IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Request for Ex Parte Reexamination	§	REQUEST FOR EX PARTE
	§	REEXAMINATION
U.S. Patent No. 6,108,704	§	
	§	
Issued: August 22, 2000	§	
	§	Attorney Docket No.: 03801.G184
For: Point-to-Point Internet Protocol	§	
	§	
Requester: Skype, Inc.	§	Customer No.: 08791

REQUEST FOR EX PARTE REEXAMINATION UNDER 35 U.S.C. § 302

Mail Stop *Ex Parte* Reexam
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

Pursuant to the provisions of 35 U.S.C. §§ 302-307, the undersigned hereby requests an *ex parte* reexamination of claims 1-7 and 10-44 of United States Patent No. 6,108,704 ("the '704 patent," Exhibit A) which issued on August 22, 2000 to Glenn W. Hutton et al. resulting from a patent application filed on September 25, 1995. The Requester hereby asserts that claims 1-7 and 10-44 of the '704 patent are unpatentable over prior art references not before the Patent and Trademark Office (PTO) during prosecution of the '704 patent.

TABLE OF CONTENTS

I. PENDING LITIGATION

The '704 patent is the subject of pending litigation, Net2Phone, Inc. v. eBay, Inc., Skype Technologies SA, and Skype, Inc., Case No. 06-2469, instituted by the current assignee, Net2Phone, Inc., in the United States District Court for the District of New Jersey. Net2Phone alleges that Skype Technologies SA, Skype, Inc. and eBay Inc. infringe claims 1, 2, 4-7, 11, 22, 32-44 of the '704 patent. The parties have submitted their claim construction briefs and a Markman hearing is currently scheduled for March 2, 2009. The Court has not yet set a schedule for summary judgment proceedings. No trial date has been set. Skype, Inc., plans to file a motion to stay the above-entitled litigation pending reexamination on the grounds that a stay of litigation at this time will permit the Court and parties to benefit from the PTO's guidance on issues of patentability and to avoid further costly legal proceedings that would otherwise burden the Court and parties. Claim Construction Briefs submitted by the parties to the pending litigation are set forth in Exhibits S-X.

II. LISTING OF PRIOR ART PATENTS AND PRINTED PUBLICATIONS

In accordance with 37 C.F.R. §§ 1.510(b)(1) and (b)(2), reexamination of claims 1-7 and 10-44 of the '704 patent is requested in view of the following references:

Exhibit B — The Open Group, Technical Standard, Protocols for X/Open PC Interworking SMB, Version 2, (1992) ("**NetBIOS**"), which **published as a single publication** containing:[1] (a) Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Concept and Methods, RFC 1001 (March 1987) ("RFC 1001"); and (b) Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications, RFC 1002 (March 1987) ("RFC 1002").

Exhibit C — Etherphone: Collected Papers 1987-1988 (May 1989) (collectively referred to herein as "**Etherphone**"). These papers, which **published together as a single publication**, include the following:[2]

[1] NetBIOS published as a single reference with RFC 1001 and RFC 2002.
[2] The five papers comprising this reference were published together as set forth on the first page of this reference. Thus, all five papers are a single reference.

a. Polle T. Zellweger, et al., *An Overview of the Etherphone System and its Applications*, IEEE CONFERENCE ON COMPUTER WORKSTATIONS (March 1988), 160-168 (hereinafter "Zellweger 1").

b. Daniel C. Swinehart, *Telephone Management in the Etherphone System*, PROCEEDINGS OF THE IEEE/IEICE GLOBAL TELECOMMUNICATIONS CONFERENCE (November 1987), 1176-1180 (hereinafter "Swinehart 1").

c. Douglas B. Terry and Daniel C. Swinehart, *Managing Stored Voice in the Etherphone System*, ACM TRANSACTIONS ON COMPUTER SYSTEMS 6(1) (February 1988), 3-27 (hereinafter "Terry").

d. Daniel C. Swinehart, *System Support Requirements for Multi-media Workstations*, PROCEEDINGS OF THE SPEECHTECH '88 CONFERENCE (April 1988), 82-83 (hereinafter "Swinehart 2").

e. Polle T. Zellweger, *Active Paths through Multimedia Documents*, DOCUMENT MANIPULATION AND TYPOGRAPHY, J.C. AN VILET (ED.), CAMBRIDGE UNIVERSITY PRESS (1988) (hereinafter "Zellweger 2").

Exhibit D Vin, Harrick M., et al., *Multimedia Conferencing in the Etherphone Environment*, IEEE COMPUTER SOCIETY (October 1991) ("**Vin**"); and

Exhibit E Droms, R., Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("**RFC 1531**")

Exhibit F Pinard, et al., U.S. Patent No. 5,533,110 ("Pinard")

Exhibit G VocalChat User's Guide, Version 2.0 (1994) ("User's Guide")

Exhibit H VocalChat Readme File, Version 2.02 (June, 1994) ("Readme")

Exhibit I VocalChat 1.01 Networking Information (March 6, 1994) ("VocalChat Networking")

Exhibit J VocalChat Information, Version 2.02 (July 18, 1994) ("Help File")

Exhibit K VocalChat Troubleshooting Help File, Version 2.02 (July 18, 1994) ("Troubleshooting Help File")

III. OVERVIEW OF THE '704 PATENT

Before providing detailed explanations of the pertinency and manner of applying the cited prior art to the claims, presented here is an overview of the '704 patent and its prosecution history. The '704 patent issued on August 22, 2000, and includes 44 claims, of which claims 1, 2, 4, 10, 21, 32, 33, 38, 43, and 44 are independent.

A. Subject Matter of the '704 Patent

The '704 patent describes two different techniques for locating computer processes on a network. Referring to Figure 1 of the '704 patent (reproduced below), one technique relies on a "connection server" (26) to locate processes and a second technique relies on a "mail server" (28) to locate processes.[3] According to the first technique, each computer (referred to as a "processing unit" in the '704 patent) registers its IP addresses with the connection server (26). The IP address of each "online" computer is stored within a database (34) on the connection server. As described in the '704 patent (referring to Figure 1):

> Upon the first user initiating the point-to-point Internet protocol when the first user is logged on to Internet 24, the first processing unit 12 automatically transmits its associated E-mail address and its dynamically allocated IP address to the connection server 26. The connection server 26 then stores these addresses in the database 34 and timestamps the stored addresses using timer 32. The first user operating the first processing unit 12 is thus established in the database 34 as an active on-line party available for communication using the disclosed point-to-point Internet protocol. Similarly, a second user operating the second processing unit 22, upon connection to the Internet 24 through a connection service provider, is processed by the connection server 26 to be established in the database 34 as an active on-line party.[4]

[3] The first technique is referred to as the "primary point-to-point Internet protocol" and the second technique is referred to as the "secondary point-to-point internet protocol." *See*, *e.g.,* '704 patent, Col. 5, line 55 - Col. 6, line 29.
[4] '704 patent, Col. 5, lines 25-38.

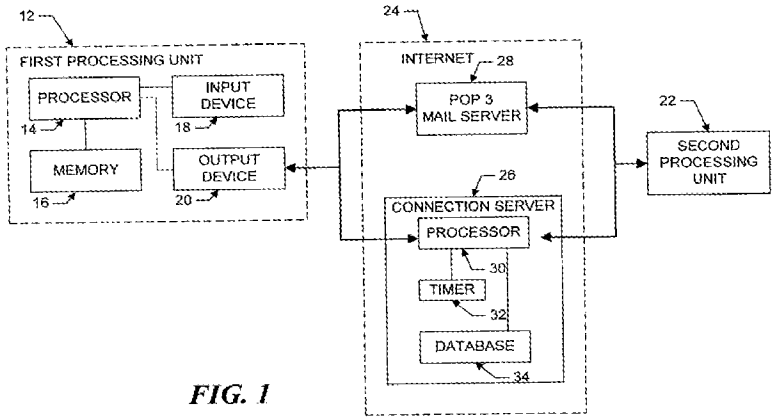


FIG. 1

In order to initiate a connection with the second computer (22) on the network, the first computer (12) retrieves the current IP address of the second computer from the connection server (26). Once the first computer knows the IP address of the second computer, it can establish a point-to-point connection with the second computer. As described in the '704 patent:

> The first processing unit 12 then sends a query, including the E-mail address of the callee, to the connection server 26. The connection server 26 then searches the database 34 to determine whether the callee is logged-in by finding any stored information corresponding to the callee's E-mail address indicating that the callee is active and on-line. If the callee is active and on-line, the connection server 26 then performs the primary point-to-point Internet protocol; i.e. the IP address of the callee is retrieved from the database 34 and sent to the first processing unit 12. The first processing unit 12 may then directly establish the point-to-point Internet communications with the callee using the IP address of the callee.[5]

The second technique for locating computers on a network (the "secondary point-to-point Internet protocol") utilizes the email server (28) illustrated in Figure 1. The second technique is used "if the connection server 26 is non-responsive, inoperative, and/or unable to perform the primary point-to-point Internet protocol, as a non-responsive condition."[6] Using the second technique, the first computer (12) transmits an email message which includes the IP address of the first user and a session number (referred to as a "<ConnectRequest>" message).[7] After receiving the email message from the mail server, the second computer (22) uses the IP address and session number to establish a point-to-point connection with the first computer (12). As described in the '704 patent:

---

[5] '704 patent, Col. 5, lines 55-67.
[6] '704 patent, Col. 6, lines 20-23.
[7] '704 patent, Col. 6, lines 31-36.

> Upon detecting and/or receiving the incoming E-mail signal from the first processing unit 12, the second processing unit 22 may assign or may be assigned a temporary IP address. Therefore, the delivery of the E-mail through the Internet 24 provides the second processing unit 22 with a session number as well as IP addresses of both the first processing unit 12 and the second processing unit 22.
>
> Point-to-point communication may then be established by the processing units 12, 22. For example, the second processing unit 22 may process the E-mail signal to extract the <ConnectRequest> message, including the IP address of the first processing unit 12 and the session number. The second processing unit 22 may then open a socket and generate a <ConnectOK> response signal, which includes the temporary IP address of the second processing unit 22 as well as the session number.[8]

While the independent claims of the '704 patent are not expressly limited to a particular protocol standard, the embodiments described in the '704 patent utilize the TCP/IP protocol.[9] Thus, the focus of the '704 patent is a central repository of IP addresses which is queried to locate computers on a network.

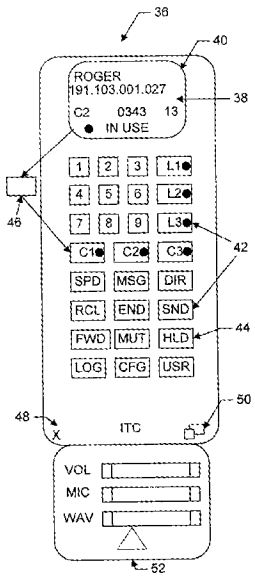


FIG. 5

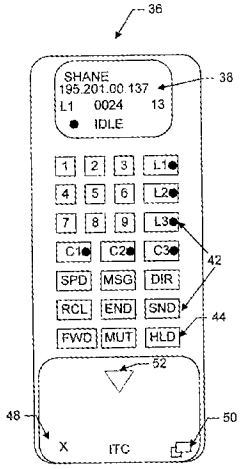


FIG. 6

The '704 patent also describes a graphical user interface ("GUI") for managing calls on a computer. The GUI, illustrated in Figures 5 and 6 of the '704 patent (reproduced above), includes a status area (38) which is used to indicate . . .

> . . . a called user by name and/or by IP address or telephone number; a current function such as C2; a current time; a current operating status such as "IN USE",

[8] '704 patent, Col. 7, lines 10-25.
[9] *See, e.g.*, '704 patent, Col. 5, lines 8-14 and 22-38.

> and other control icons such as a down arrow icon 40 for scrolling down a list of parties on a current conference line. The operating status may include such annunciators as "IN USE", "IDLE", "BUSY", "NO ANSWER", "OFFLINE", "CALL", "DIALING", "MESSAGES", and "SPEEDDIAL."[10]

Figures 5 and 6 also illustrate a set of graphical icons (42) which are "configured to substantially simulate a telephone handset or a cellular telephone interface."[11] The icons provide functions typically found on a telephone such as speed dial (SPD), hold (HLD), send (i.e., initiate call) (SND), end call (END), mute (MUT). Icons are also provided to indicate individual telephone "lines" (L1-L3) and "conference lines" (C1-C3). An active call may be transferred to a different line by "by clicking and dragging the status area 38, which is represented by a reduced icon 46. Dragging the reduced icon 46 to any one of line icons L1-L4 transfers the called party in use to the selected line, and dragging the reduced icon 46 to any one of conference line icons C1-C3 adds the called party to the selected conference call."[12]

B. Prosecution History of the '704 Patent

The application which resulted in the '704 patent was filed on September 25, 1995. The '704 patent application[13] initially included a total of 20 claims. This was extended to 53 claims via a preliminary amendment.[14]

The PTO mailed a first Office Action on June 2, 1997, rejecting all 53 claims under 35 U.S.C. § 103 as being unpatentable in view of several prior art references, including Civanlar, U.S. Patent No. 5,581,552 ("Civanlar"). As noted by the Examiner, Civanlar discloses "a communication protocol in which the requesting node sends a request for communication with another node through an address server, which contains an address database, to obtain the address and routing information necessary to complete the communication."[15]

The Applicants filed an Amendment and Response on December 2, 1997 in which they added claims 54-68 and distinguished the invention from the cited references as follows:

> Applicant's invention provides techniques for determining the current dynamically assigned network protocal [sic] address of a user process connected

---

[10] '704 patent, Col. 8, lines 42-50.
[11] '704 patent, Col. 8, lines 58-60.
[12] '704 patent, Col. 9, lines 36-42.
[13] Application serial no. 08/533,11 resulting in the '704 patent is referred to as the "'704 application."
[14] *See* Preliminary Amendment (April 5, 1996).
[15] Office Action (June 2, 1997), page 3.

> to the network. The first technique utilizes a dedicated server which acts as a network address/information directory from which calling processes can obtain information. When a first process connects to the networks, the process logs-on to the server and provides the server with the network protocal [sic] address under which the process is currently operating. A second process wishing to establish communications with the first process, connects to the server and request the network protocal [sic] address under which the first process is currently operating. Upon receipt of the network protocal [sic] address of the first process, the second process establishes communications with the first process directly, without any intervention [sic] from the address/information server.[16]

The Applicants also submitted a Declaration of Prior Invention under 37 U.S.C. § 1.131, stating "to overcome the rejection of all claims under 35 U.S.C. § 103 as being unpatentable over Civanlar, et al. in view of Morgan et al. and/or further in view of December et al . . . In light of the declaration and accompanying exhibits, all rejections based on the Civanlar et al. reference are deemed moot."[17]

The PTO mailed a second Office Action on April 14, 1998 indicating a restriction requirement under 35 U.S.C. § 121. The Applicants mailed a response to the restriction requirement on August 11, 1998, electing a group of claims (Group 1).

An Office Action mailed on October 28, 1998 rejected all pending claims under 35 U.S.C. § 102(e) and 35 U.S.C. § 103. The Applicants mailed an Amendment and Response on March 1, 1999, cancelling claims 1-4 and 6-11 and amending numerous claims. The Applicants again distinguished the alleged "invention" over the cited prior art, stating:

> Applicants' invention solves a fundamental problem associated with the Internet. . . The problem is: How can a global network user be located if he/she has no permanent network address?
>
> Applicants have disclosed a solution to the above-described problem. The solution utilizes a client/server system. In the disclosed system, a client process contacts a dedicated address directory server and forwards to the server the network protocol address to which it has been assigned upon connection to the computer network, along with other identification information. The dedicated address directory server maintains a compilation or list of entries, each of which contain a process identifier and the corresponding network protocol address forwarded to the server by the process itself. Other processes wishing to contact a desired target process simply

[16] Office Action Response (December 2, 1997), page 8.

[17] Amendment and Response (December 2, 1997), page 7. We question the sufficiency of the support document, "webph.doc," submitted with the Declaration of Prior Invention. However, because the prior art we rely upon predates the priority date of this document, we did not perform a detailed analysis related to this document.

query the address directory server to determine whether the target process is on-line and the current network protocol address at which the target process is located.[18]

In addition, with respect to independent Claims 10 and 21 of the '704 patent (and associated dependent claims), the Applicants argued that these claims were "directed to a method for establishing a point-to-point communication link with the user interface of a client process by associating elements representing a communication line and various processes."[19]

A Notice of Allowability was mailed on May 25, 1999, allowing Claims 21, 23-24, 26-64, 66, and 67 (which issued as Claims 1-44). The '704 patent issued on August 22, 2000.

IV. SUBSTANTIAL NEW QUESTION (SNQ) OF PATENTABILITY AS REQUIRED BY 37 C.F.R 1.510 (b)(1)

The following section provides a list of the SNQs and detailed explanation of the prior art references relied upon in the present request for the SNQ, including references not previously considered by the PTO.

A. SNQs Raised by NetBIOS

A SNQ as to Claims 1-7 and 32-44 is raised by NetBIOS. NetBIOS anticipates all of the limitations of these claims, including teachings of Civanlar, which were used by the Examiner in a § 103 rejection against the instant claims. This reference was removed from consideration due to the acceptance of a 37 C.F.R. § 1.131 declaration which swore behind the date of Civanlar. While Civanlar shows an address server for storing network protocol addresses usable by network nodes to establish point-to-point communications, the question of patentability was removed when the reference was antedated. *See, e.g.*, Civanlar, col. 3, lines 1-4 ("The address server contains an address data base for performing address resolution, i.e., translation, between the at least two addresses of each ELAN end-point in response to requests for such translations.").

NetBIOS, which was not cited or discussed in the prosecution of the '704 patent, presents a SNQ of patentability because it, like Civanlar, discloses an address server (referred to as a "NetBIOS Name Server" or "NBNS") with an address database for storing network protocol addresses usable by network nodes to establish point-to-point communications, as discussed in

[18] Amendment and Response (March 1, 1999), page 14.
[19] Office Action response (March 1, 1999), page 17.

detail below. *See, e.g.*, NetBIOS at 367 (describing how the NBNS acts as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."). *See also id.* at 388 ("Name query transactions are initiated by endnodes to obtain the IP address(es) and other attributes associated with a NetBIOS name.").

Recall also that, during prosecution of the '704 patent, the Applicants argued that the claimed invention . . .

> . . . utilizes a dedicated server which acts as a network address/information directory from which calling processes can obtain information. When a first process connects to the networks, the process logs-on to the server and provides the server with the network protocal [sic] address under which the process is currently operating. A second process wishing to establish communications with the first process, connects to the server and request the network protocal [sic] address under which the first process is currently operating. Upon receipt of the network protocal [sic] address of the first process, the second process establishes communications with the first process directly, without any intervention [sic] from the address/information server.

Office Action Response (December 2, 1997), page 8. This is precisely the manner in which nodes in NetBIOS register their own IP addresses with the NBNS and query the NBNS for the IP addresses of other nodes. *See, e.g., id. at* 397 ("The NetBIOS session service begins after one or more IP addresses have been found for the target name. . . . NetBIOS session service transactions, packets, and protocols are identical for all end-node types. They involve only directed (point-to-point) communications.") (emphasis added).

For all of the foregoing reasons, NetBIOS would be considered important in deciding the question of patentability for all independent claims of the '704 patent and, accordingly, presents a SNQ of patentability, particularly with respect to Claims 1-7 and 32-44.

B. SNQs Raised by NetBIOS in view of RFC 1531

A SNQ as to Claims 1-7 and 32-44 is raised by NetBIOS in view of RFC 1531. NetBIOS and RFC 1531 were not cited or discussed alone, or in combination in the prosecution of the '704 patent. The combination presents a SNQ of patentability because NetBIOS discloses an address server (referred to as a "NetBIOS Name Server" or "NBNS") for storing network protocol addresses usable by network nodes to establish point-to-point communications (as described above), and RFC 1531 discloses how TCP/IP addresses are assigned dynamically by a

Dynamic Host Configuration Protocol (DHCP) server. *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). As argued by Applicants in the prosecution of the '704 patent:

> Applicant's invention provides techniques for determining the current dynamically assigned network protocol address of a user process connected to the network. The first technique utilizes a dedicated server which acts as a network address/information directory from which calling processes can obtain information. When a first process connects to the networks, the process logs-on to the server and provides the server with the network protocal [sic] address under which the process is currently operating. A second process wishing to establish communications with the first process, connects to the server and request the network protocal [sic] address under which the first process is currently operating. Upon receipt of the network protocal [sic] address of the first process, the second process establishes communications with the first process directly, without any intervention [sic] from the address/information server.

Office Action Response (December 2, 1997), page 8. As described above, these features are all described explicitly in NetBIOS except for "dynamically assigned" network protocol addresses. Consequently, NetBIOS in view of RFC 1531 presents a SNQ of patentability for all claims which require the dynamic assignment of network protocol addresses, as discussed in detail below. Such combination would be considered important in deciding the question of patentability and accordingly present a SNQ of patentability, particularly with respect to Claim 33 which explicitly requires "locating processes having dynamically assigned network protocol addresses" and with respect to various other claims which require that the network protocol address is received "following connection to the computer network." *See, e.g.*, Claims 1, 4, and 38.

C. SNQs Raised by NetBIOS in view of Pinard

A SNQ as to Claims 10-17, 19-28, and 30-31 is raised by NetBIOS in view of Pinard. NetBIOS and Pinard were not cited or discussed alone, or in combination in the prosecution of the '704 patent. Recall that during prosecution of the '704 patent, the Applicants argued that these claims were "directed to a method for establishing a point-to-point communication link with the user interface of a client process by associating elements representing a communication line and various processes." Office Action response (March 1, 1999), page 17. Consequently,

the combination of NetBIOS and Pinard presents a substantial new question of patentability because NetBIOS discloses an address server for storing network protocol addresses usable by network nodes to establish point-to-point communications and Pinard discloses graphical elements representing communication lines and callees that may be clicked and dragged to establish and terminate calls, set up conference calls, and place calls on hold, as discussed in detail below. *See, e.g.*, Pinard, Figures 2-16 and associated text. Such combination would be considered important in deciding the question of patentability and accordingly present a substantial new question of patentability, particularly with respect to Claims 10-17, 19-28, 30-31.

D. SNQs Raised by NetBIOS in view of Pinard and further in view of VocalChat User's Guide

A SNQ as to Claims 18 and 29 is raised by NetBIOS in view of Pinard and further in view of VocalChat. NetBIOS, Pinard, and VocalChat were not cited or discussed alone, or in combination, in the prosecution of the '704 patent. The combination presents a substantial new question of patentability because the combination discloses all of the features from NetBIOS and Pinard described above and, in addition, the VocalChat User's Guide describes a "communication line on mute status" as recited in Claims 18 and 29. As described in the VocalChat User's Guide, "Manual Activation can also be used like the MUTE option in many phones: it lets you talk without being heard on the other user's system." User's Guide, page 57. Such combination would be considered important in deciding the question of patentability and accordingly would present a substantial new question of patentability, particularly with respect to Claims 18 and 29.

E. SNQs Raised by Etherphone

A SNQ as to Claims 1-2, 4-7, 10-12, 14, 19-23, 25, and 30-44 is raised by Etherphone. Etherphone anticipates all of the limitations of these claims, including teachings of Civanlar which were used in a § 103 rejection against the instant claims. This reference was removed from consideration due to the acceptance of a 37 C.F.R. § 1.131 declaration which swore behind the date of Civanlar. While Civanlar shows an address server for storing network protocol addresses usable by network nodes to establish point-to-point communications, the question of patentability was removed when the reference was antedated. *See, e.g.*, Civanlar, col. 3, lines 1-

4 ("The address server contains an address data base for performing address resolution, i.e., translation, between the at least two addresses of each ELAN end-point in response to requests for such translations.").

Etherphone, which was not cited or discussed in the prosecution of the '704 patent, presents a SNQ of patentability because it, like Civanlar, discloses an address server (referred to as a "Voice Control Server" or "Telephone Control Server") for storing network protocol addresses usable by network nodes to establish point-to-point communications, as discussed in detail below. *See, e.g.*, Swinehart 1, page 4 ("The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server.").

In addition, during prosecution of the '704 patent, the Applicants argued that the claimed invention . . .

> . . . utilizes a dedicated server which acts as a network address/information directory from which calling processes can obtain information. When a first process connects to the networks, the process logs-on to the server and provides the server with the network protocal [sic] address under which the process is currently operating. A second process wishing to establish communications with the first process, connects to the server and request the network protocal [sic] address under which the first process is currently operating. Upon receipt of the network protocal [sic] address of the first process, the second process establishes communications with the first process directly, without any intervention [sic] from the address/information server.

Office Action Response (December 2, 1997), page 8. This is precisely the manner in which Etherphones/workstations described in Etherphone register their own IP addresses with the Voice Control Server and query the Voice Control Server for the IP addresses of other Etherphones/workstations. *See, e.g.,* Swinehart 1, page 4 (after receiving the IP address of a callee's Etherphone from the Voice Control Server, "voice datagrams are transmitted directly among the participants, bypassing the control server."). *See also id.* ("The *telephone control server* controls voice conversations, implements the stand-alone behavior of telephone instruments and coordinates the activities of workstations and adjacent telephones in their implementation of the various voice capabilities . . . It uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations and the registration of *visitors* in the offices of their colleagues."); Swinehart 1, page 2 ("Calls are to

individuals, not locations . . . Logging in tells the telephone system where Karmen is."); Zellweger 1, page 5 ("An additional feature, called visiting, allows him to register his presence with a second workstation or Etherphone, such as during a meeting. Registering with the destination location allows users to travel more freely than forwarding calls from the home location does.").

For the foregoing reasons, Etherphone would be considered important in deciding the question of patentability and accordingly would present a SNQ of patentability, particularly with respect to Claims 1-2, 4-7, 10-12, 14, 19-23, 25, and 30-44.

F. SNQ Raised by Etherphone in view of NetBIOS

A SNQ as to Claim 3 is raised by Etherphone in view of NetBIOS. Etherphone and NetBIOS were not cited or discussed alone, or in combination in the prosecution of the '704 patent. The combination presents a SNQ of patentability because Etherphone discloses an address server (referred to as a "Voice Control Server" or "Telephone Control Server") for storing network protocol addresses usable by network nodes to establish point-to-point communications, as mentioned above, and NetBIOS discloses "a timer, operatively coupled to the processor, for time stamping the network protocol addresses stored in the memory" as recited in Claim 3. *See, e.g.*, NetBIOS at 382 ("[t]he NBNS may impose a 'time-to-live' on each name it registers. The registering node is made aware of this time value during the name registration procedure."). Such combination would be considered important in deciding the question of patentability and accordingly would present a SNQ of patentability, particularly with respect to Claim 3.

G. SNQ Raised by Etherphone in view of Vin

A SNQ as to Claim 32 is raised by Etherphone in view of Vin. Etherphone and Vin were not cited or discussed alone, or in combination in the prosecution of the '704 patent. The combination presents a SNQ of patentability because Etherphone discloses an address server (referred to as a "Voice Control Server" or "Telephone Control Server") for storing network protocol addresses usable by network nodes to establish point-to-point communications, as mentioned above, and Vin discloses using TCP/IP as the network protocol in an Etherphone system. *See, e.g.*, Vin, page 77, Figure 5 (illustrating a "protocol stack and format" used in an

Etherphone system which includes internet protocol (IP) packets). Such combination would be considered important in deciding the question of patentability and accordingly would present a SNQ of patentability, particularly with respect to Claim 32, which requires the "Internet protocol."

H. SNQs Raised by Etherphone in view of Vin and further in view of RFC 1531

A SNQ as to Claim 33 is raised by Etherphone in view of NetBIOS and further in view of RFC 1531. Etherphone, Vin, and RFC 1531were not cited or discussed alone, or in combination in the prosecution of the '704 patent. The combination presents a SNQ of patentability because Etherphone discloses an address server (referred to as a "Voice Control Server" or "Telephone Control Server") for storing network protocol addresses usable by network nodes to establish point-to-point communications, Vin discloses the use of TCP/IP as the network protocol in an Etherphone system, as mentioned above, and RFC 1531 discloses how TCP/IP addresses are assigned dynamically by a Dynamic Host Configuration Protocol (DHCP) server. *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). As argued by Applicants in the prosecution of the '704 patent:

> Applicant's invention provides techniques for determining the current dynamically assigned network protocol address of a user process connected to the network. The first technique utilizes a dedicated server which acts as a network address/information directory from which calling processes can obtain information. When a first process connects to the networks, the process logs-on to the server and provides the server with the network protocal [sic] address under which the process is currently operating. A second process wishing to establish communications with the first process, connects to the server and request the network protocal [sic] address under which the first process is currently operating. Upon receipt of the network protocal [sic] address of the first process, the second process establishes communications with the first process directly, without any intervention [sic] from the address/information server.

Office Action Response (December 2, 1997), page 8. Etherphone, Vin and RFC 1531, in combination, disclose all of these features. Consequently, such combination would be considered important in deciding the question of patentability and accordingly presents a SNQ of patentability, particularly with respect to Claim 33 which requires "dynamically assigned network protocol addresses."

I. SNQs Raised by Etherphone in view of Pinard

A SNQ as to Claims 10-17, 19-28, and 30-31 is raised by Etherphone in view of Pinard. Etherphone and Pinard were not cited or discussed alone, or in combination in the prosecution of the '704 patent. Recall that during prosecution of the '704 patent, the Applicants argued that these claims were "directed to a method for establishing a point-to-point communication link with the user interface of a client process by associating elements representing a communication line and various processes." Office Action response (March 1, 1999), page 17. The combination presents a SNQ of patentability because Etherphone discloses an address server for storing network protocol addresses usable by network nodes to establish point-to-point communications and Pinard discloses graphical elements representing communication lines and callees that may be clicked and dragged to establish and terminate calls, set up conference calls, and place calls on hold, as discussed in detail below. *See, e.g.*, Pinard, Figures 2-16 and associated text. Such combination would be considered important in deciding the question of patentability and accordingly present a SNQ of patentability, particularly with respect to Claims 10, 12-17, 21, and 24-28.

J. SNQ Raised by Etherphone in view of Pinard and further in view of VocalChat User's Guide

A SNQ as to Claims 18 and 29 is raised by Etherphone in view of Pinard and further in view of VocalChat. Etherphone, Pinard, and VocalChat were not cited or discussed alone, or in combination in the prosecution of the '704 patent. The combination presents a SNQ of patentability because Etherphone discloses an address server for storing network protocol addresses usable by network nodes to establish point-to-point communications; Pinard discloses graphical elements representing communication lines and callees that may be clicked and dragged to establish and terminate calls, set up conference calls, and place calls on hold (as described above); and VocalChat discloses a "communication line on mute status" as recited in Claims 18 and 29. As described in the VocalChat User's Guide, "Manual Activation can also be used like the MUTE option in many phones: it lets you talk without being heard on the other user's system." User's Guide, page 57. Such combination would be considered important in deciding the question of patentability and accordingly presents a SNQ of patentability, particularly with respect to Claims 18 and 29.

K. SNQs Raised by VocalChat User's Guide in view of VocalChat Readme, and further in view of VocalChat Networking, and further in view of VocalChat Help File, and further in view of VocalChat Troubleshooting Help File (collectively "VocalChat" or "the VocalChat references")

A SNQ as to Claims 1-2, 4, 7, 10-11, 19-22, 30-42 is raised by VocalChat User's Guide in view of Readme, and further in view of VocalChat Networking, Help File, and Troubleshooting Help File (collectively referred to as "VocalChat"). A strong motivation to combine all of these references exists because they all describe the same VocalChat system. VocalChat anticipates all of the limitations of these claims, including teachings of Civanlar, which were used in a103 rejection against the instant claims. This reference was removed from consideration due to the acceptance of a 37 C.F.R. § 1.131 declaration which swore behind the date of Civanlar. While Civanlar shows an address server for storing network protocol addresses usable by network nodes to establish point-to-point communications, the question of patentability was removed when the reference was antedated. *See, e.g.*, Civanlar, col. 3, lines 1-4 ("The address server contains an address data base for performing address resolution, i.e., translation, between the at least two addresses of each ELAN end-point in response to requests for such translations.").

VocalChat User's Guide, Readme, VocalChat Networking, Help File, and Troubleshooting Help File, which were not cited or discussed in the prosecution of the '704 patent, present a SNQ of patentability because they, like Civanlar, disclose an address server (referred to as a "Post Office") for storing network protocol addresses usable by network nodes to establish point-to-point communications, as discussed in detail below. *See, e.g.*, Readme, page 2 ("VocalChat creates a central directory on the network, shared by all users called 'Post-Office.' All users must use the same Post-Office, otherwise they won't be able to communicate or leave messages to each other. This means that all users must be attached to one file-server which will be used for the Post-Office, and all have write permission for the Post-Office directory."). In TCP/IP implementations, the Post Office directory includes a "Connection List" file (CONNLIST.VC) containing the unique usernames and IP addresses of connected VocalChat users. *See, e.g.*, Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory."). In earlier implementations, the Connection List file was called the "USERS" File. *See, e.g.*, VocalChat Network Information, page 10 ("When the network used is

not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address.")

Recall also that, during prosecution of the '704 patent, the Applicants argued that the claimed invention . . .

> . . . utilizes a dedicated server which acts as a network address/information directory from which calling processes can obtain information. When a first process connects to the networks, the process logs-on to the server and provides the server with the network protocal [sic] address under which the process is currently operating. A second process wishing to establish communications with the first process, connects to the server and request the network protocal [sic] address under which the first process is currently operating. Upon receipt of the network protocal [sic] address of the first process, the second process establishes communications with the first process directly, without any intervention [sic] from the address/information server.

Office Action Response (December 2, 1997), page 8. This is precisely the manner in which VocalChat clients register their own IP addresses with the Post Office server and query the Post Office Server for the IP addresses of other VocalChat clients. *See, e.g.*, Help File, page 22 ("VocalChat will use the CONNLIST.VC files to get network addresses and a user name should be entered in the Setup for each user."). *See also* VocalChat Network Information, page 10 ("VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address.").

For the foregoing reasons, VocalChat User's Guide in view of Readme, and further in view of VocalChat Networking, Help File, and Troubleshooting Help File would be considered important in deciding the question of patentability and accordingly would present a SNQ of patentability, particularly with respect to Claims 1-2, 4, 7, 10-11, 19-22, 30-42.

L. SNQs Raised by the VocalChat References in view of RFC 1531

A SNQ as to Claims 1-2, 4, 7, 10-11, 19-22, and 30-42 is raised by VocalChat User's Guide in view of Readme, and further in view of VocalChat Networking, Help File, and Troubleshooting Help File, and further in view of RFC 1531. These prior art references were not cited or discussed alone, or in combination in the prosecution of the '704 patent. The combination presents a SNQ of patentability because VocalChat User's Guide, Readme,

VocalChat Networking, Help File, and Troubleshooting Help File disclose an address server (referred to as a "Post Office" server) for storing network protocol addresses usable by network nodes to establish point-to-point communications over a TCP/IP network (as described above), and RFC 1531 discloses how TCP/IP addresses are assigned dynamically by a Dynamic Host Configuration Protocol (DHCP) server. *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). In addition, as argued by Applicants in the prosecution of the '704 patent:

> Applicant's invention provides techniques for determining the current dynamically assigned network protocol address of a user process connected to the network. The first technique utilizes a dedicated server which acts as a network address/information directory from which calling processes can obtain information. When a first process connects to the networks, the process logs-on to the server and provides the server with the network protocal [sic] address under which the process is currently operating. A second process wishing to establish communications with the first process, connects to the server and request the network protocal [sic] address under which the first process is currently operating. Upon receipt of the network protocal [sic] address of the first process, the second process establishes communications with the first process directly, without any intervention [sic] from the address/information server.

Office Action Response (December 2, 1997), page 8. As described above, these features are all described explicitly in the VocalChat references except for "dynamically assigned" network protocol addresses. Consequently, the VocalChat references in view of RFC 1531 present a SNQ of patentability for all claims which require the dynamic assignment of network protocol addresses, as discussed in detail below. Such combination would be considered important in deciding the question of patentability and accordingly present a SNQ of patentability, particularly with respect to Claim 33 which explicitly requires "locating processes having dynamically assigned network protocol addresses" and with respect to various other claims which require that the network protocol address is received "following connection to the computer network." *See, e.g.*, Claims 1, 4, and 38.

M. SNQ Raised by the VocalChat References in view of NetBIOS

A SNQ as to Claim 3 is raised by VocalChat in view of NetBIOS. VocalChat and NetBIOS were not cited or discussed alone, or in combination in the prosecution of the '704 patent. The combination presents a SNQ of patentability because VocalChat discloses an address server (referred to as a "Post Office" server) for storing network protocol addresses usable by network nodes to establish point-to-point communications, as mentioned above, and NetBIOS discloses "a timer, operatively coupled to the processor, for time stamping the network protocol addresses stored in the memory" as recited in Claim 3. *See, e.g.*, NetBIOS at 382 ("[t]he NBNS may impose a 'time-to-live' on each name it registers. The registering node is made aware of this time value during the name registration procedure."). Such combination would be considered important in deciding the question of patentability and accordingly would present a SNQ of patentability, particularly with respect to Claim 3.

N. SNQs Raised by the VocalChat References in view of Pinard

A SNQ as to Claims 12-18 and 23-29 is raised by VocalChat User's Guide in view of Readme, and further in view of VocalChat Networking, Help File, and Troubleshooting Help File (collectively referred to as "VocalChat" or "the VocalChat references"), and further in view of Pinard. These references were not cited or discussed alone, or in combination in the prosecution of the '704 patent. Recall that during prosecution of the '704 patent, the Applicants argued that these claims were "directed to a method for establishing a point-to-point communication link with the user interface of a client process by associating elements representing a communication line and various processes." Office Action response (March 1, 1999), page 17. The combination presents a SNQ of patentability because VocalChat User's Guide, Readme, VocalChat Networking, Help File, and Troubleshooting Help File disclose an address server for storing network protocol addresses usable by network nodes to establish point-to-point communications and Pinard discloses graphical elements representing communication lines and callees that may be clicked and dragged to establish and terminate calls, set up conference calls, and place calls on hold, as discussed in detail below. *See, e.g.*, Pinard, Figures 2-16 and associated text. Such combination would be considered important in deciding the

question of patentability and accordingly present a SNQ of patentability, particularly with respect to Claims 12-18 and 23-29.

V. OVERVIEW OF THE PRIOR ART REFERENCES PRESENTING A SNQ OF PATENTABILITY

A. NetBIOS

The Network Basic Input/Output System, known as NetBIOS, was originally developed for IBM's PC-Network in the early 1980s. In March 1987, the NetBIOS Working Group of the Internet Engineering Task Force released Request for Comments 1001 ("RFC 1001"), titled "Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Concept and Methods," and Request for Comments 1002 ("RFC 1002"), titled "Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications." Both of these documents were republished in NetBIOS in 1992 (Exhibit B).[20]

NetBIOS is a software interface which allows applications on different computers to communicate within a computer network, such as a local area network or the Internet. "NetBIOS applications employ NetBIOS mechanisms to locate resources, establish connections, send and receive data with an application peer, and terminate connections." NetBIOS at 359. NetBIOS "defines a proposed standard protocol to support NetBIOS services in a TCP/IP environment. Both local network and Internet operation are supported." *Id.* at 350.

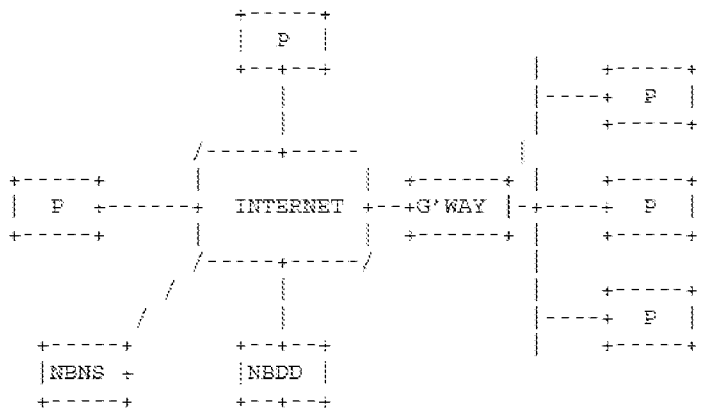


Figure From NetBIOS, page 371 (RFC 1001, page 22)

[20] The pages of NetBIOS are numbered consecutively including RFC 1001 and RFC 1002. When referencing a page in NetBIOS the consecutively numbered pages of RFC 1001 and RFC 1002 are used.

As illustrated in the figure above, NetBIOS enables point-to-point communications between two or more "point-to-point" nodes (also referred to as "P nodes"). The point-to-point connections are established over the Internet or a local area network (LAN). A NetBIOS Name Server ("NBNS") coupled to the point-to-point nodes over the Internet (or other network) provide a dedicated directory service for associating node names with IP addresses. In operation, NetBIOS point-to-point nodes register distinguishing names and corresponding IP addresses with the NBNS. When a node makes a point-to-point connection it first "queries" the NBNS to obtain the current IP address of other nodes. Having obtained the target node's IP address from the NBNS, the originating node can establish "directed (point-to-point) communications." *Id.* at 397 ("The NetBIOS session service begins after one or more IP addresses have been found for the target name. . . . NetBIOS session service transactions, packets, and protocols are identical for all end-node types. They involve only directed (point-to-point) communications.") (emphasis added). In the December 2, 1997 Office Action Response on page 8, the Patent Owner indicated, among other things, that "the second process establishes communications with the first process directly, without any intervention [sic] from the address/information server." Accordingly, the examiner would have considered NetBIOS important in determining patentability.

"Name query transactions are initiated by endnodes to obtain the IP address(es) and other attributes associated with a NetBIOS name." *Id.* at 388. *See also id.* at 376 (describing how "[e]very node has a permanent unique name."); *id.* at 377 (describing how NetBIOS point-to-point nodes perform "name resolution" by "ask[ing]" the NBNS for the IP address corresponding to a NetBIOS end-node identified by name); *id.* ("Name query (also known as 'resolution' or 'discovery') is the procedure by which the IP address(es) associated with a NetBIOS name are discovered."); *id.* at 388 ("Name query transactions are initiated by end nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."); *id.* at 389 ("An NBNS answers queries from a P [point-to-point] node with a list of IP address and other information" for the queried name.).

This is a comprehensive reference for Internet and LAN point-to-point connections as will be seen by the citation to NetBIOS in the text below and in the claim charts of Exhibit P. As discussed above, Civanlar was utilized in a § 103 rejection against the instant claims. This reference was removed from consideration due to the acceptance of a 37 C.F.R. § 1.131 declaration which swore behind the date of Civanlar. While this reference shows an address

server for storing network protocol addresses usable by network nodes to establish point-to-point communications, the question of patentability was removed when the reference was antedated. Accordingly, NetBIOS, which was not cited or discussed in the prosecution of the '704 patent, presents a SNQ of patentability because it, like Civanlar, discloses an NBNS with an address database for storing network protocol addresses usable by network nodes to establish point-to-point communications, as discussed in detail below. *See, e.g.*, NetBIOS at 367 (describing how the NBNS acts as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."). Consequently, NetBIOS provides a SNQ of patentability as the question of patentability based on Civanlar was removed.

Additionally, NetBIOS was not cited during the prosecution of the '704 patent and, as shown above, would have been considered important to an examiner in deciding patentability of the '704 patent, accordingly, it presents a substantially new question of patentability.

B. Etherphone

The Etherphone system (Exhibit C) "consists of microprocessor-based electronic telephones, a centralized switching server, a voice file server, and workstation programs to support voice communications and voice recording services. From a workstation, a user can place and receive telephone calls, maintain private telephone directories, and manage a database of voice messages."[21]

Figure 1 of Terry (reproduced below) provides an architectural overview of the Etherphone system. A Voice Control Server (sometimes called a "Telephone Control Server") registers the network addresses of workstations/Etherphones of users and provides the network addresses to requesting workstations/Etherphones upon request to establish calls between users. As described in Etherphone:

> The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server.[22]

[21] Zellweger 2, page 11.
[22] Swinehart 1, page 4.

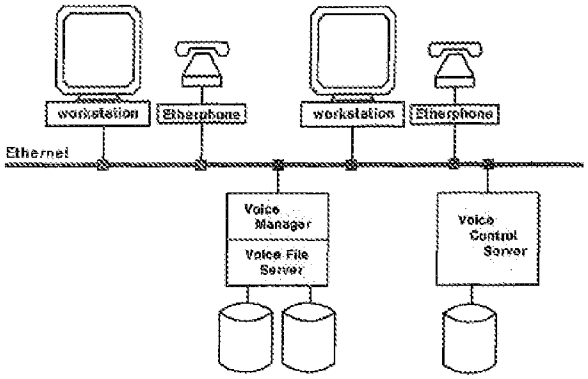


Figure 1. A simple Etherphone system environment.

Thus, after determining the current network addresses for a call, the workstation/Etherphone of the calling user and the workstation/Etherphone of a called user establish a point-to-point ("direct") connection over the network, bypassing the Voice Control Server.

In addition, the Voice Control Server associates user identifiers with each network protocol address. For example, a user may log in to any workstation. Calls to that user will then be directed to that workstation and its associated Etherphone. As described:[23]

> The *telephone control server* controls voice conversations, implements the stand-alone behavior of telephone instruments and coordinates the activities of workstations and adjacent telephones in their implementation of the various voice capabilities. In addition, it stores personal preference information about each user that allows it to support advanced features such as *ring motifs* and *subdued ringing* without involving workstation programs. It uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations and the registration of *visitors* in the offices of their colleagues.[24]

Etherphone also describes a graphical user interface (GUI) to provide various telephony functions. One example of the GUI is provided in Figure 3 of Zellweger 1 (reproduced below). As described in Zellweger 1:

> A variety of convenient workstation dialing methods are provided: a user can . . . select names or numbers from anywhere on the [Etherphone telephone

[23] Swinehart 1, page 4 (underline emphasis added). *See also* Swinehart 1, page 2 ("Calls are to individuals, not locations . . . Logging in tells the telephone system where Karmen is."); Zellweger 1, page 5 ("An additional feature, called visiting, allows him to register his presence with a second workstation or Etherphone, such as during a meeting. Registering with the destination location allows users to travel more freely than forwarding calls from the home location does.").

[24] Swinehart 1, page 4 (underline emphasis added). *See also* Swinehart 1, page 2 ("Calls are to individuals, not locations . . . Logging in tells the telephone system where Karmen is."); Zellweger 1, page 5 ("An additional feature, called visiting, allows him to register his presence with a second workstation or Etherphone, such as during a meeting. Registering with the destination location allows users to travel more freely than forwarding calls from the home location does.").

management windows], use either of two directory tools that present browsable lists of names and associated telephone numbers as speed-dialing buttons, or redial any previously-made call by clicking on its conversation log entry. Calls can also be placed by name or number from the telephone keypad.[25]

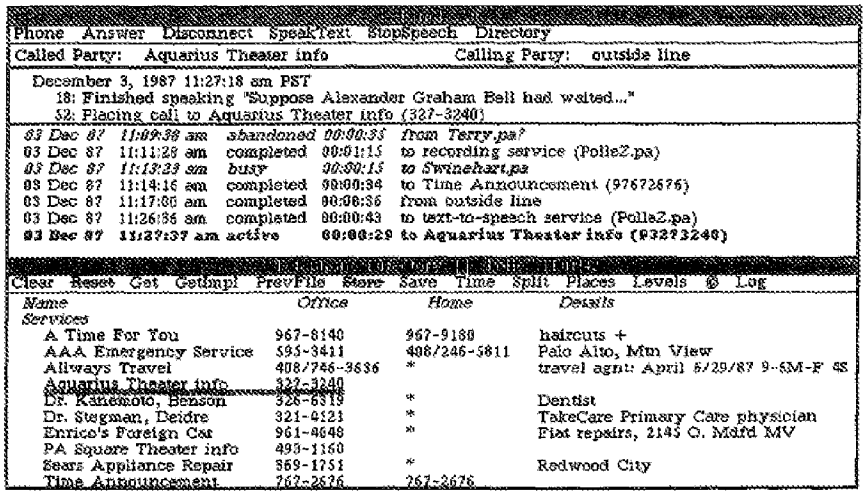


Figure 3 From Zellweger 1

In addition, the Etherphone GUI uses icons to represent callers and telephone lines. For example, Figure 4 of Zellweger 1 (reproduced below) includes a rolodex graphic to represent callees (upper left) and a telephone graphic (upper middle) and a graphic of a person talking on the phone (upper right) to represent telephone lines. When a call is placed to a callee in the rolodex, the name of the callee is associated with the active telephone line graphic (upper right). For example, the active telephone line graphic includes name of the callee and an image of a user speaking on the telephone.

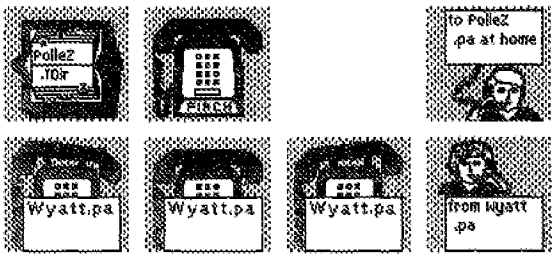


Figure 4. Etherphone system icons. The two icons at the upper left show a closed personal telephone directory and a Finch icon at rest. The icon at the upper right shows an outgoing call to Polle Zellweger's home (username PolleZ.pa). The four bottom icons show several stages of an incoming call from Doug Wyatt: the three left icons of the group are animated during ringing, while the right conversation icon is used after the call has been answered.

[25] Zellweger 1, page 4.

The Etherphone system also supports conference calling and call waiting. For example, Swinehart 1 describes how conference calls may be scheduled with other participants ("negotiated conference calls"). *See, e.g.*, Swinehart 1, page 3. In addition, using the Etherphone system, a user may receive and answer a call while already on an existing call. *See, e.g.*, Swinehart 1, page 2 (describing how users can place and receive other calls during a "background call").

The Etherphone system was intended for use in "multiple networks and communication protocols." Terry, page 3. At least one implementation of the Etherphone system used the Internet Protocol (IP) to support network communications. *See, e.g.*, Vin, page 77, Figure 5 (illustrating a "protocol stack and format" which includes internet protocol (IP) packets).

Consequently, Etherphone, which was not cited or discussed in the prosecution of the '704 patent, presents a SNQ of patentability because it, like Civanlar (which was antedated with a 37 C.F.R. § 1.131 declaration), discloses an address server (referred to as a "Voice Control Server" or "Telephone Control Server") for storing network protocol addresses usable by network nodes to establish point-to-point communications, as discussed in detail below. In view of the above, a reasonable examiner would consider Etherphone to be an important reference in deciding patentability. Accordingly, Etherphone presents a SNQ of patentability.

C. VocalChat User's Guide, VocalChat Readme File, VocalChat Networking Information, VocalChat Help File, VocalChat Troubleshooting Help File (collectively referred to as "VocalChat" or "the VocalChat references")

As mentioned above, the VocalChat system is described in VocalChat User's Guide, VocalChat Readme File, VocalChat Networking Information, VocalChat Help File, and VocalChat Troubleshooting Help File. As stated in the declaration of Alon Cohen, one of the co-founders of VocalTec, Ltd., included as **Exhibit L** with this reexamination:

> 1. VocalChat 1.01 Networking Information ("Networking Information"), attached as **Exhibit I** (referred to as "Exhibit A" in the declaration), was publicly distributed in 1994 as part of the VocalChat version 1.01 software, which was commercially released and on sale to the general public in 1994. The VocalChat version 1.01 software was sold as a boxed product, which included an electronic copy of the VocalChat 1.01 Networking Information document.

2. VocalChat 2.0 User's Guide ("User's Guide"), attached as **Exhibit G** (referred to as "Exhibit B" in the declaration), was publicly distributed in 1994 as part of the VocalChat version 2.0 software, which was commercially released and on sale to the general public in 1994. The VocalChat version 2.0 software was sold as a boxed product, which included a printed copy of the VocalChat 2.0 User's Guide.

3. The VocalChat Readme File ("Readme"), attached as **Exhibit H** (referred to as "Exhibit C" in the declaration), the VocalChat Troubleshooting Help File ("Troubleshooting Help File"), attached as **Exhibit K** (referred to as "Exhibit D" in the declaration), and VocalChat Information ("Help File"), attached as **Exhibit J** (referred to as "Exhibit E" in the declaration), are true and correct print outs of VocalChat version 2.02's README.TXT, TROUBLE.HLP, and INFO.HLP files, respectively. Electronic copies of these documents were publicly distributed in 1994 as part of the VocalChat version 2.02 software, which was commercially released and on sale to the general public as a boxed product in 1994.

VocalChat is a software-based telephone executed on a personal computer which connects to a central server to locate other personal computers on a variety of computer networks, including TCP/IP, NetBIOS, and IPX networks.[26] In particular, as illustrated in the figures on pages 4 and 5 of the VocalChat User's Guide (reproduced below), computers with VocalChat installed connect directly to a Post Office directory on a server to register their current network protocol addresses, query the Post Office directory for the network protocol addresses of other on-line computers, and establish point-to-point communications with each other using the retrieved network protocol addresses.[27]

[26] *See, e.g.*, VocalChat User's Guide, page 5 (illustrating a central server with a "post office" to enable communication between computers)

[27] *See, e.g.*, Readme File, page 2 ("VocalChat creates a central directory on the network, shared by all users called 'Post-Office.' All users must use the same Post-Office, otherwise they won't be able to communicate or leave messages to each other. This means that all users must be attached to one file-server which will be used for the Post-Office, and all have write permission for the Post-Office directory.").

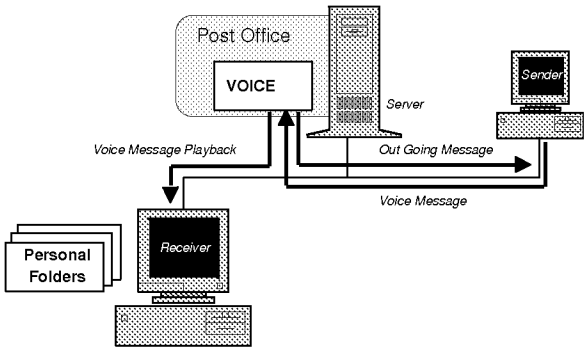


Figure from Page 5 of User's Guide

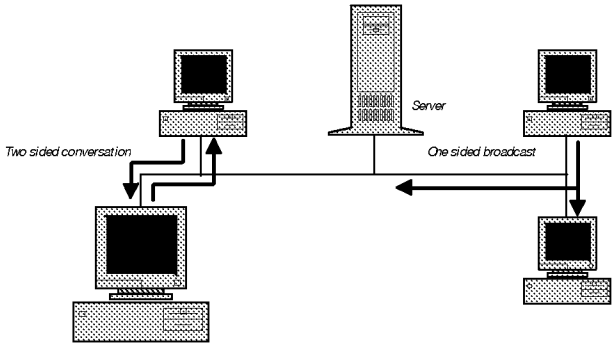


Figure from Page 4 of User's Guide

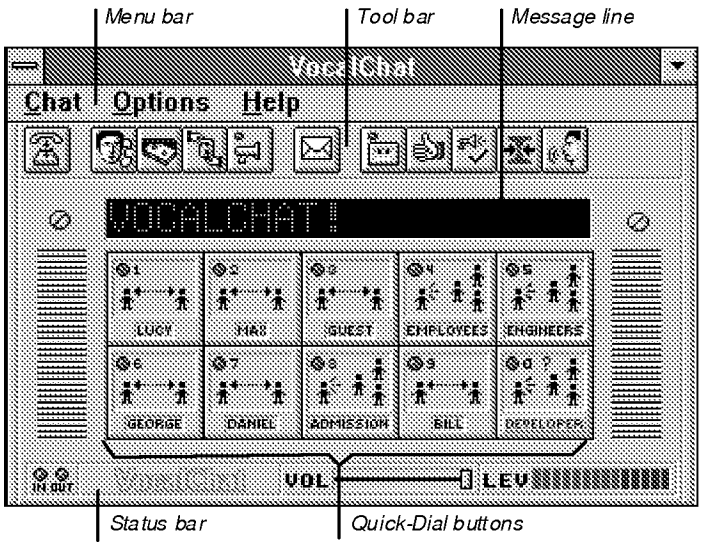
VocalChat may be implemented over a variety of network protocols including TCP/IP and NetBIOS. In a TCP/IP implementation, the Post Office directory includes a "Connection List" file (CONNLIST.VC) which contains the unique usernames and IP addresses of connected VocalChat users.[28] In the initial versions of VocalChat (versions 1.x), the Connection List file is called a "USERS file."[29] When the VocalChat client starts it transmits a user's unique username and IP address to the Connection List file.[30] User information maintained in the Connection List file is then made available to other VocalChat users, thereby enabling those users to locate and communicate with other VocalChat users.

---

[28] *See, e.g.*, Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory."). For the purpose of this reexamination we have converted the VocalChat Help File (Exhibit J) and Troubleshooting Help File (Exhibit K) into PDF files and added page numbers to simplify navigation. Aside from the addition of page numbering, the content of the help file and troubleshooting help file has not been modified in any manner.

[29] See, e.g., VocalChat Network Information, page 10 ("When the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address.")

[30] See, e.g., Help File, page 22 ("VocalChat will use the CONNLIST.VC files to get network addresses and a user name should be entered in the Setup for each user."). *See also* VocalChat Network Information, page 10.



As illustrated above, VocalChat also provides a graphical user interface (GUI) to establish point-to-point calls over the network. The user interface displays various buttons and interface elements, including a Call button, Quick Dial buttons, and an "Idle" icon, representing a temporarily disabled communication line, and a volume slider. Additionally, VocalChat includes a user interface window known as the "User List" to display a list of on-line users.[31] A VocalChat user may browse the list to find someone to call. Clicking on a username followed by the Call button establishes a point-to-point call with the selected user.[32] The VocalChat software queries the central directory database to determine whether the callee is on-line and, if so, the callee's network protocol address is returned to the VocalChat software (an IP address in the TCP/IP implementation).[33] The VocalChat software establishes a point-to-point call using the IP address.

Consequently, these VocalChat references, which were not cited or discussed in the prosecution of the '704 patent, present a SNQ of patentability because they, like Civanlar, show an address server (referred to as a "Post Office") for storing network protocol addresses usable by network nodes to establish point-to-point communications, as discussed in detail below. In view of the above a reasonable examiner would consider the VocalChat references to be

[31] *See, e.g.*, User's Guide, page 14 (illustrating an Address Book User List).
[32] *Id.*
[33] *See, e.g.*, Help File, page 22 (describing how VocalChat retrieves network addresses from the connection list file in a TCP/IP implementation).

important in deciding patentability. Accordingly, these VocalChat references present a SNQ of patentability.

D. RFC 1531

RFC 1531 discloses how TCP/IP addresses are assigned dynamically by a Dynamic Host Configuration Protocol (DHCP) server. *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). There are various benefits to using dynamic IP address assignment. For example, dynamically assigning IP addresses allows for "automatic reuse of an address that is no longer needed by the host to which it was assigned." RFC 1531, page 2 (Section 1, Introduction). Given that Claim 33 of the '704 patent explicitly requires "locating processes having dynamically assigned network protocol addresses" and that various other claims require the assignment of a network protocol address "following connection to the computer network" (see, e.g., Claim 1, 4, and 38), RFC 1531 would be considered important in deciding the question of patentability and accordingly presents a SNQ of patentability.

E. Vin

Vin is another prior art reference describing the Etherphone system which was published separately from Etherphone: Collected Papers 1987-1988 (May 1989) (collectively referred to herein as "Etherphone"). Vin describes many of the same features of the Etherphone system described in Etherphone and, in addition, describes how the Etherphone system may be used on a TCP/IP network. *See, e.g.*, Vin, page 77, Figure 5 (illustrating a "protocol stack and format" used in an Etherphone system which includes internet protocol (IP) packets). Consequently, Vin would be considered important in deciding the question of patentability and accordingly presents a SNQ of patentability, particularly with respect to Claim 32, which requires "Internet protocol" addresses.

F. Pinard

Pinard (Exhibit F) entitled "Human Machine Interface for Telephone Feature Invocation," issued on July 2, 1996 from an application filed on November 29, 1994. Pinard discloses that "[t]he ability to display icons on a computer display and to invoke commands by dragging an icon to another has long been known" in the prior art. (Col. 3, lines 15-17.) Pinard applies such a graphical user interface to the field of telephony. (Col. 1, lines 5-7.)

In Pinard "a method of providing information to a user unambiguously as to which persons are parties to a call" is described. (Col. 1, lines 55-57.) Specifically, Pinard shows how such information is represented graphically: "icons representing a subscriber's line associated with a local subscriber, the status of the line and [sic] associated with particular other subscribers to which calls are made or received are displayed in a manner that provides full information as to their status and the status of any call in progress, whether on line or being held, and whether it is a conference call or not." (Col. 2, lines 47-54.)

Also described is how call functions or processes can be displayed and then invoked using the graphical user interface: "The state of the call can be changed merely by dragging icons to particular locations on the display." (Col. 2, lines 54-55.) Specifically, Pinard describes:

> A method for calls to be made between parties, to be placed on hold, to be dropped from hold, to be conferenced or to be dropped from a conference with clear indication to the user which of the parties to any call are being dealt with. (Col. 1, lines 57-61.)

In Pinard with the graphical user interface on a personal computer, one has the ability to dial out and make and receive calls via a local area network (LAN). *See, e.g.*, Col. 1, line 64 - Col. 2, line 8; Col. 2, lines 38-41; Col. 3, lines 55-60; Col. 4; lines 1-3; and Figure 1. Pinard teaches that the described graphical user interface "can be used with any system in which a telephony application on a personal computer or [a] personal computer in conjunction with a server operates." (Col. 2, lines 41-45.) *See also* Col. 1, lines 60-62.

As show in Figure 2 below, Pinard discloses a personal computer with a display 11 running a telephone application software program. *See* Col. 4, lines 10-11. The program creates an icon 13 representing the caller ("Debbie" in the example) as well as an icon 15 representing a call setup process. (Col. 4, lines 11-18.)

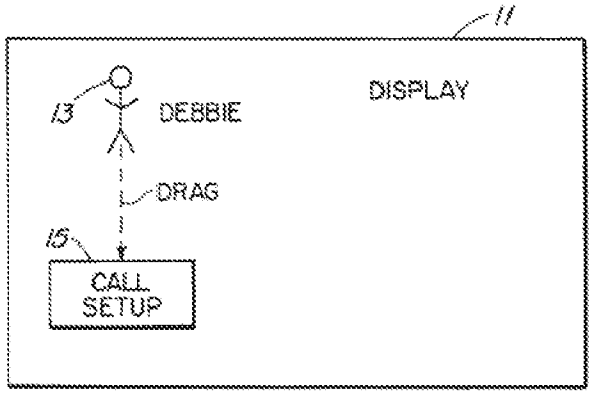


FIG. 2

Pinard discloses that the program uses the graphical user interface to permit the caller to place a call. (Col. 2, line 59 – Col. 3, line 9; Col. 3, lines10-14.) Figure 2 above and Figure 3 below are illustrative. By dragging the caller icon 13 ("Debbie") onto the call set up icon 15, the caller instructs the program that an outgoing call is to be made. (Col. 4, lines 19-21.) The program creates icons ("images of the faces of the persons listed in the directory") in directory 17 representing potential callees. *See, e.g.,* Col. 4, lines 22-31. When the user drags an icon from the directory 17 onto the call setup icon 15, the program retrieves and dials the corresponding callee's telephone number. *See, e.g.,* Col. 4, lines 38-48.

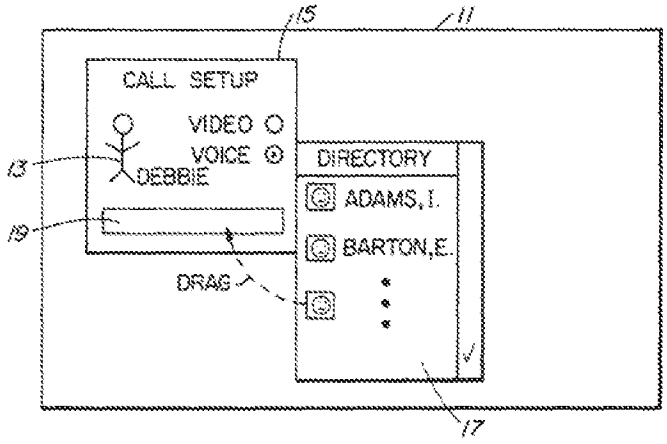


FIG. 3

As show in Figure 4 below, when caller "Debbie" and callee "John" are telephonically connected, the call setup icon 15 is transformed into a call icon 23, within whose borders the caller icon (stick figure labeled "Debbie") and the callee icon 21 ("John") are located. (Col. 4, lines 43-55.) This defined boundary signifies that a call is in progress. The program also creates

a new call setup icon 24. (Col. 4, lines 50-41.) The program allows the caller to terminate a call by dragging the callee icon 21 ("John") into a trash basket icon 26. *See, e.g.,* Col. 5, lines 1-4.

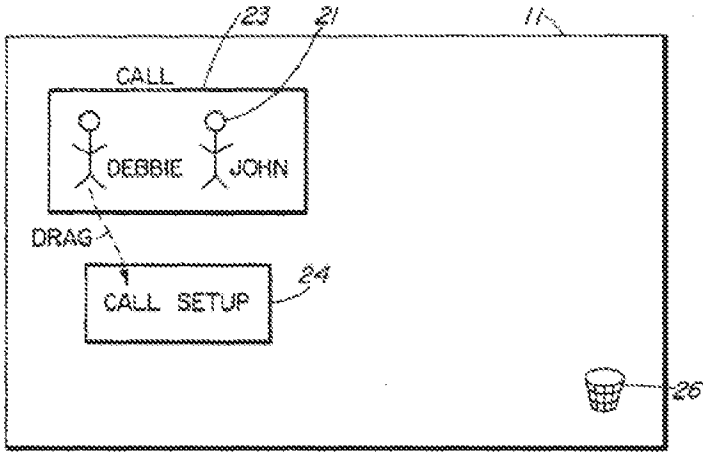


FIG. 4

The program of Pinard uses the graphical user interface to permit a caller to call a first callee on a first phone line and a second callee on a second phone line. *See, e.g.,* Col. 2, line 59 – Col. 3, line 9; Col. 3, lines 10-14. The program also uses a graphical interface to permit a caller to place on hold a first callee on one phone line and speak to a second callee on a second phone line (or vice versa). *See, e.g.*, Col. 2, line 59 – Col. 3, line 9; Col. 3, lines 10-14.

As shown in Figure 6 below, caller icon 13A ("Debbie") is ghosted in the first call icon 23 to indicate that the first phone line over which the first caller ("John," represented by icon 21) is connected is on hold. *See* Col. 5, lines 5-35; Col. 5, line 45 – Col. 6, line 5. Caller icon 13 ("Debbie") is solid in the second call icon 29 to indicate that the second phone line over which the second caller ("Mary," represented by icon 28) is connected is active. *See* Col. 5, lines 5-35; Col. 5, line 45 – Col. 6, line 5. Moreover, in the example of Figure 6, Debbie moves John from the first line (represented by call icon 23) to the second line (represented by call icon 29) by clicking and dragging John's icon 21, thereby creating a conference call between Debbie, Mary, and John. *See, e.g.*, Col. 5, lines 36-40.

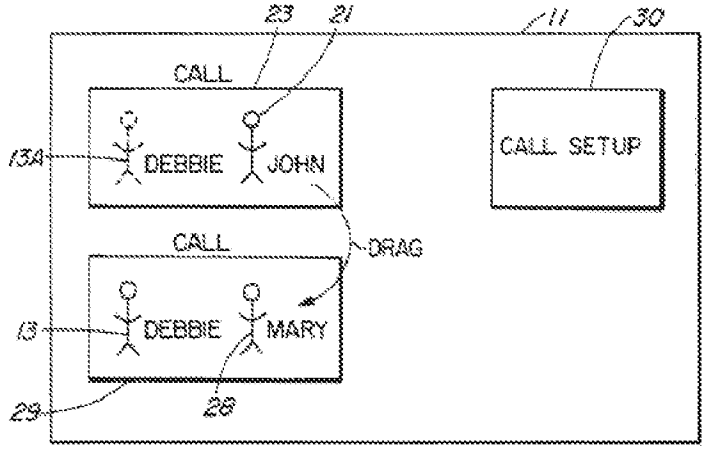


FIG. 6

A hard hold icon 39 of Pinard, shown below in Figure 12 allows the caller ("Debbie") to drag a callee icon 28 ("Mary") to the hard hold icon 39. This places the callee ("Mary") on hold. Other callers (represented by icons 41) may also be placed on hold. (Col. 6, lines 36-53.)

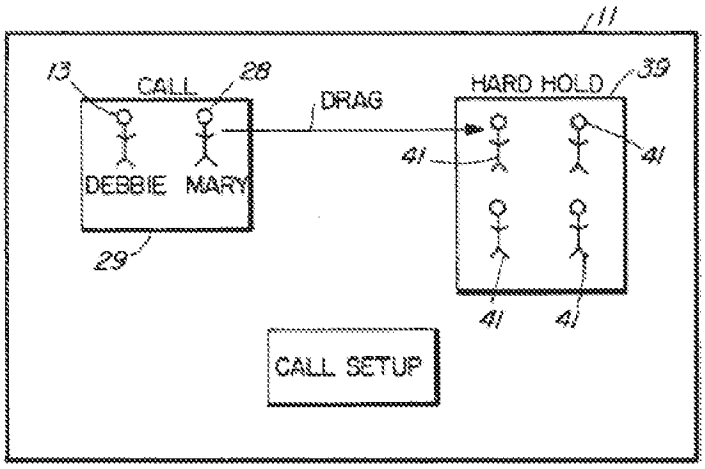


FIG. 12

Pinard discloses that the program uses the graphical user interface to permit a caller to conference a first callee and a second callee onto a single conference call. (Col. 2, line 59 – Col. 3, line 9; Col. 3, lines 10-14.) As shown in Figure 6 above, callee icon 21 ("John") is dragged onto call icon 29. As shown in Figure 7 below, this results in a conference call represented by conference icon 32 in whose borders caller icon 13 ("Debbie"), first callee icon 21 ("John"), and second callee icon 28 ("Mary") are located. (Col. 5, lines 36-44.)

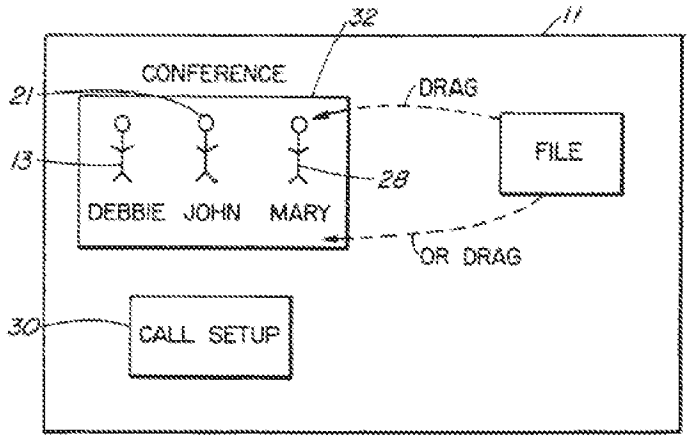


FIG. 7

Finally, Pinard teaches that "[u]sing similar principles, a person skilled in the art will now be able to provide unambiguous other features, such as call pickup, redial, speed call, callback, etc." (Col. 7, lines 48-52.) These and other features were well known to those of ordinary skill in the art insofar as they were prevalent on prior art telephones.

In summary, Pinard discloses graphical elements representing communication lines and callees that may be clicked and dragged to establish and terminate calls, set up conference calls, and place calls on hold, as recited in Claims 12-18 and 23-29 of the '704 patent. In view of the above, a reasonable examiner would consider Pinard to be an important reference in deciding patentability. Additionally, Pinard was not cited as a reference or discussed in the prosecution of the '704 Patent. Accordingly, Pinard presents a SNQ of patentability.

VI. DETAILED EXPLANATION OF THE PERTINENCY AND MANNER OF APPLYING THE PRIOR ART REFERENCES TO EVERY CLAIM FOR WHICH REEXAMINATION IS REQUESTED

As required under 37 C.F.R. § 1.510(b)(2), a detailed explanation of the pertinency and manner of applying the prior art references to the claims is provided. The following analysis is directed to prior art which was not cited during the prosecution of the claims of the '704 patent. Additional explanation of the pertinency and manner of applying the prior art references to the claims is provided in the claim charts at Exhibits M-O of this Request.

A. NetBIOS

1. Anticipation Rejections

¶ 1. The quotation of 35 U.S.C. §102 (b) forms the basis for the anticipation rejections which follow:

> A person shall be entitled to a patent unless…
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of the application for patent in the United States.

¶ 2. Claims 1-7 and 32-44 are anticipated by Protocols for X/Open PC Interworking SMB, Version 2, THE OPEN GROUP (1992) ("NetBIOS"), which includes Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Concept and Methods, RFC 1001 (March 1987) ("RFC 1001") and Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications, RFC 1002 (March 1987) ("RFC 1002").

¶ 3. During the Net2Phone Litigation, Net2Phone attempted to distinguish the claims of the '704 patent over NetBIOS. The court has yet to render an opinion on these arguments. As set forth in **Exhibit P** submitted with this reexamination, these arguments fail to distinguish the claims of the '704 patent over NetBIOS for a variety of reasons.

¶ 4. Neither NetBIOS nor its included RFC 1001/1002 were cited during the prosecution of the '704 patent. As delineated below there is a SNQ of patentability raised by NetBIOS. Below first the independent claims are set forth along with a discussion concerning the relevancy of NetBIOS to the SNQ of patentability. Then the dependent claims are set forth.

INDEPENDENT CLAIM 1

The preamble of Claim 1 reads, in pertinent part: "A computer program product for use with a computer system, the computer system executing a first process and operatively connectable to a second process and a server over a computer network . . ."

¶ 5. NetBIOS discloses a computer program product for use with a computer system which executes a "first process" and is operatively connectable to a "second process" and a server over a computer network. That NetBIOS discloses a computer program product for use with a computer system can be seen from 356 ("The NetBIOS service has become the dominant mechanism for personal computer networking. NetBIOS provides a vendor independent interface for the IBM Personal Computer (PC) and compatible systems."). In addition, NetBIOS describes that the computer systems (or "nodes") execute software, which is a computer-implemented "process." *See id.* ("NetBIOS defines a software interface . . . . NetBIOS has generally been confined to personal computers to date. However, . . . this specification has been designed to allow an implementation to be built on virtually any type of system where the TCP/IP protocol suite is available."); *id.* at 357 ("NetBIOS is the foundation of a large body of existing applications."). Finally, NetBIOS discloses a "server" to which all processes are operatively coupled over a network. For example, the figure on page 371 of NetBIOS illustrates a NetBIOS Name Server ("NBNS") coupled to point-to-point nodes ("P nodes") over the Internet.

¶ 6. In the pending litigation, Net2Phone argued that the term "server" should be defined broadly. Plaintiff Net2Phone, Inc.'s Response Brief on Claim Construction (Oct. 18, 2007) (Exhibit U), page 3. More specifically, Net2Phone argued:

> Consistent with the use of the term 'server' in the specification, the claims do not refer to any specific server configuration. They simply require a 'server' (also referred to as a 'connection server,' 'address server,' or 'server process'). There is nothing in any of the claims that require that the server be in the form of a single computer with a centralized database, as defendants contend.

Id., page 4. Similarly, Net2Phone argued that "[a] server in a 'client/server system' can be implemented in any number of ways, from one to multiple computers, in one location or many, and from a single large computer acting as the server to a network of personal computers." Plaintiff Net2Phone Inc.'s Reply Brief on Claim Construction (Oct. 19, 2007) (Exhibit W), page 7 (citing to the declaration of Professor Larry L. Peterson). Thus, under Net2Phone's

interpretation, a "server" is not limited to any particular hardware or software configuration.

¶ 7. It should be noted, however, that the requestor of the present Reexamination does not agree with this interpretation, and has stated as such in the pending litigation. *See, e.g.*, Reply Claim Construction Brief of Skype Technologies SA, Skype, Inc. and EBay Inc (Oct. 19, 2007) (Exhibit X), pages 2-9. For the sake of brevity, these interpretations are not repeated below with respect to the other claims of the '704 patent which require a "server." Under any interpretation, the NBNS described in NetBIOS is a "server."

Claim 1 requires "a computer usable medium having program code embodied in the medium."

¶ 8. NetBIOS applications are loaded into random access memory, which is a computer usable medium, and executed by a computer processor. NetBIOS describes that "[o]ne of the first implementations [of NetBIOS] was for personal computers running the PC-DOS and MS-DOS operating systems. It is possible to implement NetBIOS within other operating systems, or as processes which are, themselves, simply application programs as far as the host operating system is concerned." *id* at 359. NetBIOS further discloses that "typical use of NetBIOS is among independently-operated personal computers." *Id.* at 360.

Claim 1 further requires "program code for transmitting to the server a network protocol address received by the first process following connection to the computer network."

¶ 9. NetBIOS discloses program code executed on a node that transmits its name and IP address to the NBNS. For example, to engage in NetBIOS communications, a point-to-point ("P") or mixed ("M") node must register with a NBNS by transmitting a notice of the end node's name (a distinguishing identifier) and current IP address to the NBNS. *id.* 385 (illustrating the "P-NODE REGISTRATION PROCESS"). Specifically, a NetBIOS "Name Registration Request" sent by an M or P node to a NBNS includes the field "NB_ADDRESS," which is the "IP address of the name's owner." *id.* 431. *See also* NetBIOS at 367 (describing how the NBNS may act as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."); *id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name.");

id. at 461-464 (disclosing program code for the P-node name registration process) and *id.* 480-482 (disclosing program code for NBNS incoming packet processing for name registration). The NBNS thereby contains a list of names and corresponding IP addresses of point-to-point and mixed end-nodes.

Claim 1 also requires "program code for transmitting, to the server, a query as to whether the second process is connected to the computer network."

¶ 10. As disclosed in NetBIOS, an end-node sends a "query" to the NBNS to determine whether another end-node with the target name is currently logged onto the computer network, and hence is registered with the NBNS. "Name query (also known as 'resolution' or 'discovery') is the procedure by which the IP address(es) associated with a NetBIOS name are discovered." *id.* 377. NetBIOS point-to-point nodes "perform name resolution" by "ask[ing]" the NBNS for the IP address and other information of the target node with whom they wish to communicate. *Id. See also id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."). The NBNS "answers queries from a P node with a list of IP address and other information for" the target name. *Id.* at 389. *See also id.* at 440 (RFC 1002 describing "Name Query Request"); *id.* at 464-465 (describing "P-Node Find Name Procedure"). "Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder." *Id.* at 446.

Claim 1 further requires "program code for receiving a network protocol address of the second process from the server, when the second process is connected to the computer network."

¶ 11. NetBIOS states that the NBNS "answers queries from a P node with a list of IP address and other information for" the target name. NetBIOS at 389. The NetBIOS Name Server maintains entries only for "active" (on-line) nodes, i.e., nodes that have an on-line status with respect to the computer network. *Id.* at 446 (each entry "represents an active name"). Thus, if the NBNS has a record of the unique target name in its database, it returns a positive name query response to the requesting end-node. *Id.* at 390. The NBNS's positive name query response includes the IP address for the target node. *Id.* at 441 (describing a "POSITIVE NAME QUERY RESPONSE").

¶ 12. NetBIOS discloses a number of mechanisms to track the online status of nodes. Entries for off-line nodes are "removed" through the use of log-out messages and timers. For example, "NetBIOS names may be released explicitly or silently by an endnode. Silent release typically occurs when an end-node fails or is turned off." *id.* 377. For point-to-point nodes, the "explicit name release" involves "send[ing] a notification to their NBNS." *Id.* That is, upon going off-line, the node sends a "log-out" message to the NBNS, which then deletes the node's name/address entry from its database. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). NetBIOS also discloses mechanisms designed to detect "silent" releases, *i.e.*, when a nodes goes off-line without sending an explicit log-out message to the NBNS. *Id.* at 360 ("An explicit name deletion function is specified, so that applications may remove a name. Implicit name deletion occurs when a station ceases operation."). These mechanisms include a "Name Refresh" mechanism, whereby each point-to-point node "is responsible for sending periodic Name Request Requests" to the Name Server, which allows the Name Server "to detect if a P [] node has 'silently' gone down, so that names held by that node can be purged from the data base." *Id.* at 394; *see id.* at 378 ("The NBNS will consider a name to have been silently released if the end-node fails to send a name refresh message" prior to the expiration of a predetermined interval.). Nodes which do not send a refresh message to their NBNS within a determined period of time are deemed to have gone off-line and their name/address entry is deleted from the NBNS. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "name challenge" operation), 380 (describing "Node Status Request" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."). Thus, by design, only logged-in nodes are registered with the NBNS. *See, e.g., id.* at 446 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder."). In sum, the requesting node receives the target node's IP address from the NBNS only if the target node is currently logged in; otherwise, the NBNS responds to the requesting node's name query request with a negative response. *See, e.g., id.* at 389.

¶ 13. In Claim Construction Briefs filed in the pending litigation, the patentee argued that the term

> 'connected' means 'logged on,' and *vice versa* . . . To the extent defendants are trying to suggest that the claims require perfect information about who is on line at a given moment, that is simply incorrect. While Net2Phone's invention endeavors to identify accurately who is on line, it is not possible to achieve perfection. For example, it takes some time (albeit minimal) for the signal that a user has gone off-line to be communicated to the server, or a user's Internet connection may get interrupted before she can send an off-line message (and thus the server, for a time, assumes she is on-line, when in fact she is not). *See* Strickland Dep. at 140:7-141:7 (Ex. 21). Recognizing these issues, the patents explain that the server may use timestamps to update a person's status— *e.g.*, setting a default value of two hours, after which the server assumes that a party has gone off-line if it has not heard from her. *See* '704 patent, col. 5, ll. 39-44 (Ex. 2). In this respect, the patents explain, "the on-line status information stored in the database is ***relatively*** *current*." *Id.* at col. 5, ll. 42-43 (emphasis added). While Net2Phone believes that the claim language is clear, if the term "connected" (or "on-line") is going to be modified at all, it should be modified to say "*relatively* currently connected," because that is what the patents actually say.

Plaintiff Net2Phone Inc.'s Response Brief on Claim Construction (Oct. 18, 2007) (Exhibit U), pages 24-25. Thus, under Net2Phone's interpretation, the information retained in the "server" as to which processes are "connected to the computer network" or "online" may be imperfect. As described above, while the server "endeavors to identify accurately who is on line, it is not possible to achieve perfection." *Id.* NetBIOS employs similar techniques as NBNS entries for off-line nodes are removed through the use of log-out messages and timers.

¶ 14. Once again, the requestor of the present reexamination does not agree with this interpretation, and has stated as such in the pending litigation. *See, e.g.*, Reply Claim Construction Brief of Skype Technologies SA, Skype, Inc. and EBay Inc (Exhibit X), pages 12-14. For the sake of brevity, these interpretations are not repeated below with respect to the other claims of the '704 patent which require a process to be "connected to" the computer network or "on-line." Under any interpretation, a first NetBIOS process receives the network protocol address of a second NetBIOS process from the NBNS when the second NetBIOS process is "connected to the computer network."

Claim 1 also requires "program code, responsive to the network protocol address of the second process, for establishing a point-to-point communication link between the first process and the second process over the computer network."

¶ 15. Once the node seeking to initiate the communication has obtained from the NBNS the IP address for the node to receive the communication, a point-to-point communication is established between the nodes. "The NetBIOS session service begins after one or more IP addresses have been found for the target name . . . NetBIOS session service transactions, packets, and protocols are identical for all end-node types. They involve only directed (point-to-point) communications." NetBIOS at 397 (emphasis added). *See also id.* at 401:

> This first diagram shows the sequence of network events used to successfully establish a session without retargeting by the listener. The TCP connection is first established with the well-known NetBIOS session service TCP port, SSN_SRVC_TCP_PORT. The caller then sends a SESSION REQUEST packet over the TCP connection requesting a session with the listener. The SESSION REQUEST contains the caller's name and the listener's name. The listener responds with a POSITIVE SESSION RESPONSE informing the caller this TCP connection is accepted as the connection for the data transfer phase of the session.

See also id. at 398-400 ("16.1: Overview of NetBIOS Session Service"), 361 ("A session is a reliable message exchange, conducted between a pair of NetBIOS applications. Sessions are fullduplex, sequenced, and reliable. Data is organized into messages."). In sum, NetBIOS discloses all of the elements of, and hence anticipates, claim 1 of the '704 Patent.

INDEPENDENT CLAIM 2

Claim 2 claims "[a]n apparatus for enabling point-to-point communications between a first and a second process over a computer network, the apparatus comprising: a processor; a network interface, operatively coupled to the processor, for connecting the apparatus to the computer network."

¶ 16. NetBIOS describes a NBNS which is coupled to M and P nodes over the Internet (or other network). See, e.g., NetBIOS, page 371 (illustrating a NBNS) coupled to point-to-point nodes ("P nodes") over the Internet). The NBNS executes software using a processor and is inherently coupled to the Internet (or other network) via a network interface. See, e.g., id., page 359 ("[o]ne of the first implementations [of NetBIOS] was for personal computers running the PC-DOS and MS-DOS operating systems. It is possible to implement NetBIOS within other operating systems, or as processes which are, themselves, simply application programs as far as

the host operating system is concerned."); id. at 357 ("NetBIOS is the foundation of a large body of existing applications.").

Claim 2 requires "a memory, operatively coupled to the processor, for storing a network protocol address for selected of a plurality of processes, each network protocol address stored in the memory following connection of a respective process to the computer network."

¶ 17. As described above, to engage in NetBIOS communications, a point-to-point ("P") or mixed ("M") node must register with a NBNS by transmitting a notice of the end node's name (a distinguishing identifier) and current IP address to the NBNS. *See id.* at 385 (illustrating the "P-NODE REGISTRATION PROCESS"). This registration must inherently be stored in a "memory." A NetBIOS "Name Registration Request" sent by an M or P node to a NBNS includes the field "NB_ADDRESS," which is the "IP address of the name's owner." NetBIOS, page 431. *See also id.* at 367 (describing how the NBNS may act as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."); *id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."); *id.* at 461-464 (disclosing program code for the P-node name registration process) and 480-482 (disclosing program code for NBNS incoming packet processing for name registration). The NBNS thereby stores in a memory a list of names and corresponding IP addresses of point-to-point and mixed end-nodes. The P and M nodes execute software which is inherently a computer-implemented "process." *See, e.g., id.* at 356 ("NetBIOS defines a software interface . . . ."); *id.* at 357 ("NetBIOS is the foundation of a large body of existing applications.").

Claim 2 also requires "means, responsive to a query from the first process, for determining the on-line status of the second process and for transmitting a network protocol address of the second process to the first process in response to a positive determination of the on-line status of the second process."

¶ 18. In Claim Construction Briefs filed in the pending litigation, the patentee argued as follows with respect to the term "on-line status":

> To the extent defendants are trying to suggest that the claims require perfect information about who is on line at a given moment, that is simply incorrect. While Net2Phone's invention endeavors to identify accurately who is on line, it is

> not possible to achieve perfection. For example, it takes some time (albeit minimal) for the signal that a user has gone off-line to be communicated to the server, or a user's Internet connection may get interrupted before she can send an off-line message (and thus the server, for a time, assumes she is on-line, when in fact she is not). *See* Strickland Dep. at 140:7-141:7 (Ex. 21). Recognizing these issues, the patents explain that the server may use timestamps to update a person's status— *e.g.*, setting a default value of two hours, after which the server assumes that a party has gone off-line if it has not heard from her. *See* '704 patent, col. 5, ll. 39-44 (Ex. 2). In this respect, the patents explain, "the on-line status information stored in the database is ***relatively*** *current*." *Id.* at col. 5, ll. 42-43 (emphasis added). While Net2Phone believes that the claim language is clear, if the term "connected" (or "on-line") is going to be modified at all, it should be modified to say "*relatively* currently connected," because that is what the patents actually say.

Plaintiff Net2Phone Inc.'s Response Brief on Claim Construction (Oct. 18, 2007), pages 24-25. It should be noted, however, that the requestor of the present Reexamination does not agree with this interpretation, and has stated as such in the pending litigation. *See, e.g.*, Reply Claim Construction Brief of Skype Technologies SA, Skype, Inc. and EBay Inc, pages 12-14.

¶ 19. NetBIOS states that the NBNS "answers queries from a P node with a list of IP address and other information for" the target name. NetBIOS at 389. If the NBNS has a record of the unique target name in its database, it returns a positive name query response to the requesting end-node. *Id.* at 390. The NBNS's positive name query response includes the IP address for the target node. *Id.* at 441 (describing a "POSITIVE NAME QUERY RESPONSE"). NetBIOS discloses a number of mechanisms to track the online status of nodes. For example, "NetBIOS names may be released explicitly or silently by an endnode. Silent release typically occurs when an end-node fails or is turned off." *Id.* at 377. For point-to-point nodes, the "explicit name release" involves "send[ing] a notification to their NBNS." *Id.* That is, upon going off-line, the node sends a "log-out" message to the NBNS, which then deletes the node's name/address entry from its database. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). NetBIOS also discloses mechanisms designed to detect "silent" releases, *i.e.*, when a nodes goes off-line without sending an explicit log-out message to the NBNS. *Id.* at 360 ("An explicit name deletion function is specified, so that applications may remove a name. Implicit name deletion occurs when a station ceases operation."). These mechanisms include the refresh mechanism discussed above. Nodes which do not send a refresh message to their NBNS within a determined period of time are deemed to have gone off-line and their name/address

entry is deleted from the NBNS. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "name challenge" operation), 380 (describing "Node Status Request" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."). Thus, by design, only logged-in nodes are registered with the NBNS. *See id.* at 446 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder."). In sum, the requesting node receives the target node's IP address from the NBNS only if the target node is currently logged in; otherwise, the NBNS responds to the requesting node's name query request with a negative response. *See, e.g., id.* at 389.

INDEPENDENT CLAIM 4

Claim 4 claims "A method for enabling point-to-point communication between a first process and a second process over a computer network."

¶ 20. NetBIOS describes a method for enabling point-to-point communication between a first process (on a first node) and a second process (on a second node) over a computer network. As discussed above, the NBNS is used to resolve IP addresses of point-to-point end-nodes to facilitate point-to-point communications between such nodes. "The NetBIOS session service begins after one or more IP addresses have been found for the target name. . . . NetBIOS session service transactions, packets, and protocols are identical for all end-node types. They involve only directed (point-to-point) communications." *Id.* at 397 (emphasis added).

Claim 4 requires "receiving and storing into a computer memory a respective network protocol address for selected of a plurality of processes that have an on-line status with respect to the computer network, each of the network protocol addresses received following connection of the respective process to the computer network."

¶ 21. NetBIOS describes the NBNS receiving and storing the names and IP addresses of processes in its memory. For example, to engage in NetBIOS communications, a point-to-point ("P") or mixed ("M") node must register with a NBNS by transmitting a notice of the end node's name (a distinguishing identifier) and current IP address to the NBNS. *See* NetBIOS at page 385 (illustrating the "P-NODE REGISTRATION PROCESS"). Specifically, a NetBIOS "Name Registration Request" sent by an M or P node to a NBNS includes the field "NB_ADDRESS," which is the "IP address of the name's owner." *Id.* at 431. *See also id.* at 367

(describing how the NBNS may act as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."); *id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."); *id.* at 461-464 (disclosing program code for the P-node name registration process) and 480-482 (disclosing program code for NBNS incoming packet processing for name registration). The NBNS thereby contains a list of names and corresponding IP addresses of point-to-point and mixed end-nodes. In addition, NetBIOS discloses a number of mechanisms to track the online status of nodes. For example, "NetBIOS names may be released explicitly or silently by an endnode. Silent release typically occurs when an end-node fails or is turned off." *Id.* at 377. For point-to-point nodes, the "explicit name release" involves "send[ing] a notification to their NBNS." *Id.* That is, upon going off-line, the node sends a "log-out" message to the NBNS, which then deletes the node's name/address entry from its database. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). NetBIOS also discloses mechanisms designed to detect "silent" releases, *i.e.*, when a nodes goes off-line without sending an explicit log-out message to the NBNS. *Id.* at 360 ("An explicit name deletion function is specified, so that applications may remove a name. Implicit name deletion occurs when a station ceases operation."). These mechanisms include the refresh mechanism discussed above. Nodes which do not send a refresh message to their NBNS within a determined period of time are deemed to have gone off-line and their name/address entry is deleted from the NBNS. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "name challenge" operation), 380 (describing "Node Status Request" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."). Consequently, only logged-in nodes are registered with the NBNS. *See id.* at 446 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder."). In sum, the requesting node receives the target node's IP address from the NBNS only if the target node is currently logged in; otherwise, the NBNS responds to the requesting node's name query request with a negative response. *See, e.g., id.* at 389.

Claim 4 also requires "receiving a query from the first process to determine the on-line status of the second process."

¶ 22. As disclosed in NetBIOS, an end-node sends a "query" to the NBNS to determine whether another end-node with the target name is currently logged onto the computer network, and hence is registered with the NBNS. "Name query (also known as 'resolution' or 'discovery') is the procedure by which the IP address(es) associated with a NetBIOS name are discovered." *Id.* at 377. NetBIOS point-to-point nodes "perform name resolution" by "ask[ing]" the NBNS for the IP address and other information of the target node with whom they wish to communicate. *Id.* *See also id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."). The NBNS "answers queries from a P node with a list of IP address and other information for" the target name. *Id.* at 389. *See also id.* at 440 (RFC 1002 describing "Name Query Request"); *id.* at 464-465 (describing "P-Node Find Name Procedure"). "Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder." *Id.* at 446.

Claim 4 also requires "determining the on-line status of the second process."

¶ 23. The requesting node receives the target node's IP address from the NBNS only if the target node is currently logged in; otherwise, the NBNS responds to the requesting node's name query request with a negative response. For instance, when the NBNS receives a query for a target node's IP address, it performs a look search in its directory database for the target's current IP address. *Id.* at 389 (The NBNS "answers queries from a P node with a list of IP address and other information for" the target name.). The NBNS determines an end-node with the target name is currently registered in its database, and hence is deemed to be on-line. *See id.* at 376 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder."). As described above, the NBNS employs various mechanisms for determining the on-line status of nodes. For example, "NetBIOS names may be released explicitly or silently by an endnode. Silent release typically occurs when an end-node fails or is turned off." *Id.* at 377. For point-to-point nodes, the "explicit name release" involves "send[ing] a notification to their NBNS." *Id.* That is, upon going off-line, the node sends a "log-out" message to the NBNS, which then deletes the node's name/address entry from its database.

Finally, Claim 4 requires "transmitting an indication of the on-line status of the second process to the first process over the computer network."

¶ 24. If the end-node with the target name is currently registered in the NBNS database, the NBNS responds with a positive name query response. *See, e.g., id.* at 389 (The NBNS "answers queries from a P node with a list of IP address and other information for" the target name.); *id.* at 440 ("NAME QUERY REQUEST"), *id.* at 441 ("POSITIVE NAME QUERY RESPONSE"), *id.* at 464-465 ("P-Node Find Name Procedure"). A positive name query response includes the IP address for the target end-node, *id.* at 441, which is an indication that the target node has a positive on-line status. *See id.* at 446 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder."). A negative name query response from the NBNS may include a message that "[t]he name requested does not exist" in the NBNS database, which is an indication that the target node has an off-line status. *See, e.g., id.* at 442; *see also id.* at 484.

INDEPENDENT CLAIM 32

Claim 32 recites "[a] method of locating a process over a computer network comprising the steps of: a. maintaining an Internet accessible list having a plurality of selected entries, each entry comprising an identifier and a corresponding Internet protocol address of a process currently connected to the Internet, the Internet Protocol address added to the list following connection of the process to the computer network."

¶ 25. NetBIOS describes the NBNS maintaining an Internet accessible list having a plurality of selected entries, each entry comprising an identifier and a corresponding Internet protocol address of a process currently connected to the Internet. For example, to engage in NetBIOS communications, a point-to-point ("P") or mixed ("M") node must register with a NBNS by transmitting a notice of the end node's name (a distinguishing identifier) and current IP address to the NBNS. *See* NetBIOS, page 385 (illustrating the "P-NODE REGISTRATION PROCESS"). Specifically, a NetBIOS "Name Registration Request" sent by an M or P node to an NBNS includes the field "NB_ADDRESS," which is the "IP address of the name's owner." *Id.* at 431. *See also id.* at 367 (describing how the NBNS may act as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."); *id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."); NetBIOS at 461-464

(disclosing program code for the P-node name registration process) and 480-482 (disclosing program code for NBNS incoming packet processing for name registration). The NBNS thereby contains a list of names and corresponding IP addresses of point-to-point and mixed end-nodes. NetBIOS further discloses that this list may be accessible over the Internet. For example, the Figure on page 371 of NetBIOS clearly shows a NBNS and multiple client nodes connected over the "INTERNET." In addition, NetBIOS discloses a number of mechanisms to track the online status of nodes. For example, "NetBIOS names may be released explicitly or silently by an endnode. Silent release typically occurs when an end-node fails or is turned off." NetBIOS at 377. For point-to-point nodes, the "explicit name release" involves "send[ing] a notification to their NBNS." *Id.* That is, upon going off-line, the node sends a "log-out" message to the NBNS, which then deletes the node's name/address entry from its database. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). NetBIOS also discloses mechanisms designed to detect "silent" releases, *i.e.*, when a nodes goes off-line without sending an explicit log-out message to the NBNS. *Id.* at 360 ("An explicit name deletion function is specified, so that applications may remove a name. Implicit name deletion occurs when a station ceases operation."). These mechanisms include the refresh mechanism discussed above. Nodes which do not send a refresh message to their NBNS within a determined period of time are deemed to have gone off-line and their name/address entry is deleted from the NBNS. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "name challenge" operation), 380 (describing "Node Status Request" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."). Therefore, only logged-in nodes are registered with the NBNS. *See id.* at 446 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder."). In sum, the requesting node receives the target node's IP address from the NBNS only if the target node is currently logged in; otherwise, the NBNS responds to the requesting node's name query request with a negative response. *See, e.g.*, *id.* at 389.

Claim 32 also requires "in response to identification of one of the list entries by a requesting process, providing one of the identifier and the corresponding Internet protocol address of the identified entry to the requesting process."

¶ 26. A NBNS "answers queries from a P node with a list of IP address and other information for" the target name. NetBIOS at 389. If the NBNS has an entry in its database for the target name identified by the requesting node, the NBNS returns a positive name query response to the requesting node. *Id.* at 390. The NBNS's positive name query response includes the IP address for the target node. *Id.* at 389. *See also id.* at 441 ("POSITIVE NAME QUERY RESPONSE").

INDEPENDENT CLAIM 33

Claim 33 claims "[a] method for locating processes having dynamically assigned network protocol addresses over a computer network."

¶ 27. Because IP addresses were known to be dynamically assigned on TCP/IP networks such as the Internet, this feature is inherent in NetBIOS. *See, e.g.*, RFC 1531, Dynamic Host Configuration Protocol (1993), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). One of ordinary skill in the art would understand that the use and implementation of a NBNS enables locating point-to-point nodes that have dynamically assigned network addresses. Alternatively, it would have been obvious to combine NetBIOS with other references such as RFC 1531 which describe the use of dynamically assigned IP addresses.

Claim 33 also requires "maintaining, in a computer memory, a network accessible compilation of entries, selected of the entries comprising a network protocol address and a corresponding identifier of a process connected to the computer network, the network protocol address of the corresponding process assigned to the process upon connection to the computer network."

¶ 28. NetBIOS describes the NBNS maintaining an Internet accessible list having a plurality of selected entries, each entry comprising an identifier and a corresponding Internet protocol address of a process currently connected to the Internet. For example, to engage in NetBIOS communications, a point-to-point ("P") or mixed ("M") node must register with a NBNS by transmitting a notice of the end node's name (a distinguishing identifier) and current IP address to the NBNS. *See id.* at 385 (illustrating the "P-NODE REGISTRATION PROCESS"). Specifically, a NetBIOS "Name Registration Request" sent by an M or P node to a

NBNS includes the field "NB_ADDRESS," which is the "IP address of the name's owner." *Id.* at 431. *See also id.* at 367 (describing how the NBNS may act as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."); *id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."); NetBIOS at 461-464 (disclosing program code for the P-node name registration process) and 480-482 (disclosing program code for NBNS incoming packet processing for name registration). The NBNS thereby contains a list of names and corresponding IP addresses of point-to-point and mixed end-nodes. In addition, NetBIOS discloses a number of mechanisms to track the online status of nodes. For example, "NetBIOS names may be released explicitly or silently by an endnode. Silent release typically occurs when an end-node fails or is turned off." *Id.* at 377. For point-to-point nodes, the "explicit name release" involves "send[ing] a notification to their NBNS." *Id.* That is, upon going off-line, the node sends a "log-out" message to the NBNS, which then deletes the node's name/address entry from its database. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). NetBIOS also discloses mechanisms designed to detect "silent" releases, *i.e.*, when a nodes goes off-line without sending an explicit log-out message to the NBNS. *Id.* at 360 ("An explicit name deletion function is specified, so that applications may remove a name. Implicit name deletion occurs when a station ceases operation."). These mechanisms include the refresh mechanism discussed above. Nodes which do not send a refresh message to their NBNS within a determined period of time are deemed to have gone off-line and their name/address entry is deleted from the NBNS. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "name challenge" operation), 380 (describing "Node Status Request" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."). Thus, by design, only logged-in nodes are registered with the NBNS. *See id.* at 446 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder."). In sum, the requesting node receives the target node's IP address from the NBNS only if the target node is currently logged in; otherwise, the NBNS responds to the requesting node's name query request with a negative response. *See, e.g., id.* at 389.

Claim 33 also requires "in response to identification of one of the entries by a requesting process providing one of the identifier and the network protocol address to the requesting process."

¶ 29. A NBNS "answers queries from a P node with a list of IP address and other information for" the target name. *Id.* at 389. If the NBNS has an entry in its database for the target name identified by the requesting node, the NBNS returns a positive name query response to the requesting node. *Id.* at 390. The NBNS's positive name query response includes the IP address for the target node. *Id.* at 389. *See also id.* at 441 ("POSITIVE NAME QUERY RESPONSE").

INDEPENDENT CLAIM 38

Claim 38 claims "[a] computer program product for use with a computer system having a memory and being operatively connectable over a computer network to one or more computer processes, the computer program product comprising a computer usable medium having program code embodied in the medium."

¶ 30. NetBIOS discloses a computer program product for use with a computer system. *See id.* at 356 ("The NetBIOS service has become the dominant mechanism for personal computer networking. NetBIOS provides a vendor independent interface for the IBM Personal Computer (PC) and compatible systems."). In addition, NetBIOS describes that the computer systems (or "nodes") execute software, which is a computer-implemented "process." *See id.* ("NetBIOS defines a software interface . . . . NetBIOS has generally been confined to personal computers to date. However, . . . this specification has been designed to allow an implementation to be built on virtually any type of system where the TCP/IP protocol suite is available."); *id.* at 357 ("NetBIOS is the foundation of a large body of existing applications."). Finally, NetBIOS also discloses a "server" to which all processes are operatively coupled over a network. For example, the figure on page 371 of NetBIOS illustrates the NBNS coupled to point-to-point nodes ("P nodes") over the Internet. NetBIOS applications may be loaded into random access memory, which is a computer usable medium, and executed by a computer processor. NetBIOS describes that "[o]ne of the first implementations [of NetBIOS] was for personal computers running the PC-DOS and MS-DOS operating systems. It is possible to implement NetBIOS within other operating systems, or as processes which are, themselves, simply application programs as far as the host operating system is concerned." *Id.* at 359.

NetBIOS further discloses that "typical use of NetBIOS is among independently-operated personal computers." *Id.* at 360.

Claim 38 requires "program code configured to maintain, in the computer memory, a network accessible compilation of entries, selected of the entries comprising a network protocol address and a corresponding identifier of a process connected to the computer network, the network protocol address of the corresponding process assigned to the process upon connection to the computer network."

¶ 31. NetBIOS describes the NBNS maintaining a network accessible compilation of selected entries, each entry comprising an identifier and a corresponding Internet protocol address of a process currently connected to the Internet. For example, to engage in NetBIOS communications, a point-to-point ("P") or mixed ("M") node must register with a NBNS by transmitting a notice of the end node's name (a distinguishing identifier) and current IP address to the NBNS. *See id.* at 385 (illustrating the "P-NODE REGISTRATION PROCESS"). Specifically, a NetBIOS "Name Registration Request" sent by an M or P node to the NBNS includes the field "NB_ADDRESS," which is the "IP address of the name's owner." *Id.* at 431. *See also id.* at 367 (describing how the NBNS may act as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."); *id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."); *id.* at 461-464 (disclosing program code for the P-node name registration process) and 480-482 (disclosing program code for NBNS incoming packet processing for name registration). The NBNS thereby contains a list of names and corresponding IP addresses of point-to-point and mixed end-nodes. In addition, NetBIOS discloses a number of mechanisms to track the online status of nodes. For example, "NetBIOS names may be released explicitly or silently by an endnode. Silent release typically occurs when an end-node fails or is turned off." *Id.* at 377. For point-to-point nodes, the "explicit name release" involves "send[ing] a notification to their NBNS." *Id.* That is, upon going off-line, the node sends a "log-out" message to the NBNS, which then deletes the node's name/address entry from its database. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). NetBIOS also discloses mechanisms designed to detect "silent" releases, *i.e.*, when a nodes goes off-line without sending an explicit log-out message to the NBNS. *Id.* at 360 ("An explicit name

deletion function is specified, so that applications may remove a name. Implicit name deletion occurs when a station ceases operation."). These mechanisms include the refresh mechanism discussed above. Nodes which do not send a refresh message to their NBNS within a determined period of time are deemed to have gone off-line and their name/address entry is deleted from the NBNS. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "name challenge" operation), 380 (describing "Node Status Request" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."). Therefore, only logged-in nodes are registered with the NBNS. *See id.* at 446 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder."). In sum, the requesting node receives the target node's IP address from the NBNS only if the target node is currently logged in; otherwise, the NBNS responds to the requesting node's name query request with a negative response. *See, e.g., id.* at 389.

Claim 38 further requires "program code responsive to identification of one of the entries by a requesting process and configured to provide one of the identifier and the network protocol address to the requesting process."

¶ 32. The NBNS "answers queries from a P node with a list of IP address and other information for" the target name. *Id.* at 389. If the NBNS has an entry in its database for the target name identified by the requesting node, the NBNS returns a positive name query response to the requesting node. *Id.* at 390. The NBNS's positive name query response includes the IP address for the target node. *Id.* at 389. *See also id.* at 441 ("POSITIVE NAME QUERY RESPONSE").

INDEPENDENT CLAIM 43

Claim 43 claims "[a] computer program product for use with a computer system, the computer system executing a first process operatively coupled over a computer network to a second process and a server process, the computer program product comprising a computer usable medium having computer readable program code embodied therein."

¶ 33. NetBIOS discloses a computer program product for use with a computer system which executes a "first process" and is operatively connectable to a "second process" and a server over a computer network. First, NetBIOS discloses a computer program product for use

with a computer system. *See id.* at 356 ("The NetBIOS service has become the dominant mechanism for personal computer networking. NetBIOS provides a vendor independent interface for the IBM Personal Computer (PC) and compatible systems."). In addition, NetBIOS describes that the computer systems (or "nodes") execute software, which is a computer-implemented "process." *See id.* ("NetBIOS defines a software interface . . . . NetBIOS has generally been confined to personal computers to date. However, . . . this specification has been designed to allow an implementation to be built on virtually any type of system where the TCP/IP protocol suite is available."); *id.* at 357 ("NetBIOS is the foundation of a large body of existing applications."). Finally, NetBIOS also discloses a "server" to which all processes are operatively coupled over a network. For example, the figure on page 371 of NetBIOS illustrates the NBNS coupled to point-to-point nodes ("P nodes") over the Internet.

Claim 43 further requires "program code configured to access a directory database, the database having a network protocol address for a selected plurality of processes having on-line status with respect to the computer network, the network protocol address of each respective process forwarded to the database following connection to the computer network."

¶ 34. NetBIOS describes a "directory database" for storing network addresses of on-line processes. For example, to engage in NetBIOS communications, a point-to-point ("P") or mixed ("M") node must register with a NBNS by transmitting a notice of the end node's name (a distinguishing identifier) and current IP address to the NBNS. *See id.* at 385 (illustrating the "P-NODE REGISTRATION PROCESS"). Specifically, a NetBIOS "Name Registration Request" sent by an M or P node to a NBNS includes the field "NB_ADDRESS," which is the "IP address of the name's owner." NetBIOS *id.* at 431. *See also id.* at 367 (describing how the NBNS may act as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."); *id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."); *id.* at 461-464 (disclosing program code for the P-node name registration process) and 480-482 (disclosing program code for NBNS incoming packet processing for name registration). The NBNS thereby contains a directory database of names and corresponding IP addresses of point-to-point and mixed end-nodes.

Claim 43 also requires "program code responsive to one of the network protocol addresses and configured to establish a point-to-point communication link from the first process to the second process over the computer network."

¶ 35. Once the node seeking to initiate the communication has obtained from the NBNS the IP address for the node to receive the communication, a point-to-point communication is established between the nodes. "The NetBIOS session service begins after one or more IP addresses have been found for the target name . . . NetBIOS session service transactions, packets, and protocols are identical for all end-node types. They involve only directed (point-to-point) communications." *Id.* at 397 (emphasis added). *See also id.* at 401:

> This first diagram shows the sequence of network events used to successfully establish a session without retargeting by the listener. The TCP connection is first established with the well-known NetBIOS session service TCP port, SSN_SRVC_TCP_PORT. The caller then sends a SESSION REQUEST packet over the TCP connection requesting a session with the listener. The SESSION REQUEST contains the caller's name and the listener's name. The listener responds with a POSITIVE SESSION RESPONSE informing the caller this TCP connection is accepted as the connection for the data transfer phase of the session.

See also id. at 398-400 ("16.1: Overview of NetBIOS Session Service"), 361 ("A session is a reliable message exchange, conducted between a pair of NetBIOS applications. Sessions are fullduplex, sequenced, and reliable. Data is organized into messages."). In sum, NetBIOS discloses all of the elements of, and hence anticipates, claim 43 of the '704 Patent.

INDEPENDENT CLAIM 44

The preamble of Claim 44 reads: "In a first computer process operatively coupled over a computer network to a second process and an address server, a method of establishing a point-to-point communication between the first and second processes comprising the steps of."

¶ 36. NetBIOS describes that the various network "nodes" execute software, which is a computer-implemented "process." *See id.* ("NetBIOS defines a software interface . . . . NetBIOS has generally been confined to personal computers to date. However, . . . this specification has been designed to allow an implementation to be built on virtually any type of system where the TCP/IP protocol suite is available."); *id.* at 357 ("NetBIOS is the foundation of a large body of existing applications."). Finally, NetBIOS also discloses an "address server" to which all processes are operatively coupled over a network. For example, the figure on page 371 of

NetBIOS illustrates the NBNS coupled to point-to-point nodes ("P nodes") over the Internet. The NBNS is an "address server" because it stores names and IP addresses of nodes. *See, e.g., id.* at 367 describing how the NBNS may act as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."

Claim 44 also requires "following connection of the first process to the computer network forwarding to the address server a network protocol address at which the first process is connected to the computer network."

¶ 37. In NetBIOS, each node forwards its IP address to the NBNS following connection to the computer network. For example, to engage in NetBIOS communications, a point-to-point ("P") or mixed ("M") node must register with a NBNS by transmitting a notice of the end node's name (a distinguishing identifier) and current IP address to the NBNS. *See id.* at 385 (illustrating the "P-NODE REGISTRATION PROCESS"). More specifically, a NetBIOS "Name Registration Request" sent by an M or P node to a NBNS includes the field "NB_ADDRESS," which is the "IP address of the name's owner." *Id.* at 431. *See also id.* at 367 describing how the NBNS acts as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names.

Claim 44 also requires "querying the address server as to whether the second process is connected to the computer network."

¶ 38. As disclosed in NetBIOS, an end-node sends a "query" to the NBNS to determine whether another end-node with the target name is currently logged onto the computer network, and hence is registered with the NBNS. "Name query (also known as 'resolution' or 'discovery') is the procedure by which the IP address(es) associated with a NetBIOS name are discovered." *Id.* at 377. NetBIOS point-to-point nodes "perform name resolution" by "ask[ing]" the NBNS for the IP address and other information of the target node with whom they wish to communicate. *Id. See also id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."). The NBNS "answers queries from a P node with a list of IP address and other information for" the target name. *Id.* at 389. *See also id.* at 440 (RFC 1002 describing "Name Query Request"); *id.* at 464-465 (describing "P-Node Find Name Procedure"). "Each NODE_NAME entry represents an

active name in the same NetBIOS scope as the requesting name in the local name table of the responder." *Id.* at 446.

Claim 44 further requires "receiving a network protocol address of the second process from the address server, when the second process is connected to the computer network."

¶ 39. NetBIOS states that the NBNS "answers queries from a P node with a list of IP address and other information for" the target name. *Id.* at 389. If the NBNS has a record of the unique target name in its database, it returns a positive name query response to the requesting end-node. *Id.* at 390. The NBNS's positive name query response includes the IP address for the target node. *Id.* at 441 (describing a "POSITIVE NAME QUERY RESPONSE"). NetBIOS discloses a number of mechanisms to track the online status of nodes. For example, "NetBIOS names may be released explicitly or silently by an endnode. Silent release typically occurs when an end-node fails or is turned off." *Id.* at 377. For point-to-point nodes, the "explicit name release" involves "send[ing] a notification to their NBNS." *Id.* That is, upon going off-line, the node sends a "log-out" message to the NBNS, which then deletes the node's name/address entry from its database. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). NetBIOS also discloses mechanisms designed to detect "silent" releases, *i.e.*, when a nodes goes off-line without sending an explicit log-out message to the NBNS. *Id.* at 360 ("An explicit name deletion function is specified, so that applications may remove a name. Implicit name deletion occurs when a station ceases operation."). These mechanisms include the refresh mechanism discussed above. Nodes which do not send a refresh message to their NBNS within a determined period of time are deemed to have gone off-line and their name/address entry is deleted from the NBNS. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "name challenge" operation), 380 (describing "Node Status Request" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."). Consequently, only logged-in nodes are registered with the NBNS. *See id.* at 446 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder."). In sum, the requesting node receives the target node's IP address from the NBNS only if the

target node is currently logged in; otherwise, the NBNS responds to the requesting node's name query request with a negative response. *See, e.g.*, *id.* at 389.

Claim 44 further requires "in response to the network protocol address of the second process, establishing a point-to-point communication link with the second process over the computer network."

¶ 40. Once the node seeking to initiate the communication has obtained from the NBNS the IP address for the node to receive the communication, a point-to-point communication is established between the nodes. "The NetBIOS session service begins after one or more IP addresses have been found for the target name . . . NetBIOS session service transactions, packets, and protocols are identical for all end-node types. They involve only directed (point-to-point) communications." *Id.* at 397 (emphasis added). *See also id.* at 401:

> This first diagram shows the sequence of network events used to successfully establish a session without retargeting by the listener. The TCP connection is first established with the well-known NetBIOS session service TCP port, SSN_SRVC_TCP_PORT. The caller then sends a SESSION REQUEST packet over the TCP connection requesting a session with the listener. The SESSION REQUEST contains the caller's name and the listener's name. The listener responds with a POSITIVE SESSION RESPONSE informing the caller this TCP connection is accepted as the connection for the data transfer phase of the session.

See also id. at 398-400 ("16.1: Overview of NetBIOS Session Service"); 361 ("A session is a reliable message exchange, conducted between a pair of NetBIOS applications. Sessions are fullduplex, sequenced, and reliable. Data is organized into messages.").

DEPENDENT CLAIMS 3, 5-7, 14, 34-37, AND 39-42

Claim 3 of the '704 patent requires "a timer, operatively coupled to the processor, for time stamping the network protocol addresses stored in the memory."

¶ 41. The NBNS includes a timer for time-stamping name/IP address entries. For example, "[t]he NBNS may impose a 'time-to-live' on each name it registers. The registering node is made aware of this time value during the name registration procedure." *Id.* at 382. Similarly, as described in NetBIOS:

> If an end-node holds any names that have finite time-to-live values, then that node must periodically send a status report to the NBNS. Each name is reported using the NAME REFRESH REQUEST packet. These status reports restart the timers of both the NBNS and the reporting node. However, the only timers which are

restarted are those associated with the name found in the status report. Timers on other names are not affected. *Id.*

Claim 5 of the '704 patent requires "searching the computer memory for an entry relating the second process; and retrieving a network protocol address of the second process in response to a positive determination of the on-line status of the second process."

¶ 42. These features are inherent in the NetBIOS. In order for the NBNS to identify the IP address of a process in response to a query, it must inherently search its memory for an entry related to the process. For example, NetBIOS states that the NBNS "answers queries from a P node with a list of IP address and other information for" the target name. *Id.* at 389. If the NBNS has a record of the unique target name in its database, it returns a positive name query response to the requesting end-node. *Id.* at 390. The NBNS's positive name query response includes the IP address for the target node. *Id.* at 441 (describing a "POSITIVE NAME QUERY RESPONSE"). NetBIOS discloses a number of mechanisms to track the online status of nodes. For example, "NetBIOS names may be released explicitly or silently by an endnode. Silent release typically occurs when an end-node fails or is turned off." *Id.* at 377. For point-to-point nodes, the "explicit name release" involves "send[ing] a notification to their NBNS." *Id.* That is, upon going off-line, the node sends a "log-out" message to the NBNS, which then deletes the node's name/address entry from its database. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). NetBIOS also discloses mechanisms designed to detect "silent" releases, *i.e.*, when a nodes goes off-line without sending an explicit log-out message to the NBNS. *Id.* at 360 ("An explicit name deletion function is specified, so that applications may remove a name. Implicit name deletion occurs when a station ceases operation."). These mechanisms include the refresh mechanism discussed above. Nodes which do not send a refresh message to their NBNS within a determined period of time are deemed to have gone off-line and their name/address entry is deleted from the NBNS. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "name challenge" operation), 380 (describing "Node Status Request" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."). Therefore, only logged-in nodes are registered with the NBNS. *See id.* at 446 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name

table of the responder."). In sum, the requesting node receives the target node's IP address from the NBNS only if the target node is currently logged in; otherwise, the NBNS responds to the requesting node's name query request with a negative response. *See, e.g.*, *id.* at 389.

Claim 6 of the '704 patent requires "transmitting the network protocol address of the second process to the first process when the second process is determined in step C to have a positive on-line status with respect to the computer network."

¶ 43. NetBIOS states that the NBNS "answers queries from a P node with a list of IP address and other information for" the target name. *Id.* at 389. If the NBNS has a record of the unique target name in its database, it returns a positive name query response to the requesting end-node. *Id.* at 390. The NBNS's positive name query response includes the IP address for the target node. *Id.* at 441 (describing a "POSITIVE NAME QUERY RESPONSE"). NetBIOS discloses a number of mechanisms to track the online status of nodes. For example, "NetBIOS names may be released explicitly or silently by an endnode. Silent release typically occurs when an end-node fails or is turned off." *Id.* at 377. For point-to-point nodes, the "explicit name release" involves "send[ing] a notification to their NBNS." *Id.* That is, upon going off-line, the node sends a "log-out" message to the NBNS, which then deletes the node's name/address entry from its database. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). NetBIOS also discloses mechanisms designed to detect "silent" releases, *i.e.*, when a nodes goes off-line without sending an explicit log-out message to the NBNS. *Id.* at 360 ("An explicit name deletion function is specified, so that applications may remove a name. Implicit name deletion occurs when a station ceases operation."). These mechanisms include the refresh mechanism discussed above. Nodes which do not send a refresh message to their NBNS within a determined period of time are deemed to have gone off-line and their name/address entry is deleted from the NBNS. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "name challenge" operation), 380 (describing "Node Status Request" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."). For these reasons, only logged-in nodes are registered with the NBNS. *See id.* at 446 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder."). In sum, the requesting node receives the target node's IP address from the NBNS only if the

target node is currently logged in; otherwise, the NBNS responds to the requesting node's name query request with a negative response. *See, e.g.*, *id.* at 389.

Claim 7 requires "generating an off-line message when the second process is determined in step C to have a negative on-line status with respect to the computer network; and transmitting the off-line message to the first process."

¶ 44. The requesting node receives the target node's IP address from the NBNS only if the target node is currently logged in; otherwise, the NBNS responds to the requesting node's name query request with a negative response. *See, e.g.*, *id.* at 389 (illustrating a negative response "if the NBNS has no information about the name."). This negative response is an off-line message, which is generated when a node is determined to have a negative on-line status. As mentioned above, NetBIOS discloses a number of mechanisms to track the online status of nodes. For example, "NetBIOS names may be released explicitly or silently by an endnode. Silent release typically occurs when an end-node fails or is turned off." *Id.* at 377. For point-to-point nodes, the "explicit name release" involves "send[ing] a notification to their NBNS." *Id.* That is, upon going off-line, the node sends a "log-out" message to the NBNS, which then deletes the node's name/address entry from its database. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). NetBIOS also discloses mechanisms designed to detect "silent" releases, *i.e.*, when a nodes goes off-line without sending an explicit log-out message to the NBNS. *Id.* at 360 ("An explicit name deletion function is specified, so that applications may remove a name. Implicit name deletion occurs when a station ceases operation."). These mechanisms include the refresh mechanism discussed above. Nodes which do not send a refresh message to their NBNS within a determined period of time are deemed to have gone off-line and their name/address entry is deleted from the NBNS. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "name challenge" operation), 380 (describing "Node Status Request" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."). Therefore, only logged-in nodes are registered with the NBNS. *See id.* at 446 ("Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder.").

Claim 34 requires "modifying the compilation of entries."

¶ 45. NBNSs periodically modify their compilation of name/address entries in response to various conditions. For example, NetBIOS discloses an "explicit name release" mechanism. For point-to-point nodes, this involves "send[ing] a notification to their NBNS." NetBIOS at 377. Upon going off-line, the node sends a log-out message to the NBNS, which then deletes the node's name/address entry from the compilation, thereby modifying its compilation of entries. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). The compilation of entries may also be modified when an end node does not send a refresh message to its NBNS within a determined period of time, which may result in the deletion of its name/address entry from the NBNS's compilation of entries. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "NAME CHALLENGE" operation), 380 (describing "NODE STATUS REQUEST" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."); 384 (describing "OVERWRITE" operation).

Claim 35 requires "adding an entry to the compilation upon the occurrence of a predetermined event."

¶ 46. NBNS add entries to their compilation of name/address entries in response to various circumstances. For example, NetBIOS discloses that registrations adds the names (distinguishing identifiers) and current IP addresses of end-nodes to the NBNS' compilation of name/address entries. *Id.* at 385. *See also* id. at 431-432; id. at 367 (The NBNS acts as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."); *id.* at 461-464 (P-node name registration process) and 480-482 (NBNS incoming packet processing for name registration). Registration thereby adds names and corresponding IP addresses of end-nodes to the NBNS' compilation of name/address entries.

Claim 36 requires that "the predetermined event comprises notification by a user process of an assigned network protocol address."

¶ 47. User processes executed on P and M nodes "notify" the NBNS of their assigned names and IP addresses. For example, to engage in NetBIOS communications, a point-to-point ("P") or mixed ("M") node must register with a NBNS by transmitting a notice of the end node's

name (a distinguishing identifier) and current IP address to the NBNS. *See id.* at 385 (illustrating the "P-NODE REGISTRATION PROCESS"). More specifically, a NetBIOS "Name Registration Request" sent by an M or P node to a NBNS includes the field "NB_ADDRESS," which is the "IP address of the name's owner." *Id.* at 431. *See also id.* at 367 (describing how the NBNS acts as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."); *id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."); *id.* at 461-464 (disclosing program code for the P-node name registration process) and 480-482 (disclosing program code for NBNS incoming packet processing for name registration). The NBNS thereby contains a list of names and corresponding IP addresses of point-to-point and mixed end-nodes.

Claim 37 requires "deleting an entry from the compilation upon the occurrence of a predetermined event."

¶ 48. The NBNSs periodically delete a name/address entry from their compilation upon the occurrence of a predetermined event. For example, NetBIOS discloses an "explicit name release" mechanism. For point-to-point nodes, this involves "send[ing] a notification to their NBNS." *Id.* at 377. Upon going off-line, the node sends a log-out message to the NBNS. Upon the occurrence of this predetermined event, the NBNS deletes the node's name/address entry from the compilation of entries. NetBIOS also discloses that a name/address entry may be deleted if an end node does not send a "refresh" message to its NBNS within a predetermined period of time. *See id.* at 448, 452-453, 464-465. *See also id.* at 448 (describing "name challenge" operation), 450 (describing "Node Status Request" operation), 451 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 453 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS.").

Claim 39 requires "program code configured to modify the compilation of entries."

¶ 49. NBNSs periodically modify their compilation of name/address entries in response to various conditions. For example, NetBIOS discloses an "explicit name release" mechanism. For point-to-point nodes, this involves "send[ing] a notification to their NBNS." *Id.* at 377.

Upon going off-line, the node sends a log-out message to the NBNS, which then deletes the node's name/address entry from the compilation, thereby modifying its compilation of entries. *See also id.* at 393-394 (describing "NAME RELEASE TRANSACTIONS"). The compilation of entries may also be modified when an end node does not send a refresh message to its NBNS within a determined period of time, which may result in the deletion of its name/address entry from the NBNS's compilation of entries. *Id.* at 378, 382-383, 394-395. *See also id.* at 378 (describing "NAME CHALLENGE" operation), 380 (describing "NODE STATUS REQUEST" operation), 381 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 383 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS."); 384 (describing "OVERWRITE" operation).

Claim 40 requires "program code configured to add an entry to the compilation upon the occurrence of a predetermined event."

¶ 50. The NBNSs add entries to their compilation of name/address entries in response to various circumstances. For example, NetBIOS discloses that registration adds the names (distinguishing identifiers) and current IP addresses of end-nodes to the NBNS' compilation of name/address entries. *Id.* at 385. *See also* id. at 431-432; id. at 367 (NBNS acts as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."); *id.* at 461-464 (P-node name registration process) and 480-482 (NBNS incoming packet processing for name registration). Registration thereby adds names and corresponding IP addresses of end-nodes to the NBNS' compilation of name/address entries.

Claim 41 requires that the "predetermined event comprises notification by a process of an assigned network protocol address."

¶ 51. User processes executed on P and M nodes "notify" the NBNS of their assigned names and IP addresses. For example, to engage in NetBIOS communications, a point-to-point ("P") or mixed ("M") node must register with a NBNS by transmitting a notice of the end node's name (a distinguishing identifier) and current IP address to the NBNS. *See id.* at 385 (illustrating the "P-NODE REGISTRATION PROCESS"). More specifically, a NetBIOS "Name Registration Request" sent by an M or P node to a NBNS includes the field "NB_ADDRESS," which is the "IP address of the name's owner." *Id.* at 431. *See also id.* at 367 (describing how

the NBNS may act as a "'bulletin board' on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names."); *id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."); NetBIOS *id.* at 461-464 (disclosing program code for the P-node name registration process) and 480-482 (disclosing program code for NBNS incoming packet processing for name registration). The NBNS thereby contains a list of names and corresponding IP addresses of point-to-point and mixed end-nodes.

Claim 42 requires "program code configured to delete an entry from the compilation upon the occurrence of a predetermined event."

¶ 52. The NBNSs periodically delete a name/address entry from their compilation upon the occurrence of a predetermined event. For example, NetBIOS discloses an "explicit name release" mechanism. For point-to-point nodes, this involves "send[ing] a notification to their NBNS." *Id.* at 377. That is, upon going off-line, the node sends a log-out message to the NBNS. Upon the occurrence of this predetermined event, the NBNS deletes the node's name/address entry from the compilation of entries. NetBIOS also discloses that a name/address entry may be deleted if an end node does not send a "refresh" message to its NBNS within a predetermined period of time. *See id.* at 448, 452-453, 464-465. *See also id.* at 448 (describing "name challenge" operation), 450 (describing "Node Status Request" operation), 451 ("15.1.7 CONSISTENCY OF THE NBNS DATA BASE"), 453 ("A very cautious NBNS is free to poll nodes (by sending NAME QUERY REQUEST or NODE STATUS REQUEST packets) to verify that their name status is the same as that registered in the NBNS.").

2. Obviousness Rejections

¶ 53. The following is a quotation of 35 U.S.C. §103 (a) which forms the basis for the following obviousness rejections:

> A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains.

Patentability shall not be negatived by the manner in which the invention was made.

(i) NetBIOS in view of RFC 1531

¶ 54. Claim 33 explicitly states that the network protocol address of the client computer system is "dynamically assigned." *See* Claim 33 ("A method for locating processes having dynamically assigned network protocol addresses over a computer network"). Other independent claims state, more generally, that the network protocol address is assigned or transmitted to the database following the connection of the computer to the computer network. *See* Claim 1 ("transmitting to the server a network protocol address received by the first process following connection to the computer network"); Claim 2 ("each network protocol address stored in the memory following connection of a respective process to the computer network"); Claim 4 ("the network protocol addresses received following connection of the respective process to the computer network"); Claim 32 ("the Internet Protocol address added to the list following connection of the process to the computer network"); Claim 38 ("the network protocol address of the corresponding process assigned to the process upon connection to the computer network"); Claim 43 ("the network protocol address of each respective process forwarded to the database following connection to the computer network"); Claim 44 ("following connection of the first process to the computer network forwarding to the address server a network protocol address").

¶ 55. As described above, the NetBIOS reference inherently describes these features. For example, on many networks, including the TCP/IP networks described in NetBIOS, network addresses are assigned "following connection to the computer network." *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). Thus, in at least some instances, the computer systems on which NetBIOS was used received IP addresses following connection to the computer network. Consequently, dynamic address assignment is inherent in the NetBIOS reference.

¶ 56. Alternatively, Claims 1-7 and 32-44 should be rejected under 35 U.S.C. § 103(a) as being unpatentable over the NetBIOS reference in view of RFC 1531, which describes how TCP/IP addresses were dynamically assigned. *See, e.g.*, Dynamic Host Configuration Protocol,

RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks).

(ii) Motivation to Combine NetBIOS with RFC 1531

¶ 57. A motivation to combine the NetBIOS reference with RFC 1531 exists because the NetBIOS reference describes NetBIOS operating on a TCP/IP network and RFC 1531 describes a well known technique for dynamically assigning IP addresses within a TCP/IP network. One of ordinary skill in the art would have been motivated to combine the NetBIOS reference with RFC 1531 to realize the benefits associated with dynamic IP address assignment. For example, one of ordinary skill in the art would have been motivated to use dynamic IP address assignment because it eliminates the burdensome task of manually assigning IP addresses for all networked computers and allows for "automatic reuse of an address that is no longer needed by the host to which it was assigned." RFC 1531, page 2 (Section 1, Introduction). In fact, one of skill in the art would have understood at the time of the alleged invention of the '704 patent that personal computers connected to the Internet as described in RFC 1001/1002 of the NetBIOS reference would frequently have their IP addresses dynamically assigned.

(iii) NetBIOS in view of Pinard

INDEPENDENT CLAIM 10

The preamble to Claim 10 reads: "In a computer system, a method for establishing a point-to-point communication link from a caller process to a callee process over a computer network, the caller process having a user interface and being operatively connectable to the callee process and a server over the computer network . . ."

¶ 58. As discussed above, NetBIOS discloses a method of establishing a point-to-point communication link between nodes over a computer network such as a local network or the Internet. "NetBIOS session service transactions, packets, and protocols are identical for all end-node types. They involve only directed (point-to-point) communications." *Id.* at 397 (emphasis added). *See also id.* at 361 (describing how a call to a named callee process is used to "[i]nitiate a session with a process that is listening under the specified name. The calling entity must indicate both a calling name (properly registered to the caller) and a called name) (emphasis

added); *id.* at 359 ("NetBIOS applications employ NetBIOS mechanisms to locate resources, establish connections, send and receive data with an application peer, and terminate connections."); *id.* at 361 ("A session is a reliable message exchange, conducted between a pair of NetBIOS applications. Sessions are full-duplex, sequenced, and reliable. Data is organized into messages."). Applications which utilize NetBIOS application services inherently include "user interfaces." For example, "NetBIOS provides a vendor independent interface for the IBM Personal Computer (PC) and compatible systems." *Id.* at 356. The IBM PC included a text-based user interface known as PC-DOS. *See, e.g.*, id. (it is expected that on computers operating under the PC-DOS and MS-DOS operating systems that the existing NetBIOS interface will be preserved by implementers).

Claim 10 requires "providing a user interface element representing a first communication line."

¶ 59. Pinard discloses a user interface element representing a first communication line. For example, Figure 6 of Pinard illustrates a first call icon 23 which represents a first communication line and a second call icon 29 which represents a second communication line. In the example shown in Figure 6, the first call icon 23 represents a telephone call between "Debbie" and "John" and the second call icon 29 represents a telephone call between "Debbie" and "Mary." *See, e.g.*, Pinard, Col. 5, lines 23-30.

Claim 10 also requires "providing a user interface element representing a first callee process."

¶ 60. Pinard describes "a user interface element representing a first callee process." In the example shown in Figure 6 of Pinard, a first user interface element 21 is shown for the callee named "John" and a second user interface element is shown for the callee named "Mary." *See, e.g.*, Pinard, Col. 5, lines 23-30.

Claim 10 further requires "establishing a point-to-point communication link from the caller process to the first callee process, in response to a user associating the element representing the first callee process with the element representing the first communication line."

¶ 61. As described above, NetBIOS describes establishing a point-to-point communication link between nodes. *See, e.g.*, NetBIOS at 397 ("NetBIOS session service transactions, packets, and protocols are identical for all end-node types. They involve only

directed (point-to-point) communications.") (emphasis added). Pinard discloses that a point-to-point communication link is established in response to a user associating an element representing the first callee process with the element representing a first communication line. For example, Figure 3 of Pinard illustrates clicking and dragging an icon representing a callee from a directory 17 into a call setup icon 15. Once the callee answers the call, the call setup icon 15 becomes a call icon 23 as illustrated in Figure 4 of Pinard. *See, e.g.*, Pinard, Col. 4, lines 38-51 (describing how "[t]he user can then drag the icon or the name of the person to be called into the call setup icon . . . As soon as John answers the call, the application software program changes the call setup icon to a call icon designated as 23, and establishes a new call setup icon 24 spaced from the icon 23."). Similarly, Figure 6 illustrates how a point-to-point communication link may be established by clicking and dragging a callee icon 21 into an existing call icon 29. *See* Pinard, Col. 5, lines 36-37 ("Now to conference all parties, the user Debbie merely drags the John icon to the call icon 29.").

INDEPENDENT CLAIM 21

Claim 21 claims "[a] computer program product for use with a computer system comprising: a computer usable medium having program code embodied in the medium."

¶ 62. NetBIOS is implemented in software which is program code stored on a computer usable medium. *See, e.g.,* NetBIOS at 356 ("NetBIOS defines a software interface . . . . NetBIOS has generally been confined to personal computers to date. However, . . . this specification has been designed to allow an implementation to be built on virtually any type of system where the TCP/IP protocol suite is available.").

Claim 21 requires "establishing a point-to-point communication link from a caller process to a callee process over a computer network, the caller process having a user interface and being operatively connectable to the callee process and a server over the computer network."

¶ 63. As discussed above, NetBIOS discloses a method of establishing a point-to-point communication link between a caller process and a callee process over a computer network such as a local network or the Internet. "NetBIOS session service transactions, packets, and protocols are identical for all end-node types. They involve only directed (point-to-point) communications." NetBIOS at 397 (emphasis added). *See also id.* at 361 (describing how a call to a named callee process is used to "[i]nitiate a session with a process that is listening under the

specified name. The calling entity must indicate both a calling name (properly registered to the caller) and a called name) (emphasis added); *id.* at 359 ("NetBIOS applications employ NetBIOS mechanisms to locate resources, establish connections, send and receive data with an application peer, and terminate connections."); *id.* at 361 ("A session is a reliable message exchange, conducted between a pair of NetBIOS applications. Sessions are full-duplex, sequenced, and reliable. Data is organized into messages."). NetBIOS applications include "user interfaces," for example, "NetBIOS provides a vendor independent interface for the IBM Personal Computer (PC) and compatible systems." *Id.* at 356. The IBM PC included a text-based user interface known as PC-DOS. *See, e.g.*, id. (it is expected that on computers operating under the PC-DOS and MS-DOS operating systems that the existing NetBIOS interface will be preserved by implementers).

Claim 21 also requires "program code for generating an element representing a first communication line."

¶ 64. Pinard discloses program code for generating an element representing a first communication line. For example, Figure 6 of Pinard illustrates a first call icon 23 which represents a first communication line and a second call icon 29 which represents a second communication line. In the example shown in Figure 6, the first call icon 23 represents a telephone call between "Debbie" and "John" and the second call icon 29 represents a telephone call between "Debbie" and "Mary." *See, e.g.*, Pinard, Col. 5, lines 23-30.

Claim 21 also requires "program code for generating an element representing a first callee process."

¶ 65. Pinard describes program code for generating an element representing a first callee process. In the example shown in Figure 6 of Pinard, a first user interface element 21 is shown for the callee named "John" and a second user interface element is shown for the callee named "Mary." *See, e.g.*, Pinard, Col. 5, lines 23-30.

Claim 21 also requires "program code, responsive to a user associating the element representing the first callee process with the element representing the first communication line, for establishing a point-to-point communication link from the caller process to the first callee process."

¶ 66. As described above, NetBIOS describes establishing a point-to-point communication link between a caller process and a callee process. *See, e.g.*, NetBIOS at 397

("NetBIOS session service transactions, packets, and protocols are identical for all end-node types. They involve only directed (point-to-point) communications.") (emphasis added). Pinard discloses that a point-to-point communication link is established in response to a user associating an element representing the first callee process with the element representing a first communication line. For example, Figure 3 of Pinard illustrates clicking and dragging an icon representing a callee from a directory 17 into a call setup icon 15. Once the callee answers the call, the call setup icon 15 becomes a call icon 23 as illustrated in Figure 4 of Pinard. *See, e.g.*, Pinard, Col. 4, lines 38-51 (describing how "[t]he user can then drag the icon or the name of the person to be called into the call setup icon . . . As soon as John answers the call, the application software program changes the call setup icon to a call icon designated as 23, and establishes a new call setup icon 24 spaced from the icon 23."). Similarly, Figure 6 illustrates how a point-to-point communication link may be established by clicking and dragging a callee icon 21 into an existing call icon 29. *See* Pinard, Col. 5, lines 36-37 ("Now to conference all parties, the user Debbie merely drags the John icon to the call icon 29.").

DEPENDENT CLAIMS 11-20 AND 22-31

Claim 11 requires "querying the server as to the on-line status of the first callee process; and receiving a network protocol address of the first callee process over the computer network from the server."

¶ 67. As disclosed in NetBIOS, an end-node sends a "query" to the NBNS to determine whether another end-node with the target name is currently logged onto the computer network, and hence is registered with the NBNS. "Name query (also known as 'resolution' or 'discovery') is the procedure by which the IP address(es) associated with a NetBIOS name are discovered." *Id.* at 377. NetBIOS point-to-point nodes "perform name resolution" by "ask[ing]" the NBNS for the IP address and other information of the target node with whom they wish to communicate. *Id. See also id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."). The NBNS "answers queries from a P node with a list of IP address and other information for" the target name. *Id.* at 389. *See also id.* at 440 (RFC 1002 describing "Name Query Request"); *id.* at 464-465 (describing "P-Node Find Name Procedure"). "Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder." *Id.* at 446. If the end-node with the target name is currently registered in the NBNS

database, the NBNS responds with a positive name query response. *See, e.g., id.* at 389 (The NBNS "answers queries from a P node with a list of IP address and other information for" the target name.).

Claim 12 requires "providing an element representing a second communication line."

¶ 68. The graphical user interface described in Pinard provides an element representing a second communication line. For example, call icons 23 and 29 representing two communication lines are shown in Figure 6 of Pinard. *See* Pinard, Col. 5, lines 31-40, Figure 6 ("Now there are clearly two calls in progress . . .").

Claims 13 and 24 require "terminating the point-to-point communication link from the caller process to the first callee process, in response to the user disassociating the element representing the first callee process from the element representing the first communication line" and "program code, responsive to the user disassociating the element representing the first callee process from the element representing the first communication line, for terminating the point-to-point communication link from the caller process to the first callee process," respectively.

¶ 69. Figure 6 of Pinard illustrates how the call represented by call icon 23 is terminated by dragging the user icon for "John" 21 out of the call icon 23. Similarly, Figure 11 illustrates how a call is terminated by dragging the user icon to a "waste basket" icon 26.

Claim 13 and 24 further require "establishing a different point-to-point communication link from the caller process to the first callee process, in response to the user associating the element representing the first callee process with the element representing the second communication line" and "program code responsive to the user associating the element representing the first callee process with the element presenting the second communication line, for establishing a different point-to-point communication link from the caller process to the first callee process," respectively.

¶ 70. In Figure 6 of Pinard, the callee process icon for "John" 21 is dragged from call icon 23 to call icon 29, thereby terminating the call represented by call icon 23 and establishing a different link with the callee process represented by icon 21 (in this case a conference call with "John," "Mary," and "Debbie"). Moreover, once a callee is removed from a call by clicking and dragging the callee's icon, a new call can always be established with the callee by dragging the

callee's icon to a call setup icon. *See, e.g.*, Pinard, Figure 3 (showing a callee icon dragged from a directory to a call setup icon to establish a call). *See also* Pinard, Col. 4, lines 22-31.

Claims 14 and 25 require "providing a user interface element representing a second callee process; and . . . establishing a conference point-to-point communication link between the caller process and the first and second callee process, in response to the user associating the element representing the second callee process with the element representing the first communication line" and "program code for generating an element representing a second callee process; and program code means, responsive to the user associating the element representing the second callee process with the element representing the first communication line, for establishing a conference communication link between the caller process and the first and second callee process," respectively.

¶ 71. In Figure 6 of Pinard, the user interface element for "John" 21 is dragged from call icon 23 to call icon 29, thereby creating a conference call between "John," "Mary," and "Debbie." See Pinard, Col. 5, lines 31-44 ("Now to conference all parties, the user Debbie merely drags the John icon to the call icon 29.").

Claims 15 and 26 require "removing the second callee process from the conference point-to-point communication link in response to the user disassociating the element representing the second callee process from the element representing the first communication line" and "program code, responsive to the user disassociating the element representing the second callee process from the element representing the first communication line, for removing the second callee process from the conference communication link," respectively.

¶ 72. In Pinard, any callee process can be removed from a conference call by dragging the element representing the callee process from the conference call icon. For example, Figure 8 of Pinard shows the user icon "Debbie" removed from conference call represented by call icon 32, thereby "breaking Debbie's line from the conference." Pinard, Col. 6, lines 14-15.

Claims 16 and 27 require "providing a user interface element representing a communication line having a temporarily disabled status" and "program code for generating an element representing a communication line having a temporarily disabled status," respectively.

¶ 73. Examples of a "temporarily disabled status" provided in the '704 patent include "line on hold" and "line on mute." *See, e.g.*, '704 patent, Claims 17 and 18. Pinard describes a user interface element representing a communication line having a temporarily disabled status.

For example, Figure 12 illustrates a "hard hold" icon 39 to which user icons representing callers/callees 41 may be dragged to put the callers/callees on hold. *See, e.g.*, Pinard, Col. 6, lines 36-53 ("To place Mary on hard hold, Debbie drags Mary's icon 28 to the hard hold icon 39.").

Claims 16 and 27 also require "temporarily disabling a point-to-point communication link between the caller process and the first callee process, in response to the user associating the element representing the first callee process with the element representing the communication line having a temporarily disabled status" and "program code, responsive association of the element representing the first callee process with the element representing the communication line having a temporarily disabled status, for temporarily disabling the point-to-point communication link between the caller process and the first callee process," respectively.

¶ 74. In Pinard, in response to an icon of a caller/callee 41 being moved into the hard hold icon 39, the caller/callee is placed on hold. *See, e.g.*, Pinard, Col. 6, lines 36-53 ("To place Mary on hard hold, Debbie drags Mary's icon 28 to the hard hold icon 39.").

Claims 17 and 28 require that "the element provided in step D represents a communication line on hold status" and "the communication line having a temporarily disabled status comprises a communication line on hold status," respectively.

¶ 75. In Pinard, in response to an icon of a caller/callee 41 being moved into the hard hold icon 39, the caller/callee is placed on hold. *See, e.g.*, Pinard, Col. 6, lines 36-53 ("To place Mary on hard hold, Debbie drags Mary's icon 28 to the hard hold icon 39.").

Claim 19 requires "wherein the caller process further comprises a visual display and the user interface comprises a graphic user interface."

¶ 76. Pinard discloses a graphical user interface on a visual display which allows the caller to control the operation of the telephone. *See, e.g.*, Pinard, Figures 2-16 and Col. 6, lines 36-53 ("To place Mary on hard hold, Debbie drags Mary's icon 28 to the hard hold icon 39.").

Claim 20 requires "wherein the steps of establishing a point-to-point link as described in step C is performed in response to manipulation of the graphic elements on the graphic user interface."

¶ 77. As described above, Pinard discloses that a point-to-point communication link is established in response to a user associating a graphic element representing a callee process with a graphic element representing a communication line. For example, Figure 3 of Pinard illustrates

clicking and dragging an icon representing a callee from a directory 17 into a call setup icon 15. Once the callee answers the call, the call setup icon 15 becomes a call icon 23 as illustrated in Figure 4 of Pinard. *See, e.g.*, Pinard, Col. 4, lines 38-51 (describing how "[t]he user can then drag the icon or the name of the person to be called into the call setup icon . . . As soon as John answers the call, the application software program changes the call setup icon to a call icon designated as 23, and establishes a new call setup icon 24 spaced from the icon 23."). Similarly, Figure 6 illustrates how a point-to-point communication link may be established by clicking and dragging a callee icon 21 into an existing call icon 29. *See* Pinard, Col. 5, lines 36-37 ("Now to conference all parties, the user Debbie merely drags the John icon to the call icon 29.").

Claim 22 requires "program code for querying the server as to the on-line status of the first callee process; and program code for receiving a network protocol address of the first callee process over the computer network from the server."

¶ 78. As disclosed in NetBIOS, an end-node sends a "query" to the NBNS to determine whether another end-node with the target name is currently logged onto the computer network, and hence is registered with the NBNS. "Name query (also known as 'resolution' or 'discovery') is the procedure by which the IP address(es) associated with a NetBIOS name are discovered." NetBIOS at 377. NetBIOS point-to-point nodes "perform name resolution" by "ask[ing]" the NBNS for the IP address and other information of the target node with whom they wish to communicate. *Id.* *See also id.* at 388 ("Name query transactions are initiated by end-nodes to obtain the IP address(es) and other attributes associated with a NetBIOS name."). The NBNS "answers queries from a P node with a list of IP address and other information for" the target name. *Id.* at 389. *See also id.* at 440 (RFC 1002 describing "Name Query Request"); *id.* at 464-465 (describing "P-Node Find Name Procedure"). "Each NODE_NAME entry represents an active name in the same NetBIOS scope as the requesting name in the local name table of the responder." *Id.* at 446. If the end-node with the target name is currently registered in the NBNS database, the NBNS responds with a positive name query response. *See, e.g., id.* at 389 (The NBNS "answers queries from a P node with a list of IP address and other information for" the target name.).

Claim 23 requires "program code for generating an element representing a second communication line."

¶ 79. Pinard describes an element representing a second communication line. For example, Figure 6 of Pinard illustrates a first element (23) representing a first communication line and a second element (29) representing a second communication line.

Claim 30 requires that the "computer system further comprises a visual display and the user interface comprises a graphic user interface."

¶ 80. Figures 2-16 of Pinard illustrate a graphical user interface for managing telephone calls (which is inherently rendered on a "visual display").

Claim 31 requires that "the element representing the first communication line and the element representing the first callee process are graphic elements and wherein the program code for establishing a point-to-point communication link from the caller process to the first callee process further comprises: program code, responsive to manipulation of the graphic elements on the graphic user interface, for establishing the point-to-point communication link from the caller process to the first callee process."

¶ 81. As described above, Pinard discloses that a point-to-point communication link is established in response to a user associating a graphic element representing the first callee process with a graphic element representing a first communication line. For example, Figure 3 of Pinard illustrates clicking and dragging an icon representing a callee from a directory 17 into a call setup icon 15. Once the callee answers the call, the call setup icon 15 becomes a call icon 23 as illustrated in Figure 4 of Pinard. *See, e.g.*, Pinard, Col. 4, lines 38-51 (describing how "[t]he user can then drag the icon or the name of the person to be called into the call setup icon . . . As soon as John answers the call, the application software program changes the call setup icon to a call icon designated as 23, and establishes a new call setup icon 24 spaced from the icon 23."). Similarly, Figure 6 illustrates how a point-to-point communication link may be established by clicking and dragging a callee icon 21 into an existing call icon 29. *See* Pinard, Col. 5, lines 36-37 ("Now to conference all parties, the user Debbie merely drags the John icon to the call icon 29.").

(iv) Motivation to Combine NetBIOS and Pinard

¶ 82. A motivation to combine NetBIOS and Pinard exists considering the problem sought to be solved. The Pinard reference relates to the field of computer-implemented

telephony, and in particular to a computer-implemented method of indicating the status of various calls, to a user. *See* Pinard, Col. 1, lines 5-7. Pinard explicitly states that the invention "can be used with any system in which a . . . personal computer in conjunction with a server operates." Pinard, col. 2, lines 43-46. Given that NetBIOS describes networking software executed on personal computers (such as IBM PCs), and in particular describes a system in which a "personal computer" operates in conjunction with a "server" (i.e., a NBNS), one of ordinary skill in the art would have recognized that the particular design choices reflected in the graphical user interface of Pinard could readily be implemented within the context of the systems described in NetBIOS.

(v) NetBIOS in view of Pinard and Further in View of VocalChat User's Guide

¶ 83. Claim 18 and 29 require that "the element provided in step D represents a communication line on mute status" and "the communication line having a temporarily disabled status comprises a communication line on mute status," respectively. As described above, NetBIOS and Pinard describe all of the elements of Claim 18 and 29 except for a "communication line on mute status." VocalChat User's Guide describes a "communication line on mute status." As described in the User's Guide, "Manual Activation can also be used like the MUTE option in many phones: it lets you talk without being heard on the other user's system." User's Guide, page 57.

(vi) Motivation to Combine VocalChat User's Guide with NetBIOS and Pinard

¶ 84. A motivation to combine the VocalChat User's Guide with NetBIOS and Pinard exists considering the problem sought to be solved. All three references relate to the field of communications over a computer network, and the VocalChat User's Guide and Pinard relate to the use of a computer system to implement telephony features. *See, e.g.,* Pinard, Col. 1, lines 5-7. One of ordinary skill in the art would have recognized the need for a "mute" function to enable users to mute the audio of a call as needed.

B. Etherphone

1. Anticipation Rejections

¶ 85. The quotation of 35 U.S.C. §102 (b) forms the basis for the anticipation rejections which follow:

> A person shall be entitled to a patent unless…
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of the application for patent in the United States.

¶ 86. Claims 1-2, 4-7, 10-12, 14, 17-23, 25, and 30-44 are anticipated by *Etherphone: Collected Papers 1987-1988* (May 1989) (hereinafter "Etherphone") which published, **as a single publication**, with the following papers:

a. Polle T. Zellweger, et al., *An Overview of the Etherphone System and its Applications*, IEEE CONFERENCE ON COMPUTER WORKSTATIONS (March 1988), 160-168 (hereinafter "Zellweger 1").

b. Daniel C. Swinehart, *Telephone Management in the Etherphone System*, PROCEEDINGS OF THE IEEE/IEICE GLOBAL TELECOMMUNICATIONS CONFERENCE (November 1987), 1176-1180 (hereinafter "Swinehart 1").

c. Douglas B. Terry and Daniel C. Swinehart, *Managing Stored Voice in the Etherphone System*, ACM TRANSACTIONS ON COMPUTER SYSTEMS 6(1) (February 1988), 3-27 (hereinafter "Terry").

d. Daniel C. Swinehart, *System Support Requirements for Multi-media Workstations*, PROCEEDINGS OF THE SPEECHTECH '88 CONFERENCE (April 1988), 82-83 (hereinafter "Swinehart 2").

e. Polle T. Zellweger, *Active Paths through Multimedia Documents*, DOCUMENT MANIPULATION AND TYPOGRAPHY, J.C. AN VILET (ED.), CAMBRIDGE UNIVERSITY PRESS (1988) (hereinafter "Zellweger 2").

These papers were published together and form a single reference.

¶ 87. During the Net2Phone Litigation mentioned above, Net2Phone attempted to distinguish the claims of the '704 patent over Etherphone. The court has yet to render an opinion on these arguments. As set forth in **Exhibit Q**, these arguments fail to distinguish the claims of the '704 patent over Etherphone for a variety of reasons.

¶ 88. Etherphone was not cited during the prosecution of the '704 patent. As delineated below there is a SNQ of patentability raised by Etherphone. Below first the independent claims are set forth along with a discussion concerning the relevancy of Etherphone to the SNQ of patentability. Then the dependent claims are set forth.

INDEPENDENT CLAIM 1

The preamble of Claim 1 reads, in pertinent part: "A computer program product for use with a computer system, the computer system executing a first process and operatively connectable to a second process and a server over a computer network . . ."

¶ 89. Etherphone discloses a computer program product for use with a computer system which executes a "first process" and is operatively connectable to a "second process" and a server over a computer network. For example, the Etherphone system is:

> ...based on a hardware architecture that uses microprocessor-controlled telephones to transmit voice over an Ethernet that also supports a voice file server and a voice synthesis server, this system has been used for applications such as directory-based call placement, call logging, call filtering, and automatic call forwarding. (Zellweger 1, page 1.) *See also id.*, Figure 1 (illustrating Etherphones, computer workstations and servers communicating over an Ethernet network).

The system components shown in Figure 1 of Zellweger 1 provides communication "between two or more parties (Etherphones, servers, and so on)." *Id.*, page 3.

¶ 90. Claim 1 requires "a computer usable medium having program code embodied in the medium." The functionality of the Etherphone system is implemented in software, which is inherently stored on a computer usable medium. As described in Swinehart 2, the capabilities provided by the Etherphone system "are presented to application programmers as program packages and network services." Swinehart 2, page 1. *See also* Zellweger 1, page 2 ("Etherphone software is written in C"); *id.* ("Centralized server software limited the necessary size and speed of the Etherphone processor, and thus its cost . . ."); *id.*, page 1 (describing how the Etherphone system uses "microprocessor-controlled telephones to transmit voice over an Ethernet which also supports a voice file server and a voice synthesis server . . ."); Terry, page 4 ("The server software and the initial workstation software was developed in the Cedar programming environment.").

¶ 91. In the pending litigation, Net2Phone argued that the term "server" should be defined broadly. Plaintiff Net2Phone, Inc.'s Response Brief on Claim Construction (Oct. 18, 2007) (Exhibit U), page 3. More specifically, Net2Phone argued:

> Consistent with the use of the term 'server' in the specification, the claims do not refer to any specific server configuration. They simply require a 'server' (also referred to as a 'connection server,' 'address server,' or 'server process'). There is nothing in any of the claims that require that the server be in the form of a single computer with a centralized database, as defendants contend.

Id., page 4. Similarly, Net2Phone argued that "[a] server in a 'client/server system' can be implemented in any number of ways, from one to multiple computers, in one location or many, and from a single large computer acting as the server to a network of personal computers." Plaintiff Net2Phone Inc.'s Reply Brief on Claim Construction (Oct. 19, 2007) (Exhibit W), page 7 (citing to the declaration of Professor Larry L. Peterson). Thus, under Net2Phone's interpretation, a "server" is not limited to any particular hardware or software configuration.

¶ 92. It should be noted, however, that the requestor of the present Reexamination does not agree with this interpretation, and has stated as such in the pending litigation. *See, e.g.*, Reply Claim Construction Brief of Skype Technologies SA, Skype, Inc. and EBay Inc (Oct. 19, 2007) (Exhibit X), pages 2-9. For the sake of brevity, these interpretations are not repeated below with respect to the other claims of the '704 patent which require a "server." Under any interpretation, the Voice Control Server described in Etherphone is a "server."

Claim 1 further requires "program code for transmitting to the server a network protocol address received by the first process following connection to the computer network."

¶ 93. Etherphone processes notify the Voice Control Server (sometimes referred to as a "Telephone Control Server") of their network addresses in order to receive calls from other Etherphone processes. As described in Swinehart 1:

> The telephone control server controls voice conversations, implements the stand-alone behavior of telephone instruments and coordinates the activities of workstations and adjacent telephones in their implementation of the various voice capabilities. In addition, it stores personal preference information about each user that allows it to support advanced features such as ring motifs and subdued ringing without involving workstation programs. It uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations and the registration of visitors in the offices of their colleagues.

> Finally, and most importantly, the telephone control server provides a set of network protocols that workstations use to participate in the operation of the system.

Swinehart 1, page 4 (emphasis added). Consequently, when a user logs in to a workstation, the user's identity and the network address of the workstation are sent to the Voice Control Server to identify the user's current location. In fact, any time a computer transmits a data packet over an Ethernet network, the data packet must include the network address of the transmitting computer system (i.e., so that the receiving system knows the source of the data packet). *See id.* ("The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server."); Swinehart 1, page 2 ("Calls are to individuals, not locations . . . Logging in tells the telephone system where Karmen is."); Zellweger 1, page 5 ("An additional feature, called visiting, allows him to register his presence with a second workstation or Etherphone, such as during a meeting. Registering with the destination location allows users to travel more freely than forwarding calls from the home location does."). The network protocol address of the first process is "received" following connection to the computer network. For example, an Etherphone process (a running instance of an Etherphone application) will be assigned a network protocol address after the workstation or Etherphone on which it is running connects to the computer network – hence, following connection to the computer network. This is the case regardless of whether the workstation or Etherphone on which the given Etherphone process is running has a static network protocol address or, instead, a dynamically assigned network protocol address. The network protocol address is then transmitted to the Voice Control Server so that other Etherphone processes can locate the Etherphone process.

Claim 1 also requires "program code for transmitting, to the server, a query as to whether the second process is connected to the computer network."

¶ 94. As described in Zellweger 1, "conversations are established between two or more parties (Etherphones, servers, and so on) by performing remote procedure calls to the Voice Control Server." Zellweger 1, page 3. Moreover, as mentioned above, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server." Swinehart 1, page 4. Accordingly, when a first user at a first

Etherphone (a first "process") calls a second user at a second Etherphone (a second "process"), the first Etherphone transmits a query in the form of a remote procedure call to determine the location of the second Etherphone. *See also* Swinehart 1, page 2 ("Calls are to individuals, not locations . . . Logging in tells the telephone system where Karmen is."); Zellweger 1, page 5 ("An additional feature, called visiting, allows him to register his presence with a second workstation or Etherphone, such as during a meeting.").

Claim 1 further requires "program code for receiving a network protocol address of the second process from the server, when the second process is connected to the computer network."

¶ 95. As described in Swinehart 1: "The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4. Because of this, a first Etherphone or workstation attempting to connect to a second Etherphone or workstation receives the network address of the second Etherphone or workstation from the Voice Control Server.

¶ 96. In Claim Construction Briefs filed in the pending litigation, the patentee argued that the term

> 'connected' means 'logged on,' and *vice versa* . . . To the extent defendants are trying to suggest that the claims require perfect information about who is on line at a given moment, that is simply incorrect. While Net2Phone's invention endeavors to identify accurately who is on line, it is not possible to achieve perfection. For example, it takes some time (albeit minimal) for the signal that a user has gone off-line to be communicated to the server, or a user's Internet connection may get interrupted before she can send an off-line message (and thus the server, for a time, assumes she is on-line, when in fact she is not). *See* Strickland Dep. at 140:7-141:7 (Ex. 21). Recognizing these issues, the patents explain that the server may use timestamps to update a person's status— *e.g.*, setting a default value of two hours, after which the server assumes that a party has gone off-line if it has not heard from her. *See* '704 patent, col. 5, ll. 39-44 (Ex. 2). In this respect, the patents explain, "the on-line status information stored in the database is ***relatively*** *current*." *Id.* at col. 5, ll. 42-43 (emphasis added). While Net2Phone believes that the claim language is clear, if the term "connected" (or "on-line") is going to be modified at all, it should be modified to say "*relatively* currently connected," because that is what the patents actually say.

Plaintiff Net2Phone Inc.'s Response Brief on Claim Construction (Oct. 18, 2007) (Exhibit U), pages 24-25. Thus, under Net2Phone's interpretation, the information retained in the "server" as to which processes are "connected to the computer network" or "online" may be imperfect. As

described above, while the server "endeavors to identify accurately who is on line, it is not possible to achieve perfection." *Id.*

¶ 97. Once again, the requestor of the present reexamination does not agree with this interpretation, and has stated as such in the pending litigation. *See, e.g.*, Reply Claim Construction Brief of Skype Technologies SA, Skype, Inc. and EBay Inc (Exhibit X), pages 12-14. For the sake of brevity, these interpretations are not repeated below with respect to the other claims of the '704 patent which require a process to be "connected to" the computer network or "on-line." Under any interpretation, a first Etherphone process receives the network protocol address of a second Etherphone process from the Voice Control Server when the second Etherphone process is "connected to the computer network." Given that the Voice Control Server must track the location of individual on-line users, it is capable of determining "on-line" status with at least the same level of precision described in the '066 patent. *See, e.g.,* Swinehart 1, page 2 ("Calls are to individuals, not locations . . . Logging in tells the telephone system where Karmen is."); Zellweger 1, page 5 ("An additional feature, called visiting, allows him to register his presence with a second workstation or Etherphone, such as during a meeting.").

Claim 1 also requires "program code, responsive to the network protocol address of the second process, for establishing a point-to-point communication link between the first process and the second process over the computer network."

¶ 98. As described in Swinehart 1: "The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server." (emphasis added). For these reasons, after retrieving a network address of a callee device from the Voice Control Server, a workstation or Etherphone communicates directly over a point-to-point communication link with the callee device identified by the network address. *See also* Zellweger, page 2 ("Etherphones digitize, packetize, and encrypt telephone-quality voice (64 kilobits/second, with silence suppression) and send it to each other directly over an Ethernet . . ."); Swinehart 2, page 1 ("Etherphones digitize and encrypt telephone quality audio and transmit it in packet form directly over an Ethernet.").

INDEPENDENT CLAIM *2*

Claim 2 claims "[a]n apparatus for enabling point-to-point communications between a first and a second process over a computer network, the apparatus comprising: a processor; a network interface, operatively coupled to the processor, for connecting the apparatus to the computer network."

¶ 99. The Etherphone system includes a Voice Control Server which enables point-to-point communication between workstation and Etherphone processes. As described in Swinehart 1:

> The *telephone control server* controls voice conversations, implements the stand-alone behavior of telephone instruments, and coordinates the activities of workstations and adjacent telephones in their implementation of the various voice capabilities. In addition, it stores personal preference information about each user that allows it to support advanced features such as *ring motifs* and *subdued ringing* without involving workstation programs. It uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations, and the registration of *visitors* in the offices of their colleagues.

Swinehart 1, page 4 (emphasis in original). The Voice Control Server includes a processor and a network interface for connecting the Voice Control Server to the computer network. *See, e.g.*, Zellweger 1, Figure 1 (illustrating the Voice Control Server coupled to a 1.5 Mbit/sec Ethernet network).

Claim 2 requires "a memory, operatively coupled to the processor, for storing a network protocol address for selected of a plurality of processes, each network protocol address stored in the memory following connection of a respective process to the computer network."

¶ 100. Etherphone describes this limitation. In particular, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4. In order to send the network addresses of the other participants, the Voice Control Server must inherently store the network protocol addresses in a memory. The network addresses are stored in the memory following the connection of the processes to the computer network. For example, the Voice Control Server "uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations, and the registration of visitors in the offices of their colleagues." *Id.* Consequently, if a user logs in to any workstation, the identity of that user and the associated network address must be stored in a memory of the Voice Control Server so that other users can locate the user.

Claim 2 also requires "means, responsive to a query from the first process, for determining the on-line status of the second process and for transmitting a network protocol address of the second process to the first process in response to a positive determination of the on-line status of the second process."

¶ 101. As described in Zellweger 1, "conversations are established between two or more parties (Etherphones, servers, and so on) by performing remote procedure calls to the Voice Control Server." Moreover, as mentioned above, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server." Swinehart 1, page 4. Thus, when a first user at a first Etherphone (a first "process") calls a second user at a second Etherphone (a second "process"), the first Etherphone transmits a query in the form of a remote procedure call to determine the location of the second Etherphone. *See also* Swinehart 1, page 2 ("Calls are to individuals, not locations . . . Logging in tells the telephone system where Karmen is."); Zellweger 1, page 5 ("An additional feature, called visiting, allows him to register his presence with a second workstation or Etherphone, such as during a meeting."). Consequently, if a user is logged in to a particular Etherphone, the user's online status is "online" and associated with that Etherphone. The query will then return the current location of the user to the requesting process (executed on another Etherphone or workstation). Swinehart 1 describes various different types of "on-line status" for users including "visiting" a workstation or Etherphone and "offline." *See* Swinehart 1, page 2 (describing a "do-not-disturb option" in which "internal callers were given an on-screen explanation for being turned away, while outside callers were routed to an attendant").

INDEPENDENT CLAIM 4

Claim 4 claims "A method for enabling point-to-point communication between a first process and a second process over a computer network . . ."

¶ 102. Etherphone describes a method for enabling point-to-point communication between a first process and a second process over a computer network. For example, after receiving a network addresses of a first process, a second process establishes a point-to-point communication connection with the first process. *See, e.g.*, Swinehart 1, page 4 ("The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among

the participants, bypassing the control server.") (emphasis added). The "participants" all communicate with the system via software processes executed on computer workstations or Etherphones. *See, e.g.*, Swinehart 2, page 1 (describing how the capabilities of the Etherphone system "are presented to application programmers as program packages and network services."). *See also* Zellweger 1, page 2 ("Etherphone software is written in C"); *id.*, page 1 (describing how the Etherphone system uses "microprocessor-controlled telephones to transmit voice over an Ethernet . . ."); Terry, page 4 ("The server software and the initial workstation software was developed in the Cedar programming environment.").

Claim 4 requires "receiving and storing into a computer memory a respective network protocol address for selected of a plurality of processes that have an on-line status with respect to the computer network, each of the network protocol addresses received following connection of the respective process to the computer network."

¶ 103. As described above, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4. In order to send the network addresses of the other participants, the Voice Control Server must inherently receive and store the network protocol addresses in a computer memory. The network addresses are stored in the memory following the connection of the processes to the computer network. For example, the Voice Control Server "uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations, and the registration of visitors in the offices of their colleagues." *Id.* Thus, if a user logs in to any workstation, the identity of that user and the associated network address must be stored in a memory of the Voice Control Server so that other users can locate the user. When a user logs in to a workstation (e.g., as a "visitor"), the user is assigned an "on-line status."

Claim 4 also requires "receiving a query from the first process to determine the on-line status of the second process."

¶ 104. As described in Zellweger 1, "conversations are established between two or more parties (Etherphones, servers, and so on) by performing remote procedure calls to the Voice Control Server." Moreover, as mentioned above, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server." Swinehart 1, page 4. Thus, when a first user at a first Etherphone

(a first "process") calls a second user at a second Etherphone (a second "process"), the first Etherphone transmits a query in the form of a remote procedure call to determine the location of the second Etherphone. *See also* Swinehart 1, page 2 ("Calls are to individuals, not locations . . . Logging in tells the telephone system where Karmen is."); Zellweger 1, page 5 ("An additional feature, called visiting, allows him to register his presence with a second workstation or Etherphone, such as during a meeting."). Consequently, if a user is logged in to a particular Etherphone, the user's online status is "online" and associated with that Etherphone. The query will then return the current location of the user to the requesting process (executed on another Etherphone or workstation). Swinehart 1 describes different types of "on-line status" for users including "visiting" a workstation or Etherphone and "offline." *See* Swinehart 1, page 2 (describing how a user "turns to his workstation and registers Karmen as a visitor" and also describing a "do-not-disturb option" in which "internal callers were given an on-screen explanation for being turned away, while outside callers were routed to an attendant").

Claim 4 also requires "determining the on-line status of the second process."

¶ 105. See response to the previous claim element. As described above, if a user is logged in to a particular Etherphone, the user's online status is "online" and associated with that Etherphone. *See* Swinehart 1, page 2 ("Calls are to individuals, not locations . . . Logging in tells the telephone system where Karmen is."). Swinehart 1 describes different types of "on-line status" for users including "visiting" a workstation or Etherphone and "offline." *See* Swinehart 1, page 2 (describing how a user "turns to his workstation and registers Karmen as a visitor" and also describing a "do-not-disturb option" in which "internal callers were given an on-screen explanation for being turned away, while outside callers were routed to an attendant").

Finally, Claim 4 requires "transmitting an indication of the on-line status of the second process to the first process over the computer network."

¶ 106. The Voice Control Server will connect a first user to a second user if the second user is "online." *See, e.g.*, Swinehart 1, page 2 ("Calls are to individuals, not locations . . . Logging in tells the telephone system where Karmen is."). In addition, if the second user is offline or does not wish to receive calls, an indication is sent to the first user that the second user is unavailable. *See, e.g.*, Swinehart 1, page 2 (describing a "do-not-disturb option" in which

"internal callers were given an on-screen explanation for being turned away, while outside callers were routed to an attendant").

INDEPENDENT CLAIM 10

Claim 10 claims "a method for establishing a point-to-point communication link from a caller process to a callee process over a computer network . . ."

¶ 107. As described in Swinehart 1: "The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server." (emphasis added). Swinehart 1, page 2. Accordingly, after retrieving a network address of a callee device from the Voice Control Server, a workstation or Etherphone communicates directly over a point-to-point communication link with the callee device identified by the network address. *See also* Zellweger, page 2 ("Etherphones digitize, packetize, and encrypt telephone-quality voice (64 kilobits/second, with silence suppression) and send it to each other directly over an Ethernet . . ."); Swinehart 2, page 1 ("Etherphones digitize and encrypt telephone quality audio and transmit it in packet form directly over an Ethernet.").

Claim 10 further claims "the caller process having a user interface and being operatively connectable to the callee process and a server over the computer network . . ."

¶ 108. The workstations described in Etherphone include a graphical user interface (GUI). *See, e.g.*, Figures 1-10 of Swinehart 1 (illustrating various GUI features presented on the workstation display). *See also* Zellweger 1, Figures 3-4 (illustrating "telephone management windows" (Figure 3) and icons representing callers, callees and telephone lines (Figure 4)). The workstations may be Apple Macintoshes or Xerox 6085s. *See* Swinehart 1, page 1. The workstations are operatively connectable to the callee process and a server over the computer network. As previously described, "[t]he telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server." Swinehart 1, page 2.

Claim 10 requires "providing a user interface element representing a first communication line."

¶ 109. Etherphone discloses this limitation. For example, Figure 3 of Zellweger 1 depicts the Etherphone telephone management windows, including Phone and Answer buttons, a conversation log, and a portion of a personal telephone directory, which is a set of speed-dialing buttons. As described in Zellweger 1, "[a] variety of convenient workstation dialing methods are provided: a user can . . . select names or numbers from anywhere on the [Etherphone telephone management windows], use either of two directory tools that present browsable lists of names and associated telephone numbers as speed-dialing buttons, or redial any previously-made call by clicking on its conversation log entry. Calls can also be placed by name or number from the telephone keypad." Zellweger 1, page 4. In addition, Figure 4 of Zellweger 1 illustrates telephone icons representing telephone lines and icons with graphical images of a caller/callee which represent active telephone lines. As such, the Etherphone telephone management windows provide a "user interface element representing a first communication line."

Claim 10 also requires "providing a user interface element representing a first callee process."

¶ 110. Etherphone discloses user interface elements in the form of speed-dial buttons which represent frequently called callees. As described in Zellweger 1, the GUI provides "browsable lists of names and associated telephone numbers as speed-dialing buttons." Zellweger 1, page 4. *See also* Zellweger 1, Figure. 3 (depicting portion of a personal telephone directory, which is a set of speed-dial buttons). As another example, in Zellweger 1, Figure 4, the top left user interface icon represents a personal telephone directory in the form of a graphical rolodex.

Claim 10 further requires "establishing a point-to-point communication link from the caller process to the first callee process, in response to a user associating the element representing the first callee process with the element representing the first communication line."

¶ 111. First, Etherphone describes establishing a point-to-point communication link between a caller process and a callee process. *See, e.g.*, Swinehart 1, page 2 ("voice datagrams are transmitted directly among the participants, bypassing the control server"). Second, Etherphone discloses that the point-to-point communication link is established in response to a user associating an element representing the first callee process with the element representing a

first communication line. For example, the top row of Figure 4 of Zellweger 1 shows a series of graphical icons used for placing a call including a personal telephone directory, a telephone, and a picture of a user on the phone (to indicate a call is in process). In this example, the personal telephone directory, displayed as a graphical rolodex, includes a plurality of graphical elements representing callees (i.e., with a separate card in the rolodex for each callee). The icon of the telephone and the icon with the picture of a user talking on the phone represents a telephone communication line. As described in Zellweger 1, "[a]n active conversation is represented as a conversation between two people with a superimposed indication of the other party's name (also shown in Figure 4)." Zellweger 1, pages 4-5. Thus, when the user makes a call, the name from the graphical rolodex (PolleZ in the example) is "associated with" the graphical element representing the communication line (the image with the user talking on the phone).

¶ 112. Alternatively, as set forth below, Claim 10 is invalid under 35 U.S.C. § 103(a) as being unpatentable over Etherphone in view of Pinard.

INDEPENDENT CLAIM 21

Claim 21 claims "[a] computer program product for use with a computer system comprising: a computer usable medium having program code embodied in the medium."

¶ 113. The functionality of the Etherphone system is implemented in software, which is inherently stored on a computer usable medium. As described in Swinehart 2, the capabilities provided by the Etherphone system "are presented to application programmers as program packages and network services." Swinehart 2, page 1. *See also* Zellweger 1, page 2 ("Etherphone software is written in C"); *id.* ("Centralized server software limited the necessary size and speed of the Etherphone processor, and thus its cost . . ."); *id.*, page 1 (describing how the Etherphone system uses "microprocessor-controlled telephones to transmit voice over an Ethernet that also supports a voice file server and a voice synthesis server . . ."); Terry, page 4 ("The server software and the initial workstation software was developed in the Cedar programming environment.").

Claim 21 requires "establishing a point-to-point communication link from a caller process to a callee process over a computer network, the caller process having a user interface and being operatively connectable to the callee process and a server over the computer network."

¶ 114. First, Etherphone describes establishing a point-to-point communication link between a caller process and a callee process over a computer network. *See, e.g.*, Swinehart 1, page 2 ("voice datagrams are transmitted directly among the participants, bypassing the control server"). Second, Etherphone discloses that the caller process (i.e., the software executed on the caller's machine) includes a user interface. *See, e.g.*, Zellweger 1, page 4, Figures 3-4 ("a user can . . . select names or numbers from anywhere on the [Etherphone telephone management windows], use either of two directory tools that present browsable lists of names and associated telephone numbers as speed-dialing buttons, or redial any previously-made call by clicking on its conversation log entry. Calls can also be placed by name or number from the telephone keypad."). Finally, Etherphone describes "being operatively connectable to the callee process and a server over the computer network." *See, e.g.*, Swinehart 1, page 4 ("The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server.").

Claim 21 also requires "program code for generating an element representing a first communication line."

¶ 115. Etherphone describes a software-based graphical user interface with a graphical element representing a communication line. For example, Figure 4 of Zellweger 1 illustrates a graphical telephone icon that represents both outgoing calls (top row) and incoming calls (bottom row) over a telephone line, and graphical icons representing active calls (top and bottom right) established over the telephone line.

Claim 21 also requires "program code for generating an element representing a first callee process."

¶ 116. Etherphone describe a software-based graphical user interface with a graphical element representing a "callee process" (if the callee receives calls at an Etherphone or workstation). As described in Zellweger 1, the GUI provides "browsable lists of names and associated telephone numbers as speed-dialing buttons. Zellweger 1, page 4. *See also* Zellweger 1, Figure. 3 (depicting portion of a personal telephone directory, which is a set of speed-dial

buttons). As another example, in Zellweger 1, Figure 4, the top left user interface icon represents a personal telephone directory in the form of a graphical rolodex.

Claim 21 also requires "program code, responsive to a user associating the element representing the first callee process with the element representing the first communication line, for establishing a point-to-point communication link from the caller process to the first callee process."

¶ 117. First, Etherphone describes establishing a point-to-point communication link between a caller process and a callee process. *See, e.g.*, Swinehart 1, page 2 ("voice datagrams are transmitted directly among the participants, bypassing the control server"). Second, Etherphone discloses that the point-to-point communication link is established in response to a user associating an element representing the first callee process with the element representing a first communication line. For example, the top row of Figure 4 of Zellweger 1 shows a series of graphical icons used for placing a call including a personal telephone directory, a telephone, and a picture of a user on the phone (to indicate a call is in process). In this example, the personal telephone directory, displayed as a graphical rolodex, includes a plurality of graphical elements representing callees (i.e., with a separate card in the rolodex for each callee). The icon of the telephone and the icon with the picture of a user talking on the phone represents a telephone communication line. As described in Zellweger 1, "[a]n active conversation is represented as a conversation between two people with a superimposed indication of the other party's name (also shown in Figure 4)." Zellweger 1, pages 4-5. Thus, when the user makes a call, the name from the graphical rolodex (PolleZ in the example) is "associated with" the graphical element representing the communication line (the image with the user talking on the phone).

¶ 118. Alternatively, as set forth below, Claim 21 is invalid under 35 U.S.C. § 103(a) as being unpatentable over Etherphone in view of Pinard.

INDEPENDENT CLAIM 32

Claim 32 claims "[a] method of locating a process over a computer network comprising the steps of: a. maintaining an Internet accessible list having a plurality of selected entries, each entry comprising an identifier and a corresponding Internet protocol address of a process currently connected to the Internet, the Internet Protocol address added to the list following connection of the process to the computer network."

¶ 119. Etherphone describes these limitations. As described in Swinehart 1: "The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server." Swinehart 1, page 4. Therefore, the Telephone Control Server (also referred to as the Voice Control Server) stores a list of network addresses which are made available to workstations and Etherphones. In addition, the Voice Control Server associates different user identifiers with each network protocol address. For example, a user may log in to any workstation and, thereafter, calls to that user will be directed to that workstation and its associated Etherphone. As described in Swinehart 1:

> The *telephone control server* controls voice conversations, implements the stand-alone behavior of telephone instruments and coordinates the activities of workstations and adjacent telephones in their implementation of the various voice capabilities. In addition, it stores personal preference information about each user that allows it to support advanced features such as *ring motifs* and *subdued ringing* without involving workstation programs. It uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations and the registration of *visitors* in the offices of their colleagues.

Swinehart 1, page 4 (underline emphasis added). The network addresses may be Internet protocol addresses. For example, the Etherphone system was intended for use in "multiple networks and communication protocols." Terry, page 3. *See also* Terry, Abstract ("the voice manager stores voice on a special voice file server that is accessible via the local internet."). Moreover, another Etherphone reference, Vin, explicitly describes using the Internet Protocol (IP) within the Etherphone system. *See, e.g.*, Vin, page 77, Figure 5 (Exhibit D of this Request) (illustrating a "protocol stack and format" which includes internet protocol (IP) packets). Vin may be combined with Etherphone under 35 U.S.C. § 102. *See* MPEP 2131.01 (stating that a §102 rejection over multiple references is proper when the extra references are cited to explain the meaning of a term used in the primary reference). In this case, Vin is used to define the

complete meaning of the term "Voice Transmission Protocol" used in Etherphone. In any case, it would have been obvious to combine Vin with Etherphone because they both describe the same Etherphone system. See obviousness section below.

Claim 32 also requires "in response to identification of one of the list entries by a requesting process, providing one of the identifier and the corresponding Internet protocol address of the identified entry to the requesting process."

¶ 120. As mentioned above, when a first user attempts to call a second user from a workstation and Etherphone, the Voice Control Server provides the current network protocol address of the second user to the requesting process executed on the workstation/ Etherphone of the first user. Using the network address, the requesting process then initiates a communication session with the workstation and Etherphone of the second user. *See, e.g.*, Swinehart 1, page 4 ("The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server.").

INDEPENDENT CLAIM 33

Claim 33 claims "[a] method for locating processes having dynamically assigned network protocol addresses over a computer network."

¶ 121. As discussed above, Etherphone discloses a method of locating network protocol addresses over a computer network. *See, e.g.*, Swinehart 1, page 4 ("The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants."). The network protocol addresses may be Internet protocol addresses. For example, the Etherphone system was intended for use in "multiple networks and communication protocols." Terry, page 3. *See also* Terry, Abstract ("the voice manager stores voice on a special voice file server that is accessible via the local internet."). Moreover, another Etherphone reference, Vin, explicitly describes using the Internet Protocol (IP) within the Etherphone system. *See, e.g.*, Harrick M. Vin, et al., *Multimedia Conferencing in the Etherphone Environment*, IEEE COMPUTER SOCIETY (Oct. 1991), page 77, Figure 5 (Exhibit D of this request) (illustrating a "protocol stack and format" which includes internet protocol (IP) packets). Vin may be combined with Etherphone under 35 U.S.C. § 102. *See* MPEP 2131.01 (stating that a §102 rejection over multiple references is proper when the extra references are cited to explain the meaning of a term used in the primary reference). In this case, Vin is used to

define the complete meaning of the term "Voice Transmission Protocol" used in Etherphone. In any case, it would have been obvious to combine Vin with Etherphone because they both describe the same Etherphone system. Because dynamically assigning IP addresses was known, this feature is inherent in Etherphone or at least obvious in light of the numerous references describing dynamic IP address assignment. *See, e.g.*, RFC 1531, Dynamic Host Configuration Protocol (1993), Section 2.2 (describing the "dynamic allocation of network addresses").

Claim 33 also requires "maintaining, in a computer memory, a network accessible compilation of entries, selected of the entries comprising a network protocol address and a corresponding identifier of a process connected to the computer network, the network protocol address of the corresponding process assigned to the process upon connection to the computer network."

¶ 122. The Voice Control Server maintains "a compilation of entries . . . comprising a network protocol address and a corresponding identifier." For example, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4. In order to send the network addresses of the other participants, the Voice Control Server must inherently store the network protocol addresses in a "computer memory." The network addresses are stored in the memory with corresponding user identifiers. For example, the Voice Control Server "uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations, and the registration of visitors in the offices of their colleagues." *Id.* Thus, if a user logs in to any workstation, the identity of that user and the associated network address must be stored in a memory of the Voice Control Server so that other users can locate the user. Since the user is connecting via a workstation and Etherphone, the user identifier identifies the current software process through which the user is interacting with the system.

Claim 33 also requires "in response to identification of one of the entries by a requesting process providing one of the identifier and the network protocol address to the requesting process."

¶ 123. As described above, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4.

INDEPENDENT CLAIM 38

Claim 38 claims "[a] computer program product for use with a computer system having a memory and being operatively connectable over a computer network to one or more computer processes, the computer program product comprising a computer usable medium having program code embodied in the medium."

¶ 124. The Etherphone system is implemented with software executed on a plurality of computing devices, including servers, workstations, and Etherphones. As described in Zellweger 1, the Etherphone system uses "microprocessor-controlled telephones to transmit voice over an Ethernet that also supports a voice file server and a voice synthesis server . . ." Zellweger 1, page 1. *See also* Swinehart 2, page 1 (describing how the capabilities provided by the Etherphone system "are presented to application programmers as program packages and network services."); Zellweger 1, page 2 ("Etherphone software is written in C"); *id.* ("Centralized server software limited the necessary size and speed of the Etherphone processor, and thus its cost . . ."); Terry, page 4 ("The server software and the initial workstation software was developed in the Cedar programming environment."). The computer systems are operatively connectable over a computer network to computer processes. As described in Zellweger 1, "conversations are established between two or more parties (Etherphones, servers, and so on) by performing remote procedure calls to the Voice Control Server." Zellweger 1, page 3. Remote procedure calls are inherently directed to "computer processes."

Claim 38 requires "program code configured to maintain, in the computer memory, a network accessible compilation of entries, selected of the entries comprising a network protocol address and a corresponding identifier of a process connected to the computer network, the network protocol address of the corresponding process assigned to the process upon connection to the computer network."

¶ 125. The Voice Control Server maintains "a compilation of entries . . . comprising a network protocol address and a corresponding identifier or a process connected to the computer network." For example, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4. In order to send the network addresses of the other participants, the Voice Control Server must inherently store the network protocol addresses in a "computer memory." The network addresses are stored in the memory with corresponding user identifiers. For example, the Voice Control Server "uses dynamic information linking users to workstations in order to provide calls

to individuals rather than fixed locations, and the registration of visitors in the offices of their colleagues." *Id.* Thus, if a user logs in to any workstation, the identity of that user and the associated network address must be stored in a memory of the Voice Control Server so that other users can locate the user. Since the user is connecting via a workstation and Etherphone, the user identifier identifies the current software "process" through which the user is interacting with the system.

Claim 38 further requires "program code responsive to identification of one of the entries by a requesting process and configured to provide one of the identifier and the network protocol address to the requesting process."

¶ 126. As described above, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4.

INDEPENDENT CLAIM 43

Claim 43 claims "[a] computer program product for use with a computer system, the computer system executing a first process operatively coupled over a computer network to a second process and a server process, the computer program product comprising a computer usable medium having computer readable program code embodied therein."

¶ 127. The Etherphone system is implemented with software executed on a plurality of computing devices, including servers, workstations, and Etherphones. As described in Zellweger 1, the Etherphone system uses "microprocessor-controlled telephones to transmit voice over an Ethernet that also supports a voice file server and a voice synthesis server . . ." Zellweger 1, page 1. *See also* Swinehart 2, page 1 (describing how the capabilities provided by the Etherphone system "are presented to application programmers as program packages and network services."); Zellweger 1, page 2 ("Etherphone software is written in C"); *id.* ("Centralized server software limited the necessary size and speed of the Etherphone processor, and thus its cost . . ."); Terry, page 4 ("The server software and the initial workstation software was developed in the Cedar programming environment."). The computer systems are operatively connectable over a computer network to computer processes and server processes. As described in Zellweger 1, "conversations are established between two or more parties (Etherphones, servers, and so on) by performing remote procedure calls to the Voice Control Server." Zellweger 1, page 3. Remote procedure calls are inherently directed to "computer processes."

Claim 43 further requires "program code configured to access a directory database, the database having a network protocol address for a selected plurality of processes having on-line status with respect to the computer network, the network protocol address of each respective process forwarded to the database following connection to the computer network."

¶ 128. Etherphone describes a directory database for storing network addresses of on-line processes. For example, the Voice Control Server (also referred to as a "Telephone Control Server") stores network addresses for processes executed on each workstation and Etherphone. As described in Zellweger 1: "Users can place calls by specifying a name, a number, or other attributes of the called party. A system directory database for local Xerox employees (about 1000 entries) is stored on the Voice Control Server." Zellweger 1, page 4 (emphasis added). *See also* Swinehart 1, page 4 ("The Telephone Control Server manages voice switching by sending to each Etherphone or service the network addresses of the other participants."). The network addresses are sent to the Voice Control Server following connection to the computer network. *See, e.g.*, Swinehart 1, page 4 ("The telephone control server . . . uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations, and the registration of visitors in the offices of their colleagues."). Thus, when a user logs in to a workstation, the network protocol address of the workstation and the identity of the user are sent to the Voice Control Server.

Claim 43 also requires "program code responsive to one of the network protocol addresses and configured to establish a point-to-point communication link from the first process to the second process over the computer network."

¶ 129. After receiving the network protocol address from the Voice Control Server, the caller's workstation and Etherphone will establish a point-to-point connection with the callee's workstation and Etherphone. *See, e.g.*, Swinehart 1, page 4 (describing how after receiving a network address from the Voice Control Server, "voice datagrams are transmitted directly among the participants, bypassing the control server.").

INDEPENDENT CLAIM 44

The preamble of Claim 44 reads: "In a first computer process operatively coupled over a computer network to a second process and an address server, a method of establishing a point-to-point communication between the first and second processes comprising the steps of."

¶ 130. Etherphone describes a first computer process operatively coupled to a second process and an address server and further describe a method in the first process for establishing point-to-point communication with the second process. For example, in the Etherphone system, a first process executed on a first Etherphone or workstation contacts the Voice Control Server to learn the address of a second process executed on a second Etherphone or workstation. *See e.g.,* Swinehart 1, page 4 ("The Telephone Control Server manages voice switching by sending to each Etherphone or service the network addresses of the other participants."). The first process then uses the network address to establish point-to-point communication with the second process. *See, e.g., id.* ("Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server.").

Claim 44 also requires "following connection of the first process to the computer network forwarding to the address server a network protocol address at which the first process is connected to the computer network."

¶ 131. In the Etherphone system, when a user logs in to a particular workstation, the user's identity and the network protocol address of the workstation is forwarded to the Voice Control Server. *See, e.g.*, Swinehart 1, page 4 ("The telephone control server . . . uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations, and the registration of visitors in the offices of their colleagues.").

Claim 44 also requires "querying the address server as to whether the second process is connected to the computer network."

¶ 132. As mentioned above, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4. Moreover, "conversations are established between two or more parties (Etherphones, servers, and so on) by performing remote procedure calls to the Voice Control Server." Zellweger 1, page 3. Thus, when a first user at a first Etherphone (a first "process") calls a second user at a second Etherphone (a second "process"), the first Etherphone transmits a query in the form of a remote procedure call to determine the location of the second Etherphone.

See also Swinehart 1, page 2 ("Calls are to individuals, not locations . . . Logging in tells the telephone system where Karmen is."); Zellweger 1, page 5 ("An additional feature, called visiting, allows him to register his presence with a second workstation or Etherphone, such as during a meeting.").

Claim 44 further requires "receiving a network protocol address of the second process from the address server, when the second process is connected to the computer network."

¶ 133. As mentioned above, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4.

Claim 44 further requires "in response to the network protocol address of the second process, establishing a point-to-point communication link with the second process over the computer network."

¶ 134. As described in Swinehart 1, after receiving a network address from the Voice Control Server, "voice datagrams are transmitted directly among the participants, bypassing the control server." Swinehart 1, page 4.

DEPENDENT CLAIMS 5-7, 11-12, 14, 19-20, 22-23, 25, 30-31, 34-37, 39-42

Claim 5 of the '704 patent requires "searching the computer memory for an entry relating the second process; and retrieving a network protocol address of the second process in response to a positive determination of the on-line status of the second process."

¶ 135. These features are inherent in the Etherphone system. In order for the Voice Control Server to manage "voice switching by sending to each Etherphone or service the network addresses of the other participants" (Swinehart 1, page 4) it must inherently search the server memory for the network addresses related to the other workstation and Etherphone processes. In addition, the Voice Control Server maintains an "on-line status" for each process. For example, if a user is logged in to a particular Etherphone, the user's online status is "online" and associated with that Etherphone. A query (in the form of a remote procedure call) will then return the current location of the user's process to the requesting process (executed on another Etherphone or workstation). Swinehart 1 describes various different types of "on-line status" for users including "visiting" a workstation or Etherphone and "offline." *See* Swinehart 1, page 2

(describing a "do-not-disturb option" in which "internal callers were given an on-screen explanation for being turned away, while outside callers were routed to an attendant").

Claim 6 of the '704 patent requires "transmitting the network protocol address of the second process to the first process when the second process is determined in step C to have a positive on-line status with respect to the computer network."

¶ 136. In the Etherphone system, if a user is logged in and "online" then the Voice Control Server transmits the network address of the user's process (executed on the Etherphone or workstation) to the requesting process. *See, e.g.*, Swinehart 1, page 4 ("The *telephone control server* . . . uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations, and the registration of *visitors* in the offices of their colleagues.") (emphasis in original).

Claim 7 requires "generating an off-line message when the second process is determined in step C to have a negative on-line status with respect to the computer network; and transmitting the off-line message to the first process."

¶ 137. As described above, the Etherphone system includes a "do-not-disturb" option in which "internal callers were given an on-screen explanation for being turned away, while outside callers were routed to an attendant." Swinehart 1, page 4.

Claim 11 requires "querying the server as to the on-line status of the first callee process; and receiving a network protocol address of the first callee process over the computer network from the server."

¶ 138. The Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4. In addition, the Voice Control Server maintains an "on-line status" for each process. For example, if a user is logged in to a particular Etherphone, the user's online status is "online" and associated with that Etherphone. A query (in the form of a remote procedure call) will then return the current location of the user's process to the requesting process (executed on another Etherphone or workstation). Swinehart 1 describes various different types of "on-line status" for users including "visiting" a workstation or Etherphone and "offline." *See* Swinehart 1, page 2.

Claim 12 claims "providing an element representing a second communication line."

¶ 139. The Etherphone system inherently provides an element representing a second communication line. For example, using the Etherphone system, a user may receive and answer a call while already on an existing call. *See, e.g.*, Swinehart 1, page 2 (describing how users can place and receive other calls during a "background call"). Thus, multiple sets of graphical icons such as the ones shown in Figure 4 of Zellweger 1 were inherently displayed in the Etherphone system (e.g., graphical cards in a rolodex to represent callee processes, telephones to represent communication lines, and users talking on telephones to represent a callee process associated with a particular communication line). Alternatively, as set forth below, Claim 12 is invalid under 35 U.S.C. § 103(a) as being unpatentable over Etherphone in view of Pinard.

Claim 14 requires "providing a user interface element representing a second callee process; and establishing a conference point-to-point communication link between the caller process and the first and second callee process, in response to the user associating the element representing the second callee process with the element representing the first communication line."

¶ 140. Etherphone describes this limitation. For example, Figure 8 of Swinehart 1 illustrates four user interface elements representing four different callee processes (four different users). The four callee processes are each associated with a graphical element representing a communication line – i.e., a telephone graphic and graphical window representing a teleconference (titled "conference at 3PM re: Budget"). *See* Swinehart 1, page 3.

Claim 19 requires "wherein the caller process further comprises a visual display and the user interface comprises a graphic user interface."

¶ 141. Etherphone illustrates a visual display which allows the caller to control the operation of the Etherphone. *See, e.g.*, Figure 3 of Zellweger 1 (illustrating a "workstation telephone management windows") and Figures 1-10 of Swinehart 1 (illustrating a series of windows for controlling an Etherphone).

Claim 20 requires "wherein the steps of establishing a point-to-point link as described in step C is performed in response to manipulation of the graphic elements on the graphic user interface."

¶ 142. As described in Swinehart 1, to establish a call, user's can "select names or numbers from anywhere on the [Etherphone telephone management windows], use either of two

directory tools that present browsable lists of names and associated telephone numbers as speed-dialing buttons, or redial any previously-made call by clicking on its conversation log entry. Calls can also be placed by name or number from the telephone keypad." Zellweger 1, page 4. As described above, a call from one workstation/Etherphone to another workstation/Etherphone comprises a point-to-point link.

Claim 22 requires "program code for querying the server as to the on-line status of the first callee process; and program code for receiving a network protocol address of the first callee process over the computer network from the server."

¶ 143. As mentioned above, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4. Moreover, "conversations are established between two or more parties (Etherphones, servers, and so on) by performing remote procedure calls to the Voice Control Server." Zellweger 1, page 3. Thus, when a first user at a first Etherphone (a first "process") calls a second user at a second Etherphone (a second "process"), the first Etherphone transmits a query in the form of a remote procedure call to determine the location of the second Etherphone. *See also* Swinehart 1, page 2 ("Calls are to individuals, not locations . . . Logging in tells the telephone system where Karmen is."). In addition, the Voice Control Server maintains an "on-line status" for each process. For example, if a user is logged in to a particular Etherphone, the user's online status is "online" and associated with that Etherphone. A query (in the form of a remote procedure call) will then return the current location of the user's process to the requesting process (executed on another Etherphone or workstation). Swinehart 1 describes various different types of "on-line status" for users including "visiting" a workstation or Etherphone and "offline." *See* Swinehart 1, page 2 (describing a "do-not-disturb option" in which "internal callers were given an on-screen explanation for being turned away, while outside callers were routed to an attendant").

Claim 23 requires "program code for generating an element representing a second communication line."

¶ 144. The Etherphone system is inherently capable of providing an element representing a second communication line. For example, the Etherphone system is capable of conference calling. *See, e.g.*, Swinehart 1, page 3 (describing "negotiated conference calls"). In addition,

using the Etherphone system, a user may receive and answer a call while already on an existing call. *See, e.g.*, Swinehart 1, page 2 (describing how users can place and receive other calls during a "background call"). Thus, multiple sets of graphical icons such as the ones shown in Figure 4 of Zellweger 1 were inherently displayed in the Etherphone system (e.g., graphical cards in a rolodex to represent callee processes, telephones to represent communication lines, and users talking on telephones to represent a callee process associated with a particular communication line).

Claim 25 requires "program code for generating an element representing a second callee process; and program code means, responsive to the user associating the element representing the second callee process with the element representing the first communication line, for establishing a conference communication link between the caller process and the first and second callee process."

¶ 145. Etherphone describes this limitation. For example, Figure 8 of Swinehart 1 illustrates four user interface elements representing four different callee processes (four different users). The four callee processes are each associated with a graphical element representing a communication line – i.e., a telephone graphic and graphical window representing a teleconference (titled "conference at 3PM re: Budget"). *See* Swinehart 1, page 3.

Claim 30 requires that the "computer system further comprises a visual display and the user interface comprises a graphic user interface."

¶ 146. Etherphone illustrates a visual display which allows the caller to control the operation of the Etherphone. *See, e.g.*, Figure 3 of Zellweger 1 (illustrating a "workstation telephone management windows") and Figures 1-10 of Swinehart 1 (illustrating a series of windows for controlling an Etherphone).

Claim 31 requires that "the element representing the first communication line and the element representing the first callee process are graphic elements and wherein the program code for establishing a point-to-point communication link from the caller process to the first callee process further comprises: program code, responsive to manipulation of the graphic elements on the graphic user interface, for establishing the point-to-point communication link from the caller process to the first callee process."

¶ 147. As described above, Etherphone illustrates a visual display which allows the caller to control the operation of the Etherphone. *See, e.g.*, Figure 3 of Zellweger 1 (illustrating a "workstation telephone management windows") and Figures 1-10 of Swinehart 1 (illustrating a

series of windows for controlling an Etherphone). Figure 3 of Zellweger 1, for example, shows a "phone" button representing a first communication line and several other graphical elements representing callee processes. As described in Zellweger 1, "[a] variety of convenient workstation dialing methods are provided: a user can . . . select names or numbers from anywhere on the [Etherphone telephone management windows], use either of two directory tools that present browsable lists of names and associated telephone numbers as speed-dialing buttons, or redial any previously-made call by clicking on its conversation log entry. Calls can also be placed by name or number from the telephone keypad." Zellweger 1, page 4. In addition, the top row of Figure 4 of Zellweger 1 shows a series of graphical icons used for placing a call including a personal telephone directory, a telephone, and a picture of a user on the phone (to indicate a call is in process). In this example, the personal telephone directory, displayed as a graphical rolodex, includes a plurality of graphical elements representing callees (i.e., with a separate card in the rolodex for each callee). The icon of the telephone and the icon with the picture of a user talking on the phone represents a telephone communication line. As described in Zellweger 1, "[a]n active conversation is represented as a conversation between two people with a superimposed indication of the other party's name (also shown in Figure 4)." Zellweger 1, pages 4-5.

Claim 34 requires "modifying the compilation of entries."

¶ 148. Etherphone discloses modifying the entries stored on the Voice Control Server as users log-in and log-out of workstations. As described in Swinehart 1:

> The *telephone control server* controls voice conversations, implements the stand-alone behavior of telephone instruments and coordinates the activities of workstations and adjacent telephones in their implementation of the various voice capabilities. In addition, it stores personal preference information about each user that allows it to support advanced features such as *ring motifs* and *subdued ringing* without involving workstation programs. It uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations and the registration of *visitors* in the offices of their colleagues.

Swinehart 1, page 4 (underline emphasis added). Thus, the entries stored on the Voice Control Server are updated dynamically "in order to provide calls to individuals rather than fixed locations."

Claim 35 requires "adding an entry to the compilation upon the occurrence of a predetermined event."

¶ 149. Etherphone inherently discloses this limitation. The predetermined event may include, for example, adding a new workstation and Etherphone to the network, powering on an existing Etherphone or workstation, adding a new Voice Control Server, and/or logging in a user to the system from a new workstation/Etherphone. Each of these events may require adding an entry to the Voice Control Server. *See, e.g.,* Swinehart 1, page 4 (describing how the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants."). In order to "manage" voice switching by sending network addresses of logged in participants, the Voice Control Server must be capable of adding entries to its database in response to "predetermined events."

Claim 36 requires that "the predetermined event comprises notification by a user process of an assigned network protocol address."

¶ 150. Etherphone discloses this limitation. For example, when a user logs in to a workstation/Etherphone, the identity of the user and the network address of the workstation/Etherphone is transmitted to the voice control server so that the user can be located by other users. As mentioned above, the Voice Control Server "uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations and the registration of visitors in the offices of their colleagues." Swinehart 1, page 4.

Claim 37 requires "deleting an entry from the compilation upon the occurrence of a predetermined event."

¶ 151. Etherphone inherently describes this limitation. As described in Zellweger:

> If an Etherphone user logs in at a workstation, his calls can be automatically forwarded to the adjacent Etherphone. An additional feature, called visiting, allows him to register his presence with a second workstation or Etherphone, such as during a meeting. Registering with the destination location allows users to travel more freely than forwarding calls from the home location does. Each visit request cancels any earlier requests. The common problem of forgetting to cancel forwarding is eased by ringing both Etherphones during visiting.

Zellweger, page 5 (emphasis added). *See also* Swinehart, page 2 (describing how after Karmen leaves Lee's office "an additional call to Karmen . . . reminds Lee to terminate the visiting arrangement."). In these examples, "terminating" or "cancelling" the user's visiting status

inherently requires deleting the association between the visiting user and the network address of the workstation/Etherphone.

Claim 39 requires "program code configured to modify the compilation of entries."

¶ 152. Etherphone discloses modifying the entries stored on the Voice Control Server as users log-in and log-out of workstations. As described in Swinehart 1:

> The *telephone control server* controls voice conversations, implements the stand-alone behavior of telephone instruments and coordinates the activities of workstations and adjacent telephones in their implementation of the various voice capabilities. In addition, it stores personal preference information about each user that allows it to support advanced features such as *ring motifs* and *subdued ringing* without involving workstation programs. It uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations and the registration of *visitors* in the offices of their colleagues.

Swinehart 1, page 4 (underline emphasis added). *See also* Swinehart, page 2 (describing how after Karmen leaves Lee's office "an additional call to Karmen . . . reminds Lee to terminate the visiting arrangement."). Thus, the entries stored on the Voice Control Server are updated dynamically "in order to provide calls to individuals rather than fixed locations" and to "terminate" old, outdated entries.

Claim 40 requires "program code configured to add an entry to the compilation upon the occurrence of a predetermined event."

¶ 153. Etherphone inherently discloses this limitation. The predetermined event may include, for example, adding a new workstation and Etherphone to the network, powering on an existing Etherphone or workstation, adding a new Voice Control Server, and/or logging in a user to the system from a new workstation/Etherphone. Each of these events may require adding an entry to the Voice Control Server. *See, e.g.,* Swinehart 1, page 4 (describing how the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants."). In order to "manage" voice switching by sending network addresses of logged in participants, the Voice Control Server must be capable of adding entries to its database in response to "predetermined events."

Claim 41 requires that the "predetermined event comprises notification by a process of an assigned network protocol address."

¶ 154. Etherphone inherently discloses this limitation. For example, when a user logs in to a workstation/Etherphone, the identity of the user and the network address of the

workstation/Etherphone is transmitted to the voice control server so that the user can be located by other users. As mentioned above, the Voice Control Server "uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations and the registration of visitors in the offices of their colleagues." Swinehart 1, page 4.

Claim 42 requires "program code configured to delete an entry from the compilation upon the occurrence of a predetermined event."

¶ 155. Etherphone inherently describes this limitation. As described in Zellweger:

> If an Etherphone user logs in at a workstation, his calls can be automatically forwarded to the adjacent Etherphone. An additional feature, called visiting, allows him to register his presence with a second workstation or Etherphone, such as during a meeting. Registering with the destination location allows users to travel more freely than forwarding calls from the home location does. Each visit request cancels any earlier requests. The common problem of forgetting to cancel forwarding is eased by ringing both Etherphones during visiting.

Zellweger, page 5 (emphasis added). *See also* Swinehart, page 2 (describing how after Karmen leaves Lee's office "an additional call to Karmen . . . reminds Lee to terminate the visiting arrangement."). In these examples, "terminating" or "cancelling" the user's visiting status inherently requires deleting the association between the visiting user and the network address of the workstation/Etherphone.

2. Obviousness Rejections

¶ 156. The following is a quotation of 35 U.S.C. §103 (a) which forms the basis for the following obviousness rejections:

> A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

(i) Etherphone in view NetBIOS

¶ 157. Claim 3 is invalid under 35 U.S.C. § 103(a) as being unpatentable over Etherphone in view of NetBIOS.

Claim 3 of the '704 patent requires "a timer, operatively coupled to the processor, for time stamping the network protocol addresses stored in the memory."

¶ 158. Etherphone does not explicitly describe a timer for time stamping network protocol address entries stored on the Voice Control Server. However, time stamping was a well known technique at the time the application which resulted in the '704 patent was filed. For example, the NBNS described in NetBIOS includes a timer for time-stamping name/IP address entries. As described in NetBIOS, "[t]he NBNS may impose a 'time-to-live' on each name it registers. The registering node is made aware of this time value during the name registration procedure." NetBIOS at 382. Similarly, as described in NetBIOS:

> If an end-node holds any names that have finite time-to-live values, then that node must periodically send a status report to the NBNS. Each name is reported using the NAME REFRESH REQUEST packet. These status reports restart the timers of both the NBNS and the reporting node. However, the only timers which are restarted are those associated with the name found in the status report. Timers on other names are not affected. *Id.*

(ii) Motivation to Combine Etherphone With NetBIOS

¶ 159. The motivation to combine Etherphone with NetBIOS exists due to the problem being solved. For example, it would have been obvious to a person of ordinary skill in the art at the time of the invention to include the capability of time stamping network protocol address entries at the Etherphone Voice Control Server to determine the length of time that a user has been online at a particular workstation/Etherphone and/or to remove stale entries.

(iii) Etherphone in view of Vin

¶ 160. As described above, Claim 32 is anticipated by Etherphone. Alternatively, Claim 32 should be rejected under 35 U.S.C. § 103(a) as being unpatentable over Etherphone in view of Harrick M. Vin, et al., *Multimedia Conferencing in the Etherphone Environment*, IEEE COMPUTER SOCIETY (October 1991) ("Vin").

Claim 32 claims "[a] method of locating a process over a computer network comprising the steps of: a. maintaining an Internet accessible list having a plurality of selected entries, each entry comprising an identifier and a corresponding Internet protocol address of a process currently connected to

the Internet, the Internet Protocol address added to the list following connection of the process to the computer network."

¶ 161. As described in Etherphone: "The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server." Swinehart 1, page 4. Thus, the Telephone Control Server (also referred to as the Voice Control Server) stores a list of network addresses which are made available to workstations and Etherphones. In addition, the Voice Control Server associates different user identifiers with each network protocol address. For example, a user may log in to any workstation and, thereafter, calls to that user will be directed to that workstation and its associated Etherphone. As described in Swinehart 1:

> The *telephone control server* controls voice conversations, implements the stand-alone behavior of telephone instruments and coordinates the activities of workstations and adjacent telephones in their implementation of the various voice capabilities. In addition, it stores personal preference information about each user that allows it to support advanced features such as *ring motifs* and *subdued ringing* without involving workstation programs. It uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations and the registration of *visitors* in the offices of their colleagues.

Swinehart 1, page 4 (underline emphasis added). The network addresses may be Internet protocol addresses. For example, the Etherphone system was intended for use in "multiple networks and communication protocols." Terry, page 3. *See also* Terry, Abstract ("the voice manager stores voice on a special voice file server that is accessible via the local internet."). While the Etherphone papers do not explicitly describe using Internet Protocol (IP) addresses, another Etherphone reference, Vin, explicitly describes using IP addresses within the Etherphone system. *See, e.g.*, Vin, page 77, Figure 5 (illustrating a "protocol stack and format" used in an Etherphone system which includes internet protocol (IP) packets).

Claim 32 also requires "in response to identification of one of the list entries by a requesting process, providing one of the identifier and the corresponding Internet protocol address of the identified entry to the requesting process."

¶ 162. As mentioned above, when a first user attempts to call a second user from a workstation and Etherphone, the Voice Control Server provides the current network protocol address of the second user to the requesting process executed on the workstation/ Etherphone of the first user. Using the network address, the requesting process then initiates a communication

session with the workstation and Etherphone of the second user. *See, e.g.*, Swinehart 1, page 4 ("The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server.").

(iv) Motivation to Combine Etherphone and Vin

¶ 163. A motivation to combine Etherphone and Vin exists because they both describe the same Etherphone system. It would have been obvious to a person of ordinary skill in the art at the time of the invention to combine one reference describing the Etherphone system (Etherphone) with another reference describing the same Etherphone system (Vin). Moreover, the Etherphone system was intended for use in "multiple networks and communication protocols." Terry, page 3. Thus, it would have been obvious to a person of ordinary skill in the art at the time of the invention to combine Etherphone with other references (such as Vin) which describe the use of IP addresses.

(v) Etherphone in view of Vin and Further in View of RFC 1531

¶ 164. Claim 33 should be rejected under 35 U.S.C. § 103(a) as being unpatentable over Etherphone in view of Vin and further in view of Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531") .

Claim 33 claims "[a] method for locating processes having dynamically assigned network protocol addresses over a computer network."

¶ 165. As discussed above, Etherphone discloses a method of locating network protocol addresses over a computer network. *See, e.g.*, Swinehart 1, page 4 ("The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants."). While Etherphone does not explicitly state that the network protocol addresses may be Internet protocol addresses, the Etherphone system was intended for use in "multiple networks and communication protocols." Terry, page 3. *See also* Terry, Abstract ("the voice manager stores voice on a special voice file server that is accessible via the local internet."). Vin explicitly describes using the Internet Protocol (IP) within the Etherphone system. *See, e.g.*, Vin, page 77, Figure 5 (illustrating a "protocol stack and format" used in an Etherphone system which includes internet protocol (IP) packets). In addition, as described in

RFC 1531, IP addresses were known to be dynamically assigned. *See, e.g.*, RFC 1531, Section 2.2 (describing the "dynamic allocation of network addresses").

Claim 33 also requires "maintaining, in a computer memory, a network accessible compilation of entries, selected of the entries comprising a network protocol address and a corresponding identifier of a process connected to the computer network, the network protocol address of the corresponding process assigned to the process upon connection to the computer network."

¶ 166. The Voice Control Server maintains "a compilation of entries . . . comprising a network protocol address and a corresponding identifier." For example, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4. In order to send the network addresses of the other participants, the Voice Control Server must inherently store the network protocol addresses in a "computer memory." The network addresses are stored in the memory with corresponding user identifiers. For example, the Voice Control Server "uses dynamic information linking users to workstations in order to provide calls to individuals rather than fixed locations, and the registration of visitors in the offices of their colleagues." *Id.* Thus, if a user logs in to any workstation, the identity of that user and the associated network address must be stored in a memory of the Voice Control Server so that other users can locate the user. Since the user is connecting via a workstation and Etherphone, the user identifier identifies the current software process through which the user is interacting with the system.

Claim 33 also requires "in response to identification of one of the entries by a requesting process providing one of the identifier and the network protocol address to the requesting process."

¶ 167. As described above, the Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4.

¶ 168. The other independent claims of the '704 patent state, more generally, that the network protocol address is assigned or transmitted to the database following the connection of the computer to the computer network. *See* Claim 1 ("transmitting to the server a network protocol address received by the first process following connection to the computer network"); Claim 2 ("each network protocol address stored in the memory following connection of a

respective process to the computer network"); Claim 4 ("the network protocol addresses received following connection of the respective process to the computer network"); Claim 32 ("the Internet Protocol address added to the list following connection of the process to the computer network"); Claim 38 ("the network protocol address of the corresponding process assigned to the process upon connection to the computer network"); Claim 43 ("the network protocol address of each respective process forwarded to the database following connection to the computer network"); Claim 44 ("following connection of the first process to the computer network forwarding to the address server a network protocol address").

¶ 169. As described above, Etherphone and Vin inherently describes these features. For example, on many networks, including the TCP/IP networks, network addresses are assigned "following connection to the computer network." *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). Thus, in at least some instances, the computer system on which NetBIOS was used received IP addresses dynamically, after connecting to the computer network. Consequently, dynamic address assignment is inherent in the NetBIOS reference.

¶ 170. Alternatively, Claims 1-2, 4-7, 10-12, 14, 19-23, 25, and 30-44 should be rejected under 35 U.S.C. § 103(a) as being unpatentable over Etherphone in view of Vin and further in view of RFC 1531, which describes how TCP/IP addresses were dynamically assigned. *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks).

(vi) Motivation to Combine Etherphone, Vin, and RFC 1531

¶ 171. As mentioned above, a motivation to combine Etherphone and Vin exists because they both describe the same Etherphone system. In addition, a motivation to combine these references with RFC 1531 exists due to the problem to be solved. In particular, Vin describes the use of IP addresses within an Etherphone system and RFC 1531 describes techniques for dynamically assigning IP addresses. One of ordinary skill in the art would have been motivated to use dynamic IP address assignment because it eliminates the burdensome task of manually assigning IP addresses for all networked computers (e.g., workstations and Etherphones) and

allows for "automatic reuse of an address that is no longer needed by the host to which it was assigned." RFC 1531, page 2 (Section 1, Introduction).

(vii) Etherphone in view of Pinard

¶ 172. Claims 10-17, 19-28, and 30-31should be rejected under 35 U.S.C. § 103 as being unpatentable over Etherphone in view of Pinard.

INDEPENDENT CLAIM 10

¶ 173. Etherphone anticipates Claim 10 for the reasons stated above. Alternatively, Claim 10 is invalid under 35 U.S.C. § 103(a) as being unpatentable over Etherphone in view of Pinard.

Claim 10 claims "a method for establishing a point-to-point communication link from a caller process to a callee process over a computer network . . ."

¶ 174. As described in Swinehart 1: "The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server." (emphasis added). Swinehart 1, page 2. Thus, after retrieving a network address of a callee device from the Voice Control Server, a workstation or Etherphone communicates directly over a point-to-point communication link with the callee device identified by the network address. *See also* Zellweger, page 2 ("Etherphones digitize, packetize, and encrypt telephone-quality voice (64 kilobits/second, with silence suppression) and send it to each other directly over an Ethernet . . ."); Swinehart 2, page 1 ("Etherphones digitize and encrypt telephone quality audio and transmit it in packet form directly over an Ethernet.").

Claim 10 further claims "the caller process having a user interface and being operatively connectable to the callee process and a server over the computer network . . ."

¶ 175. The workstations described in Etherphone includes a graphical user interface (GUI). *See, e.g.*, Figures 1-10 of Swinehart 1 (illustrating various GUI features presented on the workstation display). *See also* Zellweger 1, Figures 3-4 (illustrating "telephone management windows" (Figure 3) and icons representing callers, callees and telephone lines (Figure 4)). The workstations may be Apple Macintoshes or Xerox 6085s. *See* Swinehart 1, page 1. The workstations are operatively connectable to the callee process and a server over the computer

network. As previously described, "[t]he telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server." Swinehart 1, page 2.

Claim 10 requires "providing a user interface element representing a first communication line."

¶ 176. Pinard discloses a user interface element representing a first communication line. For example, Figure 6 of Pinard illustrates a first call icon 23 which represents a first communication line and a second call icon 29 which represents a second communication line. In the example shown in Figure 6, the first call icon 23 represents a telephone call between "Debbie" and "John" and the second call icon 29 represents a telephone call between "Debbie" and "Mary." *See, e.g.*, Pinard, Col. 5, lines 23-30.

Claim 10 also requires "providing a user interface element representing a first callee process."

¶ 177. Pinard describes "a user interface element representing a first callee process." In the example shown in Figure 6 of Pinard, a first user interface element 21 is shown for the callee named "John" and a second user interface element is shown for the callee named "Mary." *See, e.g.*, Pinard, Col. 5, lines 23-30.

Claim 10 further requires "establishing a point-to-point communication link from the caller process to the first callee process, in response to a user associating the element representing the first callee process with the element representing the first communication line."

¶ 178. As described above, Etherphone describes establishing a point-to-point communication link between a caller process and a callee process. *See, e.g.*, Swinehart 1, page 2 ("voice datagrams are transmitted directly among the participants, bypassing the control server"). Pinard discloses that a point-to-point communication link is established in response to a user associating an element representing the first callee process with the element representing a first communication line. For example, Figure 3 of Pinard illustrates clicking and dragging an icon representing a callee from a directory 17 into a call setup icon 15. Once the callee answers the call, the call setup icon 15 becomes a call icon 23 as illustrated in Figure 4 of Pinard. *See, e.g.*, Pinard, Col. 4, lines 38-51 (describing how "[t]he user can then drag the icon or the name of the

person to be called into the call setup icon . . . As soon as John answers the call, the application software program changes the call setup icon to a call icon designated as 23, and establishes a new call setup icon 24 spaced from the icon 23."). Similarly, Figure 6 illustrates how a point-to-point communication link may be established by clicking and dragging a callee icon 21 into an existing call icon 29. *See* Pinard, Col. 5, lines 36-37 ("Now to conference all parties, the user Debbie merely drags the John icon to the call icon 29.").

INDEPENDENT CLAIM 21

Claim 21 claims "[a] computer program product for use with a computer system comprising: a computer usable medium having program code embodied in the medium."

¶ 179. The functionality of the Etherphone system is implemented in software, which is inherently stored on a computer usable medium. As described in Swinehart 2, the capabilities provided by the Etherphone system "are presented to application programmers as program packages and network services." Swinehart 2, page 1. *See also* Zellweger 1, page 2 ("Etherphone software is written in C"); *id.* ("Centralized server software limited the necessary size and speed of the Etherphone processor, and thus its cost . . ."); *id.*, page 1 (describing how the Etherphone system uses "microprocessor-controlled telephones to transmit voice over an Ethernet that also supports a voice file server and a voice synthesis server . . ."); Terry, page 4 ("The server software and the initial workstation software was developed in the Cedar programming environment.").

Claim 21 requires "establishing a point-to-point communication link from a caller process to a callee process over a computer network, the caller process having a user interface and being operatively connectable to the callee process and a server over the computer network."

¶ 180. First, Etherphone describes establishing a point-to-point communication link between a caller process and a callee process over a computer network. *See, e.g.*, Swinehart 1, page 2 ("voice datagrams are transmitted directly among the participants, bypassing the control server"). Second, Etherphone discloses that the caller process (i.e., the software executed on the caller's machine) includes a user interface. *See, e.g.*, Zellweger 1, page 4, Figures 3-4 ("a user can . . . select names or numbers from anywhere on the [Etherphone telephone management windows], use either of two directory tools that present browsable lists of names and associated telephone numbers as speed-dialing buttons, or redial any previously-made call by clicking on its

conversation log entry. Calls can also be placed by name or number from the telephone keypad."). Finally, Etherphone describes "being operatively connectable to the callee process and a server over the computer network." *See, e.g.*, Swinehart 1, page 4 ("The telephone control server manages voice switching by sending to each Etherphone or service the network addresses of the other participants. Thereafter, voice datagrams are transmitted directly among the participants, bypassing the control server.").

Claim 21 also requires "program code for generating an element representing a first communication line."

¶ 181. Pinard discloses program code for generating an element representing a first communication line. For example, Figure 6 of Pinard illustrates a first call icon 23 which represents a first communication line and a second call icon 29 which represents a second communication line. In the example shown in Figure 6, the first call icon 23 represents a telephone call between "Debbie" and "John" and the second call icon 29 represents a telephone call between "Debbie" and "Mary." *See, e.g.*, Pinard, Col. 5, lines 23-30.

Claim 21 also requires "program code for generating an element representing a first callee process."

¶ 182. Pinard describes program code for generating an element representing a first callee process. In the example shown in Figure 6 of Pinard, a first user interface element 21 is shown for the callee named "John" and a second user interface element is shown for the callee named "Mary." *See, e.g.*, Pinard, Col. 5, lines 23-30.

Claim 21 also requires "program code, responsive to a user associating the element representing the first callee process with the element representing the first communication line, for establishing a point-to-point communication link from the caller process to the first callee process."

¶ 183. As described above, Etherphone describes establishing a point-to-point communication link between a caller process and a callee process. *See, e.g.*, Swinehart 1, page 2 ("voice datagrams are transmitted directly among the participants, bypassing the control server"). Pinard discloses that a point-to-point communication link is established in response to a user associating an element representing the first callee process with the element representing a first communication line. For example, Figure 3 of Pinard illustrates clicking and dragging an icon representing a callee from a directory 17 into a call setup icon 15. Once the callee answers the

call, the call setup icon 15 becomes a call icon 23 as illustrated in Figure 4 of Pinard. *See, e.g.*, Pinard, Col. 4, lines 38-51 (describing how "[t]he user can then drag the icon or the name of the person to be called into the call setup icon . . . As soon as John answers the call, the application software program changes the call setup icon to a call icon designated as 23, and establishes a new call setup icon 24 spaced from the icon 23."). Similarly, Figure 6 illustrates how a point-to-point communication link may be established by clicking and dragging a callee icon 21 into an existing call icon 29. *See* Pinard, Col. 5, lines 36-37 ("Now to conference all parties, the user Debbie merely drags the John icon to the call icon 29.").

DEPENDENT CLAIMS 11-17, 19-20, 22-28, AND 30-31

Claims 11 and 22 require "querying the server as to the on-line status of the first callee process; and receiving a network protocol address of the first callee process over the computer network from the server" and "program code for querying the server as to the on-line status of the first callee process; and program code for receiving a network protocol address of the first callee process over the computer network from the server," respectively.

¶ 184. The Voice Control Server "manages voice switching by sending to each Etherphone or service the network addresses of the other participants." Swinehart 1, page 4. In addition, the Voice Control Server maintains an "on-line status" for each process. For example, if a user is logged in to a particular Etherphone, the user's online status is "online" and associated with that Etherphone. A query (in the form of a remote procedure call) will then return the current location of the user's process to the requesting process (executed on another Etherphone or workstation). Swinehart 1 describes various different types of "on-line status" for users including "visiting" a workstation or Etherphone and "offline." *See* Swinehart 1, page 2.

Claim 12 and 24 require "providing an element representing a second communication line" and "program code for generating an element representing a second communication line," respectively.

¶ 185. The graphical user interface described in Pinard provides an element representing a second communication line. For example, call icons 23 and 29 representing two communication lines are shown in Figure 6 of Pinard. *See* Pinard, Col. 5, lines 31-40, Figure 6 ("Now there are clearly two calls in progress . . .").

Claims 13 and 24 require "terminating the point-to-point communication link from the caller process to the first callee process, in response to the user

disassociating the element representing the first callee process from the element representing the first communication line" and "program code, responsive to the user disassociating the element representing the first callee process from the element representing the first communication line, for terminating the point-to-point communication link from the caller process to the first callee process," respectively.

¶ 186. Figure 6 of Pinard illustrates how the call represented by call icon 23 is terminated by dragging the user icon for "John" 21 out of the call icon 23. Similarly, Figure 11 illustrates how a call is terminated by dragging the user icon to a "waste basket" icon 26.

Claim 13 and 24 further require "establishing a different point-to-point communication link from the caller process to the first callee process, in response to the user associating the element representing the first callee process with the element representing the second communication line" and "program code responsive to the user associating the element representing the first callee process with the element presenting the second communication line, for establishing a different point-to-point communication link from the caller process to the first callee process," respectively.

¶ 187. In Figure 6, the callee process icon for "John" 21 is dragged from call icon 23 to call icon 29, thereby terminating the call represented by call icon 23 and establishing a different link with the callee process represented by icon 21 (in this case a conference call with "John," "Mary," and "Debbie"). Moreover, once a callee is removed from a call by clicking and dragging the callee's icon, a new call can always be established with the callee by dragging the callee's icon to a call setup icon. *See, e.g.*, Pinard, Figure 3 (showing a callee icon dragged from a directory to a call setup icon to establish a call). *See also* Pinard, Col. 4, lines 22-31.

Claims 14 and 25 require "providing a user interface element representing a second callee process; and . . . establishing a conference point-to-point communication link between the caller process and the first and second callee process, in response to the user associating the element representing the second callee process with the element representing the first communication line" and "program code for generating an element representing a second callee process; and program code means, responsive to the user associating the element representing the second callee process with the element representing the first communication line, for establishing a conference communication link between the caller process and the first and second callee process," respectively.

¶ 188. In Figure 6 of Pinard, the user interface element for "John" 21 is dragged from call icon 23 to call icon 29, thereby creating a conference call between "John," "Mary," and

"Debbie." See Pinard, Col. 5, lines 31-44 ("Now to conference all parties, the user Debbie merely drags the John icon to the call icon 29.").

Claims 15 and 26 require "removing the second callee process from the conference point-to-point communication link in response to the user disassociating the element representing the second callee process from the element representing the first communication line" and "program code, responsive to the user disassociating the element representing the second callee process from the element representing the first communication line, for removing the second callee process from the conference communication link," respectively.

¶ 189. In Pinard, any callee process can be removed from a conference call by dragging the element representing the callee process from the conference call icon. For example, Figure 8 of Pinard shows the user icon "Debbie" removed from conference call represented by call icon 32, thereby "breaking Debbie's line from the conference." Pinard, Col. 6, lines 14-15.

Claims 16 and 27 require "providing a user interface element representing a communication line having a temporarily disabled status" and "program code for generating an element representing a communication line having a temporarily disabled status," respectively.

¶ 190. Examples of a "temporarily disabled status" provided in the '704 patent include "line on hold" and "line on mute." *See, e.g.*, '704 patent, Claims 17 and 18. Pinard describes a user interface element representing a communication line having a temporarily disabled status. For example, Figure 12 illustrates a "hard hold" icon 39 to which user icons representing callers/callees 41 may be dragged to put the callers/callees on hold. *See, e.g.*, Pinard, Col. 6, lines 36-53 ("To place Mary on hard hold, Debbie drags Mary's icon 28 to the hard hold icon 39.").

Claims 16 and 27 also require "temporarily disabling a point-to-point communication link between the caller process and the first callee process, in response to the user associating the element representing the first callee process with the element representing the communication line having a temporarily disabled status" and "program code, responsive association of the element representing the first callee process with the element representing the communication line having a temporarily disabled status, for temporarily disabling the point-to-point communication link between the caller process and the first callee process," respectively.

¶ 191. In Pinard, in response to an icon of a caller/callee 41 being moved into the hard hold icon 39, the caller/callee is placed on hold. *See, e.g.*, Pinard, Col. 6, lines 36-53 ("To place Mary on hard hold, Debbie drags Mary's icon 28 to the hard hold icon 39.").

Claims 17 and 28 require that "the element provided in step D represents a communication line on hold status" and "the communication line having a temporarily disabled status comprises a communication line on hold status," respectively.

¶ 192. In Pinard, in response to an icon of a caller/callee 41 being moved into the hard hold icon 39, the caller/callee is placed on hold. *See, e.g.*, Pinard, Col. 6, lines 36-53 ("To place Mary on hard hold, Debbie drags Mary's icon 28 to the hard hold icon 39.").

Claims 19 and 30 require "wherein the caller process further comprises a visual display and the user interface comprises a graphic user interface" and "wherein the computer system further comprises a visual display and the user interface comprises a graphic user interface," respectively.

¶ 193. Pinard discloses a graphical user interface on a visual display which allows the caller to control the operation of the telephone. *See, e.g.*, Pinard, Figures 2-16 and Col. 6, lines 36-53 ("To place Mary on hard hold, Debbie drags Mary's icon 28 to the hard hold icon 39.").

Claims 20 and 31 requires "wherein the steps of establishing a point-to-point link as described in step C is performed in response to manipulation of the graphic elements on the graphic user interface" and "program code, responsive to manipulation of the graphic elements on the graphic user interface, for establishing the point-to-point communication link from the caller process to the first callee process," respectively.

¶ 194. As described above, Pinard discloses that a point-to-point communication link is established in response to a user associating a graphic element representing a callee process with a graphic element representing a communication line. For example, Figure 3 of Pinard illustrates clicking and dragging an icon representing a callee from a directory 17 into a call setup icon 15.

Once the callee answers the call, the call setup icon 15 becomes a call icon 23 as illustrated in Figure 4 of Pinard. *See, e.g.*, Pinard, Col. 4, lines 38-51 (describing how "[t]he user can then drag the icon or the name of the person to be called into the call setup icon . . . As soon as John answers the call, the application software program changes the call setup icon to a call icon designated as 23, and establishes a new call setup icon 24 spaced from the icon 23."). Similarly, Figure 6 illustrates how a point-to-point communication link may be established by clicking and dragging a callee icon 21 into an existing call icon 29. *See* Pinard, Col. 5, lines 36-37 ("Now to conference all parties, the user Debbie merely drags the John icon to the call icon 29.").

(viii) Motivation to Combine Etherphone and Pinard

¶ 195. A motivation to combine Etherphone and Pinard exists due to the problem to be solved. The Pinard reference relates to the field of telephony, and in particular to a method of indicating the status of various calls, to a user. *See* Pinard, Col. 1, lines 5-7. Indeed, the graphical user interface described in Pinard could be used in any system that operates a telephony application on a personal computer or on a personal computer in conjunction with a server. *See* Pinard, Col. 1, lines 60-62; Col. 2, lines 41-45. One of ordinary skill in the art would have recognized that the particular design choices reflected in the graphical user interface of Pinard could readily be implemented within the context of the Etherphone system. In fact, Etherphone discloses a graphical user interface with similar features to those described in Pinard.

(ix) Etherphone in view of Pinard and Further in View of VocalChat

Claim 18 and 29 require that "the element provided in step D represents a communication line on mute status" and "the communication line having a temporarily disabled status comprises a communication line on mute status," respectively.

¶ 196. As described above, Etherphone and Pinard describe all of the elements of Claim 18 and 29 except for a "communication line on mute status. However, VocalChat describes a "communication line on mute status." As described in the User's Guide, "Manual Activation can also be used like the MUTE option in many phones: it lets you talk without being heard on the other user's system." User's Guide, page 57.

(x) Motivation to Combine VocalChat with Etherphone and Pinard

¶ 197. A motivation to combine VocalChat with Etherphone and Pinard exists due to the problem to be solved. All three references relate to the field of telephony, and in particular to the use of computer system to implement telephony features. *See, e.g.,* Pinard, Col. 1, lines 5-7. One of ordinary skill in the art would have recognized the need for a "mute" function to enable users to mute the audio of a call as needed. In fact, the Etherphone system included a mute function, although it was not explicitly described in Etherphone. *See, e.g.*, Vin, page 73 ("Additional accelerators and features, such as manually toggling the Recv-only condition to mute the conversation, are available via mouse clicks on conversation log entries.").

C. VocalChat User's Guide in view of VocalChat Readme, and further in view of VocalChat Networking, and further in view of VocalChat Help File, and further in view of VocalChat Troubleshooting Help File

¶ 198. The following is a quotation of 35 U.S.C. §103 (a) which forms the basis for the following obviousness rejections:

> A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

¶ 199. Claims 1-2, 4, 7, 10-11, 19-22, 30-42 should be rejected under 35 U.S.C. § 103(a) as being unpatentable over VocalChat User's Guide, Version 2.0 (1994) ("User's Guide"), in view of VocalChat Readme File, Version 2.02 (June, 1994) ("Readme"), in view of VocalChat 1.01 Networking Information ("VocalChat Networking"), in view of VocalChat Information, Version 2.02 (July 18, 1994) ("Help File"), and further in view of VocalChat Troubleshooting Help File, Version 2.02 (July 18, 1994) ("Troubleshooting Help File"). These prior art publications are collectively referred to herein as "the VocalChat references" or "VocalChat."

¶ 200. As stated in the declaration of Alon Cohen, one of the co-founders of VocalTec, Ltd., included as **Exhibit L**: (1) VocalChat 1.01 Networking Information ("Networking Information"), attached as **Exhibit I** (referred to as Exhibit A in the declaration), was publicly distributed in 1994 as part of the VocalChat version 1.01 software, which was commercially

released and on sale to the general public in 1994; (2) VocalChat 2.0 User's Guide ("User's Guide"), attached as **Exhibit G** (referred to as Exhibit B in the declaration), was publicly distributed in 1994 as part of the VocalChat version 2.0 software, which was commercially released and on sale to the general public in 1994; and (3) The VocalChat Readme File ("Readme"), attached as **Exhibit H** (referred to as Exhibit C in the declaration), the VocalChat Troubleshooting Help File ("Troubleshooting Help File"), attached as **Exhibit K** (referred to as Exhibit D in the declaration), and VocalChat Information ("Help File"), attached as **Exhibit J** (referred to as Exhibit E in the declaration), are true and correct print-outs of VocalChat version 2.02's README.TXT, TROUBLE.HLP, and INFO.HLP files, respectively. In sum, electronic copies of all of these documents were publicly distributed in 1994 as part of the various VocalChat software releases.

1. Motivation to Combine the VocalChat References Under 35 U.S.C. § 103

¶ 201. A strong motivation to combine the VocalChat references under 35 U.S.C. § 103 exists because they all describe the same VocalChat system. The fact that some of the references describe different versions of the VocalChat system does not alter the fact that it would have been obvious to combine these references because all of the references share numerous common features (e.g., a central server to store addresses and VocalChat client software) which interoperate in the same basic manner.

¶ 202. During the Net2Phone Litigation, Net2Phone attempted to distinguish the claims of the '704 patent over the VocalChat References. The court has yet to render an opinion on these arguments. As set forth in **Exhibit R**, these arguments fail to distinguish the claims of the '704 patent over the combined VocalChat References for a variety of reasons.

¶ 203. These VocalChat references were not cited or discussed alone, or in combination, during the prosecution of the '704 patent. As delineated below, there is a SNQ of patentability raised by VocalChat. Below first the independent claims are set forth along with a discussion concerning the relevancy of VocalChat to the SNQ of patentability. Then the dependent claims are set forth.

INDEPENDENT CLAIM 1

The preamble of Claim 1 reads, in pertinent part: "A computer program product for use with a computer system, the computer system executing a first process and operatively connectable to a second process and a server over a computer network . . ."

¶ 204. VocalChat is a software-based telephone executed on personal computers which connects to a central server to locate other personal computers on a variety of computer networks, including TCP/IP, NetBIOS, and IPX networks. *See, e.g.*, VocalChat User's Guide, page 5 (illustrating a central server with a "post office" to enable communication between computers). *See also id.*, pages 7-8 (describing minimum personal computer requirements as a "386SX or higher IBM-compatible computer"); Readme, page 1 (listing the VocalChat files copied during installation); Help File, page 2 ("VocalChat can work with IPX, NetBIOS and TCP/IP network protocols.").

¶ 205. In the pending litigation, Net2Phone argued that the term "server" should be defined broadly. Plaintiff Net2Phone, Inc.'s Response Brief on Claim Construction (Oct. 18, 2007) (Exhibit U), page 3. More specifically, Net2Phone argued:

> Consistent with the use of the term 'server' in the specification, the claims do not refer to any specific server configuration. They simply require a 'server' (also referred to as a 'connection server,' 'address server,' or 'server process'). There is nothing in any of the claims that require that the server be in the form of a single computer with a centralized database, as defendants contend.

Id., page 4. Similarly, Net2Phone argued that "[a] server in a 'client/server system' can be implemented in any number of ways, from one to multiple computers, in one location or many, and from a single large computer acting as the server to a network of personal computers." Plaintiff Net2Phone Inc.'s Reply Brief on Claim Construction (Oct. 19, 2007) (Exhibit W), page 7 (citing to the declaration of Professor Larry L. Peterson). Thus, under Net2Phone's interpretation, a "server" is not limited to any particular hardware or software configuration.

¶ 206. It should be noted, however, that the requestor of the present Reexamination does not agree with this interpretation, and has stated as such in the pending litigation. *See, e.g.*, Reply Claim Construction Brief of Skype Technologies SA, Skype, Inc. and EBay Inc (Oct. 19, 2007) (Exhibit X), pages 2-9. For the sake of brevity, these interpretations are not repeated

below with respect to the other claims of the '704 patent which require a "server." Under any interpretation, the "post office" server in VocalChat is a "server."

Claim 1 requires "a computer usable medium having program code embodied in the medium ..."

¶ 207. As software, VocalChat is inherently stored as program code on a computer-usable medium. *See, e.g.*, Readme, page 1 (listing the VocalChat files copied during installation). *See also* VocalChat User's Guide, page 8 (describing how VocalChat is installed by inserting "the VocalChat Disk in drive A").

Claim 1 also requires "program code for transmitting to the server a network protocol address received by the first process following connection to the computer network."

¶ 208. As illustrated in the figure on page 5 of the VocalChat User's Guide (reproduced above), computers with VocalChat installed connect directly to a server to register their current network protocol addresses. In the initial VocalChat implementations (versions 1.x) each VocalChat client transmits its name and network protocol address to a USERS file stored on the server. As described in VocalChat Network Information:

> When the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users.
>
> Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file. Thus other users can add this user's name as a Quick Dial button even if the user is not running VocalChat at the moment. However, in order for VocalChat to work properly, *all users must have access to the same USERS file, and all must have read/write access to that file.*

VocalChat Network Information, page 10 (underline emphasis added); Troubleshooting Help File, page 28 ("VocalChat needs the TCP/IP software to recognize your own computers host name and IP address."). Later VocalChat implementations (e.g., version 2.02), referred to the USERS file as a "Connection List" stored within a 'Post-Office' directory. *See, e.g.*, Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory.").

¶ 209. Regardless of the file name, the Connection List/USERS file is stored on a server for access by VocalChat clients. *See, e.g.*, VocalChat Network Information, page 2 ("**Server**

Installation is used to install the VocalChat program files on the network, for use by the different network users."). *See also* Readme File, page 2 ("VocalChat creates a central directory on the network, shared by all users called 'Post-Office.' All users must use the same Post-Office, otherwise they won't be able to communicate or leave messages to each other. This means that all users must be attached to one file-server which will be used for the Post-Office, and all have write permission for the Post-Office directory."); Help File, page 8 ("the Setup program crates a Connection List File which is used to identify and access users").

¶ 210. On many networks, including TCP/IP networks, network addresses are assigned "following connection to the computer network." *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). Thus, in at least some instances, a computer system executing VocalChat receives its IP address following connection to the computer network. Consequently, dynamic address assignment is inherent in the VocalChat system. Alternatively, as set forth below, it would have been obvious to one of ordinary skill in the art to use dynamic address assignment on a TCP/IP network.

Claim 1 further requires "program code for transmitting, to the server, a query as to whether the second process is connected to the computer network."

¶ 211. VocalChat employs different techniques for locating users based on the underlying network protocol. As described in the Help File:

> Method of determining users address:
> Netware — Get Users information from Netware 2.x/3.x bindery
> WinWorkgroups — Get users information from Windows for Workgroups.
> Generic User — VocalChats files for users information. (See Generic network, below).

Help File, page 26. With any protocol other than Netware or Windows for Workgroups (such as TCP/IP or NetBIOS), a "generic" method is used where the VocalChat client queries VocalChat files (the Connection List/USERS files) locating users on the network. As described in greater detail in the Help File:

> When **NetBIOS** or **IPX** are used, but not with NetWare or Window for Workgroups, or when TCP/IP is used, a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory. In this case, the user name for each user, is

> entered when performing the **User Installation** in the Setup program. You should make sure that this name is not used by any other user on the network.

Help File, page 2. *See also* Help File, page 22 (VocalChat "will use the CONNLIST.VC files to get network addresses"); page 8 ("the Setup program creates a Connection List file which is used to identify and access users"); page 2 ("When working with the IPX and TCP/IP protocols, the network addresses of the different workstations are required for VocalChat to be able to access the network users."). With NetWare, the VocalChat client queries existing NetWare Bindery services locating "currently logged-in users;" with Windows for Workgroups, the VocalChat client queries the Windows for Workgroups services locating online users; and with other protocols, such as TCP/IP and NetBIOS, the VocalChat client queries the shared Connection List file (CONNLIST.VC). Regardless of protocol, the query determines whether the second process (the VocalChat client of another user) is connected to the computer network. For example, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file." VocalChat Network Information, page 10. Thus, a distinction is made between logged in users and logged out users. Similarly, as described above, in the NetWare implementation, the query retrieves a list of "currently logged in users."

Claim 1 also requires "program code for receiving a network protocol address of the second process from the server, when the second process is connected to the computer network."

¶ 212. When TCP/IP is used, "the network addresses of the different workstations are required for VocalChat to be able to access the network users." Help File, page 2. Moreover, as described above, with TCP/IP, "a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses." The ability to establish a call merely by identifying a unique user name demonstrates that VocalChat clients "received" the network addresses of VocalChat callees from the server's directory database. Indeed, the directory database is used to "get . . . the addresses of specific users." *Id.*; *see also id.*, page 22 ("VocalChat will use the CONNLIST.VC files to get network addresses").

¶ 213. In Claim Construction Briefs filed in the pending litigation, the patentee argued

that the term

> 'connected' means 'logged on,' and *vice versa* . . . To the extent defendants are trying to suggest that the claims require perfect information about who is on line at a given moment, that is simply incorrect. While Net2Phone's invention endeavors to identify accurately who is on line, it is not possible to achieve perfection. For example, it takes some time (albeit minimal) for the signal that a user has gone off-line to be communicated to the server, or a user's Internet connection may get interrupted before she can send an off-line message (and thus the server, for a time, assumes she is on-line, when in fact she is not). *See* Strickland Dep. at 140:7-141:7 (Ex. 21). Recognizing these issues, the patents explain that the server may use timestamps to update a person's status— *e.g.*, setting a default value of two hours, after which the server assumes that a party has gone off-line if it has not heard from her. *See* '704 patent, col. 5, ll. 39-44 (Ex. 2). In this respect, the patents explain, "the on-line status information stored in the database is ***relatively*** *current*." *Id.* at col. 5, ll. 42-43 (emphasis added). While Net2Phone believes that the claim language is clear, if the term "connected" (or "on-line") is going to be modified at all, it should be modified to say "*relatively* currently connected," because that is what the patents actually say.

Plaintiff Net2Phone Inc.'s Response Brief on Claim Construction (Oct. 18, 2007) (Exhibit U), pages 24-25. Thus, under Net2Phone's interpretation, the information retained in the "server" as to which processes are "connected to the computer network" or "online" may be imperfect. As described above, while the server "endeavors to identify accurately who is on line, it is not possible to achieve perfection." *Id.* As described above, VocalChat employs similar techniques as address entries for off-line VocalChat processes are removed through the use of log-out messages.

¶ 214. Once again, the requestor of the present reexamination does not agree with this interpretation, and has stated as such in the pending litigation. *See, e.g.*, Reply Claim Construction Brief of Skype Technologies SA, Skype, Inc. and EBay Inc (Exhibit X), pages 12-14. For the sake of brevity, these interpretations are not repeated below with respect to the other claims of the '704 patent which require a process to be "connected to" the computer network or "on-line." Under any interpretation, a first VocalChat process receives the network protocol address of a second VocalChat process from the "post office" server when the second VocalChat process is "connected to the computer network."

Claim 1 also requires "program code, responsive to the network protocol address of the second process, for establishing a point-to-point

communication link between the first process and the second process over the computer network."

¶ 215. VocalChat discloses that "[u]ser-to-user access is facilitated automatically through the [Connection List] file." Help File, page 17. VocalChat also discloses "the peer-to-peer nature of Windows for Workgroups, which VocalChat "uses . . . for user services." *Id.* In fact, VocalChat is a voice over computer network product for use on various networks that "enables communication between" VocalChat users. *Id.* at 8. *See also* User Guide, page 2 ("Talk with other users over the network, and broadcast to network users or groups. Access network users with the Address Book and Quick-Dial buttons.").

INDEPENDENT CLAIM 2

Claim 2 claims "[a]n apparatus for enabling point-to-point communications between a first and a second process over a computer network, the apparatus comprising: a processor; a network interface, operatively coupled to the processor, for connecting the apparatus to the computer network."

¶ 216. VocalChat discloses the preamble of claim 2. For example, VocalChat is a software-based telephone executed on personal computers which connects to a central server to locate other personal computers on a variety of computer networks, including TCP/IP, NetBIOS, and IPX networks. *See, e.g.*, VocalChat User's Guide, page 5 (illustrating a central server with a "post office" to enable communication between computers). *See also id.*, pages 7-8 (describing minimum personal computer requirements as a "386SX or higher IBM-compatible computer"); Readme, page 1 (listing the VocalChat files copied during installation); Help File, page 2 ("VocalChat can work with IPX, NetBIOS and TCP/IP network protocols."). Inherently, the personal computers and the server included a processor, for processing program code, and a network interface connected to the network. The VocalChat software installed on each computer system comprises a computer-implemented "process."

Claim 2 requires "a memory, operatively coupled to the processor, for storing a network protocol address for selected of a plurality of processes, each network protocol address stored in the memory following connection of a respective process to the computer network."

¶ 217. Computer systems with VocalChat installed and the central server inherently included "a memory, operatively coupled to the processor" in the form of a RAM and a hard drive. *See, e.g.*, User Guide, page 7 (describing minimum system requirements of 4 MB RAM).

In the initial VocalChat implementations (versions 1.x) each VocalChat client transmits its name and network protocol address to a USERS file stored on a server. As described in VocalChat Network Information:

> When the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users.
>
> Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file. Thus other users can add this user's name as a Quick Dial button even if the user is not running VocalChat at the moment. However, in order for VocalChat to work properly, *all users must have access to the same USERS file, and all must have read/write access to that file.*

VocalChat Network Information, page 10 (underline emphasis added); Troubleshooting Help File, page 28 ("VocalChat needs the TCP/IP software to recognize your own computers host name and IP address."). Later VocalChat implementations (e.g., version 2.02), referred to the USERS file as a "Connection List" file. *See, e.g.*, Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory."). Regardless of the file name, the Connection List/USERS file is stored on a server for access by VocalChat clients. *See, e.g.*, VocalChat Network Information, page 2 ("**Server Installation** is used to install the VocalChat program files on the network, for use by the different network users."). *See also* Readme File, page 2 ("VocalChat creates a central directory on the network, shared by all users called 'Post-Office.' All users must use the same Post-Office, otherwise they won't be able to communicate or leave messages to each other. This means that all users must be attached to one file-server which will be used for the Post-Office, and all have write permission for the Post-Office directory."); Help File, page 8("the Setup program crates a Connection List File which is used to identify and access users").

¶ 218. On many networks, including TCP/IP networks, network addresses are assigned "following connection to the computer network." *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). Thus, in at least some instances, a computer system executing VocalChat receives its IP address following connection to the computer network. Consequently, dynamic address assignment is inherent in the VocalChat system. Alternatively,

as set forth below, it would have been obvious to one of ordinary skill in the art to use dynamic address assignment on a TCP/IP network in combination with VocalChat.

> **Claim 2 also requires "means, responsive to a query from the first process, for determining the on-line status of the second process and for transmitting a network protocol address of the second process to the first process in response to a positive determination of the on-line status of the second process."**

¶ 219. In a TCP/IP implementation, the server on which the Connection List/USERS file is located transmits the network protocol address of a second VocalChat client (second process) to a first VocalChat client (first process) upon request. As described in the Help File:

> Method of determining users address:
> Netware — Get Users information from Netware 2.x/3.x bindery
> WinWorkgroups — Get users information from Windows for Workgroups.
> Generic User — VocalChats files for users information. (See Generic network, below).

Help File, page 26. With any protocol other than Netware or Windows for Workgroups (such as TCP/IP or NetBIOS), a "generic" method is used where the VocalChat client queries VocalChat files (the Connection List/USERS files) locating users on the network. As described in greater detail in the Help File:

> When **NetBIOS** or **IPX** are used, but not with NetWare or Window for Workgroups, or when TCP/IP is used, a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory. In this case, the user name for each user, is entered when performing the **User Installation** in the Setup program. You should make sure that this name is not used by any other user on the network.

Help File, page 2. With NetWare, the VocalChat client queries existing NetWare Bindery services locating "currently logged-in users;" with Windows for Workgroups, the VocalChat client queries the Windows for Workgroups services locating online users; and with other protocols, such as TCP/IP and NetBIOS, the VocalChat client queries the shared Connection List file (CONNLIST.VC). Regardless of protocol, the query determines whether the second process (the VocalChat client of another user) is connected to the computer network. For example, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its

entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file." VocalChat Network Information, page 10. Thus, a distinction is made between logged in users and logged out users. Similarly, as described above, in the NetWare implementation, the query retrieves a list of "currently logged in users."

INDEPENDENT CLAIM 4

The preamble of Claim 4 reads, in pertinent part: "A method for enabling point-to-point communication between a first process and a second process over a computer network."

¶ 220. As mentioned above, VocalChat discloses that "[u]ser-to-user access is facilitated automatically through the [Connection List] file." Help File, page 17. VocalChat also discloses "the peer-to-peer nature of Windows for Workgroups, which VocalChat "uses . . . for user services." *Id.* In fact, VocalChat is a voice over computer network product for use on various networks that "enables communication between" VocalChat users. *Id.* at 8. *See also* User Guide, page 2 ("Talk with other users over the network, and broadcast to network users or groups. Access network users with the Address Book and Quick-Dial buttons.").

Claim 4 requires "receiving and storing into a computer memory a respective network protocol address for selected of a plurality of processes that have an on-line status with respect to the computer network, each of the network protocol addresses received following connection of the respective process to the computer network."

¶ 221. As described in VocalChat Network Information:

> When the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users.
>
> Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file. Thus other users can add this user's name as a Quick Dial button even if the user is not running VocalChat at the moment. However, in order for VocalChat to work properly, *all users must have access to the same USERS file, and all must have read/write access to that file.*

VocalChat Network Information, page 10 (underline emphasis added); *see also* Troubleshooting Help File, page 28 ("VocalChat needs the TCP/IP software to recognize your own computers host name and IP address."). Later VocalChat implementations (e.g., version 2.02), referred to the USERS file as a "Connection List" file. *See, e.g.*, Help File, page 2 ("a shared

CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory."). Regardless of the file name, the Connection List/USERS file is stored on a server for access by VocalChat clients. *See, e.g.*, VocalChat Network Information, page 2 ("**Server Installation** is used to install the VocalChat program files on the network, for use by the different network users."). *See also* Readme File, page 2 ("VocalChat creates a central directory on the network, shared by all users called 'Post-Office.' All users must use the same Post-Office, otherwise they won't be able to communicate or leave messages to each other. This means that all users must be attached to one file-server which will be used for the Post-Office, and all have write permission for the Post-Office directory."); Help File, page 8 ("the Setup program crates a Connection List File which is used to identify and access users").

¶ 222. On many networks, including TCP/IP networks, network addresses are assigned "following connection to the computer network." *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). Thus, in at least some instances, a computer system executing VocalChat receives its IP address following connection to the computer network. Consequently, dynamic address assignment is inherent in the VocalChat system. Alternatively, as set forth below, it would have been obvious to one of ordinary skill in the art to use dynamic address assignment on a TCP/IP network in combination with VocalChat.

Claim 4 also requires "receiving a query from the first process to determine the on-line status of the second process."

¶ 223. In a TCP/IP implementation, the server on which the Connection List/USERS file was located received queries from VocalChat clients (first processes) to determine the on-line status of other VocalChat clients (second processes). As described in the Help File:

> Method of determining users address:
> Netware — Get Users information from Netware 2.x/3.x bindery
> WinWorkgroups — Get users information from Windows for Workgroups.
> Generic User — VocalChats files for users information. (See Generic network, below).

Help File, page 26. With any protocol other than Netware or Windows for Workgroups (such as TCP/IP or NetBIOS), a "generic" method is used where the VocalChat client queries VocalChat files (the Connection List/USERS files) locating users on the network. As described in greater detail in the Help File:

> When **NetBIOS** or **IPX** are used, but not with NetWare or Window for Workgroups, or when TCP/IP is used, a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory. In this case, the user name for each user, is entered when performing the **User Installation** in the Setup program. You should make sure that this name is not used by any other user on the network.

Help File, page 2. With NetWare, the VocalChat client queries existing NetWare Bindery services locating "currently logged-in users;" with Windows for Workgroups, the VocalChat client queries the Windows for Workgroups services locating online users; and with other protocols, such as TCP/IP and NetBIOS, the VocalChat client queries the shared Connection List file (CONNLIST.VC). Regardless of protocol, the query determines whether the second process (the VocalChat client of another user) is connected to the computer network. For example, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file." VocalChat Network Information, page 10. Thus, a distinction is made between logged in users and logged out users. Similarly, as described above, in the NetWare implementation, the query retrieves a list of "currently logged in users."

Claim 4 next requires "determining the on-line status of the second process."

¶ 224. As described with respect to the previous claim element, in a TCP/IP implementation, the server on which the Connection List/USERS file is located transmits the network protocol address of a second VocalChat client (second process) to a first VocalChat client (first process) upon request. As described in the Help File:

> Method of determining users address:
>
Netware	Get Users information from Netware 2.x/3.x bindery
WinWorkgroups	Get users information from Windows for Workgroups.
Generic User	VocalChats files for users information. (See Generic network, below).

Help File, page 26. With any protocol other than Netware or Windows for Workgroups (such as TCP/IP or NetBIOS), a "generic" method is used where the VocalChat client queries VocalChat files (the Connection List/USERS files) determining the on-line status of other users on the network. As described in greater detail in the Help File:

> When **NetBIOS** or **IPX** are used, but not with NetWare or Window for Workgroups, or when TCP/IP is used, a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory. In this case, the user name for each user, is entered when performing the **User Installation** in the Setup program. You should make sure that this name is not used by any other user on the network.

Help File, page 2. With NetWare, the VocalChat client queries existing NetWare Bindery services locating "currently logged-in users;" with Windows for Workgroups, the VocalChat client queries the Windows for Workgroups services locating online users; and with other protocols, such as TCP/IP and NetBIOS, the VocalChat client queries the shared Connection List file (CONNLIST.VC). Additionally, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file." VocalChat Network Information, page 10 (emphasis added). Thus, a distinction is made between logged in users and logged out users. Similarly, as described above, in the NetWare implementation, the query retrieves a list of "currently logged in users."

Claim 4 finally requires "transmitting an indication of the on-line status of the second process to the first process over the computer network."

¶ 225. As described above, with TCP/IP, "the network addresses of the different workstations are required for VocalChat to be able to access the network users." Help File, page 2. Moreover, with TCP/IP, "a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses." The ability to establish a call merely by identifying a unique user name demonstrates that the server "transmits" the network addresses of VocalChat callees from the server's directory database. Indeed, the directory database is used to "get . . . the addresses of specific users." *Id.*; *see also id.*, page 22 ("VocalChat will use the CONNLIST.VC files to get network addresses").

INDEPENDENT CLAIM 10

The preamble of Claim 10 reads, in pertinent part: "In a computer system, a method for establishing a point-to-point communication link from a caller process to a callee process over a computer network, the caller process having a user interface and being operatively connectable to the callee process and a server over the computer network."

¶ 226. As discussed above, VocalChat clients connect to a server to locate and establish point-to-point connections with other VocalChat clients over a network. For example, VocalChat discloses that "[u]ser-to-user access is facilitated automatically through the [Connection List] file" which is stored on a server. Help File, page 17. VocalChat also discloses "the peer-to-peer nature of Windows for Workgroups, which VocalChat "uses . . . for user services." *Id.* In fact, VocalChat is a voice over computer network product for use on various networks that "enables communication between" VocalChat users. *Id.* at 8. *See also* User Guide, page 2 ("Talk with other users over the network, and broadcast to network users or groups. Access network users with the Address Book and Quick-Dial buttons.").

Claim 10 also requires "providing a user interface element representing a first communication line."

¶ 227. A VocalChat user makes a point-to-point call to another user by using the VocalChat "Call" button, which is a user interface element representing a first communication line. *See, e.g.*, User Guide, page 14 ("Select Call from the Chat menu, or click on the tool bar Call button"). In addition, the VocalChat graphical user interface (GUI) includes a plurality of Quick Dial buttons. *See* User Guide, page 12. Depending on the implementation, either the Call button or the Quick Dial button comprises an "element representing a first communication line."

Claim 10 further requires "providing a user interface element representing a first callee process."

¶ 228. The VocalChat GUI displayed the names of potential callees in a dialog box. *See, e.g.*, Help File, page 14 ("just select a user from the user list, and choose "OK"). Callees are also represented as Quick Dial buttons. *See* Help File, pages 11, 20-21 ("Setting a Quick Dial Button"). Depending on the implementation, either the callee names listed within the dialog box or the Quick Dial buttons comprise "a user interface element representing a first callee process."

Claim 10 also requires "establishing a point-to-point communication link from the caller process to the first callee process, in response to a user associating the element representing the first callee process with the element representing the first communication line."

¶ 229. As mentioned above, a VocalChat user makes a point-to-point call to another user with the Call button or a Quick Dial Button representing a frequently called callee. *See* Help File, page 14 (describing use of the Call button) and 20 (describing use of the Quick Dial buttons). Selecting the Call button opens a dialog box displaying a list of connected VocalChat users. A caller then clicks on a user's name in the list and then clicks the OK button to establish a point-to-point communication link. *See, e.g.,* Help File, page 14. In this example, the graphical representation of the user in the list is an "element representing the first callee process" and the OK button is an "element representing a first communication line." Alternatively, a user can associate any VocalChat user with a Quick Dial button by right-clicking on a Quick Dial button, which presents the user with the VocalChat users list. *See* Help File, page 20. After the user selects a user name from the list, that user is associated with the quick dial button. *See* Help File, page 21 ("From the user list, choose the user name that you want the button to hold."). The caller then places a call to the callee by selecting the Quick Dial button. VocalChat also assigns Quick Dial buttons automatically ("When you call a user with the Call command, a vacant button changes to hold the user's name if one does not hold it already."). In these examples, the graphical representation of the user in the list is an "element representing the first callee process" and the quick dial button is an "element representing a first communication line." In both cases, the element representing the callee process is associated with an element representing a communication line.

INDEPENDENT CLAIM 21

Claim 21 reads, in pertinent part: "A computer program product for use with a computer system comprising."

¶ 230. The techniques described in VocalChat are implemented in software, which is a "computer program product." In particular, VocalChat is a software-based telephone executed on personal computers which connects to a central server to locate other personal computers on a variety of computer networks, including TCP/IP, NetBIOS, and IPX networks. *See, e.g.*, VocalChat User's Guide, page 5 (illustrating a central server with a "post office" to enable communication between computers). *See also id.*, pages 7-8 (describing minimum personal

computer requirements as a "386SX or higher IBM-compatible computer"); Readme, page 1 (listing the VocalChat files copied during installation); Help File, page 2("VocalChat can work with IPX, NetBIOS and TCP/IP network protocols.").

Claim 21 requires "a computer usable medium having program code embodied in the medium for establishing a point-to-point communication link from a caller process to a callee process over a computer network, the caller process having a user interface and being operatively connectable to the callee process and a server over the computer network."

¶ 231. As software, VocalChat is inherently stored as program code on a computer-usable medium. *See, e.g.*, Readme, page 1 (listing the VocalChat files copied during installation). *See also* VocalChat User's Guide, page 8 (describing how VocalChat is installed by inserting "the VocalChat Disk in drive A"). As discussed above, VocalChat clients connect to a server to locate and establish point-to-point connections with other VocalChat clients over a network. For example, VocalChat discloses that "[u]ser-to-user access is facilitated automatically through the [Connection List] file" which is stored on a server. Help File, page 17. VocalChat also discloses "the peer-to-peer nature of Windows for Workgroups, which VocalChat "uses . . . for user services." *Id.* In fact, VocalChat is a voice over computer network product for use on various networks that "enables communication between" VocalChat users. *Id.* at 8. *See also* User Guide, page 2 ("Talk with other users over the network, and broadcast to network users or groups. Access network users with the Address Book and Quick-Dial buttons.").

Claim 21 also requires "program code for generating an element representing a first communication line."

¶ 232. A VocalChat user makes a point-to-point call to another user by using the VocalChat "Call" button, which is a user interface element representing a first communication line. *See, e.g.*, User Guide, page 14 ("Select Call from the Chat menu, or click on the tool bar Call button"). In addition, the VocalChat graphical user interface (GUI) included a plurality of Quick Dial buttons. *See* User Guide, page 12. Depending on the implementation, either the Call button or the Quick Dial button comprises an "element representing a first communication line."

Claim 21 also requires "program code for generating an element representing a first callee process."

¶ 233. The VocalChat GUI displayed the names of potential callees in a dialog box. *See, e.g.*, Help File, page 14 ("just select a user from the user list, and choose "OK"). Callees were also represented as Quick Dial buttons. *See* Help File, pages 11, 20-21 ("Setting a Quick Dial Button"). Depending on the implementation, either the callee names listed within the dialog box or the Quick Dial buttons comprise "a user interface element representing a first callee process."

Claim 21 also requires "program code, responsive to a user associating the element representing the first callee process with the element representing the first communication line, for establishing a point-to-point communication link from the caller process to the first callee process."

¶ 234. As mentioned above, a VocalChat user makes a point-to-point call to another user with the Call button or a Quick Dial Button representing a frequently called callee. *See* Help File, page 14 (describing use of the Call button) and 20 (describing use of the Quick Dial buttons). Selecting the Call button opens a dialog box which displays a list of connected VocalChat users. A caller then clicks on a user's name in the list and then clicks the OK button to establish a point-to-point communication link. *See*, *e.g.,* Help File, page 14. In this example, the graphical representation of the user in the list is an "element representing the first callee process" and the OK button is an "element representing a first communication line." Alternatively, a user can associate any VocalChat user with a Quick Dial button by right-clicking on a Quick Dial button, which presents the user with the VocalChat users list. *See* Help File, page 20. Once the user selects a user name from the list, that user was associated with the quick dial button. *See* Help File, page 21 ("From the user list, choose the user name that you want the button to hold."). The caller may then place a call to the callee by selecting the Quick Dial button. VocalChat also assigns Quick Dial buttons automatically ("When you call a user with the Call command, a vacant button changes to hold the user's name if one does not hold it already."). In these examples, the graphical representation of the user in the list is an "element representing the first callee process" and the quick dial button is an "element representing a first communication line." In both cases, the element representing the callee process is associated with an element representing a communication line.

INDEPENDENT CLAIM 32

Claim 32 recites: "A method of locating a process over a computer network comprising the steps of: maintaining an Internet accessible list having a plurality of selected entries, each entry comprising an identifier and a corresponding Internet protocol address of a process currently connected to the Internet, the Internet Protocol address added to the list following connection of the process to the computer network."

¶ 235. In the initial VocalChat implementations (versions 1.x) each VocalChat client transmits its name and network protocol address to a USERS file stored on a server. As described in VocalChat Network Information:

> When the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users.
>
> Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file. Thus other users can add this user's name as a Quick Dial button even if the user is not running VocalChat at the moment. However, in order for VocalChat to work properly, *all users must have access to the same USERS file, and all must have read/write access to that file.*

VocalChat Network Information, page 10 (underline emphasis added); Troubleshooting Help File, page 28 ("VocalChat needs the TCP/IP software to recognize your own computers host name and IP address."). Later VocalChat implementations (e.g., version 2.02), refer to the USERS file as a "Connection List" file. *See, e.g.*, Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory."). Regardless of the file name, the Connection List/USERS file is stored on a server for access by VocalChat clients. *See, e.g.*, VocalChat Network Information, page 2 ("**Server Installation** is used to install the VocalChat program files on the network, for use by the different network users."). *See also* Readme File, page 2 ("VocalChat creates a central directory on the network, shared by all users called 'Post-Office.' All users must use the same Post-Office, otherwise they won't be able to communicate or leave messages to each other. This means that all users must be attached to one file-server which will be used for the Post-Office, and all have write permission for the Post-Office directory."); Help File, page 8 ("the Setup program crates a Connection List File which is used to identify and access users").

¶ 236. While VocalChat does not explicitly describe a server with stored names and addresses is accessible over "the Internet," it describes the use of TCP/IP, which is the protocol used on the Internet. Thus, VocalChat inherently describes that the list of users and network addresses is accessible over the Internet. Moreover, the Internet is a type of Wide Area Network (WAN) and VocalChat describes a WAN implementation. For example, the Help File describes that "[o]ver a WAN . . . it is advisable to create local copy of the executables and DLLs, and reference only the Post Office over the low-speed [WAN] connection." Help File, page 4.

¶ 237. On many networks, including TCP/IP networks, network addresses are assigned "following connection to the computer network." *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). Thus, in at least some instances, a computer system executing VocalChat receives its IP address following connection to the computer network. Consequently, dynamic address assignment is inherent in the VocalChat system. Alternatively, as set forth below, it would have been obvious to one of ordinary skill in the art to use dynamic address assignment on a TCP/IP network.

Claim 32 further requires "in response to identification of one of the list entries by a requesting process, providing one of the identifier and the corresponding Internet protocol address of the identified entry to the requesting process."

¶ 238. As discussed above, a VocalChat caller sends the directory server a query identifying a particular callee. *See, e.g.*, Help File, page 22 (VocalChat "will use the CONNLIST.VC files to get network addresses"); page 8 ("the Setup program creates a Connection List file which is used to identify and access users"). Consequently, the server identifies an entry in the directory corresponding to the identified callee (the Connection List file in a TCP/IP implementation), and, if the callee is connected, provides the corresponding IP address associated with that callee in the directory to the caller.

INDEPENDENT CLAIM 33

The preamble of Claim 33 reads, in pertinent part: "A method for locating processes having dynamically assigned network protocol addresses over a computer network."

¶ 239. As described above, VocalChat clients rely on a central server to locate the network addresses of other VocalChat clients. As described in VocalChat Network Information:

> When the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users.
>
> Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file. Thus other users can add this user's name as a Quick Dial button even if the user is not running VocalChat at the moment. However, in order for VocalChat to work properly, *all users must have access to the same USERS file, and all must have read/write access to that file.*

VocalChat Network Information, page 10 (underline emphasis added); Troubleshooting Help File, page 28 ("VocalChat needs the TCP/IP software to recognize your own computers host name and IP address."). Later VocalChat implementations (e.g., version 2.02), referred to the USERS file as a "Connection List" file. *See, e.g.*, Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory."). Regardless of the file name, the Connection List/USERS file is stored on a server for access by VocalChat clients. *See, e.g.*, VocalChat Network Information, page 2 ("**Server Installation** is used to install the VocalChat program files on the network, for use by the different network users."). *See also* Readme File, page 2 ("VocalChat creates a central directory on the network, shared by all users called 'Post-Office.' All users must use the same Post-Office, otherwise they won't be able to communicate or leave messages to each other. This means that all users must be attached to one file-server which will be used for the Post-Office, and all have write permission for the Post-Office directory."); Help File, page 8 ("the Setup program crates a Connection List File which is used to identify and access users").

¶ 240. On many networks, including TCP/IP networks, network addresses are assigned dynamically "following connection to the computer network." *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). Thus, in at least some instances, a computer system executing VocalChat receives its IP address following connection

to the computer network. Consequently, dynamic address assignment is inherent in the VocalChat system. Alternatively, as set forth below, it would have been obvious to one of ordinary skill in the art to use dynamic address assignment on a TCP/IP network.

Claim 33 also requires "maintaining, in a computer memory, a network accessible compilation of entries, selected of the entries comprising a network protocol address and a corresponding identifier of a process connected to the computer network."

¶ 241. The server on which the Connection List/USERS file is maintained inherently includes a computer memory. Moreover, the server stores a network accessible compilation of entries including a network protocol address and a name (corresponding identifier) of a process connected to the computer network. As described in VocalChat Network Information:

> When the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users.
>
> Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file. Thus other users can add this user's name as a Quick Dial button even if the user is not running VocalChat at the moment. However, in order for VocalChat to work properly, *all users must have access to the same USERS file, and all must have read/write access to that file.*

VocalChat Network Information, page 10 (underline emphasis added); Troubleshooting Help File, page 28 ("VocalChat needs the TCP/IP software to recognize your own computers host name and IP address."). Later VocalChat implementations (e.g., version 2.02), referred to the USERS file as a "Connection List" file. *See, e.g.*, Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory."). Regardless of the file name, the Connection List/USERS file is stored on a server for access by VocalChat clients. *See, e.g.*, VocalChat Network Information, page 2 ("**Server Installation** is used to install the VocalChat program files on the network, for use by the different network users."). *See also* Readme File, page 2 ("VocalChat creates a central directory on the network, shared by all users called 'Post-Office.' All users must use the same Post-Office, otherwise they won't be able to communicate or leave messages to each other. This means that all users must be attached to one file-server which will be used for the Post-Office, and all have write permission for the Post-Office directory."); Help

File, page 8 ("the Setup program crates a Connection List File which is used to identify and access users").

Claim 33 also requires that "in response to identification of one of the entries by a requesting process providing one of the identifier and the network protocol address to the requesting process."

¶ 242. As discussed above, a VocalChat caller sends the directory server a query identifying a particular callee. *See, e.g.*, Help File, page 22 (VocalChat "will use the CONNLIST.VC files to get network addresses"); page 8 ("the Setup program creates a Connection List file which is used to identify and access users"). Consequently, the server identifies an entry in the directory corresponding to the identified callee (the Connection List file in a TCP/IP implementation), and, if the callee is connected, provides the corresponding IP address that is associated with that callee in the directory to the caller.

INDEPENDENT CLAIM 38

Claim 38 recites, in pertinent part: "A computer program product for use with a computer system having a memory and being operatively connectable over a computer network to one or more computer processes, the computer program product comprising a computer usable medium having program code embodied in the medium the program code."

¶ 243. VocalChat is a software-based telephone executed on personal computers which connects to a central server to locate other personal computers on a variety of computer networks, including TCP/IP, NetBIOS, and IPX networks. *See, e.g.*, VocalChat User's Guide, page 5 (illustrating a central server with a "post office" to enable communication between computers). *See also id.*, pages 7-8 (describing minimum personal computer requirements as a "386SX or higher IBM-compatible computer"); Help File, page 2 ("VocalChat can work with IPX, NetBIOS and TCP/IP network protocols."). As software, VocalChat is inherently stored as program code on a computer-usable medium. *See, e.g.*, Readme, page 1 (listing the VocalChat files copied during installation). *See also* VocalChat User's Guide, page 8 (describing how VocalChat is installed by inserting "the VocalChat Disk in drive A").

Claim 38 requires "program code configured to maintain, in the computer memory, a network accessible compilation of entries, selected of the entries comprising a network protocol address and a corresponding identifier of a process connected to the computer network, the network protocol address of the corresponding process assigned to the process upon connection to the computer network."

¶ 244. The server on which the Connection List/USERS file is maintained inherently includes a computer memory. Moreover, the server stores a network accessible compilation of entries including a network protocol address and a name (corresponding identifier) of a process connected to the computer network. As described in VocalChat Network Information:

> When the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users.
>
> Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file. Thus other users can add this user's name as a Quick Dial button even if the user is not running VocalChat at the moment. However, in order for VocalChat to work properly, *all users must have access to the same USERS file, and all must have read/write access to that file.*

VocalChat Network Information, page 10 (underline emphasis added); Troubleshooting Help File, page 28 ("VocalChat needs the TCP/IP software to recognize your own computers host name and IP address."). Later VocalChat implementations (e.g., version 2.02), refers to the USERS file as a "Connection List" file. *See, e.g.*, Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory."). Regardless of the file name, the Connection List/USERS file is stored on a server for access by VocalChat clients. *See, e.g.*, VocalChat Network Information, page 2 ("**Server Installation** is used to install the VocalChat program files on the network, for use by the different network users."). *See also* Readme File, page 2 ("VocalChat creates a central directory on the network, shared by all users called 'Post-Office.' All users must use the same Post-Office, otherwise they won't be able to communicate or leave messages to each other. This means that all users must be attached to one file-server which will be used for the Post-Office, and all have write permission for the Post-Office directory."); Help File, page 8 ("the Setup program crates a Connection List File which is used to identify and access users").

¶ 245. On many networks, including TCP/IP networks, network addresses are assigned dynamically "following connection to the computer network." *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). Thus, in at least some instances, a computer system executing VocalChat receives its IP address following connection to the computer network. Consequently, dynamic address assignment is inherent in the VocalChat system. Alternatively, as set forth below, it would have been obvious to one of ordinary skill in the art to use dynamic address assignment on a TCP/IP network.

Claim 38 also requires "program code responsive to identification of one of the entries by a requesting process and configured to provide one of the identifier and the network protocol address to the requesting process."

¶ 246. As discussed above, a VocalChat caller sends the directory server a query identifying a particular callee. *See, e.g.*, Help File, page 22 (VocalChat "will use the CONNLIST.VC files to get network addresses"); page 8 ("the Setup program creates a Connection List file which is used to identify and access users"). Consequently, the server identifies an entry in the directory corresponding to the identified callee (the Connection List file in a TCP/IP implementation), and, if the callee is connected, provides the corresponding IP address that is associated with that callee in the directory to the caller.

INDEPENDENT CLAIM 43

Claim 43 recites, in pertinent part: "A computer program product for use with a computer system, the computer system executing a first process operatively coupled over a computer network to a second process and a server process, the computer program product comprising a computer usable medium having computer readable program code embodied therein."

¶ 247. VocalChat is a software-based telephone executed on personal computers which connected to a central server to locate other personal computers on a variety of computer networks, including TCP/IP, NetBIOS, and IPX networks. *See, e.g.*, VocalChat User's Guide, page 5 (illustrating a central server with a "post office" to enable communication between computers). *See also id.*, pages 7-8 (describing minimum personal computer requirements as a "386SX or higher IBM-compatible computer"); Help File, page 2 ("VocalChat can work with IPX, NetBIOS and TCP/IP network protocols."). As software, VocalChat is inherently stored as program code on a computer-usable medium. *See, e.g.*, Readme, page 1 (listing the VocalChat

files copied during installation). *See also* VocalChat User's Guide, page 8 (describing how VocalChat is installed by inserting "the VocalChat Disk in drive A").

Claim 43 requires "program code configured to access a directory database, the database having a network protocol address for a selected plurality of processes having on-line status with respect to the computer network, the network protocol address of each respective process forwarded to the database following connection to the computer network."

¶ 248. In a TCP/IP implementation, the server on which the Connection List/USERS file is located transmits the network protocol address of a second VocalChat client (second process) to a first VocalChat client (first process) upon request. As described in the Help File:

> Method of determining users address:
> Netware — Get Users information from Netware 2.x/3.x bindery
> WinWorkgroups — Get users information from Windows for Workgroups.
> Generic User — VocalChats files for users information. (See Generic network, below).

Help File, page 26. With any protocol other than Netware or Windows for Workgroups (such as TCP/IP or NetBIOS), a "generic" method is used where the VocalChat client queries VocalChat files (the Connection List/USERS files) locating users on the network. As described in greater detail in the Help File:

> When **NetBIOS** or **IPX** are used, but not with NetWare or Window for Workgroups, or when TCP/IP is used, a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory. In this case, the user name for each user, is entered when performing the **User Installation** in the Setup program. You should make sure that this name is not used by any other user on the network.

Help File, page 2 (underline emphasis added). Thus, With NetWare, the VocalChat client queries existing NetWare Bindery services locating "currently logged-in users;" with Windows for Workgroups, the VocalChat client queries the Windows for Workgroups services locating online users; and with other protocols, such as TCP/IP and NetBIOS, the VocalChat client queries the shared Connection List file (CONNLIST.VC). Regardless of protocol, the query determines whether the second process (the VocalChat client of another user) is connected to the computer network. For example, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file."

VocalChat Network Information, page 10 (emphasis added). Thus, a distinction is made between logged in users and logged out users. Similarly, as described above, in the NetWare implementation, the query retrieves a list of "currently logged in users."

Claim 43 also requires "program code responsive to one of the network protocol addresses and configured to establish a point-to-point communication link from the first process to the second process over the computer network."

¶ 249. VocalChat discloses that "[u]ser-to-user access is facilitated automatically through the [Connection List] file." Help File, page 17. VocalChat also discloses "the peer-to-peer nature of Windows for Workgroups, which VocalChat "uses . . . for user services." *Id.* In fact, VocalChat is a voice over computer network product for use on various networks that "enables communication between" VocalChat users. *Id.* at 8. *See also* User Guide, page 2 ("Talk with other users over the network, and broadcast to network users or groups. Access network users with the Address Book and Quick-Dial buttons.").

INDEPENDENT CLAIM 44

Claim 44 recites, in pertinent part: "In a first computer process operatively coupled over a computer network to a second process and an address server, a method of establishing a point-to-point communication between the first and second processes."

¶ 250. VocalChat is a software-based telephone executed on personal computers which connected to a central server to locate other personal computers on a variety of computer networks, including TCP/IP, NetBIOS, and IPX networks. *See, e.g.*, VocalChat User's Guide, page 5 (illustrating a central server with a "post office" to enable communication between computers). *See also id.*, pages 7-8 (describing minimum personal computer requirements as a "386SX or higher IBM-compatible computer"); Readme, page 1 (listing the VocalChat files copied during installation); Help File, page 2 ("VocalChat can work with IPX, NetBIOS and TCP/IP network protocols.").

Claim 44 requires "following connection of the first process to the computer network forwarding to the address server a network protocol address at which the first process is connected to the computer network."

¶ 251. As discussed above, VocalChat clients connect to a server to locate and establish point-to-point connections with other VocalChat clients over a network. For example, VocalChat discloses that "[u]ser-to-user access is facilitated automatically through the

[Connection List] file" which is stored on a server. Help File, page 17. VocalChat also discloses "the peer-to-peer nature of Windows for Workgroups, which VocalChat "uses . . . for user services." *Id.* In fact, VocalChat is a voice over computer network product for use on various networks that "enables communication between" VocalChat users. *Id.* at 8. *See also* User Guide, page 2 ("Talk with other users over the network, and broadcast to network users or groups. Access network users with the Address Book and Quick-Dial buttons."). Because the server stores network addresses of logged in clients, it is an "address server."

¶ 252. Inherently, a VocalChat client transmits its network protocol address "following connection of the [VocalChat client] to the computer network." Moreover, on many networks, including TCP/IP networks, network addresses are assigned "following connection to the computer network." *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). Thus, in at least some instances, a computer system executing VocalChat receives its IP address following connection to the computer network. Consequently, dynamic address assignment is inherent in the VocalChat system. Alternatively, as set forth below, it would have been obvious to one of ordinary skill in the art to use dynamic address assignment on a TCP/IP network.

Claim 44 also requires "querying the address server as to whether the second process is connected to the computer network."

¶ 253. In a TCP/IP implementation, the server on which the Connection List/USERS file is located receives queries from VocalChat clients (first processes) to determine the on-line status of other VocalChat clients (second processes). As described in the Help File:

> Method of determining users address:
> Netware — Get Users information from Netware 2.x/3.x bindery
> WinWorkgroups — Get users information from Windows for Workgroups.
> Generic User — VocalChats files for users information. (See Generic network, below).

Help File, page 26. With any protocol other than Netware or Windows for Workgroups (such as TCP/IP or NetBIOS), a "generic" method is used where the VocalChat client queries VocalChat files (the Connection List/USERS files) locating users on the network. As described in greater detail in the Help File:

> When **NetBIOS** or **IPX** are used, but not with NetWare or Window for Workgroups, or when TCP/IP is used, a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory. In this case, the user name for each user, is entered when performing the **User Installation** in the Setup program. You should make sure that this name is not used by any other user on the network.

Help File, page 2. Thus, with NetWare, the VocalChat client queries existing NetWare Bindery services locating "currently logged-in users;" with Windows for Workgroups, the VocalChat client queries the Windows for Workgroups services locating online users; and with other protocols, such as TCP/IP and NetBIOS, the VocalChat client queries the shared Connection List file (CONNLIST.VC). Regardless of protocol, the query determines whether the second process (the VocalChat client of another user) is connected to the computer network. For example, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file." VocalChat Network Information, page 10. Thus, a distinction is made between logged in users and logged out users. Similarly, as described above, in the NetWare implementation, the query retrieves a list of "currently logged in users."

Claim 44 also requires "receiving a network protocol address of the second process from the address server, when the second process is connected to the computer network."

¶ 254. As described above, with TCP/IP, "the network addresses of the different workstations are required for VocalChat to be able to access the network users." Help File, page 2. Moreover, with TCP/IP, "a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses." The ability to establish a call merely by identifying a unique user name demonstrates that the server "transmits" the network addresses of VocalChat callees from the server's directory database. Indeed, the directory database is used to "get . . . the addresses of specific users." *Id.*; *see also id.*, page 22 ("VocalChat will use the CONNLIST.VC files to get network addresses").

Claim 44 further requires "in response to the network protocol address of the second process, establishing a point-to-point communication link with the second process over the computer network."

¶ 255. VocalChat discloses that "[u]ser-to-user access is facilitated automatically through the [Connection List] file." Help File, page 17. VocalChat also discloses "the peer-to-peer nature of Windows for Workgroups, which VocalChat "uses . . . for user services." *Id.* In fact, VocalChat is a voice over computer network product for use on various networks that "enables communication between" VocalChat users. *Id.* at 8. *See also* User Guide, page 2 ("Talk with other users over the network, and broadcast to network users or groups. Access network users with the Address Book and Quick-Dial buttons.").

DEPENDENT CLAIMS 5-7, 11, 19-20, 22, 30-31, 34-37, 39-42

Claim 5 of the '704 patent requires "searching the computer memory for an entry relating the second process; and retrieving a network protocol address of the second process in response to a positive determination of the on-line status of the second process."

¶ 256. VocalChat inherently describes "searching the computer memory for an entry relating the second process." For example, as described above, VocalChat used a server to store names and network addresses of on-line users and to provide those network address to VocalChat clients upon request. *See, e.g.*, Help File, page 22 (VocalChat "will use the CONNLIST.VC files to get network addresses"); page 8 ("the Setup program creates a Connection List file which is used to identify and access users"). Consequently, the server was inherently capable of "searching the computer memory for an entry relating to the second process (i.e., a name and address of a VocalChat client).

Claim 6 of the '704 patent requires "transmitting the network protocol address of the second process to the first process when the second process is determined in step C to have a positive on-line status with respect to the computer network."

¶ 257. In a TCP/IP implementation, the server for the Connection List/USERS file receives queries from VocalChat clients (first processes) to determine the on-line status of other VocalChat clients (second processes). As described in the Help File:

> Method of determining users address:
> Netware — Get Users information from Netware 2.x/3.x bindery
> WinWorkgroups — Get users information from Windows for Workgroups.
> Generic User — VocalChats files for users information. (See Generic

network, below).

Help File, page 26. With any protocol other than Netware or Windows for Workgroups (such as TCP/IP or NetBIOS), a "generic" method is used where the VocalChat client queries VocalChat files (the Connection List/USERS files) locating users on the network. As described in greater detail in the Help File:

> When **NetBIOS** or **IPX** are used, but not with NetWare or Window for Workgroups, or when TCP/IP is used, a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory. In this case, the user name for each user, is entered when performing the **User Installation** in the Setup program. You should make sure that this name is not used by any other user on the network.

Help File, page 2. With NetWare, the VocalChat client queries existing NetWare Bindery services locating "currently logged-in users;" with Windows for Workgroups, the VocalChat client queries the Windows for Workgroups services locating online users; and with other protocols, such as TCP/IP and NetBIOS, the VocalChat client queries the shared Connection List file (CONNLIST.VC). Regardless of protocol, the query determines whether the second process (the VocalChat client of another user) is connected to the computer network. For example, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file." VocalChat Network Information, page 10. Thus, a distinction is made between logged in users and logged out users. Similarly, as described above, in the NetWare implementation, the query retrieves a list of "currently logged in users."

> **Claim 7 requires "generating an off-line message when the second process is determined in step C to have a negative on-line status with respect to the computer network; and transmitting the off-line message to the first process."**

¶ 258. First, as described above, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file." VocalChat Network Information, page 10 (emphasis added). Thus, a distinction is made

between on-line and off-line users. Similarly, as described above, in the NetWare implementation, the query retrieves a list of "currently logged in users." Moreover, various types of off-line messages are provided to indicate the unavailability of VocalChat users. *See, e.g.*, Troubleshooting Help File, page 2 (describing that when a user's name in the "New Users" dialog box, one of the causes may be that "[t]he 'Show only Logged-in' check-box is checked, and the person is not currently logged-in."). Consequently, the server inherently transmits "off-line messages" to the VocalChat clients to distinguish between online and offline users.

Claim 11 requires "querying the server as to the on-line status of the first callee process; and receiving a network protocol address of the first callee process over the computer network from the server."

¶ 259. As described above, with NetWare, the VocalChat client queries existing NetWare Bindery services to locate "currently logged-in users;" with Windows for Workgroups, the VocalChat client queries the Windows for Workgroups services to locate online users; and with other protocols, such as TCP/IP and NetBIOS, the VocalChat client queries the shared Connection List file (CONNLIST.VC). *See, e.g.*, Help File, page 2. Regardless of protocol, the query determines the online status of the callee process (the VocalChat client of a callee). For example, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file." VocalChat Network Information, page 10 (emphasis added). Thus, a distinction is made between logged in users and logged out users. Similarly, as described above, in the NetWare implementation, the query retrieves a list of "currently logged in users."

Claim 19 requires "wherein the caller process further comprises a visual display and the user interface comprises a graphic user interface."

¶ 260. The VocalChat client has a graphical user interface that is a "visual display." *See, e.g.*, User Guide, page 11 (illustrating the primary VocalChat GUI including a Call button, a volume slider and a plurality of Quick Dial buttons).

Claim 20 requires "wherein the steps of establishing a point-to-point link as described in step C is performed in response to manipulation of the graphic elements on the graphic user interface."

¶ 261. A VocalChat user makes a point-to-point call to another user with the Call button or a Quick Dial Button representing a frequently called callee. *See* User Guide, page 14 ("just select a user from the user list, and choose "OK"). Callees were also represented as Quick Dial buttons. *See* Help File, pages 11, 20-21 ("Setting a Quick Dial Button").

Claim 22 requires "program code for querying the server as to the on-line status of the first callee process; and program code for receiving a network protocol address of the first callee process over the computer network from the server."

¶ 262. As described above, with NetWare, the VocalChat client queries existing NetWare Bindery services to locate "currently logged-in users;" with Windows for Workgroups, the VocalChat client queries the Windows for Workgroups services to locate online users; and with other protocols, such as TCP/IP and NetBIOS, the VocalChat client queries the shared Connection List file (CONNLIST.VC). *See, e.g.*, Help File, page 2. Regardless of protocol, the query determines the online status of the callee process (the VocalChat client of a callee). For example, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file." VocalChat Network Information, page 10 (emphasis added). Thus, a distinction is made between logged in users and logged out users. Similarly, as described above, in the NetWare implementation, the query retrieves a list of "currently logged in users."

Claim 30 requires that the "computer system further comprises a visual display and the user interface comprises a graphic user interface."

¶ 263. The VocalChat client has a graphical user interface that is a "visual display." *See, e.g.*, User Guide, page 11 (illustrating the primary VocalChat GUI including a Call button, a volume slider and a plurality of Quick Dial buttons).

Claim 31 requires that "the element representing the first communication line and the element representing the first callee process are graphic elements and wherein the program code for establishing a point-to-point communication link from the caller process to the first callee process further

comprises: program code, responsive to manipulation of the graphic elements on the graphic user interface, for establishing the point-to-point communication link from the caller process to the first callee process."

¶ 264. A VocalChat user makes a point-to-point call to another user with the Call button or a Quick Dial Button representing a frequently called callee. *See* User Guide, page 14 ("just select a user from the user list, and choose "OK"). Callees were also represented as Quick Dial buttons. *See* Help File, pages 11, 20-21 ("Setting a Quick Dial Button").

Claim 34 requires "modifying the compilation of entries."

¶ 265. The compilation of entries stored on the server (e.g., in the Connection List/USERS file) is modified as new users install VocalChat software and as existing users log in and log off. For example, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file." VocalChat Network Information, page 10 (emphasis added). Later VocalChat implementations (e.g., version 2.02), refer to the USERS file as a "Connection List" file, which is modified in the same manner as the USERS file. *See, e.g.*, Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory.").

Claim 35 requires "adding an entry to the compilation upon the occurrence of a predetermined event."

¶ 266. An entry is added to the compilation of entries within the Connection List/USERS file when a user first sets up VocalChat or when the user logs in to VocalChat. *See, e.g.*, VocalChat Network Information, page 10 ("Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address"); Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses . . . the user name for each user, is entered when performing the User Installation in the Setup program"); page 4 (user installation "is used to . . . add the user to the Address Book"); page 10 (user name "will be used by VocalChat to identify you and will appear in the VocalChat Address Book and in the Connection List file").

Claim 36 requires that "the predetermined event comprises notification by a user process of an assigned network protocol address."

¶ 267. When a user logs in, or when a computer with VocalChat is turned on, the network address of the VocalChat client is sent to the Connection List/USERS file. For example, as described in VocalChat Network Information, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address." VocalChat Network Information, page 10.

Claim 37 requires "deleting an entry from the compilation upon the occurrence of a predetermined event."

¶ 268. VocalChat inherently discloses this limitation. Any database containing entries, such as the one used on the server containing the Connection List/USERS file, is inherently capable of deleting entries upon request from an end user and/or a network administrator. Moreover, when a user logs off the system, the user's network address is deleted from the list of "on-line" users. *See, e.g.,* VocalChat Network Information, page 10 ("VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file.") (emphasis added).

Claim 39 requires "program code configured to modify the compilation of entries."

¶ 269. The compilation of entries stored on the server (e.g., in the Connection List/USERS file) is modified as new users install VocalChat software and as existing users log in and log off. For example, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file." VocalChat Network Information, page 10 (emphasis added). Later VocalChat implementations (e.g., version 2.02), refer to the USERS file as a "Connection List" file, which is modified in the same manner as the USERS file. *See, e.g.*, Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses. This file is placed in the Post Office directory.").

Claim 40 requires "program code configured to add an entry to the compilation upon the occurrence of a predetermined event."

¶ 270. An entry is added to the compilation of entries within the Connection List/USERS file when a user first sets up VocalChat or when the user logs on to VocalChat. *See, e.g.*, VocalChat Network Information, page 10 ("Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address"); Help File, page 2 ("a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses . . . the user name for each user, is entered when performing the User Installation in the Setup program"); page 4 (user installation "is used to . . . add the user to the Address Book"); page 10 (user name "will be used by VocalChat to identify you and will appear in the VocalChat Address Book and in the Connection List file").

Claim 41 requires that the "predetermined event comprises notification by a process of an assigned network protocol address."

¶ 271. When a user logs in, or when a computer with VocalChat is turned on, the network address of the VocalChat client is sent to the Connection List/USERS file. For example, as described in VocalChat Network Information, "[w]hen the network used is not NetWare or Windows for Workgroups, VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address." VocalChat Network Information, page 10.

Claim 42 requires "program code configured to delete an entry from the compilation upon the occurrence of a predetermined event."

¶ 272. VocalChat inherently discloses this limitation. Any database containing entries, such as the one used on the server containing the Connection List/USERS file, is inherently capable of deleting entries upon request from an end user and/or a network administrator. Moreover, when a user logs off the system, the user's network address is deleted from the list of "on-line" users. *See, e.g.,* VocalChat Network Information, page 10 ("VocalChat maintains a shared USERS file with the names of logged in users. Each time a user loads VocalChat, its entry in the USERS file is updated with its IPX/NetBIOS address. When exiting VocalChat, the address is removed, but the user name is kept in the file.") (emphasis added).

2. The VocalChat references further in view of RFC 1531

¶ 273. Claim 1-2, 4, 7, 10-11, 19-22, 30-42 should be rejected under 35 U.S.C. § 103 as being unpatentable over the VocalChat references and further in view of RFC 1531.

¶ 274. Claim 33 states that the network protocol address of the client computer system is "dynamically assigned." *See* Claim 33 ("A method for locating processes having dynamically assigned network protocol addresses over a computer network"). Other independent claims state, more generally, that the network protocol address is assigned or transmitted to the database following the connection of the computer to the computer network. *See* Claim 1 ("transmitting to the server a network protocol address received by the first process following connection to the computer network"); Claim 2 ("each network protocol address stored in the memory following connection of a respective process to the computer network"); Claim 4 ("the network protocol addresses received following connection of the respective process to the computer network"); Claim 32 ("the Internet Protocol address added to the list following connection of the process to the computer network"); Claim 38 ("the network protocol address of the corresponding process assigned to the process upon connection to the computer network"); Claim 43 ("the network protocol address of each respective process forwarded to the database following connection to the computer network"); Claim 44 ("following connection of the first process to the computer network forwarding to the address server a network protocol address").

¶ 275. As described above, VocalChat inherently describes these features. By way of example, on many networks, including the TCP/IP networks of VocalChat, network addresses are assigned "following connection to the computer network." *See, e.g.*, Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks). For this reason, in at least some instances, the VocalChat computer system dynamically receives its IP address, following connection to the computer network. Consequently, dynamic address assignment is inherent in the VocalChat system.

¶ 276. Alternatively, a SNQ of patentability of Claims 1-2, 4, 7, 10-11, 19-22, 30-42 is raised under 35 U.S.C. § 103 based on the VocalChat references in view of RFC 1531, which describes how TCP/IP addresses are dynamically assigned. *See, e.g.*, Dynamic Host

Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531"), Section 2.2 (describing the "dynamic allocation of network addresses" on TCP/IP networks).

3. Motivation to Combine the VocalChat references with RFC 1531

¶ 277. A motivation to combine VocalChat with RFC 1531 exists because VocalChat describes the VocalChat software operating on a TCP/IP network and RFC 1531 describes a well known technique for dynamically assigning IP addresses within a TCP/IP network. One of ordinary skill in the art would have been motivated to combine VocalChat with RFC 1531 to realize the benefits associated with dynamic IP address assignment. For example, one of ordinary skill in the art would have been motivated to use dynamic IP address assignment because it eliminates the burdensome task of manually assigning IP addresses for all networked computers and allows for "automatic reuse of an address that is no longer needed by the host to which it was assigned." RFC 1531, page 2 (Section 1, Introduction). In fact, one of skill in the art would have understood at the time of the alleged invention of the '704 patent that VocalChat software would be installed and executed on personal computers that would frequently have their IP addresses dynamically assigned.

4. The VocalChat References in view of NetBIOS

¶ 278. Claim 3 should be rejected under 35 U.S.C. § 103(a) as being unpatentable over VocalChat references in view of NetBIOS.

Claim 3 of the '704 patent requires "a timer, operatively coupled to the processor, for time stamping the network protocol addresses stored in the memory."

¶ 279. VocalChat does not describe a timer for time stamping network protocol address. However, time stamping was a well known technique at the time the application which resulted in the '704 patent was filed. For example, the NetBIOS Name Server ("NBNS") described in NetBIOS includes a timer for time-stamping name/IP address entries. As described in NetBIOS, "[t]he NBNS may impose a 'time-to-live' on each name it registers. The registering node is made aware of this time value during the name registration procedure." NetBIOS at 382. Similarly, as described in NetBIOS:

If an end-node holds any names that have finite time-to-live values, then that node must periodically send a status report to the NBNS. Each name is reported using the NAME REFRESH REQUEST packet. These status reports restart the timers of both the NBNS and the reporting node. However, the only timers which are restarted are those associated with the name found in the status report. Timers on other names are not affected. *Id.*

5. Motivation to Combine the VocalChat References with NetBIOS

¶ 280. A motivation to combine VocalChat with NetBIOS explicitly exists within VocalChat. For example, NetBIOS is one of the network protocols explicitly supported by VocalChat. *See, e.g.*, Help File, page 2 ("When **NetBIOS** or **IPX** are used, but not with NetWare or Window for Workgroups, or when TCP/IP is used, a shared CONNLIST.VC file is used by the different running copies of VocalChat to hold user names and addresses.") (underline emphasis added). Thus, one of ordinary skill in the art would have been motivated to combine VocalChat with NetBIOS, because VocalChat explicitly states that NetBIOS may be used as the underlying network protocol.

6. The VocalChat References in view of Pinard

¶ 281. Claims 12-18 and 23-29 should be rejected under 35 U.S.C. § 103 as being unpatentable over the VocalChat references in view of Pinard.

CLAIMS 12-18 & 23-29

Claim 12 and 23 requires "providing an element representing a second communication line" and "program code for generating an element representing a second communication line," respectively.

¶ 282. The graphical user interface described in Pinard provides an element representing a second communication line. For example, call icons 23 and 29 representing two communication lines are shown in Figure 6 of Pinard. See Pinard, 5:31-40, Figure 6 ("Now there are clearly two calls in progress . . .").

Claims 13 and 24 require "terminating the point-to-point communication link from the caller process to the first callee process, in response to the user disassociating the element representing the first callee process from the element representing the first communication line" and "program code, responsive to the user disassociating the element representing the first callee

process from the element representing the first communication line, for terminating the point-to-point communication link from the caller process to the first callee process," respectively.

¶ 283. Figure 6 of Pinard illustrates how a call represented by call icon 23 is terminated by dragging the user icon for "John" 21 out of the call icon 23. Similarly, Figure 11 illustrates how a call is terminated by dragging the user icon to a "waste basket" icon 26.

Claim 13 and 24 further require "establishing a different point-to-point communication link from the caller process to the first callee process, in response to the user associating the element representing the first callee process with the element representing the second communication line" and "program code responsive to the user associating the element representing the first callee process with the element presenting the second communication line, for establishing a different point-to-point communication link from the caller process to the first callee process," respectively.

¶ 284. In Figure 6, the callee process icon for "John" 21 is dragged from call icon 23 to call icon 29, thereby terminating the call represented by call icon 23 and establishing a different link with the callee process represented by icon 21 (in this case a conference call with "John," "Mary," and "Debbie"). Moreover, once a callee is removed from a call by clicking and dragging the callee's icon, a new call can always be established with the callee by dragging the callee's icon to a call setup icon. *See, e.g.*, Pinard, Figure 3 (showing a callee icon dragged from a directory to a call setup icon to establish a call). *See also* Pinard, Col. 4, lines 22-31.

Claims 14 and 25 require "providing a user interface element representing a second callee process; and . . . establishing a conference point-to-point communication link between the caller process and the first and second callee process, in response to the user associating the element representing the second callee process with the element representing the first communication line" and "program code for generating an element representing a second callee process; and program code means, responsive to the user associating the element representing the second callee process with the element representing the first communication line, for establishing a conference communication link between the caller process and the first and second callee process," respectively.

¶ 285. In Figure 6 of Pinard, the user interface element for "John" 21 is dragged from call icon 23 to call icon 29, thereby creating a conference call between "John," "Mary," and "Debbie." See Pinard, Col. 5, lines 31-44 ("Now to conference all parties, the user Debbie merely drags the John icon to the call icon 29.").

Claims 15 and 26 require "removing the second callee process from the conference point-to-point communication link in response to the user disassociating the element representing the second callee process from the element representing the first communication line" and "program code, responsive to the user disassociating the element representing the second callee process from the element representing the first communication line, for removing the second callee process from the conference communication link," respectively.

¶ 286. In Pinard, any callee process can be removed from a conference call by dragging the element representing the callee process from the conference call icon. For example, Figure 8 of Pinard shows the user icon "Debbie" removed from conference call represented by call icon 32, thereby "breaking Debbie's line from the conference." Pinard, Col. 6, lines 14-15.

Claims 16 and 27 require "providing a user interface element representing a communication line having a temporarily disabled status" and "program code for generating an element representing a communication line having a temporarily disabled status," respectively.

¶ 287. Examples of a "temporarily disabled status" provided in the '704 patent include "line on hold" and "line on mute." *See, e.g.*, '704 patent, Claims 17 and 18. Pinard describes a user interface element representing a communication line having a temporarily disabled status. For example, Figure 12 illustrates a "hard hold" icon 39 to which user icons representing callers/callees 41 may be dragged to put the callers/callees on hold. *See, e.g.*, Pinard, Col. 6, lines 36-53 ("To place Mary on hard hold, Debbie drags Mary's icon 28 to the hard hold icon 39.").

Claims 16 and 27 also require "temporarily disabling a point-to-point communication link between the caller process and the first callee process, in response to the user associating the element representing the first callee process with the element representing the communication line having a temporarily disabled status" and "program code, responsive association of the element representing the first callee process with the element representing the communication line having a temporarily disabled status, for temporarily disabling the point-to-point communication link between the caller process and the first callee process," respectively.

¶ 288. In Pinard, in response to an icon of a caller/callee 41 being moved into the hard hold icon 39, the caller/callee is placed on hold. *See, e.g.*, Pinard, Col. 6, lines 36-53 ("To place Mary on hard hold, Debbie drags Mary's icon 28 to the hard hold icon 39.").

Claims 17 and 28 require that "the element provided in step D represents a communication line on hold status" and "the communication line having a temporarily disabled status comprises a communication line on hold status," respectively.

¶ 289. In Pinard, in response to an icon of a caller/callee 41 being moved into the hard hold icon 39, the caller/callee is placed on hold. *See, e.g.*, Pinard, Col. 6, lines 36-53 ("To place Mary on hard hold, Debbie drags Mary's icon 28 to the hard hold icon 39.").

Claim 18 and 29 require that "the element provided in step D represents a communication line on mute status" and "the communication line having a temporarily disabled status comprises a communication line on mute status," respectively.

¶ 290. VocalChat describes a "communication line on mute status." As described in the User's Guide, "Manual Activation can also be used like the MUTE option in many phones: it lets you talk without being heard on the other user's system." User's Guide, page 57.

7. Motivation to Combine VocalChat and Pinard

¶ 291. A motivation to combine VocalChat and Pinard exists due to the problem to be solved. Like VocalChat, Pinard relates to the field of computer-implemented telephony, and in particular to a method of indicating the status of various calls, to a user. *See* Pinard, Col. 1, lines 5-7. Indeed, the graphical user interface described in Pinard could be used in any system that operates a telephony application on a personal computer or on a personal computer in conjunction with a server. *See* Pinard, Col. 1, lines 60-62; Col. 2, lines 41-45. One of ordinary skill in the art would have recognized that the particular design choices reflected in the graphical user interface of Pinard could readily be implemented within the context of the network telephony system described in VocalChat. In fact, as described above, VocalChat discloses a graphical user interface with some similar features to those described in Pinard.

VII. LIST OF EXHIBITS

Exhibit A	**U.S. Patent No. 6,108,704 issued to Hutton et al. ("the '704 patent")**
Exhibit B	**Protocols for X/Open PC Interworking SMB, Version 2, THE OPEN GROUP (1992) ("NetBIOS"), which published as a single document with:**

- Protocol Standard for a NetBIOS Service on a TCP/UDP Transport:

Concept and Methods, RFC 1001 (March 1987) ("RFC 1001"); and

- Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications, RFC 1002 (March 1987) ("RFC 1002").

Exhibit C **Etherphone: Collected Papers 1987-1988. The papers <u>published together as a single document</u> include:**

- Polle T. Zellweger, et al., *An Overview of the Etherphone System and its Applications*, IEEE CONFERENCE ON COMPUTER WORKSTATIONS (March 1988), 160-168 (hereinafter "Zellweger 1").

- Daniel C. Swinehart, *Telephone Management in the Etherphone System*, PROCEEDINGS OF THE IEEE/IEICE GLOBAL TELECOMMUNICATIONS CONFERENCE (November 1987), 1176-1180 (hereinafter "Swinehart 1").

- Douglas B. Terry and Daniel C. Swinehart, *Managing Stored Voice in the Etherphone System*, ACM TRANSACTIONS ON COMPUTER SYSTEMS 6(1) (February 1988), 3-27 (hereinafter "Terry").

- Daniel C. Swinehart, *System Support Requirements for Multi-media Workstations*, PROCEEDINGS OF THE SPEECHTECH '88 CONFERENCE (April 1988), 82-83 (hereinafter "Swinehart 2").

- Polle T. Zellweger, *Active Paths through Multimedia Documents*, DOCUMENT MANIPULATION AND TYPOGRAPHY, J.C. AN VILET (ED.), CAMBRIDGE UNIVERSITY PRESS (1988) (hereinafter "Zellweger 2")

Exhibit D **Harrick M. Vin, et al., *Multimedia Conferencing in the Etherphone Environment*, IEEE COMPUTER SOCIETY (October 1991) ("Vin")**

Exhibit E **Dynamic Host Configuration Protocol, RFC 1531 (Oct. 1993) ("RFC 1531")**

Exhibit F **Pinard, et al., U.S. Patent No. 5,533,110 ("Pinard")**

Exhibit G **VocalChat User's Guide, Version 2.0 (1994) ("User's Guide")**

Exhibit H **VocalChat Readme File, Version 2.02 (June, 1994) ("Readme")**

Exhibit I **VocalChat 1.01 Networking Information ("VocalChat Networking")**

Exhibit J **VocalChat Information (July 18, 1994) ("Help File")**

Exhibit K **VocalChat Troubleshooting Help File (July 18, 1994) ("Troubleshooting Help File")**

Exhibit L	**Declaration of VocalTec, Ltd., co-founder Alon Cohen**
Exhibit M	**Claim Chart for NetBIOS**
Exhibit N	**Claim Chart for Etherphone**
Exhibit O	**Claim Chart for VocalChat**
Exhibit P	**Comments on arguments made by Net2Phone's expert to distinguish over NetBIOS**
Exhibit Q	**Comments on arguments made by Net2Phone's expert to distinguish over Etherphone**
Exhibit R	**Comments on arguments made by Net2Phone's expert to distinguish over VocalChat**
Exhibit S	**Plaintiff Net2Phone's Opening Claim Construction Brief (Oct. 18, 2007)**
Exhibit T	**Reformatted Opening Claim Construction Brief of Skype Technologies, SA, Skype, Inc., and Ebay Inc (Oct 18, 2007)**
Exhibit U	**Plaintiff Net2Phone Inc.'s Response Brief on Claim Construction (Oct. 18, 2007)**
Exhibit V	**Reformatted Responsive Claim Construction Brief of Skype Technologies SA, Skype, Inc., and EBay Inc (Oct. 18, 2007)**
Exhibit W	**Plaintiff Net2Phone, Inc.'s Reply Brief on Claim Construction (Oct. 19, 2007)**
Exhibit X	**Reply Claim Construction Brief of Skype Technologies SA, Skype, Inc., and EBay Inc. (Oct. 19, 2007)**

VIII. CONCLUSION

For the reasons set forth above, it is clear that a SNQ of patentability is raised in connection with claims 1-7 and 10-44 of the '704 patent by this Request for *Ex Parte* Reexamination since claims 1-7 and 10-44 are anticipated and/or rendered obvious in view of the above-listed prior art references. Therefore, it is requested that this request for reexamination be granted and claims 1-7 and 10-44 all be finally rejected.

As identified in the attached Certificate of Service and in accordance with 37 CFR §§1.33(c) and 1.915(b)(6), a copy of the present request, in its entirety, is being served to the address of the attorney or agent of record.

Please direct all correspondence in this matter to the undersigned.

Respectfully submitted,

_/ET/_____
Edwin Taylor
Registration No. 25,129

Dated:___/02-17-2009/

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN
1279 Oakmead Parkway
Sunnyvale, California 94085-4040
Telephone: 408/720-8300
Facsimile: 408/720-8383
Attorney Docket No.: 03801.G164