRing product practices “participating in the group, wherein participating in the group includes sending first location information to a first server and receiving second location information from the first server, the first location information comprising a location of the first device, the second location information comprising one or more locations of one or more respective second devices included in the group.” Once the devices have formed a group they share their location with each other. The AGIS software captures location information on the device through a serial communication link configured to continuously process GPS information messages. Messages such as location, precision and constellation are captured on an independent thread and when a valid message is received the thread triggers an event indicating the presence of a new message. The software then sends the new location to others through either a server or direct connection. The first server provides group management function on a processor executing a group management software code that communicates with a messaging component. The messaging component provides communications function on a processor executing a communication software code that interfaces with the outside world through various links such as Ethernet or Radio Frequency.</p> <p>The “csif” module provides a service to start a GPSThread that communicates with a serial port. In the file “csif/csif.cpp” the function “initGPS()”¹¹ is implemented where the variable comport is as the argument. The variable “gpsListener” is then assigned to a new GPSThread class using the comport variable followed by calling the “GPSThread->start()” call.</p> <p>In the file “gpslistener.cpp” the “start()”¹² method is implemented where a thread is created for processing in the background. The thread performs the function “process()”¹³ and uses a loop to scan the comport documented as “read in a message from serial port, will only wait for 1 second for data”¹⁴ and once a message is detected that it calls the “processGPSMessage()” passing the message and length of the message. The function “processGPSMessage()”¹⁵ calls the “enqueueGPSMsg()” of the “DDLDatabase” class where the GPS message is queued.</p>

¹¹ See, e.g., csif/csif.cpp at L126-148.

¹² See, e.g., csif/gpslistener.cpp at L36-51.

¹³ See, e.g., csif/gpslistener.cpp at L101-235.

¹⁴ See, e.g., csif/gpslistener.cpp at L158.

¹⁵ See, e.g., csif/gpslistener.cpp at L76-99.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>In the file “display.cpp” inside the initialization function “InitInstance()”¹⁶ the GPS message event processor is registered through the call to the “registerGPSListener()” much like the DDL messages. The GPS messages are then processed by the callback event capture routine “WndProc()”¹⁷ where the “ID_TIMER_BLINK” event documented to represent a 1 second timer makes a call to “check_gps_msgs()”, “check_for_internal_msgs()” and “check_for_db_msgs()”.</p> <p>In the file “gps.cpp” the function “check_gps_msgs()”¹⁸ is implemented where the queued GPS messages are process incrementally with a call to the “process_csif_gps_msg()”. The function “process_csif_gps_msg()”¹⁹ is implemented where the message is retrieved by a call to “get_gps_msg()” and based on one of three choices of a message type captured by the “msgType” member of the message variable “gmsg” that the position, precision or constellation of the gps message is processed. For the case where the “msgType” is of “GPS_MSG_POSITION” type the call is made to “process_gps_pos()” passing the “gmsg.body.position” variable member. In the same file the function “process_gps_pos()”²⁰ is implemented. This function utilizes the “gps_pos_decode()” function to covert the input message’s members such as “gpsp.latitude” and “gpsp.longitude” to a floating point format in the “lat” and “lon” variables. Further into the function, the “lat” and “lon” variables are used to populate the members of the “temptrk” struct variable which is of type “track_file_struct”.</p> <p>In the file “display/display.cpp” the function “check_ip_address()”²¹ is an example of communication with transmission of location information. This function is conditioned to use certain code for the case where the “PPC2003” is defined, indicative of SMS capability, or not defined. In both routines similar functionality is performed where calls to functions “send_net_msg_netmgmt()” to join a network group and “SendOwnPosString()” to send location information under various connection states.</p>

¹⁶ See, e.g., display/display.cpp at L274-581.

¹⁷ See, e.g., display/display.cpp at L584-969.

¹⁸ See, e.g., display/gps.cpp at L346-365.

¹⁹ See, e.g., display/gps.cpp at L292-336.

²⁰ See, e.g., display/gps.cpp at L77-126.

²¹ See, e.g., display/display.cpp at L1015-1037 and L1076-1115.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>In the file “display/buttons.cpp” in the main routine where it is documented that button actions are processed by “button_actions”²² and are implemented in the function “button_actions()”²³ the case for pressing the “BUTTON_PING:” is implemented where calls are made to “SendOwnPosString()” and “send_all_tracks()”.</p> <p>In the file “display/msgproc.cpp” the function “SendOwnPosString()”²⁴ is implemented where the “op” struct variable of type “OWN_POSITION” is populated with location information such as latitude and longitude. Then the “dmsg” variable of type “DDLMSG” is set to have a “msgType” of “DDL_MSG_OWN_POSITION” and variable “op” set to one of its other member variables. Then the “dmsg” is set to a buffer with the call to “put_ddl_msg()” and a call to “send_message_bulk()” when the transmit method “xmit_method” is set to “SERVER” and using a “server_ip” variable. Alternatively, direct transmission may occur with the call to “send_message_direct()”</p> <p>Similarly, in the file “display/msgproc.cpp” the function “send_track_msg()”²⁵ is implemented which includes location information along with previous location information. Similar to the previous function, the “dmsg” is set to type “DDL_MSG_TRACK_REPORT” and “tmsg” variable with location information is set to its member variable. The functions “put_ddl_msg()” followed by “send_message_bulk()” or “send_message_direct()” transmit the message.</p> <p>The “send_track_msg()” function is used under the condition where “add_track()”²⁶ and “update_track_id()”²⁷ functions in the “display/track.cpp” is invoked. The “update_track_id()” is used in user interface as views in the “buttons.cpp” file in the main routine where button actions are implemented “button_actions()” as discussed above where the case for “BUTTON_TRACK_ID_UNKN”, “BUTTON_TRACK_ID_FRND” and “BUTTON_TRACK_ID_HOSTILE” cases are handled for various features including the call to “update_track_id()”.</p>

²² See, e.g., display/buttons.cpp at L594-598 and L1344-1346.

²³ See, e.g., display/buttons.cpp at L1344-2365.

²⁴ See, e.g., display/msgproc.cpp at L170-325.

²⁵ See, e.g., display/msgproc.cpp at L615-682.


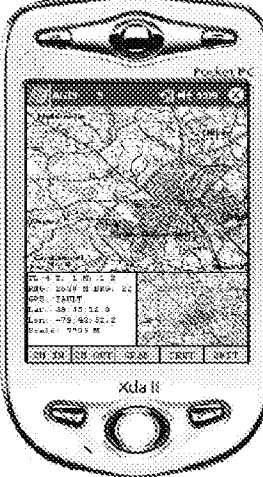
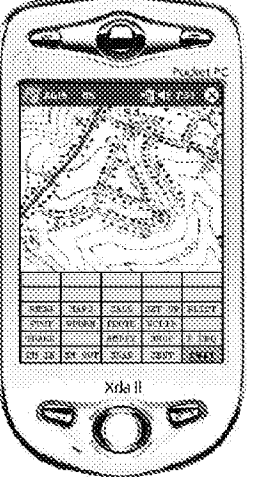
²⁶ See, e.g., display/track.cpp at L373-477.

²⁷ See, e.g., display/track.cpp at L482-510.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>On the server side, the file “server/msgproc.cpp” implements the function “process_csif_dll_msg()”²⁸ where the incoming message uses the “msgType” to branch to a section of the code depicted to represent the “DDL_MSG_OWN_POISITION” case where the message is passed through by a call to “pass_thru_msg()” function.</p> <p>The file “server/server.cpp” implements the main function “WndProc()”²⁹ as a callback where a receipt of a message triggers its execution. The case of “CSIF_DDL_MSG_AVAILABLE” is used to call the “getNextDDLMessage()” with a variable “route” that is used to call the “process_csif_dll_msg()” function as described above to make the call for pass through.</p>
<p>presenting, via an interactive display of the first device, a first interactive, georeferenced map and a first set of one or more user-selectable symbols corresponding to a first set of one or more of the second devices, wherein the first set of symbols are positioned on the first georeferenced map at respective positions corresponding to the locations of the first set of second devices, and wherein first georeferenced map data relate positions on the first</p>	<p>The AGIS LifeRing product practices “presenting, via an interactive display of the first device, a first interactive, georeferenced map and a first set of one or more user-selectable symbols corresponding to a first set of one or more of the second devices, wherein the first set of symbols are positioned on the first georeferenced map at respective positions corresponding to the locations of the first set of second devices, and wherein first georeferenced map data relate positions on the first georeferenced map to spatial coordinates.” The devices running the software display an interactive display with locations of group devices, georeferenced and shown in relation to others. The display further draws tracks and other features related to the display of devices.</p> <p>An exemplary interface from the AGIS client running on a mobile device is depicted below:</p>

²⁸ See, e.g., server/msgproc.cpp at L377-474.

²⁹ See, e.g., server/server.cpp at L228-386.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
georeferenced map to spatial coordinates;	<p>The AGIS application software does not affect the operation of the PDA, the Cell phone or the GPS until it is activated. When the AGIS application software is selected, the AGIS PDA Cell phone is turned on, the GPS is automatically connected to the PDA / Cell phone through a Bluetooth interface and the AGIS Logo appears. Shortly thereafter the AGIS operation display appears. See Figures 1, 2, and 3</p> <div style="display: flex; justify-content: space-around; align-items: center;">    </div> <p style="text-align: center;"> Figure 1 Figure 2 Figure 3 </p> <p>Exhibit 4 to Beyer Declaration, at 4.</p> <p>In the file “display/display.cpp” the main method “WndProc()”³⁰ where display events are captured and processed, the case for “WM_PAINT” where the display is drawn is implemented. When a “WM_PAINT” event occurs calls are made to “refresh_tact()”, “refresh_stat()” and “refresh_inset()” passing the display window handle where the map, locations and other information gets displayed.</p> <p>In the file “tact.cpp” the function “refresh_tact()”³¹ is implemented. There are three types of this function, two with an input handle and one without. The one without the handle first retrieves the current handle and then calls the “refresh_tact()” function version with the handle. In this function, among other features, the system draws the map by a call to “ww_draw_map()” and draws locations by a call to “track_draw()”. A special “refresh_tact()” implementation only draws the tactical area which is documented to correspond to the display’s WM_PAINT event and only accepts a window handle of type HWND. The function “refresh_tact()” is used throughout the code to signal an update to the view within “buttons.cpp”, “display.cpp”, “images.cpp”, “inset.cpp”,</p>

³⁰ See, e.g., display/display.cpp at L584-969.³¹ See, e.g., display/tact.cpp at L200-306.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“maps.cpp”, “msgproc.cpp”, “track.cpp” and others to signal the update of the graphical window.</p> <p>In the file “maps.cpp” the function “ww_draw_map()”³² is implemented. The function utilizes georeferenced latitude and longitude information to position the map using variables “urlat”, “urlon”, “lllat” and “lllon” for the upper right and lower left latitude and longitude. The variable array “ww_map” is used to hold the map data and is read in the same file using the function “ww_read_map()”³³ from local files. The function “ww_read_maps()”³⁴ makes individual calls to “ww_read_map()” to read the “world.pnt”, “plit.pnt”, “water.pnt” and a local map through invocation of the “local_read_map()” function pointed to by the “test.user.ump” file. The “ww_read_maps()” function is called in the file “display/display.cpp” by the function “InitInstance()”³⁵ when the system has started.</p> <p>In the file “stat.cpp” the function “refresh_stat()”³⁶ is implemented. There are two types of this function, one with an input handle and one without. The one without the handle first retrieves the current handle and then calls the “refresh_stat()” function version with the handle. In this function, among other features, the system “refresh status window” by a call to “stat_template_refresh()”. There are other methods that the system implements such as “show_status_window()”³⁷ and the main window process in the “WndStatProc()”³⁸ callback function “WM_PAINT” event which calls the “refresh_stat()” and in turn “stat_template_refresh()”. Upon creation of the stat view by use of the function “create_stat()”³⁹ to create the graphical interface followed by a call to the “stat_template_refresh()”. Furthermore, the function “refresh_stat()” is used throughout the code within “buttons.cpp”, “display.cpp”, “gps.cpp”, “msgproc.cpp”, “partlist.cpp” to signal the update of the stat window. The function “stat_template_refresh()”⁴⁰ is implemented with a call to “status_hook()”. The function</p>

³² See, e.g., maps.cpp at L213-276.

³³ See, e.g., maps.cpp at L69-153.

³⁴ See, e.g., maps.cpp at L726-732.

³⁵ See, e.g., display/display.cpp at L268-581.

³⁶ See, e.g., stat.cpp at L321-371.

³⁷ See, e.g., stat.cpp at L387-403.

³⁸ See, e.g., stat.cpp at L39-115.

³⁹ See, e.g., stat.cpp at L119-166.

⁴⁰ See, e.g., stat.cpp at L296-317.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“status_hook()”⁴¹ is implemented where various information gets presented such as track information, current latitude, current longitude, GPS status amongst others.</p> <p>In the file “inset.cpp” the function “refresh_inset()”⁴² is implemented. There are two types of this function, one with an input handle and one without. Both versions retrieve the Inset window handle through either a global handle or passed in handle and make a call to the “refresh_inset_window()” function. The function “refresh_inset_window()”⁴³ is implemented in the same file with calls made to “refresh_inset_map()”, “refresh_inset_gps()” and “refresh_inset_trackamp()” based on the value held in the “inset_mode” variable set to “INSETWIN_MAP_USE”, “INSETWIN_GPS_USE” or “INSETWIN_TRACKAMP_USE”.</p> <p>The function “refresh_inset_map()”⁴⁴ is implemented in the same file and the georeferenced latitude and longitude values such as “inset_mapllon” or “inset_mapurlat” that are used to draw the map with the track information used from the “track_file” variable. The function draws the map using the system “Bit_Blt()” or call to “ww_draw_image_inset() and then calls the “inset_track_draw()” function which then draws all tracks. The function “ww_draw_image_inset()”⁴⁵ is implemented in the “display/images.cpp” and utilizes georeferenced latitude and longitude values. The function “inset_track_draw()”⁴⁶ is documented that to “plot all tracks as dots in inset window” and utilizes a loop across all tracks to draw all tracks. The function “refresh_inset_gps()”⁴⁷ is implemented in the same file and primarily provides information about the gps satellites and their respective signal strengths. The function “refresh_inset_trackamp()”⁴⁸ updates the amplified track mode equipped with its own series of functions including messaging, location and amplified views.</p>
sending, to a second server, a request for	The AGIS LifeRing Product practices “sending, to a second server, a request for second georeferenced map data different from the first

⁴¹ See, e.g., stat.cpp at L207-292⁴² See, e.g., inset.cpp at L708-737.⁴³ See, e.g., inset.cpp at L684-705.⁴⁴ See, e.g., inset.cpp at L285-393.⁴⁵ See, e.g., display/images.cpp at L534-596.⁴⁶ See, e.g., inset.cpp at L220-282.⁴⁷ See, e.g., inset.cpp at L395-512.⁴⁸ See, e.g., inset.cpp at L535-657.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
second georeferenced map data different from the first georeferenced map data;	<p>georeferenced map data.” The first device makes a request for a georeferenced map by interacting with the device’s user interface. The second server provides map data by a data serving function on a processor executing a data serving software code that communicates with a messaging component. The messaging component provides communications function on a processor executing a communication software code that interfaces with the outside world through various links such as Ethernet or Radio Frequency.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁴⁹ implements the main routine for capturing the button press event and processing based on the button pressed. For the case of the “BUTTON_IMAGE_REQUEST” a call is made to the “image_server_request()” function. In the file “display/images.cpp” the function “image_server_request()”⁵⁰ is implemented. In this function the coordinate boundaries are retrieved by a call to “coord_bounds()” and sent to the function “SendImageRequest()”. In the file “coords.cpp” the function “coord_bounds()”⁵¹ is implemented where the latitude and longitude are used to generate the values required. In the file “display/msgproc.cpp” the function “SendImageRequest()”⁵² is implemented where a message is composed using the “DDL_MSG_MAP_REQUEST” type. The message is then processed by the “put_ddl_msg()” routine as described above and sent to the server with the call to “send_message_bulk()” as described above using the “server_ip” variable as the target for the server. Because the address is implemented as a server_ip variable, the variable can be set to one or more addresses, and function calls can be ultimately parametrized.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”⁵³ is implemented to process received messages. The condition “DDL_MSG_MAP_REQUEST” is captured and a call is made to “process_map_request()”. The function “process_map_request()”⁵⁴ is implemented in the same file where a message of type “DDL_MSG_MAP_RESPONSE” is formed and returns a map by calls to “put_ddl_msg()” and “send_message()”.</p>

⁴⁹ See, e.g., buttons.cpp at L1344-2365.

⁵⁰ See, e.g., display/images.cpp at L752-761.

⁵¹ See, e.g., coords.cpp at L88-98.

⁵² See, e.g., display/msgproc.cpp at L963-996.

⁵³ See, e.g., server/msgproc.cpp at L377-474.

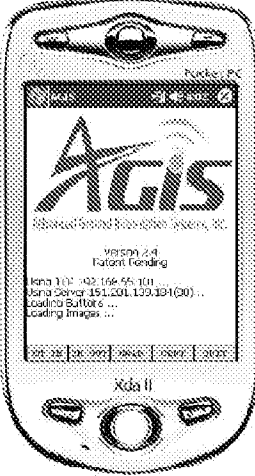
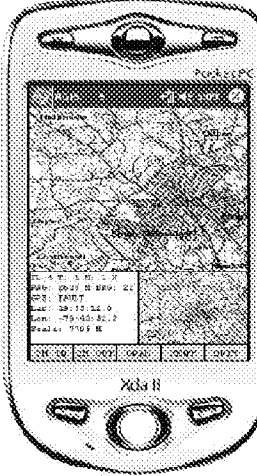
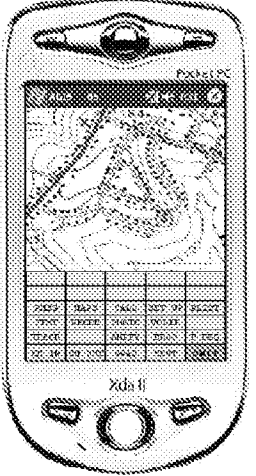
⁵⁴ See, e.g., server/msgproc.cpp at L312-332.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>In the file “display/msgproc.cpp” the function “process_csif_msg()”⁵⁵ is implemented where the condition “DDL_MSG_MAP_RESPONSE” is processed by a call to “process_image_server_response()”. In the file “display/images.cpp” the function “process_image_server_response()”⁵⁶ is implemented where it is logged that with the “response from image server”.</p>
receiving, from the second server, the second georeferenced map data;	<p>The AGIS LifeRing product practices “receiving, from the second server, the second georeferenced map data.” The first device makes a request for a georeferenced map by interacting with the device’s user interface. The second server provides map data by a data serving function on a processor executing a data serving software code that communicates with a messaging component. The messaging component provides communications function on a processor executing a communication software code that interfaces with the outside world through various links such as Ethernet or Radio Frequency.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁵⁷ implements the main routine for capturing the button press event and processing based on the button pressed. For the case of the “BUTTON_IMAGE_REQUEST” a call is made to the “image_server_request()” function. In the file “display/images.cpp” the function “image_server_request()”⁵⁸ is implemented. In this function the coordinate boundaries are retrieved by a call to “coord_bounds()” and sent to the function “SendImageRequest()”. In the file “coords.cpp” the function “coord_bounds()”⁵⁹ is implemented where the latitude and longitude are used to generate the values required. In the file “display/msgproc.cpp” the function “SendImageRequest()”⁶⁰ is implemented where a message is composed using the “DDL_MSG_MAP_REQUEST” type. The message is then processed by the “put_ddl_msg()” routine as described above and sent to the server with the call to “send_message_bulk()” as described above using the “server_ip” variable as the target for the server.</p>

⁵⁵ See, e.g., display/msgproc.cpp at L1391-1472.⁵⁶ See, e.g., display/images.cpp at L763-768.⁵⁷ See, e.g., buttons.cpp at L1344-2365.⁵⁸ See, e.g., display/images.cpp at L752-761.⁵⁹ See, e.g., coords.cpp at L88-98.⁶⁰ See, e.g., display/msgproc.cpp at L963-996.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	In the file "server/msgproc.cpp" the function "process_csif_ddl_msg()" ⁶¹ is implemented to process received messages. The condition "DDL_MSG_MAP_REQUEST" is captured and a call is made to "process_map_request()". The function "process_map_request()" ⁶² is implemented in the same file where a message of type "DDL_MSG_MAP_RESPONSE" is formed and returns a map by calls to "put_ddl_msg()" and "send_message()".
presenting, via the interactive display of the first device, a second georeferenced map and a second set of one or more user-selectable symbols corresponding to a second set of one or more of the second devices, wherein the second set of symbols are positioned on the second georeferenced map at respective positions corresponding to the locations of the second set of second devices, and wherein the second georeferenced map data relate positions on the second georeferenced map to spatial coordinates;	<p>The AGIS LifeRing product practices "presenting, via the interactive display of the first device, a second georeferenced map and a second set of one or more user-selectable symbols corresponding to a second set of one or more of the second devices, wherein the second set of symbols are positioned on the second georeferenced map at respective positions corresponding to the locations of the second set of second devices, and wherein the second georeferenced map data relate positions on the second georeferenced map to spatial coordinates." The device has an interactive display showing the locations of other devices overlaid on a georeferenced map along with its own. The display of locations and tracks are described above.</p> <p>As depicted above, the user client of the AGIS software includes a user interface display with a map overlaid with symbols representing other members of a group at their geographic locations on the georeferenced map.</p>

⁶¹ See, e.g., server/msgproc.cpp at L377-474.⁶² See, e.g., server/msgproc.cpp at L312-332.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p>The AGIS application software does not affect the operation of the PDA, the Cell phone or the GPS until it is activated. When the AGIS application software is selected, the AGIS PDA Cell phone is turned on, the GPS is automatically connected to the PDA / Cell phone through a Bluetooth interface and the AGIS Logo appears. Shortly thereafter the AGIS operation display appears. See Figures 1, 2, and 3</p> <div style="display: flex; justify-content: space-around; align-items: center;"> <div style="text-align: center;">  <p>Figure 1</p> </div> <div style="text-align: center;">  <p>Figure 2</p> </div> <div style="text-align: center;">  <p>Figure 3</p> </div> </div> <p>Exhibit 4 to Beyer Declaration, at 4.</p> <p>In the file “display/display.cpp” the main method “WndProc()”⁶³ where display events are captured and processed, the case for “WM_PAINT” where the display is drawn is implemented. When a “WM_PAINT” event occurs calls are made to “refresh_tact()”, “refresh_stat()” and “refresh_inset()” passing the display window handle where the map, locations and other information gets displayed.</p> <p>In the file “tact.cpp” the function “refresh_tact()”⁶⁴ is implemented. There are three types of this function, two with an input handle and one without. The one without the handle first retrieves the current handle and then calls the “refresh_tact()” function version with the handle. In this function, among other features, the system draws the map by a call to “ww_draw_map()” and draws locations by a call to “track_draw()”. A special “refresh_tact()” implementation only draws the tactical area which is documented to correspond to the display’s WM_PAINT event and only accepts a window handle of type HWND. The function “refresh_tact()” is used throughout the code to signal an update to the view within “buttons.cpp”, “display.cpp”, “images.cpp”, “inset.cpp”,</p>	

⁶³ See, e.g., display/display.cpp at L584-969.⁶⁴ See, e.g., display/tact.cpp at L200-306.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“maps.cpp”, “msgproc.cpp”, “track.cpp” and others to signal the update of the graphical window.</p> <p>In the file “maps.cpp” the function “ww_draw_map()”⁶⁵ is implemented. The function utilizes georeferenced latitude and longitude information to position the map using variables “urlat”, “urlon”, “lllat” and “lllon” for the upper right and lower left latitude and longitude. The variable array “ww_map” is used to hold the map data and is read in the same file using the function “ww_read_map()”⁶⁶ from local files. The function “ww_read_maps()”⁶⁷ makes individual calls to “ww_read_map()” to read the “world.pnt”, “plit.pnt”, “water.pnt” and a local map through invocation of the “local_read_map()” function pointed to by the “test.user.ump” file. The “ww_read_maps()” function is called in the file “display/display.cpp” by the function “InitInstance()”⁶⁸ when the system has started.</p> <p>In the file “stat.cpp” the function “refresh_stat()”⁶⁹ is implemented. There are two types of this function, one with an input handle and one without. The one without the handle first retrieves the current handle and then calls the “refresh_stat()” function version with the handle. In this function, among other features, the system “refresh status window” by a call to “stat_template_refresh()”. There are other methods that the system implements such as “show_status_window()”⁷⁰ and the main window process in the “WndStatProc()”⁷¹ callback function “WM_PAINT” event which calls the “refresh_stat()” and in turn “stat_template_refresh()”. Upon creation of the stat view by use of the function “create_stat()”⁷² to create the graphical interface followed by a call to the “stat_template_refresh()”. Furthermore, the function “refresh_stat()” is used throughout the code within “buttons.cpp”, “display.cpp”, “gps.cpp”, “msgproc.cpp”, “partlist.cpp” to signal the update of the stat window. The function “stat_template_refresh()”⁷³ is implemented with a call to “status_hook()”. The function</p>

⁶⁵ See, e.g., maps.cpp at L213-276.

⁶⁶ See, e.g., maps.cpp at L69-153.

⁶⁷ See, e.g., maps.cpp at L726-732.

⁶⁸ See, e.g., display/display.cpp at L268-581.

⁶⁹ See, e.g., stat.cpp at L321-371.

⁷⁰ See, e.g., stat.cpp at L387-403.

⁷¹ See, e.g., stat.cpp at L39-115.

⁷² See, e.g., stat.cpp at L119-166.

⁷³ See, e.g., stat.cpp at L296-317.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“status_hook()”⁷⁴ is implemented where various information gets presented such as track information, current latitude, current longitude, GPS status amongst others.</p> <p>In the file “inset.cpp” the function “refresh_inset()”⁷⁵ is implemented. There are two types of this function, one with an input handle and one without. Both versions retrieve the Inset window handle through either a global handle or passed in handle and make a call to the “refresh_inset_window()” function. The function “refresh_inset_window()”⁷⁶ is implemented in the same file with calls made to “refresh_inset_map()”, “refresh_inset_gps()” and “refresh_inset_trackamp()” based on the value held in the “inset_mode” variable set to “INSETWIN_MAP_USE”, “INSETWIN_GPS_USE” or “INSETWIN_TRACKAMP_USE”.</p> <p>The function “refresh_inset_map()”⁷⁷ is implemented in the same file and the georeferenced latitude and longitude values such as “inset_mapllon” or “inset_mapurlat” that are used to draw the map with the track information used from the “track_file” variable. The function draws the map using the system “Bit_Blt()” or call to “ww_draw_image_inset() and then calls the “inset_track_draw()” function which then draws all tracks. The function “ww_draw_image_inset()”⁷⁸ is implemented in the “display/images.cpp” and utilizes georeferenced latitude and longitude values. The function “inset_track_draw()”⁷⁹ is documented that to “plot all tracks as dots in inset window” and utilizes a loop across all tracks to draw all tracks. The function “refresh_inset_gps()”⁸⁰ is implemented in the same file and primarily provides information about the gps satellites and their respective signal strengths. The function “refresh_inset_trackamp()”⁸¹ updates the amplified track mode equipped with its own series of functions including messaging, location and amplified views.</p>
and identifying user interaction with the	The AGIS LifeRing product practices “and identifying user interaction with the interactive display selecting one or more of the second set of

⁷⁴ See, e.g., stat.cpp at L207-292

⁷⁵ See, e.g., inset.cpp at L708-737.

⁷⁶ See, e.g., inset.cpp at L684-705.

⁷⁷ See, e.g., inset.cpp at L285-393.

⁷⁸ See, e.g., display/images.cpp at L534-596.

⁷⁹ See, e.g., inset.cpp at L220-282.

⁸⁰ See, e.g., inset.cpp at L395-512.

⁸¹ See, e.g., inset.cpp at L535-657.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
interactive display selecting one or more of the second set of user-selectable symbols corresponding to one or more of the second devices and positioned on the second georeferenced map and user interaction with the display specifying an action and, based thereon, sending third data to the selected one or more second devices via the first server.	<p>user-selectable symbols corresponding to one or more of the second devices and positioned on the second georeferenced map and user interaction with the display specifying an action and, based thereon, sending third data to the selected one or more second devices via the first server.” The device utilizes buttons on the display to send messages, make calls and send photo or video. The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients. The first server provides group management function on a processor executing a group management software code that communicates with a messaging component. The messaging component provides communications function on a processor executing a communication software code that interfaces with the outside world through various links such as Ethernet or Radio Frequency.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁸² implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”⁸³ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”⁸⁴ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”⁸⁵ and “num_on_que_list()”⁸⁶ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file</p>

⁸² See, e.g., buttons.cpp at L1344-2365.⁸³ See, e.g., display/ftext.cpp at L176-244.⁸⁴ See, e.g., netselect.cpp at L152-221.⁸⁵ See, e.g., quelist.cpp at L16-22.⁸⁶ See, e.g., quelist.cpp at L60-71.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“display/msgproc.cpp” the function “send_ftext_msg()”⁸⁷ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netsselect.cpp” the function “create_netlist()”⁸⁸ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”⁸⁹ and subsequently to “form_net_line()”⁹⁰ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁹¹ and subsequently “send_net_msg_participants()”⁹² the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁹³ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”⁹⁴ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”⁹⁵ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the</p>

⁸⁷ See, e.g., display/msgproc.cpp at L328-399.

⁸⁸ See, e.g., netsselect.cpp at L391-492.

⁸⁹ See, e.g., netsselect.cpp at L373-384.

⁹⁰ See, e.g., netsselect.cpp at L352-370.

⁹¹ See, e.g., display/msgproc.cpp at L1234-1292.

⁹² See, e.g., display/msgproc.cpp at L828-922.

⁹³ See, e.g., photo.cpp at L189-292.

⁹⁴ See, e.g., server/msgproc.cpp at L377-474.

⁹⁵ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p> <p>In the file “display/msgproc.cpp” the function “send_net_msg_participants()”⁹⁶ is implemented where it is documented that it is able to send messages to all or subset of participants all information or subset of information. The message is composed in the “nmsg” variable with a “msgType” of “DDL_MSG_NETUSERS” and calls to “put_ddl_msg()” followed by “send_message_bulk()” as explained above with communication to a server.</p>
2. The method of claim 1, wherein the message is transmitted to the first server.	<p>The AGIS LifeRing product practices “wherein the message is transmitted to the first server.” The devices are programmed to send and receive messages to and from a server.</p> <p>The message of claim 1 is transmitted to the first server to join a group by transmission to the first server. The first server provides group management function on a processor executing a group management software code that communicates with a messaging component. The messaging component provides communications function on a processor executing a communication software code that interfaces with the outside world through various links such as Ethernet or Radio Frequency.</p> <p>The file “display/msgproc.cpp” implements the function “process_net_msg()”⁹⁷ where the logging function indicates “rcvd net msg” indicative of receiving a network message. The “nmsg” variable of type “NETUSERS” which is input to this function is tested against its member variable requestType which is specified in the file “csiftranscode.h”⁹⁸ and states to have “(J)oining or (R)equest current participants”. The “nmsg.requestType” is then branched for cases of “I” for initial joining, “J” for joining, “R” for requesting and “X” for exiting the network. For the case of “J” for joining, the function makes</p>

⁹⁶ See, e.g., display/msgproc.cpp at L828-922.⁹⁷ See, e.g., display/msgproc.cpp at L1234-1292..⁹⁸ See, e.g., csiftranscode/csiftranscode.h at L254-263.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>a call to the “update_address_book()” function with sender information as the input parameters.</p> <p>The file “initfiles.cpp” implements the “update_address_book()”⁹⁹ where the sender information is used to find its index in the address book by a call to “find_addr_number()”. The return from this call is used to update the array variable “csif_addr[]” by either adding the address if it does not exist indicative of a return value of -1 or by copying the “ipaddr” into a struct member of the indexed variable.</p> <p>The “csif_addr[]” array is an array of type “ADDRESS_ENTRY”¹⁰⁰ which is specified in the file “csiftranscode.h”¹⁰¹ The members of this struct includes “name”, “number”, “group”, “type”, “level” and “ipaddr”. The marking of the “csif_addr[]” variable array constitutes the joining of a group.</p> <p>The server further provides additional functionality in file “server/msgproc.cpp” implements the function “process_net_msg()”¹⁰² where the logging function indicates “rcvd net msg” indicative of receiving a network message. The variables “from” of type character string and “nmsg” of type “NETUSERS” as described above is input to this function. Similar to described above, the “nmsg.requestType” is then branched for cases of “J” and not “J” described to mean of type “R” are handled. For the case of joining a group where “nmsg.requestType” equals “J”, the function “update_address_book()” is called with the sender and receiver information and returns with an index into the address book array “csif_addr” as discussed above. Then the logging command indicates that the “address book [address book index] updated with ipaddr [address book ip address]” and writes “user [name] joined network” to the display. Followed by two calls to “send_net_msg_participants()” using a wildcard designator “-1” indicating everything in the respective field is to be sent. The first call sends everything in the table to the currently added user. The second call sends to everyone the currently added user. In the same file the function “send_net_msg_participants()”¹⁰³ documents that the user of “-1” in the first field indicates a “send to all” and the use of “-1” in the second field indicates a “send all net participants” condition.</p>

⁹⁹ See, e.g., display/initfiles.cpp at L701-721.¹⁰⁰ See, e.g., display/initfiles.h at L 35¹⁰¹ See, e.g., csiftranscode/csiftranscode.h at L45-54.¹⁰² See, e.g., server/msgproc.cpp at L235-293.¹⁰³ See, e.g., server/msgproc.cpp at L108-191.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>Similarly, the server further provides functionality to support the server side function similar to the client side in the file “server/initfiles.cpp” by implementing the function “update_address_book()”¹⁰⁴ where similar to the description above the call to “find_addr_number()” function retrieves an address index. If the address does not exist indicative of a return index value of “-1” calls the function “add_address_book()” otherwise updates the “csif_addr” array with the ip address input to the function. In the same file the function “add_address_book()”¹⁰⁵ is implemented where the “csif_addr” array is populated with the “name”, “number”, “group”, “ipaddr” and a “isValid” member variable.</p>
<p>3. The method of claim 1, wherein the group comprises a plurality of group members permitted to communicate with each other via the communication network.</p>	<p>The AGIS LifeRing product practices “wherein the group comprises a plurality of group members permitted to communicate with each other via the communication network.” The devices are programmed to send and receive messages to and from a server connected to a communication network.</p> <p>The message of claim 1 is transmitted to the first server to join a group by transmission to the first server. The first server provides group management function on a processor executing a group management software code that communicates with a messaging component. The messaging component provides communications function on a processor executing a communication software code that interfaces with the outside world through various links such as Ethernet or Radio Frequency.</p> <p>The file “display/msgproc.cpp” implements the function “process_net_msg()”¹⁰⁶ where the logging function indicates “rcvd net msg” indicative of receiving a network message. The “nmsg” variable of type “NETUSERS” which is input to this function is tested against its member variable requestType which is specified in the file “csiftranscode.h”¹⁰⁷ and states to have “(J)oining or (R)equest current participants”. The “nmsg.requestType” is then branched for cases of “I” for initial joining, “J” for joining, “R” for requesting and “X” for exiting the network. For the case of “J” for joining, the function makes</p>

¹⁰⁴ See, e.g., server/initfiles.cpp at L194-211.

¹⁰⁵ See, e.g., server/initfiles.cpp at L163-191.

¹⁰⁶ See, e.g., display/msgproc.cpp at L1234-1292..

¹⁰⁷ See, e.g., csiftranscode/csiftranscode.h at L254-263.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>a call to the “update_address_book()” function with sender information as the input parameters.</p> <p>The file “initfiles.cpp” implements the “update_address_book()”¹⁰⁸ where the sender information is used to find its index in the address book by a call to “find_addr_number()”. The return from this call is used to update the array variable “csif_addr[]” by either adding the address if it does not exist indicative of a return value of -1 or by copying the “ipaddr” into a struct member of the indexed variable.</p> <p>The “csif_addr[]” array is an array of type “ADDRESS_ENTRY”¹⁰⁹ which is specified in the file “csiftranscode.h”¹¹⁰ The members of this struct includes “name”, “number”, “group”, “type”, “level” and “ipaddr”. The marking of the “csif_addr[]” variable array constitutes the joining of a group.</p> <p>The server further provides additional functionality in file “server/msgproc.cpp” implements the function “process_net_msg()”¹¹¹ where the logging function indicates “rcvd net msg” indicative of receiving a network message. The variables “from” of type character string and “nmsg” of type “NETUSERS” as described above is input to this function. Similar to described above, the “nmsg.requestType” is then branched for cases of “J” and not “J” described to mean of type “R” are handled. For the case of joining a group where “nmsg.requestType” equals “J”, the function “update_address_book()” is called with the sender and receiver information and returns with an index into the address book array “csif_addr” as discussed above. Then the logging command indicates that the “address book [address book index] updated with ipaddr [address book ip address]” and writes “user [name] joined network” to the display. Followed by two calls to “send_net_msg_participants()” using a wildcard designator “-1” indicating everything in the respective field is to be sent. The first call sends everything in the table to the currently added user. The second call sends to everyone the currently added user. In the same file the function “send_net_msg_participants()”¹¹² documents that the user of “-1” in the first field indicates a “send to all” and the use of “-1” in the second field indicates a “send all net participants” condition.</p>

¹⁰⁸ See, e.g., display/initfiles.cpp at L701-721.¹⁰⁹ See, e.g., display/initfiles.h at L 35¹¹⁰ See, e.g., csiftranscode/csiftranscode.h at L45-54.¹¹¹ See, e.g., server/msgproc.cpp at L235-293.¹¹² See, e.g., server/msgproc.cpp at L108-191.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>Similarly, the server further provides functionality to support the server side function similar to the client side in the file “server/initfiles.cpp” by implementing the function “update_address_book()”¹¹³ where similar to the description above the call to “find_addr_number()” function retrieves an address index. If the address does not exist indicative of a return index value of “-1” calls the function “add_address_book()” otherwise updates the “csif_addr” array with the ip address input to the function. In the same file the function “add_address_book()”¹¹⁴ is implemented where the “csif_addr” array is populated with the “name”, “number”, “group”, “ipaddr” and a “isValid” member variable.</p>
<p>4. The method of claim 1, wherein sending the third data to the selected one or more second devices via the first server comprises using an Internet Protocol to send the third data to the first server.</p>	<p>The AGIS LifeRing product practices “wherein sending the third data to the selected one or more second devices via the first server comprises using an Internet Protocol to send the third data to the first server.” The devices are programmed to send and receive messages to and from a server connected to an Internet Protocol communication network.</p> <p>The message of claim 1 is transmitted to the first server to join a group by transmission to the first server. The first server provides group management function on a processor executing a group management software code that communicates with a messaging component. The messaging component provides communications function on a processor executing a communication software code that interfaces with the outside world through various links such as Ethernet or Radio Frequency.</p> <p>The file “display/msgproc.cpp” implements the function “process_net_msg()”¹¹⁵ where the logging function indicates “rcvd net msg” indicative of receiving a network message. The “nmsg” variable of type “NETUSERS” which is input to this function is tested against its member variable requestType which is specified in the file “csiftranscode.h”¹¹⁶ and states to have “(J)oining or (R)equest current participants”. The “nmsg.requestType” is then branched for cases of “I” for initial joining, “J” for joining, “R” for requesting and “X” for exiting the network. For the case of “J” for joining, the function makes</p>

¹¹³ See, e.g., server/initfiles.cpp at L194-211.

¹¹⁴ See, e.g., server/initfiles.cpp at L163-191.

¹¹⁵ See, e.g., display/msgproc.cpp at L1234-1292..

¹¹⁶ See, e.g., csiftranscode/csiftranscode.h at L254-263.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>a call to the “update_address_book()” function with sender information as the input parameters.</p> <p>The file “initfiles.cpp” implements the “update_address_book()”¹¹⁷ where the sender information is used to find its index in the address book by a call to “find_addr_number()”. The return from this call is used to update the array variable “csif_addr[]” by either adding the address if it does not exist indicative of a return value of -1 or by copying the “ipaddr” into a struct member of the indexed variable.</p> <p>The “csif_addr[]” array is an array of type “ADDRESS_ENTRY”¹¹⁸ which is specified in the file “csiftranscode.h”¹¹⁹ The members of this struct includes “name”, “number”, “group”, “type”, “level” and “ipaddr”. The marking of the “csif_addr[]” variable array constitutes the joining of a group.</p> <p>The server further provides additional functionality in file “server/msgproc.cpp” implements the function “process_net_msg()”¹²⁰ where the logging function indicates “rcvd net msg” indicative of receiving a network message. The variables “from” of type character string and “nmsg” of type “NETUSERS” as described above is input to this function. Similar to described above, the “nmsg.requestType” is then branched for cases of “J” and not “J” described to mean of type “R” are handled. For the case of joining a group where “nmsg.requestType” equals “J”, the function “update_address_book()” is called with the sender and receiver information and returns with an index into the address book array “csif_addr” as discussed above. Then the logging command indicates that the “address book [address book index] updated with ipaddr [address book ip address]” and writes “user [name] joined network” to the display. Followed by two calls to “send_net_msg_participants()” using a wildcard designator “-1” indicating everything in the respective field is to be sent. The first call sends everything in the table to the currently added user. The second call sends to everyone the currently added user. In the same file the function “send_net_msg_participants()”¹²¹ documents that the user of “-1” in the first field indicates a “send to all” and the use of “-1” in the second field indicates a “send all net participants” condition.</p>

¹¹⁷ See, e.g., display/initfiles.cpp at L701-721.

¹¹⁸ See, e.g., display/initfiles.h at L 35

¹¹⁹ See, e.g., csiftranscode/csiftranscode.h at L45-54.

¹²⁰ See, e.g., server/msgproc.cpp at L235-293.

¹²¹ See, e.g., server/msgproc.cpp at L108-191.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>Similarly, the server further provides functionality to support the server side function similar to the client side in the file “server/initfiles.cpp” by implementing the function “update_address_book()”¹²² where similar to the description above the call to “find_addr_number()” function retrieves an address index. If the address does not exist indicative of a return index value of “-1” calls the function “add_address_book()” otherwise updates the “csif_addr” array with the ip address input to the function. In the same file the function “add_address_book()”¹²³ is implemented where the “csif_addr” array is populated with the “name”, “number”, “group”, “ipaddr” and a “isValid” member variable.</p> <p>In the file “display/msgproc.cpp” the functions “send_message_bulk()” and “send_message_direct()”¹²⁴ are implemented where both have IP transmit capability with the addition of “send_message_direct()” also having SMS transmission capability. Within the IP transmit capability, both functions make calls to the “sendTCPMessage()” function. In the file “csif.cpp” the “sendTCPMessage()”¹²⁵ function is implemented where it utilizes the “tcpConnection” variable to call the “enqueueForSending()” function. The “tcpConnection” variable holds the necessary information to complete an IP based transmission. In the file “csif.cpp” the function “initTCP()”¹²⁶ is implemented where the “tcpConnection” variable is instantiated. In this function a network management object “nm” from the “NetworkMgmt” class is used to call the “connectIPNetwork()” function and implement an “ipMonitor” variable to monitor the IP network. Also the “tcpConnection” variable is set to a new object by a call to “TCPServerConnection()” function.</p>
5. The method of claim 4, wherein the first device does not have access to respective Internet Protocol addresses of the one or more second devices	The AGIS LifeRing product practices “wherein the first device does not have access to respective Internet Protocol addresses of the one or more second devices included in the group.” The devices are programmed to send and receive messages to and from a server connected to an Internet Protocol communication network where devices are not aware of each other addresses.

¹²² See, e.g., server/initfiles.cpp at L194-211.¹²³ See, e.g., server/initfiles.cpp at L163-191.¹²⁴ See, e.g., display/msgproc.cpp at L47-127.¹²⁵ See, e.g., csif.cpp at L378-386.¹²⁶ See, e.g., csif.cpp at L150-204.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
included in the group.	<p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”¹²⁷ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”¹²⁸ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”¹²⁹ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”¹³⁰ and “num_on_que_list()”¹³¹ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”¹³² is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function</p>

¹²⁷ See, e.g., buttons.cpp at L1344-2365.

¹²⁸ See, e.g., display/ftext.cpp at L176-244.

¹²⁹ See, e.g., netselect.cpp at L152-221.

¹³⁰ See, e.g., quelist.cpp at L16-22.

¹³¹ See, e.g., quelist.cpp at L60-71.

¹³² See, e.g., display/msgproc.cpp at L328-399.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“create_netlist()”¹³³ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”¹³⁴ and subsequently to “form_net_line()”¹³⁵ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”¹³⁶ and subsequently “send_net_msg_participants()”¹³⁷ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”¹³⁸ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”¹³⁹ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”¹⁴⁰ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
6. The method of claim 1, wherein the third data include a	The AGIS LifeRing product practices “wherein the third data include a short message service message, a text message, an image, or a video.”

¹³³ See, e.g., netselect.cpp at L391-492.¹³⁴ See, e.g., netselect.cpp at L373-384.¹³⁵ See, e.g., netselect.cpp at L352-370.¹³⁶ See, e.g., display/msgproc.cpp at L1234-1292.¹³⁷ See, e.g., display/msgproc.cpp at L828-922.¹³⁸ See, e.g., photo.cpp at L189-292.¹³⁹ See, e.g., server/msgproc.cpp at L377-474.¹⁴⁰ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
short message service message, a text message, an image, or a video.	<p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”¹⁴¹ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”¹⁴² is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”¹⁴³ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”¹⁴⁴ and “num_on_que_list()”¹⁴⁵ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”¹⁴⁶ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function</p>

¹⁴¹ See, e.g., buttons.cpp at L1344-2365.¹⁴² See, e.g., display/ftext.cpp at L176-244.¹⁴³ See, e.g., netselect.cpp at L152-221.¹⁴⁴ See, e.g., quelist.cpp at L16-22.¹⁴⁵ See, e.g., quelist.cpp at L60-71.¹⁴⁶ See, e.g., display/msgproc.cpp at L328-399.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“create_netlist()”¹⁴⁷ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”¹⁴⁸ and subsequently to “form_net_line()”¹⁴⁹ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”¹⁵⁰ and subsequently “send_net_msg_participants()”¹⁵¹ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”¹⁵² is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”¹⁵³ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”¹⁵⁴ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
7. The method of claim 6, wherein the	The AGIS LifeRing product practices “wherein the video comprise a video clip.” The device utilizes buttons on the display to send messages, make calls and send photo or video.

¹⁴⁷ See, e.g., netselect.cpp at L391-492.¹⁴⁸ See, e.g., netselect.cpp at L373-384.¹⁴⁹ See, e.g., netselect.cpp at L352-370.¹⁵⁰ See, e.g., display/msgproc.cpp at L1234-1292.¹⁵¹ See, e.g., display/msgproc.cpp at L828-922.¹⁵² See, e.g., photo.cpp at L189-292.¹⁵³ See, e.g., server/msgproc.cpp at L377-474.¹⁵⁴ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
video comprises a video clip.	<p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”¹⁵⁵ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”¹⁵⁶ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”¹⁵⁷ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”¹⁵⁸ and “num_on_que_list()”¹⁵⁹ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”¹⁶⁰ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”¹⁶¹ is implemented where the display is configured for</p>

¹⁵⁵ See, e.g., buttons.cpp at L1344-2365.¹⁵⁶ See, e.g., display/ftext.cpp at L176-244.¹⁵⁷ See, e.g., netselect.cpp at L152-221.¹⁵⁸ See, e.g., quelist.cpp at L16-22.¹⁵⁹ See, e.g., quelist.cpp at L60-71.¹⁶⁰ See, e.g., display/msgproc.cpp at L328-399.¹⁶¹ See, e.g., netselect.cpp at L391-492.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>net communication. The call to “send_to_display_net_list()”¹⁶² and subsequently to “form_net_line()”¹⁶³ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”¹⁶⁴ and subsequently “send_net_msg_participants()”¹⁶⁵ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”¹⁶⁶ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”¹⁶⁷ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”¹⁶⁸ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
8. The method of claim 1, wherein the third data include a voice recording.	<p>The AGIS LifeRing product practices “wherein the third data include a voice recording.”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p>

¹⁶² See, e.g., netselect.cpp at L373-384.¹⁶³ See, e.g., netselect.cpp at L352-370.¹⁶⁴ See, e.g., display/msgproc.cpp at L1234-1292.¹⁶⁵ See, e.g., display/msgproc.cpp at L828-922.¹⁶⁶ See, e.g., photo.cpp at L189-292.¹⁶⁷ See, e.g., server/msgproc.cpp at L377-474.¹⁶⁸ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”¹⁶⁹ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”¹⁷⁰ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”¹⁷¹ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”¹⁷² and “num_on_que_list()”¹⁷³ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”¹⁷⁴ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”¹⁷⁵ is implemented where the display is configured for</p>

¹⁶⁹ See, e.g., buttons.cpp at L1344-2365.¹⁷⁰ See, e.g., display/ftext.cpp at L176-244.¹⁷¹ See, e.g., netselect.cpp at L152-221.¹⁷² See, e.g., quelist.cpp at L16-22.¹⁷³ See, e.g., quelist.cpp at L60-71.¹⁷⁴ See, e.g., display/msgproc.cpp at L328-399.¹⁷⁵ See, e.g., netselect.cpp at L391-492.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>net communication. The call to “send_to_display_net_list()”¹⁷⁶ and subsequently to “form_net_line()”¹⁷⁷ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”¹⁷⁸ and subsequently “send_net_msg_participants()”¹⁷⁹ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”¹⁸⁰ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”¹⁸¹ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”¹⁸² is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p> <p>Once the devices have formed a group they share their location with each others. The AGIS software captures location information on the device through a serial communication link configured to continuously process GPS information messages. Messages such as location, precision and constellation are captured on an independent thread and</p>

¹⁷⁶ See, e.g., netselect.cpp at L373-384.

¹⁷⁷ See, e.g., netselect.cpp at L352-370.

¹⁷⁸ See, e.g., display/msgproc.cpp at L1234-1292.

¹⁷⁹ See, e.g., display/msgproc.cpp at L828-922.

¹⁸⁰ See, e.g., photo.cpp at L189-292.

¹⁸¹ See, e.g., server/msgproc.cpp at L377-474.

¹⁸² See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>when a valid message is received the thread triggers an event indicating the presence of a new message. The software then sends the new location to others through either a server or direct connection.</p> <p>The “csif” module provides a service to start a GPSThread that communicates with a serial port. In the file “csif/csif.cpp” the function “initGPS()”¹⁸³ is implemented where the variable comport is as the argument. The variable “gpsListener” is then assigned to a new GPSThread class using the comport variable followed by calling the “GPSThread->start()” call.</p> <p>In the file “gpslistener.cpp” the “start()”¹⁸⁴ method is implemented where a thread is created for processing in the background. The thread performs the function “process()”¹⁸⁵ and uses a loop to scan the comport documented as “read in a message from serial port, will only wait for 1 second for data”¹⁸⁶ and once a message is detected that it calls the “processGPSMessage()” passing the message and length of the message. The function “processGPSMessage()”¹⁸⁷ calls the “enqueueGPSMsg()” of the “DDLDatabase” class where the GPS message is queued.</p> <p>In the file “display.cpp” inside the initialization function “InitInstance()”¹⁸⁸ the GPS message event processor is registered through the call to the “registerGPSThread()” much like the DDL messages. The GPS messages are then processed by the callback event capture routine “WndProc()”¹⁸⁹ where the “ID_TIMER_BLINK” event documented to represent a 1 second timer makes a call to “check_gps_msgs()”, “check_for_internal_msgs()” and “check_for_db_msgs()”.</p> <p>In the file “gps.cpp” the function “check_gps_msgs()”¹⁹⁰ is implemented where the queued GPS messages are process incrementally with a call to the “process_csif_gps_msg()”. The function</p>

¹⁸³ See, e.g., csif/csif.cpp at L126-148.

¹⁸⁴ See, e.g., csif/gpslistener.cpp at L36-51.

¹⁸⁵ See, e.g., csif/gpslistener.cpp at L101-235.

¹⁸⁶ See, e.g., csif/gpslistener.cpp at L158.

¹⁸⁷ See, e.g., csif/gpslistener.cpp at L76-99.

¹⁸⁸ See, e.g., display/display.cpp at L274-581.

¹⁸⁹ See, e.g., display/display.cpp at L584-969.

¹⁹⁰ See, e.g., display/gps.cpp at L346-365.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“process_csif_gps_msg()”¹⁹¹ is implemented where the message is retrieved by a call to “get_gps_msg()” and based on one of three choices of a message type captured by the “msgType” member of the message variable “gmsg” that the position, precision or constellation of the gps message is processed. For the case where the “msgType” is of “GPS_MSG_POSITION” type the call is made to “process_gps_pos()” passing the “gmsg.body.position” variable member. In the same file the function “process_gps_pos()”¹⁹² is implemented. This function utilizes the “gps_pos_decode()” function to covert the input message’s members such as “gpsp.latitude” and “gpsp.longitude” to a floating point format in the “lat” and “lon” variables. Further into the function, the “lat” and “lon” variables are used to populate the members of the “temptrk” struct variable which is of type “track_file_struct”.</p> <p>In the file “display/display.cpp” the function “check_ip_address()”¹⁹³ is an example of communication with transmission of location information. This function is conditioned to use certain code for the case where the “PPC2003” is defined, indicative of SMS capability, or not defined. In both routines similar functionality is performed where calls to functions “send_net_msg_netmgmt()” to join a network group and “SendOwnPosString()” to send location information under various connection states.</p> <p>In the file “display/buttons.cpp” in the main routine where it is documented that button actions are processed by “button_actions”¹⁹⁴ and are implemented in the function “button_actions()”¹⁹⁵ the case for pressing the “BUTTON_PING:” is implemented where calls are made to “SendOwnPosString()” and “send_all_tracks()”.</p> <p>In the file “display/msgproc.cpp” the function “SendOwnPosString()”¹⁹⁶ is implemented where the “op” struct variable of type “OWN_POSITION” is populated with location information such as latitude and longitude. Then the “dmsg” variable of type “DDLMSG” is set to have a “msgType” of “DDL_MSG_OWN_POSITION” and variable “op” set to one of its other member variables. Then the “dmsg” is set to a buffer with the call</p>

¹⁹¹ See, e.g., display/gps.cpp at L292-336.

¹⁹² See, e.g., display/gps.cpp at L77-126.

¹⁹³ See, e.g., display/display.cpp at L1015-1037 and L1076-1115.

¹⁹⁴ See, e.g., display/buttons.cpp at L594-598 and L1344-1346.

¹⁹⁵ See, e.g., display/buttons.cpp at L1344-2365.

¹⁹⁶ See, e.g., display/msgproc.cpp at L170-325.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>to “put_ddl_msg()” and a call to “send_message_bulk()” when the transmit method “xmit_method” is set to “SERVER” and using a “server_ip” variable. Alternatively, direct transmission may occur with the call to “send_message_direct()”</p> <p>Similarly, in the file “display/msgproc.cpp” the function “send_track_msg()”¹⁹⁷ is implemented which includes location information along with previous location information. Similar to the previous function, the “dmsg” is set to type “DDL_MSG_TRACK_REPORT” and “msg” variable with location information is set to its member variable. The functions “put_ddl_msg()” followed by “send_message_bulk()” or “send_message_direct()” transmit the message.</p> <p>The “send_track_msg()” function is used under the condition where “add_track()”¹⁹⁸ and “update_track_id()”¹⁹⁹ functions in the “display/track.cpp” is invoked. The “update_track_id()” is used in user interface as views in the “buttons.cpp” file in the main routine where button actions are implemented “button_actions()” as discussed above where the case for “BUTTON_TRACK_ID_UNKN”, “BUTTON_TRACK_ID_FRND” and “BUTTON_TRACK_ID_HOSTILE” cases are handled for various features including the call to “update_track_id()”.</p> <p>On the server side, the file “server/msgproc.cpp” implements the function “process_csif_ddl_msg()”²⁰⁰ where the incoming message uses the “msgType” to branch to a section of the code depicted to represent the “DDL_MSG_OWN_POISITION” case where the message is passed through by a call to “pass_thru_msg()” function.</p> <p>The file “server/server.cpp” implements the main function “WndProc()”²⁰¹ as a callback where a receipt of a message triggers its execution. The case of “CSIF_DDL_MSG_AVAILABLE” is used to call the “getNextDDLMessage()” with a variable “route” that is used to call the “process_csif_ddl_msg()” function as described above to make the call for pass through.</p>

¹⁹⁷ See, e.g., display/msgproc.cpp at L615-682.

¹⁹⁸ See, e.g., display/track.cpp at L373-477.

¹⁹⁹ See, e.g., display/track.cpp at L482-510.

²⁰⁰ See, e.g., server/msgproc.cpp at L377-474.

²⁰¹ See, e.g., server/server.cpp at L228-386.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
9. The method of claim 1, wherein sending the third data to the selected one or more second devices comprises transmitting a text message to at least one of the selected one or more second devices using an Internet Protocol (IP).	<p>The AGIS LifeRing product practices “wherein sending the third data to the selected one or more second devices comprises transmitting a text message to at least one of the selected one or more second devices using an Internet Protocol (IP).”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”²⁰² implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”²⁰³ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”²⁰⁴ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”²⁰⁵ and “num_on_que_list()”²⁰⁶ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”²⁰⁷ is implemented where the free text message is transmitted directly or through a server as described above through the use of the</p>

²⁰² See, e.g., buttons.cpp at L1344-2365.

²⁰³ See, e.g., display/ftext.cpp at L176-244.

²⁰⁴ See, e.g., netselect.cpp at L152-221.

²⁰⁵ See, e.g., quelist.cpp at L16-22.

²⁰⁶ See, e.g., quelist.cpp at L60-71.

²⁰⁷ See, e.g., display/msgproc.cpp at L328-399.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netsselect.cpp” the function “create_netlist()”²⁰⁸ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”²⁰⁹ and subsequently to “form_net_line()”²¹⁰ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”²¹¹ and subsequently “send_net_msg_participants()”²¹² the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”²¹³ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”²¹⁴ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”²¹⁵ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p>

²⁰⁸ See, e.g., netsselect.cpp at L391-492.

²⁰⁹ See, e.g., netsselect.cpp at L373-384.

²¹⁰ See, e.g., netsselect.cpp at L352-370.

²¹¹ See, e.g., display/msgproc.cpp at L1234-1292.

²¹² See, e.g., display/msgproc.cpp at L828-922.

²¹³ See, e.g., photo.cpp at L189-292.

²¹⁴ See, e.g., server/msgproc.cpp at L377-474.

²¹⁵ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.
10. The method of claim 1, further comprising performing by the first device:	
identifying user interaction with the interactive display selecting at least one of the second set of user-selectable symbols corresponding to at least one of the second devices and user interaction with the display specifying an action and, based thereon, initiating a phone call or phone conference with the at least one second device.	<p>The AGIS LifeRing product practices “identifying user interaction with the interactive display selecting at least one of the second set of user-selectable symbols corresponding to at least one of the second devices and user interaction with the display specifying an action and, based thereon, initiating a phone call or phone conference with the at least one second device.”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”²¹⁶ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”²¹⁷ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”²¹⁸ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of</p>

²¹⁶ See, e.g., buttons.cpp at L1344-2365.²¹⁷ See, e.g., display/ftext.cpp at L176-244.²¹⁸ See, e.g., netselect.cpp at L152-221.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”²¹⁹ and “num_on_que_list()”²²⁰ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”²²¹ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”²²² is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”²²³ and subsequently to “form_net_line()”²²⁴ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”²²⁵ and subsequently “send_net_msg_participants()”²²⁶ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”²²⁷ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_dll_msg()”²²⁸ is implemented as described above with</p>

²¹⁹ See, e.g., *quelist.cpp* at L16-22.

²²⁰ See, e.g., *quelist.cpp* at L60-71.

²²¹ See, e.g., *display/msgproc.cpp* at L328-399.

²²² See, e.g., *netselect.cpp* at L391-492.

²²³ See, e.g., *netselect.cpp* at L373-384.

²²⁴ See, e.g., *netselect.cpp* at L352-370.

²²⁵ See, e.g., *display/msgproc.cpp* at L1234-1292.

²²⁶ See, e.g., *display/msgproc.cpp* at L828-922.

²²⁷ See, e.g., *photo.cpp* at L189-292.

²²⁸ See, e.g., *server/msgproc.cpp* at L377-474.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”²²⁹ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
<p>11. The method of claim 1, further comprising performing by the first device: identifying user interaction with the interactive display selecting a particular user-selectable symbol corresponding to a particular second device and user interaction with the display specifying an action and, based thereon, initiating voice-over-IP (VOIP) communication with the particular second device.</p>	<p>The AGIS LifeRing product practices “performing, by the first device: identifying user interaction with the interactive display selecting a particular user-selectable symbol corresponding to a particular second device and user interaction with the display specifying an action and, based thereon, initiating voice-over-IP (VOIP) communication with the particular second device.”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”²³⁰ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”²³¹ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the</p>

²²⁹ See, e.g., server/msgproc.cpp at L 51-61.

²³⁰ See, e.g., buttons.cpp at L1344-2365.

²³¹ See, e.g., display/ftext.cpp at L176-244.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“build_sending_text()”²³² function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”²³³ and “num_on_que_list()”²³⁴ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”²³⁵ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netsselect.cpp” the function “create_netlist()”²³⁶ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”²³⁷ and subsequently to “form_net_line()”²³⁸ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”²³⁹ and subsequently “send_net_msg_participants()”²⁴⁰ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”²⁴¹ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p>

²³² See, e.g., netsselect.cpp at L152-221.

²³³ See, e.g., quelist.cpp at L16-22.

²³⁴ See, e.g., quelist.cpp at L60-71.

²³⁵ See, e.g., display/msgproc.cpp at L328-399.

²³⁶ See, e.g., netsselect.cpp at L391-492.

²³⁷ See, e.g., netsselect.cpp at L373-384.

²³⁸ See, e.g., netsselect.cpp at L352-370.

²³⁹ See, e.g., display/msgproc.cpp at L1234-1292.

²⁴⁰ See, e.g., display/msgproc.cpp at L828-922.

²⁴¹ See, e.g., photo.cpp at L189-292.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>In the file “server/msgproc.cpp” the function “process_csif_dll_msg()”²⁴² is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”²⁴³ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
<p>12. The method of claim 1, further comprising performing by the first device: identifying user interaction with the interactive display selecting a particular user-selectable symbol corresponding to a particular second device and user interaction with the display specifying an action and, based thereon, initiating a data call with the particular second device.</p>	<p>The AGIS LifeRing product practices “identifying user interaction with the interactive display selecting a particular user-selectable symbol corresponding to a particular second device and user interaction with the display specifying an action and, based thereon, initiating a data call with the particular second device.”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”²⁴⁴ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p>

²⁴² See, e.g., server/msgproc.cpp at L377-474.

²⁴³ See, e.g., server/msgproc.cpp at L 51-61.

²⁴⁴ See, e.g., buttons.cpp at L1344-2365.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>In the file “display/ftext.cpp” the function “create_ftext()”²⁴⁵ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”²⁴⁶ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”²⁴⁷ and “num_on_que_list()”²⁴⁸ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”²⁴⁹ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”²⁵⁰ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”²⁵¹ and subsequently to “form_net_line()”²⁵² forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”²⁵³ and subsequently “send_net_msg_participants()”²⁵⁴ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”²⁵⁵ is implemented where the</p>

²⁴⁵ See, e.g., display/ftext.cpp at L176-244.

²⁴⁶ See, e.g., netselect.cpp at L152-221.

²⁴⁷ See, e.g., quelist.cpp at L16-22.

²⁴⁸ See, e.g., quelist.cpp at L60-71.

²⁴⁹ See, e.g., display/msgproc.cpp at L328-399.

²⁵⁰ See, e.g., netselect.cpp at L391-492.

²⁵¹ See, e.g., netselect.cpp at L373-384.

²⁵² See, e.g., netselect.cpp at L352-370.

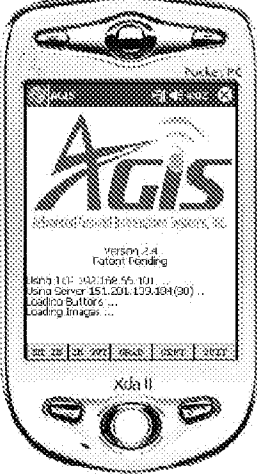
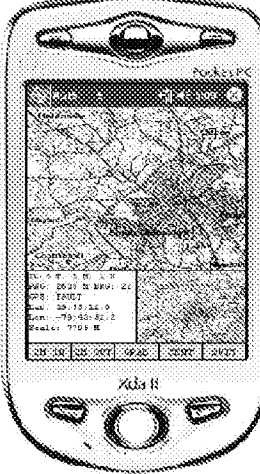
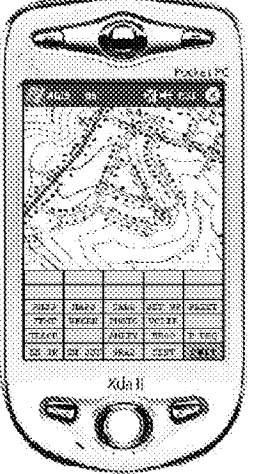
²⁵³ See, e.g., display/msgproc.cpp at L1234-1292.

²⁵⁴ See, e.g., display/msgproc.cpp at L828-922.

²⁵⁵ See, e.g., photo.cpp at L189-292.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_dll_msg()”²⁵⁶ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”²⁵⁷ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
13. The method of claim 1, wherein the first device is a personal digital assistant (PDA) or a personal computer (PC).	<p>The AGIS LifeRing product includes systems and methods for executing an application on PDA/cell phones, now referred to as a smartphone devices.</p> <p>See, e.g., “call.cpp” references to “phone” and “other phones.”</p>
14. The method of claim 1, wherein the first device is a smart phone.	<p>The AGIS LifeRing product includes systems and methods for executing an application on PDA/cell phones, now referred to as a smartphone devices.</p> <p>See, e.g., “call.cpp” references to “phone” and “other phones.”</p>
15. The method of claim 1, wherein the display of the first device is a touch screen display, and wherein the user interaction with the display selecting the	<p>The AGIS LifeRing product practices “wherein the display of the first device is a touch screen display, and wherein the user interaction with the display selecting the one or more user-selectable symbols in the second set of symbols comprises touching the one or more user-selectable symbols in the second set of symbols.”</p> <p>As depicted above, the user client of the AGIS software includes a user interface display with a map overlaid with symbols representing other</p>

²⁵⁶ See, e.g., server/msgproc.cpp at L377-474.²⁵⁷ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p>one or more user-selectable symbols in the second set of symbols comprises touching the one or more user-selectable symbols in the second set of symbols.</p>	<p>members of a group at their geographic locations on the georeferenced map. For example, the “operator selects the TRACK SoftSwitch and touches the point on the map display where he sees the tank is located and selects the HOST, ARMR, and MOVE SoftSwitches.” Exhibit 4 at 5.</p> <p>The AGIS application software does not affect the operation of the PDA, the Cell phone or the GPS until it is activated. When the AGIS application software is selected, the AGIS PDA Cell phone is turned on, the GPS is automatically connected to the PDA / Cell phone through a Bluetooth interface and the AGIS Logo appears. Shortly thereafter the AGIS operation display appears. See Figures 1, 2, and 3</p> <div style="display: flex; justify-content: space-around; align-items: center;">    </div> <p style="text-align: center;"> Figure 1 Figure 2 Figure 3 </p> <p>Exhibit 4 to Beyer Declaration, at 4.</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”²⁵⁸ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are</p>

²⁵⁸ See, e.g., buttons.cpp at l1344-2365.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”²⁵⁹ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”²⁶⁰ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”²⁶¹ and “num_on_que_list()”²⁶² functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”²⁶³ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”²⁶⁴ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”²⁶⁵ and subsequently to “form_net_line()”²⁶⁶ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”²⁶⁷ and subsequently “send_net_msg_participants()”²⁶⁸ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p>

²⁵⁹ See, e.g., display/ftext.cpp at L176-244.

²⁶⁰ See, e.g., netselect.cpp at L152-221.

²⁶¹ See, e.g., quelist.cpp at L16-22.

²⁶² See, e.g., quelist.cpp at L60-71.

²⁶³ See, e.g., display/msgproc.cpp at L328-399.

²⁶⁴ See, e.g., netselect.cpp at L391-492.

²⁶⁵ See, e.g., netselect.cpp at L373-384.

²⁶⁶ See, e.g., netselect.cpp at L352-370.

²⁶⁷ See, e.g., display/msgproc.cpp at L1234-1292.

²⁶⁸ See, e.g., display/msgproc.cpp at L828-922.

<p>U.S. Patent No. 9,467,838</p>	<p>October 22, 2005 AGIS LifeRing</p>
	<p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”²⁶⁹ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_dll_msg()”²⁷⁰ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”²⁷¹ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
<p>16. The method of claim 1, further comprising performing by the first device: sending updated location information comprising an updated location of the first device, the updated location information being sent based on passage of a predetermined time interval since sending previous location information</p>	<p>The AGIS LifeRing product practices “sending updated location information comprising an updated location of the first device, the updated location information being sent based on passage of a predetermined time interval since sending previous location information comprising a previous location of the first device, displacement of the first device by a predetermined distance relative to a previous location of the first device, or both.”</p> <p>The AGIS software captures location information on the device through a serial communication link configured to continuously process GPS information messages. Messages such as location, precision and constellation are captured on an independent thread and when a valid message is received the thread triggers an event indicating the presence of a new message. The software then sends the new location to others through either a server or direct connection.</p> <p>The “csif” module provides a service to start a GPSThread that communicates with a serial port. In the file “csif/csif.cpp” the function</p>

²⁶⁹ See, e.g., photo.cpp at L189-292.

²⁷⁰ See, e.g., server/msgproc.cpp at L377-474.

²⁷¹ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p>comprising a previous location of the first device, displacement of the first device by a predetermined distance relative to a previous location of the first device, or both.</p>	<p>“initGPS()”²⁷² is implemented where the variable comport is as the argument. The variable “gpsListener” is then assigned to a new GPSTListener class using the comport variable followed by calling the “GPSTListener->start()” call.</p> <p>In the file “gpslistener.cpp” the “start()”²⁷³ method is implemented where a thread is created for processing in the background. The thread performs the function “process()”²⁷⁴ and uses a loop to scan the comport documented as “read in a message from serial port, will only wait for 1 second for data”²⁷⁵ and once a message is detected that it calls the “processGPSMessage()” passing the message and length of the message. The function “processGPSMessage()”²⁷⁶ calls the “enqueueGPSMsg()” of the “DDLDatabase” class where the GPS message is queued.</p> <p>In the file “display.cpp” inside the initialization function “InitInstance()”²⁷⁷ the GPS message event processor is registered through the call to the “registerGPSTListener()” much like the DDL messages. The GPS messages are then processed by the callback event capture routine “WndProc()”²⁷⁸ where the “ID_TIMER_BLINK” event documented to represent a 1 second timer makes a call to “check_gps_msgs()”, “check_for_internal_msgs()” and “check_for_db_msgs()”.</p> <p>In the file “gps.cpp” the function “check_gps_msgs()”²⁷⁹ is implemented where the queued GPS messages are process incrementally with a call to the “process_csif_gps_msg()”. The function “process_csif_gps_msg()”²⁸⁰ is implemented where the message is retrieved by a call to “get_gps_msg()” and based on one of three choices of a message type captured by the “msgType” member of the message variable “gmsg” that the position, precision or constellation of the gps message is processed. For the case where the “msgType” is of “GPS_MSG_POSITION” type the call is made to “process_gps_pos()”</p>

²⁷² See, e.g., csif/csif.cpp at L126-148.

²⁷³ See, e.g., csif/gpslistener.cpp at L36-51.

²⁷⁴ See, e.g., csif/gpslistener.cpp at L101-235.

²⁷⁵ See, e.g., csif/gpslistener.cpp at L158.

²⁷⁶ See, e.g., csif/gpslistener.cpp at L76-99.

²⁷⁷ See, e.g., display/display.cpp at L274-581.

²⁷⁸ See, e.g., display/display.cpp at L584-969.

²⁷⁹ See, e.g., display/gps.cpp at L346-365.

²⁸⁰ See, e.g., display/gps.cpp at L292-336.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>passing the “gmsg.body.position” variable member. In the same file the function “process_gps_pos()”²⁸¹ is implemented. This function utilizes the “gps_pos_decode()” function to covert the input message’s members such as “gpsp.latitude” and “gpsp.longitude” to a floating point format in the “lat” and “lon” variables. Further into the function, the “lat” and “lon” variables are used to populate the members of the “temptrk” struct variable which is of type “track_file_struct”.</p> <p>In the file “display/display.cpp” the function “check_ip_address()”²⁸² is an example of communication with transmission of location information. This function is conditioned to use certain code for the case where the “PPC2003” is defined, indicative of SMS capability, or not defined. In both routines similar functionality is performed where calls to functions “send_net_msg_netmgmt()” to join a network group and “SendOwnPosString()” to send location information under various connection states.</p> <p>In the file “display/buttons.cpp” in the main routine where it is documented that button actions are processed by “button_actions”²⁸³ and are implemented in the function “button_actions()”²⁸⁴ the case for pressing the “BUTTON_PING:” is implemented where calls are made to “SendOwnPosString()” and “send_all_tracks()”.</p> <p>In the file “display/msgproc.cpp” the function “SendOwnPosString()”²⁸⁵ is implemented where the “op” struct variable of type “OWN_POSITION” is populated with location information such as latitude and longitude. Then the “dmsg” variable of type “DDLMSG” is set to have a “msgType” of “DDL_MSG_OWN_POSITION” and variable “op” set to one of its other member variables. Then the “dmsg” is set to a buffer with the call to “put_ddl_msg()” and a call to “send_message_bulk()” when the transmit method “xmit_method” is set to “SERVER” and using a “server_ip” variable. Alternatively, direct transmission may occur with the call to “send_message_direct()”</p>

²⁸¹ See, e.g., display/gps.cpp at L77-126.

²⁸² See, e.g., display/display.cpp at L1015-1037 and L1076-1115.

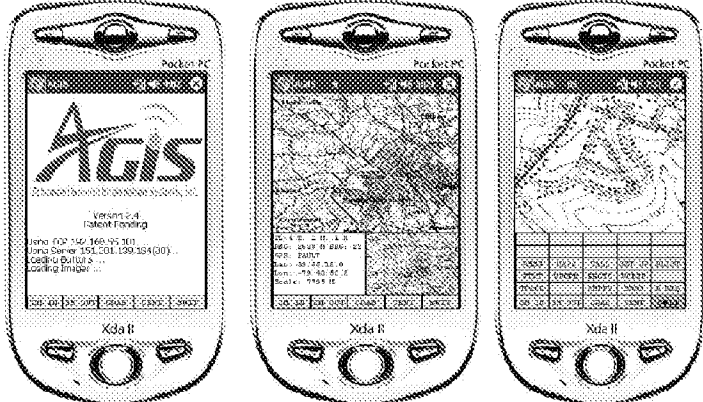
²⁸³ See, e.g., display/buttons.cpp at L594-598 and L1344-1346.

²⁸⁴ See, e.g., display/buttons.cpp at L1344-2365.

²⁸⁵ See, e.g., display/msgproc.cpp at L170-325.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>Similarly, in the file “display/msgproc.cpp” the function “send_track_msg()”²⁸⁶ is implemented which includes location information along with previous location information. Similar to the previous function, the “dmsg” is set to type “DDL_MSG_TRACK_REPORT” and “tmsg” variable with location information is set to its member variable. The functions “put_ddl_msg()” followed by “send_message_bulk()” or “send_message_direct()” transmit the message.</p> <p>The “send_track_msg()” function is used under the condition where “add_track()”²⁸⁷ and “update_track_id()”²⁸⁸ functions in the “display/track.cpp” is invoked. The “update_track_id()” is used in user interface as views in the “buttons.cpp” file in the main routine where button actions are implemented “button_actions()” as discussed above where the case for “BUTTON_TRACK_ID_UNKN”, “BUTTON_TRACK_ID_FRND” and “BUTTON_TRACK_ID_HOSTILE” cases are handled for various features including the call to “update_track_id()”.</p> <p>On the server side, the file “server/msgproc.cpp” implements the function “process_csif_ddl_msg()”²⁸⁹ where the incoming message uses the “msgType” to branch to a section of the code depicted to represent the “DDL_MSG_OWN_POISITION” case where the message is passed through by a call to “pass_thru_msg()” function.</p> <p>The file “server/server.cpp” implements the main function “WndProc()”²⁹⁰ as a callback where a receipt of a message triggers its execution. The case of “CSIF_DDL_MSG_AVAILABLE” is used to call the “getNextDDLMessage()” with a variable “route” that is used to call the “process_csif_ddl_msg()” function as described above to make the call for pass through.</p>
17. The method of claim 1, further comprising performing by the first device:	<p>The AGIS LifeRing product practices the computer implemented method of claim 1.</p> <p>The AGIS LifeRing product includes systems and methods for executing an application on PDA/cell phones, now referred to as a smartphone devices.</p>

²⁸⁶ See, e.g., display/msgproc.cpp at L615-682.²⁸⁷ See, e.g., display/track.cpp at L373-477.²⁸⁸ See, e.g., display/track.cpp at L482-510.²⁸⁹ See, e.g., server/msgproc.cpp at L377-474.²⁹⁰ See, e.g., server/server.cpp at L228-386.

<p>U.S. Patent No. 9,467,838</p>	<p>October 22, 2005 AGIS LifeRing</p>
	<p>See, e.g., “call.cpp” references to “phone” and “other phones.”</p> <p>The operator can then send voice conference or digital data to those AGIS equipped units by simply selecting their assigned NET SoftSwitch.</p> <p>The AGIS application software does not affect the operation of the PDA, the Cell phone or the GPS until it is activated. When the AGIS application software is selected, the AGIS PDA Cell phone is turned on, the GPS is automatically connected to the PDA / Cell phone through a Bluetooth interface and the AGIS Logo appears. Shortly thereafter the AGIS operation display appears. See Figures 1, 2, and 3</p> <div style="text-align: center;">  <p>Figure 1 Figure 2 Figure 3</p> </div> <p>Exhibit 4 to the Beyer Declaration, depicting AGIS PDA/cell phones.</p>
<p>using a Global Positioning Satellite (GPS) receiver of the first device to obtain data indicative of the location of the first device, wherein sending the first location information to the first server comprises using an Internet Protocol (IP) to send the first location information to the first server.</p>	<p>The AGIS LifeRing product practices “using a Global Positioning Satellite (GPS) receiver of the first device to obtain data indicative of the location of the first device, wherein sending the first location information to the first server comprises using an Internet Protocol (IP) to send the first location information to the first server.”</p> <p>The device utilizes a GPS receiver to locate a first device and the first device sends it’s location to the server using IP.</p> <p>In the file “gpslistener.cpp” the “start()”²⁹¹ method is implemented where a thread is created for processing in the background. The thread performs the function “process()”²⁹² and uses a loop to scan the comport documented as “read in a message from serial port, will only wait for 1 second for data”²⁹³ and once a message is detected that it calls the “processGPSMessage()” passing the message and length of the</p>

²⁹¹ See, e.g., csif/gpslistener.cpp at L36-51.

²⁹² See, e.g., csif/gpslistener.cpp at L101-235.

²⁹³ See, e.g., csif/gpslistener.cpp at L158.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>message. The function “processGPSMessage()”²⁹⁴ calls the “enQueueGPSMsg()” of the “DDLDatabase” class where the GPS message is queued.</p> <p>In the file “display.cpp” inside the initialization function “InitInstance()”²⁹⁵ the GPS message event processor is registered through the call to the “registerGPSListener()” much like the DDL messages. The GPS messages are then processed by the callback event capture routine “WndProc()”²⁹⁶ where the “ID_TIMER_BLINK” event documented to represent a 1 second timer makes a call to “check_gps_msgs()”, “check_for_internal_msgs()” and “check_for_db_msgs()”.</p> <p>In the file “gps.cpp” the function “check_gps_msgs()”²⁹⁷ is implemented where the queued GPS messages are process incrementally with a call to the “process_csif_gps_msg()”. The function “process_csif_gps_msg()”²⁹⁸ is implemented where the message is retrieved by a call to “get_gps_msg()” and based on one of three choices of a message type captured by the “msgType” member of the message variable “gmsg” that the position, precision or constellation of the gps message is processed. For the case where the “msgType” is of “GPS_MSG_POSITION” type the call is made to “process_gps_pos()” passing the “gmsg.body.position” variable member. In the same file the function “process_gps_pos()”²⁹⁹ is implemented. This function utilizes the “gps_pos_decode()” function to covert the input message’s members such as “gpsp.latitude” and “gpsp.longitude” to a floating point format in the “lat” and “lon” variables. Further into the function, the “lat” and “lon” variables are used to populate the members of the “temptrk” struct variable which is of type “track_file_struct”.</p> <p>In the file “display/display.cpp” the function “check_ip_address()”³⁰⁰ is an example of communication with transmission of location information. This function is conditioned to use certain code for the case where the “PPC2003” is defined, indicative of SMS capability, or not defined. In both routines similar functionality is performed where</p>

²⁹⁴ See, e.g., csif/gpslistener.cpp at L76-99.

²⁹⁵ See, e.g., display/display.cpp at L274-581.

²⁹⁶ See, e.g., display/display.cpp at L584-969.

²⁹⁷ See, e.g., display/gps.cpp at L346-365.

²⁹⁸ See, e.g., display/gps.cpp at L292-336.

²⁹⁹ See, e.g., display/gps.cpp at L77-126.

³⁰⁰ See, e.g., display/display.cpp at L1015-1037 and L1076-1115.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>calls to functions “send_net_msg_netmgmt()” to join a network group and “SendOwnPosString()” to send location information under various connection states.</p> <p>In the file “display/buttons.cpp” in the main routine where it is documented that button actions are processed by “button_actions”³⁰¹ and are implemented in the function “button_actions()”³⁰² the case for pressing the “BUTTON_PING.” is implemented where calls are made to “SendOwnPosString()” and “send_all_tracks()”.</p> <p>In the file “display/msgproc.cpp” the function “SendOwnPosString()”³⁰³ is implemented where the “op” struct variable of type “OWN_POSITION” is populated with location information such as latitude and longitude. Then the “dmsg” variable of type “DDLMSG” is set to have a “msgType” of “DDL_MSG_OWN_POSITION” and variable “op” set to one of its other member variables. Then the “dmsg” is set to a buffer with the call to “put_ddl_msg()” and a call to “send_message_bulk()” when the transmit method “xmit_method” is set to “SERVER” and using a “server_ip” variable. Alternatively, direct transmission may occur with the call to “send_message_direct()”</p> <p>Similarly, in the file “display/msgproc.cpp” the function “send_track_msg()”³⁰⁴ is implemented which includes location information along with previous location information. Similar to the previous function, the “dmsg” is set to type “DDL_MSG_TRACK_REPORT” and “tmsg” variable with location information is set to its member variable. The functions “put_ddl_msg()” followed by “send_message_bulk()” or “send_message_direct()” transmit the message.</p> <p>In the file “display/msgproc.cpp” the functions “send_message_bulk()” and “send_message_direct()”³⁰⁵ are implemented where both have IP transmit capability with the addition of “send_message_direct()” also having SMS transmission capability. Within the IP transmit capability, both functions make calls to the “sendTCPMessage()” function. In the</p>

³⁰¹ See, e.g., display/buttons.cpp at L594-598 and L1344-1346.

³⁰² See, e.g., display/buttons.cpp at L1344-2365.

³⁰³ See, e.g., display/msgproc.cpp at L170-325.

³⁰⁴ See, e.g., display/msgproc.cpp at L615-682.

³⁰⁵ See, e.g., display/msgproc.cpp at L47-127.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	file "csif.cpp" the "sendTCPMessage()" ³⁰⁶ function is implemented where it utilizes the "tcpConnection" variable to call the "enqueueForSending()" function. The "tcpConnection" variable holds the necessary information to complete an IP based transmission. In the file "csif.cpp" the function "initTCP()" ³⁰⁷ is implemented where the "tcpConnection" variable is instantiated. In this function a network management object "nm" from the "NetworkMgmt" class is used to call the "connectIPNetwork()" function and implement an "ipMonitor" variable to monitor the IP network. Also the "tcpConnection" variable is set to a new object by a call to "TCPServerConnection()" function.
18. The method of claim 17, wherein sending the first location information to the first server further comprises sending the first location information via the Internet.	<p>The AGIS LifeRing product practices "wherein sending the first location information to the first server further comprises sending the first location information via the Internet."</p> <p>Once the devices have formed a group they share their location with each other. The AGIS software captures location information on the device through a serial communication link configured to continuously process GPS information messages. Messages such as location, precision and constellation are captured on an independent thread and when a valid message is received the thread triggers an event indicating the presence of a new message. The software then sends the new location to others through either a server or direct connection.</p> <p>The "csif" module provides a service to start a GPSThread that communicates with a serial port. In the file "csif/csif.cpp" the function "initGPS()"³⁰⁸ is implemented where the variable "comport" is the argument. The variable "gpsListener" is then assigned to a new GPSThread class using the comport variable followed by calling the "GPSThread->start()" call.</p> <p>See also references to "address_IP" and "IP__address" in display.cpp and csif files for transmission over IP.</p> <p>In the file "gpslistener.cpp" the "start()"³⁰⁹ method is implemented where a thread is created for processing in the background. The thread performs the function "process()"³¹⁰ and uses a loop to scan the</p>

³⁰⁶ See, e.g., csif.cpp at L378-386.³⁰⁷ See, e.g., csif.cpp at L150-204.³⁰⁸ See, e.g., csif/csif.cpp at L126-148.³⁰⁹ See, e.g., csif/gpslistener.cpp at L36-51.³¹⁰ See, e.g., csif/gpslistener.cpp at L101-235.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>comport documented as “read in a message from serial port, will only wait for 1 second for data”³¹¹ and once a message is detected that it calls the “processGPSMessage()” passing the message and length of the message. The function “processGPSMessage()”³¹² calls the “enQueueGPSMsg()” of the “DDLDatabase” class where the GPS message is queued.</p> <p>In the file “display.cpp” inside the initialization function “InitInstance()”³¹³ the GPS message event processor is registered through the call to the “registerGPSListener()” much like the DDL messages. The GPS messages are then processed by the callback event capture routine “WndProc()”³¹⁴ where the “ID_TIMER_BLINK” event documented to represent a 1 second timer makes a call to “check_gps_msgs()”, “check_for_internal_msgs()” and “check_for_db_msgs()”.</p> <p>In the file “gps.cpp” the function “check_gps_msgs()”³¹⁵ is implemented where the queued GPS messages are processed incrementally with a call to the “process_csif_gps_msg()”. The function “process_csif_gps_msg()”³¹⁶ is implemented where the message is retrieved by a call to “get_gps_msg()” and based on one of three choices of a message type captured by the “msgType” member of the message variable “gmsg” that the position, precision or constellation of the gps message is processed. For the case where the “msgType” is of “GPS_MSG_POSITION” type the call is made to “process_gps_pos()” passing the “gmsg.body.position” variable member. In the same file the function “process_gps_pos()”³¹⁷ is implemented. This function utilizes the “gps_pos_decode()” function to covert the input message’s members such as “gpsp.latitude” and “gpsp.longitude” to a floating point format in the “lat” and “lon” variables. Further into the function, the “lat” and “lon” variables are used to populate the members of the “temptrk” struct variable which is of type “track_file_struct”.</p>

³¹¹ See, e.g., csif/gpslistener.cpp at L158.

³¹² See, e.g., csif/gpslistener.cpp at L76-99.

³¹³ See, e.g., display/display.cpp at L274-581.

³¹⁴ See, e.g., display/display.cpp at L584-969.

³¹⁵ See, e.g., display/gps.cpp at L346-365.

³¹⁶ See, e.g., display/gps.cpp at L292-336.

³¹⁷ See, e.g., display/gps.cpp at L77-126.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>In the file “display/display.cpp” the function “check_ip_address()”³¹⁸ is an example of communication with transmission of location information. This function is conditioned to use certain code for the case where the “PPC2003” is defined, indicative of SMS capability, or not defined. In both routines similar functionality is performed where calls to functions “send_net_msg_netmgmt()” to join a network group and “SendOwnPosString()” to send location information under various connection states.</p> <p>In the file “display/buttons.cpp” in the main routine where it is documented that button actions are processed by “button_actions”³¹⁹ and are implemented in the function “button_actions()”³²⁰ the case for pressing the “BUTTON_PING:” is implemented where calls are made to “SendOwnPosString()” and “send_all_tracks()”.</p> <p>In the file “display/msgproc.cpp” the function “SendOwnPosString()”³²¹ is implemented where the “op” struct variable of type “OWN_POSITION” is populated with location information such as latitude and longitude. Then the “dmsg” variable of type “DDLMSG” is set to have a “msgType” of “DDL_MSG_OWN_POSITION” and variable “op” set to one of its other member variables. Then the “dmsg” is set to a buffer with the call to “put_ddl_msg()” and a call to “send_message_bulk()” when the transmit method “xmit_method” is set to “SERVER” and using a “server_ip” variable. Alternatively, direct transmission may occur with the call to “send_message_direct()”</p> <p>Similarly, in the file “display/msgproc.cpp” the function “send_track_msg()”³²² is implemented which includes location information along with previous location information. Similar to the previous function, the “dmsg” is set to type “DDL_MSG_TRACK_REPORT” and “tmsg” variable with location information is set to its member variable. The functions “put_ddl_msg()” followed by “send_message_bulk()” or “send_message_direct()” transmit the message.</p>

³¹⁸ See, e.g., display/display.cpp at L1015-1037 and L1076-1115.

³¹⁹ See, e.g., display/buttons.cpp at L594-598 and L1344-1346.

³²⁰ See, e.g., display/buttons.cpp at L1344-2365.

³²¹ See, e.g., display/msgproc.cpp at L170-325.

³²² See, e.g., display/msgproc.cpp at L615-682.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>As a further example of receiving messages from the second devices that include location information, each time a message is received by the server from a device, the location information from that device is updated at the server. The location information for each device in the group (e.g. the second devices) is then transmitted as part of the update message to the first device, as depicted in <code>nmsg.numMapEntries</code>. <code>msgproc.cpp</code> at ll. 862-892.</p> <p>The “<code>send_track_msg()</code>” function is used under the condition where “<code>add_track()</code>”³²³ and “<code>update_track_id()</code>”³²⁴ functions in the “<code>display/track.cpp</code>” is invoked. The “<code>update_track_id()</code>” is used in user interface as views in the “<code>buttons.cpp</code>” file in the main routine where button actions are implemented “<code>button_actions()</code>” as discussed above where the case for “<code>BUTTON_TRACK_ID_UNKN</code>”, “<code>BUTTON_TRACK_ID_FRND</code>” and “<code>BUTTON_TRACK_ID_HOSTILE</code>” cases are handled for various features including the call to “<code>update_track_id()</code>”.</p> <p>On the server side, the file “<code>server/msgproc.cpp</code>” implements the function “<code>process_csif_ddl_msg()</code>”³²⁵ where the incoming message uses the “<code>msgType</code>” to branch to a section of the code depicted to represent the “<code>DDL_MSG_OWN_POSITION</code>” case where the message is passed through by a call to “<code>pass_thru_msg()</code>” function.</p> <p>The file “<code>server/server.cpp</code>” implements the main function “<code>WndProc()</code>”³²⁶ as a callback where a receipt of a message triggers its execution. The case of “<code>CSIF_DDL_MSG_AVAILABLE</code>” is used to call the “<code>getNextDDLMessage()</code>” with a variable “<code>route</code>” that is used to call the “<code>process_csif_ddl_msg()</code>” function as described above to make the call for pass through.</p>
19. The method of claim 1, wherein participating in the group further includes sending first status	The AGIS LifeRing product practices “wherein participating in the group further includes sending first status information to the first server and receiving second status information from the first server, wherein the first status information comprises data indicative of a battery level of the first device, a signal strength of a wireless signal of the first device, a status of a Global Positioning Satellite (GPS) receiver of the

³²³ See, e.g., `display/track.cpp` at L373-477.³²⁴ See, e.g., `display/track.cpp` at L482-510.³²⁵ See, e.g., `server/msgproc.cpp` at L377-474.³²⁶ See, e.g., `server/server.cpp` at L228-386.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p>information to the first server and receiving second status information from the first server, wherein the first status information comprises data indicative of a battery level of the first device, a signal strength of a wireless signal of the first device, a status of a Global Positioning Satellite (GPS) receiver of the first device, or a combination thereof, and wherein the second status information comprises data indicative of one or more battery levels of the respective one or more second devices included in the group, one or more signal strengths of wireless signals of the respective one or more second devices included in the group, one or more statuses of</p>	<p>first device, or a combination thereof, and wherein the second status information comprises data indicative of one or more battery levels of the respective one or more second devices included in the group, one or more signal strengths of wireless signals of the respective one or more second devices included in the group, one or more statuses of GPS receivers of the respective one or more second devices included in the group, or a combination thereof.”</p> <p>The devices process battery levels to populate a navbar display that includes the display of the battery levels that are transmitted to the server for display and sharing with other devices. The devices also process signal strength</p> <p>In the file “display.cpp” the function “InitInstance()”³²⁷ and the function “WndProc()”³²⁸ are implemented for display initialization and run time processing where calls to “display_nav_bar()” and “timed_display_nav_bar()” are made respectively. Also, there are calls to “check_ip_address()” in both initialization and run time processing where subsequent uses of navbar data is present.</p> <p>In the file “navbar.cpp” the function “timed_display_nav_bar()”³²⁹ is implemented where it calls the function “display_nav_bar()”. The function “display_nav_bar()”³³⁰ is also implemented where it is documented to “display all indicators from previous update”. The function stores the result of the function “battery_check()” in the variable “bat_ret” and builds the “str” array at location [10] to empty for full battery, “b” for very low level and “B” for low level battery. The “str” variable is then copied to the “outstr” array and is used to display the result with calls to either “SHSetNavBarText()” or “system_readout()” based on the type of processor running the code where it stores the navbar data in the system memory.</p> <p>In the file “display.cpp” the function “check_ip_address()”³³¹ is implemented for two different cases of “PPC2003” or else. The case of “PPC2003” indicative of a mobile handset having battery levels the call to “display_nav_bar()” is made followed by “SendOwnPosString()” which is described above. The code is also conditioned on the use of</p>

³²⁷ See, e.g., display.cpp at L268-581.

³²⁸ See, e.g., display.cpp at L584-969.

³²⁹ See, e.g., navbar.cpp at L163-173.

³³⁰ See, e.g., navbar.cpp at L69-161.

³³¹ See, e.g., display.cpp at L1014-1117.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
GPS receivers of the respective one or more second devices included in the group, or a combination thereof.	<p>“SMS” or not, indicative of a server. The position and battery level data are used as part of a message to be sent to either the server or other devices directly.</p> <p>In the file “msgproc.cpp” the function “SendOwnPosString()”³³² is implemented as described above for creating and sending a message. In the message as described above the “op” variable struct also stores the “signalStrength” which is populated by a call to the function “getSignalStrength()”. Also stored in the “op” variable struct is the “GPSstatus” which is set to either “OFF”, “ON”, “ON2D”, “ON3D”, “STATIONARY” and “FAULT”. Furthermore, each track with associated device membership in the group holds “signalStrength” and “GPSstatus” members in the “track_file” indexed array.</p> <p>In the file “csif.cpp” the function “getSignalStrength()”³³³ where the call is made to the “getSignalStrength()” of the “Telephony” class.</p> <p>In the file “telephony.cpp” the function “getSignalStrength()”³³⁴ where a call is made to “lineGetLineDevStatus()” storing the result in the “ls” variable with member “dwSignalLevel” and is further processed from a range of 0-65535 to a range of 0-10000.</p> <p>In the file “display.cpp” the function “WndProc()”³³⁵ indicative of runtime processing of display events captures the “ID_TIMER_COMM” where a call to the “GPS_status_check()” is made as described above for the case of connection to the GPS device via a serial link.</p>
20. The method of claim 1, wherein the second georeferenced map data comprise a satellite image or aerial photograph.	The AGIS LifeRing product practices “wherein the second georeferenced map data comprise a satellite image or aerial photograph.” See, e.g., “satelliteID” in “csiftranscode.h” and “display\gps.cpp.”
21. The method of claim 1, wherein the	The AGIS LifeRing product practices “wherein the spatial coordinates comprise latitude and longitude coordinates.”

³³² See, e.g., display/msgproc.cpp at L170-325.³³³ See, e.g., csif.cpp at L673-680.³³⁴ See, e.g., telephony.cpp at L218-234.³³⁵ See, e.g., display/display.cpp at L584-969.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
spatial coordinates comprise latitude and longitude coordinates.	<p>The devices utilize interface with the GPS that calculates the latitude and longitude coordinates of the spatial coordinates.</p> <p>In the file “gps.cpp” the function “check_gps_msgs()”³³⁶ is implemented where the queued GPS messages are process incrementally with a call to the “process_csif_gps_msg()”. The function “process_csif_gps_msg()”³³⁷ is implemented where the message is retrieved by a call to “get_gps_msg()” and based on one of three choices of a message type captured by the “msgType” member of the message variable “gmsg” that the position, precision or constellation of the gps message is processed. For the case where the “msgType” is of “GPS_MSG_POSITION” type the call is made to “process_gps_pos()” passing the “gmsg.body.position” variable member. In the same file the function “process_gps_pos()”³³⁸ is implemented. This function utilizes the “gps_pos_decode()” function to covert the input message’s members such as “gpsp.latitude” and “gpsp.longitude” to a floating point format in the “lat” and “lon” vaiables. Further into the function, the “lat” and “lon” variables are used to populate the members of the “temptrk” struct variable which is of type “track_file_struct”.</p>
22. The method of claim 1, further comprising identifying, by the first device, user interaction with the display selecting a particular user-selectable symbol positioned on the second georeferenced map and corresponding to a particular second device, wherein identifying the user interaction	<p>The AGIS LifeRing product practices “identifying, by the first device, user interaction with the display selecting a particular user-selectable symbol positioned on the second georeferenced map and corresponding to a particular second device, wherein identifying the user interaction selecting the particular user-selectable symbol comprises.”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”³³⁹ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON MSG FREETEXT”, “BUTTON NET”,</p>

³³⁶ See, e.g., display/gps.cpp at L346-365.³³⁷ See, e.g., display/gps.cpp at L292-336.³³⁸ See, e.g., display/gps.cpp at L77-126.³³⁹ See, e.g., buttons.cpp at L1344-2365.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
selecting the particular user-selectable symbol comprises:	<p>“BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”³⁴⁰ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”³⁴¹ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”³⁴² and “num_on_que_list()”³⁴³ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”³⁴⁴ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”³⁴⁵ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”³⁴⁶ and subsequently to “form_net_line()”³⁴⁷ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”³⁴⁸ and subsequently “send_net_msg_participants()”³⁴⁹ the message is transmitted directly or through a server as described above through the use of the</p>

³⁴⁰ See, e.g., display/ftext.cpp at L176-244.³⁴¹ See, e.g., netselect.cpp at L152-221.³⁴² See, e.g., quelist.cpp at L16-22.³⁴³ See, e.g., quelist.cpp at L60-71.³⁴⁴ See, e.g., display/msgproc.cpp at L328-399.³⁴⁵ See, e.g., netselect.cpp at L391-492.³⁴⁶ See, e.g., netselect.cpp at L373-384.³⁴⁷ See, e.g., netselect.cpp at L352-370.³⁴⁸ See, e.g., display/msgproc.cpp at L1234-1292.³⁴⁹ See, e.g., display/msgproc.cpp at L828-922.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”³⁵⁰ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”³⁵¹ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”³⁵² is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
detecting user selection of a portion of the interactive display corresponding to a position on the second georeferenced map;	<p>The AGIS LifeRing product practices “detecting user selection of a portion of the interactive display corresponding to a position on the second georeferenced map.”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”³⁵³ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for</p>

³⁵⁰ See, e.g., photo.cpp at L189-292.³⁵¹ See, e.g., server/msgproc.cpp at L377-474.³⁵² See, e.g., server/msgproc.cpp at L 51-61.³⁵³ See, e.g., buttons.cpp at L1344-2365.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”³⁵⁴ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”³⁵⁵ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”³⁵⁶ and “num_on_que_list()”³⁵⁷ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”³⁵⁸ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”³⁵⁹ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”³⁶⁰ and subsequently to “form_net_line()”³⁶¹ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”³⁶² and subsequently “send_net_msg_participants()”³⁶³ the message is transmitted directly or through a server as described above through the use of the</p>

³⁵⁴ See, e.g., display/ftext.cpp at L176-244.³⁵⁵ See, e.g., netselect.cpp at L152-221.³⁵⁶ See, e.g., quelist.cpp at L16-22.³⁵⁷ See, e.g., quelist.cpp at L60-71.³⁵⁸ See, e.g., display/msgproc.cpp at L328-399.³⁵⁹ See, e.g., netselect.cpp at L391-492.³⁶⁰ See, e.g., netselect.cpp at L373-384.³⁶¹ See, e.g., netselect.cpp at L352-370.³⁶² See, e.g., display/msgproc.cpp at L1234-1292.³⁶³ See, e.g., display/msgproc.cpp at L828-922.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”³⁶⁴ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”³⁶⁵ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”³⁶⁶ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
based at least in part on coordinates of the selected position on the second georeferenced map and on the second georeferenced map data relating positions on the second georeferenced map to spatial coordinates, determining spatial coordinates of a location represented	<p>The AGIS LifeRing product practices “based at least in part on coordinates of the selected position on the second georeferenced map and on the second georeferenced map data relating positions on the second georeferenced map to spatial coordinates, determining spatial coordinates of a location represented by the selected position on the second georeferenced map.”</p> <p>The device utilizes buttons on the display limited to other entities in a latitude and longitude limited space to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p>

³⁶⁴ See, e.g., photo.cpp at L189-292.³⁶⁵ See, e.g., server/msgproc.cpp at L377-474.³⁶⁶ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
by the selected position on the second georeferenced map;	<p>In the file “tact.cpp” the function “refresh_tact()”³⁶⁷ is implemented to draw the tactical display including when some change to location or extent of the display is made. In this function a call is made to the function “track_draw()” to draw the extent of the tracks.</p> <p>In the file “tact.cpp” the function “track_draw()”³⁶⁸ is implemented where based on a latitude and longitude value position of a track the function “screen_clip_check()” is called and if the entity is not on the display that the processing loop exits the loop not drawing the track leaving only tracks corresponding to other entities that are close to a selected position is drawn and subsequently the user can interact with.</p> <p>In the file “buttons.cpp” the function “button_actions()”³⁶⁹ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”³⁷⁰ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”³⁷¹ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”³⁷² and “num_on_que_list()”³⁷³ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”³⁷⁴ is implemented where the free text message is transmitted directly or</p>

³⁶⁷ See, e.g., tact.cpp at L200-277.³⁶⁸ See, e.g., tact.cpp at L557-742.³⁶⁹ See, e.g., buttons.cpp at L1344-2365.³⁷⁰ See, e.g., display/ftext.cpp at L176-244.³⁷¹ See, e.g., netselect.cpp at L152-221.³⁷² See, e.g., quelist.cpp at L16-22.³⁷³ See, e.g., quelist.cpp at L60-71.³⁷⁴ See, e.g., display/msgproc.cpp at L328-399.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netsselect.cpp” the function “create_netlist()”³⁷⁵ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”³⁷⁶ and subsequently to “form_net_line()”³⁷⁷ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”³⁷⁸ and subsequently “send_net_msg_participants()”³⁷⁹ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”³⁸⁰ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”³⁸¹ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”³⁸² is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p>

³⁷⁵ See, e.g., netsselect.cpp at L391-492.

³⁷⁶ See, e.g., netsselect.cpp at L373-384.

³⁷⁷ See, e.g., netsselect.cpp at L352-370.

³⁷⁸ See, e.g., display/msgproc.cpp at L1234-1292.

³⁷⁹ See, e.g., display/msgproc.cpp at L828-922.

³⁸⁰ See, e.g., photo.cpp at L189-292.

³⁸¹ See, e.g., server/msgproc.cpp at L377-474.

³⁸² See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.
and identifying the particular user-selectable symbol based, at least in part, on the spatial coordinates represented by the selected position.	<p>The AGIS LifeRing product practices “identifying the particular user-selectable symbol based, at least in part, on the spatial coordinates represented by the selected position.”</p> <p>The device utilizes buttons on the display limited to other entities in a latitude and longitude limited space to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “tact.cpp” the function “refresh_tact()”³⁸³ is implemented to draw the tactical display including when some change to location or extent of the display is made. In this function a call is made to the function “track_draw()” to draw the extent of the tracks.</p> <p>In the file “tact.cpp” the function “track_draw()”³⁸⁴ is implemented where based on a latitude and longitude value position of a track the function “screen_clip_check()” is called and if the entity is not on the display that the processing loop exits the loop not drawing the track leaving only tracks corresponding to other entities that are close to a selected position is drawn and subsequently the user can interact with.</p> <p>In the file “buttons.cpp” the function “button_actions()”³⁸⁵ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p>

³⁸³ See, e.g., tact.cpp at L200-277.³⁸⁴ See, e.g., tact.cpp at L557-742.³⁸⁵ See, e.g., buttons.cpp at L1344-2365.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>In the file “display/ftext.cpp” the function “create_ftext()”³⁸⁶ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”³⁸⁷ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”³⁸⁸ and “num_on_que_list()”³⁸⁹ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”³⁹⁰ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”³⁹¹ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”³⁹² and subsequently to “form_net_line()”³⁹³ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”³⁹⁴ and subsequently “send_net_msg_participants()”³⁹⁵ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”³⁹⁶ is implemented where the</p>

³⁸⁶ See, e.g., display/ftext.cpp at L176-244.³⁸⁷ See, e.g., netselect.cpp at L152-221.³⁸⁸ See, e.g., quelist.cpp at L16-22.³⁸⁹ See, e.g., quelist.cpp at L60-71.³⁹⁰ See, e.g., display/msgproc.cpp at L328-399.³⁹¹ See, e.g., netselect.cpp at L391-492.³⁹² See, e.g., netselect.cpp at L373-384.³⁹³ See, e.g., netselect.cpp at L352-370.³⁹⁴ See, e.g., display/msgproc.cpp at L1234-1292.³⁹⁵ See, e.g., display/msgproc.cpp at L828-922.³⁹⁶ See, e.g., photo.cpp at L189-292.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_dll_msg()”³⁹⁷ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”³⁹⁸ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
23. The method of claim 22, wherein identifying the particular user-selectable symbol based, at least in part, on the spatial coordinates represented by the selected position comprises:	<p>The AGIS LifeRing product practices “wherein identifying the particular user-selectable symbol based, at least in part, on the spatial coordinates represented by the selected position comprises.”</p> <p>The device utilizes buttons on the display limited to other entities in a latitude and longitude limited space to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “tact.cpp” the function “refresh_tact()”³⁹⁹ is implemented to draw the tactical display including when some change to location or extent of the display is made. In this function a call is made to the function “track_draw()” to draw the extent of the tracks.</p> <p>In the file “tact.cpp” the function “track_draw()”⁴⁰⁰ is implemented where based on a latitude and longitude value position of a track the function “screen_clip_check()” is called and if the entity is not on the</p>

³⁹⁷ See, e.g., server/msgproc.cpp at L377-474.³⁹⁸ See, e.g., server/msgproc.cpp at L 51-61.³⁹⁹ See, e.g., tact.cpp at L200-277.⁴⁰⁰ See, e.g., tact.cpp at L557-742.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>display that the processing loop exits the loop not drawing the track leaving only tracks corresponding to other entities that are close to a selected position is drawn and subsequently the user can interact with.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁴⁰¹ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”⁴⁰² is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”⁴⁰³ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”⁴⁰⁴ and “num_on_que_list()”⁴⁰⁵ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”⁴⁰⁶ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”⁴⁰⁷ is implemented where the display is configured for net communication. The call to “send to display net list()”⁴⁰⁸ and</p>

⁴⁰¹ See, e.g., buttons.cpp at L1344-2365.

⁴⁰² See, e.g., display/ftext.cpp at L176-244.

⁴⁰³ See, e.g., netselect.cpp at L152-221.

⁴⁰⁴ See, e.g., quelist.cpp at L16-22.

⁴⁰⁵ See, e.g., quelist.cpp at L60-71.

⁴⁰⁶ See, e.g., display/msgproc.cpp at L328-399.

⁴⁰⁷ See, e.g., netselect.cpp at L391-492.

⁴⁰⁸ See, e.g., netselect.cpp at L373-384.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>subsequently to “form_net_line()”⁴⁰⁹ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁴¹⁰ and subsequently “send_net_msg_participants()”⁴¹¹ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁴¹² is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”⁴¹³ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”⁴¹⁴ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
searching a database of entities for an entity located nearest to the spatial coordinates represented by the selected position,	The AGIS LifeRing product practices “searching a database of entities for an entity located nearest to the spatial coordinates represented by the selected position, wherein the entities represented by data in the database include the one or more second devices included in the group, wherein the database data include locations of the respective entities, and wherein the database is searchable by location.”

⁴⁰⁹ See, e.g., netselect.cpp at L352-370.⁴¹⁰ See, e.g., display/msgproc.cpp at L1234-1292.⁴¹¹ See, e.g., display/msgproc.cpp at L828-922.⁴¹² See, e.g., photo.cpp at L189-292.⁴¹³ See, e.g., server/msgproc.cpp at L377-474.⁴¹⁴ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
wherein the entities represented by data in the database include the one or more second devices included in the group, wherein the database data include locations of the respective entities, and wherein the database is searchable by location;	<p>The device utilizes buttons on the display limited to other entities in a latitude and longitude limited space to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “tact.cpp” the function “refresh_tact()”⁴¹⁵ is implemented to draw the tactical display including when some change to location or extent of the display is made. In this function a call is made to the function “track_draw()” to draw the extent of the tracks.</p> <p>In the file “tact.cpp” the function “track_draw()”⁴¹⁶ is implemented where based on a latitude and longitude value position of a track the function “screen_clip_check()” is called and if the entity is not on the display that the processing loop exits the loop not drawing the track leaving only tracks corresponding to other entities that are close to a selected position is drawn and subsequently the user can interact with.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁴¹⁷ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”⁴¹⁸ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”⁴¹⁹ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on que_list()” functions. The file “quelist.cpp” implements the</p>

⁴¹⁵ See, e.g., tact.cpp at L200-277.⁴¹⁶ See, e.g., tact.cpp at L557-742.⁴¹⁷ See, e.g., buttons.cpp at L1344-2365.⁴¹⁸ See, e.g., display/ftext.cpp at L176-244.⁴¹⁹ See, e.g., netselect.cpp at L152-221.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“on_que_list()”⁴²⁰ and “num_on_que_list()”⁴²¹ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”⁴²² is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netsselect.cpp” the function “create_netlist()”⁴²³ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”⁴²⁴ and subsequently to “form_net_line()”⁴²⁵ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁴²⁶ and subsequently “send_net_msg_participants()”⁴²⁷ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁴²⁸ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”⁴²⁹ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the</p>

⁴²⁰ See, e.g., quelist.cpp at L16-22.

⁴²¹ See, e.g., quelist.cpp at L60-71.

⁴²² See, e.g., display/msgproc.cpp at L328-399.

⁴²³ See, e.g., netsselect.cpp at L391-492.

⁴²⁴ See, e.g., netsselect.cpp at L373-384.

⁴²⁵ See, e.g., netsselect.cpp at L352-370.

⁴²⁶ See, e.g., display/msgproc.cpp at L1234-1292.

⁴²⁷ See, e.g., display/msgproc.cpp at L828-922.

⁴²⁸ See, e.g., photo.cpp at L189-292.

⁴²⁹ See, e.g., server/msgproc.cpp at L377-474.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”⁴³⁰ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
<p>and based on a result of searching the database, identifying the particular second device as the entity located nearest to the spatial coordinates represented by the selected position, wherein the particular user-selectable symbol corresponds to the particular second device.</p>	<p>The AGIS LifeRing product practices “based on a result of searching the database, identifying the particular second device as the entity located nearest to the spatial coordinates represented by the selected position, wherein the particular user-selectable symbol corresponds to the particular second device.”</p> <p>The device utilizes buttons on the display limited to other entities in a latitude and longitude limited space to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “tact.cpp” the function “refresh_tact()”⁴³¹ is implemented to draw the tactical display including when some change to location or extent of the display is made. In this function a call is made to the function “track_draw()” to draw the extent of the tracks.</p> <p>In the file “tact.cpp” the function “track_draw()”⁴³² is implemented where based on a latitude and longitude value position of a track the function “screen_clip_check()” is called and if the entity is not on the display that the processing loop exits the loop not drawing the track leaving only tracks corresponding to other entities that are close to a selected position is drawn and subsequently the user can interact with.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁴³³ implements the main routine for capturing the button press event and processing</p>

⁴³⁰ See, e.g., server/msgproc.cpp at L 51-61.

⁴³¹ See, e.g., tact.cpp at L200-277.

⁴³² See, e.g., tact.cpp at L557-742.

⁴³³ See, e.g., buttons.cpp at L1344-2365.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”⁴³⁴ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”⁴³⁵ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”⁴³⁶ and “num_on_que_list()”⁴³⁷ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”⁴³⁸ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”⁴³⁹ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”⁴⁴⁰ and subsequently to “form_net_line()”⁴⁴¹ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁴⁴² and subsequently “send_net_msg_participants()”⁴⁴³ the message is transmitted directly or</p>

⁴³⁴ See, e.g., display/ftext.cpp at L176-244.

⁴³⁵ See, e.g., netselect.cpp at L152-221.

⁴³⁶ See, e.g., quelist.cpp at L16-22.

⁴³⁷ See, e.g., quelist.cpp at L60-71.

⁴³⁸ See, e.g., display/msgproc.cpp at L328-399.

⁴³⁹ See, e.g., netselect.cpp at L391-492.

⁴⁴⁰ See, e.g., netselect.cpp at L373-384.


⁴⁴¹ See, e.g., netselect.cpp at L352-370.

⁴⁴² See, e.g., display/msgproc.cpp at L1234-1292.

⁴⁴³ See, e.g., display/msgproc.cpp at L828-922.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁴⁴⁴ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”⁴⁴⁵ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”⁴⁴⁶ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
24. The method of claim 23, wherein the entity is a first entity, and wherein the method further comprises performing by the first device:	<p>The AGIS LifeRing product practices “wherein the entity is a first entity, and wherein the method further comprises performing by the first device.”</p> <p>The AGIS LifeRing product includes systems and methods for executing an application on PDA/cell phones, now referred to as a smartphone devices.</p> <p>See, e.g., “call.cpp” references to “phone” and “other phones.”</p>

⁴⁴⁴ See, e.g., photo.cpp at L189-292.⁴⁴⁵ See, e.g., server/msgproc.cpp at L377-474.⁴⁴⁶ See, e.g., server/msgproc.cpp at L 51-61.

<p>U.S. Patent No. 9,467,838</p>	<p>October 22, 2005 AGIS LifeRing</p>
	<p>The operator can then send voice conference or digital data to those AGIS equipped units by simply selecting their assigned NET SoftSwitch.</p> <p>The AGIS application software does not affect the operation of the PDA, the Cell phone or the GPS until it is activated. When the AGIS application software is selected, the AGIS PDA Cell phone is turned on, the GPS is automatically connected to the PDA / Cell phone through a Bluetooth interface and the AGIS Logo appears. Shortly thereafter the AGIS operation display appears. See Figures 1, 2, and 3</p>  <p style="text-align: center;">Figure 1 Figure 2 Figure 3</p> <p>Exhibit 4 to the Beyer Declaration, depicting AGIS PDA/cell phones.</p>
<p>receiving user input via user interaction with the interactive display of the first device, the user input specifying a location and a symbol corresponding to a second entity other than the first device and the one or more second devices included in the group;</p>	<p>The AGIS LifeRing product practices “receiving user input via user interaction with the interactive display of the first device, the user input specifying a location and a symbol corresponding to a second entity other than the first device and the one or more second devices included in the group.”</p> <p>The device utilizes buttons on the display limited to other entities in a latitude and longitude limited space to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “tact.cpp” the function “refresh_tact()”⁴⁴⁷ is implemented to draw the tactical display including when some change to location or extent of the display is made. In this function a call is made to the function “track_draw()” to draw the extent of the tracks.</p>

⁴⁴⁷ See, e.g., tact.cpp at L200-277.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>In the file “tact.cpp” the function “track_draw()”⁴⁴⁸ is implemented where based on a latitude and longitude value position of a track the function “screen_clip_check()” is called and if the entity is not on the display that the processing loop exits the loop not drawing the track leaving only tracks corresponding to other entities that are close to a selected position is drawn and subsequently the user can interact with.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁴⁴⁹ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”⁴⁵⁰ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”⁴⁵¹ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”⁴⁵² and “num_on_que_list()”⁴⁵³ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”⁴⁵⁴ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function</p>

⁴⁴⁸ See, e.g., tact.cpp at L557-742.

⁴⁴⁹ See, e.g., buttons.cpp at L1344-2365.

⁴⁵⁰ See, e.g., display/ftext.cpp at L176-244.

⁴⁵¹ See, e.g., netselect.cpp at L152-221.

⁴⁵² See, e.g., quelist.cpp at L16-22.

⁴⁵³ See, e.g., quelist.cpp at L60-71.

⁴⁵⁴ See, e.g., display/msgproc.cpp at L328-399.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“create_netlist()”⁴⁵⁵ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”⁴⁵⁶ and subsequently to “form_net_line()”⁴⁵⁷ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁴⁵⁸ and subsequently “send_net_msg_participants()”⁴⁵⁹ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁴⁶⁰ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”⁴⁶¹ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”⁴⁶² is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
and based on the user input, adding the user-specified	The AGIS LifeRing product practices “based on the user input, adding the user-specified symbol to the interactive display at a position on the

⁴⁵⁵ See, e.g., netselect.cpp at L391-492.

⁴⁵⁶ See, e.g., netselect.cpp at L373-384.

⁴⁵⁷ See, e.g., netselect.cpp at L352-370.

⁴⁵⁸ See, e.g., display/msgproc.cpp at L1234-1292.

⁴⁵⁹ See, e.g., display/msgproc.cpp at L828-922.

⁴⁶⁰ See, e.g., photo.cpp at L189-292.

⁴⁶¹ See, e.g., server/msgproc.cpp at L377-474.

⁴⁶² See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
symbol to the interactive display at a position on the second georeferenced map corresponding to the user-specified location of the second entity.	<p>second georeferenced map corresponding to the user-specified location of the second entity.”</p> <p>The device utilizes buttons on the display limited to other entities in a latitude and longitude limited space to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “tact.cpp” the function “refresh_tact()”⁴⁶³ is implemented to draw the tactical display including when some change to location or extent of the display is made. In this function a call is made to the function “track_draw()” to draw the extent of the tracks.</p> <p>In the file “tact.cpp” the function “track_draw()”⁴⁶⁴ is implemented where based on a latitude and longitude value position of a track the function “screen_clip_check()” is called and if the entity is not on the display that the processing loop exits the loop not drawing the track leaving only tracks corresponding to other entities that are close to a selected position is drawn and subsequently the user can interact with.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁴⁶⁵ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”⁴⁶⁶ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”⁴⁶⁷ function is implemented. List of participants</p>

⁴⁶³ See, e.g., tact.cpp at L200-277.

⁴⁶⁴ See, e.g., tact.cpp at L557-742.

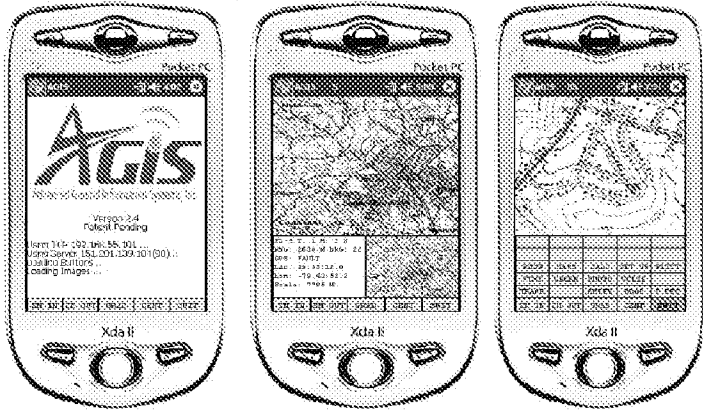
⁴⁶⁵ See, e.g., buttons.cpp at L1344-2365.

⁴⁶⁶ See, e.g., display/ftext.cpp at L176-244.

⁴⁶⁷ See, e.g., netselect.cpp at L152-221.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”⁴⁶⁸ and “num_on_que_list()”⁴⁶⁹ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”⁴⁷⁰ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”⁴⁷¹ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”⁴⁷² and subsequently to “form_net_line()”⁴⁷³ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁴⁷⁴ and subsequently “send_net_msg_participants()”⁴⁷⁵ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁴⁷⁶ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p>

⁴⁶⁸ See, e.g., *quelist.cpp* at L16-22.⁴⁶⁹ See, e.g., *quelist.cpp* at L60-71.⁴⁷⁰ See, e.g., *display/msgproc.cpp* at L328-399.⁴⁷¹ See, e.g., *netselect.cpp* at L391-492.⁴⁷² See, e.g., *netselect.cpp* at L373-384.⁴⁷³ See, e.g., *netselect.cpp* at L352-370.⁴⁷⁴ See, e.g., *display/msgproc.cpp* at L1234-1292.⁴⁷⁵ See, e.g., *display/msgproc.cpp* at L828-922.⁴⁷⁶ See, e.g., *photo.cpp* at L189-292.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>In the file “server/msgproc.cpp” the function “process_csif_dll_msg()”⁴⁷⁷ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”⁴⁷⁸ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
25. The method of claim 24, further comprising performing by the first device:	<p>The AGIS LifeRing product includes systems and methods for executing an application on PDA/cell phones, now referred to as a smartphone devices.</p> <p>See, e.g., “call.cpp” references to “phone” and “other phones.”</p> <p>The operator can then send voice conference or digital data to those AGIS equipped units by simply selecting their assigned NET SoftSwitch.</p> <p>The AGIS application software does not affect the operation of the PDA, the Cell phone or the GPS until it is activated. When the AGIS application software is selected, the AGIS PDA Cell phone is turned on, the GPS is automatically connected to the PDA / Cell phone through a Bluetooth interface and the AGIS Logo appears. Shortly thereafter the AGIS operation display appears. See Figures 1, 2, and 3</p> <div style="text-align: center;">  <p>Figure 1 Figure 2 Figure 3</p> </div> <p>Exhibit 4 to the Beyer Declaration, depicting AGIS PDA/cell phones.</p>

⁴⁷⁷ See, e.g., server/msgproc.cpp at L377-474.⁴⁷⁸ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
transmitting the user-specified symbol and location of the second entity to the one or more second devices included in the group for addition of the user-specified symbol to respective interactive displays of the one or more second devices at respective positions on respective georeferenced maps corresponding to the user-specified location of the second entity.	<p>The AGIS LifeRing product practices “transmitting the user-specified symbol and location of the second entity to the one or more second devices included in the group for addition of the user-specified symbol to respective interactive displays of the one or more second devices at respective positions on respective georeferenced maps corresponding to the user-specified location of the second entity.”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁴⁷⁹ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”⁴⁸⁰ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”⁴⁸¹ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”⁴⁸² and “num_on_que_list()”⁴⁸³ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”⁴⁸⁴ is implemented where the free text message is transmitted directly or</p>

⁴⁷⁹ See, e.g., buttons.cpp at L1344-2365.⁴⁸⁰ See, e.g., display/ftext.cpp at L176-244.⁴⁸¹ See, e.g., netselect.cpp at L152-221.⁴⁸² See, e.g., quelist.cpp at L16-22.⁴⁸³ See, e.g., quelist.cpp at L60-71.⁴⁸⁴ See, e.g., display/msgproc.cpp at L328-399.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netsselect.cpp” the function “create_netlist()”⁴⁸⁵ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”⁴⁸⁶ and subsequently to “form_net_line()”⁴⁸⁷ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁴⁸⁸ and subsequently “send_net_msg_participants()”⁴⁸⁹ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁴⁹⁰ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”⁴⁹¹ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”⁴⁹² is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p>

⁴⁸⁵ See, e.g., netsselect.cpp at L391-492.

⁴⁸⁶ See, e.g., netsselect.cpp at L373-384.

⁴⁸⁷ See, e.g., netsselect.cpp at L352-370.

⁴⁸⁸ See, e.g., display/msgproc.cpp at L1234-1292.

⁴⁸⁹ See, e.g., display/msgproc.cpp at L828-922.

⁴⁹⁰ See, e.g., photo.cpp at L189-292.

⁴⁹¹ See, e.g., server/msgproc.cpp at L377-474.

⁴⁹² See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.
26. The method of claim 25, wherein the user input further specifies information associated with the second entity, and wherein the method further comprises performing by the first device:	<p>The AGIS LifeRing product practices “wherein the user input further specifies information associated with the second entity, and wherein the method further comprises performing by the first device.”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁴⁹³ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”⁴⁹⁴ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”⁴⁹⁵ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”⁴⁹⁶ and “num_on_que_list()”⁴⁹⁷ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”⁴⁹⁸ is implemented where the free text message is transmitted directly or</p>

⁴⁹³ See, e.g., buttons.cpp at L1344-2365.⁴⁹⁴ See, e.g., display/ftext.cpp at L176-244.⁴⁹⁵ See, e.g., netselect.cpp at L152-221.⁴⁹⁶ See, e.g., quelist.cpp at L16-22.⁴⁹⁷ See, e.g., quelist.cpp at L60-71.⁴⁹⁸ See, e.g., display/msgproc.cpp at L328-399.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netsselect.cpp” the function “create_netlist()”⁴⁹⁹ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”⁵⁰⁰ and subsequently to “form_net_line()”⁵⁰¹ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁵⁰² and subsequently “send_net_msg_participants()”⁵⁰³ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁵⁰⁴ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”⁵⁰⁵ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”⁵⁰⁶ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p>

⁴⁹⁹ See, e.g., netsselect.cpp at L391-492.

⁵⁰⁰ See, e.g., netsselect.cpp at L373-384.

⁵⁰¹ See, e.g., netsselect.cpp at L352-370.

⁵⁰² See, e.g., display/msgproc.cpp at L1234-1292.

⁵⁰³ See, e.g., display/msgproc.cpp at L828-922.

⁵⁰⁴ See, e.g., photo.cpp at L189-292.

⁵⁰⁵ See, e.g., server/msgproc.cpp at L377-474.

⁵⁰⁶ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.
transmitting the user-specified information associated with the second entity to the one or more second devices included in the group.	<p>The AGIS LifeRing product practices “transmitting the user-specified information associated with the second entity to the one or more second devices included in the group.”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁵⁰⁷ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”⁵⁰⁸ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”⁵⁰⁹ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”⁵¹⁰ and “num_on_que_list()”⁵¹¹ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”⁵¹² is</p>

⁵⁰⁷ See, e.g., buttons.cpp at L1344-2365.

⁵⁰⁸ See, e.g., display/ftext.cpp at L176-244.

⁵⁰⁹ See, e.g., netselect.cpp at L152-221.

⁵¹⁰ See, e.g., quelist.cpp at L16-22.

⁵¹¹ See, e.g., quelist.cpp at L60-71.

⁵¹² See, e.g., display/msgproc.cpp at L328-399.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netsselect.cpp” the function “create_netlist()”⁵¹³ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”⁵¹⁴ and subsequently to “form_net_line()”⁵¹⁵ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁵¹⁶ and subsequently “send_net_msg_participants()”⁵¹⁷ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁵¹⁸ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”⁵¹⁹ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”⁵²⁰ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p>

⁵¹³ See, e.g., netsselect.cpp at L391-492.⁵¹⁴ See, e.g., netsselect.cpp at L373-384.⁵¹⁵ See, e.g., netsselect.cpp at L352-370.⁵¹⁶ See, e.g., display/msgproc.cpp at L1234-1292.⁵¹⁷ See, e.g., display/msgproc.cpp at L828-922.⁵¹⁸ See, e.g., photo.cpp at L189-292.⁵¹⁹ See, e.g., server/msgproc.cpp at L377-474.⁵²⁰ See, e.g., server/msgproc.cpp at L 51-61.

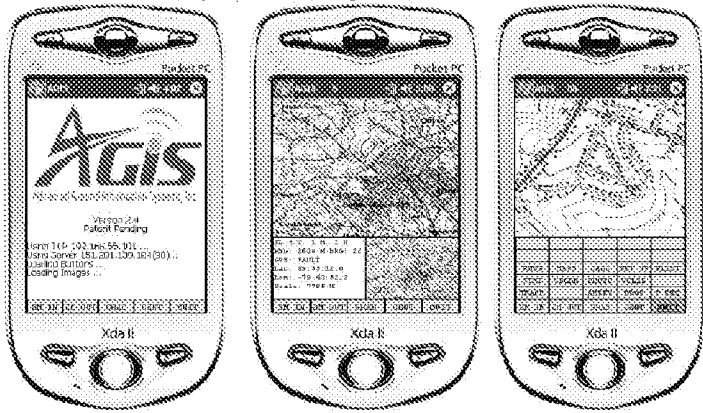
U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.
27. The method of claim 26, wherein the information comprises a category of the second entity.	<p>The AGIS LifeRing product practices “wherein the information comprises a category of the second entity.”</p> <p><i>See Claim 26</i> where the categories are divided further as “UNK”, “FRND” and “HOSTILE”.</p> <p>The “send_track_msg()” function is used under the condition where “add_track()”⁵²¹ and “update_track_id()”⁵²² functions in the “display/track.cpp” is invoked. The “update_track_id()” is used in user interface as views in the “buttons.cpp” file in the main routine where button actions are implemented “button_actions()” as discussed above where the case for “BUTTON_TRACK_ID_UNKN”, “BUTTON_TRACK_ID_FRND” and “BUTTON_TRACK_ID_HOSTILE” cases are handled for various features including the call to “update_track_id()”.</p>
28. The method of claim 27, wherein the category comprises a vehicle, a person, an event, a site, a building, or a facility.	<p>The AGIS LifeRing product practices “wherein the category comprises a vehicle, a person, an event, a site, a building, or a facility.”</p> <p><i>See Claim 27</i> where further the categories are divided further as “UNKN_GRD”⁵²³, “ARMOR”, “ARTILLARY”, “VEHICLE”, “INF”, “UNKN_SEA”, “MIL_SEA”, “COM_SEA”, “PRV_SEA”, “UNKN_AIR”, “BUTTON_TRACK_TYPE_MIL_AIR” and “COM_AIR”.</p> <p>In the file “buttons.h” the button types are defined as “BUTTON_TRACK_TYPE_UNKN_GRD”⁵²⁴, “BUTTON_TRACK_TYPE_ARMOR”, “BUTTON_TRACK_TYPE_ARTILLARY”, “BUTTON_TRACK_TYPE_VEHICLE”, “BUTTON_TRACK_TYPE_INF”, “BUTTON_TRACK_TYPE_UNKN_SEA”, “BUTTON_TRACK_TYPE_MIL_SEA”,</p>

⁵²¹ See, e.g., display/track.cpp at L373-477.⁵²² See, e.g., display/track.cpp at L482-510.⁵²³ See, e.g., buttons.h at L117-132.⁵²⁴ See, e.g., buttons.h at L117-132.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“BUTTON_TRACK_TYPE_COM_SEA”, “BUTTON_TRACK_TYPE_PRV_SEA”, “BUTTON_TRACK_TYPE_UNKN_AIR”, “BUTTON_TRACK_TYPE_MIL_AIR” and “BUTTON_TRACK_TYPE_COM_AIR” where the user can assign a button type to a track, gets assigned to a track and during the process of update to other devices the track type in terms of a button symbol is also communicated.</p>
29. The method of claim 26, wherein the information comprises an image.	<p>The AGIS LifeRing product practices “wherein the information comprises an image.”</p> <p><i>See Claim 26</i> where further information includes an image.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁵²⁵ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p>
30. The method of claim 26, wherein the information comprises at least one type of information selected from the group consisting of text and video.	<p>The AGIS LifeRing product practices “wherein the information comprises at least one type of information selected from the group consisting of text and video.”</p> <p><i>See Claim 26</i> where further information includes text and video.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁵²⁶ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p>

⁵²⁵ See, e.g., buttons.cpp at l1344-2365.

⁵²⁶ See, e.g., buttons.cpp at l1344-2365.

<p>U.S. Patent No. 9,467,838</p>	<p>October 22, 2005 AGIS LifeRing</p>
<p>31. The method of claim 26, further comprising performing by the first device:</p>	<p>The AGIS LifeRing product includes systems and methods for executing an application on PDA/cell phones, now referred to as a smartphone devices.</p> <p>See, e.g., “call.cpp” references to “phone” and “other phones.”</p> <p>The operator can then send voice conference or digital data to those AGIS equipped units by simply selecting their assigned NET SoftSwitch.</p> <p>The AGIS application software does not affect the operation of the PDA, the Cell phone or the GPS until it is activated. When the AGIS application software is selected, the AGIS PDA Cell phone is turned on, the GPS is automatically connected to the PDA / Cell phone through a Bluetooth interface and the AGIS Logo appears. Shortly thereafter the AGIS operation display appears. See Figures 1, 2, and 3</p>  <p>Figure 1 Figure 2 Figure 3</p> <p>Exhibit 4 to the Beyer Declaration, depicting AGIS PDA/cell phones.</p>
<p>identifying user interaction with the interactive display selecting the symbol corresponding to the second entity, and based thereon, displaying the information associated with the second entity.</p>	<p>The AGIS LifeRing product practices “identifying user interaction with the interactive display selecting the symbol corresponding to the second entity, and based thereon, displaying the information associated with the second entity.”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>For example, the The “AGIS operator selects to display and transmit information by touching switches drawn on the PDA display (a SoftSwitch) or by touching (hooking) symbols appearing on the LCD using a stylus or his/her finger. This action activates the SoftSwitch or</p>

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>causes information concerning the hooked item to appear in an auxiliary readout area at the bottom of the display.” Exhibit 4 at 3.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁵²⁷ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”⁵²⁸ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”⁵²⁹ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”⁵³⁰ and “num_on_que_list()”⁵³¹ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”⁵³² is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”⁵³³ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”⁵³⁴ and</p>

⁵²⁷ See, e.g., buttons.cpp at L1344-2365.

⁵²⁸ See, e.g., display/ftext.cpp at L176-244.

⁵²⁹ See, e.g., netselect.cpp at L152-221.

⁵³⁰ See, e.g., quelist.cpp at L16-22.

⁵³¹ See, e.g., quelist.cpp at L60-71.

⁵³² See, e.g., display/msgproc.cpp at L328-399.

⁵³³ See, e.g., netselect.cpp at L391-492.

⁵³⁴ See, e.g., netselect.cpp at L373-384.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>subsequently to “form_net_line()”⁵³⁵ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁵³⁶ and subsequently “send_net_msg_participants()”⁵³⁷ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁵³⁸ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_ddl_msg()”⁵³⁹ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”⁵⁴⁰ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
32. The method of claim 31, wherein the first device uses an Internet Protocol to transmit the user-specified symbol, location, and	See claim 31.

⁵³⁵ See, e.g., netselect.cpp at L352-370.⁵³⁶ See, e.g., display/msgproc.cpp at L1234-1292.⁵³⁷ See, e.g., display/msgproc.cpp at L828-922.⁵³⁸ See, e.g., photo.cpp at L189-292.⁵³⁹ See, e.g., server/msgproc.cpp at L377-474.⁵⁴⁰ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
information associated with the second entity.	
33. The method of claim 26, further comprising performing by the first device: adding data representing the spatial coordinates of the location of the second entity and data representing the information associated with the second entity to the database.	See claim 26.
34. The method of claim 24, wherein the portion of the interactive display is a first portion, wherein the position of the symbol corresponding to the particular second device is a first position, and wherein receiving the user input specifying the location of the second entity comprises: detecting user selection of a second portion of the interactive display	See claim 24.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p>corresponding to a second position on the second georeferenced map; and based at least in part on coordinates of the second position on the second georeferenced map and on the second georeferenced map data relating positions on the second georeferenced map to spatial coordinates, determining spatial coordinates of a location represented by the second position on the second georeferenced map, wherein the location represented by the second position is the location of the second entity.</p>	
<p>35. The method of claim 23, wherein the database is stored on the first device.</p> <p>36. The method of claim 23, wherein the database is stored on the first server.</p>	See claim 23.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p>37. The method of claim 1, further comprising performing by the first device: receiving user-specified information transmitted by a particular second device, the user-specified information including a user-specified location and a user-specified symbol corresponding to an entity other than the first device and the one or more second devices included in the group; and adding the user-specified symbol to the interactive display at a position on the second georeferenced map corresponding to the user-specified location.</p>	<p>The AGIS LifeRing product practices “receiving user-specified information transmitted by a particular second device, the user-specified information including a user-specified location and a user-specified symbol corresponding to an entity other than the first device and the one or more second devices included in the group; and adding the user-specified symbol to the interactive display at a position on the second georeferenced map corresponding to the user-specified location.”</p> <p>The device utilizes buttons on the display limited to other entities in a latitude and longitude limited space to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients.</p> <p>In the file “tact.cpp” the function “refresh_tact()”⁵⁴¹ is implemented to draw the tactical display including when some change to location or extent of the display is made. In this function a call is made to the function “track_draw()” to draw the extent of the tracks.</p> <p>In the file “tact.cpp” the function “track_draw()”⁵⁴² is implemented where based on a latitude and longitude value position of a track the function “screen_clip_check()” is called and if the entity is not on the display that the processing loop exits the loop not drawing the track leaving only tracks corresponding to other entities that are close to a selected position is drawn and subsequently the user can interact with.</p> <p>In the file “buttons.cpp” the function “button_actions()”⁵⁴³ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p>

⁵⁴¹ See, e.g., tact.cpp at L200-277.

⁵⁴² See, e.g., tact.cpp at L557-742.

⁵⁴³ See, e.g., buttons.cpp at L1344-2365.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>In the file “display/ftext.cpp” the function “create_ftext()”⁵⁴⁴ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the “build_sending_text()”⁵⁴⁵ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”⁵⁴⁶ and “num_on_que_list()”⁵⁴⁷ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”⁵⁴⁸ is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”⁵⁴⁹ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”⁵⁵⁰ and subsequently to “form_net_line()”⁵⁵¹ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁵⁵² and subsequently “send_net_msg_participants()”⁵⁵³ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁵⁵⁴ is implemented where the</p>

⁵⁴⁴ See, e.g., display/ftext.cpp at L176-244.⁵⁴⁵ See, e.g., netselect.cpp at L152-221.⁵⁴⁶ See, e.g., quelist.cpp at L16-22.⁵⁴⁷ See, e.g., quelist.cpp at L60-71.⁵⁴⁸ See, e.g., display/msgproc.cpp at L328-399.⁵⁴⁹ See, e.g., netselect.cpp at L391-492.⁵⁵⁰ See, e.g., netselect.cpp at L373-384.⁵⁵¹ See, e.g., netselect.cpp at L352-370.⁵⁵² See, e.g., display/msgproc.cpp at L1234-1292.⁵⁵³ See, e.g., display/msgproc.cpp at L828-922.⁵⁵⁴ See, e.g., photo.cpp at L189-292.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p> <p>In the file “server/msgproc.cpp” the function “process_csif_dll_msg()”⁵⁵⁵ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the “pass_thru_msg()” function. In the file “server/msgproc.cpp” the function “pass_thru_msg()”⁵⁵⁶ is implemented where the “trouting” variable is parsed to populate the address list for the recipients using the “parse_routing()” function followed by the call to “send_message()” to each individual in the list.</p> <p>As described above the final leg of transmission is done by the “pass_thru_msg()” function implemented in the “server/msgproc.cpp” file.</p>
38. The method of claim 37, further comprising performing by the first device: identifying user interaction with the interactive display selecting the user-specified symbol corresponding to the entity, and based thereon, displaying information associated with the entity, wherein the user-specified information further includes the information	See claim 37.

⁵⁵⁵ See, e.g., server/msgproc.cpp at L377-474.⁵⁵⁶ See, e.g., server/msgproc.cpp at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
associated with the entity.	
39. The method of claim 1, wherein the message including the identifier corresponding to the group is a first message, and wherein the method further comprises performing by the first device: sending, to a particular second device via the first server, a second message related to remotely controlling the particular second device to perform an action, wherein the particular second device is configured to perform the action based on receiving the second message.	See claim 1.
40. The method of claim 39, wherein the second message indicates the action to be performed, and wherein the action is selected from the group consisting of	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
playing audio, initiating a phone call, vibrating, converting text to speech, changing sound intensity, and displaying information.	
41. The method of claim 40, wherein playing audio comprises playing an audio message announcing an emergency.	See claim 1.
42. The method of claim 1, wherein the message including the identifier corresponding to the group is a first message, and wherein the method further comprises performing by the first device: receiving a second message sent by a particular second device, wherein the second message indicates an action to be performed by the first device; and performing the indicated action.	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
43. The method of claim 42, wherein the indicated action is selected from the group consisting of playing audio, initiating a phone call, vibrating, converting text to speech, changing sound intensity, and displaying information.	See claim 1.
44. The method of claim 1, further comprising performing by the first device: presenting another symbol on the second georeferenced map corresponding to a fixed location and associated with a telephone number; and receiving user selection of the other symbol and, based thereon, initiating a telephone call to the telephone number associated with the symbol.	<p>The AGIS LifeRing product practices “presenting another symbol on the second georeferenced map corresponding to a fixed location and associated with a telephone number; and receiving user selection of the other symbol and, based thereon, initiating a telephone call to the telephone number associated with the symbol.”</p> <p>The device utilizes buttons on the display to send messages, make calls and send photo or video.</p> <p>The device transmits a message to the server using a participant list and the server makes the final delivery of messages and the device does not have access to the Internet Protocol addresses of the recipients. In the file “buttons.cpp” the function “button_actions()”⁵⁵⁷ implements the main routine for capturing the button press event and processing based on the button pressed. For example, cases for “BUTTON_MSG_FREETEXT”, “BUTTON_NET”, “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” perform an action to use Internet Protocol to send data via a server. Calls are made to “create_ftext()”, “create_netlist()” and “create_photo()” perform these actions based on the user action.</p> <p>In the file “display/ftext.cpp” the function “create_ftext()”⁵⁵⁸ is implemented where the display is configured to receive user interface for communicating through free text. The call to “build_sending_text()” builds the receiving participants and in file “netselect.cpp” the</p>

⁵⁵⁷ See, e.g., buttons.cpp at L1344-2365.⁵⁵⁸ See, e.g., display/ftext.cpp at L176-244.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>“build_sending_text()”⁵⁵⁹ function is implemented. List of participants is organized in the “indiv” array or the que_list based on the usage of the build and que_list is used via the “on_que_list()” and “num_on_que_list()” functions. The file “quelist.cpp” implements the “on_que_list()”⁵⁶⁰ and “num_on_que_list()”⁵⁶¹ functions where the “que_list” array is maintained. Once the message is compiled the message is send via “send_ftext_msg()” function. In the file “display/msgproc.cpp” the function “send_ftext_msg()”⁵⁶² is implemented where the free text message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>Similarly for the case of “BUTTON_NET” a call is made to “create_netlist()”. In the file netselect.cpp” the function “create_netlist()”⁵⁶³ is implemented where the display is configured for net communication. The call to “send_to_display_net_list()”⁵⁶⁴ and subsequently to “form_net_line()”⁵⁶⁵ forms the list of participants to receive the communication. In the file “display/msgproc.cpp” the function “process_net_msg()”⁵⁶⁶ and subsequently “send_net_msg_participants()”⁵⁶⁷ the message is transmitted directly or through a server as described above through the use of the “send_message_direct()” and “send_message_bulk()” using the “server_ip” variable.</p> <p>For the case of “BUTTON_MSG_PHOTO” and “BUTTON_MSG_VIDEO” the call to “create_photo()” is made. In the file “photo.cpp” the function “create_photo()”⁵⁶⁸ is implemented where the directly listing of either photo or video files are gathered and displayed to the user eventually followed by the call to the function “build_sending_text()” similar to the free text mode above.</p>

⁵⁵⁹ See, e.g., netselect.cpp at L152-221.

⁵⁶⁰ See, e.g., quelist.cpp at L16-22.

⁵⁶¹ See, e.g., quelist.cpp at L60-71.

⁵⁶² See, e.g., display/msgproc.cpp at L328-399.

⁵⁶³ See, e.g., netselect.cpp at L391-492.

⁵⁶⁴ See, e.g., netselect.cpp at L373-384.

⁵⁶⁵ See, e.g., netselect.cpp at L352-370.

⁵⁶⁶ See, e.g., display/msgproc.cpp at L1234-1292.

⁵⁶⁷ See, e.g., display/msgproc.cpp at L828-922.

⁵⁶⁸ See, e.g., photo.cpp at L189-292.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
	<p>See also <code>BUTTON_CALL</code> and <code>BUTTON_CCALL</code> in <code>buttons.h.</code> and <code>buttons.cpp</code> for phone calls and conferences.</p> <p>In the file <code>server/msgproc.cpp</code> the function <code>“process_csif_ddl_msg()”</code>⁵⁶⁹ is implemented as described above with various conditions. For the default condition catching messages for Free Text, Net, Photo and Video messaging, the code calls the <code>“pass_thru_msg()”</code> function. In the file <code>server/msgproc.cpp</code> the function <code>“pass_thru_msg()”</code>⁵⁷⁰ is implemented where the <code>“trouting”</code> variable is parsed to populate the address list for the recipients using the <code>“parse_routing()”</code> function followed by the call to <code>“send_message()”</code> to each individual in the list.</p> <p>As described above the final leg of transmission is done by the <code>“pass_thru_msg()”</code> function implemented in the <code>“server/msgproc.cpp”</code> file.</p>
45. The method of claim 1, further comprising performing, by the first device: presenting a symbol corresponding to a facility, wherein the facility is selected from the group consisting of a hospital, a police station, and a fire station, and wherein	See claim 1.

⁵⁶⁹ See, e.g., `server/msgproc.cpp` at L377-474.⁵⁷⁰ See, e.g., `server/msgproc.cpp` at L 51-61.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
the symbol corresponding to the facility is positioned on the second georeferenced map at a position corresponding to a location of the facility.	
46. The method of claim 45, further comprising performing, by the first device: identifying user interaction with the interactive display selecting the symbol corresponding to the facility, and based thereon, displaying information associated with the facility.	See claim 1.
47. The method of claim 46, wherein the information associated with the facility comprises a uniform resource locator (URL) of a web site associated with the facility.	See claim 1.
48. The method of claim 45, further comprising performing, by the	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
first device: identifying user interaction with the interactive display selecting the symbol corresponding to the facility and user interaction with the display specifying an action, and based thereon, loading a web page associated with the facility.	
49. The method of claim 1, further comprising performing by the first device: identifying user interaction with the interactive display selecting a subset of the user-selectable symbols corresponding to a subset of the one or more second devices included in the group; and identifying user interaction with the interactive display specifying an action and, based thereon, assigning the subset of second devices to a sub-net.	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
50. The method of claim 49, further comprising performing by the first device: identifying user interaction with the interactive display selecting the sub-net and user interaction with the display specifying an action; and based thereon, sending fourth data to the subset of second devices via the first server or initiating a phone conference with the subset of second devices.	
51. The method of claim 1, wherein the first server is the second server.	See claim 1.
52. The method of claim 1, wherein the first set of second devices and the second set of second devices are identical.	See claim 1.
53. The method of claim 1, wherein the message further includes an identifier	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
corresponding to the first device.	
54. A system comprising: a first device programmed to perform operations comprising:	See claim 1.
joining a communication network corresponding to a group, wherein joining the communication network comprises transmitting a message including an identifier corresponding to the group;	See claim 1.
participating in the group, wherein participating in the group includes sending first location information to a first server and receiving second location information from the first server, the first location information comprising a location of the first device, the second location information comprising one or more locations of	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
one or more respective second devices included in the group;	
presenting, via an interactive display of the first device, a first interactive, georeferenced map and a first set of one or more user- selectable symbols corresponding to a first set of one or more of the second devices, wherein the first set of symbols are positioned on the first georeferenced map at respective positions corresponding to the locations of the first set of second devices, and wherein first georeferenced map data relate positions on the first georeferenced map to spatial coordinates;	See claim 1.
sending, to a second server, a request for second georeferenced map data different from the first georeferenced map	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
data; receiving, from the second server, the second georeferenced map data;	
presenting, via the interactive display of the first device, a second georeferenced map and a second set of one or more user- selectable symbols corresponding to a second set of one or more of the second devices, wherein the second set of symbols are positioned on the second georeferenced map at respective positions corresponding to the locations of the second set of second devices, and wherein the second georeferenced map data relate positions on the second georeferenced map to spatial coordinates;	See claim 1.
and identifying user interaction with the interactive display selecting one or more of the second	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
set of user-selectable symbols corresponding to one or more of the second devices and positioned on the second georeferenced map and user interaction with the display specifying an action and, based thereon, sending third data to the selected one or more second devices via the first server.	
55. A computer-implemented method comprising: performing, by a server: receiving, from a first device, a message including an identifier corresponding to a group;	See claim 1.
permitting the first device to join a communication network corresponding to the group, wherein permitting the first device to join the communication network comprises:	See claim 1.
receiving first location information	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p>from the first device, the first location information comprising a location of the first device, sending the first location information to one or more second devices included in the group, receiving second location information from the one or more second devices, the second location information comprising one or more locations of the respective one or more second devices, and sending the second location information to the first device, wherein the first device is configured to present, via an interactive display of the first device, a first interactive, georeferenced map and a first set of one or more user-selectable symbols corresponding to a first set of one or more of the second devices, wherein the first set of symbols are positioned on the first</p>	

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
georeferenced map at respective positions corresponding to the locations of the first set of second devices, and wherein first georeferenced map data relate positions on the first georeferenced map to spatial coordinates;	
receiving, from the first device, a request for second georeferenced map data different from the first georeferenced map data;	See claim 1.
sending, to the first device, the second georeferenced map data, wherein the first device is configured to present, via the interactive display of the first device, a second georeferenced map and a second set of one or more user-selectable symbols corresponding to a second set of one or more of the second devices, wherein the	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
second set of symbols are positioned on the second georeferenced map at respective positions corresponding to the locations of the second set of second devices, and wherein the second georeferenced map data relate positions on the second georeferenced map to spatial coordinates;	
receiving, from the first device, (1) third data, and (2) fourth data indicating user selection of one or more of the second set of user-selectable symbols corresponding to one or more of the second devices;	See claim 1.
and based on receiving the third data and the fourth data, sending the third data to the selected one or more second devices.	See claim 1.
56. The method of claim 55, wherein	See claim 55.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
sending the first location information to the one or more second devices included in the group comprises pushing the first location information to the one or more second devices.	
57. The method of claim 55, wherein an Internet Protocol (IP) address of the server is accessible to the first device and the one or more second devices.	See claim 55.
58. The method of claim 57, further comprising performing by the server: storing an IP address of the first device; receiving, from a particular second device, a request to send fifth data to the first device, wherein the request to send the fifth data to the first device identifies the first device using an identifier corresponding to the first device; and sending the fifth data to the first	See claim 55.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
device in a message addressed to the stored IP address of the first device.	
59. The method of claim 55, wherein the group comprises a plurality of group members permitted to communicate with each other via the communication network.	See claim 3.
60. The method of claim 55, wherein receiving the third data from the first device comprises using an Internet Protocol to receive the third data to the first device.	See claim 4.
61. The method of claim 55, wherein the third data include a short message service message, a text message, an image, or a video.	See claim 6.
62. The method of claim 61, wherein the video comprises a video clip.	See claim 7.
63. The method of claim 61, wherein the third data	See claim 8.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
include a voice recording.	
64. The method of claim 55, wherein sending the third data to the selected one or more second devices comprises transmitting a text message to at least one of the selected one or more second devices using an Internet Protocol (IP).	See claim 9.
65. The method of claim 55, further comprising performing by the server: receiving, from the first device, fifth data indicating user selection of at least one symbol in the second set of user-selectable symbols corresponding to at least one of the second devices; and based on receiving the fifth data, initiating voice-over-IP (VOIP) communication between the first device and the at least one second device.	See claim 11.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
66. The method of claim 1, wherein receiving the first location information from the first device comprises using an Internet Protocol (IP) to receive the first location information from the first device.	See claim 1
67. The method of claim 66, wherein receiving the first location information from the first device further comprises receiving the first location information via the Internet.	See claim 1.
68. The method of claim 55, wherein permitting the first device to join the communication network further comprises: receiving first status information from the first device and sending second status information to the first device, wherein the first status information comprises data indicative of a battery level of the first device, a signal	See claim 19.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p>strength of a wireless signal of the first device, a status of a Global Positioning Satellite (GPS) receiver of the first device, or a combination thereof, and wherein the second status information comprises data indicative of one or more battery levels of the respective one or more second devices included in the group, one or more signal strengths of wireless signals of the respective one or more second devices included in the group, one or more statuses of GPS receivers of the respective one or more second devices included in the group, or a combination thereof.</p>	
69. The method of claim 55, wherein the second georeferenced map data comprise a satellite image or aerial photograph.	See claim 20.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
70. The method of claim 55, wherein the spatial coordinates comprise latitude and longitude coordinates.	
71. The method of claim 55, further comprising performing by the server: receiving, from the first device, fifth data comprising a user-specified location and a user-specified symbol corresponding to an entity other than the first device and the one or more second devices included in the group; based on receiving the fifth data, transmitting the user-specified symbol and location of the entity to the one or more second devices included in the group for addition of the user-specified symbol to respective interactive displays of the one or more second devices at respective positions on respective georeferenced maps	See claims 24-25.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
corresponding to the user-specified location of the entity.	
72. The method of claim 71, wherein the fifth data further comprises user-specified information associated with the entity, and wherein the method further comprises performing by the server: based on receiving the fifth data, transmitting the user-specified information associated with the entity to the one or more second devices included in the group.	See claim 26.
73. The method of claim 72, wherein the information comprises a category of the second entity.	See claim 27.
74. The method of claim 73, wherein the category comprises a vehicle, a person, an event, a	See claim 28.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
site, a building, or a facility.	
75. The method of claim 72, wherein the information comprises an image.	See claim 29.
76. The method of claim 72, wherein the information comprises at least one type of information selected from the group consisting of text and video.	See claim 30
77. The method of claim 72, wherein the first device uses an Internet Protocol to transmit the user-specified symbol, location, and information associated with the entity.	See claim 32.
78. The method of claim 71, further comprising performing by the server: adding data representing spatial coordinates of the location of the entity and data representing the information associated with the entity to a database.	See claim 33.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
79. The method of claim 1, wherein the message including the identifier corresponding to the group is a first message, and wherein the method further comprises performing by the server: receiving, from the first device, a second message related to remotely controlling a particular second device to perform an action; and after receiving the second message, sending, to the particular second device, a third message related to remotely controlling the particular second device to perform the action, wherein the particular second device is configured to perform the action based on receiving the third message.	See claim 39.
80. The method of claim 79, wherein the second message indicates the action	See claim 40.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
to be performed, and wherein the action is selected from the group consisting of playing audio, initiating a phone call, vibrating, converting text to speech, changing sound intensity, and displaying information.	
81. The method of claim 80, wherein playing audio comprises playing an audio message announcing an emergency.	See claim 41.
82. The method of claim 55, wherein the first set of second devices and the second set of second devices are identical.	See claim 52.
83. The method of claim 55, wherein the message further includes an identifier corresponding to the first device.	See claim 53.
84. A system comprising: one or more servers programmed to	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
perform operations comprising:	
receiving, from a first device, a message including an identifier corresponding to a group;	See claim 1.
permitting the first device to join a communication network corresponding to the group, wherein permitting the first device to join the communication network comprises:	See claim 1.
receiving first location information from the first device, the first location information comprising a location of the first device, sending the first location information to one or more second devices included in the group, receiving second location information from the one or more second devices, the second location information comprising one or more locations of	See claim 1.

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p>the respective one or more second devices, and sending the second location information to the first device, wherein the first device is configured to present, via an interactive display of the first device, a first interactive, georeferenced map and a first set of one or more user-selectable symbols corresponding to a first set of one or more of the second devices, wherein the first set of symbols are positioned on the first georeferenced map at respective positions corresponding to the locations of the first set of second devices, and wherein first georeferenced map data relate positions on the first georeferenced map to spatial coordinates;</p>	
receiving, from the first device, a request for second georeferenced map	See claim 1.

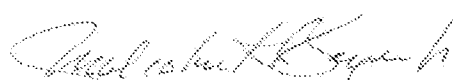
U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
<p>data different from the first georeferenced map data; sending, to the first device, the second georeferenced map data, wherein the first device is configured to present, via the interactive display of the first device, a second georeferenced map and a second set of one or more user-selectable symbols corresponding to a second set of one or more of the second devices, wherein the second set of symbols are positioned on the second georeferenced map at respective positions corresponding to the locations of the second set of second devices, and wherein the second georeferenced map data relate positions on the second georeferenced map to spatial coordinates; receiving, from the first device, (1) third</p>	

Attorney Docket No. 2525.996REX0
 Control No.: 90/014,510 (Re-exam of U.S. Patent No. 9,467,838)

U.S. Patent No. 9,467,838	October 22, 2005 AGIS LifeRing
data, and (2) fourth data indicating user selection of one or more of the second set of user-selectable symbols corresponding to one or more of the second devices;	
and based on receiving the third data and the fourth data, sending the third data to the selected one or more second devices.	See claim 1.

I declare under the penalty of perjury that the foregoing is true and correct to the best of my knowledge.

Executed this 24th day of March, 2021 in Jupiter, FL.



Malcolm K. Beyer, Jr.