



(19) **United States**

(12) **Patent Application Publication**

Mehta et al.

(10) **Pub. No.: US 2003/0028671 A1**

(43) **Pub. Date:**

Feb. 6, 2003

(54) **METHOD AND SYSTEM FOR TWO-WAY INITIATED DATA COMMUNICATION WITH WIRELESS DEVICES**

(75) Inventors: **Samir Narendra Mehta**, Renton, WA (US); **Mazin Ramadan**, Seattle, WA (US); **Geoffrey S. Heller**, Seattle, WA (US); **Vineet R. Sharma**, Renton, WA (US); **Markus L. Jansen**, Renton, WA (US); **Edward L. Gow**, Seattle, WA (US); **Ngochan T. Nguyen**, Seattle, WA (US)

Correspondence Address:
SEED INTELLECTUAL PROPERTY LAW GROUP PLLC
701 FIFTH AVE
SUITE 6300
SEATTLE, WA 98104-7092 (US)

(73) Assignee: **4th pass Inc.**, Seattle, WA

(21) Appl. No.: **10/167,240**

(22) Filed: **Jun. 10, 2002**

Related U.S. Application Data

(60) Provisional application No. 60/296,902, filed on Jun. 8, 2001.

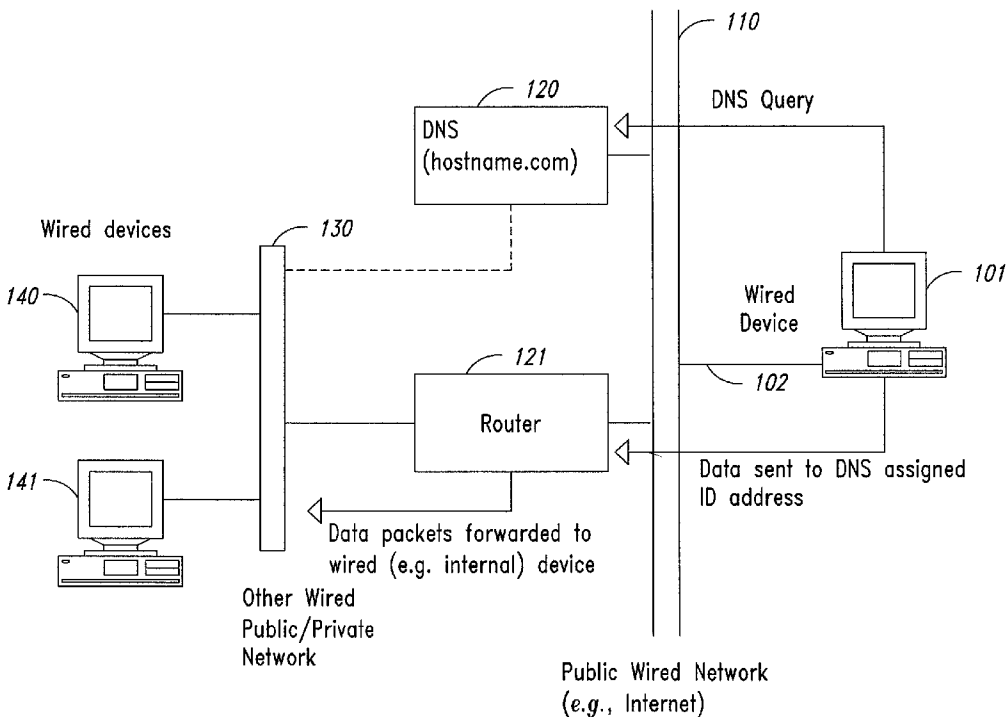
Publication Classification

(51) Int. Cl.⁷ **G06F 15/16**

(52) **U.S. Cl.** **709/245; 709/230**

(57) **ABSTRACT**

Methods and systems for providing two-way initiated, bi-directional communication with wireless devices using connection-based or connection-less protocols, such as, for example, TCP/IP and UDP/IP, are provided. Example embodiments provide an Address Management Proxy System (“AMPS”), which enables devices and systems connected to a public internet, such as the Internet, to initiate communication with and to send data to wireless devices connected to a private wireless network, without exposing the non-routable private addresses of these wireless devices. The AMPS allocates a public (routable) network address for temporarily use by a requesting device on a public network to communicate with a wireless device on a wireless network. In one embodiment, a pool of public addresses, for example, public IP addresses, is maintained by the AMPS and allocated dynamically to wireless network devices as required. In one embodiment, the AMPS comprises one or more modified DNS/API servers, one or more Address Proxy/Routers, an Address Management Data Server, one or more data repositories, and optionally a load balancer. The AMPS DNS’/API server receives a request from a device on a public network for a particular wireless device, and returns an appropriate temporary public address, which is internally mapped to the private address of the wireless device. The public address is then usable by the device on the public network to send data to the wireless device.



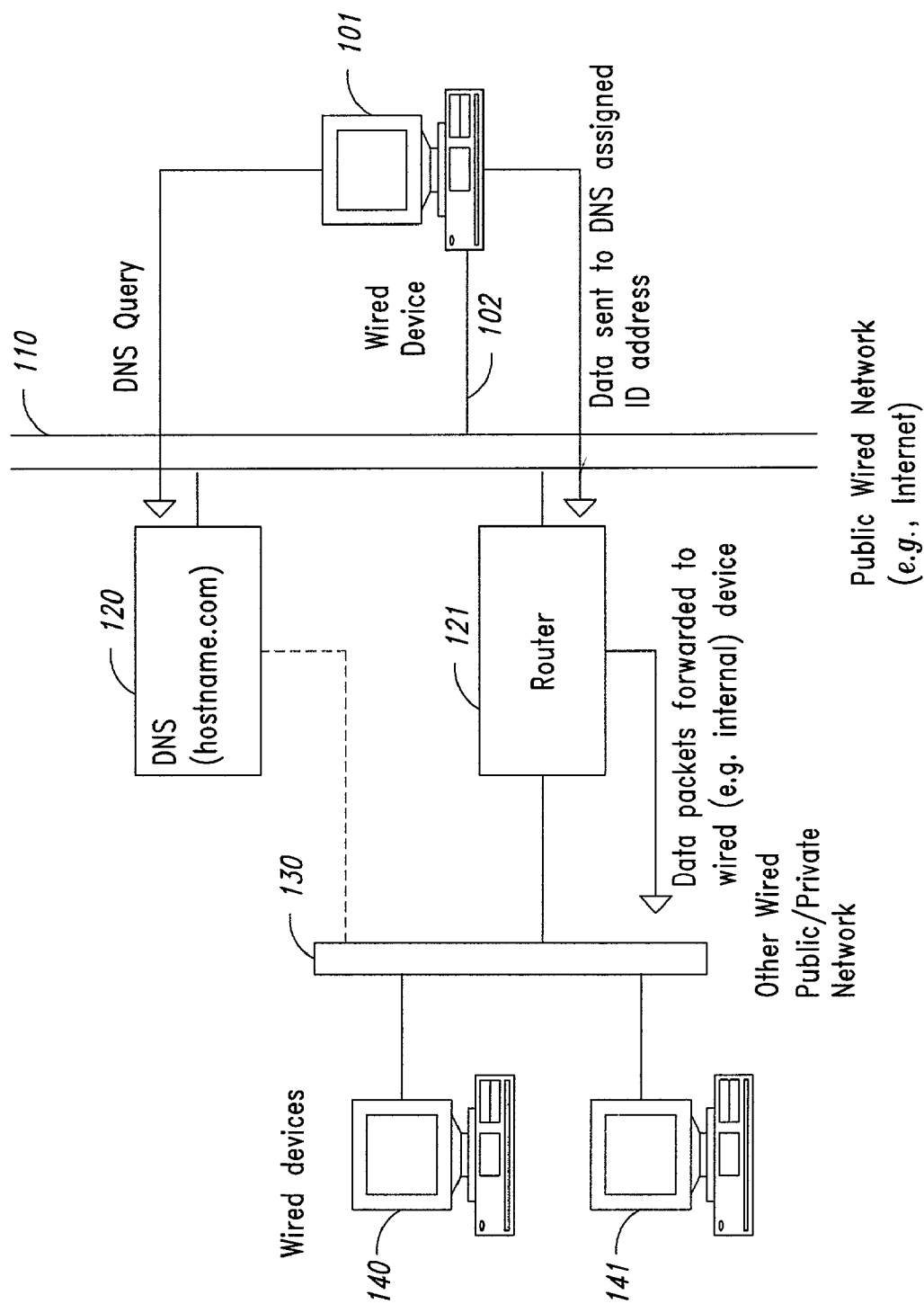


Fig. 1

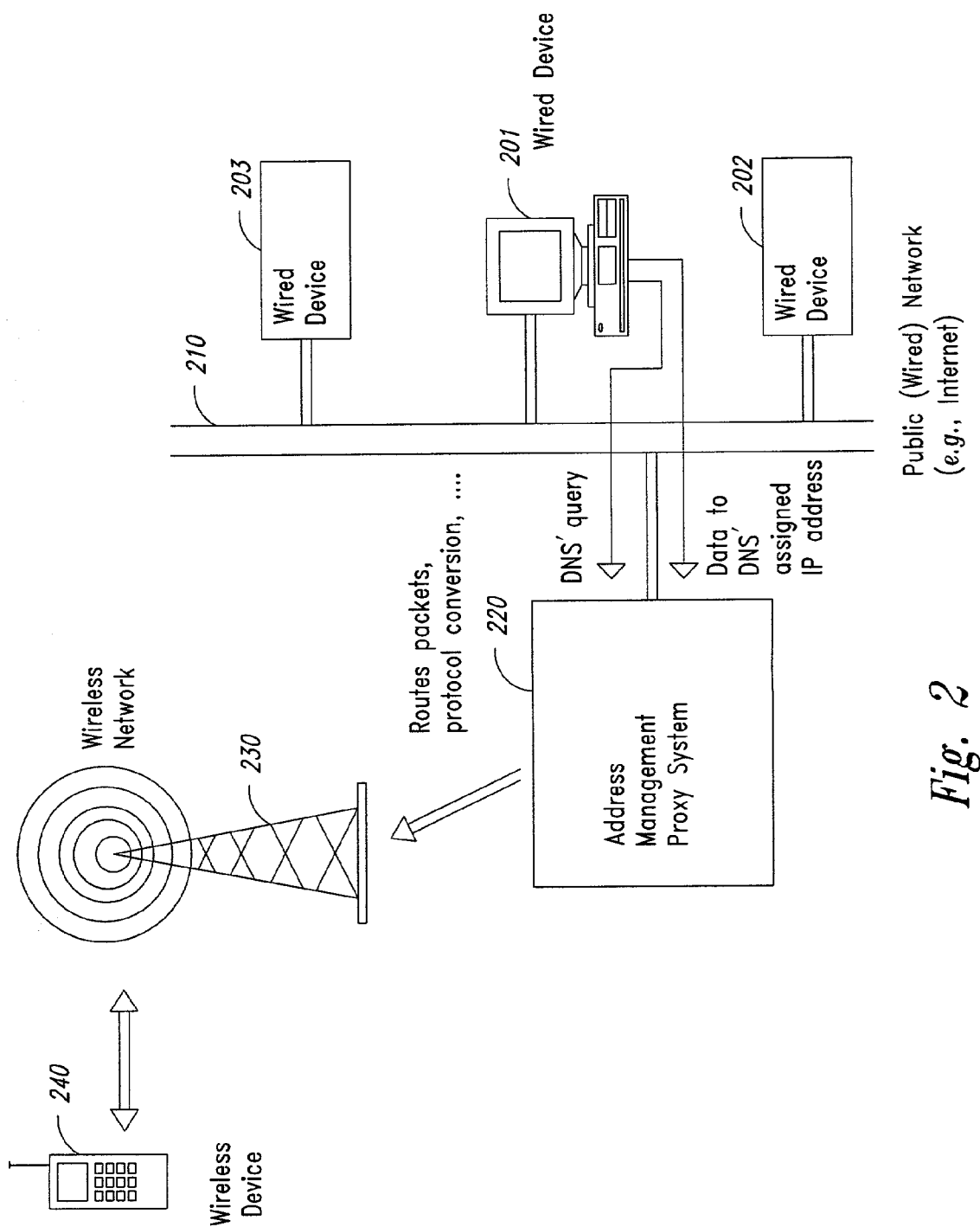


Fig. 2

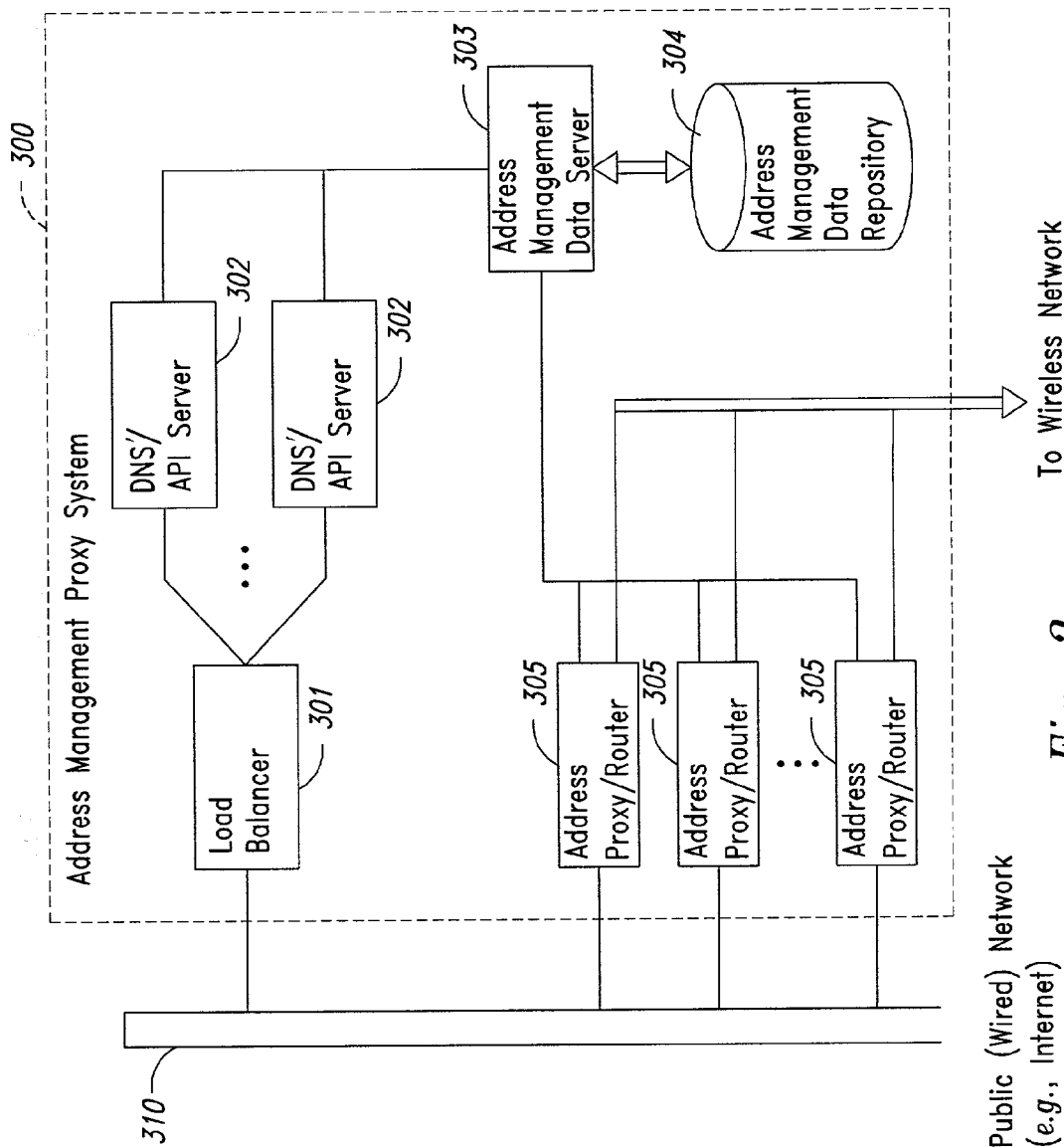


Fig. 3

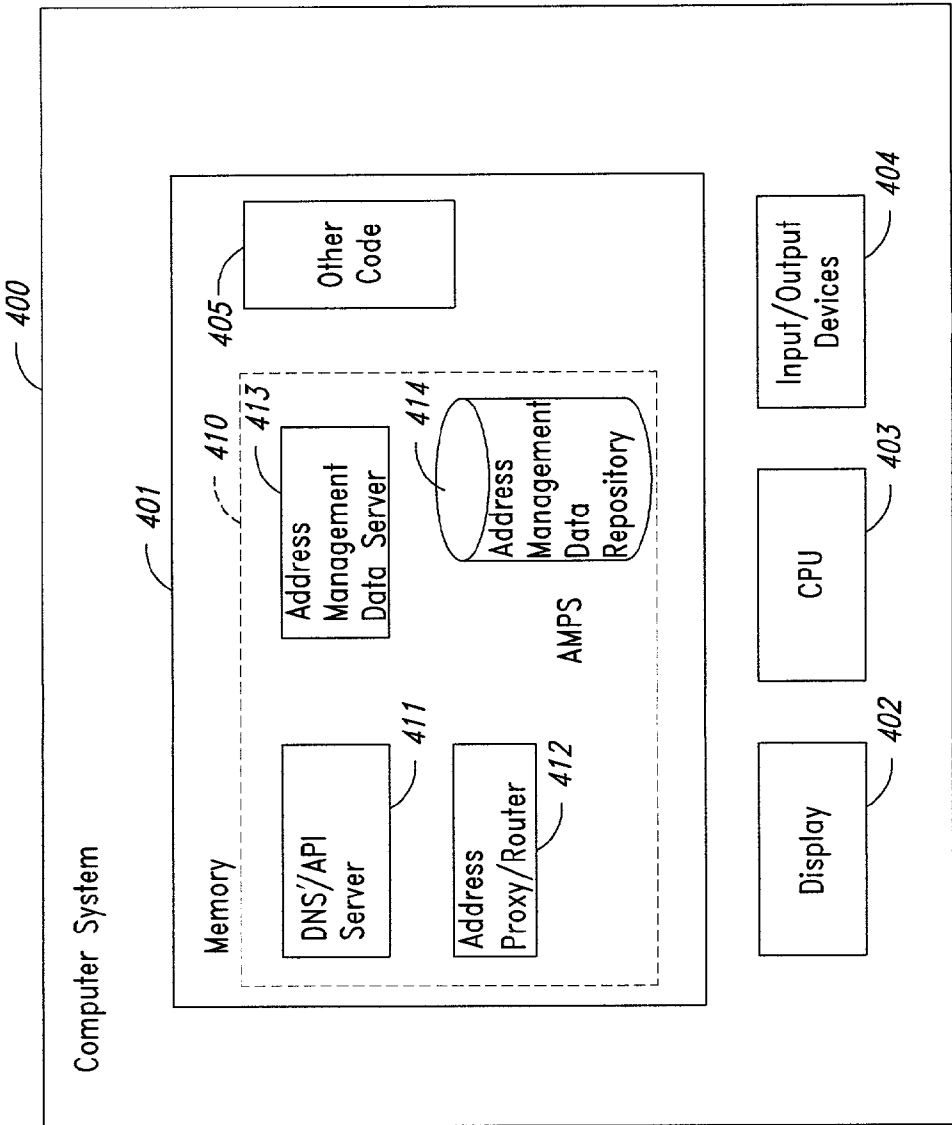


Fig. 4

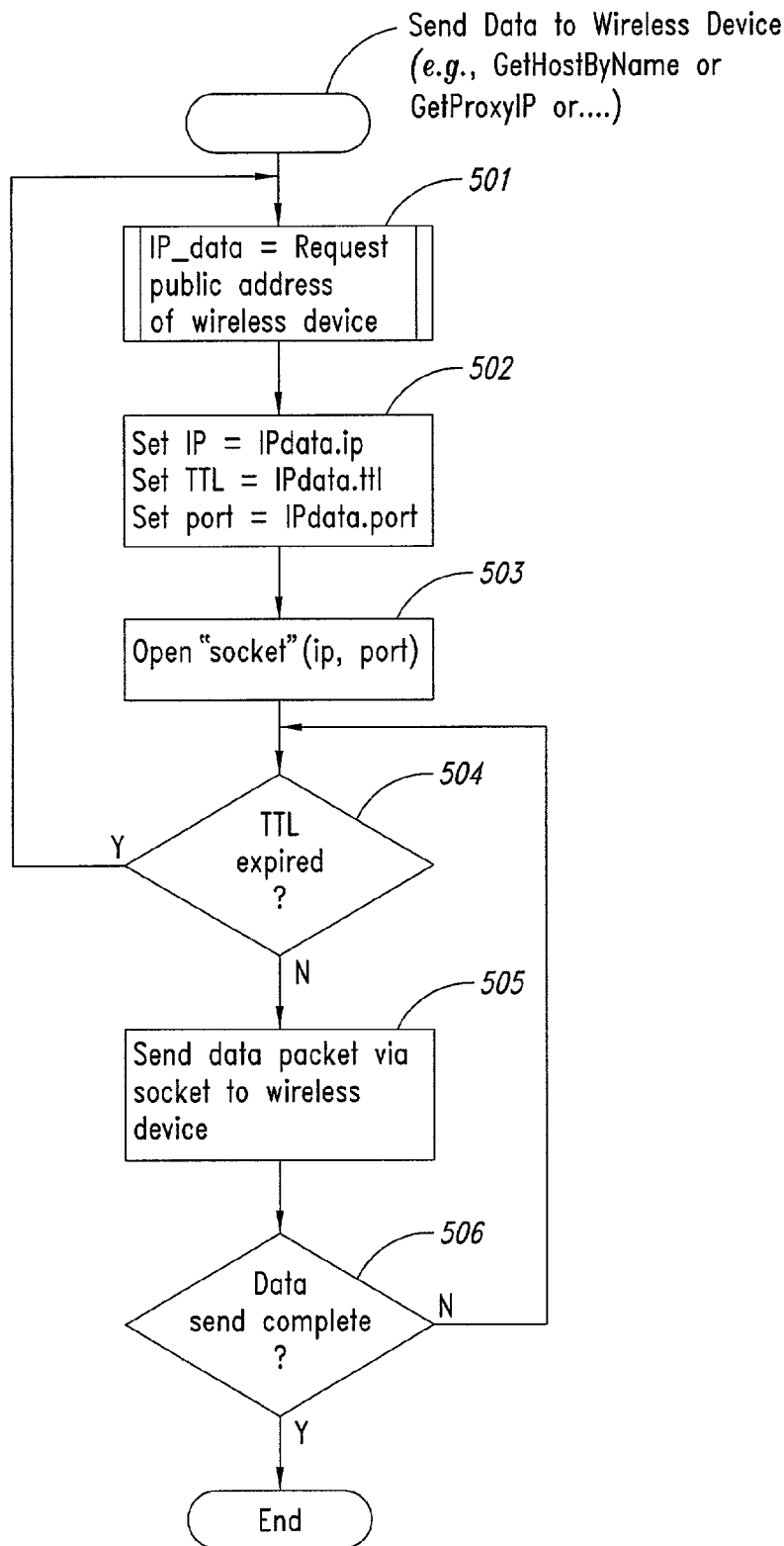


Fig. 5

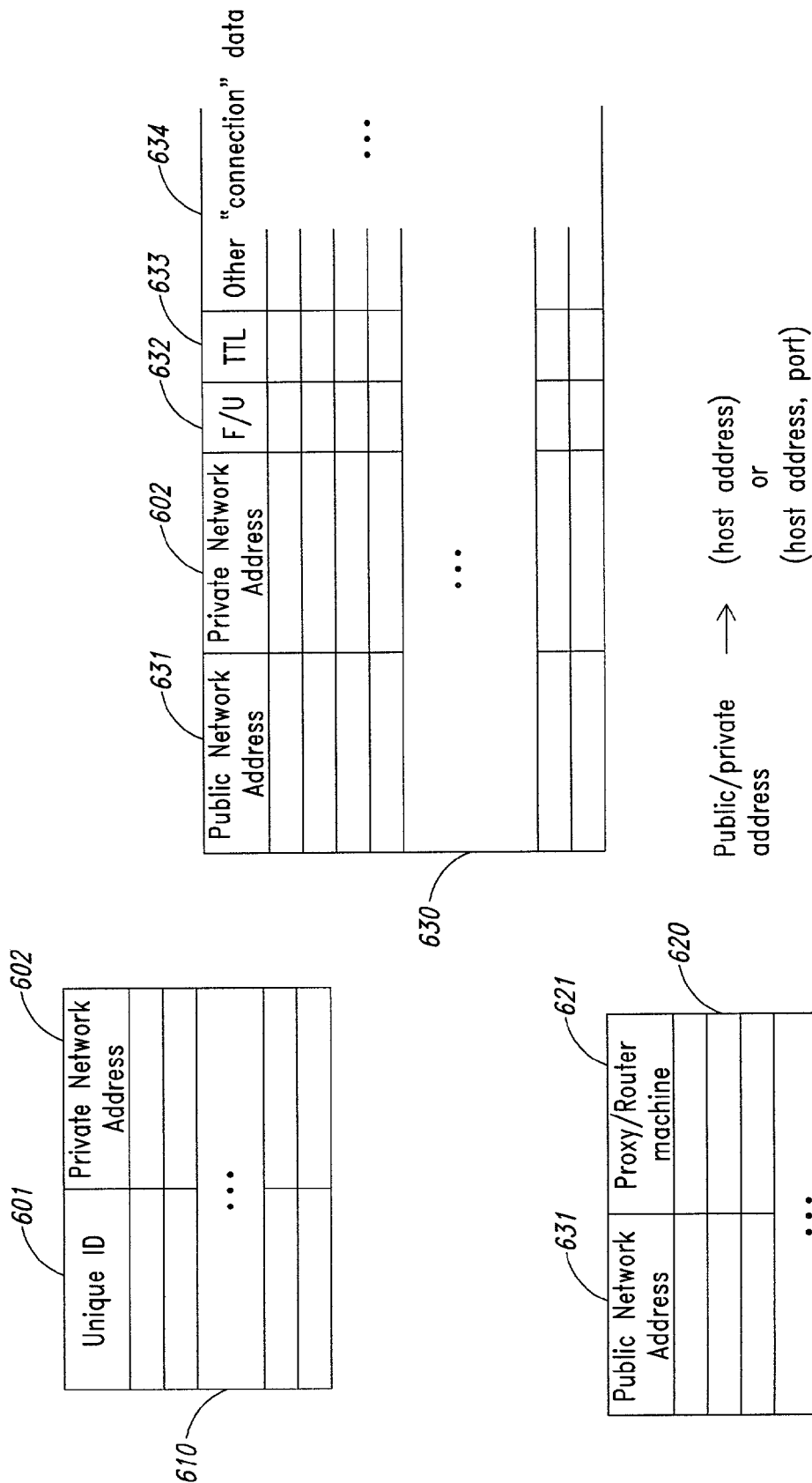
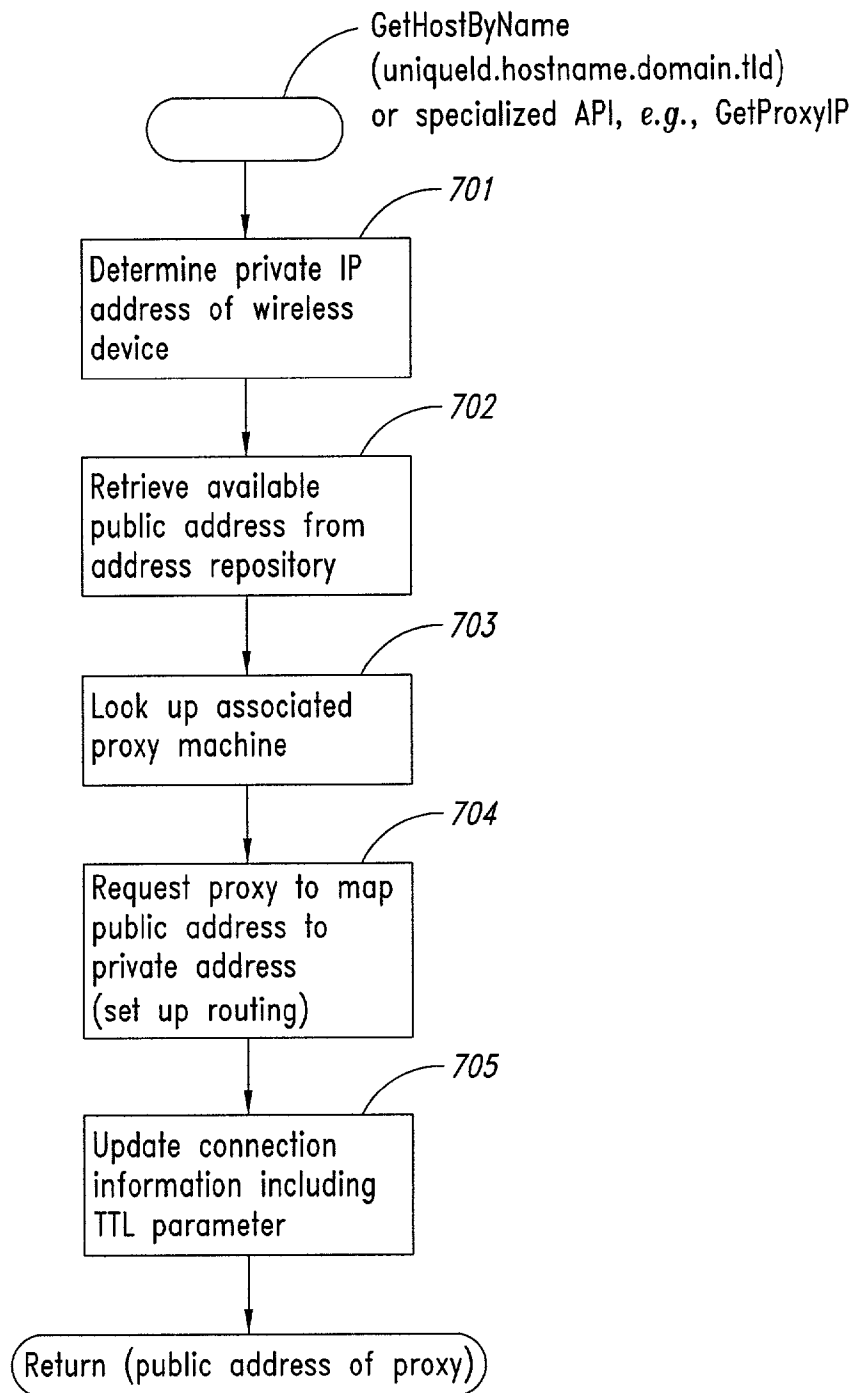


Fig. 6

*Fig. 7*

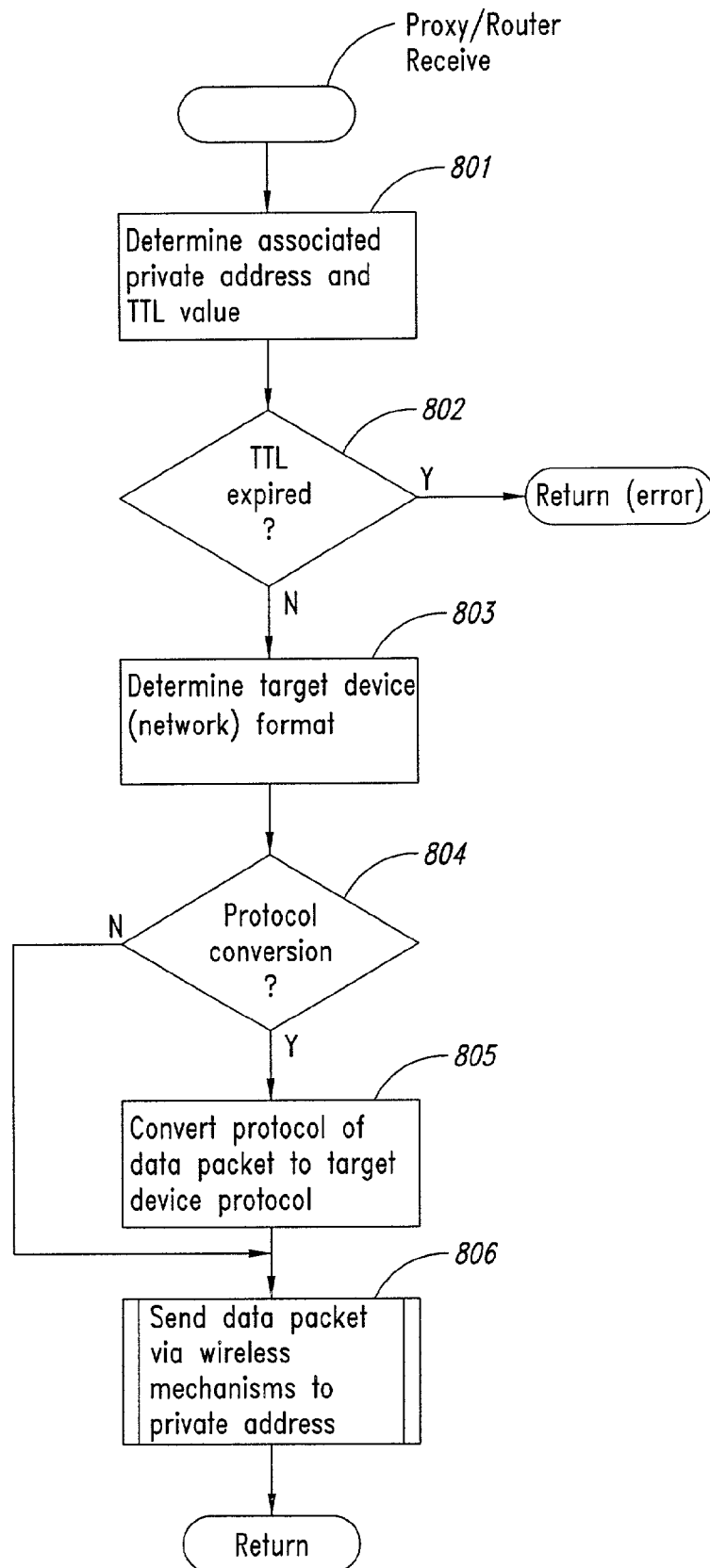


Fig. 8

METHOD AND SYSTEM FOR TWO-WAY INITIATED DATA COMMUNICATION WITH WIRELESS DEVICES

BACKGROUND OF THE INVENTION

[0001] 1. Field of the Invention

[0002] The present invention relates to methods and systems for initiating communication with wireless devices and, in particular, to methods and systems for initiating communication with a device on a private network from a device on a public network to achieve virtual end-to-end connectivity.

[0003] 2. Background Information

[0004] The need to connect networked computers together with other networks of computers has become increasingly important as our global society strives to communicate for business and personal reasons on an international level. This need has driven the technical community to devise ways to interconnect devices on different physical networks so that devices on heterogeneous and homogeneous networks can communicate with one another as in a uniform (or standard) manner. Such interconnection technology is often referred to as “internetworking,” “internetworks,” or simply “internets.” One such global internetworking implementation is the Internet (first letter capitalized to distinguish from “internets” generally), as originally developed by the ARPA, NSF, etc. for educational and government purposes. The Internet, in its now expanded version, exists as a complex infrastructure of network backbones that connect to networks around the world. Other (local, regional, private, or public) networks connect to the Internet backbone by means of gateways or routers (also known as gateway servers, gateway systems, router servers, and router systems). For purposes of this document, the term “router” or “gateway,” used alone or to modify another noun, are used interchangeably and will refer generically to any hardware or software, subsystem, system, or code that is able to transfer a data packet (at any level of the network or physical data transfer) between two or more homogeneous or heterogeneous machines, systems, or subsystems, without regard to a particular gateway/router machine or service. In the Internet environment, the presence of a router or gateway typically indicates an ability to route an Internet datagram to another router or machine. Machines intended to run applications are generally referred to using networking terminology as “hosts” or “host machines;” for purposes herein, a router/gateway may be a function of a host machine or may be a separate device.

[0005] In the Internet environment, the various networks and devices communicate using standard network protocols and models, such as TCP/IP (or UDP/IP). Although for the purpose of this document, the reader is presumed to be familiar with networking concepts, protocols, and standards, a brief summary is provided herein for ease. Extensive background information of internetworking, internets, the Internet, and related terms and background technology can be found in Tanenbaum, A., *Computer Networks*, Prentice-Hall, N.J., 3d ed. 1996, and in Comer, D., *Internetworking with TCP/IP, Principles, Protocols, and Architectures*, Prentice Hall, N.J., 4th ed. 2000, which are herein incorporated by reference in their entirety.

[0006] TCP/IP and UDP/IP (or often referred to as “UDP”) refer to a popular transport layer and network layer

protocol combinations, which are commonly found in inter-networking environments (other protocols are available at each of these layers as well). The “IP” in TCP/IP, refers to “Internet Protocol,” and is a network layer protocol that defines how data is organized and represented (the format and meaning of datagrams) and the communications transactions used to route them. TCP (“Transmission Control Protocol”) and UDP (“User Datagram Protocol”), which are both transport layer protocols that can be implemented on top of an IP protocol, support connection-based and connectionless data transfer, respectively. Connection-based data transfer implies that a source and destination device request a connection and then send data to each other over the “virtual” connection in a manner that is sequenced (and typically involves handshaking mechanisms). The term “virtual” here implies that, although the connection is not necessarily hard-wired, the connection appears to provide a direct conduit between a source (requestor) and destination (receiver). The data actually flows through lower protocol layers to be eventually transmitted over a physical transmission medium. Connection-less data transfer implies that the source sends a datagram to the destination, but doesn’t insure delivery or that data will arrive in a particular order.

[0007] The IP protocol also defines a uniform network addressing mechanism, so that machines that implement protocols that use IP, such as TCP or UDP, are able to specify sources and destinations for their data transfers. The network addressing mechanism specifies an abstract address and a binding protocol for mapping the abstract address to a physical hardware address. For purposes of this description, the uniform network addressing mechanism will be referred to generically as an “IP address.” IP addresses are often written in “dotted decimal” notation, which specifies an address as “w.x.y.z” (e.g., in a 32-bit address, each letter designates a byte of information). For example, an IP address such as “192.251.0.1” may specify a host machine at network “192,” subnet “251,” and domain “0.” Using IP, typically a range of addresses are reserved for private network use and the remaining IP addresses are either reserved for other uses or for public (routable) addresses, which are assigned by an authority organization. Multiple private networks may assign the same (non-routable) IP addresses internally (currently the Internet Corporation for Assigned Names and Numbers, ICANN), but use unique public IP addresses to connect to the Internet. For example, in Class B networks (which, using class-based addressing conventions, allow up to 254 host machines each), private addresses may range from 192.168.0.0 to 192.168.255.0. Addressing conventions are well known to one skilled in the art and are presumed known to the reader. For purposes of this application, an IP address is understood to be whatever address makes sense (and is defined) in a particular internetworking environment, although examples which use Internet IP addressing may be referred to.

[0008] A network that supports IP can be connected to another TCP/IP-based or UDP/IP-based internet or the Internet, by providing a router which forwards data to another router or host machine based upon an IP address. The router (or routing server/service) typically contains a routing table, which determines to which machine (and optionally to which port) to send a datagram, given a destination IP address. The IP address uniquely identifies a router/host machine, and, in a typical TCP/IP network, can be mapped to a string name that identifies, for example, a particular

machine as part of a larger domain. (Although referred to herein often as a TCP/IP-based network, one skilled in the art will recognize that the network may also be UDP-based (connectionless), and may support another session management system.)

[0009] In a typical TCP/IP (inter)network, machines are organized according to a naming hierarchy. One such hierarchy is the Domain Name System ("DNS"), which specifies how a particular machine is connected to others. (The term DNS is sometimes used also to identify a server or service that implements the DNS protocol.) For example, the string "initials_machine.4thpass.service_provider.com," may be used to specify a machine belonging to you and connected to the 4thpass company LAN, which is in turn connected to a service_provider's WAN (wide area network). TCP/IP and UDP/IP define tools/libraries for client systems to use to determine an IP address (a logical address on an internet) given a string name of a particular router/machine. One such tool is referred to as a DNS query and includes, for example, an API called "GetHostByName." GetHostByName returns an IP address given a string. A server machine that implements the DNS standard for a particular organization, network, or subnetwork is referred to herein simply as a DNS, although DNS services may be provided in conjunction with other networking services on a single physical machine.

[0010] FIG. 1 is an example block diagram of basic Internet or internet communication to send a data packet using TCP/IP or UDP/IP. In FIG. 1, a host machine 101 is shown connected to the Internet 110 via wire 102. Other wired devices, such as host machines 140 and 141, are shown connected to a private (or public) network 130, which could be, for example, a local area network (LAN). The devices on network 130 communicate over the Internet 110 by means of an additional machine, a router 121, which is shown connected via a wire (e.g., a phone line). A DNS server 120 is shown, also connected via a wire, to Internet 110. In basic typical operation source device (host machine) 101, intending to send a data packet to device 140 (via TCP or UDP), first performs a DNS query to obtain an IP address that corresponds to a string name that represents the device 140. For example, to send data to "initials_machine.4thpass.service_provider.com," DNS 120 may correspond to a DNS server for "4thpass.com." This server knows how to locate machines in its "domain," for example, the "initials_machine," and can retrieve their corresponding public IP addresses. Upon determining the correct (public) IP address, the DNS 120 returns this address to the source device 101. The source device 101 then can either open up a TCP connection to communicate with the destination device 140 or simply send packets via UDP or other connection-less protocol using the returned public IP address.

BRIEF SUMMARY OF THE INVENTION

[0011] Embodiments of the present invention provide computer- and network-based methods and systems for two-way initiated, bi-directional communication with wireless devices using connection-based or connection-less protocols, such as, for example, TCP/IP and UDP/IP. Example embodiments provide an Address Management Proxy System ("AMPS"), which enables devices and systems connected to a public internetwork, such as the Internet, to initiate communication with and send data to wireless

devices connected to a private wireless network, without exposing the non-routable private addresses of these wireless devices. The AMPS allocates a public (routable) network address for temporary use by a requesting device on an external public network to communicate with a wireless device on a private wireless network. In one embodiment, a pool of public addresses, for example, public IP addresses, is maintained by the AMPS and allocated dynamically to wireless network devices as required. The mapping of a temporary public address to the private address of a wireless device is maintained and updated transparently by the AMPS using routing tables and other mapping data structures.

[0012] In one embodiment, the AMPS comprises one or more modified DNS/API servers, one or more Address Proxy/Routers, an Address Management Data Server, one or more Address Management Data Repositories, and optionally a load balancer. The AMPS DNS'/API server receives a request from a device on a public network for a particular wireless device, and returns an appropriate temporary public address, which is internally mapped to the private address of the wireless device. The public address is then usable by the device on the external public network to send data to the wireless device. The temporary public address is, for example, an address associated with one of the Address Proxy/Routers, which are connected to the external public network and have access to the private addresses on the private wireless network. In some cases, the device on the public network that desires to send data to the wireless device uses a connection-based protocol, such as TCP/IP to send data. In other cases the device uses a connection-less protocol, such as UDP (UDP/IP) to send the data.

[0013] According to one approach, the AMPS provides a modified DNS server that changes what is returned as a result of a call to a standard DNS query function, such as GetHostByName. In another approach, the AMPS implements a specialized API for a device on a public network to use to query for a public address of a designated wireless device. Using a specialized API implementation in conjunction with Internet Protocol, the AMPS can map private address to addresses that comprise only IP addresses, or can map to an (IP address, port) pair. This latter implementation can extend the usefulness of a particular public address space.

[0014] The AMPS techniques can be used by a wired device on a private or public network and by a wireless device on such a network acting in a capacity of a source device, as long as the device is connected to a public network that can address and be addressed by the AMPS. Thus, the AMPS mechanisms can be used by devices on different private wireless networks to communicate with one another. In addition, the AMPS mechanisms can be used with other internetworks, such as ATM networks and with data transfer protocols other than TCP/IP or UDP.

[0015] To insure a greater degree of security, according to one embodiment, the AMPS maintains a Time to Live (TTL) parameter with each address mapping. In this way, once the TTL value indicates that the mapping has expired, the AMPS can destroy the mapping, and also any connection. Further, in some embodiments, the AMPS also puts a timestamp in its own mapping tables. After some timeout

period based upon the timestamp, the AMPS can destroy the mapping, thereby forcing a new mapping to be initiated on a periodic basis.

BRIEF DESCRIPTION OF THE DRAWINGS

[0016] FIG. 1 is an example block diagram of basic Internet or internet communication to send a data packet using TCP/IP or UDP/IP.

[0017] FIG. 2 is a block diagram of an example Address Management Proxy System used in bi-directional communication with a wireless device.

[0018] FIG. 3 is an example block diagram of components of an example Address Management Proxy System.

[0019] FIG. 4 is an example block diagram of a general purpose computer system for practicing embodiments of the Address Management Proxy System.

[0020] FIG. 5 is an example flow diagram of the process performed by a device on a public network to send data to a wireless device located on a private wireless network using the Address Management Proxy System.

[0021] FIG. 6 is an example block diagram of some of the Address Management Proxy System data repository tables used to support routines of the DNS'/API servers and the Address Proxy/Routers.

[0022] FIG. 7 is an example flow diagram of an example routine provided by a DNS'/API server of the Address Management Proxy System to return a public address that corresponds to a designated unique identifier.

[0023] FIG. 8 is an example flow diagram of an example routine within an Address Proxy/Router of an example Address Management Proxy System that receives data.

DETAILED DESCRIPTION OF THE INVENTION

[0024] Embodiments of the present invention provide computer- and network-based methods and systems for two-way initiated, bi-directional communication with wireless devices using connection-based or connection-less protocols, such as, for example, TCP/IP and UDP/IP. Example embodiments provide an Address Management Proxy System ("AMPS"), which enables devices and systems connected to a public internet, such as the Internet, to initiate communication with and to send data to wireless devices connected to a private wireless network, without exposing the non-routable private addresses of these wireless devices.

[0025] In existing systems, data communication (communication on a data channel) between a wireless device that is connected to a private wireless network and a wired device connected to a public wired network (e.g., the Internet) can only be initiated by the wireless device. Some carriers have assigned fixed public IP addresses to wireless devices; however, the wireless devices need to then have client programs (e.g., a UDP stack) capability of receiving and handling the incoming communication packets. Moreover, these wireless devices are then part of a public wireless network and not a private wireless network. Since public IP addresses are becoming a scarcer commodity and currently expensive to service a network of multi-million devices, carriers cannot in a practical sense count on having a fixed

public IP address for each device on its network. (Although movement to IPv6 will allow more addresses, the current IPv4 definitions are limited and the potential number of wireless users subscribing to larger carriers is very high. In some regions of the world, the current public address scheme is even more limited.) Further, such addressing capability would expose each device to further security risks, because each such device is part of a publicly accessible network and it becomes more difficult to implement and enforce security measures. Thus, assigning private IP addresses to devices is preferred in existing wireless networks over assigning fixed public IP addresses to devices. When wireless networks are private, the locations (addresses) of the wireless devices are intentionally hidden from public view by a carrier system's (or, as referred to in some countries, operator's) infrastructure.

[0026] Wireless systems on private networks use techniques similar to Network Address Translation (NAT) technology to send data from a wireless device to the public networking world. In a typical carrier infrastructure that uses a private network, a wireless device "registers" itself with the carrier infrastructure when it powers on (or in other circumstances, when the device attempts to initiate data services). The carrier dynamically assigns the wireless device a private non-routable address by means of DHCP (or DHCP-like) server. The information on the mapping of the transient private IP address to the device public IP address is stored in an internal carrier database and managed by carrier services such as a RADIUS server.

[0027] The actual path taken by data when sent from a wireless device on a private network to the internet (or vice versa) is very much network infrastructure, carrier technology dependent, and proprietary. Although several standards have emerged, they are very diverse in nature. In a GPRS network, for example, a SGSN (Service GPRS node) manages the communication with wireless devices; whereas the SGSN connects to a GGSN (Gateway GPRS Node) to connect to the Internet network. Regardless of the mobile network used, data communication between a wireless device and internet involves various network elements like GGSN (or similar elements, based on GPRS, GSM, CDMA or any other network), DNS servers, routers and gateways.

[0028] Limited facilities exist for sending short sequence messages from a wired device on a public network to a mobile device on a wireless network (with a private address), again dependent upon carrier and network technology. SMS (Short Message System) protocol defines a format for such messages, but doesn't provide any guidance for underlying structure or data transfer. In addition, such messages are of a fixed (very short) length and use a specialized channel for sending control information (a signaling channel), not a data channel.

[0029] Therefore, there is no existing mechanism for setting up a bi-directional communication channel to communicate with wireless devices that are connected via a private network to the carrier infrastructure using a standard addressing scheme. Nor, is there any mechanism for initiating the communication with such a wireless device from a device connected to a public network. Thus, embodiments of existing systems are not able to support any kind of programmatic "push" of data to wireless devices from wired devices that reside external to a carrier's infrastructure. In

addition, messages sent from the wireless device to a device on a public (wired) network cannot be directly responded to by the wired device, because the address of the wireless device contained in the message may no longer be valid. Moreover, it is not possible using these systems to provide over-the-air provisioning of applications and to send other code to a wireless device from a device external to the carrier infrastructure. Over-the-air provisioning and related techniques for dynamic provisioning and downloading applications to wireless devices are discussed in detail in U.S. application Ser. No. 09/997,402, entitled "Method and System for Maintaining and Distributing Wireless Applications," filed on Nov. 28, 2001. Thus, a mechanism for allowing an external device with a public IP address to initiate bi-directional communication with a wireless device is highly desirable.

[0030] The Address Management Proxy System achieves two way initiated bi-directional communication by implementing a modified DNS server and serving as a proxy/router for devices on the private wireless network as they interface to the public wired internetworking world. In summary, a pool of public addresses is maintained and dynamically distributed among active wireless devices as needed by the AMPS. FIG. 2 is a block diagram of an example Address Management Proxy System used in bi-directional communication with a wireless device. The term "bi-directional" as used herein means that data paths and communication can flow in either direction between two endpoint systems. FIG. 2 shows wired devices 201, 202, and 203 connected to public network 210. One skilled in the art will recognize that these devices could be connected to another private or public network, which is then connected by one or more wired devices to the public network 210, yet still achieve the functionality discussed here. Any such variation provides equivalent functionality and is explicitly contemplated and presumed to be part of the present invention. On the wireless side, the AMPS 220, acting in its capacity as a proxy (and router) for wireless devices, is shown both connected via wire to the public network 210 and via standard carrier infrastructure elements (not shown) to the wireless network 230. It is presumed that the reader has a working knowledge of the elements of a carrier's infrastructure and the basic mechanisms for routing and mechanisms converting packets from a wired network to a wireless network. These may use analog or digital technology and may require protocol conversions in order to send the physical data and transmit it, for example through a satellite, ultimately to the wireless device. Detailed background information on wireless technology and wireless routing mechanisms is described in Stallings, W., *Wireless Communications and Networks*, Prentice Hall, N.J., 2002, which is herein incorporated by reference in its entirety. In FIG. 2, wireless device 240 is shown connected via various wireless elements (not shown) to the wireless network 230.

[0031] In operation, when a wired device such as wired device 201, desires to send data to wireless device 240, the wired device needs first to determine a routable address location of the wireless device 240 on the public network 210. However, as can be observed from FIG. 2, the wireless device 240 is not directly connected to the public network 210. Thus, the wired device must use the Address Management Proxy System 220 to determine a routable (public) address to which to send the data destined for the wireless device 240. To accomplish this task, the wired device 201

performs a query (for example, a modified DNS query) to locate a routable public address that corresponds to wireless device 240. The Address Management Proxy System 220 receives the query and determines and assigns a public (routable) address from a pool of addresses, which address can be used as a destination address for data packets targeted to wireless device 240. One should note that, although shown here as a DNA query, the Address Management Proxy System, as will be described further, modifies the DNS query implementation to handle wireless device technology, and/or provides an alternative API for determining a suitable routable address. Once the Address Management Proxy System 220 has returned a routable address to wired device 201, the wired device 201 can send data to that address. The Address Management Proxy System 220 receives the data, converts formats and protocols, etc., as needed, and sends the converted data through the wireless network 230 to the wireless device 240.

[0032] FIG. 3 is an example block diagram of components of an example Address Management Proxy System. In one embodiment, the Address Management Proxy System (AMPS) comprises one or more modified DNS'/API servers 302, one or more Address Proxy/Routers 305, an Address Management Data Server 303 which manages a database or other repositories such as Address Management Data Repository 304, and optionally a load balancer 301. The DNS'/API servers 302 are either individually connected to a public network 310 or are connected to the load balancer 301 which is in turn connected to public network 310. Similarly, each Address Proxy/Router 305 is also connected to the public network 310, via the routable (public) address to which data from the external network 310 is sent that is destined for the wireless devices. The DNS'/API servers 302 are either modified implementations of a DNS server to add functionality necessary to communicate with wireless devices, or are servers that implement one or more specialized APIs, as will be described further below. The DNS'/API servers 302 use the Address Management Data Server 303 to assist in mapping a unique identifier (e.g., string name) for a wireless device to a public address on public network 310. The pool of public addresses is also maintained by the Address Management Data Server 303 and Data Repository 304. The Address Management Data server 303 and Address Management Data Repository 304 may be implemented using existing database technology, for example, ODBC technology or, may be implemented as a structure such as a simple text file. One skilled in the art will recognize that any embodiment for storing a set of tables, data, lists, or mappings can be used. Each Address Proxy/Router 305 also uses the Address Management Data Server 303 or equivalent to create and update a series of routing tables that are used to assign public addresses to wireless devices as needed and to update the various mappings between public addresses and the non-routable (private) addresses of the wireless devices. The tables and mappings that are maintained on behalf of the DNS'/API servers 302 and the Address Proxy/Routers 305 by the Address Management Data Server 303 are described below with reference to FIG. 6.

[0033] Although the techniques of the AMPS are generally applicable to any a wired device communicating with a wireless device, the phrase "public network" (or "wired network") is used generally to imply any type of internetworked environment including a public network or a backbone that is somewhere down the line connected to one or

more private or public networks. In addition, although the examples described herein often refer to the Internet, one skilled in the art will recognize that the concepts and inventions described are applicable to other forms and embodiments of internetworking, including, for example ATM type networks. Thus, techniques of the present invention can also be used by one device on a first wireless network to communicate with another wireless device on a second network—each device ends up communicating with the Address Proxy/Router of the other network. This scenario is feasible because each wireless network (or its carrier infrastructure) is connected to a proxy/router that is also connected (via a public address) to a public network. In addition, although a public network is sometimes also referred to herein as a wired network, one skilled in the art will recognize that any network that exposes routable (public) addresses may be implied. Thus, a wireless network with unique public (and routable) address can also employ techniques of the present invention to perform bi-directional communication. Also, one skilled in the art will recognize that terms such as wireless device, phone, handheld, etc., are used interchangeably to indicate any type of wireless device that is capable of operating with the AMPS. In addition, terms may have alternate spellings which may or may not be explicitly mentioned, and one skilled in the art will recognize that all such variations of terms are intended to be included.

[0034] Example embodiments described herein provide applications, tools, data structures and other support to implement private to public address mappings over one or more wired and wireless networks to be used for bi-directional communication. One skilled in the art will recognize that other embodiments of the methods and systems of the present invention may be used for many other purposes, including to push information and/or data or code from a public network such as the Internet to a wireless device. In addition, although this description primarily refers to “data” as being sent via the networks, one skilled in the art will recognize that all types of data can be communicated using the techniques described herein including, but not limited to, text, graphics, audio, and video.

[0035] Also, in the following description, numerous specific details are set forth, such as data formats and code sequences, etc., in order to provide a thorough understanding of the techniques of the methods and systems of the present invention. One skilled in the art will recognize, however, that the present invention also can be practiced without some of the specific details described herein, or with other specific details, such as changes with respect to the ordering of the code flow.

[0036] FIG. 4 is an example block diagram of a general purpose computer system for practicing embodiments of the Address Management Proxy System. The general purpose computer system 400 may comprise one or more server and/or client computing systems and may span distributed locations. In addition, each block may represent one or more such blocks as appropriate to a specific embodiment or may be combined with other blocks. The various blocks of the Address Management Proxy System 410 may physically reside on one or more machines, which use standard inter-process communication mechanisms to communicate with each other. In the embodiment shown, computer system 400 comprises a computer memory (“memory”) 01, a display

402, a Central Processing Unit (“CPU”) 403, and Input/Output devices 404. The Address Management Proxy System 410 is shown residing in memory 401. The components of the Address Management Proxy System 410 preferably execute on CPU 403 and manage the address mapping of wireless devices on a wireless network, as described in previous figures, to allow other wired systems to communicate with the wireless devices. Other downloaded code 405 and potentially other data repositories also reside in the memory 410, and preferably execute on one or more CPU’s 403. In a typical embodiment, the AMPS 410 includes one or more DNS’/API servers 411, one or more Address Proxy/Routers 412, an Address Management Data Server 413, and Address Management Data Repositories 414. As described earlier, the AMPS may include other data repositories and components, such as a load balancer, depending upon the particular implementation.

[0037] In an example embodiment, components of the AMPS 410 are implemented by modifying existing UDP-based technology, such as DNS servers and routers, which are typically implemented on Linux/Unix systems written in the C programming language. Programming interfaces to the DNS’/API servers 411 and Address Proxy/Routers 412 can be available by standard means such as through C, C++, C#, and Java API and through scripting languages such as XML, or through web servers supporting such. The Address Management Data Server 413 and Address Management Data Repositories 414 are preferably implemented for scalability reasons as a database system rather than as a text file and may be implemented using a SQL/ODBC database management system. The DNS’/API servers 411 and Address Proxy/Routers 412 are typically implemented using Linux, UNIX, or other Unix-based or Unix-like machines.

[0038] One skilled in the art will recognize that the AMPS 410 may be implemented in a distributed environment that is comprised of multiple, even heterogeneous, computer systems and networks. For example, in one embodiment, the DNS’/API servers 411, the Address Proxy/Router components 412, and the Address Management Data Servers 413 with their data repositories 414 are all located in physically different computer systems. In another embodiment, various components of the AMPS 410 are hosted each on a separate server machine and may be remotely located from the tables which are stored in the address management data repository 414. In addition, under some scenarios, the entire AMPS system 410 may be hosted within a carrier’s infrastructure and be completely subsumed by it. Different configurations and locations of programs and data are contemplated for use with techniques of the present invention. In example embodiments, these components may execute concurrently and asynchronously; thus the components may communicate using well-known message passing techniques. One skilled in the art will recognize that equivalent synchronous embodiments are also supported by an AMPS implementation. Also, other steps could be implemented for each routine, and in different orders, and in different routines, yet still achieve the functions of the AMPS.

[0039] There are several implementation approaches to the components of the Address Management Proxy System, three of which are described herein. One skilled in the art will recognize that various other approaches and combinations are possible. All three approaches allocate a public (routable) network address for temporary use by a wired

device to communicate with a wireless device. In one embodiment, a pool of public addresses, for example, public IP addresses, is maintained and allocated dynamically to wireless network devices as required. For example, a typical Class B Internet network address block allows for approximately 65,000 simultaneous connections to wireless devices. Although this number may seem large at first blush, when one considers the number of cell phones and handsets, for example, connected to a carrier, this number can be quite limiting.

[0040] The first approach provides a public address mapping for a wireless device and makes it available using a UDP protocol. This is a store and forward type of approach. One advantage of this approach is that it facilitates UDP-based traffic to wireless devices over a wireless connection and allows a device connected to the public side of the connection to initiate the connection. A disadvantage, however, is that either the standard UDP protocol requires modification or an extra API call is added so that the requesting device can identify the destined wireless device. These modifications are necessary because the UDP protocol only supports the designation of an IP address and does not provide for an alternative means of uniquely identifying a device (such as a string similar to the TCP/IP protocol). Since it is desirable to hide this (private) address, the standard UDP call to send data does not suffice.

[0041] The second approach used to implement the AMPS supports full bi-directional communication through point-to-point connections established, for example, using TCP/IP protocol. (Note that these same techniques also support connection-less UDP bi-directional communication). The second approach can be implemented by providing a modified implementation of a standard UDP or TCP/IP function, "GetHostByName." The GetHostByName API allows a string designation to identify the designated device and returns an IP address data structure. Alternatively, to increase the level of security provided, the AMPS can implement a specialized API to return the dynamically allocated public address that (now) corresponds to the requested wireless device. A disadvantage of the specialized API approach is that the device on the public network (or other device that wishes to obtain a connection to the wireless device) needs to include specialized code in the application on the requesting device.

[0042] The third approach is similar to the second approach, but provides for greater potential simultaneous connections using a "port" concept. Using this approach, instead of returning just a host address on the wired network that corresponds to the wireless device (a public address of an Address Proxy/Router), the AMPS also returns a particular port designation on the host machine (the Address Proxy/Router). Ports and their implementation are well known in the art and, in general, correspond to queues implemented by the machine that receives the data to track messages destined for different locations. The machines that receive the data forward messages to the different destinations based upon the port designations in the message and handle, where necessary, sequencing and handshaking on a by-port basis.

[0043] Ports are typically used to open multiple virtual channels on a single physical address and potentially extend a single physical address to another approximately 65,000

connections. This enables an AMPS to use Class C IP addresses, for example, which are typically much cheaper and more readily available than the other types of addresses, to achieve approximately 16 million simultaneous connections to wireless devices (using a UNIX-based system). One skilled in the art will recognize that the number of ports available and their particular implementation is operating system dependent and directly correlates to the number of bits used to specify a port address.

[0044] The UDP (as well as TCP/IP) protocol currently supports a port abstraction; however, the standard DNS query used with UDP and TCP/IP systems (e.g., GetHostByName) does not allow for the returning of a port designation and leaves control of the port designation up to the requesting device instead of the receiving router. Thus, according to the third approach to implementing the AMPS, ports are specified preferably by the Address Proxy/Router using a specialized API and returned to the requesting device with the host machine designation.

[0045] In one embodiment, a scripting language interface, such as XML, can be used instead of a specialized API (code interface) to interface to a specialized address mapping function. When using XML, the DNS/API server accepts API calls as XML post events and returns XML formatted responses. Similar support for invoking this mapping function using other language models and/or other programming languages is also contemplated and will operate with techniques of the present invention. An XML type interface minimizes the cost to a requesting device of integrating an API into its software.

[0046] In addition, when using a specialized API, the requesting device can push additional data to the private wireless device with no further coding effort required. For example, billing code orienting to billing the wireless device for the type of connection being established with the wired device can be pushed to the wireless device using this technique. An example billing code mechanism is discussed in U.S. patent application Ser. No. 10/085,981, entitled "Method and System for Transmission-Based Billing of Applications," filed on Feb. 26, 2002.

[0047] There are several security related reasons that also may indicate a desire to use a specialized API rather than the standard GetHostByName API of UDP and TCP/IP. These security reasons may indicate a desire to use the port-based mechanism which also requires a specialized API. First, any bi-directional communication that uses the "GetHostByName" API and router protocol is susceptible to denial of service (DOS) attacks associated with that host name. DOS attacks can occur as a result of one or more machines specifying the same GetHostByName with the same input parameter simultaneously so as to close out the possibility of enough public addresses (of proxy/router machine addresses) being available. Second, use of the GetHostByName (or any other standard) API also makes the AMPS susceptible to security attacks as a result of DNS zone transfer. DNS zone transfer occurs, for example, when one DNS transfers a DNS query to one or more other DNS servers in order to find a requested host name. One typical scenario involves multiple DNS hops from country to country. The idea of DNS zone server attacks is to capture information on a particular DNS server by inserting malicious code as a malicious DNS server and impersonate the

destination address. Third, inattentiveness to the duration of a connection can make data available to unauthorized requesting devices. Specifically, since a mapping is established between the public device and a wireless device, the public device may assume the mapping to exist longer than it really does and potentially end up (maliciously or not) communicating with the wrong device. This could occur because some DNS server implementations ignore Time to Live settings. Having a modified DNS server implementation, whether or not it implements the standard GetHostByName API, or a specialized API, avoids this problem by abiding by the TTL characteristics of the established mapping between the private and the public addresses.

[0048] FIGS. 5-8 describe various example embodiments of the specific routines implemented by the DNS/API servers and the Address Proxy/Routers to achieve the functionality described with reference to FIGS. 2 and 3. FIG. 5 is an example flow diagram of the process performed by a device on a public network to send data to a wireless device located on a private wireless network using the Address Management Proxy System. In step 501, the source (e.g., wired) device requests a public (routable) address for the designated wireless device from the AMPS. As described above, one mechanism for retrieving such an address is for the AMPS to implement a modified interface for DNS services, such as a modified GetHostByName routine. For example, a GetHostByName routine may be invoked to specify a unique wireless device designation, in a string form. The string “personname.phone.wsp.com” is an example string that indicates a wireless device that corresponds to a person by the name of “person,” with a phone number of “phone,” located at a wireless service provider’s website (DNS) that is abbreviated “wsp.com.” (For example, susan.ph2065551212.sprint.com may refer to a person, Susan, with phone number 206 555 1212, on a sprint network.) Another mechanism that may be implemented by the AMPS is to provide a separate (specialized) API, such as “GetProxyIP” which returns an indication of a public (routable) address that corresponds to the designated wireless device. Using a separate API is useful for security reasons; for example, it becomes more difficult for a rogue device to intercept the routable address by spoofing. Another reason is to control the actual format of the address information returned, such as to be able to provide a port designation in addition to a host address of the Address Proxy/Router. The implementation of either GetHostByName, or a specialized API such as GetProxyIP, is discussed further below with respect to FIG. 7.

[0049] In step 502, once the public address for the wireless device has been returned by the AMPS, the source device extracts from the indicated address information an actual address location (IPdata.ip), a Time to Live parameter (IPdata.TTL), and optionally, where applicable, a port designation (IPdata.port) for example, when the standard GetHostByName API is not used. In step 503, when the source device desires to perform connection-based communication with the wireless device, it opens a connection, such as a socket. In step 504, the wired device determines whether the Time to Live parameter specifies that the returned public address of the wireless device has expired, and, if so, returns to step 501 to request a different address, else continues in step 505. The Time to Live parameter is used by the AMPS to ensure that a public address for a wireless device (which may or may not be always connected to the wireless network

and powered on) does not extend past a period of usefulness for that wireless device. In some embodiments, a very short TTL parameter is used, to prevent security breaches in which a wired device is able to monitor activity on a particular public address and then use that address for other purposes. In step 505, the wired device sends a data packet (via the connection or not) to the wireless device. It is presumed, but not shown, that the wired device and the wireless device communicate using the standard calls specified in the transfer protocol being used, for example UDP or TCP/IP. In step 506, if the data transaction is complete, then the wired device is done, else it returns to check the TTL parameter in step 504 in order to send more data.

[0050] Table 1 below contains pseudocode that describes one example implementation of how a public (requester) device can use a modified GetHostByName query as described in FIG. 5 to obtain a public address that corresponds to a user identified by the phone number “2065551212” using the Address Management Proxy System. From the public device’s perspective, standard UDP and TCP/IP calls are invoked in a standard manner.

TABLE 1

InetAddress ip = InetAddress.GetHostByName(“2065551212.fourthDNs.att.com”); byte [] data; int destPort = 80; // data buffer filled by some internal routine say // popuiateDataBuffer(), which in turn uses setData() Java API. int datasize = populateDataBuffer(data); DatagramSocket socket = new DatagramSocket(destPort, ip); // Now send some data . . . DatagramPacket packet = new DatagramPacket (data, datasize); socket.send (packet);

[0051] Table 2 below contains pseudocode that describes an example implementation of the specialized API mechanism described in FIG. 5 as used by a public (requester) device to obtain a public address of the user identified by the same phone number. In Table 2, the public device calls a specialized API, GetProxyIP, to get a data structure that contains the needed information including the assigned public address of the Address Proxy/Server. Thereafter, the public device can use the same techniques (standard UDP or TCP/IP protocol) to send the data.

TABLE 2

public class IPInfo { private String __publicIPAddress; private int __publicPort, __TTL; public String getPublicIPAddress() { return __publicIPAddress; } public int getPublicPort() { return __publicPort; } public int getTTL() { return __TTL; } } int destPort = 80; IPInfo ipi = GetProxyIP(“2065551212.fourthDNS.att.com”,
--

TABLE 2-continued

<pre>destPort); // Return value of ipi.getPublicIPAddress() in format x.y.z.q // For example: 142.432.14.2 InetAddress ip = InetAddress.getByName (ipi.getPublicIPAddress ()); byte [] data; // data buffer filled by some internal routine say // populateDataBuffer(), which in turn uses setData() Java API. int datasize = populateDataBuffer(data); DatagramSocket socket = new DatagramSocket (destPort, ipi.getPublicPort ()); // Now send some data. . . DatagramPacket packet = new DatagramPacket (data, datasize); socket.send (packet)</pre>
--

[0052] FIG. 6 is an example block diagram of some of the Address Management Proxy System data repository tables used to support routines of the DNS'/API servers and the Address Proxy/Routers. In one embodiment, the Address Management Data Repository comprises three tables: a unique identifier (unique ID) to private address table 610, a public-to-private address table 630, and a public address to proxy/router machine table 620. Although three tables are shown, one skilled in the art will recognize that these tables could contain other data and may be organized differently, including a different number of tables and with different columns or fields. In addition, any technique for storing a table or list of data may be used. To support embodiments that map a host address plus a port designator to a wireless device, the tables are correspondingly modified.

[0053] The unique ID table 610 maps unique string names of wireless devices to the private wireless network addresses that have been assigned typically by a carrier's infrastructure. In some embodiments, the carrier infrastructure dynamically allocates, using methods similar to a DHCP protocol, a private wireless network address when the wireless device registers itself with the carrier infrastructure upon being powered on. Thus, the unique ID table 610 may be sparsely filled or entries created dynamically and then deleted dynamically as devices register and unregister with the carrier infrastructure system.

[0054] The public-to-private address table 630 comprises several fields/columns including a public network address 631, a private (wireless) network address 602, a flag 632 that specifies whether the public address stored in field 631 is free or is already used, a Time to Live (TTL) parameter 633, and other connection data 634. In one embodiment, the DNS'/API servers of the AMPS query table 630 to determine a public network address that corresponds to a designated private wireless network address or to allocate an unused public network address (as indicated in field 632) and map the determined unused public address to a private network address stored in field 602.

[0055] Public address to proxy/router machine table 620 comprises a public network address field 631 and an indication of a functioning proxy/router machine 621. By maintaining such a mapping, the AMPS is able to substitute proxy/router machines for other proxy/router machines to provide a higher degree of robustness. Each proxy/router

machine has a preconfigured set of public network addresses, such as are typically configured by network cards inserted into the proxy/router machine. These address are allocated in a standard fashion through prior purchase or obtaining from an address authorizing authority, currently the Internet Corporation for Assigned Names and Numbers (ICANN). When a machine is inserted for use into the AMPS, indications of these addresses are stored in field 631.

[0056] In one embodiment, a timestamp of the mapping between a public network address and a private address is also noted in table 630. After a defined time out, based upon this timestamp, the Address Management Data Server sends a request to the Address Proxy/Router associated with the mapping to unmap the public-to-private address mapping and updates the mapping table 630, thereby returning the associated public address to the pool of unused public network addresses.

[0057] FIG. 7 is an example flow diagram of an example routine provided by a DNS'/API server of the Address Management Proxy System to return a public address that corresponds to a designated unique identifier. In essence, this routine implements a DNS query or DNS-like query capability for the AMPS using a modified GetHostByName interface or a specialized API, for example GetProxyIP, as described with reference to FIG. 5. In summary, the routine dynamically allocates an appropriate Address Proxy/Router machine to associate with the wireless device and returns the public address of that machine (along with a TTL parameter and potentially other parameters). Specifically, in step 701, the routine determines the private (non-routable) address of the wireless device designated by a string parameter passed as input to the routine. For example, the string parameter may use fields such as "uniqueID.hostname.domain.tld," which specifies a typical hierarchy of person/service on a machine named "hostname" on a domain such as a company's network on a top level domain such as "org," "com," "edu," etc. One skilled in the art will recognize that many other string parameter designations could be used. One mechanism for implementing this routine is to request information from the Address Management Data Repository. In one embodiment, the data repository stores a table that maps unique IDs to private network addresses (see Table 610 in FIG. 6). Preferably, any mechanism that is used by the AMPS stores this data in a secure manner in order to keep the wireless network addresses private. In step 702, the routine retrieves the public network address that corresponds to the private wireless network address of the designated device if one has already been assigned by the AMPS to that device and is still valid. In one embodiment, the data repository stores this mapping information between private wireless network address and public network address (see for example table 630 in FIG. 6). If a public network address has not already been assigned or is not valid, then the routine causes a new public network address to be allocated and that new public address is associated with the private wireless network address. Appropriate tables in the data repository are then updated. In step 703, the routine determines the Address Proxy/Router machine that is associated with the assigned public network address (for example using table 620 in FIG. 6). In step 704, the routine sends a request to the determined proxy/router machine to update its routing tables to map the determined public network address to the private wireless network address. In step 705, the routine updates information in the data repository to include any other

connection related information (e.g., field 634 in Table 630 in FIG. 6) and indicates a Time to Live (TTL) parameter for the public-private address association (e.g., field 633 in Table 630 in FIG. 6). Once all of the tables have been updated in both the proxy/router and in the data repository, the DNS'/API server returns the determined public network address of the associated Address Proxy/Router machine. As described earlier, the public address may be a (hostname, port) pair when a port-based implementation is used.

[0058] FIG. 8 is an example flow diagram of an example routine within an Address Proxy/Router of an example Address Management Proxy System that receives data. This routine is implemented by an Address Proxy/Router to receive data from wired devices using the relevant protocol (e.g., TCP/IP, UDP/IP, etc.). (Data is sent to a particular Address Proxy/Router because the public (routable) address of that Address Proxy/Router was returned to the requesting/source device in response to a query of the DNS'/API server of the AMPS.) The proxy/router is responsible for any conversions of the data necessary to send data from the wired network to the wireless network (if, for example, the carrier infrastructure desires such conversion to be performed at this level). The proxy/router is also responsible for forwarding the data via the wireless network to the private wireless address of the wireless device that corresponds to the public address used to invoke the proxy/router.

[0059] Specifically, in step 801, the routine determines (for example, from the Address Management Data Repository) the private wireless address that corresponds to the invoked public address and the TTL parameter associated with this mapping. These values can be obtained, for example, from the public-to-private address table 630 of FIG. 6. In step 802, the routine determines whether the value of the determined TTL parameter indicates that the mapping has expired, and if so, returns an error, else continues in step 803. In step 803, the routine determines the format required by the target device (the wireless network format). In step 804, the routine determines whether any protocol conversion is necessary and, if so, continues in step 805 else continues in step 806. Note that protocol conversion for a specific wireless network such as converting the data to an "HTTP" packet may be done by the proxy/router or may be done by some other component within the carrier infrastructure. One skilled in the art will recognize that these are example steps and that different formatting or different protocol conversion routines may be added or omitted as specific to the environment. In step 806, the Address Proxy/Router routine sends the data packet (which has been formatted and whose protocol is converted as necessary) to the determined associated private address of the wireless device, and returns.

[0060] All of the above U.S. patents, U.S. patent application publications, U.S. patent applications, foreign patents, foreign patent applications and non-patent publications referred to in this specification and/or listed in the Application Data Sheet, including but not limited to U.S. Provisional Patent Application No. 60/296,902, entitled "Method and System for Two-Way Initiated Data Communication with Wireless Devices," filed Jun. 8, 2001; U.S. patent application Ser. No. 09/997,402, entitled "Method and System for Maintaining and Distributing Wireless Applications," filed Nov. 28, 2001; and U.S. patent application Ser. No. 10/085,981, entitled "Method and System for Transmission-Based

Billing of Applications," filed on Feb. 26, 2002 are incorporated herein by reference, in their entirety.

[0061] From the foregoing it will be appreciated that, although specific embodiments of the invention have been described herein for purposes of illustration, various modifications may be made without deviating from the spirit and scope of the invention. For example, one skilled in the art will recognize that the methods and systems for establishing bi-directional communication discussed herein are applicable to other types of public networks and protocols other than the Internet, TCP/IP, and UDP, or even a plurality of such networks. For example, private address to public address mappings can also be provided for ATM networks to connect devices on an ATM network with a wireless device. One skilled in the art will also recognize that the methods and systems discussed herein are applicable to differing protocols, communication media (optical, wireless, cable, etc.) and wireless devices (such as wireless handsets, electronic organizers, personal digital assistants, portable email machines, game machines, pagers, navigation devices such as GPS receivers, etc.) to different wired device (such as kiosks, personal computers, mainframes), and to wireless devices having a wired connection capabilities, e.g., wireless email or PDA devices that can be connected to a docking station for synchronization purposes.

1. A method in a computer network environment that is connected to a wireless communications network through an address management proxy system, the wireless communications network connected to a wireless device using a private network address of the wireless communications network, the wireless device having an associated unique identifier, the address management proxy system having a public network address of the computer network environment, and the computer network environment connected to a wired device using a public network address of the computer network environment, comprising:

under control of the wired device,

using the public network address of the address management proxy system, sending a request to the address management proxy system for an indication of a public network address of the computer network environment to use for communicating with the wireless device, the request indicating the unique identifier associated with the wireless device;

under control of the address management proxy system,

receiving the request for the indication of the public network address;

determining a public network address from a plurality of available public network addresses of computer network environment;

associating the determined public network address with the private network address of the wireless device that corresponds to the indicated unique identifier; and

forwarding an indication of the determined public network address to the wired device; and

under control of the wired device,

receiving the indicated public network address; and

sending data to the wireless device using the indicated public network address, such that the wired device perceives that the wired device is communicating directly with the wireless device.

2. The method of claim 1, further comprising:

under control of the address management proxy system, receiving the data sent from the wired device;

determining the private network address that is associated with the indicated public network address; and

routing the received data to the private network address over the wireless communication network in a manner that is transparent to the wired device.

3. The method of claim 1 wherein the computer network environment is the Internet and each public network address is specified in accordance with Internet Protocol (IP) addressing conventions.

4. The method of claim 1 wherein the computer network environment is an ATM network.

5. The method of claim 1 wherein the indicated public network address is the public network address of the address management proxy system, wherein data sent from the wired device indicates the unique identifier of the wireless device, and wherein the address management proxy system stores and forwards the data to the wireless device associated with the indicated unique identifier.

6. The method of claim 1 wherein the indicated public network address is received as a result of a call by the wired device to an address mapping function of the computer network environment.

7. The method of claim 6 wherein the address mapping function is a standard function of the computer network environment.

8. The method of claim 6 wherein the address mapping function is a GetHostByName function.

9. The method of claim 1 wherein the indicated public network address is received as a result of a call by the wired device to a function that returns a host system address and a port specification.

10. The method of claim 9 wherein the function is a customized API.

11. The method of claim 9 wherein the function supports an XML interface.

12. The method of claim 1, further comprising:

under control of the wireless device, sending data to the wired device using the public network address of the wired device, thereby performing bi-directional communication.

13. The method of claim 1 wherein a virtual end-to-end connection is established between the wired device and the wireless device.

14. The method of claim 1 wherein a connectionless communications path is established between the wired device and the wireless device.

15. A computer network environment comprising:

a wireless device connected to a wireless communications network using a private network address of the wireless communications network, the wireless device having an associated unique identifier;

a wired device connected to the computer network environment using a public network address of the computer network environment, that is structured to:

request an indication of a public network address of the computer network environment to use for communicating with the wireless device, the request indicating the unique identifier associated with the wireless device,

receive the indicated public network address, and,

send data to the wireless device using the indicated public network address, such that the wired device communicates directly with the wireless device using a virtual end-to-end connection; and

an address management proxy system connected to the computer network environment using a public network address of the computer network environment, that is structured to:

receive the request from the wired device,

determine a public network address from a plurality of available public network addresses of computer network environment,

associate the determined public network address with the private network address of the wireless device that corresponds to the indicated unique identifier, and

forward an indication of the determined public network address to the wired device.

16. The computer network environment of claim 15 wherein the address management proxy system is further structured to:

receive the data sent from the wired device,

determine the private network address that is associated with the indicated public network address, and

route the received data to the private network address over the wireless communication network in a manner that is transparent to the wired device.

17. The computer network environment of claim 15 wherein the computer network environment is the Internet and each public network address is specified in accordance with Internet Protocol (IP) addressing conventions.

18. The computer network environment of claim 15 wherein the computer network environment is an ATM network.

19. The computer network environment of claim 15 wherein the indicated public network address is the public network address of the address management proxy system, wherein data sent from the wired device indicates the unique identifier of the wireless device, and wherein the address management proxy system is structured to store and forward the data to the wireless device associated with the indicated unique identifier.

20. The computer network environment of claim 15 wherein the indicated public network address is received as a result of a call by the wired device to an address mapping function of the computer network environment.

21. The computer network environment of claim 20 wherein the address mapping function is a standard function of the computer network environment.

22. The computer network environment of claim 20 wherein the address mapping function is a GetHostByName function.

23. The computer network environment of claim 15 wherein the indicated public network address is received as a result of a call by the wired device to a function that returns a host system address and a port specification.

24. The computer network environment of claim 23 wherein the function is a customized API.

25. The computer network environment of claim 23 wherein the function supports an XML interface.

26. The computer network environment of claim 15, wherein the wireless device is further structured to send data to the wired device using the public network address of the wired device, thereby performing bi-directional communication.

27. The computer network environment of claim 15 wherein a virtual end-to-end connection is established between the wired device and the wireless device.

28. The computer network environment of claim 15 wherein a connectionless communications path is established between the wired device and the wireless device.

29. A method in a computer network environment for establishing a bi-directional communication channel virtual communication channel between a first device connected to the computer network environment using a public network address and a second device connected to a wireless communications network and having a private address, comprising:

receiving a request from the first device to communicate with the second device;

dynamically determining a public network address from a pool of public network addresses;

associating the determined public network address with the private address of the second device; and

returning an indication of the determined public network address to the first device so that the first device can thereafter send data to the second device using the determined public network address.

30. The method of claim 29 wherein the request indicates a unique identifier of the second device that is different from the private address of the second device, and the associating the determined public address with the private address of the second device is performed using the unique identifier.

31. The method of claim 29, further comprising:

receiving data from the first device directed to the determined public network address; and

transparently routing the received data to the second device using the private network address associated with the determined public network address.

32. The method of claim 31 wherein the transparent routing is performed by a masquerading system connected to the computer network environment by a public network address of the computer network environment.

33. The method of claim 29 wherein the first device is a wired device.

34. The method of claim 29 wherein the first device is a wireless device.

35. The method of claim 29 wherein the second device is a device capable of a wired connection.

36. The method of claim 29 wherein the second device is a wireless device.

37. The method of claim 29 wherein the determined public network address specifies a host system address and a port specification.

38. The method of claim 29 wherein the determined public network address specifies an Internet Protocol (IP) address.

39. The method of claim 29 wherein the method is performed by a modified DNS server.

40. The method of claim 29 wherein the received request is a standard API of the computer network environment.

41. The method of claim 40 wherein the API is a GetHostByName API.

42. The method of claim 29 wherein the received request is a modified API of the computer network environment that specifies a name of a host machine and an indication of a port.

43. The method of claim 29, further comprising:

associating the determined public network address with an expiration period; and

destroying the association between the determined public network address and the private address of the second device when the expiration period as expired.

44. The method of claim 43 wherein the expiration period is specified as Time to Live (TTL) data.

45. An address proxy management system connected to a computer network environment for establishing a virtual communication channel between a first device connected to the computer network environment using a public network address and a second device connected to a wireless communications network and having a private address comprising:

domain name service (DNS) that is structured to:

receive a request from the first device to communicate with the second device,

dynamically determine a public network address from a pool of public network addresses;

associate the determined public network address with the private address of the second device; and

return an indication of the determined public network address to the first device so that the first device can thereafter send data to the second device using the determined public network address.

46. The system of claim 45 wherein the DNS is a modified DNS that abides by at least one of UDP and TCP/IP standards.

47. The system of claim 45 wherein the computer network environment is the Internet.

48. The system of claim 45 wherein each public network address is specified in accordance with Internet Protocol (IP) addressing conventions.

49. The system of claim 45 wherein the DNS uses a database to associate the determined public network address with the private address of the second device.

50. The system of claim 45 wherein the DNS uses a database to dynamically determine a public network address from a pool of public network addresses.

51. The system of claim 45 wherein the request indicates a unique identifier of the second device that is different from the private address of the second device, and the DNS associates the determined public address with the private address of the second device using the unique identifier.

52. The system of claim 45, further comprising:

private address management router that is structured to receive data from the first device directed to the determined public network address, and transparently routes the received data to the second device using the private network address associated with the determined public network address.

53. The system of claim 52 wherein the private address management router is connected to the computer network environment by a public network address of the computer network environment, and transparently routes the received data to the public address of the second device by determining the private network address associated with the received data and causes the data to be forwarded over the wireless communications network to the second device at the determined private network address.

54. The system of claim 52 wherein the private address management router causes protocol conversion of the received data to be performed when applicable before transparently routing the received data.

55. The system of claim 45 wherein the first device is a wired device.

56. The system of claim 45 wherein the first device is a wireless device.

57. The system of claim 45 wherein the second device is a device capable of a wired connection.

58. The system of claim 45 wherein the second device is a wireless device.

59. The system of claim 45 wherein the determined public network address specifies a host system address and a port specification.

60. The system of claim 45 wherein the determined public network address specifies an Internet Protocol (IP) address.

61. The system of claim 45 wherein the received request is a standard API of the computer network environment.

62. The system of claim 61 wherein the API is a GetHostByName API.

63. The system of claim 45 wherein the received request is a new API that specifies a name of a host machine and an indication of a port.

64. The system of claim 45, wherein the DNS is further structured to:

associate the determined public network address with an expiration period; and

cause the association between the determined public network address and the private address of the second device to be destroyed when the expiration period as expired.

65. The system of claim 64 wherein the expiration period is specified as Time to Live (TTL) data.

66. An address proxy management system connected to a computer network environment for establishing a virtual communication channel between a first device connected to the computer network environment using a public network address and a second device connected to a wireless communications network and having a private address comprising:

means for receiving a request from the first device to communicate with the second device;

means for dynamically determining a public network address from a pool of public network addresses;

means for associating the determined public network address with the private address of the second device; and

means for returning an indication of the determined public network address to the first device so that the first device can thereafter send data to the second device using the determined public network address.

67. A computer-readable memory medium containing instructions for controlling a processor in a computer system to establish a bi-directional communication channel virtual communication channel between a first device connected to a computer network environment using a public network address and a second device connected to a wireless communications network and having a private address, by:

receiving a request from the first device to communicate with the second device;

dynamically determining a public network address from a pool of public network addresses;

associating the determined public network address with the private address of the second device; and

returning an indication of the determined public network address to the first device so that the first device can thereafter send data to the second device using the determined public network address.

68. The computer-readable memory medium of claim 67 wherein the request indicates a unique identifier of the second device that is different from the private address of the second device, and the associating the determined public address with the private address of the second device is performed using the unique identifier.

69. The computer-readable memory medium of claim 67, the instructions further controlling the computer processor by:

receiving data from the first device directed to the determined public network address; and

transparently routing the received data to the second device using the private network address associated with the determined public network address.

70. The computer-readable memory medium of claim 69 wherein the transparent routing is performed by a masquerading system connected to the computer network environment by a public network address of the computer network environment.

71. The computer-readable memory medium of claim 67 wherein the first device is a wired device.

72. The computer-readable memory medium of claim 67 wherein the first device is a wireless device.

73. The computer-readable memory medium of claim 67 wherein the second device is a device capable of a wired connection.

74. The computer-readable memory medium of claim 67 wherein the second device is a wireless device.

75. The computer-readable memory medium of claim 67 wherein the determined public network address specifies a host system address and a port specification.

76. The computer-readable memory medium of claim 67 wherein the determined public network address specifies an Internet Protocol (IP) address.

77. The computer-readable memory medium of claim 67 wherein the method is performed by a modified DNS server.

78. The computer-readable memory medium of claim 67 wherein the received request is a standard API of the computer network environment.

79. The computer-readable memory medium of claim 78 wherein the API is a GetHostByName API.

80. The computer-readable memory medium of claim 67 wherein the received request is a modified API of the computer network environment that specifies a name of a host machine and an indication of a port.

81. The computer-readable memory medium of claim 67, the instructions further controlling the computer processor by:

associating the determined public network address with an expiration period; and

destroying the association between the determined public network address and the private address of the second device when the expiration period as expired.

82. The computer-readable memory medium of claim 81 wherein the expiration period is specified as Time to Live (TTL) data.

83. A method in a computer network environment for communicating with a wireless device connected to an address management proxy system using a private network address, the wireless device having an associated unique identifier that is not the private network address, the address management proxy system connected to the computer network environment using a public network address of the network environment, comprising:

requesting from the address management proxy system a public network address that corresponds to the unique identifier associated with the wireless device;

receiving an indication of the public network address of the wireless device; and

sending data to the indicated public network address.

84. The method of claim 83 wherein the computer network environment is the Internet.

85. The method of claim 83 wherein the indicated public network address is specified in accordance with Internet Protocol (IP) addressing standards.

86. The method of claim 83 wherein the unique identifier is at least one of a person's name and telephone number, an MSISDN number, and an ISMI number.

87. The method of claim 83 wherein the request is a standard API, such that communication with the wireless device is performed in the same manner as communication with a wired device connected to the computer network environment.

88. The method of claim 87 wherein the API is a GetHostByName function call.

89. The method of claim 83 wherein the indication of the public network address of the wireless device is a host machine address.

90. The method of claim 83 wherein the indication of the public network address of the wireless device is a host machine address and a port specification.

91. A computer-readable memory medium containing instructions for controlling a computer processor to communicate in a computer network environment with a wireless device connected to an address management proxy system using a private network address, the wireless device having an associated unique identifier that is not the private network address, the address management proxy system

connected to the computer network environment using a public network address of the network environment, by:

requesting from the address management proxy system a public network address that corresponds to the unique identifier associated with the wireless device;

receiving an indication of the public network address of the wireless device; and

sending data to the indicated public network address.

92. The computer-readable memory medium of claim 91 wherein the computer network environment is the Internet.

93. The computer-readable memory medium of claim 91 wherein the indicated public network address is specified in accordance with Internet Protocol (IP) addressing standards.

94. The computer-readable memory medium of claim 91 wherein the unique identifier is at least one of a person's name and telephone number, an MSISDN number, and an ISMI number.

95. The computer-readable memory medium of claim 91 wherein the request is a standard API, such that communication with the wireless device is performed in the same manner as communication with a wired device connected to the computer network environment.

96. The computer-readable memory medium of claim 95 wherein the API is a GetHostByName function call.

97. The computer-readable memory medium of claim 91 wherein the indication of the public network address of the wireless device is a host machine address.

98. The computer-readable memory medium of claim 91 wherein the indication of the public network address of the wireless device is a host machine address and a port specification.

99. A wired device that is connected to a computer network environment using a public network address of the computer network environment, the network environment connected to an address management proxy system using a public network address of the network environment, the address management proxy system connected to a wireless device using a private network address, the wireless device having an associated unique identifier that is not the private network address, comprising:

communications code module that is structured to:

request from the address management proxy system a public network address that corresponds to the unique identifier associated with the wireless device,

receive an indication of the public network address of the wireless device; and

send data to the indicated public network address.

100. The device of claim 99 wherein the computer network environment is the Internet.

101. The device of claim 99 wherein the indicated public network address is specified in accordance with Internet Protocol (IP) addressing standards.

102. The device of claim 99 wherein the unique identifier is at least one of a person's name and telephone number, an MSISDN number, and an ISMI number.

103. The device of claim 99 wherein the request is a standard API, such that communication with the wireless device is performed in the same manner as communication with a wired device connected to the computer network environment.

- 104.** The device of claim **103** wherein the API is a GetHostByName function call.
- 105.** The device of claim 99 wherein the indication of the public network address of the wireless device is a host machine address.
- 106.** The device of claim 99 wherein the indication of the public network address of the wireless device is a host machine address and a port specification.

- 107.** The device of claim 99 wherein the communications code module implements at least one of UDP and TCP/IP communication with the wireless device.
- 108.** The device of claim 99 wherein the communications code module implements UDP communication with the wireless device.

* * * * *