Seth W. Wiener, CSBN No. 203747
LAW OFFICES OF SETH W. WIENER
609 Karina Court
San Ramon, CA 94582
Telephone: (925) 487-5607
Facsimile (925) 828-8648
Email: seth@sethwienerlaw.com

Attorneys for Plaintiff
COMMUNICATION INTERFACE TECHNOLOGIES, LLC

UNITED STATES DISTRICT COURT
FOR THE CENTRAL DISTRICT OF CALIFORNIA
WESTERN DIVISION

COMMUNICATION INTERFACE TECHNOLOGIES, LLC,

Plaintiff,

v.

CKE RESTAURANTS HOLDINGS, INC.,

Defendant.

Case No.: 2:22-cv-7610

COMPLAINT FOR PATENT INFRINGEMENT

[DEMAND FOR JURY TRIAL]

Plaintiff Communication Interface Technologies, LLC ("CIT" or "Plaintiff"), for its Complaint against Defendant CKE Restaurants Holdings, Inc. ("CKE" or "Defendant"), alleges the following:

NATURE OF THE ACTION

1. This is an action for patent infringement arising under the patent laws of the United States, 35 U.S.C. § 1 *et seq*.

THE PARTIES

2. Plaintiff CIT is a limited liability company organized under the laws of the state of Delaware with a place of business at 3107 Boardwalk, Atlantic City, NJ 08401.

3. Upon information and belief, CKE is a corporation organized and existing under the laws of the state of Delaware, with a principal place of business located at 6700 Tower Circle, Suite 1000, Franklin, Tennessee 37067.

4. Upon information and belief, CKE sells and offers to sell products and services throughout the United States, including in this District, and introduces products and services into the stream of commerce and incorporates infringing technology knowing that they would be sold in this District and elsewhere in the United States.

JURISDICTION AND VENUE

5. This is an action for patent infringement arising under the patent laws of the United States, Title 35 of the United States Code.

6. This Court has subject matter jurisdiction under 28 U.S.C. §§ 1331 and 1338(a).

7. Venue is proper in this judicial district pursuant to 28 U.S.C. §1400(b). On information and belief, CKE has committed acts of infringement in this District and maintains multiple established places of business in the state of California and in this District, specifically including the CKE locations at 1) 7 South Pilas Street, Santa Barbara, CA 93103, 2) 2320 E 4th Street, Los Angeles, CA, and 3) 2110 W 7th Street, Los Angeles, CA 90057.

8. Upon information and belief, CKE is subject to this Court's general and specific personal jurisdiction due at least to their substantial business in California and in this District, directly or through intermediaries, including: (i) at least a portion of the infringements alleged herein; and (ii) regularly doing or soliciting business, engaging in other persistent courses of conduct and/or deriving substantial revenue from goods and services provided to individuals and entities in the State of California.

BACKGROUND OF THE PATENTS IN SUIT

The Invention

9. Eric Morgan Dowling and Mark Nicholas Anastasi are the inventors of U.S. patent nos. 6,574,239 ("the '239 patent"), 8,266,296 ("the '296 patent"), and 8,291,010 ("the '010 patent"). A true and correct copy of the '239 patent is attached as Exhibit 1. A true and correct copy of the '296 patent is attached as Exhibit 2. A true and correct copy of the '010 patent is attached as Exhibit 3.

10. The '239 patent, the '296 patent, and the '010 patent resulted from the pioneering efforts of Dr. Dowling and Mr. Anastasi (hereinafter "the Inventors") in the late 1990s, in the area of quickly resumed client-server communication sessions. (*See* Ex. 8, Dowling Decl., ¶¶ at 15-19.) These efforts resulted in the development of methods and apparatuses for virtual connection of a remote unit to a server and methods and apparatuses for application-layer evaluation of communications received by a mobile device. (*See id.* at ¶¶ 17, 19, 21.)

11. At the time of these pioneering efforts, the most widely implemented technology that was in use involved client-server communication sessions that could be instantiated and torn down. (*See id.* at ¶ 15.) If communications between client and server were needed again, the widely implemented technology would simply instantiate a brand new session between the same client and server. (*See id.*) Secure Sockets Layer (SSL) is an example of the earlier technology. Unlike Transport Layer Security (TLS), SSL did not allow session reactivation, and instead required a new session to be negotiated from scratch after an older session was deactivated (torn down).

12. Creating a new session required the renegotiation of a set of session keys that included computation of new cryptographic keys. (*See id.* at ¶ 15.) This process required significant start up times and computational resources. (*See id.*) The invention encompassed by the patents in suit, instead of tearing down an old session and instantiating a new session, places the old session into an inactive state, and then reactivates the old session to place it back into the active state using a much shorter renegotiation sequence that makes use of saved session parameters. (*See id.* at ¶ 26.) The saved session parameters include precomputed client-server encryption keys that are used to quickly and efficiently reactivate the inactive sessions. Some embodiments allow the session layer connection between the client and server devices to be reactivated without the need to create a new session by negotiating new session parameters and session keys. (*See* Exhibit 1 at Figs. 1A, 2, 3:45-63, 8:34-9:14, 9:54-60.) Other embodiments additionally or alternatively allow the application layer session to be reactivated without the need for the user to enter his/her user authentication credentials at the time of each session reactivation.

13. The Inventors first conceived of the inventions claimed in the ’239 patent, the ’296 patent, and the ’010 patent as a way to shorten the connection time of the dialup modems in use back in the 1990s. (*See* Ex. 8 at ¶ 16.) Each time a new dialup modem connection needed to be reestablished, there would be a several-second period (typically around 10-12 seconds) during which the user would hear audio modem tones and hissing sounds while the modems reconnected and negotiated a new data session. (*See id.*) The virtual session inventions allowed the modems to reconnect by remembering the previously negotiated modem parameters, thereby greatly shortening this renegotiation time to being almost unnoticeable. (*See* Ex. 1 at 13:42-43, 17:50-58; Ex. 8 at ¶ 16.)

14. While developing their invention, the inventors contemplated virtual sessions would also be very useful in wireless applications (*see*, *e.g.*, Ex. 1 at Fig. 2, 9:32-35, 13:4-8) to allow a client-side remote unit to maintain a virtual presence with a

remote server. (*See* Ex. 8 at ¶ 17.) The inventors taught that virtual sessions could be layered over wireless connections to allow remote units such as wireless Internet devices to be virtually connected to one or more server-side application programs running on one or more remote server systems without wasting wireless physical layer resources to maintain the one or more session layer connections. (*See* Ex. 1 at 9:28-60; Ex. 8 at ¶ 17.) The physical layer could be inactive, while the virtual session layer connections could be maintained without using wireless resources. (*See* Ex. 1 at 3:45-49, 8:56-58, 9:7-10; Ex. 8 at ¶ 17.) When the client-side remote unit needed to communicate with the server, or when the server needed to send newly received information to the remote unit, the virtual session could be reactivated without the need to tediously set up and authenticate a new secure cryptographic session with the server. (*See* Ex. 1 at Fig. 1A, 9:53-60, 13:48-14:17; Ex. 8 at ¶ 17.)

15. For example, the inventors developed methods for controlling virtual sessions between a server-side program and a client-side application program. (*See* Ex. 1 at 14:32-43.) When the virtual session is not needed, it is placed into an inactive state (like a sleep state). (*See, e.g.*, Ex. 1 at 3:45-49, 10:6-11:22; Ex. 2 at 3:56-60.) In this state, no communication resources are used. (*See* Ex. 1 at 3:37-44, 17:36-45.) When a virtual session is needed again, for example when the server receives new information for the client-side application program, the server can, for example, send a message that causes the client-side application program to resume the virtual session with the server. (*See* Ex. 1 at 3:60-63.) This session resumption is accomplished using saved session parameters instead of going through the full session authentication and negotiation process, as was needed in the prior art. In modern day parlance, the client-side application program is typically called an "App."

Advantage Over the Prior Art

16. The patented inventions disclosed in the '239 patent, the '296 patent, and the '010 patent provide many advantages over the prior art, and in particular improved the operations of communications between remote units such as wireless computing and

communications devices and remote servers. (*See* Ex. 1 at Figs. 1, 2; Ex. 2 at 3:48-4:39; Ex. 3 at 3:48-4:39.) One advantage of the patented inventions is providing systems and methods to enable users such as remote workers or other types of users to stay connected to one or more central servers without the need to continuously remain connected via one or more physical channels. (*See, e.g.*, Ex. 1 at 3:37-40; Ex. 2 at 3:48-51; Ex. 3 at 3:48-51; Ex. 8 at ¶ 25.) A central aspect of the inventions is the concept of fast reconnect. (*See, e.g.*, Ex. 1 at Abstract, 17:50-58; Ex. 2 at Abstract.) Users of remote devices can reconnect via a previously established communication session to a server-side application program, without the need to use the prior art's long and tedious session establishment procedures each time a reconnect is needed after a session has been deactivated. (*See* Ex. 1 at 12:49-53, 17:36-42; Ex. 8 at ¶¶ 20, 25.)

17. Another advantage offered by the patented inventions is to allow a remote unit to maintain a private/secured session layer connection to support communication between a client-side application program and a server-side application program over long periods of session inactivity. (*See* Ex. 8 at ¶ 25.) This may be achieved, for example, by computing cryptographic session parameters (*e.g.*, according to public key cryptography techniques) that can be used to quickly resume the session without the user needing to start a new authentication process from scratch. (*See* Ex. 1 at 3:2-5, 3:55-60, 4:22-25, 8:45-53, 10:2-15, 10:51-55, 10:57-62, 11:15-21, 14:32-33, 18:61-66, 20:40-43, 20:50-55, 21-49-55, 22:1-7; Figs. 6, 7; Ex. 8 at ¶ 25.) This connection can be referred to as a sustained secure connection that persists, for example, when the user has turned off his or her user device or put it in airplane mode and then turned it back on again. In the prior art, the secure cryptographic session would need to be terminated under such conditions, and a new secure session between the client and the server would need to be established from scratch. The session layer connection can preferentially be used to support various different kinds of application layer communications between the remote unit and the server-side application program. (*See* Ex. 1 at Figs. 1A, 2, 3:45-63, 8:34-9:14, 9:54-60.)

18. Another advantage offered by some embodiments of the patented inventions of the patents in suit is to allow a user using a remote unit to maintain a private/secured logon type session between a client-side application program and a server-side application program over longer periods of time, without the need for the user to repeatedly reenter his or her logon credentials such as username and password. This is achieved by computing cryptographic session parameters (*e.g.*, according to public key cryptography techniques) that can be used to quickly resume the session without the user needing to start a new authentication process from scratch. (*See* Ex. 1 at 3:2-5, 3:55-60, 4:22-25, 8:45-53, 10:2-15, 10:51-55, 10:57-62, 11:15-21, 14:32-33, 18:61-66, 20:40-43, 20:50-55, 21-49-55, 22:1-7, Figs. 6, 7; Ex. 8 at ¶ 25.) This can be referred to as a sustained secure connection that persists, for example, when the user has turned off his or her user device or put it in airplane mode and then turned it back on again. In the prior art, the secure cryptographic session would need to be terminated under such conditions, and a new secure session between the client and the server would need to be manually established in which the user would need to present his or her user credentials to establish a new session.

19. Another advantage offered by the patented inventions of the patents in suit is that the invention contemplated that the remote unit 100 of Fig. 1 and Fig. 2 of the ’239 patent would be able to wirelessly connect (207) (Ex. 1 at Fig. 2) to a plurality of different server-side application programs (220) (Ex. 1 at Fig. 2). (*See* also Ex. 1 at 7:21-25, 7:50-52, 14:62-64.) Typically, a smart phone device will have many different downloaded Apps, and each App will communicate with its own corresponding remote server-side application program. Furthermore, as disclosed in the ’239 patent (Ex. 1 at 7:41-44), each such connection between each App on the remote unit and each different server-side application program could be connected by its own virtual session, using a separate set of saved session parameters including cryptographic session reauthentication parameters for fast/accelerated session reconnect. The prior art required all the different sessions to be tediously and manually established and torn

down each time they were separately needed. (*See* Ex. 1 at 7:56-8:10, 17:50-54, 18:40-48, 19:57-60; Ex. 8 at ¶ 20.)

20. Yet another advantage offered by various embodiments of the patented inventions of the patents-in-suit is that any given server-side application program can use a table to maintain multiple virtual sessions with a plurality of remote units using a database of pre-computed and prestored cryptographic session keys. (*See* Ex. 1 at 8:61-9:4, 10:57-59, 11:12-21; Ex. 8 at ¶¶ 20-21.) That is, the server-side application program can manage a large number of secure cryptographic virtual sessions with a large number of different client-side wireless remote units that have downloaded the corresponding client-side App. (*See, e.g.*, Ex. 1 at Fig. 2, 9:61-10:13.) The prior art required these different sessions to be tediously and manually established and torn down each time they were separately needed.

21. Yet another advantage offered by various embodiments of the patented inventions of the patents-in-suit is that the virtual session can be reactivated based on either the remote unit requesting data or the server sending data. (*See* Ex. 1 at Figs. 3, 7, 13:21-28, 13:48-54, 13:59-14:3.) The prior art did not provide any means to use fast virtual session reconnection techniques to make the client/server experience seamless over extended periods of usage. (*See* Ex. 8 at ¶ 20.) Instead, techniques like SSL would require new sessions to be set up and torn down over and over again.

22. Yet another advantage offered by various embodiments of the patented inventions of the patents-in-suit is the ability of the server-side application program to send an unsolicited message to the client-side application running on the wireless remote unit to cause one or more virtual sessions to be reestablished. (*See* Ex. 1 at 3:61-63, 13:48-14:17, 24:61-64.) This message makes special use of saved cryptographic authentication parameters and information needed to identify the relevant client-side application program (App) that runs on the remote unit. (*See* Ex. 1 at Figs. 7, 8.) The specification not only describes specific exemplary embodiments that make use of caller ID type packets to send the outbound notification message, but the specification also

describes many more general alternative embodiments directed toward wireless applications. (*See* Ex. 1 at 6:45-51, 13:65-14:17, 22:39-55, 22:64-23:6, 23:29-32, 23:39-64, 24:31-25:8, and 25:20-26.)

23. Because of these significant advantages that can be achieved through the use of various embodiments of the patented inventions, the ’239 patent, the ’296 patent, and the ’010 patent present significant commercial value for companies like Defendant. Indeed, Defendant coordinates its products and services using its mobile Apps, providing convenience and efficiency for its customers, enhancing the customer engagement and experience of its customers, and increasing the efficiency of its own operations, in addition to other benefits.

<u>Prior Litigation</u>

24. The ’239 patent was previously litigated in the Eastern District of Texas (2-04-CV-00108, 2-03-CV-00465) and in the Northern District of Texas (3-04-CV-00281). These cases settled before any claim construction hearings were conducted, although in one case a joint claim construction and prehearing statement was submitted by the parties. *See* Dkt. 130, *East Texas Technology Partners, L.P. v. Toshiba America, Inc., et al.*, No. 2:03-CV-465(TJW) (E.D. Tex. Jan. 5, 2005).

25. The ’239 patent, the ’296 patent, and the ’010 patent were more recently asserted by CIT in several cases in the Eastern District of Texas. *See e.g., Communication Interface Technologies, LLC v. PepsiCo., Inc.* (4:20-cv-00286); *Communication Interface Technologies, LLC v. Rent-A-Center, Inc.* (4:20-cv-00287); *Communication Interface Technologies, LLC v. Texas Instruments, Inc.* (4:20-cv-00288*); Communication Interface Technologies, LLC v. Yum! Brands, Inc.* (4:20-cv-00289); *Communication Interface Technologies, LLC v. FedEx Corp.* (4:20-cv-00305); *Communication Interface Technologies, LLC v. Cinemark Holdings, Inc., et al.* (4:20-cv-00306); *Communication Interface Technologies, LLC v. Capital One Financial Corp.* (4:20-cv-00307); *Communication Interface Technologies, LLC v. American Messaging Services, LLC* (4:20-cv-00308); *Communication Interface Technologies,*

LLC v. Farmers Group, Inc. (4:20-cv-00526); *Communication Interface Technologies, LLC v. JPMorgan Chase & Co.* (4:20-cv-00527); *Communication Interface Technologies, LLC v. TD Ameritrade, Inc.* (4:20-cv-00528); *Communication Interface Technologies, LLC v. United Parcel Service of America, Inc.* (4:20-cv-00529); *Communication Interface Technologies, LLC v. Albertson's, LLC et al.* (4:20-cv-00550); *Communication Interface Technologies, LLC v. Aldi, Inc. et al.* (4:20-cv-00551); *Communication Interface Technologies, LLC v. The Allstate Corporation et al.* (4:20-cv-00552); *Communication Interface Technologies, LLC v. Tractor Supply Company* (4:20-cv-00805); *Communication Interface Technologies, LLC v. McDonald's Corporation et al.* (4:20-cv-00804); *Communication Interface Technologies, LLC v. Foot Locker, Inc.* (4:20-cv-00802); *Communication Interface Technologies, LLC v. 7-Eleven, Inc.* (4:20-cv-00800); *Communication Interface Technologies, LLC v. Teachers Insurance and Annuity Association of America et al.* (4:20-cv-00842)*; Communication Interface Technologies, LLC v. Applebee's Restaurants LLC* (4:21-cv-00778); *Communication Interface Technologies, LLC v. Choice Hotel's International, Inc.* (4:21-cv-00780); and *Communication Interface Technologies, LLC v. PNC Financial Services Group, Inc.* (4:21-cv-00782), among others.

<u>COUNT I – INFRINGEMENT OF U.S. PATENTS NO. 6,574,239</u>

26. The allegations set forth in the foregoing paragraphs 1 through 25 are incorporated into this First Claim for Relief.

27. On June 3, 2003, the '239 patent, entitled VIRTUAL CONNECTION OF A REMOTE UNIT TO A SERVER was duly and legally issued by the United States Patent and Trademark Office and the '239 patent expired on or about October 7, 2018.

28. CIT is the assignee and owner of the right, title and interest in and to the '239 patent, including the right to assert all causes of action arising under said patent and the right to any remedies for infringement of them.

29. As set forth above, the inventions of the '239 patent resolve technical problems related to client-server computing architecture. (*See* Ex. 8 at ¶ 21.)

30. The claims of the ’239 patent do not merely recite the performance of some business practice known from the pre-Internet world along with the requirement to perform it on the Internet. Instead, the claims of the ’239 patent recite one or more inventive concepts that are rooted in computerized client-server computing architecture technology and overcome problems specifically arising in the realm of computerized client-server computing architecture technologies. (*See id*. at ¶¶ 19, 21-26.)

31. As set forth above, the claims of the ’239 patent recite an invention that is not merely the routine or conventional use of computers. (*See id*. at ¶¶ 22-24.) Instead, the invention makes use of specific client-server computer architecture functionalities. The ’239 patent claims thus specify how computing devices and remote servers are manipulated to yield a desired result.

32. The technology claimed in the ’239 patent does not preempt all ways of using client-server computing architectures or the use of all communication session technologies, or any other well-known or prior art technology.

33. Each claim of the ’239 patent recites a combination of elements sufficient to ensure that the claim in practice amounts to significantly more than a patent on an ineligible concept.

34. As of the date of this filing, there are more than 180 licensees to the ’239 patent.

35. Upon information and belief, CKE has directly infringed under 35 U.S.C. § 271(a), literally and/or under the doctrine of equivalents, at least one claim of the ’239 patent by making, using, selling, offering to sell, importing and/or providing and/or causing to be used products, specifically one or more mobile device applications, which by way of example include the CKE Apps: “Carl’s JR” and other CKE Apps (the “Accused Instrumentalities”). (*See*, CKE Apps via Google Play, https://play.google.com/store/apps/details?id=com.carlsjr.ordering&hl=en_US&gl=US (last accessed and downloaded October 14, 2022).)

36. Upon information and belief, the exemplary versions herein and previous versions of the Accused Instrumentalities distributed prior to expiration of the patents in suit operated materially in the same manner.

37. Upon information and belief, at relevant times, the Accused Instrumentalities perform a method in which wireless push notification messages are sent over TLS sessions, and the remote server and the client-side application establish a separate TLS connection for traditional client-server communications. Earlier versions of the CKE Apps were developed and published in and before 2018 based on the version history. (*See* https://apps.apple.com/us/app/carls-jr-mobile-ordering/id1577790958 (last accessed and downloaded October 14, 2022).) User reviews of prior versions of the CKE App listed on Google Play extend to well prior to the expiration of the ’239 patent. (*See* https://play.google.com/store/apps/details?id=com.carlsjr.ordering&hl=en_US&gl=US (last accessed and downloaded October 14, 2022).)

38. Attached hereto as Exhibit 4, and incorporated herein by reference, is a claim chart detailing how one or more of the Accused Instrumentalities infringe claim 7 of the ’239 patent.

39. This infringement analysis is necessarily preliminary, as it is provided in advance of any discovery provided by CKE with respect to the ’239 patent.

40. CIT reserves all rights to amend, supplement and modify this preliminary infringement analysis.

41. Nothing in the attached chart should be construed as any express or implied contention or admission regarding the construction of any term or phrase of the claims of the ’239 patent.

42. The Accused Instrumentality infringed at least one claim of the ’239 patent during the pendency of the ’239 patent.

43. CIT has been harmed by CKE’s infringing activities regarding the ’239 patent.

COUNT II – INFRINGEMENT OF U.S. PATENTS NO. 8,266,296

44. The allegations set forth in the foregoing paragraphs 1 through 43 are incorporated into this Second Claim for Relief.

45. On September 11, 2012, the ’296 patent, entitled APPLICATION-LAYER EVALUATION OF COMMUNICATIONS RECEIVED BY A MOBILE DEVICE was duly and legally issued by the United States Patent and Trademark Office and the ’296 patent expired on or about March 30, 2019.

46. CIT is the assignee and owner of the right, title and interest in and to the ’296 patent, including the right to assert all causes of action arising under said patent and the right to any remedies for infringement of them.

47. As set forth above, the inventions of the ’296 patent resolve technical problems related to client-server computing architecture. (*See* Ex. 8 at ¶ 21.)

48. The claims of the ’296 patent do not merely recite the performance of some business practice known from the pre-Internet world along with the requirement to perform it on the Internet. Instead, the claims of the ’296 patent recite one or more inventive concepts that are rooted in computerized client-server computing architecture technology and overcome problems specifically arising in the realm of computerized client-server computing architecture technologies. (*See id*. at ¶¶ 19, 21-26.)

49. As set forth above, the claims of the ’296 patent recite an invention that is not merely the routine or conventional use of computers. (*See id*. at ¶¶ 22-24.) Instead, the invention makes use of specific client-server computer architecture functionalities. The ’296 patent claims thus specify how computing devices and remote servers are manipulated to yield a desired result.

50. The technology claimed in the ’296 patent does not preempt all ways of using client-server computing architectures or the use of all communication session technologies, or any other well-known or prior art technology.

51. Each claim of the ’296 patent recites a combination of elements sufficient to ensure that the claim in practice amounts to significantly more than a patent on an ineligible concept.

52. As of the date of this filing, there are more than 180 licensees to the ’296 patent.

53. Upon information and belief, CKE has directly infringed under 35 U.S.C. § 271(a), literally and/or under the doctrine of equivalents, at least one claim of the ’296 patent by making, using, selling, offering to sell, importing and/or providing and/or causing to be used products, specifically one or more mobile device applications, which by way of example include the CKE Apps: “Carl’s JR” and other CKE Apps (the “Accused Instrumentalities”). (*See*, CKE Apps via Google Play, https://play.google.com/store/apps/details?id=com.carlsjr.ordering&hl=en_US&gl=US (last accessed and downloaded October 14, 2022).)

54. Upon information and belief, the exemplary versions herein and previous versions of the Accused Instrumentalities distributed prior to expiration of the patents in suit operated materially in the same manner.

55. Upon information and belief, at relevant times, the Accused Instrumentalities perform a method in which wireless push notification messages are sent over TLS sessions, and the remote server and the client-side application establish a separate TLS connection for traditional client-server communications. Earlier versions of the CKE Apps were developed and published in and before 2018 based on the version history. (*See* https://apps.apple.com/us/app/carls-jr-mobile-ordering/id1577790958 (last accessed and downloaded October 14, 2022).) User reviews of prior versions of the CKE App listed on Google Play extend to well prior to the expiration of the ’296 patent. (*See* https://play.google.com/store/apps/details?id=com.carlsjr.ordering&hl=en_US&gl=US (last accessed and downloaded October 14, 2022).

56. Attached hereto as Exhibit 5, and incorporated herein by reference, is a claim chart detailing how one or more of the Accused Instrumentalities infringe claim 1 of the ’296 patent.

57. This infringement analysis is necessarily preliminary, as it is provided in advance of any discovery provided by CKE with respect to the ’296 patent.

58. CIT reserves all rights to amend, supplement and modify this preliminary infringement analysis.

59. Nothing in the attached chart should be construed as any express or implied contention or admission regarding the construction of any term or phrase of the claims of the ’296 patent.

60. The Accused Instrumentality infringed at least one claim of the ’296 patent during the pendency of the ’296 patent.

61. CIT has been harmed by CKE’s infringing activities regarding the ’296 patent.

COUNT III – INFRINGEMENT OF U.S. PATENTS NO. 8,291,010

62. The allegations set forth in the foregoing paragraphs 1 through 61 are incorporated into this Third Claim for Relief.

63. On October 16, 2012, the ’010 patent, entitled VIRTUAL CONNECTION OF A REMOTE UNIT TO A SERVER was duly and legally issued by the United States Patent and Trademark Office and the ’010 patent expired on or about March 30, 2019.

64. CIT is the assignee and owner of the right, title and interest in and to the ’010 patent, including the right to assert all causes of action arising under said patent and the right to any remedies for infringement of them.

65. As set forth above, the inventions of the ’010 patent resolve technical problems related to client-server computing architecture. (*See* Ex. 8 at ¶ 21.)

66. The claims of the ’010 patent do not merely recite the performance of some business practice known from the pre-Internet world along with the requirement to perform it on the Internet. Instead, the claims of the ’010 patent recite one or

more inventive concepts that are rooted in computerized client-server computing architecture technology and overcome problems specifically arising in the realm of computerized client-server computing architecture technologies. (*See id*. at ¶¶ 19, 21-26.)

67. As set forth above, the claims of the ’010 patent recite an invention that is not merely the routine or conventional use of computers. (*See id*. at ¶¶ 22-24.) Instead, the invention makes use of specific client-server computer architecture functionalities. The ’010 patent claims thus specify how computing devices and remote servers are manipulated to yield a desired result.

68. The technology claimed in the ’010 patent does not preempt all ways of using client-server computing architectures or the use of all communication session technologies, or any other well-known or prior art technology.

69. Each claim of the ’010 patent recites a combination of elements sufficient to ensure that the claim in practice amounts to significantly more than a patent on an ineligible concept.

70. As of the date of this filing, there are more than 180 licensees to the ’010 patent.

71. Upon information and belief, CKE has directly infringed under 35 U.S.C. § 271(a), literally and/or under the doctrine of equivalents, at least one claim of the ’010 patent by making, using, selling, offering to sell, importing and/or providing and/or causing to be used products, specifically one or more mobile device applications, which by way of example include the CKE Apps: “Carl’s JR” and other CKE Apps (the “Accused Instrumentalities”). (*See*, CKE Apps via Google Play, https://play.google.com/store/apps/details?id=com.carlsjr.ordering&hl=en_US&gl=US (last accessed and downloaded October 14, 2022).)

72. Upon information and belief, the exemplary versions herein and previous versions of the Accused Instrumentalities distributed prior to expiration of the patents in suit operated materially in the same manner.

73. Upon information and belief, at relevant times, the Accused Instrumentalities perform a method in which wireless push notification messages are sent over TLS sessions, and the remote server and the client-side application establish a separate TLS connection for traditional client-server communications. Earlier versions of the CKE Apps were developed and published in and before 2018 based on the version history. (*See* https://apps.apple.com/us/app/carls-jr-mobile-ordering/id1577790958 (last accessed and downloaded October 14, 2022).) User reviews of prior versions of the CKE App listed on Google Play extend to well prior to the expiration of the ’010 patent. (*See* https://play.google.com/store/apps/details?id=com.carlsjr.ordering&hl=en_US&gl=US (last accessed and downloaded October 14, 2022).)

74. Attached hereto as Exhibit 6, and incorporated herein by reference, is a claim chart detailing how one or more of the Accused Instrumentalities infringe claim 1 of the ’010 patent.

75. Attached hereto as Exhibit 7, and incorporated herein by reference, is a claim chart detailing how one or more of the Accused Instrumentalities infringe claim 17 of the ’010 patent.

76. These infringement analyses are necessarily preliminary, as they are provided in advance of any discovery provided by CKE with respect to the ’010 patent.

77. CIT reserves all rights to amend, supplement and modify this preliminary infringement analysis.

78. Nothing in the attached charts should be construed as any express or implied contention or admission regarding the construction of any term or phrase of the claims of the ’010 patent.

79. The Accused Instrumentality infringed at least one claim of the ’010 patent during the pendency of the ’010 patent.

80. CIT has been harmed by CKE’s infringing activities regarding the ’010 patent.

PRAYER FOR RELIEF

WHEREFORE, CIT demands judgment for itself and against CKE as follows:

A. An adjudication that CKE has infringed the patents in suit;

B. An award of damages to be paid by CKE adequate to compensate CIT for CKE's past infringement of the patents in suit, including interest, costs, expenses and an accounting of all infringing acts including, but not limited to, those acts not presented at trial;

C. A declaration that this case is exceptional under 35 U.S.C. § 285, and an award of CIT's reasonable attorneys' fees; and

D. An award to CIT of such further relief at law or in equity as the Court deems just and proper.

DEMAND FOR TRIAL BY JURY

Pursuant to Local Rule 38-1, CIT hereby demands a trial by jury on all claims so triable.

Dated: October 19, 2022

LAW OFFICES OF SETH W. WIENER

Seth Wiener

By: ______________________________
Seth W. Wiener
Attorneys for Plaintiff
COMMUNICATION INTERFACE TECHNOLOGIES, LLC

EXHIBIT 1

US006574239B1

(12) United States Patent
Dowling et al.

(10) Patent No.: US 6,574,239 B1
(45) Date of Patent: Jun. 3, 2003

(54) **VIRTUAL CONNECTION OF A REMOTE UNIT TO A SERVER**

(76) Inventors: **Eric Morgan Dowling**, 1132 W. Lookout Dr., Richardson, TX (US) 75080; **Mark Nicholas Anastasi**, 405 Copperas Trail, Highland Village, TX (US) 75077

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/167,698**

(22) Filed: **Oct. 7, 1998**

(51) **Int. Cl.**^{7} **H04J 3/00**
(52) **U.S. Cl.** **370/469**; 370/329
(58) **Field of Search** 370/329, 389, 370/468, 469; 709/203, 206, 227

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,285,061 A		8/1981	Ho	
4,416,015 A		11/1983	Gitlin	
4,489,416 A		12/1984	Stuart	
4,578,796 A		3/1986	Charalambous et al.	
4,852,151 A		7/1989	Dittakavi et al.	
4,995,074 A		2/1991	Goldman et al.	
5,127,051 A		6/1992	Chan et al.	
5,287,401 A		2/1994	Lin	
5,321,722 A		6/1994	Ogawa	
5,367,563 A		11/1994	Sainton	
5,519,767 A		5/1996	O'Horo	
5,600,712 A		2/1997	Hanson et al.	
5,751,796 A		5/1998	Scott et al.	
5,757,890 A		5/1998	Venkatakrishnan	
5,771,353 A	*	6/1998	Eggleston et al.	709/227
5,896,444 A		4/1999	Perlman et al.	
5,903,602 A		5/1999	Torkkel	
5,958,006 A	*	9/1999	Eggleston et al.	709/206
6,023,493 A		2/2000	Olafsson	
6,101,531 A	*	8/2000	Eggleston et al.	709/206
6,308,281 B1		10/2001	Hall, Jr. et al.	
6,317,455 B1		11/2001	Williams et al.	
6,426,946 B1		7/2002	Takagi et al.	

FOREIGN PATENT DOCUMENTS

WO	WO 99/27702	6/1999

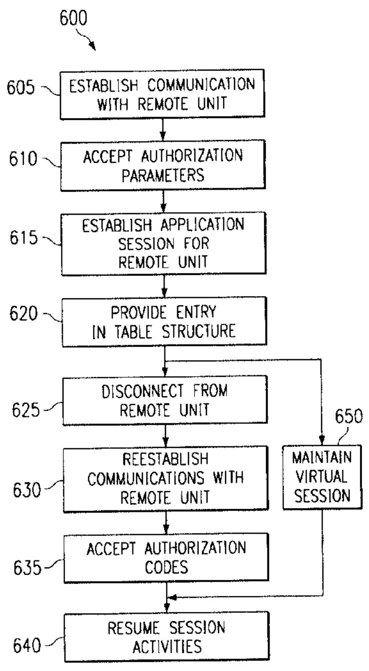
* cited by examiner

Primary Examiner—Salvatore Cangialosi
(74) *Attorney, Agent, or Firm*—Eric M. Dowling

(57) **ABSTRACT**

A method is provided for reconnecting a telephone modem with a reduced delay by reducing a time associated with retraining. A wireline communication connection is initialized by a telephone modem to train a set of parameters. The parameters are stored in a memory structure. The connection is used for communication, and is then terminated. At a later time, the connection is reestablished by accessing the parameters from memory and using them to reconnect the modem with a reduced set-up delay. Another method involves coupling to a first physical layer and establishing a session with a server, and then decoupling from the first physical layer while maintaining the session. Later, the session is resumed using a second physical layer. At least one of the physical layers involves a local interface unit that includes a landline connection to a WAN.

75 Claims, 4 Drawing Sheets


600
605
ESTABLISH COMMUNICATION WITH REMOTE UNIT
610
ACCEPT AUTHORIZATION PARAMETERS
615
ESTABLISH APPLICATION SESSION FOR REMOTE UNIT
620
PROVIDE ENTRY IN TABLE STRUCTURE
625
DISCONNECT FROM REMOTE UNIT
650
MAINTAIN VIRTUAL SESSION
630
REESTABLISH COMMUNICATIONS WITH REMOTE UNIT
635
ACCEPT AUTHORIZATION CODES
640
RESUME SESSION ACTIVITIES

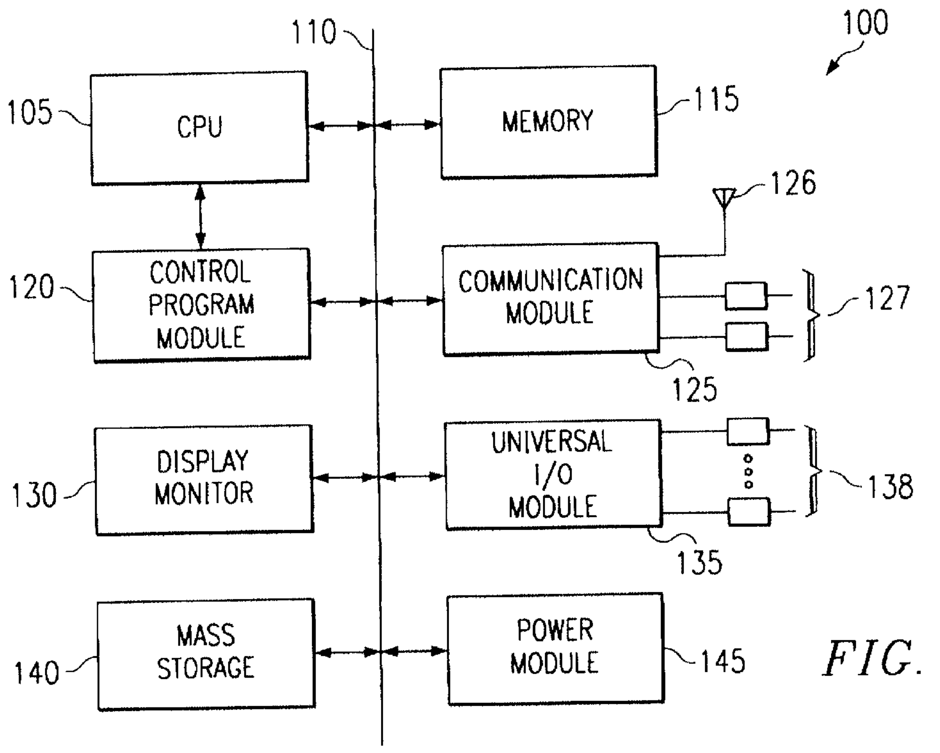

100
110
105
CPU
MEMORY
115
126
120
CONTROL PROGRAM MODULE
COMMUNICATION MODULE
127
125
130
DISPLAY MONITOR
UNIVERSAL I/O MODULE
138
135
140
MASS STORAGE
POWER MODULE
145

FIG. 1

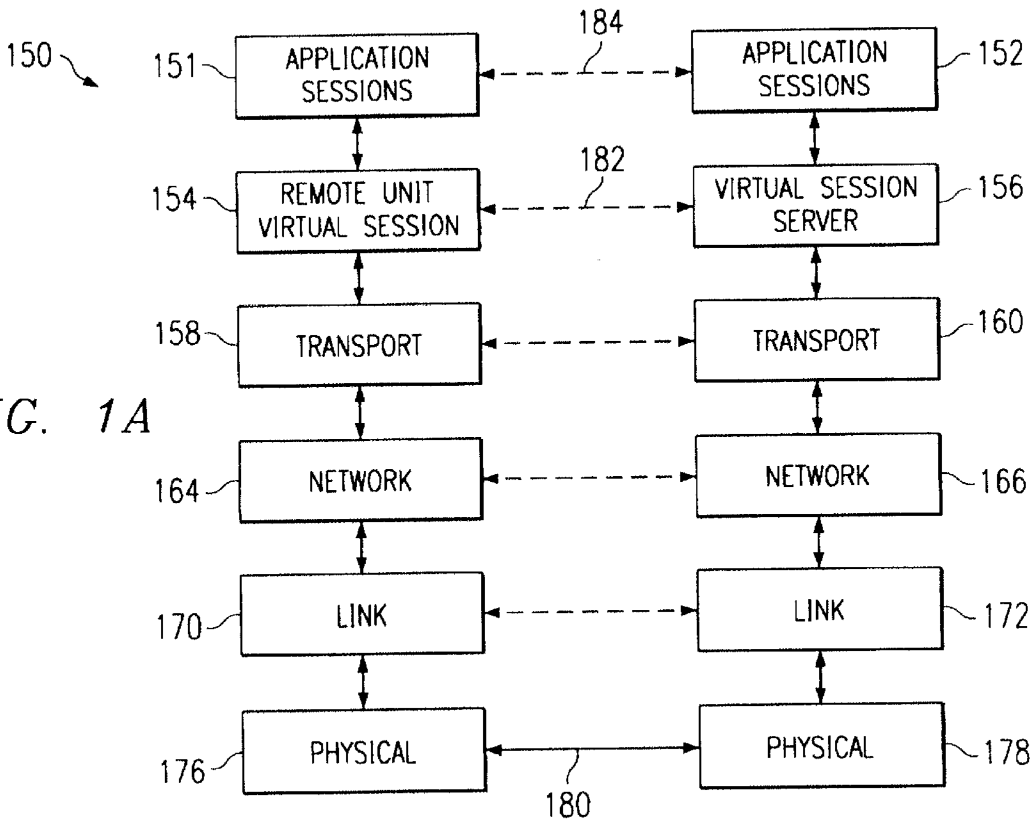

150
151
APPLICATION SESSIONS
184
APPLICATION SESSIONS
152
154
REMOTE UNIT VIRTUAL SESSION
182
VIRTUAL SESSION SERVER
156
158
TRANSPORT
TRANSPORT
160
164
NETWORK
NETWORK
166
170
LINK
LINK
172
176
PHYSICAL
PHYSICAL
178
180

FIG. 1A

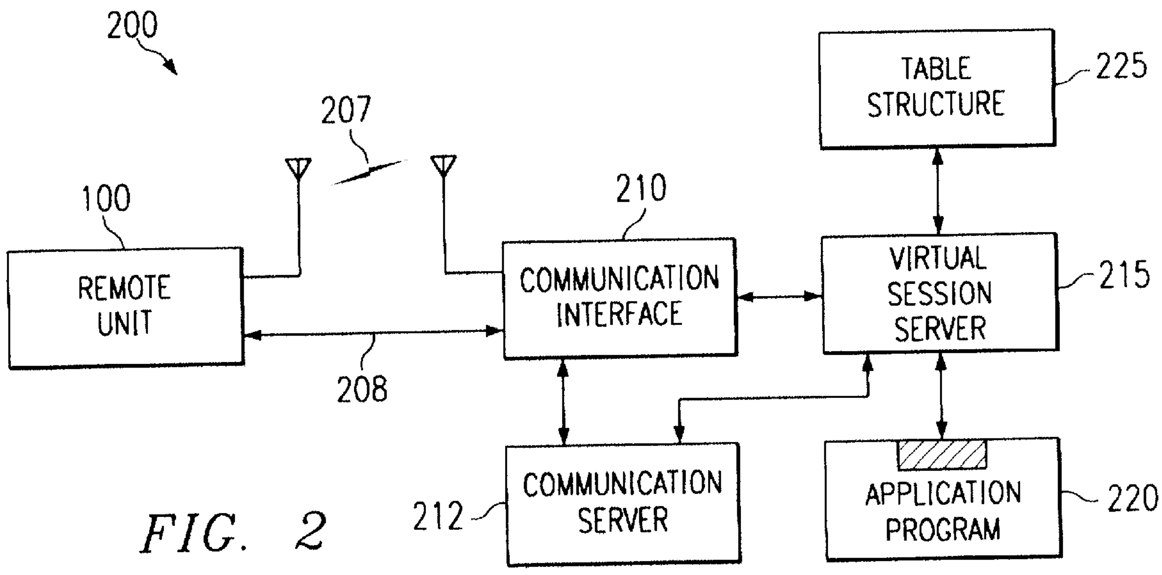

200
207
100
REMOTE UNIT
208
210
COMMUNICATION INTERFACE
212
COMMUNICATION SERVER
225
TABLE STRUCTURE
215
VIRTUAL SESSION SERVER
220
APPLICATION PROGRAM

FIG. 2

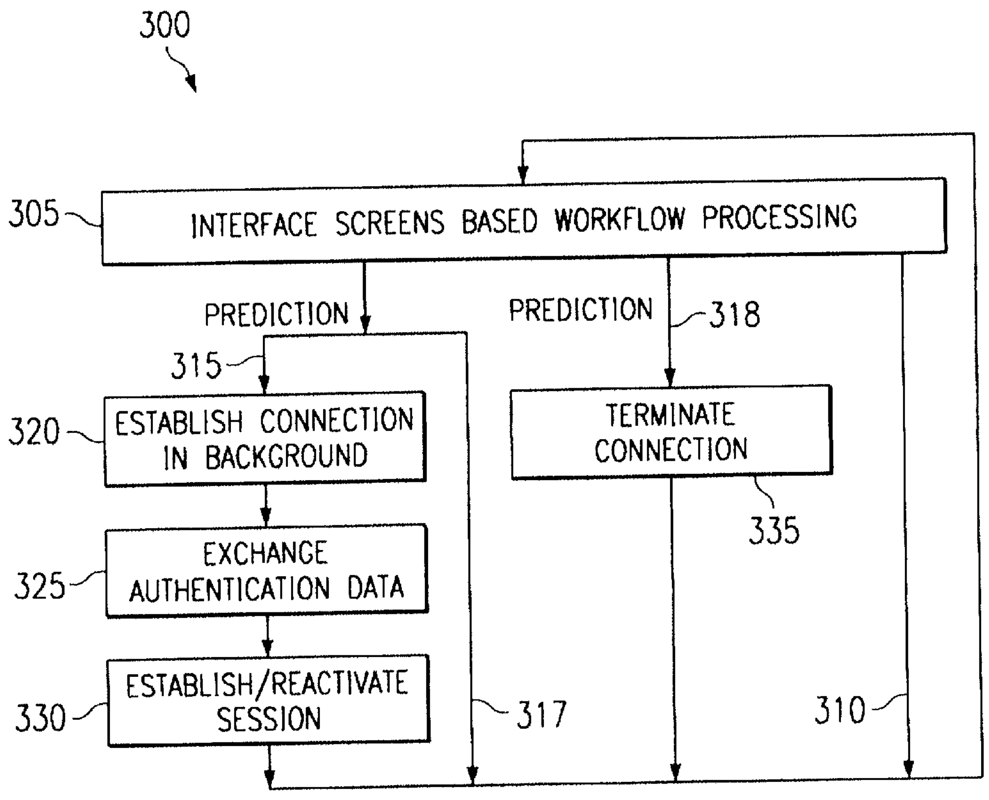

300
305
INTERFACE SCREENS BASED WORKFLOW PROCESSING
PREDICTION
PREDICTION
318
315
320
ESTABLISH CONNECTION IN BACKGROUND
325
EXCHANGE AUTHENTICATION DATA
330
ESTABLISH/REACTIVATE SESSION
TERMINATE CONNECTION
335
317
310

FIG. 3

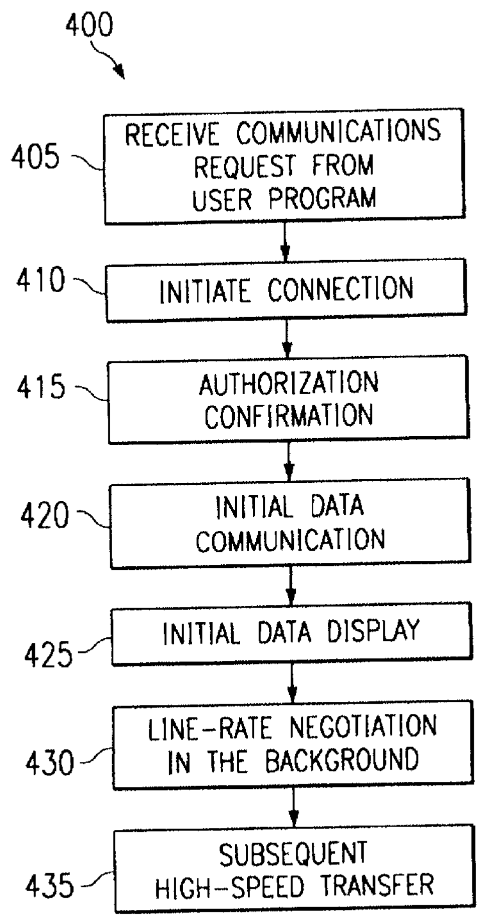

400
405
RECEIVE COMMUNICATIONS REQUEST FROM USER PROGRAM
410
INITIATE CONNECTION
415
AUTHORIZATION CONFIRMATION
420
INITIAL DATA COMMUNICATION
425
INITIAL DATA DISPLAY
430
LINE-RATE NEGOTIATION IN THE BACKGROUND
435
SUBSEQUENT HIGH-SPEED TRANSFER

FIG. 4

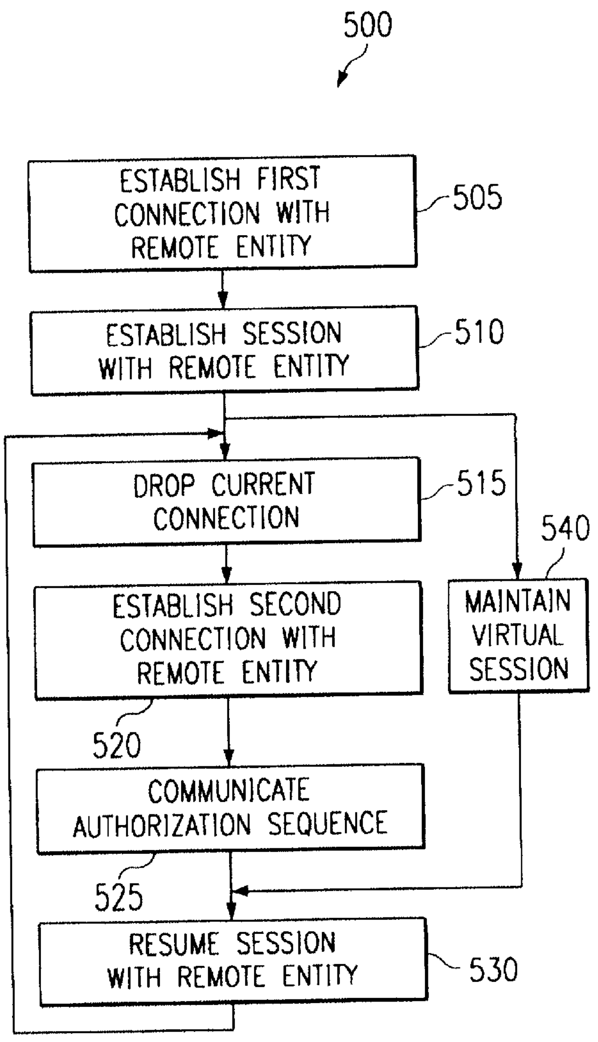

500
ESTABLISH FIRST CONNECTION WITH REMOTE ENTITY
505
ESTABLISH SESSION WITH REMOTE ENTITY
510
DROP CURRENT CONNECTION
515
540
MAINTAIN VIRTUAL SESSION
ESTABLISH SECOND CONNECTION WITH REMOTE ENTITY
520
COMMUNICATE AUTHORIZATION SEQUENCE
525
RESUME SESSION WITH REMOTE ENTITY
530

FIG. 5

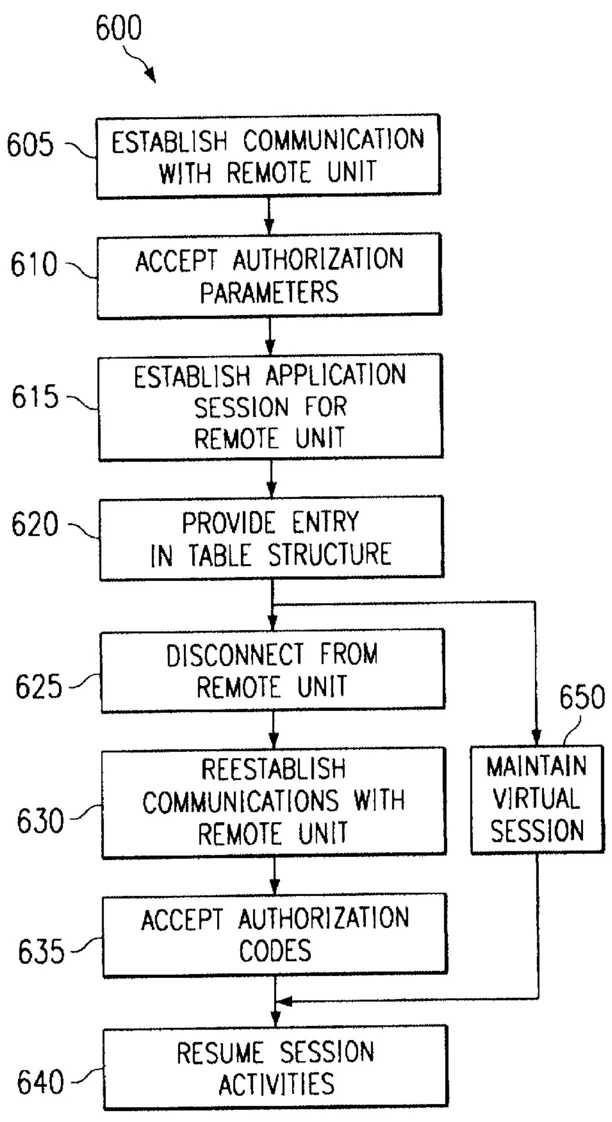

600
605 ESTABLISH COMMUNICATION WITH REMOTE UNIT
610 ACCEPT AUTHORIZATION PARAMETERS
615 ESTABLISH APPLICATION SESSION FOR REMOTE UNIT
620 PROVIDE ENTRY IN TABLE STRUCTURE
625 DISCONNECT FROM REMOTE UNIT
630 REESTABLISH COMMUNICATIONS WITH REMOTE UNIT
635 ACCEPT AUTHORIZATION CODES
640 RESUME SESSION ACTIVITIES
650 MAINTAIN VIRTUAL SESSION

FIG. 6

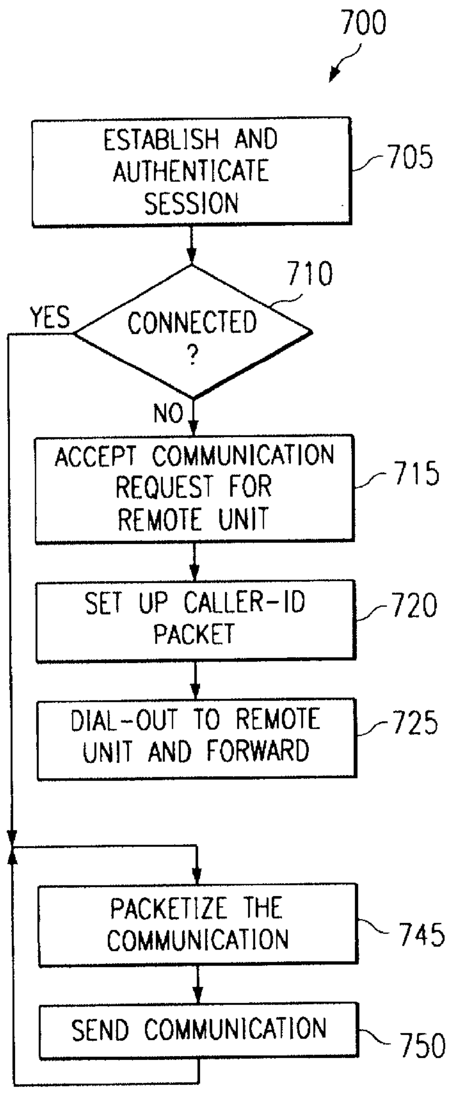

700
705 ESTABLISH AND AUTHENTICATE SESSION
710 CONNECTED ?
YES
NO
715 ACCEPT COMMUNICATION REQUEST FOR REMOTE UNIT
720 SET UP CALLER-ID PACKET
725 DIAL-OUT TO REMOTE UNIT AND FORWARD
745 PACKETIZE THE COMMUNICATION
750 SEND COMMUNICATION

FIG. 7

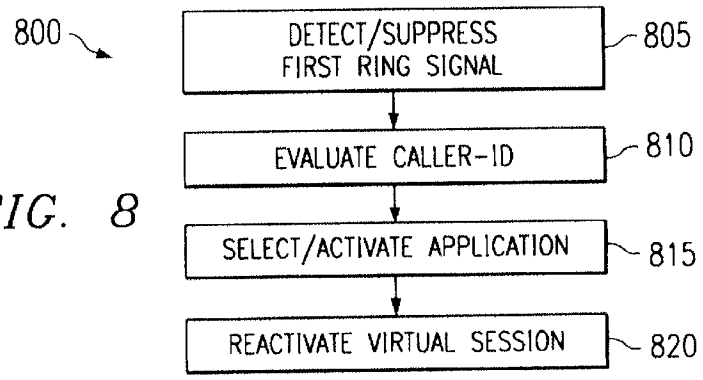

800
805 DETECT/SUPPRESS FIRST RING SIGNAL
810 EVALUATE CALLER-ID
815 SELECT/ACTIVATE APPLICATION
820 REACTIVATE VIRTUAL SESSION

FIG. 8

VIRTUAL CONNECTION OF A REMOTE UNIT TO A SERVER

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates generally to client-server computing architectures and communication techniques. More particularly, the invention relates to a system whereby a mobile worker and a central server may maintain a virtually continuous connection without the need to maintain a physical connection continuously.

2. Description of the Related Art

The concept of a virtual connection has arisen in connection with telecommuting and related applications. Such a system is described in U.S. Pat. No. 5,764,639. A telecommuter dials into a server using a standard telephone line. The telecommuter's modem and a modem controlled by the central server establish a connection therebetween. Once a connection is established, the telecommuter may access a computer connected to the server, read emails and receive phone calls and faxes. For example, if a customer attempts to call the telecommuter at work by dialing into a private branch exchange (PBX), the server will convert the incoming call to a packetized form, such as H.323, and redirect the call via the existing connection between the telecommuter and the server. Using this system, the telecommuter may access a computer at work, answer phone calls and answer emails. The telecommuter thus appears to be present in his or her office and thus has a virtual presence there. Note for this system to properly function, the telecommuter must stay connected to the server at all times. While this does not present a significant problem for local telecommuting, this solution is quite costly for long distance telecommuting. Likewise, this solution is very costly if the telecommuter is mobile and must maintain a virtual presence with the server using a cellular wireless connection. Furthermore, in some areas it may be difficult to maintain a wireless connection continuously. A lost connection may also prevent one from regaining access to the system until some period of time has passed. Some mobile workers require only intermittent access to the server, but find it too inconvenient to place a dial-in call and to log onto the system every time access is needed.

There is a need to provide mobile workers with various forms of virtual connectivity. Mobile workers differ from telecommuters in that while a telecommuter typically works from a single home location or remote office, a mobile worker moves from location to location during the course of a normal working day. An example of a mobile worker is a home-care professional. A home-care professional is a medical worker who periodically travels to visit with different sets of homebound patients according to their individual needs. The individual patients each have a set of medical records indicative of their medical histories. A patient's medical record is preferably maintained as an interactive electronic document containing multiple parts. For example, the medical record indicates to the home-care professional precisely what procedures are to be performed and what medications are to be administered or otherwise given to the patient. Once the services are performed, the home-care professional must annotate the medical record accordingly. The medical record is updated to reflect the patient's vital signs and other information related to patient progress. Also, a billing system takes note to track expendables and services rendered. For example, the patient may be billed per visit

and each visit may involve the expenditure of billable resources such as medicines.

In the above scenario, a mobile worker must interact with a central server during the course of a day. The worker may wish to access the central server while visiting a patient. The worker may also wish to access the server from a location where only a wireless connection can be established. From a performance perspective, an ideal solution is to provide the mobile worker with a wireless connection from a remote unit to a central server. Such a wireless connection could be established via a high-powered radio connection with a broad area of coverage or via an existing cellular or personal communication system (PCS) network. Solutions using high-powered radio links have the disadvantage that costly spectrum may be required. Maintaining a link on a cellular or PCS system is expensive in that a continuous connection consumes billable airtime which is also very costly. From an airtime-cost perspective, an ideal solution would be to force the worker to create a connection, download or up load information, and work locally with data on the remote unit as often as possible. This solution is tedious, and while saving airtime costs, may actually represent the more costly solution when professional service costs are factored in. This method has the added disadvantage that when files are uploaded or downloaded the data must be synchronized in case another user has changed the data in parallel with the mobile worker. Alternatively, other users must be "locked out" of the file from the time the mobile user downloads it until it is finally uploaded with any changes made. This is the problem solved using semaphores in shared memory systems. In the context of the present invention, a "file semaphore" is a semaphore used to lock a second user out of a file while a first user is using it. Due to the aforementioned reasons, in many applications forcing the worker to repeatedly connect, disconnect, upload and download information is unacceptable.

Some mobile networks have been constructed using what is known as cellular digital packet data (CDPD). In a CDPD network, a remote unit transmits a data packet on an unused analog channel. In this sense the mobile unit remains virtually connected to a CDPD communication server. Wireless airtime is only consumed when data is actually sent. A disadvantage to this approach is CDPD networks are not universally available. Cellular coverage is much more ubiquitous than CDPD coverage. Also, CDPD network subscribers must often pay high fees and hence CDPD may not represent the most economical solution.

In some systems such as packet switched network routers, offices make use of dial-out links. Dial-out links are useful when remote offices are separated by long distances. In such systems, when a packet must be routed from a first office to a second office, a call is placed to route the packet. The dial-out connection remains connected until a no-traffic condition is detected, indicating the line is no longer active. When the no-traffic condition is detected the connection is dropped until it is again needed. Dial-out links are thus used to reduce long distance fees associated with maintaining a constant connection, and represents a useful starting point for solving the foregoing problems relating to the establishment of a virtual presence of a mobile worker. Client-server protocols and fast automated connection strategies employing dial-out links are needed to provide new ways for a mobile worker to maintain a virtual presence. Also, new methods are needed to enable dial-out links to be set up with low delays to make them more useful for novel systems.

It would be desirable to provide a system whereby a remote worker could maintain a seamless connection with a central server without the need to maintain a dedicated channel. It would be desirable if the remote worker could communicate with the central server without the need to spend time to enter a password, reconnect, and wait for a line negotiation sequence to proceed before being able to use the connection. It would be desirable for a protocol stack to activate a virtual session based on a prediction derived from a workflow. It would be desirable to use this prediction to set up a connection in the background without disturbing the mobile worker while the mobile worker performed tasks in a workflow. It would also be desirable to have a remote unit which contains most of the screen-related information needed to provide the appearance of an established connection before the connection has been fully established. It would be desirable for the remote unit to download information before it is needed and upload information after it is gathered without the user even being aware these actions are being performed. It would further be desirable to establish a virtual session using a first communication medium such as a landline and to later communicate using the same virtual session using a second communication medium such as a wireless link. This would allow a mobile worker to select the most economical or convenient means of communications at a given time. In embodiments involving modem-based connections, it would be desirable to transmit data immediately using instantly available but lower line speeds. It would be desirable to then negotiate a higher line speed in the background while the remote worker and/or the server perform other tasks. Moreover, it would be desirable to establish a session between a remote unit and a server so that various forms of communications may proceed while providing the user with the appearance the user is continuously connected to the server and has a virtual presence with the server.

SUMMARY OF THE INVENTION

The present invention solves these and other problems by providing systems and methods to enable a remote worker to stay virtually connected to a central server without the need to continuously remain connected via a physical channel. The present invention is useful when costs are associated with maintaining a connection, for example when the connection has associated with it long distance, wireless, or other usage-related toll charges.

A first aspect of the present invention involves a communication protocol making use of a virtual session layer. The virtual session layer allows a communication session and an application session to be maintained in a deactivated state when no physical connection exists. When a remote unit later reconnects with a server, the virtual session is placed into an active state and session communications resumes as though uninterrupted. A remote unit, a virtual session server, and a communication system including the remote unit and the virtual session server are presented to support virtual sessions communications. In one embodiment, the virtual session server manages a logon session between the remote unit and a server-side application program. The virtual session server emulates the presence of the remote unit to the server-side application program and thereby maintains the logon. In related embodiments, the server-side application program involves a communication server capable of relaying messages and establishing communication channels with the remote unit using the virtual session layer.

A second aspect of the present invention involves a method of accessing a central server from a remote unit. A first step involves presenting a workflow to a user via a user interface. A second step involves predicting, based upon the

workflow, when the user will require connectivity to the central server. Based upon the prediction and in the background, a third step involves initiating the establishment of a physical layer communication connection to the central server.

A third aspect of the present invention involves a method of establishing a connection with a low connection set-up time. In a first step, the method initiates the establishment of a communication connection to be used to communicate with a remote entity. Next the method communicates application layer data via the communication connection prior to the completion of a line-rate negotiation process. Next the method negotiates a line speed in the background.

A fourth aspect of the present invention involves a method of setting up and operating a virtual session. This method can be practiced by a client-side remote unit or a server-side virtual session server. A first connection is established to a remote entity. This first connection is then used to establish a set of parameters needed to define a communication session with the remote entity. Next the first connection disconnected and a set of communication session parameters are maintained. Next a second connection to the remote entity is established and an authorization sequence is communicated. The communication session is next reactivated using the communication session using the second connection. A related method is used to allow a remote unit to maintain a virtual communications presence with a remote communication server coupled to a virtual session server.

A fifth aspect of the invention involves a method of accessing a server from a remote unit. At a first time, a first physical layer is used to establish a session with the server. The session involves a communication path that involves at least partially the first physical layer. The first physical layer is then decoupled while maintaining the session. At a second time, a second physical layer is used to resume the session so that the communication path of the session involves at least partially the second physical layer. At least one of the physical layers involves a local coupling to a local interface unit that couples to a WAN via a landline connection and the communication path of the session to the server comprises the landline connection.

Another aspect of the invention involves a method of reconnecting a telephone modem with a reduced delay by reducing a time associated with retraining. A first communication connection over a wireline telephone communication channel is initialized using the telephone modem. The initializing is performed at least partially by performing a line rate negotiation sequence with a far end modem to train a set of parameters to be used to support communications over the wireline telephone communication channel. The set of parameters can include signal constellation parameters, echo canceller coefficients, and/or equalizer coefficients for example. The set of parameters are then stored in a memory structure. Communication then commence at a negotiated date rate via the first communication connection with the far end modem using the parameters. Then the first communication connection is terminated. Next, the parameters are accessed from the memory structure and are used to at least partially initialize a second communication connection over the wireline telephone communication channel using the telephone modem. Communicaiton next commences at the negotiated date rate via said second communication connection by reusing at least some of the parameters. This allows a setup delay time associated with the initialization of the first communication connection to be longer than a setup delay time associated with the initialization of the seocnd communication connection.

BRIEF DESCRIPTION OF THE FIGURES

The various novel features of the present invention are illustrated in the figures listed below and described in the detailed description which follows.

FIG. **1** is a block diagram representing an embodiment of a remote unit designed in accordance with the present invention.

FIG. **1**A is a block diagram illustrating a layered software architecture representative of the communication protocols of the present invention.

FIG. **2** is a block diagram illustrating a system comprising a remote unit operably coupled to a server via a communication medium.

FIG. **3** is a flow chart illustrating a method of processing whereby an application program implementing a workflow provides a prediction of when the user will need a connection and establishes a connection in the background just before it is needed.

FIG. **4** is a flow chart illustrating a method of establishing communication with a remote entity with a near-immediate set up time.

FIG. **5** is a flow chart illustrating a method of communicating by maintaining a virtual presence without the need to continuously maintain a physical connection.

FIG. **6** is a flow chart illustrating a method of processing performed on a server acting as a front-end to an application program to maintain sessions for remote users who are not continuously physically connected to the application program.

FIG. **7** is a flow chart illustrating a method of processing performed on a server managing virtual connections for users who are not continuously physically connected to the server.

FIG. **8** is a flow chart illustrating a method of processing performed by a remote unit to accept different types of incoming calls.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

FIG. **1** is a block diagram representing an embodiment of a remote unit **100** designed in accordance with the present invention. The remote unit **100** may be implemented as a laptop computer, a personal digital assistant, a desktop computer or workstation, or as a dedicated unit customized for a particular application. The remote unit **100** includes a central processing unit (CPU) **105** connected to a central bus **110**. The central processing unit may be implemented using an available microprocessor, microcontroller, or customized logic. For example, a Pentium™ processor from Intel Corp. may be used to implement the CPU **105**. The central bus is preferably constructed as a set of unbroken wires used to carry signals between a set of component subsystems within the remote unit **100**. It should be noted, in some embodiments of the present invention, the bus **110** may be implemented equivalently using a set of direct parallel and/or serial connections between individual modules. The bus **110** as illustrated in FIG. **1** shows a low cost and a preferred means to connect the illustrated subsystems. Any combination of bus connections and direct serial or parallel links may be used to implement the connection structure provided by the bus **110**. Different implementations represent different price-to-performance ratios and will be dictated by the needs of an individual embodiment. The bus **110** also comprehends multi-layered bus structures. For example, some embodiments make use of a local processor bus connected to the

CPU **105**, and a peripheral interconnect bus for other subsystems. In multi-layered bus based designs, the different layers are preferably connected by bus bridges. All of these and other equivalent embodiments of the bus **110** are known to the skilled artisan. From here forward, the discussion will center on the illustrated embodiment of the remote unit **100** whereby all subsystems are directly connected via the bus **110**. Embodiments where the bus **110** represents a different physical interconnection topology are implicitly included in the discussion below.

A memory **115** is also coupled to the bus **110**. The memory **115** may be implemented using static random access memory (SRAM) or dynamic random access memory (DRAM). One type of SRAM is read-only memory (ROM). Preferably the memory **115** includes a ROM for use at boot-up, and a DRAM to hold a significant amount of data storage for use while executing programs. The remote unit **100** also includes a control program module **120**. The control program module **120** is controllably coupled to the CPU **105** and is also coupled to the bus **110**. The central program module **120** typically exists as a software module executed from the memory **115** by the CPU **105**. The control program module **120** effectively configures the remote unit **100** to operate in accordance with aspects of the present invention as discussed herein below.

A communications module **125** is also coupled to the bus **110**. The communications module includes at least one communication interface to allow the remote unit to communicate with a remote entity such as a virtual session server as will be discussed in detail hereinafter. In a preferred embodiment, the communications module **125** includes a plurality of communication interfaces. For example, a first interface **126** provides a wireless link, and a second communication interface **127** provides one or more wireline links. Also, the wireline communication interface **127** may include a standard telephone modem interface and a packet style interface designed to plug directly into an Ethernet connection to be coupled to a local area network (LAN), a wide area network (WAN) or the Internet. The Internet is the well-known and ubiquitous World Wide Web. In some embodiments, the communications module **125** includes a caller-identification packet processor. A caller-identification packet processor receives a caller-identification packet, extracts information therefrom, and passes the information to the CPU **105**. Caller-identification packets may be advantageously used to identify incoming calls with a virtual session as discussed in connection with FIGS. **7–8**. The communications module **125** may optionally include a voice interface to allow a user to engage in telephone conversations using the remote unit **100**. In this case a separate handset or a built-in handset may be used. Alternatively a speakerphone may be built into the remote unit using a microphone, a speaker, and an echo canceller.

The remote unit **100** also includes a display monitor **130**. The display monitor **130** is also coupled to the bus **110**. The display monitor **130** is preferably implemented using a liquid crystal display (LCD), although other display technologies may equivalently be used. Also connected to the bus **110** is an optional universal input-output (I/O) module **135**. The universal I/O module includes a coupling to a set of external devices **138**. The external devices are preferably data collection units as described below. The universal I/O module preferably provides a standard link layer interface to the software module **120** executing on the CPU **105**. The remote unit **100** also preferably includes a mass storage device **140**. The mass storage device **140** is also connected to the bus **110**. The mass storage device is preferably implemented using magnetic disk or optical disk technology, but any mass storage device, to include a non-volatile memory, may be used.

The remote unit **100** also includes a power module **145**. The power module is preferably and optionally coupled to the bus **110** to receive power management control information. The power module preferably includes a battery, an alternating current (AC) connector, a direct current (DC) connector, and a power management control interface. The AC connector allows the remote unit **100** to be powered from a standard **110** V wall outlet. The DC connector allows the remote unit **100** to be powered from a vehicle, for example by plugging the unit into a cigarette lighter outlet. Either of these connectors may be preferably used to also charge the battery in the remote unit. The power module **145** is coupled to supply power to a power bus which is connected to all subsystems. Depending on the power management configuration of an individual system, different subsystems may accept power from separate connections to allow portions of the remote unit to be selectively turned off while they are not being used.

The remote unit **100** is operative to execute an application program. The application program is operative to supply a sequence of interactive screens or a menu based interface to the user. The sequence of interactive screens or a particular usage of a menu based system implements a workflow. In an example embodiment, the remote unit **100** is carried by a home-care professional. The home care professional has a sequence of procedures which need to be implemented in the course of working with a patient. This sequence of procedures gives rise to the workflow implemented in the control program **120** which executes on the remote unit **100**. In the example embodiment involving a home-care professional, the universal I/O module is connected to a set of peripheral units to collect vital information such as blood pressure, temperature, insulin level and the like. Other information such as the patient's weight may be entered manually by the home-care professional as a part of the workflow. At certain times in the workflow, an external communication connection will be needed because data may need to be uploaded or downloaded to/from a central server. In accordance with the present invention, the remote unit **100** is operative to provide a seamless and transparent virtual presence with the central server. In general, the central server may itself be segmented into two or more individual central servers. The discussion herein focuses on an embodiment whereby a virtual presence is maintained with a single central server having multiple server components. The present invention may be equivalently practiced by embodiments involving a virtual presence with more than one central server. Thus, one remote unit could maintain multiple virtual connections to totally separate server systems. In such a configuration the application workflow would dictate to which server system the remote unit would physically connect while other servers remain virtually connected.

A key aspect of the operation of the remote unit **100** is its ability to maintain a virtual presence with the central server without continuously maintaining a physical connection. The remote unit **100** is operative to provide communications when it is needed without the user needing to go through a set of normally associated connection sequences. For example, in accordance with one aspect of the invention, the user need only interact with the screens provided to implement the workflow while the remote unit **100** automatically sets up a connection in the background to be available when it is needed. In embodiments where file synchronization is not an issue or is handled using file semaphores, the software implementing the workflow automatically downloads information before it is needed and later automatically uploads new information after it has been gathered. This way, users need not even be aware they are not connected at all times. The user is not burdened with the need to connect and reconnect, and need not be burdened with downloading and uploading data. The user experiences the full benefit of being continuously connected to the central server without the associated cost of remaining continuously connected via a physical connection. In systems where file semaphores are not employed, the physical connection is established just before the workflow indicates it will be needed and is dropped when the workflow indicates it will not be needed for some time. Further details of the operation of the remote unit **100** are given in the discussions provided in connection with FIGS. **2**–**8**.

A central aspect of the present invention involves the concept of a "virtual session." A session as defined herein is similar to the definition provided in the open systems interconnect (OSI) reference model from the International Standards Organization (ISO). The OSI model is a model of a layered software structure used in computer communications. A software system which implements a layered model of communication is known as a "protocol stack." The OSI model is well known and divides a computer communications process into seven layers. At each layer is a software module which communicates with a peer software module at the same layer. Within a protocol stack, each layer communicates with the layer above and/or below. Actual communication systems often deviate from the seven layer OSI model. A protocol stack using basic concepts similar to the seven layer OSI model is next discussed which represents an aspect of the present invention.

FIG. **1A** illustrates a representative protocol **150** used to support the present invention. At the top layer is an application session layer. A first protocol stack with an application session layer software module **151** communicates with a second protocol stack with an applications session layer software module **152**. The application session layer software module **151** is typically implemented as a client-side software module which presents a user interface to a user. The application session layer software module **152** is typically implemented by a server-side software module operative to provide communication and/or computer related services to the client-side software module. For example the application sessions layer software module **152** may involve a logon session with a database program, or may represent a unified messaging server supplying voice mail, email and fax mail. Similarly, the application session layer software module **152** may involve a telephony application operative to provide a packet switched or a circuit switched telephone connection to the client-side application session layer software module **151**. One layer down in the protocol stack is a virtual session layer. In the example embodiment, the first protocol stack implements a virtual session layer software module **154** in the remote unit **100**. The virtual session layer software module **154** communicates with a peer virtual session software module **156** via the peer-to-peer communication path **182**. In the exemplary embodiment, the virtual session layer software module **156** is implemented within a virtual session server as discussed in connection with FIG. **2**. The virtual session server typically maintains a table linking one or more application sessions to a virtual session. For example, this linking of application sessions into the table structure may be accomplished by including a pointer to a data structure containing application session control data, or by placing the data structure holding the application session control data directly in the table structure. Additionally, the table structure allows the virtual session server to maintain a plurality of virtual sessions with a plurality of client remote units. In the OSI model, the OSI-session layer provides a set of rules used to establish and terminate data streams between nodes in a network. A set of OSI-session layer services include establishing and terminating node connections, message flow control, dialog control, and end-to-end data control. The session layer controls dialogs, which involve conversational protocols as used in mainframe computer terminal communications. The virtual session layer **154, 152** of the protocol **150** may perform any of these functions in addition to maintaining the table linking to the application sessions.

The next software modules in the first protocol stack are the transport layer **158**, the network layer **164**, the link layer **170** and the physical layer **176**. These software modules respectively perform peer communications with the server-side protocol stack's software modules **160**, **166**, **172**, and **178**. The physical layer defines the low-level mechanical and electrical channel protocols and the physical connection itself. These four lower layers are well known in the art of data communications and can be implemented in various well-known ways. Likewise, alternative and equivalent protocol stacks may be constructed, for example, with the transport layer removed, various layers merged into one, or new layers added.

An important aspect of a virtual session oriented communication protocol such as the protocol **150** is the ability to maintain a peer-to-peer virtual session communication path **182** without the presence of a physical layer communication path **180**. The physical layer communication path **180** represents a physical layer communication connection, for example, a wireline connection, a cellular wireless connection, or a network connection to the Internet. When the physical layer communication path **180** is disconnected, no physical channel exists between the client-side software and the server-side software, and the physical layer communication path **180** is said to be in a "disconnected state." However, data structures maintained at the virtual session layer allow one or more peer-to-peer application session communication paths **184** to remain in a deactivated but existent state, even when the physical layer communication path is in the disconnected state. Likewise, the virtual session communication path **182** established between the remote unit and the virtual session server also remains in a deactivated but existent state. This is made possible through the use of the table structure maintained in memory which retains its information after the physical layer communication path **180** has been disconnected. When the physical layer communication path **180** has been reconnected, the physical layer communication path is said to be placed into a "connected state." At such time, the virtual session layer software modules **154** and **156** are operative to reactivate the virtual session layer communication path **182** and the application sessions layer communication path **184**. When these paths are reactivated, peer-to-peer communication may once again proceed over the application session layer communication path **184** and the virtual session layer communication path **182**.

As defined herein, a distinction is made between a communication session and an application session. A communication session is defined as a session between nodes or communication endpoints, and an application session is defined as a session between applications. For example, a remote unit may establish a communication session with a central server. In this case a communication session is

established between the communication endpoints, i.e., the remote unit and the central server. Also, an application program running on the remote unit may need to establish an application session with an application program running on the central server. In such case an application session is created using a connection stream provided and governed by the communication session. A table structure is used to maintain both the communication session parameters and the application session parameters. For example, a first user authentication parameter may be used to establish a communication session with the server. A second user authentication parameter may be used to establish an application session with the application program. This second user authentication parameter may include a user identification parameter and a password, for example.

In light of the aforesaid concepts, a "virtual session" is next defined. A virtual session is preferably implemented as a communication session as defined above. A virtual session, like an OSI session, provides a set of rules for establishing data streams between nodes or endpoints. The virtual session also may provide other session features such as dialog control, message flow control, and end-to-end data control. A virtual session is controlled using a data structure which provides a way to associate the virtual session with the lower layers of a protocol stack, leading down to a physical layer. As mentioned above, in most embodiments, a virtual session is implemented as a communication session. Application sessions are then added onto the virtual session as connection streams within the communication session.

In a virtual session, a communication session may be suspended with some or all of the lower layers of the protocol stack missing. In particular, a virtual session may be maintained while a physical layer connection has been removed. The virtual session can then be reassociated with a physical layer connection at a later time. The virtual session thus also preferably provides connect and reconnect rules used to establish a virtual session and then to reassociate the virtual session to a new physical connection to set up a new data stream in support of a dialog at a later time. Related activities such as the initiation of dial-out links to reestablish a physical layer communication path is also preferably handled by the virtual session in response to a signal from an application layer program.

An aspect of a virtual session is the maintenance of an application between an application program and a virtual session server as will be described below. A virtual session server acts as a proxy agent for a remote unit. When the remote unit is not connected via a physical layer communication path, the virtual session server maintains a proxy-presence with the application program on behalf of the disconnected remote unit. At a later time, when the remote unit reconnects into the virtual session by passing a set of communication session authentication parameters, the remote unit is thereby granted access to one or more application sessions which have been maintained in proxy by the virtual session server.

In a preferred embodiment, the virtual session uses a set of authentication parameters and a set of encryption keys to maintain a secure connection. A separate set of authentication parameters is used by an application running on the remote unit to gain access to an individual application session. Once the application session has been established over a virtual session, a table is used to maintain a set of parameters needed to maintain the application session, even though no physical layer connection exists between the endpoints of the virtual session. When a virtual session data structure is set up and no physical layer connection exists to support communication over the virtual session, the virtual session is said to be "inactive." When a virtual session data structure is set up and a physical layer connection does exist to support communication over the virtual session, the virtual session is said to be "active." A transition from an active state to an inactive state is called "deactivating a virtual session," and a transition from an inactive state to an active state is called "activating a virtual session." The process of transitioning from an active state to an inactive state is also known as "disconnecting from a physical connection." When this occurs, the physical layer connection is no longer available to support communication over the virtual session. In a preferred embodiment, a table structure is used to maintain the virtual session parameters as well as a set of parameters for each application session established over the virtual session. When a virtual session is activated, there is no need to reauthenticate the individual application sessions. This is because the table typically includes a user identification parameter, a user password, a set of application session parameters, a communication session identification parameter, and an encryption key for the communication session. Additional data such as modem initiation parameters may be added to the table as required by the system configuration and usage.

Referring now to FIG. **2**, a block diagram illustrating a system configuration **200** is shown. The system configuration **200** includes the remote unit **100** operatively coupled to a communication interface **210**. A direct wireless link **207** optionally couples the remote unit **100** to the communication interface **210**. A direct wireless link is used in embodiments where the remote unit **100** maintains a direct wireless link with the communication interface **210**. The communication interface thus provides an air interface for the direct wireless link. Alternatively or in addition to the direct wireless link **207**, a wireline link **208** couples the remote unit **100** to the communication interface **210**. The communication interface **210** maintains the connection **208** via a network interface coupled to a public switched telephone network (PSTN) or a network such as the Internet. This connection **208** may itself involve a microwave link, a wireless link through a public switched cellular network or a wireless link in a PCS network.

The communication interface **210** is preferably coupled to a communication server **212**. The communication server **212** may be thought of as generalization of a private branch exchange (PBX). The communication server **212** accepts tele-traffic from any variety of sources and provides switchable connections to couple different sources together. For example, the communication server **212** may be implemented as a PBX which receives a set of direct inward dial lines from a central office operated by the public telephone network. The PBX then provides local users with extensions and allows local users to call each other by dialing the last four digits of their telephone numbers. The PBX typically provides an outside line to a user once the user has dialed a nine.

The communication server **212** may also be configured to provide additional types of connections, such as packet based voice and video connections according to the H.323 international standard. In such an embodiment, the communication server **212** provides a gateway function passing calls between the public switched telephone network and a network such as the Internet. The communication server **212** may also provide other communications services such as voice mail, email, fax-mail, call distribution and the like. In systems involving Internet telephony, the communication server may operate only using packet protocols and not include an interface for circuit switched connections.

The communication interface **210** is also coupled to a virtual session server **215**. The virtual session server **215** is coupled to a table structure **225** and an application program **220**. The table structure **225** is preferably implemented as a software entity and may be located in a memory module within the server **215**. The virtual session server **215** may be implemented as a software entity which executes on a hardware platform. The hardware platform of the virtual session server **215** may be designed with an internal architecture similar to the remote unit **100** but is designed to provide a higher computation capacity and to handle multiple users. When supporting a virtual session server, the display monitor **130** is optional as users may control the virtual session server **215** remotely. The control program module **120**, when implemented in the virtual session server **215** provides the server side of the communication protocols discussed in connection with FIGS. **3**–**8**. Hence the remote unit **100** and the virtual session server **215** involve similar architectures and respectively implement the client and server sides of a set of virtual-session-related communication protocols of the present invention.

The application program **220** may execute on the same hardware platform as the virtual session server **215**. In general, both the virtual session server **215** and the application program **220** may be implemented as software modules running on personal computers, workstations, dedicated custom hardware, mainframe, or file servers. For example, the virtual session server **215** may be implemented as a software module running on an UltaSparc™ workstation or file server from Sun Microsystems Inc. The software may be written to execute over a multitasking operating system such as Solaris™ from Sun Microsystems Inc. or WindowsNT™ from Micrsoft Inc. In a first preferred embodiment, the application program **220** includes a distributed database program running on a collection of networked servers such as Sun UltraSparc™ servers. In a second preferred embodiment, the application program may itself be a communication server as provided by an Internet service provider (ISP).

The system **200** is operative to implement a set of virtual session communication protocols according to the present invention. The remote unit **100** establishes a session via the virtual session server **215** to set up a virtual presence with the application program **220**. Preferably, the virtual session server **215** also provides a link to the communication server **212** to provide it access to the virtual session. When the remote unit **100** disconnects from a physical connection **207** or **208**, the virtual session is maintained within the table structure **225**. When the remote unit **100** later wishes to reestablish communication with the application program **220**, the virtual connection server **215** is operative to keep the virtual session active and to allow the user rapid and nearly transparent access to the application program **220**. Similarly, the virtual session also preferably is used to provide a virtual communication link between the communication sever **212** and the remote unit **100**. In some systems, a first virtual session is established between the remote unit **100** and the application program **220**, and a second virtual session is established between the remote unit **100** and the communication server **212**. The details of the operation of the virtual session server **215** and the virtual session protocols are discussed below in connection FIGS. **3**-**8**. Before proceeding to these portions of the detailed description, two embodiments of the system **200** are described.

In a first exemplary embodiment of the system **200**, a mobile worker such as a home-care professional operates the remote unit **100** to establish and maintain a virtual session with the application program **220**. In one embodiment, the application program **220** controls access to a database including complete medical and billing records for individual patients. Depending on working conditions, the home-care professional may require access from a wireless connection such as a cellular connection, or else may be able to communicate via a wireline connection provided within a patient's home. As the home care professional proceeds through a given workflow, the professional will eventually need to communicate with the application program **220**. When this time arrives, the present invention is operative to establish a physical connection between the home-care professional and the application program **220**. The professional need not be aware the physical connection has not been available since the time the virtual session was first established. The virtual session is maintained by the virtual session server **215** and the protocols of the present invention are employed to ensure such a virtual connectivity is provided without the need for the remote unit **100** to be continuously connected to the application program **220**.

In a second exemplary embodiment, the application program **220** is a communication server operated by an ISP. In this example, the remote unit **100** is operated by an Internet user. After the Internet user has remained inactive for a period of time, the connection **208** is terminated. At a later time, when the Internet user clicks on a hyperlink, thus demanding service, a short delay is incurred while the connection is reestablished. The remote unit is provided access without the user needing to reestablish a connection. When the user clicks on a hyperlink, the telephone is rapidly dialed without presenting dialing tones to the user. An authentication packet and a request packet are sent using a low data rate protocol such as one used for line-rate negotiation in modems. The user is authenticated by the server and the request packet is forwarded through the Internet to the Internet site referenced by the hyperlink. While the remote Internet server takes time to respond to the request, a higher line speed is negotiated in the background without burdening the user. Because a home Internet user uses the same analog connection between the user's premises and a network interface, the modem parameters may be preferably saved by the server in the table **225** to accelerate re-negotiation. The user is provided access almost immediately, and the connection is reestablished transparently. Note while this example focuses on an Internet application, the techniques apply to any application whereby a network site is accessed by activating a hyperlink.

As will be discussed below, the virtual session between the remote unit **100** and the virtual session server **215** provides a means to initiate transfers in both an uplink and a downlink direction. The uplink direction is from the remote unit **100** to the virtual session server **215**, and the downlink direction is from the virtual session server **215** to the remote unit **100**. A virtual session is said to exist between the remote unit and the virtual session server **215**. This virtual session may be used to create individual virtual sessions between the remote unit **100** and the application program **220**, and between the remote unit **100** and the communication server **212**. For example, an uplink connection is established, and when a home Internet user has been inactive for a period, the connection is dropped. As discussed above, the connection is reactivated transparently when the user once again activates an Internet link, as in an Internet browser. In the same example, a user may have an email reader program connected through a virtual session. If an email comes in for the user and the virtual session is in place, the email should be rapidly forwarded to the user. To

do this, the user's phone is dialed in a downlink direction dial-out link by the virtual session server **215** via the communication interface **210**. The remote unit preferably suppresses the first ring and examines caller identification data. When the caller identification data indicates the calling party is the virtual session server **215**, the remote unit **100** automatically picks up the call and in this example, accepts the email. If caller identification is not used, a substitute protocol should be employed to assure that connection has been made to the proper application session defined within the virtual session. The substitute protocol preferably involves sending a packet header at the beginning of a call whereby the packet header contains one or more fields which identify associated the application session. Again, the user need not even realize a connection has been reestablished. Instead, the user receives the email message as though the connection had remained continuously active.

Another type of operation may occur when the user of the remote unit **100** is actively connected to the virtual session server **215** and a call comes in directed to the remote user's extension. At this point the call is preferably converted into packets and is sent to the user over the existing connection. In an alternative embodiment, the physical connection is automatically and temporarily dropped and the call is forwarded to the remote user. The virtual connection to the application is maintained through the virtual server. The communications module **125** preferably analyzes caller-identification data to determine the incoming call is a voice call to cause the optional telephone aspect of the remote unit **100** to ring. More details related to the foregoing system operation are discussed in connection with FIGS. **3**–**8** below.

The virtual session server **215** is able to maintain an open logon to the application program **220**. In one embodiment, the virtual session server **215** executes a client-side software which interfaces with the application program **220**. That is, if the application program **220** employs a client-server architecture, the application program **220** will implement a server-side software module which interacts with the client-side software. The server-side program performs database or other server oriented functions, while the client-side software provides a user interface to the user. The remote unit **100** can then control the operations of the virtual session server **215** using standard remote session software. An example of commercially available remote session software is PCAnywhere™ from Semantec Corporation. In another embodiment, the virtual session server executes the client-side software in parallel with the remote unit. In still another embodiment, the remote unit executes the client-side software, and the virtual session server merely provides a connection stream to pass data from the application program **220** to the remote unit **100**. When the virtual session is in a deactivated state, the virtual session server emulates the client-side software as needed to maintain an active session with the application program **220** in the absence of the remote unit **100**. A wide variety of equivalent techniques may be used to allow the virtual session server **215** to maintain a pointer or re-entry point into the application **220** while acting as a proxy agent to maintain the logon for the remote unit **100**. A table structure is preferably used to allow the virtual session server to simultaneously maintain a plurality of logons for a plurality of different remote units.

In some embodiments, the remote unit **100** may need to maintain a plurality of virtual sessions with a plurality of different virtual session servers. For example, an independent contractor may provide home-care services for two distinct health regions. Each health region may use a separate database. The remote unit **100** may then access these separate databases using a first and a second client-side application software module. During the course of a day, the remote unit may need to activate the first or the second client-side application software modules. In such case the remote unit **100** is operative to maintain a table structure similar to the table structure **225**. The table structure maintained by the remote unit links an application software module through an application session to a virtual session. When the first client-side application program demands access to a first database, the virtual session layer software **154** in the remote unit causes a physical connection to be established to support virtual session communications **182** with the first database application program. Likewise, if the second client-side application software module desires to access a second database, the virtual session layer software module **154** activates a physical layer connection back to the second database server. In other applications a single application program may be used which accesses information on more than one virtual session server. In such case a single application program can select the virtual session to activate based on the communications request generated from within the application program. In still other embodiments, a single physical connection **208** or **207** may be used to communicate with the communication interface **210**. The communication server **212** then forwards packets to a first local virtual session server such as the virtual session server **215**. If the received communication packets are destined for a second virtual session server, then the communication server **212** preferably forwards the packets to a remote virtual session server using a network connection such as an Internet connection.

Referring now to FIG. **3**, a method **300** is illustrated to show how the remote unit **100** preferably operates to activate a connection. The method **300** is preferably practiced by the remote unit **100** in support of a virtual session with the virtual session server **215**. A first step **305** of the method **300** involves actions within a workflow process **305**. The workflow process **305** includes the step **305** of the method **300** and also performs other activities to interact with a user's workflow requirements. Control loops from the first step **305** back to the first step **305** via a control path **310**. The workflow, as discussed above, is preferably made up of a menu system and/or a set of interactive screens traversed by a worker in performing a set of tasks. For example, a home-care professional's workflow involves accessing and displaying a patient's medical record, entering a set of data into the medical record, and performing tasks indicated by the doctor's directions as annotated in the medical record. In this example, as the home-care professional moves from one screen to the next, control loops via the control path **310**. The workflow process **305** is an application program which executes on the CPU **105**. The workflow process **305** is preferably implemented as a process running on the CPU **105** in a multitasking operating environment. A multitasking operating environment is one in which multiple programs or processes may execute in parallel by sharing time slots within the CPU **105**. Multitasking operating system software is well known and is readily available. In a multitasking-programming environment a first process may execute in a normal fashion and provide an interface to a user. At the same time a second process may be executed by sharing CPU cycles without the user's intervention or knowledge. In such a case the second process is said to be a background process or is said to perform background processing. At some point in the course of the workflow, a physical layer communication connection will be needed to communicate information between the remote **100** and the application program **220**.

When a step in the workflow process **305** is performed leading up to the need for a physical layer communication connection, control next passes from the first step **305** to a second step **320** via the control path **315**. The control path **315** is activated when the workflow process **305** provides a prediction indicating a physical layer communication connection will subsequently be needed. In some cases the prediction may be provided right when the physical layer communication connection is needed. In other cases, the prediction **315** may be used to initiate background processing to download data which will not be needed until a later time. In menu based systems, the prediction **315** may be learned by observing the workflow habits of a user. The prediction **315** is a function of the application program or workflow **305** and is optional. In the second step **320**, a connection is established in the background. Background processing enables the user to continue interacting with the workflow process **305** while a physical layer communication connection is simultaneously and transparently established. That is, the physical layer communication path is reestablished without inhibiting the user from interacting with the workflow **305**. Hence when control passes via the control path **315** to the step **320**, control preferably simultaneously passes via the control path **317** back into the workflow. The background process is preferably forked as a separate task and two execution flows proceed in parallel by time sharing the CPU **105**. Multitasking is well known in the art and is implemented using interrupt based processing. In alternative embodiments the control path **317** may be deleted and a single control flow may be implemented using the control path **315**. However, this embodiment may require the user to wait for the connection to be established and is hence not deemed to be the preferred embodiment of the method. Other equivalent embodiments set up the communication path transparently by multiplexing the CPU **105**'s computation cycles from within the workflow process or some other process.

Once control has been forked via the control path **315** to the second step **320**, a dialer within the communications module **125** preferably dials to establish a physical layer communication connection with the communication interface **210**. In embodiments using dedicated radio links, the connection may be established over the wireless link **207**. One preferred embodiment of a remote unit **100** incorporates a cellular radio. In this case the dialer dials a telephone number and a connection is established using a public switched cellular telephone network so that the connection is set up on the link **208**. Stationary Internet based embodiments perform the second step **320** by dialing a telephone number using an automatic dialer which dials a land line connection for a modem. In all cases, it is preferred to suppress the dialing tones and line-rate negotiation signals so the connection may be established transparently to the user.

Control next passes from the second step **320** to a third step **325**. In the step **325**, an authentication code is transmitted from the remote unit **100** to the communication interface **210**. This authentication code is then passed to the virtual session server **215**. The virtual session server evaluates the authentication code to determine if access is to be permitted. In a preferred embodiment, the authentication code involves a digital signature as is known in the field of public key cryptography. In a preferred embodiment, all transmissions are encrypted using public key cryptography. Some systems may be implemented using various encryption standards such as secure sockets layer based encryption. The amount of authentication and encryption used in any given embodiment is left to the system designer, but preferably all transactions are encrypted as described above.

Control next passes from the third step **325** to a fourth step **330**. In the fourth step **330**, a session is established/reactivated with the virtual session server **215**. The session is established the first time the method **300** passes control to the step **330**. Subsequently the step **330** is operative to reactivate the session with the virtual session server. When the session between the remote unit **100** and the virtual session server **220** is reactivated, virtual session communications resume. At this point, the virtual session server **215** correlates information stored in the table structure **225** with the connection and provides the remote unit **100** access to the application program **220**. If no data is stored in table structure **225**, access is provided to a default logon screen allowing remote unit **100** to establish a new application session. The virtual session server **215** then populates the table structure **225** to establish a virtual session. The step **330** involves setting up a stream connection between the workflow process **305** and a protocol stack. The protocol stack is operative to read information bits from the stream connection and communicate the bits across an external communication link. Bits received over the external communication link are converted by the protocol stack into information bits to be sent back to the workflow process **305** across the connection stream. Once the appropriate communication processes are configured, control next passes back to the workflow process **305**. Due to the aforementioned forking operation, the passing of control back to the workflow process **305** may have already occurred via the control path **317**. In this case the passing of control from the step **330** to the workflow process is not explicitly performed.

When control loops back from the fourth step **330** to the workflow process **305**, a physical layer communication connection is activated for current or subsequent communication. When the user gets to a point in the workflow where communication with the application program **220** is needed, the connection has already been transparently set up in the background. Hence the user gets the feel of being connected to the application program **220** all the time, where in fact the remote **100** is connected via a physical channel to the application program **220** only a fraction of the time. This virtual connection saves communication resources and money when a toll is charged based on the amount of usage on the link **207** or the link **208**. In some embodiments, the fourth step **330**, or an execution thread within the workflow **305** is operative to upload or download information in the background. This way the user has ready access to data contained in the application program **220**, but in general a shorter connect time is required. While with prior art systems it is burdensome for a user to connect to a central server and download and upload information, with the virtual session of the present invention the user need not even be aware this process is occurring. Rather the user feels as though he or she is continuously connected with a fast connection because the data needed at a given point in the workflow is already available locally or has been uploaded in the background transparently without user intervention. In systems where server synchronization is an issue, file semaphores and/or direct active sessions not employing uploading and/or downloading of records may be used.

Based on another point in the workflow, another prediction is made to predict when the communication channel will not be needed for some time. For example, it may be known, based on the workflow, the home-care professional will next perform a sequence of tests and enter data into a screen displayed on the remote unit. Only at a later time will the workflow come to a point where this information is to be uploaded to the application program **220**. When such a prediction is made, control passes from the first step **305** via the control path **318** to a fifth step **335**. The fifth step **335** is operative to deactivate the connection established over the link **207** or the link **208**. The step **335** may optionally involve forking a separate execution thread or otherwise accessing a separate process in a multitasking environment. Alternatively, the fifth step **335** may be performed by executing a set of instructions in the workflow process **305**. At a later time, a prediction may be made indicating the link **207** or **208** needs to once again be activated, whereby control again passes over to the second step **320** via the control path **315**. It should be noted different systems will typically set their prediction times according to the economic conditions involved. For example, in some systems the first minute of connection time may cost five times as much as all subsequent minutes. In this case predictions would be preferably set according to a criterion to minimize cost by not establishing and terminating connections more often than necessary. If a flat rate were charged per minute connections would be set-up and terminated more often. If automatic uploading and downloading is performed in the background, a very efficient use of airtime can often be achieved while presenting the user with the appearance of being continuously connected to the application program **220**.

Referring now to FIG. **4**, a method **400** of establishing a communication link with low delay is illustrated. The method **400** may be practiced by both the remote unit **100** and the communication interface **210**. This method is most applicable to systems involving modems whereby digital data is transferred over an analog channel requiring receiver training. Receiver training involves transmitting data sequences through a channel and allowing a receiver to adjust its receiver parameters. Receiver parameters include echo canceller and equalizer filter coefficients. Most systems also adjust their data rates and signal constellations based on observed conditions. In modems, this entire process is known as line-rate negotiation. Prior art systems involving receiver training are tedious to use because they force the user to wait while the receiver is trained. Most systems play the training signals though a speaker to allow the user to hear the training process. This lets the user know what the computer is doing for the duration of the delay. The method **400** improves upon this prior art solution by allowing the user to gain almost immediate access without a significant delay.

In a first step **405**, a protocol stack or other process practicing the method **400** receives a communications request from a user program. For example, this occurs when a user clicks on an icon to initiate the establishment of an Internet connection. Control next passes to a step **410** where the connection is initiated. This step typically involves an automatic dialer dialing the number of an Internet service provider (ISP). The ISP software may be implemented as a communication server application corresponding to the application program **220**. In this case access to the application program is governed by the virtual session server **215**.

Control next passes from the second step **410** to a third step **415**. In the third step **415** an authorization sequence is exchanged. In a preferred embodiment public key cryptography involving digital signatures and keys is used. Embodiments involving a virtual session server **215** either set up a session or activate an existing session during the third step **415**. Control next passes from the third step **415** to a fourth step **420**. In the fourth step **420**, one or more initial application layer data packets are transmitted across the connection using a low speed protocol. A low speed protocol is used by the transmitter and receiver when performing line-rate negotiations. For more details of line-rate negotiation protocols, see, for example, the V.34 and V.90 standards from the International Telecommunications Union. In the present invention, the low speed protocol is used to transmit application layer data before the line-rate negotiation procedures have completed. This avoids the need for the user to wait for line-rate negotiation to complete before being able to access a communication path.

Control next passes from the fourth step **420** to a fifth step **425**. In the fifth step **425**, initial data is displayed. Software located locally in the remote unit **100** preferably contains high-volume graphics related data so that the initial data exchange of the step **420** only requires a small amount of data to be transferred. For example, the user logs onto the Internet and almost immediately sees a screen of information indicating the user is connected and the system is ready to accept inputs. This is made possible by displaying locally held screens of graphical data and allowing a small amount of specific information such as time, date, and headlines to be received and displayed. If the user then immediately clicks on a link, an application layer request packet is sent using the line-rate negotiation protocol's data format. This allows the user to immediately begin making requests before the line-rate negotiations have completed. In many cases the user will pause and read the headline information, giving the system even more time to perform line-rate negotiation in the background.

Control next passes from the fifth step **425** to a sixth step **430**. In the sixth step **430**, a background process is forked to perform line-rate negotiation. Line-rate negotiation is allowed to proceed in the background while the user is reading the information provided on the initial display of the step **425**. Likewise, if the user had rapidly clicked on a link, a request packet is sent out and while the server is responding to the request, the background line-rate negotiation may proceed. The step **430** is operative to perform line-rate negotiation so as to set up a high-speed connection for subsequent higher volume data transfers. In embodiments involving a virtual session server **215**, the user's line speed parameters may be stored in the table structure **225**. For example, if the user is an Internet user and the application program **220** is an ISP, the user will often dial in from the same location. Thus parameters derived in a previous activation of a communication channel will be either identical or similar to those used in a current activation. Hence the sixth step **430** optionally involves accessing from the table **225** a set of starting parameters derived from the activation of the communication channel. If communication is needed before the line-rate negotiation has completed, communication preferably proceeds at the highest rate negotiated up to that point.

Once the line-rate negotiation process of the step **430** has completed, control passes to a seventh step **435**. In the step **435**, communication is able to proceed at full speed. In most cases where this method is implemented, the user will get the full benefit of being connected almost immediately without the normal delay associated with prior art systems. This is so because initial low-volume data is allowed to pass through the channel before the line-rate negotiation has completed. Line rate negotiation then proceeds in the background in parallel with other activities such as the user reading headline information or a distant server accessing data and responding to the initial data request packet sent across the Internet. This technique is useful when a user is

maintaining a virtual session with a remote server because it is imperative to allow the user to appear to be connected without having to experience delays when accessing data. The method **300** and the method **400** may be performed together in a complementary fashion to make the virtual session appear to be constantly available.

The method **400** may be practiced by the remote unit **100** and the virtual session server **215**. When the remote unit **100** initiates the method, the virtual session server **215** executes steps **410**, **415**, **420**, **430** and **435**. When the virtual session server **215**, the application program **220**, or the communication server **212** initiates the method, one or a combination of these servers practice the steps **405**, **410**, **415**, **420**, **430**, and **435**. The first step **405** involves, for example, receiving a communication request such as a telephony call or an email for the remote unit **100**. In some systems, the first step **405** may involve a request generated from within the application program **220**.

Referring now to FIG. **5**, a method **500** of establishing and operating a virtual session is illustrated. For example, the method **500** establishes a virtual session between the remote unit **100** and the virtual session server **215**. The method **500** is practiced by both the remote unit **100** and the virtual session server **215**. In a first step **505** a first physical layer communication connection is established with a remote entity. If the method is practiced by the remote unit **100**, then the remote entity typically corresponds to the virtual session server **215**. The virtual session may be used to support virtual sub-sessions between the remote unit **100** and the application program **220**. Also, a virtual sub-session may be established between the remote unit **100** and the communication server **212**. For the purposes of discussion herein, all of these virtual sessions will be referred to simply as virtual sessions. If the method **500** is practiced by the virtual session server **215**, then the remote entity typically corresponds to the remote unit. The step **505** may be activated according to the prediction **315**, and the step **505** may use the method **400** to allow the connection to be set up with very low delay.

Control next passes from the first step **505** to a second step **510**. In the second step **510** a session is established with the remote entity. In a preferred embodiment, this involves exchanging password information and agreeing upon a set of keys to encrypt data transacted in the session. Also, the virtual session server **215** preferably sets up a table entry in the table structure **225**. The table entry indicates the presence of a virtual session. The table entry may include modem parameters as discussed in connection with FIG. **4**. Also, the virtual session as set up in the table entry links the remote unit to a user identification and a password as presented to the application program **220**. For example, a user name and a password may be used as user authentication parameters. Preferably public key encryption is used to encrypt all information so the password sent from the remote unit **100** to the application program **220** cannot be effectively intercepted. The remote unit **100** also preferably sets up a virtual session data structure to hold similar information related to the virtual session. Once the virtual session has been set up, the remote unit **100** may access the application program **220**. Also, the remote unit **100** may optionally access the communication server **212** for communication services.

Control next passes from the second step **510** to a third step **515**. In the third step **515**, the physical connection established in the first step **505** is dropped. Meanwhile the virtual session data structures and table entries established in the second step **510** are retained. The session is allowed to proceed while no physical layer connection exists. That is, the step **510** is operative to set up a table structure including one or more data structures which allows a virtual session to be maintained in memory while other activities occur. Hence a passive background thread of execution passes from the step **510** to a passive step **540** whereby the virtual session is maintained. This allows the remote unit **100** to stand by or be used for steps of the workflow process **305** not requiring communication with the application program **220**. Once the user needs to communicate with the application program **220**, or when a prediction **315** is made, control next passes from the third step **515** to a fourth step **520**. The step **520** is operative to reestablish a second physical layer communication connection to allow communication to proceed once again using the session established in the second step **510**. This connection reestablishment may be performed in response to the prediction **315** and may use the low-delay connection establishment technique of the method **400**.

In some embodiments, the present invention involves using distinct and separate communications media to perform the step **505** and the step **520**. For example, a mobile worker may call in from home to set up the virtual session in the step **510** using the first physical layer communication connection established in the step **505**. Later in the day, the worker may call in from a restaurant while catching up on some records keeping. This second use of the virtual session involves use of the second physical layer communication connection which in this example is a wireless connection different from the landline connection used to initiate the session from home earlier in the day. At a still later time, the worker may call in from a patient's home via a third physical layer communication connection while performing home-care services. If modem starting parameters have been stored in table **225**, they are preferably updated whenever the communication connection is changed. Hence the virtual session of the present invention enables a mobile worker to continue communications via the most expedient and/or economical means without causing the user to have to reestablish a communication connection. Preferably, when the remote unit **100** is connected to a communications source via the connector **127**, the remote unit **100** automatically detects this connectivity and makes use of it for subsequent virtual-session communications. That is, the present invention contemplates the availability of various forms of "pigtail" connectors being available so the remote unit **100** can operate in a "plug-and-play" fashion. Pigtails may be supplied to allow the remote unit to connect to the PSTN, the Internet, or to another computer via a universal serial bus, for example.

Control next passes from the fourth step **520** to a fifth step **525**. In the fifth step **525** an authorization sequence is exchanged. This is preferably implemented using public key encryption and digital signatures. Some embodiments may be developed which do not implement the fifth step **525**, but preferred embodiments do make use of user authentication. After the fifth step **525** has completed, the session is resumed in a sixth step **530**. Over the course of the virtual session, control may loop back to the third step **515** as many times as the virtual session is activated with a new physical connection. When the sixth step **530** is entered, the virtual session is once again activated so that the passive step **540** also passes control to the sixth step **530**. In a minimal implementation of the method, no looping occurs and the method terminates after the first pass through the sixth step **530**.

Referring now to FIG. **6**, a method **600** practiced by the virtual session server **210** is illustrated. In a first step **605**, a first physical layer communication connection is established for communicating with the remote unit **100**. Control next passes to a second step **610** whereby a set of authorization parameters are accepted and authenticated. As discussed in connection with FIG. **5**, the authentication parameters preferably include the exchange of public keys which include a digital signature in accordance with public key cryptography. Control next passes to a third step **615** where a user identification and a password are passed by the virtual session server **215** to the application program **220** on behalf of the remote unit **100**. As discussed in connection with FIG. **5**, the user identification and the password to be presented to the application program **220** are preferably transmitted in encrypted form. Once the application program **220** authenticates the user identification and password needed to gain access, the virtual session server **215** enters an entry into the table structure **225** to hold a set of session parameters. The session parameters include the user identification, a session identifier, encryption data and possibly other data such as modem starting parameters. Once the session has been logged into the table, the user may use it to communicate with the application program **220**.

Control next passes from the fourth step **620** to a fifth step **625**. In the fifth step **625** the physical layer connection is dropped. This step is performed when the remote unit does not currently require communications with the application program **220**. In step **650** the virtual server maintains the application session while the physical connection is disconnected. At a later time, when the user needs access to the application program or when the prediction **315** is made, control next passes to a sixth step **630**. In the sixth step **630** a second physical layer connection is established to allow communication between the remote unit **100** and the application **220** to resume. As discussed in connection with FIG. **5**, the second physical layer connection may involve a different communication path and/or medium as was used for the first physical layer connection. That is, a plurality of communications media are preferably supported to allow the user to call in via different means, for example via a wireless or a wireline connection. The step **630** may be initiated due to actions at the remote unit **100** or in response to events occurring in the server. For example, the communication server **212** may receive a call for the remote unit. Alternatively an email may be received which needs to be forwarded to the remote unit. In such a case, the sixth step **630** optionally involves sending a caller identification packet to let the remote unit know what type of communication, such as a voice telephony call, an email, or a fax, is inbound. A caller identification packet is a sequence of information bits sent across a communication connection identifying the calling party of the connection. In standard telephone systems, the caller identification packet is transmitted between the first and second rings when the telephone call is being set-up. More details relating to communications initiated by the virtual session server **215** back to the remote unit **100** are discussed in connection with FIG. **7**.

Once the second physical layer communication connection is established in the sixth step **630**, possibly according to the method **400**, control next passes to a seventh step **635**. In the seventh step **635**, authorization codes are verified similarly to the second step **610**. Once the user codes have been verified to be correct, control next passes to an eighth step **640** whereby communication once again resumes using the previously established virtual session.

Referring now to FIG. **7**, a method **700** of processing communication requests in a virtual session is illustrated. This method is preferably practiced by the virtual session server **215** simultaneously with the method **600**. In a first step **705** a virtual session is established between the virtual session server **215** and the remote unit **100** as discussed in connection with FIGS. **5** and **6**. At some later time, while the virtual session is active, the communication server **212** receives an incoming communication request for the remote unit **100**. Because the virtual session server **215** practices the method **500** and/or the method **600**, depending on the time of arrival of the communication, the remote unit **100** may or may not be physically connected to the virtual session server **215** by a physical communication link. Hence when the communication is received, control passes from the step **705** based on a decision **710** which determines whether a physical connection currently exists to the remote unit **100**.

If the virtual session is presently in a state whereby the physical connection has been disconnected, control passes from the first step **705** to a second step **715**. In the second step **715** the communication request is accepted by the communication server **212** through a direct connection or via the communication interface **210**. Control next passes to a third and optional step **720** whereby a specific caller identification packet is associated with the communication type. For example, if the communication involves a telephone call a fist caller identification packet is sent identifying an extension used for telephone calls. If the communication involves an email, a second caller identification packet is sent identifying an extension used for email. On the other hand, if the communication comes from the application program **220**, still another caller identification packet is sent. When this optional use of a caller identification packet is employed, the remote unit **100** has the information needed to properly and immediately respond to an incoming call as discussed in connection with FIG. **8**. If a call is received by the remote unit from a source other than the virtual connection server **215**, the caller identification information will identify the call as not being associated with the virtual session. Control next passes to a fourth step **725** whereby an automatic dialer responds to communication requests and the communication is forwarded to the remote unit **100**.

Another situation arises when the communication request arrives while the remote unit **100** and the virtual session server **215** are currently connected by an existing physical channel. According to one mode of processing, control stays in the first step **705** while communication proceeds in an active phase of a virtual session. When the communication request arrives, the communication server **212** signals to the virtual session server that a new call has arrived for the remote unit **100**. The virtual session server then causes the existing physical layer communication connection to be dropped and thus control passes from the first step **705** to the second step **715** as in the foregoing discussion. In another mode of processing, the existing physical layer communication connection is left in tact and control passes from the first step **705** to a fifth step **745**. In the fifth step **745**, the communication is packetized, and in a sixth step **750** the communication packets are passed along the existing physical layer communication connection. Control continues to loop back from the sixth step **750** to the fifth step **745** during the course of the communication. In this mode of operation the existing physical layer communication connection is shared to provide the remote user with a means to stay connected to the application program **220** and communicate at the same time. In this case the physical layer is time shared by the virtual session to allow multiple modes of communications to proceed in parallel.

Note the method **700** provides the user of the remote unit **100** with a virtual presence in the work place while actually being remote. Independent of whether the remote unit is

presently actively connected to the virtual session server, the communication request may be forwarded to the remote unit **100** making the remote user appear to be present in the office at all times. The only time the remote unit **100** would not be reachable is when it is engaged in a communication with an entity other than the virtual connection server. This problem may be mitigated by allowing the remote unit to only be reachable through the access number provided by the communication server **212**. If a call is placed from the remote unit to another point, this call too may be routed through the communication server **212**. Systems which do allow the remote unit to make calls outside the virtual session may preferably employ voice mail at the communication server **212**. When the remote unit again becomes available, the virtual session server **215** may forward the communication to the remote unit according to the method **700**. Remote units may also be designed using call-waiting concepts whereby the virtual session may be re-activated by interrupting another call.

Referring now to FIG. **8**, an optional method **800** practiced by the remote unit **100** is illustrated. This method is practiced when a virtual session exists, the remote unit and the virtual session server are presently not connected via a physical channel, and a communication request is to be forwarded to the remote unit **100**. In a first step a first ring signal is detected. Optionally, the first ring signal is suppressed so the user will not hear it. In some systems a vibrational first ring signal may be allowed to pass through to notify the user of an incoming communication. In still other embodiments, the remote unit may be programmed to sound a normal ring on the first ring signal. In some embodiments the ring signal will not be a traditional telephone ring signal but will in general be any signal indicative of an incoming communication request.

In current systems, a caller identification packet is presented to the called party after the first ring signal. Hence identification of the calling party becomes available at this time. After the first ring signal, control passes from the first step **805** to a second step **810**. In the second step **810**, the caller identification information is evaluated. Note it would be preferable to accept the caller identification data before the first ring, and the present invention contemplates systems whereby the virtual session server **215** signals to the PSTN to provide a caller identification packet before the first ring. This service does not appear to be available from telephone service providers at this time. Embodiments also comprehended by the present invention include systems whereby the remote unit **100** immediately picks up an incoming call and caller identification information is sent by the virtual connection server **215** over the connection identifying the call-type. During the second step **810**, the caller identification packet is evaluated. As discussed in connection with the third step **720** of the method **700**, the virtual session server **215** sends out a caller identification packet to identify the type of incoming call. For example, different caller identification packets indicate whether the incoming call is an email, a voice telephone call, or a communication from the application program **220**.

Control next passes from the second step **810** to a third step **815**. In the third step **815**, an application layer program is selected to process the incoming call. In the foregoing examples, an application may be launched to accept an email message, a voice telephone call, or to accept a communication from the application program **220**. If the communication is an email, it may be desirable to pop a mailbox icon on the screen or to produce a speech signal stating "you've got mail." If the incoming call is a voice call, it may be desirable to allow the telephone to ring like a normal telephone. If the communication is from the application program **220**, it may be desirable to update data located in the screens of the workflow or otherwise signal the presence of new data. Once the appropriate application has been launched to handle the incoming communication, control next passes to a fourth step **820** whereby communication session is reactivated and the communication is processed. For example, one or more packets of information may be received and related information such as an email message may be displayed. Also, a telephone call may be allowed to proceed or a set of information may be downloaded from the application program **220**.

Although the present invention has been described with reference to specific embodiments, other embodiments may occur to those skilled in the art without deviating from the intended scope. For example, other forms of communications such as fax messages may be accepted and displayed by the remote unit **100**. Also, the present invention may be used for applications other than mobile workers and Internet users. Virtual sessions according to the present invention are applicable to any situation where a continual connectivity is required but the cost to remain continuously connected is high. Vehicle computers with cellular radio based Internet connections are an example. In such systems, the remote unit **100** may be a vehicle-mounted computer or may include a connection to a vehicle-mounted computer. Also, virtual sessions according to the present invention are applicable to any situation where the user should not be burdened with the need to upload and/or download data and go through connection and disconnection procedures. Therefore, it is to be understood that the invention herein encompasses all such embodiments which do not depart from the spirit and scope of the invention as defined in the appended claims.

What is claimed is:

1. For use in controlling a virtual session, a method comprising:

establishing a virtual session with a remote entity, the virtual session being instantiated to support an application layer program;

placing the virtual session in an inactive state;

receiving an incoming call;

reading a set of caller identification information from said call;

checking the set of caller identification information to see if it identifies the application layer program; and

if the step of checking results in a match, activating the virtual session.

2. The method of claim **1**, wherein the virtual session is also established to support a second application program, the step of checking involves determining whether any of the application program or the second application are identified by the set of caller identification information, and if the step of checking results in a match, additionally selecting which of the application programs to launch.

3. The method of claim **1**, wherein a second virtual session is also established to support a second application program, the step of checking involves determining whether any of the application program or the second application are identified by the set of caller identification information, and if the step of checking results in a match, additionally checking which of virtual sessions to activate and which the application programs to launch, and in the step of activating, activating the identified virtual session and application.

4. For use in controlling a virtual session, a method comprising:

establishing a virtual session with a remote entity, the virtual session being instantiated to support at least one application layer program;

placing the virtual session in an inactive state;

receiving a ring signal via a communication path;

reading a set of information that follows the ring signal;

checking the set of information to see if it identifies the application layer program; and

if the step of checking results in a match, activating the virtual session and coupling the at least one application to the remote entity thereby.

5. For use in controlling a virtual session, a method comprising:

establishing a virtual session with a remote entity, the virtual session being instantiated to support at least one application layer program;

placing the virtual session in an inactive state;

receiving an incoming communication request;

reading a set of information associated with said incoming communication request;

checking the set of information to see if it identifies the application layer program; and

if the step of checking results in a match, activating the virtual session and coupling the at least one application to the remote entity thereby.

6. For use in controlling a virtual session on a server, a method comprising:

establishing a virtual session with a remote unit, the virtual session being instantiated to support at least one application layer program;

placing the virtual session in an inactive state;

dialing a telephone number corresponding to said remote unit to cause a ring signal followed by a set of application-program identifying caller identification data to be delivered to said remote unit; and

placing the virtual session back into the active state and transferring data between the application and the remote unit via the virtual session in response to said step of dialing.

7. For use in controlling a virtual session on a server, a method comprising:

establishing a virtual session with a remote unit, the virtual session being instantiated to support at least one application layer program;

placing the virtual session in an inactive state;

sending a signal indicative of an incoming communication request and an application-program identifying packet to said remote unit, said application-program identifying packet identifying an application program that needs to resume a virtual session and communicate with said remote unit; and

placing the virtual session back into the active state and transferring data between the application and the remote unit via the virtual session in response to said step of sending.

8. A method of pausing a modem connection and reconnecting a telephone modem to a far end modem with a reduced delay by reducing a time associated with line rate re-negotiation, the method comprising:

initializing a first communication connection with the far end modem over a wireline telephone communication channel using the telephone modem, the initializing being performed at least partially by performing a line rate negotiation sequence with the far end modem to derive a first set of modem parameters to be used to support communications over the wireline telephone communication channel;

storing the set of modem parameters in a memory;

communicating at a negotiated data rate via the first communication connection with the far end modem using at least one of the first set of modem parameters;

receiving a first indication associated with a request to temporarily suspend modem communications;

in response to the first indication, terminating the first communication connection;

receiving a second indication associated with a request to resume modem communications;

in response to the second indication, accessing from the memory a plurality of the first set of modem parameters and using the accessed modem parameters as modem starting parameters in a line rate re-negotiation sequence, and initializing a second communication connection to the far end modem over the wireline telephone communication channel using the telephone modem, the initializing of the second communication connection being performed at least partially by performing the line rate re-negotiation sequence to derive a re-negotiated set of modem parameters to be used to support communications over the second communication connection of the wireline telephone communication channel; and

communicating at a re-negotiated data rate via the second communication connection using at least one of the re-negotiated set of modem parameters;

wherein the accessed modem starting parameters are used to accelerate the derivation of the re-negotiated set of modem parameters with respect to the time required to derive the first set of modem parameters, and whereby a setup delay time associated with the initialization of the second communication connection is shorter than a setup delay time associated with the initialization of the first communication connection.

9. The method of claim **8**, wherein the line rate negotiation is performed at least partially in accordance with the V.34 modem standard.

10. The method of claim **8**, wherein the line rate negotiation is performed at least partially in accordance with the V.90 modem standard.

11. The method of claim **8**, wherein the memory includes a table structure.

12. The method of claim **8**, wherein the modem parameters comprise adjustable echo cancellation coefficients that are adjusted in response to a training signal sent through the wireline telephone communication channel.

13. The method of claim **8**, wherein the modem parameters comprise adjustable equalizer coefficients that are adjusted in response to a training signal sent through the wireline telephone communication channel.

14. The method of claim **8**, wherein the modem parameters comprise adjustable signal constellation configuration parameters that are adjusted in response to a training signal sent through the wireline telephone communication channel.

15. The method of claim **8**, wherein:

a communication session is established in conjunction with the first communication connection;

the communication session is placed in an inactive state substantially in conjunction wit the termination of the first communication connection; and,

the communication session is placed back into an active state substantially in conjunction with the initialization of the second communication connection.

16. The method of claim **8**, wherein the communication session supports a peer-to-peer Internet connection software layer above the physical layer, and the Internet session is between a client software layer co-located with the telephone modem and a server software layer co-located with the far end modem.

17. The method of claim **8**, wherein the first indication is generated in response to a user input to a graphical user interface of a computer program.

18. The method of claim **8**, wherein the first indication is generated in response to a user input to a work flow as presented by a graphical user interface of a computer program.

19. The method of claim **8**, wherein the second indication is generated in response to a user action.

20. The method of claim **8**, wherein the second indication is generated in response to a user input to a graphical user interface of a computer program.

21. The method of claim **8**, wherein the second indication is generated in response to a user input to a work flow as presented by a graphical user interface of a computer program.

22. The method of claim **8**, wherein first indication is generated in response to a first user action and the second indication is generated in response to a second user action.

23. The method of claim **22**, wherein the first user action is associated with a call waiting signal and the second user action is associated with the user hanging up a voice telephone call.

24. The method of claim **22**, wherein the first indication is generated in response to first user action indicating a desire to pause communication in order to free up the wireline telephone communication channel to be used for a voice telephone call, and the second indication is generated in response to a second user action indicative of the desire to resume modem communication with the far end modem.

25. The method of claim **8**, further comprising:

establishing a communication session between a first software layer coupled to the telephone modem and a second software layer coupled to the far end modem, wherein the communication session assumes an active state and supports communication between the first and second software layers via the first communication connection;

placing the communication session into an inactive state substantially in conjunction with the first indication; and

placing the communication session back into the active state substantially in conjunction with the initializing of the second communication connection, and using the communication session to support communication between the first and second software layers via the second communication connection.

26. The method of claim **25**, herein the communication session involves an authentication parameter.

27. The method of claim **25**, wherein:

the communication session corresponds to a communication services logon session that involves an authentication parameter; and,

the first software layer corresponds to client-side software and the second software layer corresponds to server-side software.

28. The method of claim **25**, wherein the communication session is placed into the inactive state in response to a first user action indicating a desire to pause communication in order to free up the wireline telephone communication channel to be used for a voice call.

29. The method of claim **25**, wherein the communication session is placed back into the active state in response to a second user action indicating a desire to resume the communication session with the software layer coupled to the far end modem.

30. The method of claim **25**, wherein the authentication parameter is associated with an Internet-related protocol above the physical layer to support client-server communications using the Internet.

31. The method of claim **25**, wherein the communication session is at least partially transacted in accordance with an Internet-related protocol above the physical layer in order to support client-server communications using the Internet.

32. The method of claim **8**, further comprising:

establishing an application session between a first application software layer coupled to the telephone modem and a second application software layer coupled to the far end modem, wherein the application session assumes an active state and supports communication between the first and second application software layers via the first communication connection;

placing the application session into an inactive state substantially in conjunction with the terminating; and

placing the application session back into the active state substantially in conjunction with the initializing the second communication connection, and using the application session to support communication between the first and second application software layers via the second communication connection.

33. The method of claim **32**, wherein the application session involves an authentication parameter.

34. The method of claim **32**, wherein the application session involves an authentication parameter corresponding to an application logon session conducted between a client and server across the Internet.

35. The method of claim **32**, wherein the communication session is placed into the inactive state in response to a user action indicating a desire to pause communication in order to free up the wireline telephone communication channel to be used for a voice call.

36. The method of claim **32**, wherein the communication session is placed back into the active state in response to a user action indicating a desire resume the communication session wit the software layer coupled to the far end modem.

37. The method of claim **8**, wherein the telephone modem is an analog client modem and the far end modem is a digital ISP modem.

38. The method of claim **8**, wherein the telephone modem is an analog client modem and the far end modem is a digital ISP modem, and both modems support the V.90 communication protocol.

39. The method of claim **8**, wherein the far end modem is an analog client modem and the telephone modem is a digital ISP modem, and both modems support the V.90 communication protocol

40. The method of claim **8**, wherein the re-negotiated data rate can be re-negotiated to be the same as the negotiated data rate.

41. A device, comprising;

a telephone modem that supports at least one of PCM pulse amplitude data transmission and reception, and includes a coupling for connecting to a telephone line;

a memory;

a computer-readable storage medium;

a software instantiated on the computer-readable storage medium, the software including:

a first function that causes a first communication connection with a far end modem to be initialized over a wireline telephone communication channel using the telephone modem, the initializing being performed at least partially by performing a line rate negotiation sequence with the far end modem to derive a first set of modem parameters to be used to support communications over the wireline telephone communication channel;

a second function that causes the set of modem parameters to be stored in the memory;

a third function that causes communication to proceed at a negotiated data rate via the first communication connection with the far end modem using at least one of the first set of modem parameters;

a fourth function, responsive to a first indication associated with a request to temporarily suspend modem communications, to causes the first communication connection to be terminated;

a fifth function, responsive to a second indication associated with a request to resume modem communications, to causes a plurality of the first set of modem parameters to be accessed from memory, the accessed modem parameters to be used as modem starting parameters in a line rate re-negotiation sequence, and a second communication connection to be initialized to the far end modem over the wireline telephone communication channel using the telephone modem, wherein the initializing of the second communication connection is performed at least partially by performing the line rate re-negotiation sequence to derive a re-negotiated set of modem parameters to be used to support communications over the second communication connection of the wireline telephone communication channel, wherein the plurality of the stored modem starting parameters are used to accelerate the derivation of the re-negotiated set of modem parameters with respect to the time required to derive the first set of modem parameters, and whereby a setup delay time associated with the initialization of the second communication connection is shorter than a setup delay time associated with the initialization of the first communication connection; and

a sixth function that causes communication to resume at a re-negotiated data rate via the second communication connection using at least one of the re-negotiated set of modem parameters.

42. The device of claim **41**, wherein the line rate negotiation is performed at least partially in accordance with the V.34 modem standard.

43. The device of claim **41**, wherein the line rate negotiation is performed at least partially in accordance with the V.90 modem standard.

44. The device of claim **41**, wherein the memory includes a table structure.

45. The device of claim **41**, wherein the modem parameters comprise adjustable echo cancellation coefficients that are adjusted in response to a training signal sent through the wireline telephone communication channel.

46. The device of claim **41**, wherein the modem parameters comprise adjustable equalizer coefficients that are adjusted in response to a training signal sent through the wireline telephone communication channel.

47. The device of claim 41, wherein the modem parameters comprise adjustable signal constellation configuration parameters that are adjusted in response to a training signal sent through the wireline telephone communication channel.

48. The device of claim 41, wherein:

a communication session is established in conjunction with the first communication connection;

the communication session is placed in an inactive state substantially in conjunction with the termination of the first communication connection; and,

the communication session is placed back into an active state substantially in conjunction with the initialization of the second communication connection.

49. The method of claim 48, wherein the communication session supports a peer-to-peer Internet connection software layer above the physical layer, and the Internet session is between a client software layer co-located with the telephone modem and a server software layer co-located with the far end modem.

50. The device of claim 41, wherein the first indication is generated in response to a user action.

51. The device of claim 41, wherein the first indication is generated in response to a user input to a graphical user interface of a computer program.

52. The device of claim 41, wherein the first indication is generated in response to a user input to a work flow as presented by a graphical user interface of a computer program.

53. The device of claim 41, wherein the second indication is generated in response to a user action.

54. The device of claim 41, wherein the second indication is generated in response to a user input to a interface of a computer program.

55. The device of claim 41, wherein the second indication is generated in response to a user input to a work flow as presented by a graphical user interface of a computer program.

56. The device of claim 41, wherein the first indication is generated in response to a first user action and the second indication is generated in response to a second user action.

57. The device of claim 56, wherein the first user action is associated with a call waiting signal and the second user action is associated with the user hanging up a voice telephone call.

58. The device of claim 56, wherein the first indication is generated in response to a first user action indicating a desire to pause communication in order to free up the wireline telephone communication channel to be used for a voice telephone call, and the second indication is generated in response to a second user action indicating a desire to resume modem communication with the far end modem.

59. The device of claim 41, further comprising:

a seventh function that causes a communication session to be established between a first software layer coupled to the telephone modem and a second software layer coupled to the far end modem, wherein the communication session assumes an active state and supports communication between the first and second software layers via the first communication connection;

a eighth function that causes the communication session to be placed into an inactive state substantially in conjunction wit the terminating; and

a ninth function that causes the communication session to be placed back into the active state substantially in conjunction with the initializing of the second communication connection, and using the communication session to support communication between the first and second software layers via the second communication connection.

60. The device of claim 59, wherein the communication session involves an authentication parameter.

61. The device of claim 59, wherein the communication session corresponds to a communication services logon session that involves an authentication parameter, the first software layer corresponds to client-side software and the second software layer corresponds to server-side software.

62. The device of claim 59, wherein the communication session is placed into the inactive state in response to a first user action indicating a desire to pause communication in order to free up the wireline telephone communication channel to be used for a voice call.

63. The device of claim 62, wherein the communication session is placed back into the active state in response to a second user action indicating a desire to resume the communication session with the software layer coupled to the far end modem.

64. The method of claim 62, wherein the authentication parameter is associated with an Internet-related protocol above the physical layer to support client-server communications using the Internet.

65. The method of claim 62, wherein the communication session is at least partially transacted in accordance with an Internet-related protocol above the physical layer in order to support client-server communications using the Internet.

66. The device of claim 41, further comprising:

an seventh function that causes an application session to be established between a first application software layer coupled to the telephone modem and a second application software layer coupled to the far end modem, wherein the application session assumes an active state and supports communication between the first and second application software layers via the first communication connection;

a eighth function that causes the application session to be placed into an inactive state substantially in conjunction with the terminating; and

a ninth function that causes the application session to be placed back into the active state substantially in conjunction with the initializing the second communication connection, and using the application session to support communication between the first and second application software lay is via the second communication connection.

67. The device of claim 66, wherein the application session involves an authentication parameter.

68. The device of claim 66, wherein the application session involves an authentication parameter corresponding to an application logon session conducted between a client and server across the Internet.

69. The device of claim 66, wherein the application session is placed into the inactive state in response to a user action indicating a desire to pause communication in order to free up the wireline telephone communication channel to be used for a voice call.

70. The device of claim 66, wherein the application session is placed back into the active state in response to a user action indicating a desire resume the communication session with the software layer coupled to the far end modem.

71. The device of claim 66, wherein the memory comprises the computer-readable storage medium.

72. The device of claim 41, wherein the memory comprises nonvolatile memory.

73. The method of claim **41**, wherein the telephone modem is an analog client modem and the far end modem is a digital ISP modem, and both modems support the V.90 communication protocol.

74. The method of claim **41**, wherein the far end modem is an analog client modem and the telephone modem is a

digital ISP modem, and both modems support the V.90 communication protocol.

75. The method of claim **41**, wherein the re-negotiated data rate can be re-negotiated to be the same as the negotiated data rate.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 6,574,239 B1
DATED : June 3, 2003
INVENTOR(S) : Dowling et al.

Page 1 of 1

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Column 27,
Line 66, delete "wit" and insert -- with --

Column 28,
Line 57, delete "herein" and insert -- wherein --

Column 29,
Line 48, delete "wit" and insert -- with --

Column 30,
Lines 21 and 25, delete "causes" and insert -- cause --

Column 31,
Line 14, delete "method" and insert -- device --
Line 32, delete "a interface" and insert -- a graphical user interface --
Line 62, delete "wit" and insert -- with --

Column 32,
Lines 21 and 25, delete "method" and insert -- device --
Line 46, delete "lay is" and insert -- layer --

Column 33,
Lines 1 and 5, delete "method" and insert -- device --

Column 34,
Line 3, delete "method" and insert -- device --

Signed and Sealed this

Twenty-ninth Day of March, 2005

Jon W. Dudas

JON W. DUDAS
Director of the United States Patent and Trademark Office

EXHIBIT 2

US008266296B2

(12) **United States Patent**
Dowling et al.

(10) **Patent No.: US 8,266,296 B2**
(45) **Date of Patent: *Sep. 11, 2012**

(54) **APPLICATION-LAYER EVALUATION OF COMMUNICATIONS RECEIVED BY A MOBILE DEVICE**

(75) Inventors: **Eric Morgan Dowling**, Richardson, TX (US); **Mark Nicholas Anastasi**, Highland Village, TX (US)

(73) Assignee: **East Texas Technology Partners, LP**, Plano, TX (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 174 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **12/272,481**

(22) Filed: **Nov. 17, 2008**

(65) **Prior Publication Data**

US 2009/0083376 A1 Mar. 26, 2009

Related U.S. Application Data

(60) Division of application No. 10/920,817, filed on Aug. 18, 2004, now abandoned, which is a continuation of application No. 10/335,821, filed on Jan. 2, 2003, now abandoned, which is a continuation of application No. 09/167,698, filed on Oct. 7, 1998, now Pat. No. 6,574,239.

(51) **Int. Cl.**
G06F 15/16 (2006.01)

(52) **U.S. Cl.** **709/227**; 709/203; 709/217; 709/223; 709/234

(58) **Field of Classification Search** 709/203, 709/205, 223, 217, 227, 234
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,285,061	A	8/1981	Ho
4,416,015	A	11/1983	Gitlin
4,489,416	A	12/1984	Stuart
4,578,796	A	3/1986	Charalambous et al.

(Continued)

FOREIGN PATENT DOCUMENTS

EP	0169548	1/1986
EP	0741481	11/1996
WO	9927702	6/1999

OTHER PUBLICATIONS

"Defendant Dell Inc.'s Answer to Plaintiff's Second Amended Complaint", Civil Action No. 2-03CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 15, 2004), 8 pages.

"Defendant International Business Machines Corporation's Answer to Plaintiff's Second Amended Complaint," Civil Action No. 2-03CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 15, 2004), 8 pages.

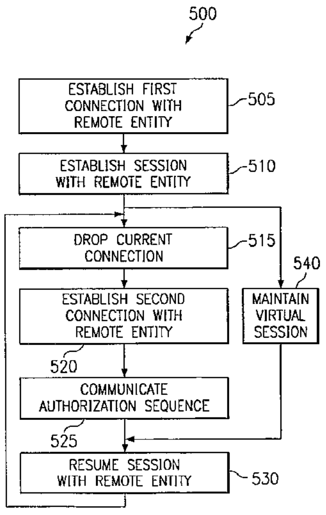
(Continued)

Primary Examiner — Ramy M Osman

(57) **ABSTRACT**

Apparatus and associated methods are provided which allow a remote user to maintain a virtual session with a server. A virtual session allows a remote and possibly mobile user to maintain a virtual presence in an office environment without actually being present. Using the present invention, a remote user can access a central application program such as an Internet service provider, a database system, an inventory system or billing system. Likewise, the remote user can receive calls and other forms of communications as though he or she were present in an office environment. A virtual session does not require a physical connection to be continuously present in order to provide a virtual connectivity. This is especially important for mobile applications where the remote user may incur long distance and/or wireless toll charges. Also, methods are presented to allow a remote unit to rapidly reconnect in a transparent and seamless way without burdening the user with the need to connect and reconnect or to upload and download information. Related methods are provided to allow the virtual session to be established, operated and maintained.

20 Claims, 4 Drawing Sheets


500
ESTABLISH FIRST CONNECTION WITH REMOTE ENTITY
505
ESTABLISH SESSION WITH REMOTE ENTITY
510
DROP CURRENT CONNECTION
515
540
MAINTAIN VIRTUAL SESSION
ESTABLISH SECOND CONNECTION WITH REMOTE ENTITY
520
COMMUNICATE AUTHORIZATION SEQUENCE
525
RESUME SESSION WITH REMOTE ENTITY
530

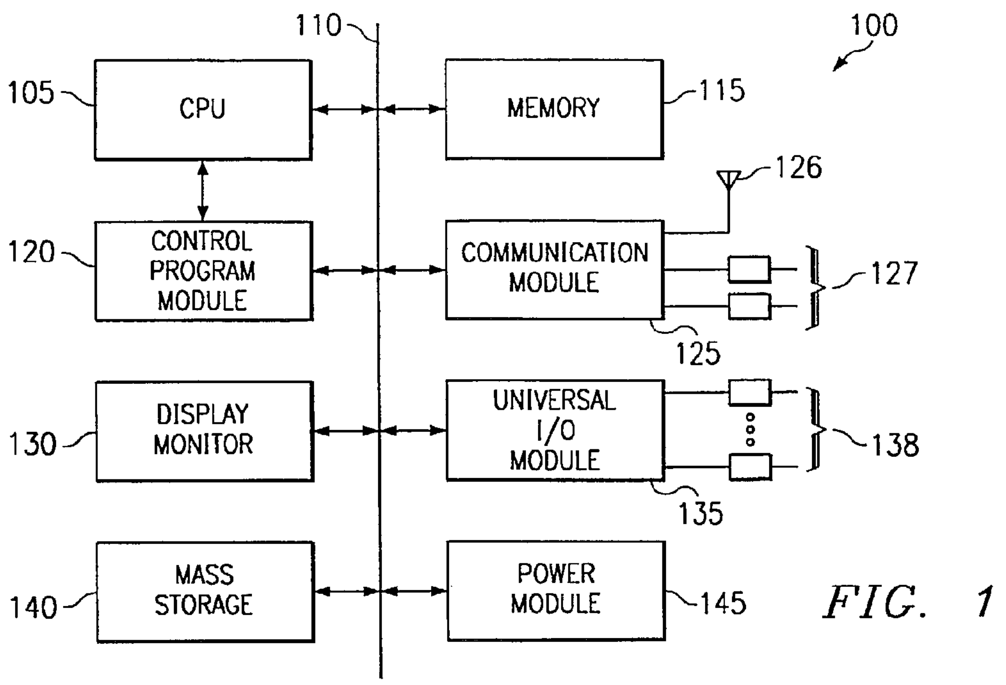
U.S. PATENT DOCUMENTS

4,852,151	A	7/1989	Dittakavi et al.	
4,995,074	A	2/1991	Goldman et al.	
5,027,269	A	6/1991	Grant et al.	
5,127,051	A	6/1992	Chan et al.	
5,287,401	A	2/1994	Lin	
5,321,722	A	6/1994	Ogawa	
5,329,619	A	7/1994	Page et al.	
5,339,392	A	8/1994	Risberg et al.	
5,367,563	A	11/1994	Sainton	
5,513,216	A	4/1996	Gadot et al.	
5,519,767	A	5/1996	O'Horo et al.	
5,539,885	A	7/1996	Ono et al.	
5,550,908	A	8/1996	Cai et al.	
5,572,528	A	11/1996	Shuen	
5,600,712	A	2/1997	Hanson et al.	
5,604,769	A	2/1997	Wang	
5,606,719	A	2/1997	Nichols et al.	
5,715,464	A	2/1998	Crump et al.	
5,724,412	A	3/1998	Srinivasan	
5,745,695	A	4/1998	Gilchrist et al.	
5,751,796	A	5/1998	Scott et al.	
5,757,890	A	5/1998	Venkatakrishnan	
5,764,639	A	6/1998	Staples et al.	
5,771,353	A *	6/1998	Eggleston et al.	709/227
5,784,562	A	7/1998	Diener	
5,802,293	A	9/1998	van der Sijpt	
5,826,085	A	10/1998	Bennett et al.	
5,842,199	A	11/1998	Miller et al.	
5,857,201	A	1/1999	Wright et al.	
5,859,971	A	1/1999	Bittinger et al.	
5,896,444	A	4/1999	Perlman et al.	
5,903,602	A	5/1999	Torkkel	
5,924,097	A *	7/1999	Hill et al.	709/208
5,928,363	A	7/1999	Ruvolo	
5,958,006	A	9/1999	Eggleston et al.	
5,978,567	A	11/1999	Rebane et al.	
5,982,774	A *	11/1999	Foladare et al.	370/401
6,023,493	A	2/2000	Olafsson	
6,052,779	A	4/2000	Jackson et al.	
6,058,422	A	5/2000	Ayanoglu et al.	
6,085,222	A	7/2000	Fujino et al.	
6,088,594	A	7/2000	Kingdon et al.	
6,088,600	A	7/2000	Rasmussen	
6,101,482	A	8/2000	DiAngelo et al.	
6,101,531	A	8/2000	Eggleston et al.	
6,119,165	A	9/2000	Li et al.	
6,119,167	A *	9/2000	Boyle et al.	709/234
6,134,582	A *	10/2000	Kennedy	709/203
6,157,630	A	12/2000	Adler et al.	
6,157,941	A	12/2000	Verkler et al.	
6,199,204	B1 *	3/2001	Donohue	717/178
6,201,962	B1	3/2001	Sturniolo et al.	
6,247,055	B1	6/2001	Cotner et al.	
6,263,016	B1	7/2001	Bellenger et al.	
6,266,701	B1	7/2001	Sridhar et al.	
6,269,402	B1	7/2001	Lin et al.	
6,289,464	B1	9/2001	Wecker et al.	
6,295,549	B1 *	9/2001	Riddle	709/227
6,308,281	B1	10/2001	Hall, Jr. et al.	
6,317,455	B1	11/2001	Williams et al.	
6,393,467	B1	5/2002	Potvin	
6,418,214	B1 *	7/2002	Smythe et al.	379/202.01
6,421,707	B1	7/2002	Miller et al.	
6,426,946	B1	7/2002	Takagi et al.	
6,453,430	B1	9/2002	Singh et al.	
6,456,603	B1	9/2002	Ismailov et al.	
6,490,610	B1 *	12/2002	Rizvi et al.	709/205
6,496,572	B1	12/2002	Liang et al.	
6,542,489	B1	4/2003	Kari et al.	
6,546,425	B1	4/2003	Hanson et al.	
6,553,490	B1 *	4/2003	Kottapurath et al.	713/1
6,560,239	B1 *	5/2003	Frid et al.	370/216
6,560,456	B1	5/2003	Lohtia et al.	
6,574,239	B1	6/2003	Dowling et al.	
6,584,321	B1	6/2003	Coan et al.	
6,594,682	B2	7/2003	Peterson et al.	
6,594,692	B1	7/2003	Reisman	
6,975,710	B2	12/2005	Fujino et al.	
7,206,816	B2 *	4/2007	Gorty et al.	709/227
7,373,144	B1	5/2008	Kirkpatrick et al.	
2002/0048354	A1 *	4/2002	Perlman et al.	379/93.25
2003/0055327	A1	3/2003	Shaw et al.	
2003/0156039	A1	8/2003	Tester	
2004/0120277	A1	6/2004	Holur et al.	
2005/0039048	A1 *	2/2005	Tosey	713/201

OTHER PUBLICATIONS

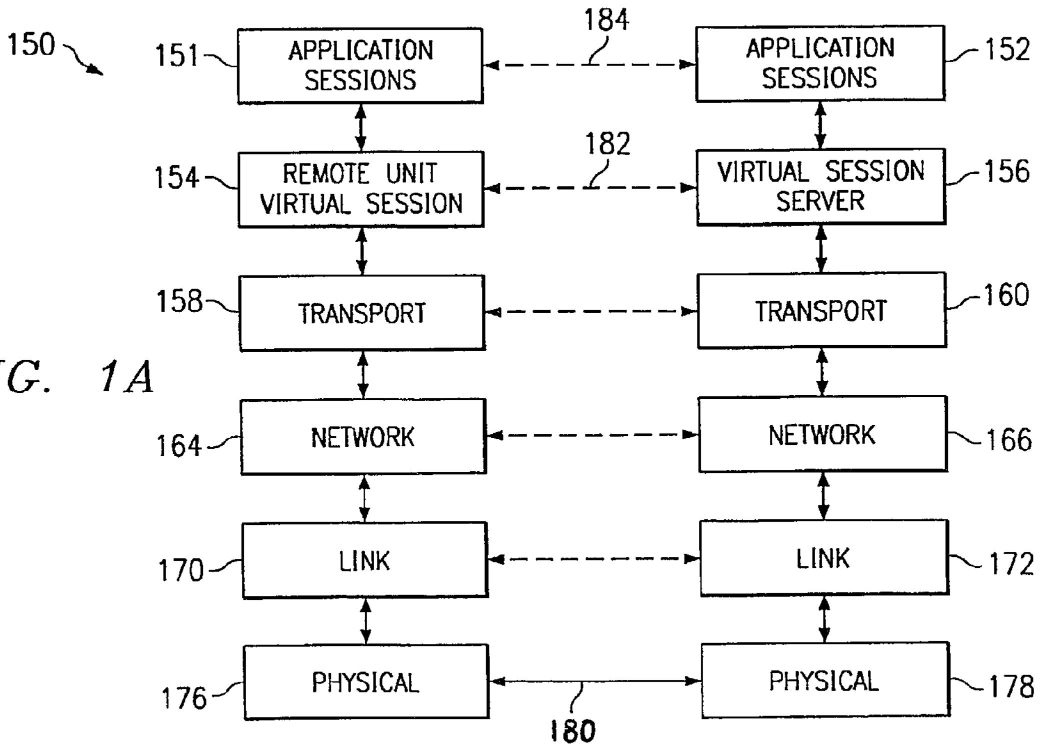
"Answer of Defendant Toshiba America, Inc. to Plaintiff's Second Amended Complaint," Civil Action No. 2-03CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 15, 2004), 5 pages.
"Gateway, Inc.'s Original Answer and Counterclaim," Civil Action No. 2-03CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 18, 2004), 8 pages.
"Hewlett-Packard Co.'s Original Answer and Counterclaim," Civil Action No. 2-03CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 18, 2004), 8 pages.
Agere Systems, Inc. "Complaint," Civil Action No. 2-04CV-108, in the United States District Court for the Eastern District of Texas, (Mar. 17, 2004), 6 pages.
"Plaintiff Agere Systems Inc.'s First Amended Complaint," Civil Action No. 2-04CV-108, in the United States District Court for the Eastern District of Texas Marshall Division, (May 11, 2004), 29 pages. (including Appendix A).
"Plaintiff East Texas Technology Partners' Answer and Counterclaim to Intervenor Conextant's Complaint," Civil Action No. 2-03-CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Apr. 21, 2004), 10 pages.
"Emachines, Inc.'s Original Answer and Counterclaim," Civil Action No. 2-03-CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Apr. 22, 2004), 8 pages.
"Answer of Defendant in Intervention Conexant Systems, Inc. to Plaintiff East Texas Technology Partners' Counterclaim," Civil Action No. 2-03-CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (May 12, 2004), 6 pages.
"Conexant Systems, Inc.'s Motion to Intervene," Civil Action No. 2-03-CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 10, 2004), 27 pages (including Exhibits A & B).
"Answer of Defendant Acer America Corp. to Plaintiff's Second Amended Complaint," Civil Action No. 2-03-CV-465-(Ward), in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 29, 2004), 8 pages.
WAP Architecture Version Apr. 30, 1998, "Wireless Application Protocol Architecture Specification", Wireless Application Protocol Forum, Ltd., (Apr. 30, 1998), pp. 1-20.
WAP WTA Draft Version Apr. 30, 1998, "Wireless Application Protocol Wireless Telephony Application Specification," Wireless Application Protocol Forum, Ltd., (Apr. 30, 1998), pp. 1-31.
"Defendant's Supplemental Preliminary Invalidity Contentions", Civil Action No. 2-03CV-465-TJW in the United States District court for the Eastern District of Texas Marshall Division, 73 pages.
Warrier, Padmanand, "Universal DSL Deployment of G.Lite", Texas Instruments Application Report SPAA007A, (Sep. 1998), 33 pages.
"239 Invalidity Chart for G.Lite Standard", no date, 26 pages.
"239 Invalidity Chart for European Patent No. 0169548", no date, 47 pages.
"239 Invalidity Chart for European Patent No. 0741481", no date, 22 pages.
Wireless Application Protocol, Wireless Session Protocol, Draft version Apr. 30, 1998, copyright Wireless Application Protocol Forum, Ltd., 1998, 95 pages.
Official Action in U.S. Appl. No. 12/194,311 issued Apr. 4, 2011, 34 pages.
Office Action in U.S. Appl. No. 12/194,311 Oct. 6, 2011, 21 pages.

* cited by examiner



100
110
105
CPU
MEMORY
115
126
120
CONTROL PROGRAM MODULE
COMMUNICATION MODULE
127
125
130
DISPLAY MONITOR
UNIVERSAL I/O MODULE
138
135
140
MASS STORAGE
POWER MODULE
145

FIG. 1



150
151
APPLICATION SESSIONS
184
APPLICATION SESSIONS
152
154
REMOTE UNIT VIRTUAL SESSION
182
VIRTUAL SESSION SERVER
156
158
TRANSPORT
TRANSPORT
160
164
NETWORK
NETWORK
166
170
LINK
LINK
172
176
PHYSICAL
PHYSICAL
178
180

FIG. 1A

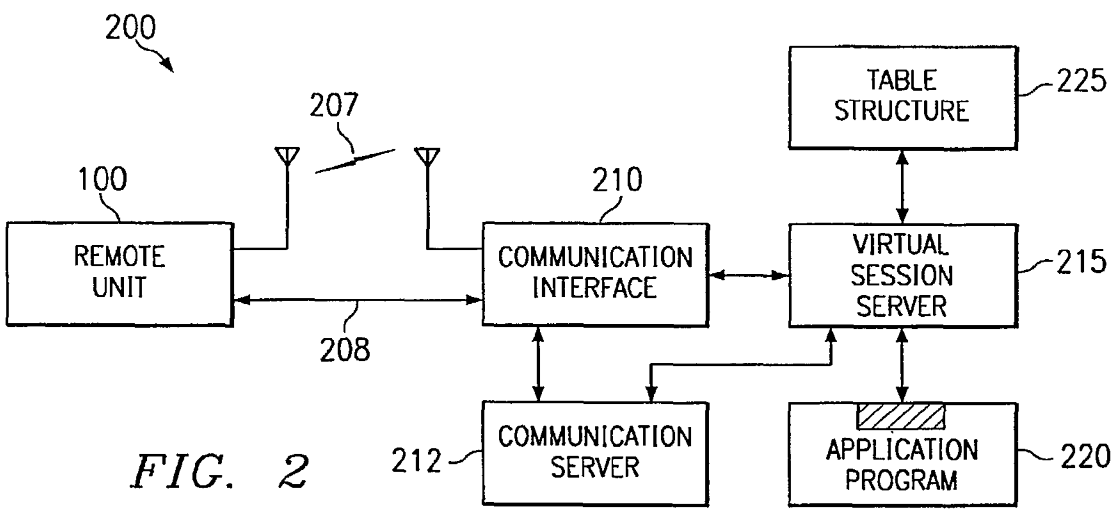

200
207
100
REMOTE UNIT
210
COMMUNICATION INTERFACE
208
TABLE STRUCTURE
225
VIRTUAL SESSION SERVER
215
212
COMMUNICATION SERVER
APPLICATION PROGRAM
220

FIG. 2

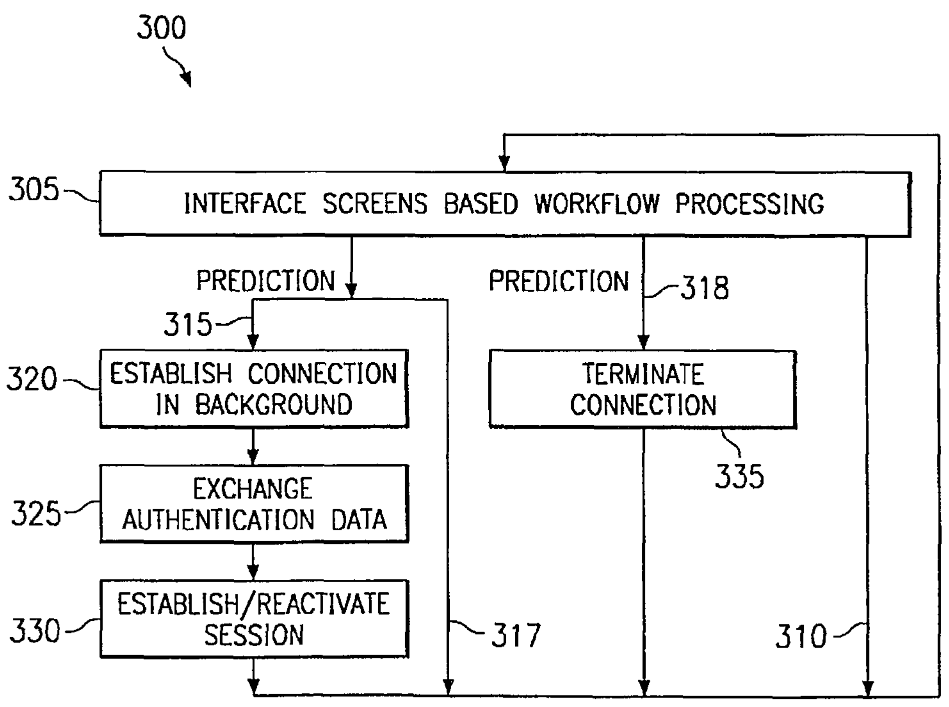

300
305
INTERFACE SCREENS BASED WORKFLOW PROCESSING
PREDICTION
PREDICTION
318
315
320
ESTABLISH CONNECTION IN BACKGROUND
TERMINATE CONNECTION
335
325
EXCHANGE AUTHENTICATION DATA
330
ESTABLISH/REACTIVATE SESSION
317
310

FIG. 3

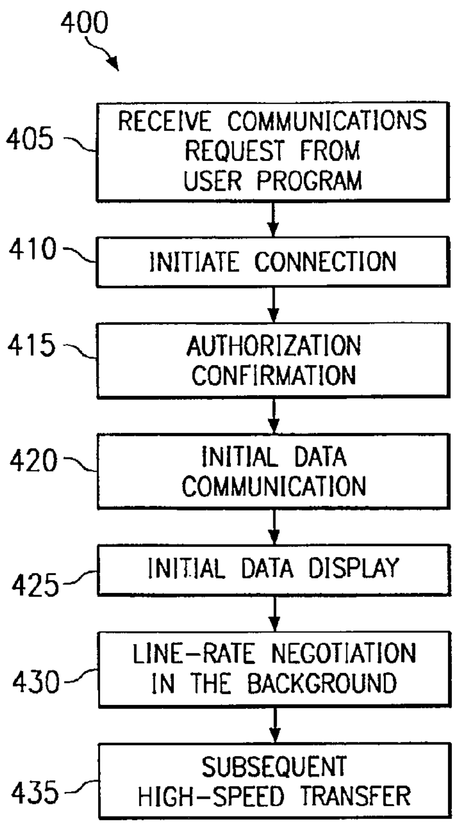

400
405
RECEIVE COMMUNICATIONS REQUEST FROM USER PROGRAM
410
INITIATE CONNECTION
415
AUTHORIZATION CONFIRMATION
420
INITIAL DATA COMMUNICATION
425
INITIAL DATA DISPLAY
430
LINE-RATE NEGOTIATION IN THE BACKGROUND
435
SUBSEQUENT HIGH-SPEED TRANSFER

FIG. 4

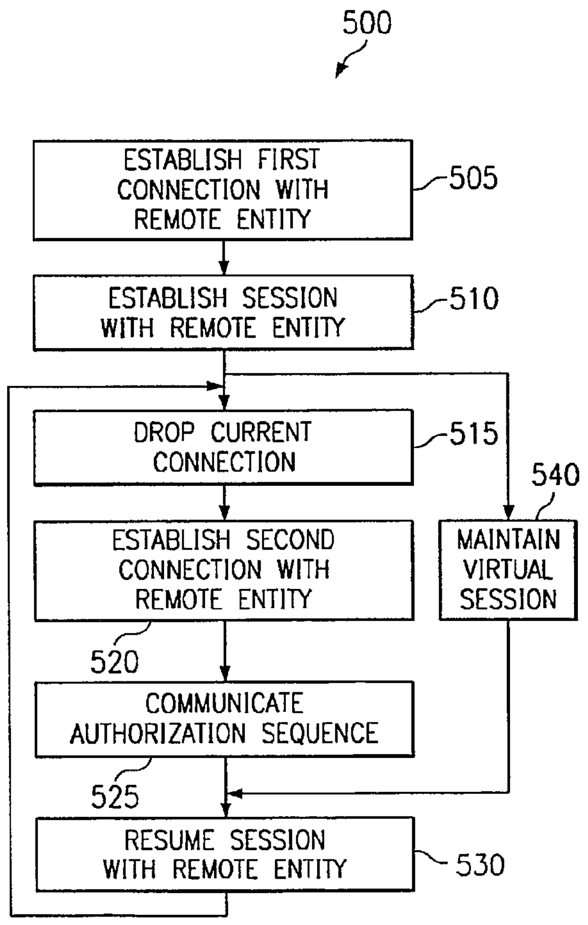

500
ESTABLISH FIRST CONNECTION WITH REMOTE ENTITY
505
ESTABLISH SESSION WITH REMOTE ENTITY
510
DROP CURRENT CONNECTION
515
540
MAINTAIN VIRTUAL SESSION
ESTABLISH SECOND CONNECTION WITH REMOTE ENTITY
520
COMMUNICATE AUTHORIZATION SEQUENCE
525
RESUME SESSION WITH REMOTE ENTITY
530

FIG. 5

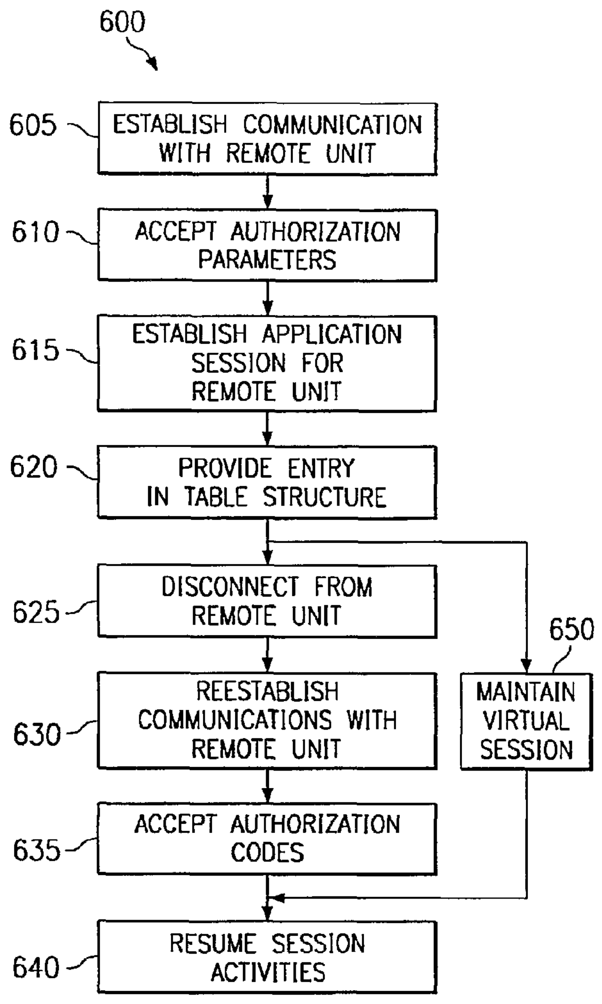

600
605
ESTABLISH COMMUNICATION WITH REMOTE UNIT
610
ACCEPT AUTHORIZATION PARAMETERS
615
ESTABLISH APPLICATION SESSION FOR REMOTE UNIT
620
PROVIDE ENTRY IN TABLE STRUCTURE
625
DISCONNECT FROM REMOTE UNIT
630
REESTABLISH COMMUNICATIONS WITH REMOTE UNIT
635
ACCEPT AUTHORIZATION CODES
640
RESUME SESSION ACTIVITIES
650
MAINTAIN VIRTUAL SESSION

FIG. 6

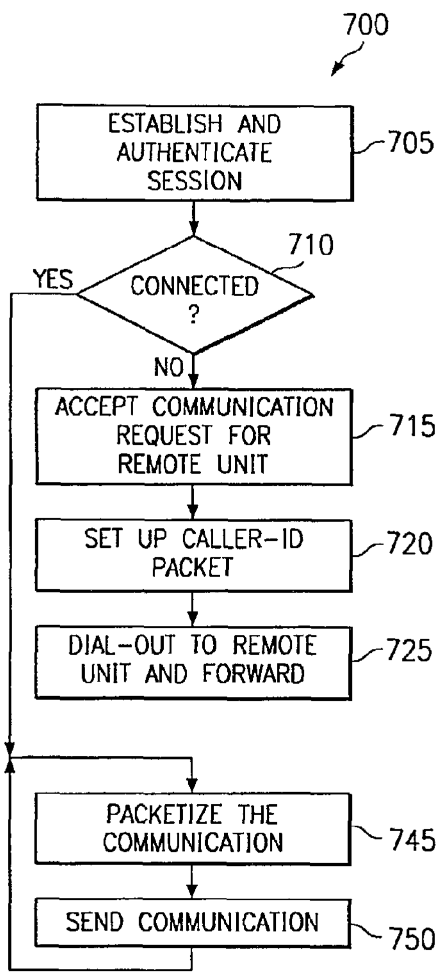

700
ESTABLISH AND AUTHENTICATE SESSION
705
710
CONNECTED ?
YES
NO
ACCEPT COMMUNICATION REQUEST FOR REMOTE UNIT
715
SET UP CALLER-ID PACKET
720
DIAL-OUT TO REMOTE UNIT AND FORWARD
725
PACKETIZE THE COMMUNICATION
745
SEND COMMUNICATION
750

FIG. 7

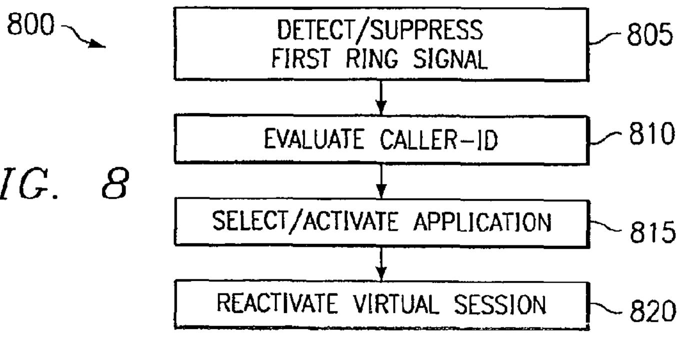

800
DETECT/SUPPRESS FIRST RING SIGNAL
805
EVALUATE CALLER-ID
810
SELECT/ACTIVATE APPLICATION
815
REACTIVATE VIRTUAL SESSION
820

FIG. 8

APPLICATION-LAYER EVALUATION OF COMMUNICATIONS RECEIVED BY A MOBILE DEVICE

CROSS-REFERENCE TO RELATED APPLICATIONS

The present application is a divisional of U.S. patent application Ser. No. 10/920,817, filed Aug. 18, 2004, which is a continuation of U.S. patent application Ser. No. 10/335,821 filed Jan. 2, 2003, now abandoned, which is a continuation of U.S. patent application Ser. No. 09/167,698, filed Oct., 7, 1998, now U.S. Pat. No. 6,574,239, entitled "Virtual Connection of a Remote Unit to a Server." The present application claims priority to these applications and incorporates them by reference herein in their entireties.

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates generally to client-server computing architectures and communication techniques. More particularly, the invention relates to a system whereby a mobile worker and a central server may maintain a virtually continuous connection without the need to maintain a physical connection continuously.

2. Description of the Related Art

The concept of a virtual connection has arisen in connection with telecommuting and related applications. Such a system is described in U.S. Pat. No. 5,764,639. A telecommuter dials into a server using a standard telephone line. The telecommuter's modem and a modem controlled by the central server establish a connection therebetween. Once a connection is established, the telecommuter may access a computer connected to the server, read emails and receive phone calls and faxes. For example, if a customer attempts to call the telecommuter at work by dialing into a private branch exchange (PBX), the server will convert the incoming call to a packetized form, such as H.323, and redirect the call via the existing connection between the telecommuter and the server. Using this system, the telecommuter may access a computer at work, answer phone calls and answer emails. The telecommuter thus appears to be present in his or her office and thus has a virtual presence there. Note for this system to properly function, the telecommuter must stay connected to the server at all times. While this does not present a significant problem for local telecommuting, this solution is quite costly for long distance telecommuting. Likewise, this solution is very costly if the telecommuter is mobile and must maintain a virtual presence with the server using a cellular wireless connection. Furthermore, in some areas it may be difficult to maintain a wireless connection continuously. A lost connection may also prevent one from regaining access to the system until some period of time has passed. Some mobile workers require only intermittent access to the server, but find it too inconvenient to place a dial-in call and to log onto the system every time access is needed.

There is a need to provide mobile workers with various forms of virtual connectivity. Mobile workers differ from telecommuters in that while a telecommuter typically works from a single home location or remote office, a mobile worker moves from location to location during the course of a normal working day. An example of a mobile worker is a home-care professional. A home-care professional is a medical worker who periodically travels to visit with different sets of home-bound patients according to their individual needs. The individual patients each have a set of medical records indicative

of their medical histories. A patient's medical record is preferably maintained as an interactive electronic document containing multiple parts. For example, the medical record indicates to the home-care professional precisely what procedures are to be performed and what medications are to be administered or otherwise given to the patient. Once the services are performed, the home-care professional must annotate the medical record accordingly. The medical record is updated to reflect the patient's vital signs and other information related to patient progress. Also, a billing system takes note to track expendables and services rendered. For example, the patient may be billed per visit and each visit may involve the expenditure of billable resources such as medicines.

In the above scenario, a mobile worker must interact with a central server during the course of a day. The worker may wish to access the central server while visiting a patient. The worker may also wish to access the server from a location where only a wireless connection can be established. From a performance perspective, an ideal solution is to provide the mobile worker with a wireless connection from a remote unit to a central server. Such a wireless connection could be established via a high-powered radio connection with a broad area of coverage or via an existing cellular or personal communication system (PCS) network. Solutions using high-powered radio links have the disadvantage that costly spectrum may be required. Maintaining a link on a cellular or PCS system is expensive in that a continuous connection consumes billable airtime which is also very costly. From an airtime-cost perspective, an ideal solution would be to force the worker to create a connection, download or up load information, and work locally with data on the remote unit as often as possible. This solution is tedious, and while saving airtime costs, may actually represent the more costly solution when professional service costs are factored in. This method has the added disadvantage that when files are uploaded or downloaded the data must be synchronized in case another user has changed the data in parallel with the mobile worker. Alternatively, other users must be "locked out" of the file from the time the mobile user downloads it until it is finally uploaded with any changes made. This is the problem solved using semaphores in shared memory systems. In the context of the present invention, a "file semaphore" is a semaphore used to lock a second user out of a file while a first user is using it. Due to the aforementioned reasons, in many applications forcing the worker to repeatedly connect, disconnect, upload and download information is unacceptable.

Some mobile networks have been constructed using what is known as cellular digital packet data (CDPD). In a CDPD network, a remote unit transmits a data packet on an unused analog channel. In this sense the mobile unit remains virtually connected to a CDPD communication server. Wireless airtime is only consumed when data is actually sent. A disadvantage to this approach is CDPD networks are not universally available. Cellular coverage is much more ubiquitous than CDPD coverage. Also, CDPD network subscribers must often pay high fees and hence CDPD may not represent the most economical solution.

In some systems such as packet switched network routers, offices make use of dial-out links. Dial-out links are useful when remote offices are separated by long distances. In such systems, when a packet must be routed from a first office to a second office, a call is placed to route the packet. The dial-out connection remains connected until a no-traffic condition is detected, indicating the line is no longer active. When the no-traffic condition is detected the connection is dropped until it is again needed. Dial-out links are thus used to reduce long distance fees associated with maintaining a constant connection, and represents a useful starting point for solving the foregoing problems relating to the establishment of a virtual presence of a mobile worker. Client-server protocols and fast automated connection strategies employing dial-out links are needed to provide new ways for a mobile worker to maintain a virtual presence. Also, new methods are needed to enable dial-out links to be set up with low delays to make them more useful for novel systems.

It would be desirable to provide a system whereby a remote worker could maintain a seamless connection with a central server without the need to maintain a dedicated channel. It would be desirable if the remote worker could communicate with the central server without the need to spend time to enter a password, reconnect, and wait for a line negotiation sequence to proceed before being able to use the connection. It would be desirable for a protocol stack to activate a virtual session based on a prediction derived from a workflow. It would be desirable to use this prediction to set up a connection in the background without disturbing the mobile worker while the mobile worker performed tasks in a workflow. It would also be desirable to have a remote unit which contains most of the screen-related information needed to provide the appearance of an established connection before the connection has been fully established. It would be desirable for the remote unit to download information before it is needed and upload information after it is gathered without the user even being aware these actions are being performed. It would further be desirable to establish a virtual session using a first communication medium such as a landline and to later communicate using the same virtual session using a second communication medium such as a wireless link. This would allow a mobile worker to select the most economical or convenient means of communications at a given time. In embodiments involving modem-based connections, it would be desirable to transmit data immediately using instantly available but lower line speeds. It would be desirable to then negotiate a higher line speed in the background while the remote worker and/or the server perform other tasks. Moreover, it would be desirable to establish a session between a remote unit and a server so that various forms of communications may proceed while providing the user with the appearance the user is continuously connected to the server and has a virtual presence with the server.

SUMMARY OF THE INVENTION

The present invention solves these and other problems by providing systems and methods to enable a remote worker to stay virtually connected to a central server without the need to continuously remain connected via a physical channel. The present invention is useful when costs are associated with maintaining a connection, for example when the connection has associated with it long distance, wireless, or other usage-related toll charges.

A first aspect of the present invention involves a communication protocol making use of a virtual session layer. The virtual session layer allows a communication session and an application session to be maintained in an inactive state when no physical connection exists. When a remote unit later reconnects with a server, the virtual session is placed into an active state and session communications resumes as though uninterrupted. A remote unit, a virtual session server, and a communication system including the remote unit and the virtual session server are presented to support virtual sessions communications. In one embodiment, the virtual session server manages a logon session between the remote unit and a server-side application program. The virtual session server emulates the presence of the remote unit to the server-side application program and thereby maintains the logon. In related embodiments, the server-side application program involves a communication server capable of relaying messages and establishing communication channels with the remote unit using the virtual session layer.

A second aspect of the present invention involves a method of accessing a central server from a remote unit. A first step involves presenting a workflow to a user via a user interface. A second step involves predicting, based upon the workflow, when the user will require connectivity to the central server. Based upon the prediction and in the background, a third step involves initiating the establishment of a physical layer communication connection to the central server.

A third aspect of the present invention involves a method of establishing a connection with a low connection set-up time. In a first step, the method initiates the establishment of a communication connection to be used to communicate with a remote entity. Next the method communicates application layer data via the communication connection prior to the completion of a line-rate negotiation process. Next the method negotiates a line speed in the background.

A fourth aspect of the present invention involves a method of setting up and operating a virtual session. This method can be practiced by a client-side remote unit or a server-side virtual session server. A first connection is established to a remote entity. This first connection is then used to establish a set of parameters needed to define a communication session with the remote entity. Next the first connection disconnected and a set of communication session parameters are maintained. Next a second connection to the remote entity is established and an authorization sequence is communicated. The communication session is next reactivated using the communication session using the second connection. A related method is used to allow a remote unit to maintain a virtual communications presence with a remote communication server coupled to a virtual session server.

BRIEF DESCRIPTION OF THE FIGURES

The various novel features of the present invention are illustrated in the figures listed below and described in the detailed description which follows.

FIG. **1** is a block diagram representing an embodiment of a remote unit designed in accordance with the present invention.

FIG. **1**A is a block diagram illustrating a layered software architecture representative of the communication protocols of the present invention.

FIG. **2** is a block diagram illustrating a system comprising a remote unit operably coupled to a server via a communication medium.

FIG. **3** is a flow chart illustrating a method of processing whereby an application program implementing a workflow provides a prediction of when the user will need a connection and establishes a connection in the background just before it is needed.

FIG. **4** is a flow chart illustrating a method of establishing communication with a remote entity with a near-immediate set up time.

FIG. **5** is a flow chart illustrating a method of communicating by maintaining a virtual presence without the need to continuously maintain a physical connection.

FIG. **6** is a flow chart illustrating a method of processing performed on a server acting as a front-end to an application program to maintain sessions for remote users who are not continuously physically connected to the application program.

FIG. **7** is a flow chart illustrating a method of processing performed on a server managing virtual connections for users who are not continuously physically connected to the server.

FIG. **8** is a flow chart illustrating a method of processing performed by a remote unit to accept different types of incoming calls.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

FIG. **1** is a block diagram representing an embodiment of a remote unit **100** designed in accordance with the present invention. The remote unit **100** may be implemented as a laptop computer, a personal digital assistant, a desktop computer or workstation, or as a dedicated unit customized for a particular application. The remote unit **100** includes a central processing unit (CPU) **105** connected to a central bus **110**. The central processing unit may be implemented using an available microprocessor, microcontroller, or customized logic. For example, a Pentium™. processor from Intel Corp. may be used to implement the CPU **105**. The central bus is preferably constructed as a set of unbroken wires used to carry signals between a set of component subsystems within the remote unit **100**. It should be noted, in some embodiments of the present invention, the bus **110** may be implemented equivalently using a set of direct parallel and/or serial connections between individual modules. The bus **110** as illustrated in FIG. **1** shows a low cost and a preferred means to connect the illustrated subsystems. Any combination of bus connections and direct serial or parallel links may be used to implement the connection structure provided by the bus **110**. Different implementations represent different price-to-performance ratios and will be dictated by the needs of an individual embodiment. The bus **110** also comprehends multi-layered bus structures. For example, some embodiments make use of a local processor bus connected to the CPU **105**, and a peripheral interconnect bus for other subsystems. In multi-layered bus based designs, the different layers are preferably connected by bus bridges. All of these and other equivalent embodiments of the bus **110** are known to the skilled artisan. From here forward, the discussion will center on the illustrated embodiment of the remote unit **100** whereby all subsystems are directly connected via the bus **110**. Embodiments where the bus **110** represents a different physical interconnection topology are implicitly included in the discussion below.

A memory **115** is also coupled to the bus **110**. The memory **115** may be implemented using static random access memory (SRAM) or dynamic random access memory (DRAM). One type of SRAM is read-only memory (ROM). Preferably the memory **115** includes a ROM for use at boot-up, and a DRAM to hold a significant amount of data storage for use while executing programs. The remote unit **100** also includes a control program module **120**. The control program module **120** is controllably coupled to the CPU **105** and is also coupled to the bus **110**. The central program module **120** typically exists as a software module executed from the memory **115** by the CPU **105**. The control program module **120** effectively configures the remote unit **100** to operate in accordance with aspects of the present invention as discussed herein below.

A communications module **125** is also coupled to the bus **110**. The communications module includes at least one communication interface to allow the remote unit to communicate

with a remote entity such as a virtual session server as will be discussed in detail hereinafter. In a preferred embodiment, the communications module **125** includes a plurality of communication interfaces. For example, a first interface **126** provides a wireless link, and a second communication interface **127** provides one or more wireline links. Also, the wireline communication interface **127** may include a standard telephone modem interface and a packet style interface designed to plug directly into an Ethernet connection to be coupled to a local area network (LAN), a wide area network (WAN) or the Internet. The Internet is the well-known and ubiquitous World Wide Web. In some embodiments, the communications module **125** includes a caller-identification packet processor. A caller-identification packet processor receives a caller-identification packet, extracts information therefrom, and passes the information to the CPU **105**. Caller-identification packets may be advantageously used to identify incoming calls with a virtual session as discussed in connection with FIGS. **7**-**8**. The communications module **125** may optionally include a voice interface to allow a user to engage in telephone conversations using the remote unit **100**. In this case a separate handset or a built-in handset may be used. Alternatively a speakerphone may be built into the remote unit using a microphone, a speaker, and an echo canceller.

The remote unit **100** also includes a display monitor **130**. The display monitor **130** is also coupled to the bus **110**. The display monitor **130** is preferably implemented using a liquid crystal display (LCD), although other display technologies may equivalently be used. Also connected to the bus **110** is an optional universal input-output (I/O) module **135**. The universal I/O module includes a coupling to a set of external devices **138**. The external devices are preferably data collection units as described below. The universal I/O module preferably provides a standard link layer interface to the software module **120** executing on the CPU **105**. The remote unit **100** also preferably includes a mass storage device **140**. The mass storage device **140** is also connected to the bus **110**. The mass storage device is preferably implemented using magnetic disk or optical disk technology, but any mass storage device, to include a non-volatile memory, may be used.

The remote unit **100** also includes a power module **145**. The power module is preferably and optionally coupled to the bus **110** to receive power management control information. The power module preferably includes a battery, an alternating current (AC) connector, a direct current (DC) connector, and a power management control interface. The AC connector allows the remote unit **100** to be powered from a standard 110 V wall outlet. The DC connector allows the remote unit **100** to be powered from a vehicle, for example by plugging the unit into a cigarette lighter outlet. Either of these connectors may be preferably used to also charge the battery in the remote unit. The power module **145** is coupled to supply power to a power bus which is connected to all subsystems. Depending on the power management configuration of an individual system, different subsystems may accept power from separate connections to allow portions of the remote unit to be selectively turned off while they are not being used.

The remote unit **100** is operative to execute an application program. The application program is operative to supply a sequence of interactive screens or a menu based interface to the user. The sequence of interactive screens or a particular usage of a menu based system implements a workflow. In an example embodiment, the remote unit **100** is carried by a home-care professional. The home care professional has a sequence of procedures which need to be implemented in the course of working with a patient. This sequence of procedures gives rise to the workflow implemented in the control program **120** which executes on the remote unit **100**. In the example embodiment involving a home-care professional, the universal I/O module is connected to a set of peripheral units to collect vital information such as blood pressure, temperature, insulin level and the like. Other information such as the patient's weight may be entered manually by the home-care professional as a part of the workflow. At certain times in the workflow, an external communication connection will be needed because data may need to be uploaded or downloaded to/from a central server. In accordance with the present invention, the remote unit **100** is operative to provide a seamless and transparent virtual presence with the central server. In general, the central server may itself be segmented into two or more individual central servers. The discussion herein focuses on an embodiment whereby a virtual presence is maintained with a single central server having multiple server components. The present invention may be equivalently practiced by embodiments involving a virtual presence with more than one central server. Thus, one remote unit could maintain multiple virtual connections to totally separate server systems. In such a configuration the application workflow would dictate to which server system the remote unit would physically connect while other servers remain virtually connected.

A key aspect of the operation of the remote unit **100** is its ability to maintain a virtual presence with the central server without continuously maintaining a physical connection. The remote unit **100** is operative to provide communications when it is needed without the user needing to go through a set of normally associated connection sequences. For example, in accordance with one aspect of the invention, the user need only interact with the screens provided to implement the workflow while the remote unit **100** automatically sets up a connection in the background to be available when it is needed. In embodiments where file synchronization is not an issue or is handled using file semaphores, the software implementing the workflow automatically downloads information before it is needed and later automatically uploads new information after it has been gathered. This way, users need not even be aware they are not connected at all times. The user is not burdened with the need to connect and reconnect, and need not be burdened with downloading and uploading data. The user experiences the full benefit of being continuously connected to the central server without the associated cost of remaining continuously connected via a physical connection. In systems where file semaphores are not employed, the physical connection is established just before the workflow indicates it will be needed and is dropped when the workflow indicates it will not be needed for some time. Further details of the operation of the remote unit **100** are given in the discussions provided in connection with FIGS. **2-8**.

A central aspect of the present invention involves the concept of a "virtual session." A session as defined herein is similar to the definition provided in the open systems interconnect (OSI) reference model from the International Standards Organization (ISO). The OSI model is a model of a layered software structure used in computer communications. A software system which implements a layered model of communication is known as a "protocol stack." The OSI model is well known and divides a computer communications process into seven layers. At each layer is a software module which communicates with a peer software module at the same layer. Within a protocol stack, each layer communicates with the layer above and/or below. Actual communication systems often deviate from the seven layer OSI model. A protocol

stack using basic concepts similar to the seven layer OSI model is next discussed which represents an aspect of the present invention.

FIG. **1**A illustrates a representative protocol **150** used to support the present invention. At the top layer is an application session layer. A first protocol stack with an application session layer software module **151** communicates with a second protocol stack with an applications session layer software module **152**. The application session layer software module **151** is typically implemented as a client-side software module which presents a user interface to a user. The application session layer software module **152** is typically implemented by a server-side software module operative to provide communication and/or computer related services to the client-side software module. For example the application sessions layer software module **152** may involve a logon session with a database program, or may represent a unified messaging server supplying voice mail, email and fax mail. Similarly, the application session layer software module **152** may involve a telephony application operative to provide a packet switched or a circuit switched telephone connection to the client-side application session layer software module **151**. One layer down in the protocol stack is a virtual session layer. In the example embodiment, the first protocol stack implements a virtual session layer software module **154** in the remote unit **100**. The virtual session layer software module **154** communicates with a peer virtual session software module **156** via the peer-to-peer communication path **182**. In the exemplary embodiment, the virtual session layer software module **156** is implemented within a virtual session server as discussed in connection with FIG. **2**. The virtual session server typically maintains a table linking one or more application sessions to a virtual session. For example, this linking of application sessions into the table structure may be accomplished by including a pointer to a data structure containing application session control data, or by placing the data structure holding the application session control data directly in the table structure. Additionally, the table structure allows the virtual session server to maintain a plurality of virtual sessions with a plurality of client remote units. In the OSI model, the OSI-session layer provides a set of rules used to establish and terminate data streams between nodes in a network. A set of OSI-session layer services include establishing and terminating node connections, message flow control, dialog control, and end-to-end data control. The session layer controls dialogs, which involve conversational protocols as used in mainframe computer terminal communications. The virtual session layer **154**, **152** of the protocol **150** may perform any of these functions in addition to maintaining the table linking to the application sessions.

The next software modules in the first protocol stack are the transport layer **158**, the network layer **164**, the link layer **170** and the physical layer **176**. These software modules respectively perform peer communications with the server-side protocol stack's software modules **160**, **166**, **172**, and **178**. The physical layer defines the low-level mechanical and electrical channel protocols and the physical connection itself. These four lower layers are well known in the art of data communications and can be implemented in various well-known ways. Likewise, alternative and equivalent protocol stacks may be constructed, for example, with the transport layer removed, various layers merged into one, or new layers added.

An important aspect of a virtual session oriented communication protocol such as the protocol **150** is the ability to maintain a peer-to-peer virtual session communication path **182** without the presence of a physical layer communication path **180**. The physical layer communication path **180** represents a physical layer communication connection, for example, a wireline connection, a cellular wireless connection, or a network connection to the Internet. When the physical layer communication path **180** is disconnected, no physical channel exists between the client-side software and the server-side software, and the physical layer communication path **180** is said to be in a "disconnected state." However, data structures maintained at the virtual session layer allow one or more peer-to-peer application session communication paths **184** to remain in a deactivated but existent state, even when the physical layer communication path is in the disconnected state. Likewise, the virtual session communication path **182** established between the remote unit and the virtual session server also remains in a deactivated but existent state. This is made possible through the use of the table structure maintained in memory which retains its information after the physical layer communication path **180** has been disconnected. When the physical layer communication path **180** has been reconnected, the physical layer communication path is said to be placed into a "connected state." At such time, the virtual session layer software modules **154** and **156** are operative to reactivate the virtual session layer communication path **182** and the application sessions layer communication path **184**. When these paths are reactivated, peer-to-peer communication may once again proceed over the application session layer communication path **184** and the virtual session layer communication path **182**.

As defined herein, a distinction is made between a communication session and an application session. A communication session is defined as a session between nodes or communication endpoints, and an application session is defined as a session between applications. For example, a remote unit may establish a communication session with a central server. In this case a communication session is established between the communication endpoints, i.e., the remote unit and the central server. Also, an application program running on the remote unit may need to establish an application session with an application program running on the central server. In such case an application session is created using a connection stream provided and governed by the communication session. A table structure is used to maintain both the communication session parameters and the application session parameters. For example, a first user authentication parameter may be used to establish a communication session with the server. A second user authentication parameter may be used to establish an application session with the application program. This second user authentication parameter may include a user identification parameter and a password, for example.

In light of the aforesaid concepts, a "virtual session" is next defined. A virtual session is preferably implemented as a communication session as defined above. A virtual session, like an OSI session, provides a set of rules for establishing data streams between nodes or endpoints. The virtual session also may provide other session features such as dialog control, message flow control, and end-to-end data control. A virtual session is controlled using a data structure which provides a way to associate the virtual session with the lower layers of a protocol stack, leading down to a physical layer. As mentioned above, in most embodiments, a virtual session is implemented as a communication session. Application sessions are then added onto the virtual session as connection streams within the communication session.

In a virtual session, a communication session may be suspended with some or all of the lower layers of the protocol stack missing. In particular, a virtual session may be maintained while a physical layer connection has been removed. The virtual session can then be reassociated with a physical

layer connection at a later time. The virtual session thus also preferably provides connect and reconnect rules used to establish a virtual session and then to reassociate the virtual session to a new physical connection to set up a new data stream in support of a dialog at a later time. Related activities such as the initiation of dial-out links to reestablish a physical layer communication path is also preferably handled by the virtual session in response to a signal from an application layer program.

An aspect of a virtual session is the maintenance of an application between an application program and a virtual session server as will be described below. A virtual session server acts as a proxy agent for a remote unit. When the remote unit is not connected via a physical layer communication path, the virtual session server maintains a proxy-presence with the application program on behalf of the disconnected remote unit. At a later time, when the remote unit reconnects into the virtual session by passing a set of communication session authentication parameters, the remote unit is thereby granted access to one or more application sessions which have been maintained in proxy by the virtual session server.

In a preferred embodiment, the virtual session uses a set of authentication parameters and a set of encryption keys to maintain a secure connection. A separate set of authentication parameters is used by an application running on the remote unit to gain access to an individual application session. Once the application session has been established over a virtual session, a table is used to maintain a set of parameters needed to maintain the application session, even though no physical layer connection exists between the endpoints of the virtual session. When a virtual session data structure is set up and no physical layer connection exists to support communication over the virtual session, the virtual session is said to be "inactive." When a virtual session data structure is set up and a physical layer connection does exist to support communication over the virtual session, the virtual session is said to be "active." A transition from an active state to an inactive state is called "deactivating a virtual session," and a transition from an inactive state to an active state is called "activating a virtual session." The process of transitioning from an active state to an inactive state is also known as "disconnecting from a physical connection." When this occurs, the physical layer connection is no longer available to support communication over the virtual session. In a preferred embodiment, a table structure is used to maintain the virtual session parameters as well as a set of parameters for each application session established over the virtual session. When a virtual session is activated, there is no need to reauthenticate the individual application sessions. This is because the table typically includes a user identification parameter, a user password, a set of application session parameters, a communication session identification parameter, and an encryption key for the communication session. Additional data such as modem initiation parameters may be added to the table as required by the system configuration and usage.

Referring now to FIG. **2**, a block diagram illustrating a system configuration **200** is shown. The system configuration **200** includes the remote unit **100** operatively coupled to a communication interface **210**. A direct wireless link **207** optionally couples the remote unit **100** to the communication interface **210**. A direct wireless link is used in embodiments where the remote unit **100** maintains a direct wireless link with the communication interface **210**. The communication interface thus provides an air interface for the direct wireless link. Alternatively or in addition to the direct wireless link **207**, a wireline link **208** couples the remote unit **100** to the communication interface **210**. The communication interface **210** maintains the connection **208** via a network interface coupled to a public switched telephone network (PSTN) or a network such as the Internet. This connection **208** may itself involve a microwave link, a wireless link through a public switched cellular network or a wireless link in a PCS network.

The communication interface **210** is preferably coupled to a communication server **212**. The communication server **212** may be thought of as generalization of a private branch exchange (PBX). The communication server **212** accepts tele-traffic from any variety of sources and provides switchable connections to couple different sources together. For example, the communication server **212** may be implemented as a PBX which receives a set of direct inward dial lines from a central office operated by the public telephone network. The PBX then provides local users with extensions and allows local users to call each other by dialing the last four digits of their telephone numbers. The PBX typically provides an outside line to a user once the user has dialed a nine.

The communication server **212** may also be configured to provide additional types of connections, such as packet based voice and video connections according to the H.323 international standard. In such an embodiment, the communication server **212** provides a gateway function passing calls between the public switched telephone network and a network such as the Internet. The communication server **212** may also provide other communications services such as voice mail, email, fax-mail, call distribution and the like. In systems involving Internet telephony, the communication server may operate only using packet protocols and not include an interface for circuit switched connections.

The communication interface **210** is also coupled to a virtual session server **215**. The virtual session server **215** is coupled to a table structure **225** and an application program **220**. The table structure **225** is preferably implemented as a software entity and may be located in a memory module within the server **215**. The virtual session server **215** may be implemented as a software entity which executes on a hardware platform. The hardware platform of the virtual session server **215** may be designed with an internal architecture similar to the remote unit **100** but is designed to provide a higher computation capacity and to handle multiple users. When supporting a virtual session server, the display monitor **130** is optional as users may control the virtual session server **215** remotely. The control program module **120**, when implemented in the virtual session server **215** provides the server side of the communication protocols discussed in connection with FIGS. **3**-**8**. Hence the remote unit **100** and the virtual session server **215** involve similar architectures and respectively implement the client and server sides of a set of virtual-session-related communication protocols of the present invention.

The application program **220** may execute on the same hardware platform as the virtual session server **215**. In general, both the virtual session server **215** and the application program **220** may be implemented as software modules running on personal computers, workstations, dedicated custom hardware, mainframe, or file servers. For example, the virtual session server **215** may be implemented as a software module running on an UltaSparc™. workstation or file server from Sun Microsystems Inc. The software may be written to execute over a multitasking operating system such as Solaris™. from Sun Microsystems Inc. or WindowsNT™. from Micrsoft Inc. In a first preferred embodiment, the application program **220** includes a distributed database program running on a collection of networked servers such as Sun UltraSparc™. servers. In a second preferred embodiment, the

application program may itself be a communication server as provided by an Internet service provider (ISP).

The system **200** is operative to implement a set of virtual session communication protocols according to the present invention. The remote unit **100** establishes a session via the virtual session server **215** to set up a virtual presence with the application program **220**. Preferably, the virtual session server **215** also provides a link to the communication server **212** to provide it access to the virtual session. When the remote unit **100** disconnects from a physical connection **207** or **208**, the virtual session is maintained within the table structure **225**. When the remote unit **100** later wishes to reestablish communication with the application program **220**, the virtual connection server **215** is operative to keep the virtual session active and to allow the user rapid and nearly transparent access to the application program **220**. Similarly, the virtual session also preferably is used to provide a virtual communication link between the communication sever **212** and the remote unit **100**. In some systems, a first virtual session is established between the remote unit **100** and the application program **220**, and a second virtual session is established between the remote unit **100** and the communication server **212**. The details of the operation of the virtual session server **215** and the virtual session protocols are discussed below in connection FIGS. **3**-**8**. Before proceeding to these portions of the detailed description, two embodiments of the system **200** are described.

In a first exemplary embodiment of the system **200**, a mobile worker such as a home-care professional operates the remote unit **100** to establish and maintain a virtual session with the application program **220**. In one embodiment, the application program **220** controls access to a database including complete medical and billing records for individual patients. Depending on working conditions, the home-care professional may require access from a wireless connection such as a cellular connection, or else may be able to communicate via a wireline connection provided within a patient's home. As the home care professional proceeds through a given workflow, the professional will eventually need to communicate with the application program **220**. When this time arrives, the present invention is operative to establish a physical connection between the home-care professional and the application program **220**. The professional need not be aware the physical connection has not been available since the time the virtual session was first established. The virtual session is maintained by the virtual session server **215** and the protocols of the present invention are employed to ensure such a virtual connectivity is provided without the need for the remote unit **100** to be continuously connected to the application program **220**.

In a second exemplary embodiment, the application program **220** is a communication server operated by an ISP. In this example, the remote unit **100** is operated by an Internet user. After the Internet user has remained inactive for a period of time, the connection **208** is terminated. At a later time, when the Internet user clicks on a hyperlink, thus demanding service, a short delay is incurred while the connection is reestablished. The remote unit is provided access without the user needing to reestablish a connection. When the user clicks on a hyperlink, the telephone is rapidly dialed without presenting dialing tones to the user. An authentication packet and a request packet are sent using a low data rate protocol such as one used for line-rate negotiation in modems. The user is authenticated by the server and the request packet is forwarded through the Internet to the Internet site referenced by the hyperlink. While the remote Internet server takes time to respond to the request, a higher line speed is negotiated in the background without burdening the user. Because a home Internet user uses the same analog connection between the user's premises and a network interface, the modem parameters may be preferably saved by the server in the table **225** to accelerate re-negotiation. The user is provided access almost immediately, and the connection is reestablished transparently. Note while this example focuses on an Internet application, the techniques apply to any application whereby a network site is accessed by activating a hyperlink.

As will be discussed below, the virtual session between the remote unit **100** and the virtual session server **215** provides a means to initiate transfers in both an uplink and a downlink direction. The uplink direction is from the remote unit **100** to the virtual session server **215**, and the downlink direction is from the virtual session server **215** to the remote unit **100**. A virtual session is said to exist between the remote unit and the virtual session server **215**. This virtual session may be used to create individual virtual sessions between the remote unit **100** and the application program **220**, and between the remote unit **100** and the communication server **212**. For example, an uplink connection is established, and when a home Internet user has been inactive for a period, the connection is dropped. As discussed above, the connection is reactivated transparently when the user once again activates an Internet link, as in an Internet browser. In the same example, a user may have an email reader program connected through a virtual session. If an email comes in for the user and the virtual session is in place, the email should be rapidly forwarded to the user. To do this, the user's phone is dialed in a downlink direction dial-out link by the virtual session server **215** via the communication interface **210**. The remote unit preferably suppresses the first ring and examines caller identification data. When the caller identification data indicates the calling party is the virtual session server **215**, the remote unit **100** automatically picks up the call and in this example, accepts the email. If caller identification is not used, a substitute protocol should be employed to assure that connection has been made to the proper application session defined within the virtual session. The substitute protocol preferably involves sending a packet header at the beginning of a call whereby the packet header contains one or more fields which identify associated the application session. Again, the user need not even realize a connection has been reestablished. Instead, the user receives the email message as though the connection had remained continuously active.

Another type of operation may occur when the user of the remote unit **100** is actively connected to the virtual session server **215** and a call comes in directed to the remote user's extension. At this point the call is preferably converted into packets and is sent to the user over the existing connection. In an alternative embodiment, the physical connection is automatically and temporarily dropped and the call is forwarded to the remote user. The virtual connection to the application is maintained through the virtual server. The communications module **125** preferably analyzes caller-identification data to determine the incoming call is a voice call to cause the optional telephone aspect of the remote unit **100** to ring. More details related to the foregoing system operation are discussed in connection with FIGS. **3-8** below.

The virtual session server **215** is able to maintain an open logon to the application program **220**. In one embodiment, the virtual session server **215** executes a client-side software which interfaces with the application program **220**. That is, if the application program **220** employs a client-server architecture, the application program **220** will implement a server-side software module which interacts with the client-side software. The server-side program performs database or other

server oriented functions, while the client-side software provides a user interface to the user. The remote unit **100** can then control the operations of the virtual session server **215** using standard remote session software. An example of commercially available remote session software is PCAnywhere™. from Semantec Corporation. In another embodiment, the virtual session server executes the client-side software in parallel with the remote unit. In still another embodiment, the remote unit executes the client-side software, and the virtual session server merely provides a connection stream to pass data from the application program **220** to the remote unit **100**. When the virtual session is in a deactivated state, the virtual session server emulates the client-side software as needed to maintain an active session with the application program **220** in the absence of the remote unit **100**. A wide variety of equivalent techniques may be used to allow the virtual session server **215** to maintain a pointer or re-entry point into the application **220** while acting as a proxy agent to maintain the logon for the remote unit **100**. A table structure is preferably used to allow the virtual session server to simultaneously maintain a plurality of logons for a plurality of different remote units.

In some embodiments, the remote unit **100** may need to maintain a plurality of virtual sessions with a plurality of different virtual session servers. For example, an independent contractor may provide home-care services for two distinct health regions. Each health region may use a separate database. The remote unit **100** may then access these separate databases using a first and a second client-side application software module. During the course of a day, the remote unit may need to activate the first or the second client-side application software modules. In such case the remote unit **100** is operative to maintain a table structure similar to the table structure **225**. The table structure maintained by the remote unit links an application software module through an application session to a virtual session. When the first client-side application program demands access to a first database, the virtual session layer software **154** in the remote unit causes a physical connection to be established to support virtual session communications **182** with the first database application program. Likewise, if the second client-side application software module desires to access a second database, the virtual session layer software module **154** activates a physical layer connection back to the second database server. In other applications a single application program may be used which accesses information on more than one virtual session server. In such case a single application program can select the virtual session to activate based on the communications request generated from within the application program. In still other embodiments, a single physical connection **208** or **207** may be used to communicate with the communication interface **210**. The communication server **212** then forwards packets to a first local virtual session server such as the virtual session server **215**. If the received communication packets are destined for a second virtual session server, then the communication server **212** preferably forwards the packets to a remote virtual session server using a network connection such as an Internet connection.

Referring now to FIG. **3**, a method **300** is illustrated to show how the remote unit **100** preferably operates to activate a connection. The method **300** is preferably practiced by the remote unit **100** in support of a virtual session with the virtual session server **215**. A first step **305** of the method **300** involves actions within a workflow process **305**. The workflow process **305** includes the step **305** of the method **300** and also performs other activities to interact with a user's workflow requirements. Control loops from the first step **305** back to the first step **305** via a control path **310**. The workflow, as discussed above, is preferably made up of a menu system and/or a set of interactive screens traversed by a worker in performing a set of tasks. For example, a home-care professional's workflow involves accessing and displaying a patient's medical record, entering a set of data into the medical record, and performing tasks indicated by the doctor's directions as annotated in the medical record. In this example, as the home-care professional moves from one screen to the next, control loops via the control path **310**. The workflow process **305** is an application program which executes on the CPU **105**. The workflow process **305** is preferably implemented as a process running on the CPU **105** in a multitasking operating environment. A multitasking operating environment is one in which multiple programs or processes may execute in parallel by sharing time slots within the CPU **105**. Multitasking operating system software is well known and is readily available. In a multitasking-programming environment a first process may execute in a normal fashion and provide an interface to a user. At the same time a second process may be executed by sharing CPU cycles without the user's intervention or knowledge. In such a case the second process is said to be a background process or is said to perform background processing. At some point in the course of the workflow, a physical layer communication connection will be needed to communicate information between the remote **100** and the application program **220**.

When a step in the workflow process **305** is performed leading up to the need for a physical layer communication connection, control next passes from the first step **305** to a second step **320** via the control path **315**. The control path **315** is activated when the workflow process **305** provides a prediction indicating a physical layer communication connection will subsequently be needed. In some cases the prediction may be provided right when the physical layer communication connection is needed. In other cases, the prediction **315** may be used to initiate background processing to download data which will not be needed until a later time. In menu based systems, the prediction **315** may be learned by observing the workflow habits of a user. The prediction **315** is a function of the application program or workflow **305** and is optional. In the second step **320**, a connection is established in the background. Background processing enables the user to continue interacting with the workflow process **305** while a physical layer communication connection is simultaneously and transparently established. That is, the physical layer communication path is reestablished without inhibiting the user from interacting with the workflow **305**. Hence when control passes via the control path **315** to the step **320**, control preferably simultaneously passes via the control path **317** back into the workflow. The background process is preferably forked as a separate task and two execution flows proceed in parallel by time sharing the CPU **105**. Multitasking is well known in the art and is implemented using interrupt based processing. In alternative embodiments the control path **317** may be deleted and a single control flow may be implemented using the control path **315**. However, this embodiment may require the user to wait for the connection to be established and is hence not deemed to be the preferred embodiment of the method. Other equivalent embodiments set up the communication path transparently by multiplexing the CPU **105**'s computation cycles from within the workflow process or some other process.

Once control has been forked via the control path **315** to the second step **320**, a dialer within the communications module **125** preferably dials to establish a physical layer communication connection with the communication interface **210**. In embodiments using dedicated radio links, the connection

may be established over the wireless link **207**. One preferred embodiment of a remote unit **100** incorporates a cellular radio. In this case the dialer dials a telephone number and a connection is established using a public switched cellular telephone network so that the connection is set up on the link **208**. Stationary Internet based embodiments perform the second step **320** by dialing a telephone number using an automatic dialer which dials a land line connection for a modem. In all cases, it is preferred to suppress the dialing tones and line-rate negotiation signals so the connection may be established transparently to the user.

Control next passes from the second step **320** to a third step **325**. In the step **325**, an authentication code is transmitted from the remote unit **100** to the communication interface **210**. This authentication code is then passed to the virtual session server **215**. The virtual session server evaluates the authentication code to determine if access is to be permitted. In a preferred embodiment, the authentication code involves a digital signature as is known in the field of public key cryptography. In a preferred embodiment, all transmissions are encrypted using public key cryptography. Some systems may be implemented using various encryption standards such as secure sockets layer based encryption. The amount of authentication and encryption used in any given embodiment is left to the system designer, but preferably all transactions are encrypted as described above.

Control next passes from the third step **325** to a fourth step **330**. In the fourth step **330**, a session is established/reactivated with the virtual session server **215**. The session is established the first time the method **300** passes control to the step **330**. Subsequently the step **330** is operative to reactivate the session with the virtual session server. When the session between the remote unit **100** and the virtual session server **220** is reactivated, virtual session communications resume. At this point, the virtual session server **215** correlates information stored in the table structure **225** with the connection and provides the remote unit **100** access to the application program **220**. If no data is stored in table structure **225**, access is provided to a default logon screen allowing remote unit **100** to establish a new application session. The virtual session server **215** then populates the table structure **225** to establish a virtual session. The step **330** involves setting up a stream connection between the workflow process **305** and a protocol stack. The protocol stack is operative to read information bits from the stream connection and communicate the bits across an external communication link. Bits received over the external communication link are converted by the protocol stack into information bits to be sent back to the workflow process **305** across the connection stream. Once the appropriate communication processes are configured, control next passes back to the workflow process **305**. Due to the aforementioned forking operation, the passing of control back to the workflow process **305** may have already occurred via the control path **317**. In this case the passing of control from the step **330** to the workflow process is not explicitly performed.

When control loops back from the fourth step **330** to the workflow process **305**, a physical layer communication connection is activated for current or subsequent communication. When the user gets to a point in the workflow where communication with the application program **220** is needed, the connection has already been transparently set up in the background. Hence the user gets the feel of being connected to the application program **220** all the time, where in fact the remote **100** is connected via a physical channel to the application program **220** only a fraction of the time. This virtual connection saves communication resources and money when a toll is charged based on the amount of usage on the link **207** or the link **208**. In some embodiments, the fourth step **330**, or an execution thread within the workflow **305** is operative to upload or download information in the background. This way the user has ready access to data contained in the application program **220**, but in general a shorter connect time is required. While with prior art systems it is burdensome for a user to connect to a central server and download and upload information, with the virtual session of the present invention the user need not even be aware this process is occurring. Rather the user feels as though he or she is continuously connected with a fast connection because the data needed at a given point in the workflow is already available locally or has been uploaded in the background transparently without user intervention. In systems where server synchronization is an issue, file semaphores and/or direct active sessions not employing uploading and/or downloading of records may be used.

Based on another point in the workflow, another prediction is made to predict when the communication channel will not be needed for some time. For example, it may be known, based on the workflow, the home-care professional will next perform a sequence of tests and enter data into a screen displayed on the remote unit. Only at a later time will the workflow come to a point where this information is to be uploaded to the application program **220**. When such a prediction is made, control passes from the first step **305** via the control path **318** to a fifth step **335**. The fifth step **335** is operative to deactivate the connection established over the link **207** or the link **208**. The step **335** may optionally involve forking a separate execution thread or otherwise accessing a separate process in a multitasking environment. Alternatively, the fifth step **335** may be performed by executing a set of instructions in the workflow process **305**. At a later time, a prediction may be made indicating the link **207** or **208** needs to once again be activated, whereby control again passes over to the second step **320** via the control path **315**. It should be noted different systems will typically set their prediction times according to the economic conditions involved. For example, in some systems the first minute of connection time may cost five times as much as all subsequent minutes. In this case predictions would be preferably set according to a criterion to minimize cost by not establishing and terminating connections more often than necessary. If a flat rate were charged per minute connections would be set-up and terminated more often. If automatic uploading and downloading is performed in the background, a very efficient use of air-time can often be achieved while presenting the user with the appearance of being continuously connected to the application program **220**.

Referring now to FIG. **4**, a method **400** of establishing a communication link with low delay is illustrated. The method **400** may be practiced by both the remote unit **100** and the communication interface **210**. This method is most applicable to systems involving modems whereby digital data is transferred over an analog channel requiring receiver training. Receiver training involves transmitting data sequences through a channel and allowing a receiver to adjust its receiver parameters. Receiver parameters include echo canceller and equalizer filter coefficients. Most systems also adjust their data rates and signal constellations based on observed conditions. In modems, this entire process is known as line-rate negotiation. Prior art systems involving receiver training are tedious to use because they force the user to wait while the receiver is trained. Most systems play the training signals though a speaker to allow the user to hear the training process. This lets the user know what the computer is doing for the duration of the delay. The method **400** improves upon this

prior art solution by allowing the user to gain almost immediate access without a significant delay.

In a first step **405**, a protocol stack or other process practicing the method **400** receives a communications request from a user program. For example, this occurs when a user clicks on an icon to initiate the establishment of an Internet connection. Control next passes to a step **410** where the connection is initiated. This step typically involves an automatic dialer dialing the number of an Internet service provider (ISP). The ISP software may be implemented as a communication server application corresponding to the application program **220**. In this case access to the application program is governed by the virtual session server **215**.

Control next passes from the second step **410** to a third step **415**. In the third step **415** an authorization sequence is exchanged. In a preferred embodiment public key cryptography involving digital signatures and keys is used. Embodiments involving a virtual session server **215** either set up a session or activate an existing session during the third step **415**. Control next passes from the third step **415** to a fourth step **420**. In the fourth step **420**, one or more initial application layer data packets are transmitted across the connection using a low speed protocol. A low speed protocol is used by the transmitter and receiver when performing line-rate negotiations. For more details of line-rate negotiation protocols, see, for example, the V.34 and V.90 standards from the International Telecommunications Union. In the present invention, the low speed protocol is used to transmit application layer data before the line-rate negotiation procedures have completed. This avoids the need for the user to wait for line-rate negotiation to complete before being able to access a communication path.

Control next passes from the fourth step **420** to a fifth step **425**. In the fifth step **425**, initial data is displayed. Software located locally in the remote unit **100** preferably contains high-volume graphics related data so that the initial data exchange of the step **420** only requires a small amount of data to be transferred. For example, the user logs onto the Internet and almost immediately sees a screen of information indicating the user is connected and the system is ready to accept inputs. This is made possible by displaying locally held screens of graphical data and allowing a small amount of specific information such as time, date, and headlines to be received and displayed. If the user then immediately clicks on a link, an application layer request packet is sent using the line-rate negotiation protocol's data format. This allows the user to immediately begin making requests before the line-rate negotiations have completed. In many cases the user will pause and read the headline information, giving the system even more time to perform line-rate negotiation in the background.

Control next passes from the fifth step **425** to a sixth step **430**. In the sixth step **430**, a background process is forked to perform line-rate negotiation. Line-rate negotiation is allowed to proceed in the background while the user is reading the information provided on the initial display of the step **425**. Likewise, if the user had rapidly clicked on a link, a request packet is sent out and while the server is responding to the request, the background line-rate negotiation may proceed. The step **430** is operative to perform line-rate negotiation so as to set up a high-speed connection for subsequent higher volume data transfers. In embodiments involving a virtual session server **215**, the user's line speed parameters may be stored in the table structure **225**. For example, if the user is an Internet user and the application program **220** is an ISP, the user will often dial in from the same location. Thus parameters derived in a previous activation of a communication channel will be either identical or similar to those used in a current activation. Hence the sixth step **430** optionally involves accessing from the table **225** a set of starting parameters derived from the activation of the communication channel. If communication is needed before the line-rate negotiation has completed, communication preferably proceeds at the highest rate negotiated up to that point.

Once the line-rate negotiation process of the step **430** has completed, control passes to a seventh step **435**. In the step **435**, communication is able to proceed at full speed. In most cases where this method is implemented, the user will get the full benefit of being connected almost immediately without the normal delay associated with prior art systems. This is so because initial low-volume data is allowed to pass through the channel before the line-rate negotiation has completed. Line rate negotiation then proceeds in the background in parallel with other activities such as the user reading headline information or a distant server accessing data and responding to the initial data request packet sent across the Internet. This technique is useful when a user is maintaining a virtual session with a remote server because it is imperative to allow the user to appear to be connected without having to experience delays when accessing data. The method **300** and the method **400** may be performed together in a complementary fashion to make the virtual session appear to be constantly available.

The method **400** may be practiced by the remote unit **100** and the virtual session server **215**. When the remote unit **100** initiates the method, the virtual session server **215** executes steps **410**, **415**, **420**, **430** and **435**. When the virtual session server **215**, the application program **220**, or the communication server **212** initiates the method, one or a combination of these servers practice the steps **405**, **410**, **415**, **420**, **430**, and **435**. The first step **405** involves, for example, receiving a communication request such as a telephony call or an email for the remote unit **100**. In some systems, the first step **405** may involve a request generated from within the application program **220**.

Referring now to FIG. **5**, a method **500** of establishing and operating a virtual session is illustrated. For example, the method **500** establishes a virtual session between the remote unit **100** and the virtual session server **215**. The method **500** is practiced by both the remote unit **100** and the virtual session server **215**. In a first step **505** a first physical layer communication connection is established with a remote entity. If the method is practiced by the remote unit **100**, then the remote entity typically corresponds to the virtual session server **215**. The virtual session may be used to support virtual sub-sessions between the remote unit **100** and the application program **220**. Also, a virtual sub-session may be established between the remote unit **100** and the communication server **212**. For the purposes of discussion herein, all of these virtual sessions will be referred to simply as virtual sessions. If the method **500** is practiced by the virtual session server **215**, then the remote entity typically corresponds to the remote unit. The step **505** may be activated according to the prediction **315**, and the step **505** may use the method **400** to allow the connection to be set up with very low delay.

Control next passes from the first step **505** to a second step **510**. In the second step **510** a session is established with the remote entity. In a preferred embodiment, this involves exchanging password information and agreeing upon a set of keys to encrypt data transacted in the session. Also, the virtual session server **215** preferably sets up a table entry in the table structure **225**. The table entry indicates the presence of a virtual session. The table entry may include modem parameters as discussed in connection with FIG. **4**. Also, the virtual session as set up in the table entry links the remote unit to a

user identification and a password as presented to the application program **220**. For example, a user name and a password may be used as user authentication parameters. Preferably public key encryption is used to encrypt all information so the password sent from the remote unit **100** to the application program **220** cannot be effectively intercepted. The remote unit **100** also preferably sets up a virtual session data structure to hold similar information related to the virtual session. Once the virtual session has been set up, the remote unit **100** may access the application program **220**. Also, the remote unit **100** may optionally access the communication server **212** for communication services.

Control next passes from the second step **510** to a third step **515**. In the third step **515**, the physical connection established in the first step **505** is dropped. Meanwhile the virtual session data structures and table entries established in the second step **510** are retained. The session is allowed to proceed while no physical layer connection exists. That is, the step **510** is operative to set up a table structure including one or more data structures which allows a virtual session to be maintained in memory while other activities occur. Hence a passive background thread of execution passes from the step **510** to a passive step **540** whereby the virtual session is maintained. This allows the remote unit **100** to stand by or be used for steps of the workflow process **305** not requiring communication with the application program **220**. Once the user needs to communicate with the application program **220**, or when a prediction **315** is made, control next passes from the third step **515** to a fourth step **520**. The step **520** is operative to reestablish a second physical layer communication connection to allow communication to proceed once again using the session established in the second step **510**. This connection reestablishment may be performed in response to the prediction **315** and may use the low-delay connection establishment technique of the method **400**.

In some embodiments, the present invention involves using distinct and separate communications media to perform the step **505** and the step **520**. For example, a mobile worker may call in from home to set up the virtual session in the step **510** using the first physical layer communication connection established in the step **505**. Later in the day, the worker may call in from a restaurant while catching up on some records keeping. This second use of the virtual session involves use of the second physical layer communication connection which in this example is a wireless connection different from the landline connection used to initiate the session from home earlier in the day. At a still later time, the worker may call in from a patient's home via a third physical layer communication connection while performing home-care services. If modem starting parameters have been stored in table **225**, they are preferably updated whenever the communication connection is changed. Hence the virtual session of the present invention enables a mobile worker to continue communications via the most expedient and/or economical means without causing the user to have to reestablish a communication connection. Preferably, when the remote unit **100** is connected to a communications source via the connector **127**, the remote unit **100** automatically detects this connectivity and makes use of it for subsequent virtual-session communications. That is, the present invention contemplates the availability of various forms of "pigtail" connectors being available so the remote unit **100** can operate in a "plug-and-play" fashion. Pigtails may be supplied to allow the remote unit to connect to the PSTN, the Internet, or to another computer via a universal serial bus, for example.

Control next passes from the fourth step **520** to a fifth step **525**. In the fifth step **525** an authorization sequence is exchanged. This is preferably implemented using public key encryption and digital signatures. Some embodiments may be developed which do not implement the fifth step **525**, but preferred embodiments do make use of user authentication. After the fifth step **525** has completed, the session is resumed in a sixth step **530**. Over the course of the virtual session, control may loop back to the third step **515** as many times as the virtual session is activated with a new physical connection. When the sixth step **530** is entered, the virtual session is once again activated so that the passive step **540** also passes control to the sixth step **530**. In a minimal implementation of the method, no looping occurs and the method terminates after the first pass through the sixth step **530**.

Referring now to FIG. **6**, a method **600** practiced by the virtual session server **210** is illustrated. In a first step **605**, a first physical layer communication connection is established for communicating with the remote unit **100**. Control next passes to a second step **610** whereby a set of authorization parameters are accepted and authenticated. As discussed in connection with FIG. **5**, the authentication parameters preferably include the exchange of public keys which include a digital signature in accordance with public key cryptography. Control next passes to a third step **615** where a user identification and a password are passed by the virtual session server **215** to the application program **220** on behalf of the remote unit **100**. As discussed in connection with FIG. **5**, the user identification and the password to be presented to the application program **220** are preferably transmitted in encrypted form. Once the application program **220** authenticates the user identification and password needed to gain access, the virtual session server **215** enters an entry into the table structure **225** to hold a set of session parameters. The session parameters include the user identification, a session identifier, encryption data and possibly other data such as modem starting parameters. Once the session has been logged into the table, the user may use it to communicate with the application program **220**.

Control next passes from the fourth step **620** to a fifth step **625**. In the fifth step **625** the physical layer connection is dropped. This step is performed when the remote unit does not currently require communications with the application program **220**. In step **650** the virtual server maintains the application session while the physical connection is disconnected. At a later time, when the user needs access to the application program or when the prediction **315** is made, control next passes to a sixth step **630**. In the sixth step **630** a second physical layer connection is established to allow communication between the remote unit **100** and the application **220** to resume. As discussed in connection with FIG. **5**, the second physical layer connection may involve a different communication path and/or medium as was used for the first physical layer connection. That is, a plurality of communications media are preferably supported to allow the user to call in via different means, for example via a wireless or a wireline connection. The step **630** may be initiated due to actions at the remote unit **100** or in response to events occurring in the server. For example, the communication server **212** may receive a call for the remote unit. Alternatively an email may be received which needs to be forwarded to the remote unit. In such a case, the sixth step **630** optionally involves sending a caller identification packet to let the remote unit know what type of communication, such as a voice telephony call, an email, or a fax, is inbound. A caller identification packet is a sequence of information bits sent across a communication connection identifying the calling party of the connection. In standard telephone systems, the caller identification packet is transmitted between the first and second rings when the telephone call is being set-up. More details relating to communications initiated by the virtual session server **215** back to the remote unit **100** are discussed in connection with FIG. **7**.

Once the second physical layer communication connection is established in the sixth step **630**, possibly according to the method **400**, control next passes to a seventh step **635**. In the seventh step **635**, authorization codes are verified similarly to the second step **610**. Once the user codes have been verified to be correct, control next passes to an eighth step **640** whereby communication once again resumes using the previously established virtual session.

Referring now to FIG. **7**, a method **700** of processing communication requests in a virtual session is illustrated. This method is preferably practiced by the virtual session server **215** simultaneously with the method **600**. In a first step **705** a virtual session is established between the virtual session server **215** and the remote unit **100** as discussed in connection with FIGS. **5** and **6**. At some later time, while the virtual session is active, the communication server **212** receives an incoming communication request for the remote unit **100**. Because the virtual session server **215** practices the method **500** and/or the method **600**, depending on the time of arrival of the communication, the remote unit **100** may or may not be physically connected to the virtual session server **215** by a physical communication link. Hence when the communication is received, control passes from the step **705** based on a decision **710** which determines whether a physical connection currently exists to the remote unit **100**.

If the virtual session is presently in a state whereby the physical connection has been disconnected, control passes from the first step **705** to a second step **715**. In the second step **715** the communication request is accepted by the communication server **212** through a direct connection or via the communication interface **210**. Control next passes to a third and optional step **720** whereby a specific caller identification packet is associated with the communication type. For example, if the communication involves a telephone call a first caller identification packet is sent identifying an extension used for telephone calls. If the communication involves an email, a second caller identification packet is sent identifying an extension used for email. On the other hand, if the communication comes from the application program **220**, still another caller identification packet is sent. When this optional use of a caller identification packet is employed, the remote unit **100** has the information needed to properly and immediately respond to an incoming call as discussed in connection with FIG. **8**. If a call is received by the remote unit from a source other than the virtual connection server **215**, the caller identification information will identify the call as not being associated with the virtual session. Control next passes to a fourth step **725** whereby an automatic dialer responds to communication requests and the communication is forwarded to the remote unit **100**.

Another situation arises when the communication request arrives while the remote unit **100** and the virtual session server **215** are currently connected by an existing physical channel. According to one mode of processing, control stays in the first step **705** while communication proceeds in an active phase of a virtual session. When the communication request arrives, the communication server **212** signals to the virtual session server that a new call has arrived for the remote unit **100**. The virtual session server then causes the existing physical layer communication connection to be dropped and thus control passes from the first step **705** to the second step **715** as in the foregoing discussion. In another mode of processing, the existing physical layer communication connection is left in tact and control passes from the first step **705** to a fifth step **745**. In the fifth step **745**, the communication is packetized, and in a sixth step **750** the communication packets are passed along the existing physical layer communication connection. Control continues to loop back from the sixth step **750** to the fifth step **745** during the course of the communication. In this mode of operation the existing physical layer communication connection is shared to provide the remote user with a means to stay connected to the application program **220** and communicate at the same time. In this case the physical layer is time shared by the virtual session to allow multiple modes of communications to proceed in parallel.

Note the method **700** provides the user of the remote unit **100** with a virtual presence in the work place while actually being remote. Independent of whether the remote unit is presently actively connected to the virtual session server, the communication request may be forwarded to the remote unit **100** making the remote user appear to be present in the office at all times. The only time the remote unit **100** would not be reachable is when it is engaged in a communication with an entity other than the virtual connection server. This problem may be mitigated by allowing the remote unit to only be reachable through the access number provided by the communication server **212**. If a call is placed from the remote unit to another point, this call too may be routed through the communication server **212**. Systems which do allow the remote unit to make calls outside the virtual session may preferably employ voice mail at the communication server **212**. When the remote unit again becomes available, the virtual session server **215** may forward the communication to the remote unit according to the method **700**. Remote units may also be designed using call-waiting concepts whereby the virtual session may be re-activated by interrupting another call.

Referring now to FIG. **8**, an optional method **800** practiced by the remote unit **100** is illustrated. This method is practiced when a virtual session exists, the remote unit and the virtual session server are presently not connected via a physical channel, and a communication request is to be forwarded to the remote unit **100**. In a first step a first ring signal is detected. Optionally, the first ring signal is suppressed so the user will not hear it. In some systems a vibrational first ring signal may be allowed to pass through to notify the user of an incoming communication. In still other embodiments, the remote unit may be programmed to sound a normal ring on the first ring signal. In some embodiments the ring signal will not be a traditional telephone ring signal but will in general be any signal indicative of an incoming communication request.

In current systems, a caller identification packet is presented to the called party after the first ring signal. Hence identification of the calling party becomes available at this time. After the first ring signal, control passes from the first step **805** to a second step **810**. In the second step **810**, the caller identification information is evaluated. Note it would be preferable to accept the caller identification data before the first ring, and the present invention contemplates systems whereby the virtual session server **215** signals to the PSTN to provide a caller identification packet before the first ring. This service does not appear to be available from telephone service providers at this time. Embodiments also comprehended by the present invention include systems whereby the remote unit **100** immediately picks up an incoming call and caller identification information is sent by the virtual connection server **215** over the connection identifying the call-type. During the second step **810**, the caller identification packet is evaluated. As discussed in connection with the third step **720** of the method **700**, the virtual session server **215** sends out a caller identification packet to identify the type of incoming

call. For example, different caller identification packets indicate whether the incoming call is an email, a voice telephone call, or a communication from the application program **220**.

Control next passes from the second step **810** to a third step **815**. In the third step **815**, an application layer program is selected to process the incoming call. In the foregoing examples, an application may be launched to accept an email message, a voice telephone call, or to accept a communication from the application program **220**. If the communication is an email, it may be desirable to pop a mailbox icon on the screen or to produce a speech signal stating "you've got mail." If the incoming call is a voice call, it may be desirable to allow the telephone to ring like a normal telephone. If the communication is from the application program **220**, it may be desirable to update data located in the screens of the workflow or otherwise signal the presence of new data. Once the appropriate application has been launched to handle the incoming communication, control next passes to a fourth step **820** whereby communication session is reactivated and the communication is processed. For example, one or more packets of information may be received and related information such as an email message may be displayed. Also, a telephone call may be allowed to proceed or a set of information may be downloaded from the application program **220**.

Although the present invention has been described with reference to specific embodiments, other embodiments may occur to those skilled in the art without deviating from the intended scope. For example, other forms of communications such as fax messages may be accepted and displayed by the remote unit **100**. Also, the present invention may be used for applications other than mobile workers and Internet users. Virtual sessions according to the present invention are applicable to any situation where a continual connectivity is required but the cost to remain continuously connected is high. Vehicle computers with cellular radio based Internet connections are an example. In such systems, the remote unit **100** may be a vehicle-mounted computer or may include a connection to a vehicle-mounted computer. Also, virtual sessions according to the present invention are applicable to any situation where the user should not be burdened with the need to upload and/or download data and go through connection and disconnection procedures. Therefore, it is to be understood that the invention herein encompasses all such embodiments which do not depart from the spirit and scope of the invention as defined in the appended claims.

In an embodiment, a remote unit is provided, the remote unit comprising a central processing unit operatively coupled to a bus; a memory operatively coupled to the bus; a display monitor operatively coupled to the bus; a communications module operatively coupled to the bus; a control program module controllably coupled to the central processing unit and operative to cause the central processing unit to display a user interface to a user via the display monitor, the control program further being operative to cause the processor to establish a first virtual session with a first remote entity using a first physical communication connection is provided.

In another embodiment, the first remote entity is a communication interface coupled to a virtual session server, the virtual session server providing access to an application program via the virtual session. The remote unit may further comprise a table structure, the table structure comprising a plurality of data structures, the plurality of data structures comprising a first data structure and a second data structure, the first data structure comprising information relating to the first virtual session, and the second data structure comprising information relating to a second virtual session with a second remote entity; wherein the control program module is operative to selectively reactivate one of the first virtual session and the second virtual session in response to a communication request from the user. Alternatively, the remote unit may further comprise a table structure, the table structure comprising a plurality of data structures, the plurality of data structures comprising a first data structure and a second data structure, the first data structure comprising information relating to the first virtual session, and the second data structure comprising information relating to a second virtual session with a second remote entity; wherein the control program module is operative to selectively reactivate one of the first virtual session and the second virtual session in response to a communication request received via the communications module.

In another embodiment, the first remote entity is a communication interface coupled to a virtual session server, and the remote unit maintains a virtual presence with the virtual session server by communicating using any of a plurality of communication types via the virtual session. The plurality of communication types may include voice telephony and email.

In yet another embodiment, the control program is further operative to disconnect from the first physical communication connection, present a workflow to the user and predict, based upon the workflow, when external communication will be required, the control program further being operative to cause a second physical communication connection to be established based upon the prediction, the second physical communication connection providing a physical layer connection for the virtual session. The control program may be further operative to predict, based upon the workflow, when information is to be uploaded and/or downloaded to and/or from the first remote entity, and to cause the information to be uploaded and/or downloaded in the background using the second physical communication connection. The second physical communication connection may be established transparently without inhibiting the user interface from supporting the workflow. The user interface may implement a workflow by presenting a sequence of interactive screens on the display monitor.

In yet another embodiment, the control program is further operative to drop the first physical connection, cause a second physical communication path to be established, and communicate application layer data via the virtual session using the second physical connection before a line-rate negotiation associated with the second physical connection has completed.

In another embodiment of the present invention, a virtual session server is provided, the virtual session server comprising a first coupling to a communication interface the first coupling providing access to a physical layer connection to a remote unit; a table structure, the table structure operatively coupled to receive information from the first coupling, the table structure storing parameters relating a communication session; a second coupling to an application program, the second coupling comprising a communication path whose access requires a user authentication parameter; and a control program module controllably coupled to the first coupling, the table structure, and the second coupling, the control program module operative to establish an application session for use between the remote unit and the application program, store a parameter relating to the application session in the table structure, and maintain the application session in both the presence and absence of the physical layer communication connection, such that the application session is continuously activated both when the physical layer communication path is in a connected state and a disconnected state.

In another embodiment, the control program module is further operative to accept from the remote unit a user authentication parameter when the physical layer connection transitions from the disconnected state to the connected state, and provide a communication path for whereby the remote unit is able to access the application program using the application session.

In another embodiment, the virtual session server further comprises a third coupling to a communication server, whereby the control program module is further operative to accept a communication request from the communication server to be forwarded to the remote unit, and if a physical layer communication connection is present, the control program further operative to activate a second application session, the second application session between the remote unit and a communication program responsible for the communication request. The control program may issue a signal indicating to convert a media stream related to the communication request into a packet stream to be routed via the second application session. Upon reception of the communication request, if no physical communication connection is presently active, the control program may be further operative to process the communication request by dialing out to establish an active physical connection and to then activate a second application session, the second application session between the remote unit and a communication program responsible for the communication request.

In another embodiment, activation of the physical layer communication connection, application layer data is transmitted via the physical layer communication connection prior to the completion of a line-rate negotiation associated with the physical layer communication connection.

In another embodiment, different connections of the physical layer communication connection use distinct communications media.

In another embodiment of the present invention, a client-server system is provided, the client-server system comprising a remote unit comprising a control program module, the remote unit operative to maintain a virtual session; a virtual session server, the virtual session server operatively coupled to a communication interface via a first coupling, the communication interface coupled to the remote unit by at least one communication link, the communication interface providing a data stream including information received from the communication link, the virtual session server operatively coupled to an application program via a second coupling; a table structure, the table structure including a link to a data structure, the data structure including a plurality of bits arranged to represent information relating to an communication session between the remote unit and the virtual session server; and a control program module controllably coupled to the virtual session server, the table structure and the application program, the control program module operative to cause the virtual session server to establish an application session for use between the virtual session server and the application program, store an application session parameter in the data structure and to link the data structure into the table structure, and maintain the application session in both the presence and absence of the physical layer communication connection, such that the application session is continuously activated when the physical layer communication path is in a connected state and a disconnected state.

In another embodiment, the control program module is further operative to cause the virtual session server to execute a client-side application software module on the virtual server under remote control of the remote unit.

In another embodiment, the control program module is further operative to cause the virtual session server to execute a client-side application software module on the virtual session server in parallel with the remote unit; and maintain a presence with the application program while the physical layer communication path is in the disconnected state.

In another embodiment, the control program module is further operative to cause the virtual session server to maintain a communication path and grant the remote unit access to the application program via the communication path after the physical layer has been placed into the connected state.

In another embodiment, the control program module is further operative to cause the virtual session server to grant the remote unit access to the application program via the communication path only after a virtual session authentication procedure has been successfully completed.

In another embodiment of the present invention, a method of accessing a central server from a remote unit is provided, the method comprising the steps of presenting a workflow to a user via a user interface; predicting, based upon the workflow, when the user will require connectivity to the central server; and based upon the prediction and in the background, initiating the establishment of a physical layer communication connection to the central server.

In another embodiment, the physical layer connection is established to support a virtual session and the remote entity comprises a virtual session server.

In another embodiment, the method further comprises the steps of making a second prediction, based upon the workflow, the second prediction indicating when information is to be uploaded and/or downloaded to and/or from the central server; and based upon the second prediction, causing the information to be uploaded and/or downloaded in the background using the physical layer communication connection.

In another embodiment of the present invention, a method of establishing a connection with a low connection set-up time is provided, the method comprising the steps of initiating the establishment of a communication connection to be used to communicate with a remote entity; communicating application layer data via the communication connection prior to the completion of a line-rate negotiation; and in the background, negotiating a line speed for subsequent high-speed communication.

In another embodiment, the method further comprises the steps of receiving an input, the input indicative of a request to communicate with a remote entity; and transacting an authorization sequence with the remote entity.

In another embodiment, the input may be received by a user interface and the input may be indicative of a command to, access a network site using a hyperlink. Furthermore, the method may comprise the step of resuming the use of a virtual session via the communication connection.

In another embodiment, the step of initiating may be performed in response to a ring signal received from the remote entity.

In another embodiment of the present invention, a method is provided, the method comprising the steps of establishing a first connection to a remote entity; using the first connection to establish a set of parameters needed to define a communication session with the remote entity; disconnecting the first connection and maintaining the parameters related to the communication session; establishing a second connection to the remote entity; communicating an authorization sequence with the remote entity; and reactivating the communication session using as a communication path the second connection.

In another embodiment, the step of using further comprises the steps of accepting a user authentication parameter and passing the parameter to an application program to establish an application session; and linking into a table structure a data structure associating the remote unit with the application session, the data structure thereby providing an access by the remote unit to the application program upon reactivation of the communication session.

In another embodiment, the first and second connections involve distinct communications media. The set of parameters may be stored in a table, and the table may hold a plurality of such sets of parameters and maintains a plurality of communication sessions with a plurality of remote entities. The method may further comprise the steps of converting a communications media stream to a packet stream and multiplexing the packet stream via the application session over the communication session. Alternatively, the method may further comprise the steps of disconnecting from the second connection while maintaining the communication session; initiating the establishment of a third connection; and processing the communication request by forwarding a communications media stream via the communication session using the third communication connection.

In another embodiment, the method further comprises the steps of evaluating a header received over the second physical layer communication connection to determine an application program to process information transacted on the second connection; and based upon the evaluation, launching the application program to communicate via the second connection.

In another embodiment, a method of providing a virtual presence of a remote unit with a central server is provided, the method comprising the steps of establishing a communication session with the remote unit using a first physical layer communication path and dropping the first physical layer communication connection; configuring a data structure, the data structure linking the communication session to an application session; accepting a communication request from the application program, the communication request requiring a physical layer communication path to the remote unit; dialing-out to establish a second physical layer communication path to reactivate the communication session with the remote unit; delivering a packet via the physical layer communication path, the packet containing information indicative of the type of the communication request; and reactivating the communication session with the remote unit and using the communication session to support a connection stream between the application program and the remote unit.

In another embodiment, the data structure links a plurality of application sessions to the communication session, the application sessions including a first an application program which provides a communication service.

In another embodiment, the communication service comprises a voice telephony connection.

In another embodiment, the plurality of application sessions includes a second an application program which provides access to a database.

In another embodiment, the database comprises a medical record.

In another embodiment, a system for maintaining a virtual session is provided, the system comprising a virtual session server, the virtual session server being operatively coupled to an application program, the virtual session server maintaining an application session with the application program on behalf of a remote unit, the virtual session server maintaining a virtual communication session with the remote unit; and a table structure comprising a plurality of memory locations arranged within a storage unit, the table structure operative to maintain set of links correlating the virtual communication session and the application session.

In another embodiment, the system further comprises the remote unit, whereby the remote unit comprises a protocol stack used to maintain a communication session in the presence and absence of a physical layer communication path.

In another embodiment, the system further comprises the application program.

In another embodiment, the virtual session server is operative to execute a client-side application software module under remote control of the remote unit.

In another embodiment, the virtual session server is operative to execute a client-side application software module on the virtual session server in parallel with the remote unit; and maintain a presence with the application program while the physical layer communication path is in the disconnected state.

In another embodiment, the control program module is further operative to cause the virtual session server to maintain a communication path and grant the remote unit access to the application program via the communication path after the physical layer has been placed into the connected state.

In another embodiment, the virtual session server maintains a proxy-presence with the application program to prevent the application program from terminating in the absence of a physical layer communication path to the remote unit.

In another embodiment, the virtual session server accesses the table structure to provide access to the application program by the remote unit via the application session upon the reactivation of the virtual communication session.

What is claimed is:

1. A method, comprising:

receiving, at a control program executing on a mobile handset, a first communication initiated by a remote entity, wherein the first communication includes a set of information identifying an application layer program that is installed on the mobile handset, and wherein initiation of the first communication by the remote entity was not in response to a request sent by the mobile handset;

the control program causing the mobile handset to evaluate the set of information included in the first communication; and

in response to determining, based on the evaluating, that the set of information identifies the application layer program, the control program causing the mobile handset to:

launch the application layer program; and

reactivate, from an inactive state, a communication session between the mobile handset and the remote entity.

2. The method of claim **1**,

wherein causing the mobile handset to reactivate the communication session includes the mobile handset establishing a physical layer connection with the remote entity.

3. The method of claim **2**, wherein the communication session supports communication between the application layer program at the mobile handset and a remote program executing on a server coupled to the remote entity.

4. The method of claim **1**, wherein the remote entity is a server.

5. The method of claim **1**, wherein the first communication was initiated by the remote entity in response to information received by the remote entity from a server coupled to the remote entity via the Internet.

6. The method of claim **1**, wherein launching the application layer program includes reactivating an inactive communication session.

7. The method of claim **1**, wherein the first communication includes at least one data packet, and wherein the first communication is included in an incoming communication request that is received by a wireless interface of the mobile handset from a cellular or PCS network.

8. The method of claim **1**, wherein the remote entity is a computer system that is remote from the mobile handset executing an application program.

9. The method of claim **1**, wherein the application layer program is a messaging program.

10. The method of claim **9**, wherein the application layer program is an email program.

11. The method of claim **10**, wherein the information directed to the application layer program includes at least a portion of an email message.

12. The method of claim **1**, wherein the application layer program is a telephony program.

13. The method of claim **12**, wherein the information directed to the application layer program includes voice data corresponding to a telephone call.

14. The method of claim **1**, wherein the first communication includes a caller identification packet.

15. A device, comprising:

a wireless interface; and

a processor unit;

wherein the device is configured to:

receive an unsolicited communication from a remote computer system, wherein the unsolicited communication is received using the wireless interface; and

in response to determining that the unsolicited communication identifies an application that is in an inactive state:

launch the application; and

reactivate a communication session between the device and the remote computer system from an inactive state.

16. The device of claim **15**, wherein reactivating the communication session includes the device establishing a physical layer connection with the remote computer system.

17. The device of claim **15**, wherein the application is an email program that is in an inactive state on the device, and wherein the device is further configured to cause the email program to enter an active state on the device in response to the determining that the unsolicited communication identifies the email program.

18. The device of claim **15**, wherein the application is a messaging program that is in an inactive state on the device and is connected through a virtual session to a messaging program executing on the remote computer system.

19. An article of manufacture, including a non-transitory computer-readable medium having instructions stored thereon that, in response to execution by a first computing device, cause the first computing device to perform operations comprising:

receiving an unsolicited communication from a second computing device that is remote from the first computing device, wherein the unsolicited communication identifies an application that is in an inactive state at the first computing device;

evaluating the unsolicited communication to determine the application; and

based on the evaluating:

launching the application, wherein the launching the application causes the application to enter an active state; and

reactivating, from an inactive state, a communication session supporting communication between the first computing device and the second computing device.

20. The article of manufacture of claim **19**, wherein the application is connected through a virtual session to a program executing on the second computing device.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO.	: 8,266,296 B2	Page 1 of 1
APPLICATION NO.	: 12/272481	
DATED	: September 11, 2012	
INVENTOR(S)	: Dowling et al.	

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

On Title Page 2, in Field (56), under "OTHER PUBLICATIONS", in Column 2, Line 18, delete "pages." and insert -- pages --, therefor.

In Column 5, Line 23, delete "Pentium™." and insert -- Pentium™ --, therefor.

In Column 11, Line 60, delete "UltaSparc™." and insert -- UltaSparc™ --, therefor.

In Column 11, Line 63, delete "Solaris™." and insert -- Solaris™ --, therefor.

In Column 11, Line 63, delete "WindowsNT™." and insert -- WindowsNT™ --, therefor.

In Column 11, Line 67, delete "UltaSparc™." and insert -- UltaSparc™ --, therefor.

In Column 12, Line 2, delete "service provider" and insert -- Service Provider --, therefor.

In Column 14, Lines 5-6, delete "PCAnywhere™. from Semantec" and insert -- PCAnywhere™ from Symantec --, therefor.

In Column 22, Line 67, delete "in tact" and insert -- intact --, therefor.

Signed and Sealed this
Fifth Day of February, 2013

Teresa Stanek Rea

Teresa Stanek Rea
Acting Director of the United States Patent and Trademark Office

EXHIBIT 3

US008291010B2

(12) **United States Patent**
Dowling et al.

(10) **Patent No.:** **US 8,291,010 B2**
(45) **Date of Patent:** ***Oct. 16, 2012**

(54) VIRTUAL CONNECTION OF A REMOTE UNIT TO A SERVER

(75) Inventors: **Eric Morgan Dowling**, Richardson, TX (US); **Mark Nicholas Anastasi**, Highland Village, TX (US)

(73) Assignee: **East Texas Technology Partners, LP**, Plano, TX (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 668 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **12/194,311**

(22) Filed: **Aug. 19, 2008**

(65) **Prior Publication Data**

US 2008/0310392 A1 Dec. 18, 2008

Related U.S. Application Data

(63) Continuation of application No. 10/920,817, filed on Aug. 18, 2004, now abandoned, which is a continuation of application No. 10/335,821, filed on Jan. 2, 2003, now abandoned, which is a continuation of application No. 09/167,698, filed on Oct. 7, 1998, now Pat. No. 6,574,239.

(51) **Int. Cl.**
G06F 15/16 (2006.01)

(52) **U.S. Cl.** **709/203**; 709/200; 370/349

(58) **Field of Classification Search** 709/200, 709/203; 370/349
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,285,061	A	8/1981	Ho
4,416,015	A	11/1983	Gitlin
4,489,416	A	12/1984	Stuart
4,578,796	A	3/1986	Charalambous et al.

(Continued)

FOREIGN PATENT DOCUMENTS

EP	0169548	1/1986

(Continued)

OTHER PUBLICATIONS

"Defendant Dell Inc.'s Answer to Plaintiff's Second Amended Complaint", Civil Action No. 2-03CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 15, 2004), 8 pages.

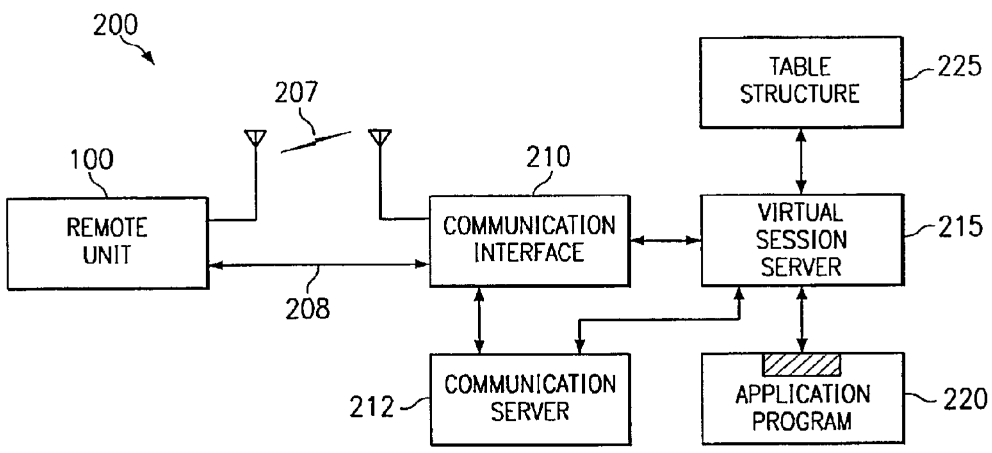
(Continued)

Primary Examiner — John Follansbee
Assistant Examiner — Anthony Mejia

(57) **ABSTRACT**

Apparatus and associated methods are provided which allow a remote user to maintain a virtual session with a server. A virtual session allows a remote and possibly mobile user to maintain a virtual presence in an office environment without actually being present. Using the present invention, a remote user can access a central application program such as an Internet service provider, a database system, an inventory system or billing system. Likewise, the remote user can receive calls and other forms of communications as though he or she were present in an office environment. A virtual session does not require a physical connection to be continuously present in order to provide a virtual connectivity. This is especially important for mobile applications where the remote user may incur long distance and/or wireless toll charges. Also, methods are presented to allow a remote unit to rapidly reconnect in a transparent and seamless way without burdening the user with the need to connect and reconnect or to upload and download information. Related methods are provided to allow the virtual session to be established, operated and maintained.

22 Claims, 4 Drawing Sheets



U.S. PATENT DOCUMENTS

4,852,151	A		7/1989	Dittakavi et al.	
4,995,074	A		2/1991	Goldman et al.	
5,027,269	A		6/1991	Grant et al.	
5,127,051	A		6/1992	Chan et al.	
5,287,401	A		2/1994	Lin	
5,321,722	A		6/1994	Ogawa	
5,329,619	A	*	7/1994	Page et al.	709/203
5,339,392	A	*	8/1994	Risberg et al.	715/762
5,367,563	A		11/1994	Sainton	
5,513,216	A		4/1996	Gadot et al.	
5,519,767	A		5/1996	O'Horo et al.	
5,539,885	A	*	7/1996	Ono et al.	709/203
5,550,908	A		8/1996	Cai et al.	
5,572,528	A		11/1996	Shuen	
5,600,712	A		2/1997	Hanson et al.	
5,604,769	A		2/1997	Wang	
5,606,719	A	*	2/1997	Nichols et al.	710/56
5,715,464	A		2/1998	Crump et al.	
5,724,412	A		3/1998	Srinivasan	
5,745,695	A		4/1998	Gilchrist et al.	
5,751,796	A		5/1998	Scott et al.	
5,757,890	A		5/1998	Venkatakrishnan	
5,764,639	A		6/1998	Staples et al.	
5,771,353	A	*	6/1998	Eggleston et al.	709/227
5,784,562	A	*	7/1998	Diener	709/217
5,802,293	A	*	9/1998	van der Sijpt	709/203
5,826,085	A	*	10/1998	Bennett et al.	719/316
5,842,199	A	*	11/1998	Miller et al.	1/1
5,857,201	A	*	1/1999	Wright, Jr. et al.	1/1
5,859,971	A	*	1/1999	Bittinger et al.	709/203
5,896,444	A		4/1999	Perlman et al.	
5,903,602	A		5/1999	Torkkel	
5,924,097	A		7/1999	Hill et al.	
5,928,363	A	*	7/1999	Ruvolo	726/22
5,958,006	A		9/1999	Eggleston et al.	
5,978,567	A	*	11/1999	Rebane et al.	709/219
5,982,774	A		11/1999	Foladare et al.	
5,999,947	A	*	12/1999	Zollinger et al.	1/1
6,023,493	A		2/2000	Olafsson	
6,038,602	A	*	3/2000	Ishikawa	709/227
6,052,779	A	*	4/2000	Jackson et al.	713/2
6,058,422	A		5/2000	Ayanoglu et al.	
6,085,222	A	*	7/2000	Fujino et al.	709/202
6,088,594	A		7/2000	Kingdon et al.	
6,088,600	A		7/2000	Rasmussen	
6,094,485	A	*	7/2000	Weinstein et al.	380/30
6,101,482	A	*	8/2000	DiAngelo et al.	705/26.62
6,101,531	A		8/2000	Eggleston et al.	
6,119,165	A	*	9/2000	Li et al.	709/229
6,119,167	A		9/2000	Boyle et al.	
6,134,582	A		10/2000	Kennedy	
6,156,336	A	*	12/2000	Bracht	424/449
6,157,630	A		12/2000	Adler et al.	
6,157,941	A	*	12/2000	Verkler et al.	709/202
6,199,110	B1	*	3/2001	Rizvi et al.	709/227
6,199,204	B1		3/2001	Donohue	
6,201,962	B1		3/2001	Sturniolo et al.	
6,216,151	B1	*	4/2001	Antoun	709/203
6,226,750	B1	*	5/2001	Trieger	726/3
6,247,055	B1	*	6/2001	Cotner et al.	709/227
6,263,016	B1		7/2001	Bellenger et al.	
6,266,701	B1	*	7/2001	Sridhar et al.	709/232
6,269,402	B1		7/2001	Lin et al.	
6,289,464	B1		9/2001	Wecker et al.	
6,295,549	B1		9/2001	Riddle	
6,301,590	B1	*	10/2001	Siow et al.	715/234
6,308,281	B1		10/2001	Hall, Jr. et al.	
6,317,455	B1		11/2001	Williams et al.	
6,336,147	B1	*	1/2002	Brownell et al.	719/310
6,393,467	B1		5/2002	Potvin	
6,397,253	B1	*	5/2002	Quinlan et al.	709/227
6,418,214	B1		7/2002	Smythe et al.	
6,421,707	B1		7/2002	Miller et al.	
6,426,946	B1		7/2002	Takagi et al.	
6,453,430	B1	*	9/2002	Singh et al.	714/47.3
6,456,603	B1		9/2002	Ismailov et al.	
6,463,078	B1	*	10/2002	Engstrom et al.	370/466
6,490,610	B1	*	12/2002	Rizvi et al.	718/101
6,496,572	B1		12/2002	Liang et al.	
6,542,489	B1		4/2003	Kari et al.	
6,546,425	B1		4/2003	Hanson et al.	
6,553,490	B1		4/2003	Kottapurath et al.	
6,560,239	B1	*	5/2003	Frid et al.	370/426
6,560,456	B1		5/2003	Lohtia et al.	
6,574,239	B1		6/2003	Dowling et al.	
6,584,321	B1		6/2003	Coan et al.	
6,594,682	B2	*	7/2003	Peterson et al.	718/102
6,594,692	B1		7/2003	Reisman	
6,681,017	B1	*	1/2004	Matias et al.	380/277
6,728,747	B1	*	4/2004	Jenkins et al.	718/101
6,975,710	B2	*	12/2005	Fujino et al.	379/93.09
7,206,816	B2		4/2007	Gorty et al.	
7,373,144	B1	*	5/2008	Kirkpatrick et al.	455/421
2002/0048354	A1		4/2002	Perlman et al.	
2003/0055327	A1	*	3/2003	Shaw et al.	600/407
2003/0156039	A1	*	8/2003	Tester	340/825.28
2004/0120277	A1	*	6/2004	Holur et al.	370/328
2005/0039048	A1		2/2005	Tosey	
2005/0233297	A1	*	10/2005	Guy et al.	434/350

FOREIGN PATENT DOCUMENTS

EP	0741481	11/1996
WO	9927702	6/1999

OTHER PUBLICATIONS

"Defendant International Business Machines Corporation's Answer to Plaintiff's Second Amended Complaint," Civil Action No. 2-03CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 15, 2004), 8 pages.

"Answer of Defendant Toshiba America, Inc. to Plaintiff's Second Amended Complaint," Civil Action No. 2-03CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 15, 2004), 5 pages.

"Gateway, Inc.'s Original Answer and Counterclaim," Civil Action No. 2-03CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 18, 2004), 8 pages.

"Hewlett-Packard Co.'s Original Answer and Counterclaim," Civil Action No. 2-03CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 18, 2004), 8 pages.

Agere Systems, Inc. "Complaint," Civil Action No. 2-04CV-108, in the United States District Court for the Eastern District of Texas, (Mar. 17, 2004), 6 pages.

"Plaintiff Agere Systems Inc.'s First Amended Complaint," Civil Action No. 2-04CV-108, in the United States District Court for the Eastern District of Texas Marshall Division, (May 11, 2004), 29 pages (including Appendix A).

"Plaintiff East Texas Technology Partners' Answer and Counterclaim to Intervenor Conextant's Complaint," Civil Action No. 2-03-CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Apr. 21, 2004), 10 pages.

"Emachines, Inc.'s Original Answer and Counterclaim," Civil Action No. 2-03-CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Apr. 22, 2004), 8 pages.

"Answer of Defendant in Intervention Conexant Systems, Inc. to Plaintiff East Texas Technology Partners' Counterclaim," Civil Action No. 2-03-CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (May 12, 2004), 6 pages.

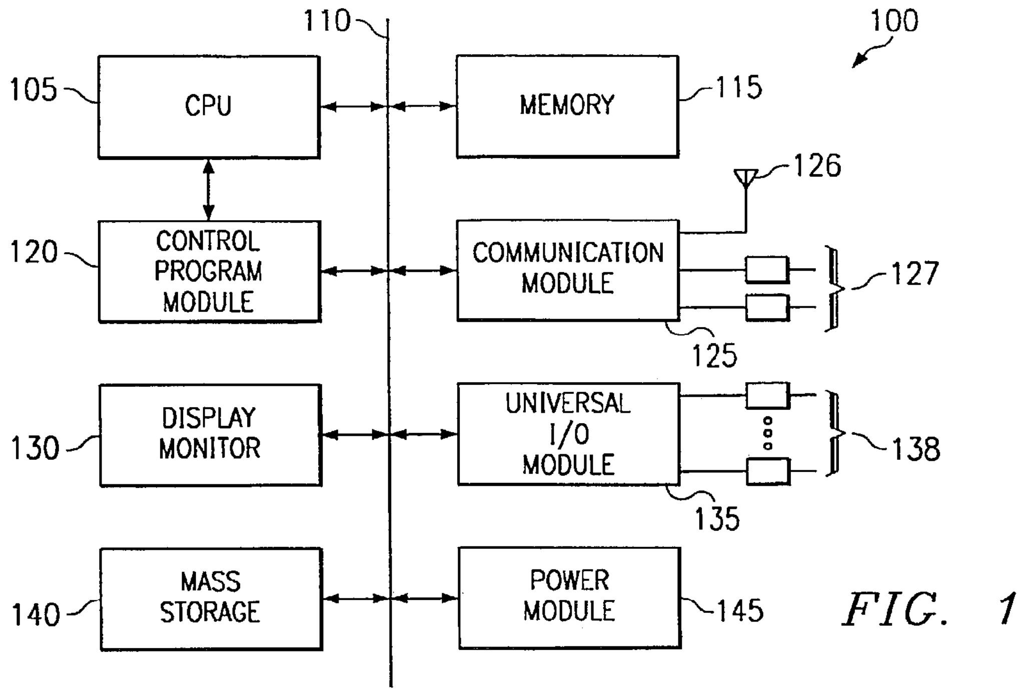
"Conexant Systems, Inc.'s Motion to Intervene," Civil Action No. 2-03-CV-465-TJW, in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 10, 2004), 27 pages (including Exhibits A & B).

"Answer of Defendant Acer America Corp. to Plaintiff's Second Amended Complaint," Civil Action No. 2-03-CV-465-(Ward), in the United States District Court for the Eastern District of Texas Marshall Division, (Mar. 29, 2004), 8 pages.

WAP Architecture Version Apr. 30, 1998, "Wireless Application Protocol Architecture Specification", Wireless Application Protocol Forum, Ltd., (Apr. 30, 1998), pp. 1-20.

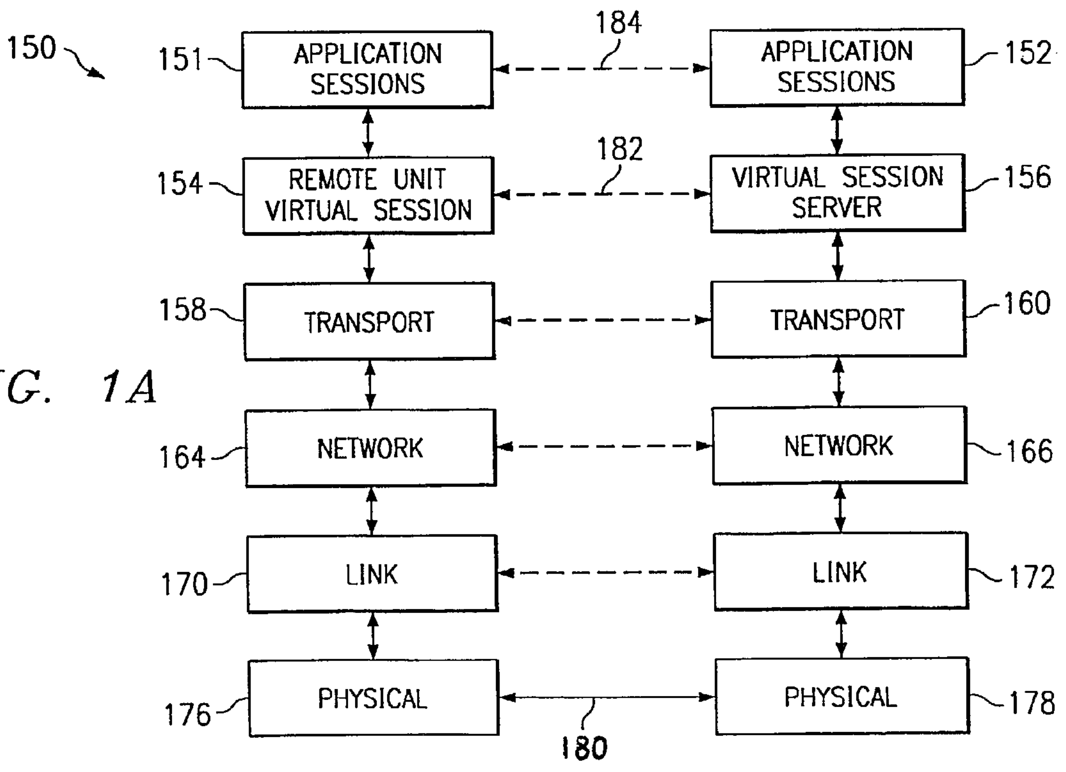
WAP WTA Draft Version Apr. 30, 1998, "Wireless Application Protocol Wireless Telephony Application Specification," Wireless Application Protocol Forum, Ltd., (Apr. 30, 1998), pp. 1-31.
"Defendant's Supplemental Preliminary Invalidity Contentions", Civil Action No. 2-03CV-465-TJW in the United States District court for the Eastern District of Texas Marshall Division, 73 pages.
Warrier, Padmanand, "Universal DSL Deployment of G.Lite", Texas Instruments Application Report SPAA007A, (Sep. 1998), 33 pages.
"239 Invalidity Chart for G.Lite Standard", no date, 26 pages.
"239 Invalidity Chart for European Patent No. 0169548", no date, 47 pages.
"239 Invalidity Chart for European Patent No. 0741481", no date, 22 pages.
Wireless Application Protocol, Wireless Session Protocol, Draft version Apr. 30, 1998, copyright Wireless Application Protocol Forum, Ltd., 1998, 95 pages.
Official Action in U.S. Appl. No. 12/272,481 issued Apr. 19, 2011, 7 pages.
Office Action in U.S. Application No. 12/272,481 issued Oct. 11, 2011, 7 pages.

* cited by examiner



110
100
105
CPU
MEMORY
115
126
120
CONTROL PROGRAM MODULE
COMMUNICATION MODULE
127
125
130
DISPLAY MONITOR
UNIVERSAL I/O MODULE
138
135
140
MASS STORAGE
POWER MODULE
145

FIG. 1



150
151
APPLICATION SESSIONS
184
APPLICATION SESSIONS
152
154
REMOTE UNIT VIRTUAL SESSION
182
VIRTUAL SESSION SERVER
156
158
TRANSPORT
TRANSPORT
160
164
NETWORK
NETWORK
166
170
LINK
LINK
172
176
PHYSICAL
PHYSICAL
178
180

FIG. 1A

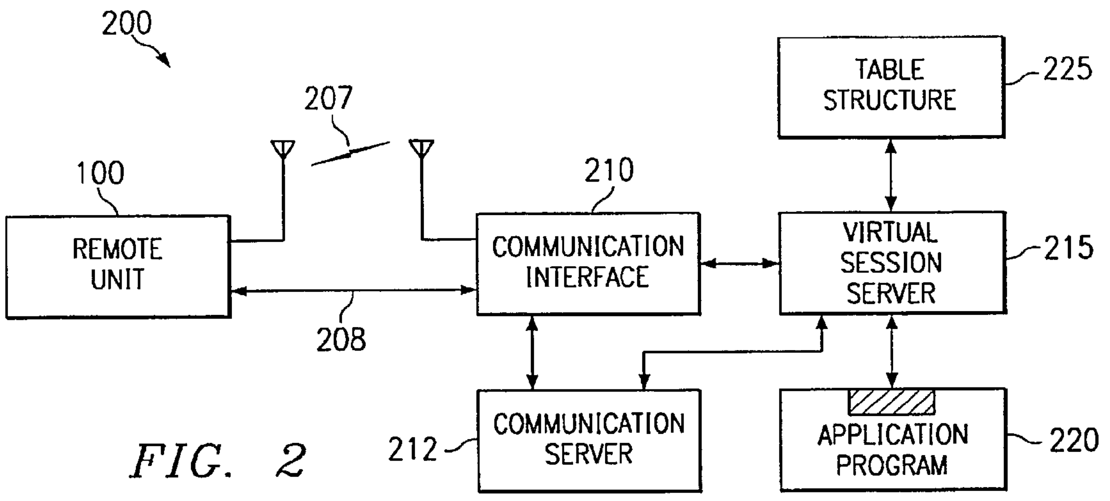

200
207
100
REMOTE UNIT
210
COMMUNICATION INTERFACE
208
TABLE STRUCTURE
225
VIRTUAL SESSION SERVER
215
212
COMMUNICATION SERVER
APPLICATION PROGRAM
220

FIG. 2

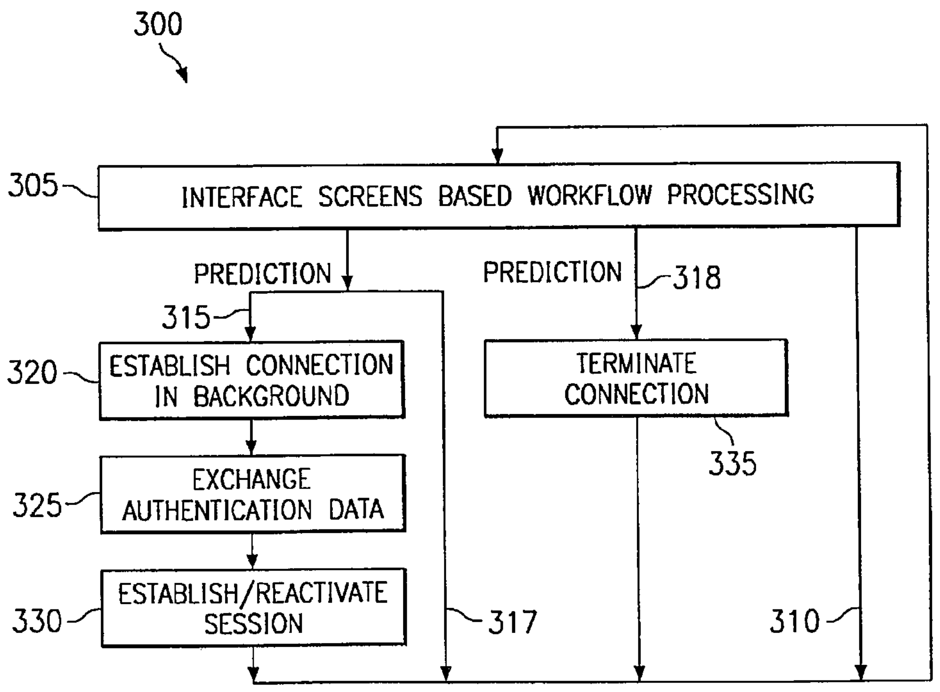

300
305
INTERFACE SCREENS BASED WORKFLOW PROCESSING
PREDICTION
PREDICTION
318
315
320
ESTABLISH CONNECTION IN BACKGROUND
TERMINATE CONNECTION
335
325
EXCHANGE AUTHENTICATION DATA
330
ESTABLISH/REACTIVATE SESSION
317
310

FIG. 3

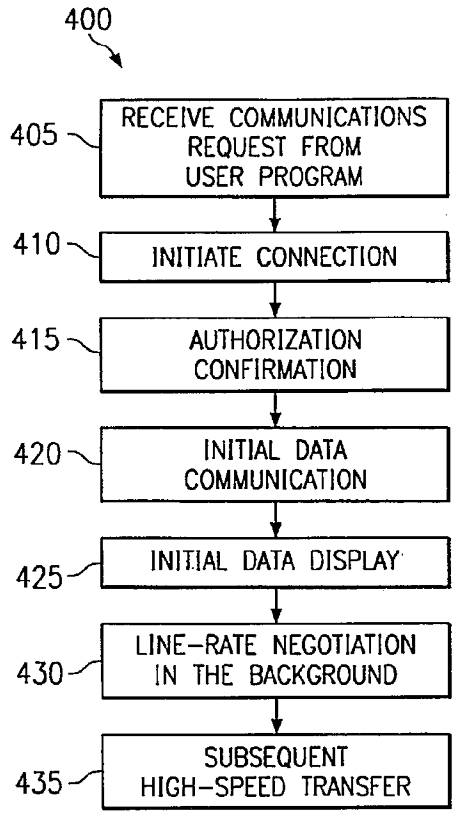

400
405 RECEIVE COMMUNICATIONS REQUEST FROM USER PROGRAM
410 INITIATE CONNECTION
415 AUTHORIZATION CONFIRMATION
420 INITIAL DATA COMMUNICATION
425 INITIAL DATA DISPLAY
430 LINE-RATE NEGOTIATION IN THE BACKGROUND
435 SUBSEQUENT HIGH-SPEED TRANSFER

FIG. 4

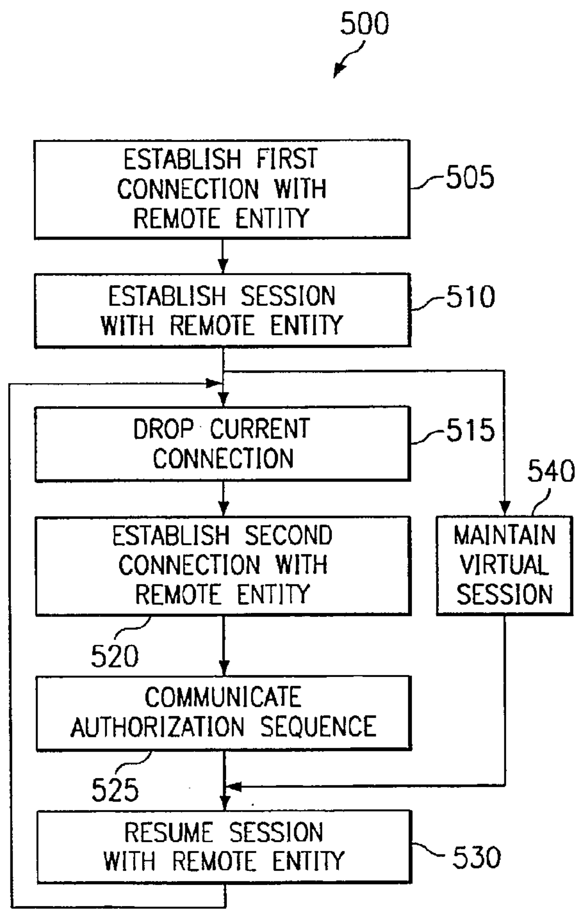

500
ESTABLISH FIRST CONNECTION WITH REMOTE ENTITY 505
ESTABLISH SESSION WITH REMOTE ENTITY 510
DROP CURRENT CONNECTION 515
540
MAINTAIN VIRTUAL SESSION
ESTABLISH SECOND CONNECTION WITH REMOTE ENTITY
520
COMMUNICATE AUTHORIZATION SEQUENCE
525
RESUME SESSION WITH REMOTE ENTITY 530

FIG. 5

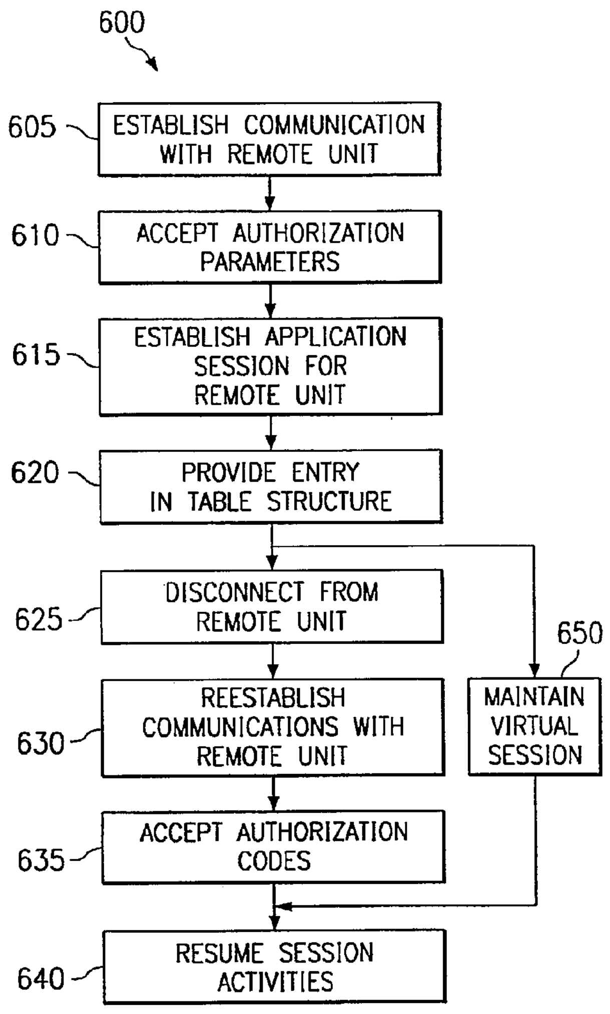

600
605 ESTABLISH COMMUNICATION WITH REMOTE UNIT
610 ACCEPT AUTHORIZATION PARAMETERS
615 ESTABLISH APPLICATION SESSION FOR REMOTE UNIT
620 PROVIDE ENTRY IN TABLE STRUCTURE
625 DISCONNECT FROM REMOTE UNIT
630 REESTABLISH COMMUNICATIONS WITH REMOTE UNIT
635 ACCEPT AUTHORIZATION CODES
640 RESUME SESSION ACTIVITIES
650 MAINTAIN VIRTUAL SESSION

FIG. 6

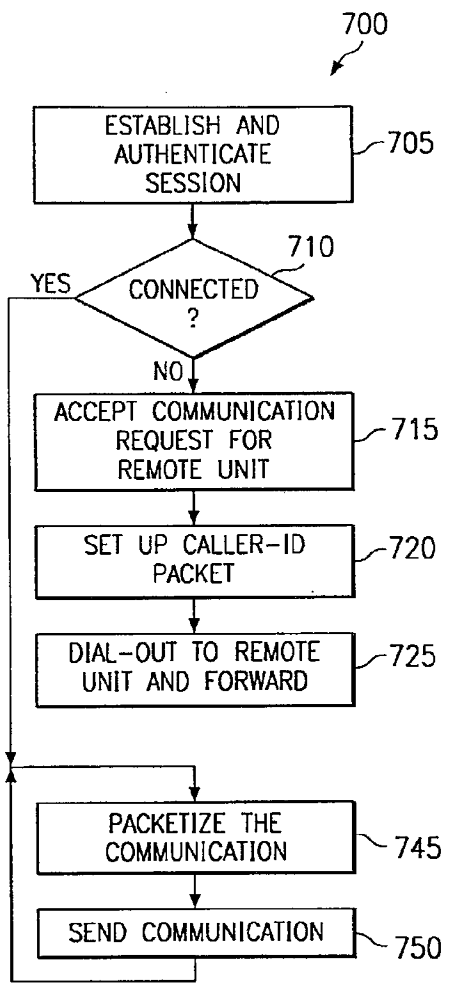

700
705 ESTABLISH AND AUTHENTICATE SESSION
710 CONNECTED ?
YES
NO
715 ACCEPT COMMUNICATION REQUEST FOR REMOTE UNIT
720 SET UP CALLER-ID PACKET
725 DIAL-OUT TO REMOTE UNIT AND FORWARD
745 PACKETIZE THE COMMUNICATION
750 SEND COMMUNICATION

FIG. 7

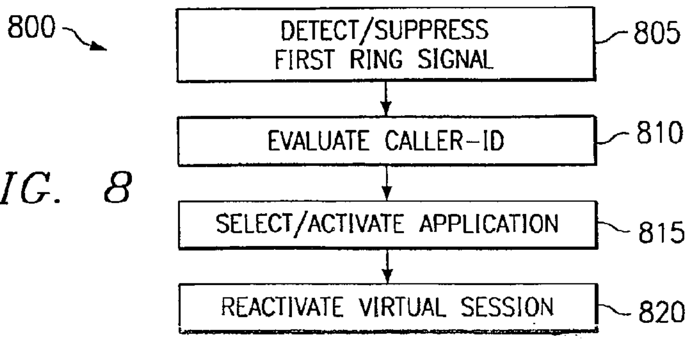

800
805 DETECT/SUPPRESS FIRST RING SIGNAL
810 EVALUATE CALLER-ID
815 SELECT/ACTIVATE APPLICATION
820 REACTIVATE VIRTUAL SESSION

FIG. 8

VIRTUAL CONNECTION OF A REMOTE UNIT TO A SERVER

CROSS-REFERENCE TO RELATED APPLICATIONS

The present application is a continuation of U.S. patent application Ser. No. 10/920,817, filed Aug. 18, 2004 now abandoned, which is a continuation of U.S. patent application Ser. No. 10/335,821 filed Jan. 2, 2003, now abandoned, which is a continuation of U.S. patent application Ser. No. 09/167,698, filed Oct., 7, 1998, now U.S. Pat. No. 6,574,239, entitled "Virtual Connection of a Remote Unit to a Server." The present application claims priority to these applications and incorporates them by reference herein in their entireties.

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates generally to client-server computing architectures and communication techniques. More particularly, the invention relates to a system whereby a mobile worker and a central server may maintain a virtually continuous connection without the need to maintain a physical connection continuously.

2. Description of the Related Art

The concept of a virtual connection has arisen in connection with telecommuting and related applications. Such a system is described in U.S. Pat. No. 5,764,639. A telecommuter dials into a server using a standard telephone line. The telecommuter's modem and a modem controlled by the central server establish a connection therebetween. Once a connection is established, the telecommuter may access a computer connected to the server, read emails and receive phone calls and faxes. For example, if a customer attempts to call the telecommuter at work by dialing into a private branch exchange (PBX), the server will convert the incoming call to a packetized form, such as H.323, and redirect the call via the existing connection between the telecommuter and the server. Using this system, the telecommuter may access a computer at work, answer phone calls and answer emails. The telecommuter thus appears to be present in his or her office and thus has a virtual presence there. Note for this system to properly function, the telecommuter must stay connected to the server at all times. While this does not present a significant problem for local telecommuting, this solution is quite costly for long distance telecommuting. Likewise, this solution is very costly if the telecommuter is mobile and must maintain a virtual presence with the server using a cellular wireless connection. Furthermore, in some areas it may be difficult to maintain a wireless connection continuously. A lost connection may also prevent one from regaining access to the system until some period of time has passed. Some mobile workers require only intermittent access to the server, but find it too inconvenient to place a dial-in call and to log onto the system every time access is needed.

There is a need to provide mobile workers with various forms of virtual connectivity. Mobile workers differ from telecommuters in that while a telecommuter typically works from a single home location or remote office, a mobile worker moves from location to location during the course of a normal working day. An example of a mobile worker is a home-care professional. A home-care professional is a medical worker who periodically travels to visit with different sets of homebound patients according to their individual needs. The individual patients each have a set of medical records indicative of their medical histories. A patient's medical record is preferably maintained as an interactive electronic document containing multiple parts. For example, the medical record indicates to the home-care professional precisely what procedures are to be performed and what medications are to be administered or otherwise given to the patient. Once the services are performed, the home-care professional must annotate the medical record accordingly. The medical record is updated to reflect the patient's vital signs and other information related to patient progress. Also, a billing system takes note to track expendables and services rendered. For example, the patient may be billed per visit and each visit may involve the expenditure of billable resources such as medicines.

In the above scenario, a mobile worker must interact with a central server during the course of a day. The worker may wish to access the central server while visiting a patient. The worker may also wish to access the server from a location where only a wireless connection can be established. From a performance perspective, an ideal solution is to provide the mobile worker with a wireless connection from a remote unit to a central server. Such a wireless connection could be established via a high-powered radio connection with a broad area of coverage or via an existing cellular or personal communication system (PCS) network. Solutions using high-powered radio links have the disadvantage that costly spectrum may be required. Maintaining a link on a cellular or PCS system is expensive in that a continuous connection consumes billable airtime which is also very costly. From an airtime-cost perspective, an ideal solution would be to force the worker to create a connection, download or up load information, and work locally with data on the remote unit as often as possible. This solution is tedious, and while saving airtime costs, may actually represent the more costly solution when professional service costs are factored in. This method has the added disadvantage that when files are uploaded or downloaded the data must be synchronized in case another user has changed the data in parallel with the mobile worker. Alternatively, other users must be "locked out" of the file from the time the mobile user downloads it until it is finally uploaded with any changes made. This is the problem solved using semaphores in shared memory systems. In the context of the present invention, a "file semaphore" is a semaphore used to lock a second user out of a file while a first user is using it. Due to the aforementioned reasons, in many applications forcing the worker to repeatedly connect, disconnect, upload and download information is unacceptable.

Some mobile networks have been constructed using what is known as cellular digital packet data (CDPD). In a CDPD network, a remote unit transmits a data packet on an unused analog channel. In this sense the mobile unit remains virtually connected to a CDPD communication server. Wireless airtime is only consumed when data is actually sent. A disadvantage to this approach is CDPD networks are not universally available. Cellular coverage is much more ubiquitous than CDPD coverage. Also, CDPD network subscribers must often pay high fees and hence CDPD may not represent the most economical solution.

In some systems such as packet switched network routers, offices make use of dial-out links. Dial-out links are useful when remote offices are separated by long distances. In such systems, when a packet must be routed from a first office to a second office, a call is placed to route the packet. The dial-out connection remains connected until a no-traffic condition is detected, indicating the line is no longer active. When the no-traffic condition is detected the connection is dropped until it is again needed. Dial-out links are thus used to reduce long distance fees associated with maintaining a constant connection, and represents a useful starting point for solving the foregoing problems relating to the establishment of a virtual presence of a mobile worker. Client-server protocols and fast automated connection strategies employing dial-out links are needed to provide new ways for a mobile worker to maintain a virtual presence. Also, new methods are needed to enable dial-out links to be set up with low delays to make them more useful for novel systems.

It would be desirable to provide a system whereby a remote worker could maintain a seamless connection with a central server without the need to maintain a dedicated channel. It would be desirable if the remote worker could communicate with the central server without the need to spend time to enter a password, reconnect, and wait for a line negotiation sequence to proceed before being able to use the connection. It would be desirable for a protocol stack to activate a virtual session based on a prediction derived from a workflow. It would be desirable to use this prediction to set up a connection in the background without disturbing the mobile worker while the mobile worker performed tasks in a workflow. It would also be desirable to have a remote unit which contains most of the screen-related information needed to provide the appearance of an established connection before the connection has been fully established. It would be desirable for the remote unit to download information before it is needed and upload information after it is gathered without the user even being aware these actions are being performed. It would further be desirable to establish a virtual session using a first communication medium such as a landline and to later communicate using the same virtual session using a second communication medium such as a wireless link. This would allow a mobile worker to select the most economical or convenient means of communications at a given time. In embodiments involving modem-based connections, it would be desirable to transmit data immediately using instantly available but lower line speeds. It would be desirable to then negotiate a higher line speed in the background while the remote worker and/or the server perform other tasks. Moreover, it would be desirable to establish a session between a remote unit and a server so that various forms of communications may proceed while providing the user with the appearance the user is continuously connected to the server and has a virtual presence with the server.

SUMMARY OF THE INVENTION

The present invention solves these and other problems by providing systems and methods to enable a remote worker to stay virtually connected to a central server without the need to continuously remain connected via a physical channel. The present invention is useful when costs are associated with maintaining a connection, for example when the connection has associated with it long distance, wireless, or other usage-related toll charges.

A first aspect of the present invention involves a communication protocol making use of a virtual session layer. The virtual session layer allows a communication session and an application session to be maintained in an inactive state when no physical connection exists. When a remote unit later reconnects with a server, the virtual session is placed into an active state and session communications resumes as though uninterrupted. A remote unit, a virtual session server, and a communication system including the remote unit and the virtual session server are presented to support virtual sessions communications. In one embodiment, the virtual session server manages a logon session between the remote unit and a server-side application program. The virtual session server

emulates the presence of the remote unit to the server-side application program and thereby maintains the logon. In related embodiments, the server-side application program involves a communication server capable of relaying messages and establishing communication channels with the remote unit using the virtual session layer.

A second aspect of the present invention involves a method of accessing a central server from a remote unit. A first step involves presenting a workflow to a user via a user interface. A second step involves predicting, based upon the workflow, when the user will require connectivity to the central server. Based upon the prediction and in the background, a third step involves initiating the establishment of a physical layer communication connection to the central server.

A third aspect of the present invention involves a method of establishing a connection with a low connection set-up time. In a first step, the method initiates the establishment of a communication connection to be used to communicate with a remote entity. Next the method communicates application layer data via the communication connection prior to the completion of a line-rate negotiation process. Next the method negotiates a line speed in the background.

A fourth aspect of the present invention involves a method of setting up and operating a virtual session. This method can be practiced by a client-side remote unit or a server-side virtual session server. A first connection is established to a remote entity. This first connection is then used to establish a set of parameters needed to define a communication session with the remote entity. Next the first connection disconnected and a set of communication session parameters are maintained. Next a second connection to the remote entity is established and an authorization sequence is communicated. The communication session is next reactivated using the communication session using the second connection. A related method is used to allow a remote unit to maintain a virtual communications presence with a remote communication server coupled to a virtual session server.

BRIEF DESCRIPTION OF THE FIGURES

The various novel features of the present invention are illustrated in the figures listed below and described in the detailed description which follows.

FIG. **1** is a block diagram representing an embodiment of a remote unit designed in accordance with the present invention.

FIG. **1**A is a block diagram illustrating a layered software architecture representative of the communication protocols of the present invention.

FIG. **2** is a block diagram illustrating a system comprising a remote unit operably coupled to a server via a communication medium.

FIG. **3** is a flow chart illustrating a method of processing whereby an application program implementing a workflow provides a prediction of when the user will need a connection and establishes a connection in the background just before it is needed.

FIG. **4** is a flow chart illustrating a method of establishing communication with a remote entity with a near-immediate set up time.

FIG. **5** is a flow chart illustrating a method of communicating by maintaining a virtual presence without the need to continuously maintain a physical connection.

FIG. **6** is a flow chart illustrating a method of processing performed on a server acting as a front-end to an application program to maintain sessions for remote users who are not continuously physically connected to the application program.

FIG. **7** is a flow chart illustrating a method of processing performed on a server managing virtual connections for users who are not continuously physically connected to the server.

FIG. **8** is a flow chart illustrating a method of processing performed by a remote unit to accept different types of incoming calls.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

FIG. **1** is a block diagram representing an embodiment of a remote unit **100** designed in accordance with the present invention. The remote unit **100** may be implemented as a laptop computer, a personal digital assistant, a desktop computer or workstation, or as a dedicated unit customized for a particular application. The remote unit **100** includes a central processing unit (CPU) **105** connected to a central bus **110**. The central processing unit may be implemented using an available microprocessor, microcontroller, or customized logic. For example, a Pentium™. processor from Intel Corp. may be used to implement the CPU **105**. The central bus is preferably constructed as a set of unbroken wires used to carry signals between a set of component subsystems within the remote unit **100**. It should be noted, in some embodiments of the present invention, the bus **110** may be implemented equivalently using a set of direct parallel and/or serial connections between individual modules. The bus **110** as illustrated in FIG. **1** shows a low cost and a preferred means to connect the illustrated subsystems. Any combination of bus connections and direct serial or parallel links may be used to implement the connection structure provided by the bus **110**. Different implementations represent different price-to-performance ratios and will be dictated by the needs of an individual embodiment. The bus **110** also comprehends multi-layered bus structures. For example, some embodiments make use of a local processor bus connected to the CPU **105**, and a peripheral interconnect bus for other subsystems. In multi-layered bus based designs, the different layers are preferably connected by bus bridges. All of these and other equivalent embodiments of the bus **110** are known to the skilled artisan. From here forward, the discussion will center on the illustrated embodiment of the remote unit **100** whereby all subsystems are directly connected via the bus **110**. Embodiments where the bus **110** represents a different physical interconnection topology are implicitly included in the discussion below.

A memory **115** is also coupled to the bus **110**. The memory **115** may be implemented using static random access memory (SRAM) or dynamic random access memory (DRAM). One type of SRAM is read-only memory (ROM). Preferably the memory **115** includes a ROM for use at boot-up, and a DRAM to hold a significant amount of data storage for use while executing programs. The remote unit **100** also includes a control program module **120**. The control program module **120** is controllably coupled to the CPU **105** and is also coupled to the bus **110**. The central program module **120** typically exists as a software module executed from the memory **115** by the CPU **105**. The control program module **120** effectively configures the remote unit **100** to operate in accordance with aspects of the present invention as discussed herein below.

A communications module **125** is also coupled to the bus **110**. The communications module includes at least one communication interface to allow the remote unit to communicate

with a remote entity such as a virtual session server as will be discussed in detail hereinafter. In a preferred embodiment, the communications module **125** includes a plurality of communication interfaces. For example, a first interface **126** provides a wireless link, and a second communication interface **127** provides one or more wireline links. Also, the wireline communication interface **127** may include a standard telephone modem interface and a packet style interface designed to plug directly into an Ethernet connection to be coupled to a local area network (LAN), a wide area network (WAN) or the Internet. The Internet is the well-known and ubiquitous World Wide Web. In some embodiments, the communications module **125** includes a caller-identification packet processor. A caller-identification packet processor receives a caller-identification packet, extracts information therefrom, and passes the information to the CPU **105**. Caller-identification packets may be advantageously used to identify incoming calls with a virtual session as discussed in connection with FIGS. **7**-**8**. The communications module **125** may optionally include a voice interface to allow a user to engage in telephone conversations using the remote unit **100**. In this case a separate handset or a built-in handset may be used. Alternatively a speakerphone may be built into the remote unit using a microphone, a speaker, and an echo canceller.

The remote unit **100** also includes a display monitor **130**. The display monitor **130** is also coupled to the bus **110**. The display monitor **130** is preferably implemented using a liquid crystal display (LCD), although other display technologies may equivalently be used. Also connected to the bus **110** is an optional universal input-output (I/O) module **135**. The universal I/O module includes a coupling to a set of external devices **138**. The external devices are preferably data collection units as described below. The universal I/O module preferably provides a standard link layer interface to the software module **120** executing on the CPU **105**. The remote unit **100** also preferably includes a mass storage device **140**. The mass storage device **140** is also connected to the bus **110**. The mass storage device is preferably implemented using magnetic disk or optical disk technology, but any mass storage device, to include a non-volatile memory, may be used.

The remote unit **100** also includes a power module **145**. The power module is preferably and optionally coupled to the bus **110** to receive power management control information. The power module preferably includes a battery, an alternating current (AC) connector, a direct current (DC) connector, and a power management control interface. The AC connector allows the remote unit **100** to be powered from a standard 110 V wall outlet. The DC connector allows the remote unit **100** to be powered from a vehicle, for example by plugging the unit into a cigarette lighter outlet. Either of these connectors may be preferably used to also charge the battery in the remote unit. The power module **145** is coupled to supply power to a power bus which is connected to all subsystems. Depending on the power management configuration of an individual system, different subsystems may accept power from separate connections to allow portions of the remote unit to be selectively turned off while they are not being used.

The remote unit **100** is operative to execute an application program. The application program is operative to supply a sequence of interactive screens or a menu based interface to the user. The sequence of interactive screens or a particular usage of a menu based system implements a workflow. In an example embodiment, the remote unit **100** is carried by a home-care professional. The home care professional has a sequence of procedures which need to be implemented in the course of working with a patient. This sequence of procedures gives rise to the workflow implemented in the control program **120** which executes on the remote unit **100**. In the example embodiment involving a home-care professional, the universal I/O module is connected to a set of peripheral units to collect vital information such as blood pressure, temperature, insulin level and the like. Other information such as the patient's weight may be entered manually by the home-care professional as a part of the workflow. At certain times in the workflow, an external communication connection will be needed because data may need to be uploaded or downloaded to/from a central server. In accordance with the present invention, the remote unit **100** is operative to provide a seamless and transparent virtual presence with the central server. In general, the central server may itself be segmented into two or more individual central servers. The discussion herein focuses on an embodiment whereby a virtual presence is maintained with a single central server having multiple server components. The present invention may be equivalently practiced by embodiments involving a virtual presence with more than one central server. Thus, one remote unit could maintain multiple virtual connections to totally separate server systems. In such a configuration the application workflow would dictate to which server system the remote unit would physically connect while other servers remain virtually connected.

A key aspect of the operation of the remote unit **100** is its ability to maintain a virtual presence with the central server without continuously maintaining a physical connection. The remote unit **100** is operative to provide communications when it is needed without the user needing to go through a set of normally associated connection sequences. For example, in accordance with one aspect of the invention, the user need only interact with the screens provided to implement the workflow while the remote unit **100** automatically sets up a connection in the background to be available when it is needed. In embodiments where file synchronization is not an issue or is handled using file semaphores, the software implementing the workflow automatically downloads information before it is needed and later automatically uploads new information after it has been gathered. This way, users need not even be aware they are not connected at all times. The user is not burdened with the need to connect and reconnect, and need not be burdened with downloading and uploading data. The user experiences the full benefit of being continuously connected to the central server without the associated cost of remaining continuously connected via a physical connection. In systems where file semaphores are not employed, the physical connection is established just before the workflow indicates it will be needed and is dropped when the workflow indicates it will not be needed for some time. Further details of the operation of the remote unit **100** are given in the discussions provided in connection with FIGS. **2-8**.

A central aspect of the present invention involves the concept of a "virtual session." A session as defined herein is similar to the definition provided in the open systems interconnect (OSI) reference model from the International Standards Organization (ISO). The OSI model is a model of a layered software structure used in computer communications. A software system which implements a layered model of communication is known as a "protocol stack." The OSI model is well known and divides a computer communications process into seven layers. At each layer is a software module which communicates with a peer software module at the same layer. Within a protocol stack, each layer communicates with the layer above and/or below. Actual communication systems often deviate from the seven layer OSI model. A protocol stack using basic concepts similar to the seven layer OSI model is next discussed which represents an aspect of the present invention.

FIG. **1**A illustrates a representative protocol **150** used to support the present invention. At the top layer is an application session layer. A first protocol stack with an application session layer software module **151** communicates with a second protocol stack with an applications session layer software module **152**. The application session layer software module **151** is typically implemented as a client-side software module which presents a user interface to a user. The application session layer software module **152** is typically implemented by a server-side software module operative to provide communication and/or computer related services to the client-side software module. For example the application sessions layer software module **152** may involve a logon session with a database program, or may represent a unified messaging server supplying voice mail, email and fax mail. Similarly, the application session layer software module **152** may involve a telephony application operative to provide a packet switched or a circuit switched telephone connection to the client-side application session layer software module **151**. One layer down in the protocol stack is a virtual session layer. In the example embodiment, the first protocol stack implements a virtual session layer software module **154** in the remote unit **100**. The virtual session layer software module **154** communicates with a peer virtual session software module **156** via the peer-to-peer communication path **182**. In the exemplary embodiment, the virtual session layer software module **156** is implemented within a virtual session server as discussed in connection with FIG. **2**. The virtual session server typically maintains a table linking one or more application sessions to a virtual session. For example, this linking of application sessions into the table structure may be accomplished by including a pointer to a data structure containing application session control data, or by placing the data structure holding the application session control data directly in the table structure. Additionally, the table structure allows the virtual session server to maintain a plurality of virtual sessions with a plurality of client remote units. In the OSI model, the OSI-session layer provides a set of rules used to establish and terminate data streams between nodes in a network. A set of OSI-session layer services include establishing and terminating node connections, message flow control, dialog control, and end-to-end data control. The session layer controls dialogs, which involve conversational protocols as used in mainframe computer terminal communications. The virtual session layer **154**, **152** of the protocol **150** may perform any of these functions in addition to maintaining the table linking to the application sessions.

The next software modules in the first protocol stack are the transport layer **158**, the network layer **164**, the link layer **170** and the physical layer **176**. These software modules respectively perform peer communications with the server-side protocol stack's software modules **160**, **166**, **172**, and **178**. The physical layer defines the low-level mechanical and electrical channel protocols and the physical connection itself. These four lower layers are well known in the art of data communications and can be implemented in various well-known ways. Likewise, alternative and equivalent protocol stacks may be constructed, for example, with the transport layer removed, various layers merged into one, or new layers added.

An important aspect of a virtual session oriented communication protocol such as the protocol **150** is the ability to maintain a peer-to-peer virtual session communication path **182** without the presence of a physical layer communication path **180**. The physical layer communication path **180** represents a physical layer communication connection, for example, a wireline connection, a cellular wireless connection, or a network connection to the Internet. When the physical layer communication path **180** is disconnected, no physical channel exists between the client-side software and the server-side software, and the physical layer communication path **180** is said to be in a "disconnected state." However, data structures maintained at the virtual session layer allow one or more peer-to-peer application session communication paths **184** to remain in a deactivated but existent state, even when the physical layer communication path is in the disconnected state. Likewise, the virtual session communication path **182** established between the remote unit and the virtual session server also remains in a deactivated but existent state. This is made possible through the use of the table structure maintained in memory which retains its information after the physical layer communication path **180** has been disconnected. When the physical layer communication path **180** has been reconnected, the physical layer communication path is said to be placed into a "connected state." At such time, the virtual session layer software modules **154** and **156** are operative to reactivate the virtual session layer communication path **182** and the application sessions layer communication path **184**. When these paths are reactivated, peer-to-peer communication may once again proceed over the application session layer communication path **184** and the virtual session layer communication path **182**.

As defined herein, a distinction is made between a communication session and an application session. A communication session is defined as a session between nodes or communication endpoints, and an application session is defined as a session between applications. For example, a remote unit may establish a communication session with a central server. In this case a communication session is established between the communication endpoints, i.e., the remote unit and the central server. Also, an application program running on the remote unit may need to establish an application session with an application program running on the central server. In such case an application session is created using a connection stream provided and governed by the communication session. A table structure is used to maintain both the communication session parameters and the application session parameters. For example, a first user authentication parameter may be used to establish a communication session with the server. A second user authentication parameter may be used to establish an application session with the application program. This second user authentication parameter may include a user identification parameter and a password, for example.

In light of the aforesaid concepts, a "virtual session" is next defined. A virtual session is preferably implemented as a communication session as defined above. A virtual session, like an OSI session, provides a set of rules for establishing data streams between nodes or endpoints. The virtual session also may provide other session features such as dialog control, message flow control, and end-to-end data control. A virtual session is controlled using a data structure which provides a way to associate the virtual session with the lower layers of a protocol stack, leading down to a physical layer. As mentioned above, in most embodiments, a virtual session is implemented as a communication session. Application sessions are then added onto the virtual session as connection streams within the communication session.

In a virtual session, a communication session may be suspended with some or all of the lower layers of the protocol stack missing. In particular, a virtual session may be maintained while a physical layer connection has been removed. The virtual session can then be reassociated with a physical

layer connection at a later time. The virtual session thus also preferably provides connect and reconnect rules used to establish a virtual session and then to reassociate the virtual session to a new physical connection to set up a new data stream in support of a dialog at a later time. Related activities such as the initiation of dial-out links to reestablish a physical layer communication path is also preferably handled by the virtual session in response to a signal from an application layer program.

An aspect of a virtual session is the maintenance of an application between an application program and a virtual session server as will be described below. A virtual session server acts as a proxy agent for a remote unit. When the remote unit is not connected via a physical layer communication path, the virtual session server maintains a proxy-presence with the application program on behalf of the disconnected remote unit. At a later time, when the remote unit reconnects into the virtual session by passing a set of communication session authentication parameters, the remote unit is thereby granted access to one or more application sessions which have been maintained in proxy by the virtual session server.

In a preferred embodiment, the virtual session uses a set of authentication parameters and a set of encryption keys to maintain a secure connection. A separate set of authentication parameters is used by an application running on the remote unit to gain access to an individual application session. Once the application session has been established over a virtual session, a table is used to maintain a set of parameters needed to maintain the application session, even though no physical layer connection exists between the endpoints of the virtual session. When a virtual session data structure is set up and no physical layer connection exists to support communication over the virtual session, the virtual session is said to be "inactive." When a virtual session data structure is set up and a physical layer connection does exist to support communication over the virtual session, the virtual session is said to be "active." A transition from an active state to an inactive state is called "deactivating a virtual session," and a transition from an inactive state to an active state is called "activating a virtual session." The process of transitioning from an active state to an inactive state is also known as "disconnecting from a physical connection." When this occurs, the physical layer connection is no longer available to support communication over the virtual session. In a preferred embodiment, a table structure is used to maintain the virtual session parameters as well as a set of parameters for each application session established over the virtual session. When a virtual session is activated, there is no need to reauthenticate the individual application sessions. This is because the table typically includes a user identification parameter, a user password, a set of application session parameters, a communication session identification parameter, and an encryption key for the communication session. Additional data such as modem initiation parameters may be added to the table as required by the system configuration and usage.

Referring now to FIG. **2**, a block diagram illustrating a system configuration **200** is shown. The system configuration **200** includes the remote unit **100** operatively coupled to a communication interface **210**. A direct wireless link **207** optionally couples the remote unit **100** to the communication interface **210**. A direct wireless link is used in embodiments where the remote unit **100** maintains a direct wireless link with the communication interface **210**. The communication interface thus provides an air interface for the direct wireless link. Alternatively or in addition to the direct wireless link **207**, a wireline link **208** couples the remote unit **100** to the communication interface **210**. The communication interface **210** maintains the connection **208** via a network interface coupled to a public switched telephone network (PSTN) or a network such as the Internet. This connection **208** may itself involve a microwave link, a wireless link through a public switched cellular network or a wireless link in a PCS network.

The communication interface **210** is preferably coupled to a communication server **212**. The communication server **212** may be thought of as generalization of a private branch exchange (PBX). The communication server **212** accepts tele-traffic from any variety of sources and provides switchable connections to couple different sources together. For example, the communication server **212** may be implemented as a PBX which receives a set of direct inward dial lines from a central office operated by the public telephone network. The PBX then provides local users with extensions and allows local users to call each other by dialing the last four digits of their telephone numbers. The PBX typically provides an outside line to a user once the user has dialed a nine.

The communication server **212** may also be configured to provide additional types of connections, such as packet based voice and video connections according to the H.323 international standard. In such an embodiment, the communication server **212** provides a gateway function passing calls between the public switched telephone network and a network such as the Internet. The communication server **212** may also provide other communications services such as voice mail, email, fax-mail, call distribution and the like. In systems involving Internet telephony, the communication server may operate only using packet protocols and not include an interface for circuit switched connections.

The communication interface **210** is also coupled to a virtual session server **215**. The virtual session server **215** is coupled to a table structure **225** and an application program **220**. The table structure **225** is preferably implemented as a software entity and may be located in a memory module within the server **215**. The virtual session server **215** may be implemented as a software entity which executes on a hardware platform. The hardware platform of the virtual session server **215** may be designed with an internal architecture similar to the remote unit **100** but is designed to provide a higher computation capacity and to handle multiple users. When supporting a virtual session server, the display monitor **130** is optional as users may control the virtual session server **215** remotely. The control program module **120**, when implemented in the virtual session server **215** provides the server side of the communication protocols discussed in connection with FIGS. **3**-**8**. Hence the remote unit **100** and the virtual session server **215** involve similar architectures and respectively implement the client and server sides of a set of virtual-session-related communication protocols of the present invention.

The application program **220** may execute on the same hardware platform as the virtual session server **215**. In general, both the virtual session server **215** and the application program **220** may be implemented as software modules running on personal computers, workstations, dedicated custom hardware, mainframe, or file servers. For example, the virtual session server **215** may be implemented as a software module running on an UltaSparc™. workstation or file server from Sun Microsystems Inc. The software may be written to execute over a multitasking operating system such as Solaris™. from Sun Microsystems Inc. or WindowsNT™. from Microsoft Inc. In a first preferred embodiment, the application program **220** includes a distributed database program running on a collection of networked servers such as Sun UltraSparc™. servers. In a second preferred embodiment, the application program may itself be a communication server as provided by an Internet service provider (ISP).

The system **200** is operative to implement a set of virtual session communication protocols according to the present invention. The remote unit **100** establishes a session via the virtual session server **215** to set up a virtual presence with the application program **220**. Preferably, the virtual session server **215** also provides a link to the communication server **212** to provide it access to the virtual session. When the remote unit **100** disconnects from a physical connection **207** or **208**, the virtual session is maintained within the table structure **225**. When the remote unit **100** later wishes to reestablish communication with the application program **220**, the virtual connection server **215** is operative to keep the virtual session active and to allow the user rapid and nearly transparent access to the application program **220**. Similarly, the virtual session also preferably is used to provide a virtual communication link between the communication sever **212** and the remote unit **100**. In some systems, a first virtual session is established between the remote unit **100** and the application program **220**, and a second virtual session is established between the remote unit **100** and the communication server **212**. The details of the operation of the virtual session server **215** and the virtual session protocols are discussed below in connection FIGS. **3**-**8**. Before proceeding to these portions of the detailed description, two embodiments of the system **200** are described.

In a first exemplary embodiment of the system **200**, a mobile worker such as a home-care professional operates the remote unit **100** to establish and maintain a virtual session with the application program **220**. In one embodiment, the application program **220** controls access to a database including complete medical and billing records for individual patients. Depending on working conditions, the home-care professional may require access from a wireless connection such as a cellular connection, or else may be able to communicate via a wireline connection provided within a patient's home. As the home care professional proceeds through a given workflow, the professional will eventually need to communicate with the application program **220**. When this time arrives, the present invention is operative to establish a physical connection between the home-care professional and the application program **220**. The professional need not be aware the physical connection has not been available since the time the virtual session was first established. The virtual session is maintained by the virtual session server **215** and the protocols of the present invention are employed to ensure such a virtual connectivity is provided without the need for the remote unit **100** to be continuously connected to the application program **220**.

In a second exemplary embodiment, the application program **220** is a communication server operated by an ISP. In this example, the remote unit **100** is operated by an Internet user. After the Internet user has remained inactive for a period of time, the connection **208** is terminated. At a later time, when the Internet user clicks on a hyperlink, thus demanding service, a short delay is incurred while the connection is reestablished. The remote unit is provided access without the user needing to reestablish a connection. When the user clicks on a hyperlink, the telephone is rapidly dialed without presenting dialing tones to the user. An authentication packet and a request packet are sent using a low data rate protocol such as one used for line-rate negotiation in modems. The user is authenticated by the server and the request packet is forwarded through the Internet to the Internet site referenced by the hyperlink. While the remote Internet server takes time to respond to the request, a higher line speed is negotiated in the background without burdening the user. Because a home Internet user uses the same analog connection between the user's premises and a network interface, the modem parameters may be preferably saved by the server in the table **225** to accelerate re-negotiation. The user is provided access almost immediately, and the connection is reestablished transparently. Note while this example focuses on an Internet application, the techniques apply to any application whereby a network site is accessed by activating a hyperlink.

As will be discussed below, the virtual session between the remote unit **100** and the virtual session server **215** provides a means to initiate transfers in both an uplink and a downlink direction. The uplink direction is from the remote unit **100** to the virtual session server **215**, and the downlink direction is from the virtual session server **215** to the remote unit **100**. A virtual session is said to exist between the remote unit and the virtual session server **215**. This virtual session may be used to create individual virtual sessions between the remote unit **100** and the application program **220**, and between the remote unit **100** and the communication server **212**. For example, an uplink connection is established, and when a home Internet user has been inactive for a period, the connection is dropped. As discussed above, the connection is reactivated transparently when the user once again activates an Internet link, as in an Internet browser. In the same example, a user may have an email reader program connected through a virtual session. If an email comes in for the user and the virtual session is in place, the email should be rapidly forwarded to the user. To do this, the user's phone is dialed in a downlink direction dial-out link by the virtual session server **215** via the communication interface **210**. The remote unit preferably suppresses the first ring and examines caller identification data. When the caller identification data indicates the calling party is the virtual session server **215**, the remote unit **100** automatically picks up the call and in this example, accepts the email. If caller identification is not used, a substitute protocol should be employed to assure that connection has been made to the proper application session defined within the virtual session. The substitute protocol preferably involves sending a packet header at the beginning of a call whereby the packet header contains one or more fields which identify associated the application session. Again, the user need not even realize a connection has been reestablished. Instead, the user receives the email message as though the connection had remained continuously active.

Another type of operation may occur when the user of the remote unit **100** is actively connected to the virtual session server **215** and a call comes in directed to the remote user's extension. At this point the call is preferably converted into packets and is sent to the user over the existing connection. In an alternative embodiment, the physical connection is automatically and temporarily dropped and the call is forwarded to the remote user. The virtual connection to the application is maintained through the virtual server. The communications module **125** preferably analyzes caller-identification data to determine the incoming call is a voice call to cause the optional telephone aspect of the remote unit **100** to ring. More details related to the foregoing system operation are discussed in connection with FIGS. **3**-**8** below.

The virtual session server **215** is able to maintain an open logon to the application program **220**. In one embodiment, the virtual session server **215** executes a client-side software which interfaces with the application program **220**. That is, if the application program **220** employs a client-server architecture, the application program **220** will implement a server-side software module which interacts with the client-side software. The server-side program performs database or other server oriented functions, while the client-side software provides a user interface to the user. The remote unit **100** can then control the operations of the virtual session server **215** using standard remote session software. An example of commercially available remote session software is PCAnywhere™. from Semantec Corporation. In another embodiment, the virtual session server executes the client-side software in parallel with the remote unit. In still another embodiment, the remote unit executes the client-side software, and the virtual session server merely provides a connection stream to pass data from the application program **220** to the remote unit **100**. When the virtual session is in a deactivated state, the virtual session server emulates the client-side software as needed to maintain an active session with the application program **220** in the absence of the remote unit **100**. A wide variety of equivalent techniques may be used to allow the virtual session server **215** to maintain a pointer or re-entry point into the application **220** while acting as a proxy agent to maintain the logon for the remote unit **100**. A table structure is preferably used to allow the virtual session server to simultaneously maintain a plurality of logons for a plurality of different remote units.

In some embodiments, the remote unit **100** may need to maintain a plurality of virtual sessions with a plurality of different virtual session servers. For example, an independent contractor may provide home-care services for two distinct health regions. Each health region may use a separate database. The remote unit **100** may then access these separate databases using a first and a second client-side application software module. During the course of a day, the remote unit may need to activate the first or the second client-side application software modules. In such case the remote unit **100** is operative to maintain a table structure similar to the table structure **225**. The table structure maintained by the remote unit links an application software module through an application session to a virtual session. When the first client-side application program demands access to a first database, the virtual session layer software **154** in the remote unit causes a physical connection to be established to support virtual session communications **182** with the first database application program. Likewise, if the second client-side application software module desires to access a second database, the virtual session layer software module **154** activates a physical layer connection back to the second database server. In other applications a single application program may be used which accesses information on more than one virtual session server. In such case a single application program can select the virtual session to activate based on the communications request generated from within the application program. In still other embodiments, a single physical connection **208** or **207** may be used to communicate with the communication interface **210**. The communication server **212** then forwards packets to a first local virtual session server such as the virtual session server **215**. If the received communication packets are destined for a second virtual session server, then the communication server **212** preferably forwards the packets to a remote virtual session server using a network connection such as an Internet connection.

Referring now to FIG. **3**, a method **300** is illustrated to show how the remote unit **100** preferably operates to activate a connection. The method **300** is preferably practiced by the remote unit **100** in support of a virtual session with the virtual session server **215**. A first step **305** of the method **300** involves actions within a workflow process **305**. The workflow process **305** includes the step **305** of the method **300** and also performs other activities to interact with a user's workflow requirements. Control loops from the first step **305** back to the first step **305** via a control path **310**. The workflow, as discussed above, is preferably made up of a menu system and/or a set of interactive screens traversed by a worker in performing a set of tasks. For example, a home-care professional's workflow involves accessing and displaying a patient's medical record, entering a set of data into the medical record, and performing tasks indicated by the doctor's directions as annotated in the medical record. In this example, as the home-care professional moves from one screen to the next, control loops via the control path **310**. The workflow process **305** is an application program which executes on the CPU **105**. The workflow process **305** is preferably implemented as a process running on the CPU **105** in a multitasking operating environment. A multitasking operating environment is one in which multiple programs or processes may execute in parallel by sharing time slots within the CPU **105**. Multitasking operating system software is well known and is readily available. In a multitasking-programming environment a first process may execute in a normal fashion and provide an interface to a user. At the same time a second process may be executed by sharing CPU cycles without the user's intervention or knowledge. In such a case the second process is said to be a background process or is said to perform background processing. At some point in the course of the workflow, a physical layer communication connection will be needed to communicate information between the remote **100** and the application program **220**.

When a step in the workflow process **305** is performed leading up to the need for a physical layer communication connection, control next passes from the first step **305** to a second step **320** via the control path **315**. The control path **315** is activated when the workflow process **305** provides a prediction indicating a physical layer communication connection will subsequently be needed. In some cases the prediction may be provided right when the physical layer communication connection is needed. In other cases, the prediction **315** may be used to initiate background processing to download data which will not be needed until a later time. In menu based systems, the prediction **315** may be learned by observing the workflow habits of a user. The prediction **315** is a function of the application program or workflow **305** and is optional. In the second step **320**, a connection is established in the background. Background processing enables the user to continue interacting with the workflow process **305** while a physical layer communication connection is simultaneously and transparently established. That is, the physical layer communication path is reestablished without inhibiting the user from interacting with the workflow **305**. Hence when control passes via the control path **315** to the step **320**, control preferably simultaneously passes via the control path **317** back into the workflow. The background process is preferably forked as a separate task and two execution flows proceed in parallel by time sharing the CPU **105**. Multitasking is well known in the art and is implemented using interrupt based processing. In alternative embodiments the control path **317** may be deleted and a single control flow may be implemented using the control path **315**. However, this embodiment may require the user to wait for the connection to be established and is hence not deemed to be the preferred embodiment of the method. Other equivalent embodiments set up the communication path transparently by multiplexing the CPU **105**'s computation cycles from within the workflow process or some other process.

Once control has been forked via the control path **315** to the second step **320**, a dialer within the communications module **125** preferably dials to establish a physical layer communication connection with the communication interface **210**. In embodiments using dedicated radio links, the connection may be established over the wireless link **207**. One preferred embodiment of a remote unit **100** incorporates a cellular radio. In this case the dialer dials a telephone number and a connection is established using a public switched cellular telephone network so that the connection is set up on the link **208**. Stationary Internet based embodiments perform the second step **320** by dialing a telephone number using an automatic dialer which dials a land line connection for a modem. In all cases, it is preferred to suppress the dialing tones and line-rate negotiation signals so the connection may be established transparently to the user.

Control next passes from the second step **320** to a third step **325**. In the step **325**, an authentication code is transmitted from the remote unit **100** to the communication interface **210**. This authentication code is then passed to the virtual session server **215**. The virtual session server evaluates the authentication code to determine if access is to be permitted. In a preferred embodiment, the authentication code involves a digital signature as is known in the field of public key cryptography. In a preferred embodiment, all transmissions are encrypted using public key cryptography. Some systems may be implemented using various encryption standards such as secure sockets layer based encryption. The amount of authentication and encryption used in any given embodiment is left to the system designer, but preferably all transactions are encrypted as described above.

Control next passes from the third step **325** to a fourth step **330**. In the fourth step **330**, a session is established/reactivated with the virtual session server **215**. The session is established the first time the method **300** passes control to the step **330**. Subsequently the step **330** is operative to reactivate the session with the virtual session server. When the session between the remote unit **100** and the virtual session server **220** is reactivated, virtual session communications resume. At this point, the virtual session server **215** correlates information stored in the table structure **225** with the connection and provides the remote unit **100** access to the application program **220**. If no data is stored in table structure **225**, access is provided to a default logon screen allowing remote unit **100** to establish a new application session. The virtual session server **215** then populates the table structure **225** to establish a virtual session. The step **330** involves setting up a stream connection between the workflow process **305** and a protocol stack. The protocol stack is operative to read information bits from the stream connection and communicate the bits across an external communication link. Bits received over the external communication link are converted by the protocol stack into information bits to be sent back to the workflow process **305** across the connection stream. Once the appropriate communication processes are configured, control next passes back to the workflow process **305**. Due to the aforementioned forking operation, the passing of control back to the workflow process **305** may have already occurred via the control path **317**. In this case the passing of control from the step **330** to the workflow process is not explicitly performed.

When control loops back from the fourth step **330** to the workflow process **305**, a physical layer communication connection is activated for current or subsequent communication. When the user gets to a point in the workflow where communication with the application program **220** is needed, the connection has already been transparently set up in the background. Hence the user gets the feel of being connected to the application program **220** all the time, where in fact the remote **100** is connected via a physical channel to the application program **220** only a fraction of the time. This virtual connection saves communication resources and money when a toll is charged based on the amount of usage on the link **207** or the link **208**. In some embodiments, the fourth step **330**, or an execution thread within the workflow **305** is operative to upload or download information in the background. This way the user has ready access to data contained in the application program **220**, but in general a shorter connect time is required. While with prior art systems it is burdensome for a user to connect to a central server and download and upload information, with the virtual session of the present invention the user need not even be aware this process is occurring. Rather the user feels as though he or she is continuously connected with a fast connection because the data needed at a given point in the workflow is already available locally or has been uploaded in the background transparently without user intervention. In systems where server synchronization is an issue, file semaphores and/or direct active sessions not employing uploading and/or downloading of records may be used.

Based on another point in the workflow, another prediction is made to predict when the communication channel will not be needed for some time. For example, it may be known, based on the workflow, the home-care professional will next perform a sequence of tests and enter data into a screen displayed on the remote unit. Only at a later time will the workflow come to a point where this information is to be uploaded to the application program **220**. When such a prediction is made, control passes from the first step **305** via the control path **318** to a fifth step **335**. The fifth step **335** is operative to deactivate the connection established over the link **207** or the link **208**. The step **335** may optionally involve forking a separate execution thread or otherwise accessing a separate process in a multitasking environment. Alternatively, the fifth step **335** may be performed by executing a set of instructions in the workflow process **305**. At a later time, a prediction may be made indicating the link **207** or **208** needs to once again be activated, whereby control again passes over to the second step **320** via the control path **315**. It should be noted different systems will typically set their prediction times according to the economic conditions involved. For example, in some systems the first minute of connection time may cost five times as much as all subsequent minutes. In this case predictions would be preferably set according to a criterion to minimize cost by not establishing and terminating connections more often than necessary. If a flat rate were charged per minute connections would be set-up and terminated more often. If automatic uploading and downloading is performed in the background, a very efficient use of air-time can often be achieved while presenting the user with the appearance of being continuously connected to the application program **220**.

Referring now to FIG. **4**, a method **400** of establishing a communication link with low delay is illustrated. The method **400** may be practiced by both the remote unit **100** and the communication interface **210**. This method is most applicable to systems involving modems whereby digital data is transferred over an analog channel requiring receiver training. Receiver training involves transmitting data sequences through a channel and allowing a receiver to adjust its receiver parameters. Receiver parameters include echo canceller and equalizer filter coefficients. Most systems also adjust their data rates and signal constellations based on observed conditions. In modems, this entire process is known as line-rate negotiation. Prior art systems involving receiver training are tedious to use because they force the user to wait while the receiver is trained. Most systems play the training signals though a speaker to allow the user to hear the training process. This lets the user know what the computer is doing for the duration of the delay. The method **400** improves upon this

prior art solution by allowing the user to gain almost immediate access without a significant delay.

In a first step **405**, a protocol stack or other process practicing the method **400** receives a communications request from a user program. For example, this occurs when a user clicks on an icon to initiate the establishment of an Internet connection. Control next passes to a step **410** where the connection is initiated. This step typically involves an automatic dialer dialing the number of an Internet service provider (ISP). The ISP software may be implemented as a communication server application corresponding to the application program **220**. In this case access to the application program is governed by the virtual session server **215**.

Control next passes from the second step **410** to a third step **415**. In the third step **415** an authorization sequence is exchanged. In a preferred embodiment public key cryptography involving digital signatures and keys is used. Embodiments involving a virtual session server **215** either set up a session or activate an existing session during the third step **415**. Control next passes from the third step **415** to a fourth step **420**. In the fourth step **420**, one or more initial application layer data packets are transmitted across the connection using a low speed protocol. A low speed protocol is used by the transmitter and receiver when performing line-rate negotiations. For more details of line-rate negotiation protocols, see, for example, the V.34 and V.90 standards from the International Telecommunications Union. In the present invention, the low speed protocol is used to transmit application layer data before the line-rate negotiation procedures have completed. This avoids the need for the user to wait for line-rate negotiation to complete before being able to access a communication path.

Control next passes from the fourth step **420** to a fifth step **425**. In the fifth step **425**, initial data is displayed. Software located locally in the remote unit **100** preferably contains high-volume graphics related data so that the initial data exchange of the step **420** only requires a small amount of data to be transferred. For example, the user logs onto the Internet and almost immediately sees a screen of information indicating the user is connected and the system is ready to accept inputs. This is made possible by displaying locally held screens of graphical data and allowing a small amount of specific information such as time, date, and headlines to be received and displayed. If the user then immediately clicks on a link, an application layer request packet is sent using the line-rate negotiation protocol's data format. This allows the user to immediately begin making requests before the line-rate negotiations have completed. In many cases the user will pause and read the headline information, giving the system even more time to perform line-rate negotiation in the background.

Control next passes from the fifth step **425** to a sixth step **430**. In the sixth step **430**, a background process is forked to perform line-rate negotiation. Line-rate negotiation is allowed to proceed in the background while the user is reading the information provided on the initial display of the step **425**. Likewise, if the user had rapidly clicked on a link, a request packet is sent out and while the server is responding to the request, the background line-rate negotiation may proceed. The step **430** is operative to perform line-rate negotiation so as to set up a high-speed connection for subsequent higher volume data transfers. In embodiments involving a virtual session server **215**, the user's line speed parameters may be stored in the table structure **225**. For example, if the user is an Internet user and the application program **220** is an ISP, the user will often dial in from the same location. Thus parameters derived in a previous activation of a communication channel will be either identical or similar to those used in a current activation. Hence the sixth step **430** optionally involves accessing from the table **225** a set of starting parameters derived from the activation of the communication channel. If communication is needed before the line-rate negotiation has completed, communication preferably proceeds at the highest rate negotiated up to that point.

Once the line-rate negotiation process of the step **430** has completed, control passes to a seventh step **435**. In the step **435**, communication is able to proceed at full speed. In most cases where this method is implemented, the user will get the full benefit of being connected almost immediately without the normal delay associated with prior art systems. This is so because initial low-volume data is allowed to pass through the channel before the line-rate negotiation has completed. Line rate negotiation then proceeds in the background in parallel with other activities such as the user reading headline information or a distant server accessing data and responding to the initial data request packet sent across the Internet. This technique is useful when a user is maintaining a virtual session with a remote server because it is imperative to allow the user to appear to be connected without having to experience delays when accessing data. The method **300** and the method **400** may be performed together in a complementary fashion to make the virtual session appear to be constantly available.

The method **400** may be practiced by the remote unit **100** and the virtual session server **215**. When the remote unit **100** initiates the method, the virtual session server **215** executes steps **410**, **415**, **420**, **430** and **435**. When the virtual session server **215**, the application program **220**, or the communication server **212** initiates the method, one or a combination of these servers practice the steps **405**, **410**, **415**, **420**, **430**, and **435**. The first step **405** involves, for example, receiving a communication request such as a telephony call or an email for the remote unit **100**. In some systems, the first step **405** may involve a request generated from within the application program **220**.

Referring now to FIG. **5**, a method **500** of establishing and operating a virtual session is illustrated. For example, the method **500** establishes a virtual session between the remote unit **100** and the virtual session server **215**. The method **500** is practiced by both the remote unit **100** and the virtual session server **215**. In a first step **505** a first physical layer communication connection is established with a remote entity. If the method is practiced by the remote unit **100**, then the remote entity typically corresponds to the virtual session server **215**. The virtual session may be used to support virtual sub-sessions between the remote unit **100** and the application program **220**. Also, a virtual sub-session may be established between the remote unit **100** and the communication server **212**. For the purposes of discussion herein, all of these virtual sessions will be referred to simply as virtual sessions. If the method **500** is practiced by the virtual session server **215**, then the remote entity typically corresponds to the remote unit. The step **505** may be activated according to the prediction **315**, and the step **505** may use the method **400** to allow the connection to be set up with very low delay.

Control next passes from the first step **505** to a second step **510**. In the second step **510** a session is established with the remote entity. In a preferred embodiment, this involves exchanging password information and agreeing upon a set of keys to encrypt data transacted in the session. Also, the virtual session server **215** preferably sets up a table entry in the table structure **225**. The table entry indicates the presence of a virtual session. The table entry may include modem parameters as discussed in connection with FIG. **4**. Also, the virtual session as set up in the table entry links the remote unit to a

user identification and a password as presented to the application program **220**. For example, a user name and a password may be used as user authentication parameters. Preferably public key encryption is used to encrypt all information so the password sent from the remote unit **100** to the application program **220** cannot be effectively intercepted. The remote unit **100** also preferably sets up a virtual session data structure to hold similar information related to the virtual session. Once the virtual session has been set up, the remote unit **100** may access the application program **220**. Also, the remote unit **100** may optionally access the communication server **212** for communication services.

Control next passes from the second step **510** to a third step **515**. In the third step **515**, the physical connection established in the first step **505** is dropped. Meanwhile the virtual session data structures and table entries established in the second step **510** are retained. The session is allowed to proceed while no physical layer connection exists. That is, the step **510** is operative to set up a table structure including one or more data structures which allows a virtual session to be maintained in memory while other activities occur. Hence a passive background thread of execution passes from the step **510** to a passive step **540** whereby the virtual session is maintained. This allows the remote unit **100** to stand by or be used for steps of the workflow process **305** not requiring communication with the application program **220**. Once the user needs to communicate with the application program **220**, or when a prediction **315** is made, control next passes from the third step **515** to a fourth step **520**. The step **520** is operative to reestablish a second physical layer communication connection to allow communication to proceed once again using the session established in the second step **510**. This connection reestablishment may be performed in response to the prediction **315** and may use the low-delay connection establishment technique of the method **400**.

In some embodiments, the present invention involves using distinct and separate communications media to perform the step **505** and the step **520**. For example, a mobile worker may call in from home to set up the virtual session in the step **510** using the first physical layer communication connection established in the step **505**. Later in the day, the worker may call in from a restaurant while catching up on some records keeping. This second use of the virtual session involves use of the second physical layer communication connection which in this example is a wireless connection different from the landline connection used to initiate the session from home earlier in the day. At a still later time, the worker may call in from a patient's home via a third physical layer communication connection while performing home-care services. If modem starting parameters have been stored in table **225**, they are preferably updated whenever the communication connection is changed. Hence the virtual session of the present invention enables a mobile worker to continue communications via the most expedient and/or economical means without causing the user to have to reestablish a communication connection. Preferably, when the remote unit **100** is connected to a communications source via the connector **127**, the remote unit **100** automatically detects this connectivity and makes use of it for subsequent virtual-session communications. That is, the present invention contemplates the availability of various forms of "pigtail" connectors being available so the remote unit **100** can operate in a "plug-and-play" fashion. Pigtails may be supplied to allow the remote unit to connect to the PSTN, the Internet, or to another computer via a universal serial bus, for example.

Control next passes from the fourth step **520** to a fifth step **525**. In the fifth step **525** an authorization sequence is exchanged. This is preferably implemented using public key encryption and digital signatures. Some embodiments may be developed which do not implement the fifth step **525**, but preferred embodiments do make use of user authentication. After the fifth step **525** has completed, the session is resumed in a sixth step **530**. Over the course of the virtual session, control may loop back to the third step **515** as many times as the virtual session is activated with a new physical connection. When the sixth step **530** is entered, the virtual session is once again activated so that the passive step **540** also passes control to the sixth step **530**. In a minimal implementation of the method, no looping occurs and the method terminates after the first pass through the sixth step **530**.

Referring now to FIG. **6**, a method **600** practiced by the virtual session server **210** is illustrated. In a first step **605**, a first physical layer communication connection is established for communicating with the remote unit **100**. Control next passes to a second step **610** whereby a set of authorization parameters are accepted and authenticated. As discussed in connection with FIG. **5**, the authentication parameters preferably include the exchange of public keys which include a digital signature in accordance with public key cryptography. Control next passes to a third step **615** where a user identification and a password are passed by the virtual session server **215** to the application program **220** on behalf of the remote unit **100**. As discussed in connection with FIG. **5**, the user identification and the password to be presented to the application program **220** are preferably transmitted in encrypted form. Once the application program **220** authenticates the user identification and password needed to gain access, the virtual session server **215** enters an entry into the table structure **225** to hold a set of session parameters. The session parameters include the user identification, a session identifier, encryption data and possibly other data such as modem starting parameters. Once the session has been logged into the table, the user may use it to communicate with the application program **220**.

Control next passes from the fourth step **620** to a fifth step **625**. In the fifth step **625** the physical layer connection is dropped. This step is performed when the remote unit does not currently require communications with the application program **220**. In step **650** the virtual server maintains the application session while the physical connection is disconnected. At a later time, when the user needs access to the application program or when the prediction **315** is made, control next passes to a sixth step **630**. In the sixth step **630** a second physical layer connection is established to allow communication between the remote unit **100** and the application **220** to resume. As discussed in connection with FIG. **5**, the second physical layer connection may involve a different communication path and/or medium as was used for the first physical layer connection. That is, a plurality of communications media are preferably supported to allow the user to call in via different means, for example via a wireless or a wireline connection. The step **630** may be initiated due to actions at the remote unit **100** or in response to events occurring in the server. For example, the communication server **212** may receive a call for the remote unit. Alternatively an email may be received which needs to be forwarded to the remote unit. In such a case, the sixth step **630** optionally involves sending a caller identification packet to let the remote unit know what type of communication, such as a voice telephony call, an email, or a fax, is inbound. A caller identification packet is a sequence of information bits sent across a communication connection identifying the calling party of the connection. In standard telephone systems, the caller identification packet is transmitted between the first and second rings when the telephone call is being set-up. More details relating to communications initiated by the virtual session server **215** back to the remote unit **100** are discussed in connection with FIG. **7**.

Once the second physical layer communication connection is established in the sixth step **630**, possibly according to the method **400**, control next passes to a seventh step **635**. In the seventh step **635**, authorization codes are verified similarly to the second step **610**. Once the user codes have been verified to be correct, control next passes to an eighth step **640** whereby communication once again resumes using the previously established virtual session.

Referring now to FIG. **7**, a method **700** of processing communication requests in a virtual session is illustrated. This method is preferably practiced by the virtual session server **215** simultaneously with the method **600**. In a first step **705** a virtual session is established between the virtual session server **215** and the remote unit **100** as discussed in connection with FIGS. **5** and **6**. At some later time, while the virtual session is active, the communication server **212** receives an incoming communication request for the remote unit **100**. Because the virtual session server **215** practices the method **500** and/or the method **600**, depending on the time of arrival of the communication, the remote unit **100** may or may not be physically connected to the virtual session server **215** by a physical communication link. Hence when the communication is received, control passes from the step **705** based on a decision **710** which determines whether a physical connection currently exists to the remote unit **100**.

If the virtual session is presently in a state whereby the physical connection has been disconnected, control passes from the first step **705** to a second step **715**. In the second step **715** the communication request is accepted by the communication server **212** through a direct connection or via the communication interface **210**. Control next passes to a third and optional step **720** whereby a specific caller identification packet is associated with the communication type. For example, if the communication involves a telephone call a first caller identification packet is sent identifying an extension used for telephone calls. If the communication involves an email, a second caller identification packet is sent identifying an extension used for email. On the other hand, if the communication comes from the application program **220**, still another caller identification packet is sent. When this optional use of a caller identification packet is employed, the remote unit **100** has the information needed to properly and immediately respond to an incoming call as discussed in connection with FIG. **8**. If a call is received by the remote unit from a source other than the virtual connection server **215**, the caller identification information will identify the call as not being associated with the virtual session. Control next passes to a fourth step **725** whereby an automatic dialer responds to communication requests and the communication is forwarded to the remote unit **100**.

Another situation arises when the communication request arrives while the remote unit **100** and the virtual session server **215** are currently connected by an existing physical channel. According to one mode of processing, control stays in the first step **705** while communication proceeds in an active phase of a virtual session. When the communication request arrives, the communication server **212** signals to the virtual session server that a new call has arrived for the remote unit **100**. The virtual session server then causes the existing physical layer communication connection to be dropped and thus control passes from the first step **705** to the second step **715** as in the foregoing discussion. In another mode of processing, the existing physical layer communication connection is left in tact and control passes from the first step **705** to a fifth step **745**. In the fifth step **745**, the communication is packetized, and in a sixth step **750** the communication packets are passed along the existing physical layer communication connection. Control continues to loop back from the sixth step **750** to the fifth step **745** during the course of the communication. In this mode of operation the existing physical layer communication connection is shared to provide the remote user with a means to stay connected to the application program **220** and communicate at the same time. In this case the physical layer is time shared by the virtual session to allow multiple modes of communications to proceed in parallel.

Note the method **700** provides the user of the remote unit **100** with a virtual presence in the work place while actually being remote. Independent of whether the remote unit is presently actively connected to the virtual session server, the communication request may be forwarded to the remote unit **100** making the remote user appear to be present in the office at all times. The only time the remote unit **100** would not be reachable is when it is engaged in a communication with an entity other than the virtual connection server. This problem may be mitigated by allowing the remote unit to only be reachable through the access number provided by the communication server **212**. If a call is placed from the remote unit to another point, this call too may be routed through the communication server **212**. Systems which do allow the remote unit to make calls outside the virtual session may preferably employ voice mail at the communication server **212**. When the remote unit again becomes available, the virtual session server **215** may forward the communication to the remote unit according to the method **700**. Remote units may also be designed using call-waiting concepts whereby the virtual session may be re-activated by interrupting another call.

Referring now to FIG. **8**, an optional method **800** practiced by the remote unit **100** is illustrated. This method is practiced when a virtual session exists, the remote unit and the virtual session server are presently not connected via a physical channel, and a communication request is to be forwarded to the remote unit **100**. In a first step a first ring signal is detected. Optionally, the first ring signal is suppressed so the user will not hear it. In some systems a vibrational first ring signal may be allowed to pass through to notify the user of an incoming communication. In still other embodiments, the remote unit may be programmed to sound a normal ring on the first ring signal. In some embodiments the ring signal will not be a traditional telephone ring signal but will in general be any signal indicative of an incoming communication request.

In current systems, a caller identification packet is presented to the called party after the first ring signal. Hence identification of the calling party becomes available at this time. After the first ring signal, control passes from the first step **805** to a second step **810**. In the second step **810**, the caller identification information is evaluated. Note it would be preferable to accept the caller identification data before the first ring, and the present invention contemplates systems whereby the virtual session server **215** signals to the PSTN to provide a caller identification packet before the first ring. This service does not appear to be available from telephone service providers at this time. Embodiments also comprehended by the present invention include systems whereby the remote unit **100** immediately picks up an incoming call and caller identification information is sent by the virtual connection server **215** over the connection identifying the call-type. During the second step **810**, the caller identification packet is evaluated. As discussed in connection with the third step **720** of the method **700**, the virtual session server **215** sends out a caller identification packet to identify the type of incoming call. For example, different caller identification packets indicate whether the incoming call is an email, a voice telephone call, or a communication from the application program **220**.

Control next passes from the second step **810** to a third step **815**. In the third step **815**, an application layer program is selected to process the incoming call. In the foregoing examples, an application may be launched to accept an email message, a voice telephone call, or to accept a communication from the application program **220**. If the communication is an email, it may be desirable to pop a mailbox icon on the screen or to produce a speech signal stating "you've got mail." If the incoming call is a voice call, it may be desirable to allow the telephone to ring like a normal telephone. If the communication is from the application program **220**, it may be desirable to update data located in the screens of the workflow or otherwise signal the presence of new data. Once the appropriate application has been launched to handle the incoming communication, control next passes to a fourth step **820** whereby communication session is reactivated and the communication is processed. For example, one or more packets of information may be received and related information such as an email message may be displayed. Also, a telephone call may be allowed to proceed or a set of information may be downloaded from the application program **220**.

Although the present invention has been described with reference to specific embodiments, other embodiments may occur to those skilled in the art without deviating from the intended scope. For example, other forms of communications such as fax messages may be accepted and displayed by the remote unit **100**. Also, the present invention may be used for applications other than mobile workers and Internet users. Virtual sessions according to the present invention are applicable to any situation where a continual connectivity is required but the cost to remain continuously connected is high. Vehicle computers with cellular radio based Internet connections are an example. In such systems, the remote unit **100** may be a vehicle-mounted computer or may include a connection to a vehicle-mounted computer. Also, virtual sessions according to the present invention are applicable to any situation where the user should not be burdened with the need to upload and/or download data and go through connection and disconnection procedures. Therefore, it is to be understood that the invention herein encompasses all such embodiments which do not depart from the spirit and scope of the invention as defined in the appended claims.

In an embodiment, a remote unit is provided, the remote unit comprising a central processing unit operatively coupled to a bus; a memory operatively coupled to the bus; a display monitor operatively coupled to the bus; a communications module operatively coupled to the bus; a control program module controllably coupled to the central processing unit and operative to cause the central processing unit to display a user interface to a user via the display monitor, the control program further being operative to cause the processor to establish a first virtual session with a first remote entity using a first physical communication connection is provided.

In another embodiment, the first remote entity is a communication interface coupled to a virtual session server, the virtual session server providing access to an application program via the virtual session. The remote unit may further comprise a table structure, the table structure comprising a plurality of data structures, the plurality of data structures comprising a first data structure and a second data structure, the first data structure comprising information relating to the first virtual session, and the second data structure comprising information relating to a second virtual session with a second remote entity; wherein the control program module is operative to selectively reactivate one of the first virtual session and the second virtual session in response to a communication request from the user. Alternatively, the remote unit may further comprise a table structure, the table structure comprising a plurality of data structures, the plurality of data structures comprising a first data structure and a second data structure, the first data structure comprising information relating to the first virtual session, and the second data structure comprising information relating to a second virtual session with a second remote entity; wherein the control program module is operative to selectively reactivate one of the first virtual session and the second virtual session in response to a communication request received via the communications module.

In another embodiment, the first remote entity is a communication interface coupled to a virtual session server, and the remote unit maintains a virtual presence with the virtual session server by communicating using any of a plurality of communication types via the virtual session. The plurality of communication types may include voice telephony and email.

In yet another embodiment, the control program is further operative to disconnect from the first physical communication connection, present a workflow to the user and predict, based upon the workflow, when external communication will be required, the control program further being operative to cause a second physical communication connection to be established based upon the prediction, the second physical communication connection providing a physical layer connection for the virtual session. The control program may be further operative to predict, based upon the workflow, when information is to be uploaded and/or downloaded to and/or from the first remote entity, and to cause the information to be uploaded and/or downloaded in the background using the second physical communication connection. The second physical communication connection may be established transparently without inhibiting the user interface from supporting the workflow. The user interface may implement a workflow by presenting a sequence of interactive screens on the display monitor.

In yet another embodiment, the control program is further operative to drop the first physical connection, cause a second physical communication path to be established, and communicate application layer data via the virtual session using the second physical connection before a line-rate negotiation associated with the second physical connection has completed.

In another embodiment of the present invention, a virtual session server is provided, the virtual session server comprising a first coupling to a communication interface the first coupling providing access to a physical layer connection to a remote unit; a table structure, the table structure operatively coupled to receive information from the first coupling, the table structure storing parameters relating a communication session; a second coupling to an application program, the second coupling comprising a communication path whose access requires a user authentication parameter; and a control program module controllably coupled to the first coupling, the table structure, and the second coupling, the control program module operative to establish an application session for use between the remote unit and the application program, store a parameter relating to the application session in the table structure, and maintain the application session in both the presence and absence of the physical layer communication connection, such that the application session is continuously activated both when the physical layer communication path is in a connected state and a disconnected state.

In another embodiment, the control program module is further operative to accept from the remote unit a user authentication parameter when the physical layer connection transitions from the disconnected state to the connected state, and provide a communication path for whereby the remote unit is able to access the application program using the application session.

In another embodiment, the virtual session server further comprises a third coupling to a communication server, whereby the control program module is further operative to accept a communication request from the communication server to be forwarded to the remote unit, and if a physical layer communication connection is present, the control program further operative to activate a second application session, the second application session between the remote unit and a communication program responsible for the communication request. The control program may issue a signal indicating to convert a media stream related to the communication request into a packet stream to be routed via the second application session. Upon reception of the communication request, if no physical communication connection is presently active, the control program may be further operative to process the communication request by dialing out to establish an active physical connection and to then activate a second application session, the second application session between the remote unit and a communication program responsible for the communication request.

In another embodiment, activation of the physical layer communication connection, application layer data is transmitted via the physical layer communication connection prior to the completion of a line-rate negotiation associated with the physical layer communication connection.

In another embodiment, different connections of the physical layer communication connection use distinct communications media.

In another embodiment of the present invention, a client-server system is provided, the client-server system comprising a remote unit comprising a control program module, the remote unit operative to maintain a virtual session; a virtual session server, the virtual session server operatively coupled to a communication interface via a first coupling, the communication interface coupled to the remote unit by at least one communication link, the communication interface providing a data stream including information received from the communication link, the virtual session server operatively coupled to an application program via a second coupling; a table structure, the table structure including a link to a data structure, the data structure including a plurality of bits arranged to represent information relating to an communication session between the remote unit and the virtual session server; and a control program module controllably coupled to the virtual session server, the table structure and the application program, the control program module operative to cause the virtual session server to establish an application session for use between the virtual session server and the application program, store an application session parameter in the data structure and to link the data structure into the table structure, and maintain the application session in both the presence and absence of the physical layer communication connection, such that the application session is continuously activated when the physical layer communication path is in a connected state and a disconnected state.

In another embodiment, the control program module is further operative to cause the virtual session server to execute a client-side application software module on the virtual server under remote control of the remote unit.

In another embodiment, the control program module is further operative to cause the virtual session server to execute a client-side application software module on the virtual session server in parallel with the remote unit; and maintain a presence with the application program while the physical layer communication path is in the disconnected state.

In another embodiment, the control program module is further operative to cause the virtual session server to maintain a communication path and grant the remote unit access to the application program via the communication path after the physical layer has been placed into the connected state.

In another embodiment, the control program module is further operative to cause the virtual session server to grant the remote unit access to the application program via the communication path only after a virtual session authentication procedure has been successfully completed.

In another embodiment of the present invention, a method of accessing a central server from a remote unit is provided, the method comprising the steps of presenting a workflow to a user via a user interface; predicting, based upon the workflow, when the user will require connectivity to the central server; and based upon the prediction and in the background, initiating the establishment of a physical layer communication connection to the central server.

In another embodiment, the physical layer connection is established to support a virtual session and the remote entity comprises a virtual session server.

In another embodiment, the method further comprises the steps of making a second prediction, based upon the workflow, the second prediction indicating when information is to be uploaded and/or downloaded to and/or from the central server; and based upon the second prediction, causing the information to be uploaded and/or downloaded in the background using the physical layer communication connection.

In another embodiment of the present invention, a method of establishing a connection with a low connection set-up time is provided, the method comprising the steps of initiating the establishment of a communication connection to be used to communicate with a remote entity; communicating application layer data via the communication connection prior to the completion of a line-rate negotiation; and in the background, negotiating a line speed for subsequent high-speed communication.

In another embodiment, the method further comprises the steps of receiving an input, the input indicative of a request to communicate with a remote entity; and transacting an authorization sequence with the remote entity.

In another embodiment, the input may be received by a user interface and the input may be indicative of a command to, access a network site using a hyperlink. Furthermore, the method may comprise the step of resuming the use of a virtual session via the communication connection.

In another embodiment, the step of initiating may be performed in response to a ring signal received from the remote entity.

In another embodiment of the present invention, a method is provided, the method comprising the steps of establishing a first connection to a remote entity; using the first connection to establish a set of parameters needed to define a communication session with the remote entity; disconnecting the first connection and maintaining the parameters related to the communication session; establishing a second connection to the remote entity; communicating an authorization sequence with the remote entity; and reactivating the communication session using as a communication path the second connection.

In another embodiment, the step of using further comprises the steps of accepting a user authentication parameter and passing the parameter to an application program to establish an application session; and linking into a table structure a data structure associating the remote unit with the application session, the data structure thereby providing an access by the remote unit to the application program upon reactivation of the communication session.

In another embodiment, the first and second connections involve distinct communications media. The set of parameters may be stored in a table, and the table may hold a plurality of such sets of parameters and maintains a plurality of communication sessions with a plurality of remote entities. The method may further comprise the steps of converting a communications media stream to a packet stream and multiplexing the packet stream via the application session over the communication session. Alternatively, the method may further comprise the steps of disconnecting from the second connection while maintaining the communication session; initiating the establishment of a third connection; and processing the communication request by forwarding a communications media stream via the communication session using the third communication connection.

In another embodiment, the method further comprises the steps of evaluating a header received over the second physical layer communication connection to determine an application program to process information transacted on the second connection; and based upon the evaluation, launching the application program to communicate via the second connection.

In another embodiment, a method of providing a virtual presence of a remote unit with a central server is provided, the method comprising the steps of establishing a communication session with the remote unit using a first physical layer communication path and dropping the first physical layer communication connection; configuring a data structure, the data structure linking the communication session to an application session; accepting a communication request from the application program, the communication request requiring a physical layer communication path to the remote unit; dialing-out to establish a second physical layer communication path to reactivate the communication session with the remote unit; delivering a packet via the physical layer communication path, the packet containing information indicative of the type of the communication request; and reactivating the communication session with the remote unit and using the communication session to support a connection stream between the application program and the remote unit.

In another embodiment, the data structure links a plurality of application sessions to the communication session, the application sessions including a first an application program which provides a communication service.

In another embodiment, the communication service comprises a voice telephony connection.

In another embodiment, the plurality of application sessions includes a second an application program which provides access to a database.

In another embodiment, the database comprises a medical record.

In another embodiment, a system for maintaining a virtual session is provided, the system comprising a virtual session server, the virtual session server being operatively coupled to an application program, the virtual session server maintaining an application session with the application program on behalf of a remote unit, the virtual session server maintaining a virtual communication session with the remote unit; and a table structure comprising a plurality of memory locations arranged within a storage unit, the table structure operative to maintain set of links correlating the virtual communication session and the application session.

In another embodiment, the system further comprises the remote unit, whereby the remote unit comprises a protocol stack used to maintain a communication session in the presence and absence of a physical layer communication path.

In another embodiment, the system further comprises the application program.

In another embodiment, the virtual session server is operative to execute a client-side application software module under remote control of the remote unit.

In another embodiment, the virtual session server is operative to execute a client-side application software module on the virtual session server in parallel with the remote unit; and maintain a presence with the application program while the physical layer communication path is in the disconnected state.

In another embodiment, the control program module is further operative to cause the virtual session server to maintain a communication path and grant the remote unit access to the application program via the communication path after the physical layer has been placed into the connected state.

In another embodiment, the virtual session server maintains a proxy-presence with the application program to prevent the application program from terminating in the absence of a physical layer communication path to the remote unit.

In another embodiment, the virtual session server accesses the table structure to provide access to the application program by the remote unit via the application session upon the reactivation of the virtual communication session.

What is claimed is:

1. A method, comprising:

establishing, at a computing device, a communication session supporting communication between a first program executing at an application layer of the computing device and a remote server;

subsequent to deactivation of the established communication session, the computing device receiving an incoming communication from the remote server, wherein the incoming communication is not in response to a request sent by the computing device;

at the application layer, the computing device reading a set of information included in the incoming communication;

in response to determining that the set of information read at the application layer includes information identifying the first program executing at the computing device, the computing device reactivating the communication session between the first program and the remote server.

2. The method of claim **1**, wherein the computing device is a mobile handset.

3. The method of claim **1**, wherein the communication session is established over one of: a public switched telephone network (PSTN); a cellular network, a PCS network.

4. The method of claim **1**, wherein the communication session is established over an Internet connection.

5. The method of claim **1**, wherein the incoming communication corresponds to a web page.

6. The method of claim **1**, further comprising:

deactivating the communication session in response to a prediction generated by the computing device.

7. The method of claim **1**, wherein the establishing the communication session includes establishing a first physical layer connection between the computing device and the remote server.

8. A mobile handset, comprising:

a wireless communication interface configured to communicate over a wireless network with a remote entity;

a processor unit;

memory storing program instructions executable by the mobile handset, using the processor unit, to cause the mobile handset to:

establish a communication session supporting communication between the remote entity and one or more programs executing on the mobile handset;

inactivate the established communication session;

while the communication session is inactive, receive a set of information encoded on a wireless signal received from the remote entity via the wireless communication interface, wherein the set of information was not sent in response to a request from the mobile handset;

at an application layer, read identifying information within the received set of information;

in response to determining that the identifying information read at the application layer identifies a first program of the one or more programs executing on the mobile handset, reactivate the communication session between the remote entity and the one or more programs.

9. The mobile handset of claim **8**, wherein the first program is a messaging program.

10. The mobile handset of claim **8**, wherein the first program is a browser program.

11. The mobile handset of claim **10**, wherein the received set of information corresponds to a requested web page.

12. The mobile handset of claim **8**, wherein the communication session is established over a physical layer connection, wherein the physical layer connection is disconnected when the communication session is inactive, and wherein the communication session is reactivated by reestablishing the physical layer connection.

13. An apparatus, comprising:

an external communication interface configured to couple to one or more networks external to the apparatus;

memory storing program instructions that, in response to execution by the apparatus, cause the apparatus to perform operations comprising:

establishing a virtual session with a remote entity;

inactivating the established virtual session;

receiving a set of information from the remote entity while the virtual session is inactive, wherein the set of information was not sent in response to a request from the apparatus; and

reactivating the virtual session in response to determining, at an application layer, that identifying information in the received set of information identifies a first application layer program that is stored on the apparatus.

14. The apparatus of claim **13**, wherein the first application-layer program is a messaging program.

15. A mobile handset, comprising:

a wireless communication interface configured to communicate via one or more wireless networks;

a processor unit;

memory storing program instructions executable by the mobile handset, using the processor unit, to cause the mobile handset to:

establish a communication session with a remote entity via the wireless communication interface;

inactivate the established communication session;

while the communication session is inactive, receive a first communication from the remote entity, wherein the first communication was not initiated by the remote entity in response to a previous communication from the mobile handset;

reactivate the communication session in response to determining that the received first communication includes information identifying an application at the mobile handset.

16. The mobile handset of claim **15**, wherein the communication session is inactive while a wireless physical layer connection between the remote entity and the mobile handset is connected.

17. A method performed by a computing device, the method comprising:

during a communication session that has previously been established with a remote entity and that is currently inactive, receiving a first communication initiated by the remote entity, wherein the first communication has not been initiated by the remote entity in response to a previous communication from the computing device;

in response to determining that the received first communication includes information identifying an application at the computing device, reactivating the communication session with the remote entity.

18. The method of claim **17**, wherein the computing device is a mobile handset.

19. The method of claim **17**, wherein the first communication is a wireless communication, and wherein the communication session is a virtual session.

20. An apparatus, comprising:

a wireless communication interface configured to couple the apparatus to a remote entity;

one or more processors;

memory storing program instructions executable by the apparatus, using the one or more processors, to cause the apparatus to:

establish a virtual session with the remote entity;

subsequent to inactivation of the established virtual session, receive a wireless communication from the remote entity that is not responsive to a previous communication from the apparatus;

reactivate the virtual session in response to determining that the received wireless communication includes information identifying an application at the apparatus.

21. A system, comprising:

a mobile phone storing a first program executable at an application layer of the mobile phone; and

a server configured to:

maintain a communication session between the server and the mobile phone, the communication session supporting communication with a first program stored on the mobile phone; and

subsequent to deactivation of the communication session, send a wireless signal to the mobile phone, wherein the wireless signal includes a set of information identifying the first program, wherein sending the wireless signal to the mobile phone is not in response to request from the mobile phone;

wherein the mobile phone is configured to:

receive the wireless signal sent by the server subsequent to the deactivation of the communication session;

at the application layer of the mobile phone, read the set of information included in the wireless signal; and

in response to determining that the set of information identifies the first program, reactivate the communication session.

22. The method of claim **7**,

wherein the deactivation of the established communication session includes disconnecting the first physical layer connection; and

wherein the reactivating the communication session includes establishing a second physical layer connection between the computing device and the remote server.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO.	: 8,291,010 B2
APPLICATION NO.	: 12/194311
DATED	: October 16, 2012
INVENTOR(S)	: Dowling et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

On Title Page 2, in item (56), under "OTHER PUBLICATIONS", in Column 2, Line 24, delete "Conextant's" and insert -- Conexant's --, therefor.

In the Specification:

In Column 1, Line 8, delete "2004" and insert -- 2004, --, therefor.

In Column 1, Line 10, delete "10/335,821" and insert -- 10/335,821, --, therefor.

In Column 5, Line 23, delete "Pentium™." and insert -- Pentium™ --, therefor.

In Column 5, Line 59, delete "central" and insert -- control --, therefor.

In Column 11, Line 60, delete "UltaSparc™." and insert -- UltraSparc™ --, therefor.

In Column 11, Line 63, delete "Solaris™." and insert -- Solaris™ --, therefor.

In Column 11, Line 63, delete "WindowsNT™." and insert -- WindowsNT™ --, therefor.

In Column 11, Line 67, delete "UltraSparc™." and insert -- UltraSparc™ --, therefor.

In Column 14, Line 5, delete "PCAnywhere™." and insert -- PCAnywhere™ --, therefor.

In Column 14, Line 18, delete "application" and insert -- application program --, therefor.

In Column 14, Line 38, delete "software" and insert -- software module --, therefor.

In Column 16, Line 33, delete "220" and insert -- 215 --, therefor.

Signed and Sealed this
Twenty-sixth Day of November, 2013

Margaret A. Focarino

Margaret A. Focarino
Commissioner for Patents of the United States Patent and Trademark Office

CERTIFICATE OF CORRECTION (continued)

U.S. Pat. No. 8,291,010 B2

In Column 21, Line 15, delete “210” and insert -- 215 --, therefor.

In Column 21, Line 48, delete “application” and insert -- application program --, therefor.

EXHIBIT 4

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 6,574,239 – Defendant / CKE Restaurants Holdings, Inc.
Claim 7

Communication Interface Technologies LLC ("CIT") provides evidence of infringement of claim 7 of U.S. Patent No. 6,574,239 (hereinafter "the '239 patent") by CKE Restaurants Holdings, Inc. ("CKE"). In support thereof, CIT provides the following claim charts.

"Accused Instrumentalities" as used herein refers to at least CKE's application ("App") in systems and methods, including hardware and software products including, but not limited to the CKE App as developed for mobile electronic devices (*see, e.g.,* https://play.google.com/store/apps/details?id=com.carlsjr.ordering&hl=en_US&gl=US), along with any hardware and/or software for provisioning the App. Upon information and belief, the exemplary version herein and previous versions of the Accused Instrumentalities distributed prior to expiration of the patents in suit operated materially in the same manner.

These claim charts demonstrate CKE's infringement, and provide notice of such infringement, by comparing each element of the asserted claim to corresponding components, aspects, and/or features of the Accused Instrumentalities. These claim charts are not intended to constitute an expert report on infringement. These claim charts include information provided by way of example, and not by way of limitation.

The analysis set forth below is based only upon information from publicly available resources regarding the Accused Instrumentalities, as CKE has not yet provided any non-public information. An analysis of CKE's (or a third parties') technical documentation and/or software source code may assist in fully identifying all infringing features and functionality. Accordingly, CIT reserves the right to supplement this infringement analysis once such information is made available to CIT. Furthermore, CIT reserves the right to revise this infringement analysis, as appropriate, upon issuance of a court order construing any terms recited in the asserted claims.

CIT provides this evidence of infringement and related analysis without the benefit of claim construction or expert reports or discovery. CIT reserves the right to supplement, amend or otherwise modify this analysis and/or evidence based on any such claim construction or expert reports or discovery.

Unless otherwise noted, CIT contends that CKE has directly infringed the '239 patent in violation of 35 U.S.C. § 271(a) by selling, offering to sell, making, using, and/or importing the Accused Instrumentalities. The following exemplary analysis demonstrates that infringement.

Communication Interface Technologies LLC's First Infringement Analysis

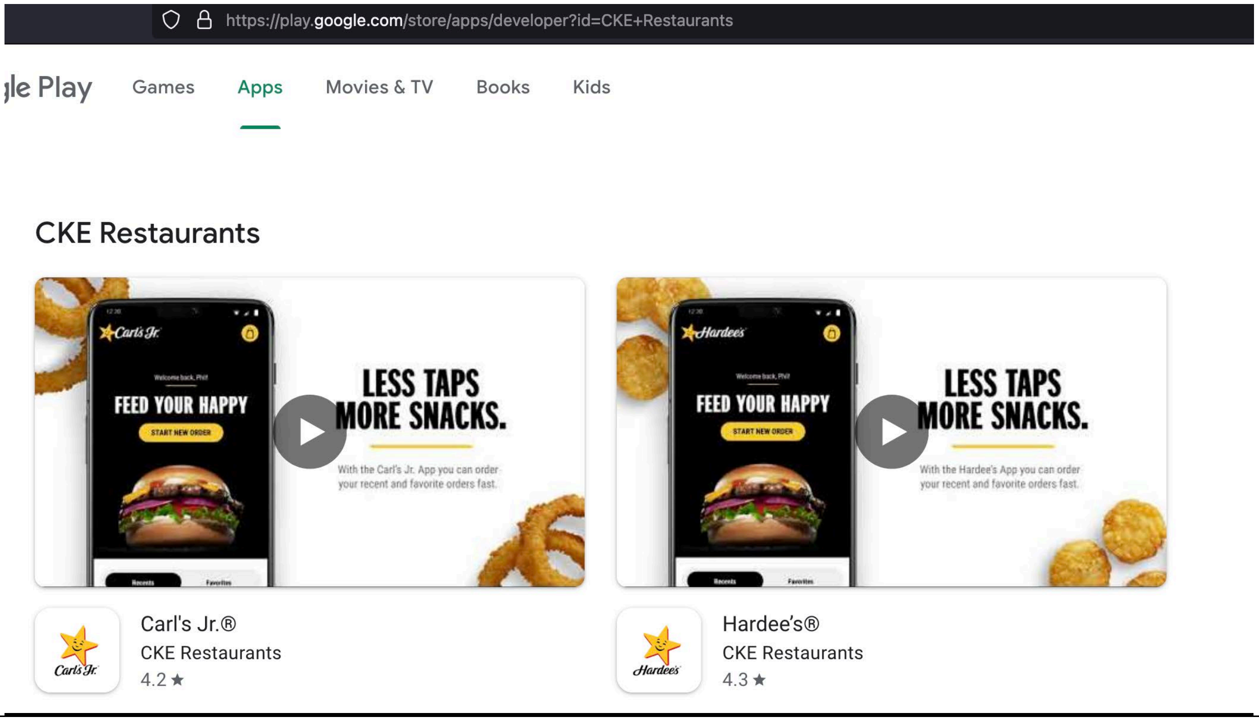
U.S. Patent No. 6,574,239 – Defendant / CKE Restaurants Holdings, Inc.
Claim 7

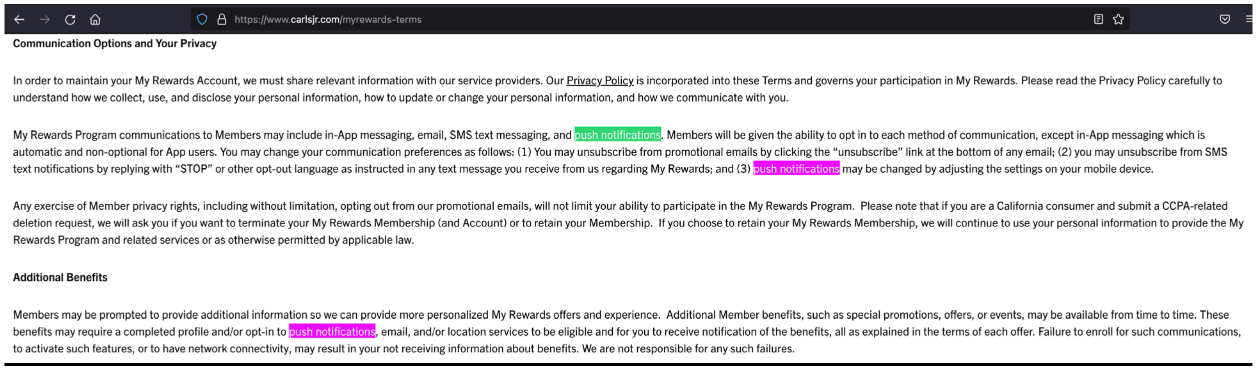
Unless otherwise noted, CIT believes and contends that each element of each claim asserted herein is literally met through CKE's provision of the Accused Instrumentalities. However, to the extent that CKE attempts to allege that any asserted claim element is not literally met, CIT believes and contends that such elements are met under the doctrine of equivalents. More specifically, in its investigation and analysis of the Accused Instrumentalities, CIT did not identify any substantial differences between the elements of the patent claims and the corresponding features of the Accused Instrumentalities, as set forth herein. In each instance, the identified feature of the Accused Instrumentalities performs at least substantially the same function in substantially the same way to achieve substantially the same result as the corresponding claim element.

To the extent the chart of an asserted claim relies on evidence about certain specifically identified Accused Instrumentalities, CIT asserts that, on information and belief, any similarly functioning instrumentalities also infringes the charted claim. CIT reserves the right to amend this infringement analysis based on other products made, used, sold, imported, or offered for sale by CKE. CIT also reserves the right to amend this infringement analysis by citing other claims of the '239 patent, not listed in the claim chart, that are infringed by the Accused Instrumentalities. CIT further reserves the right to amend this infringement analysis by adding, subtracting, or otherwise modifying content in the "Accused Instrumentalities" column of each chart.

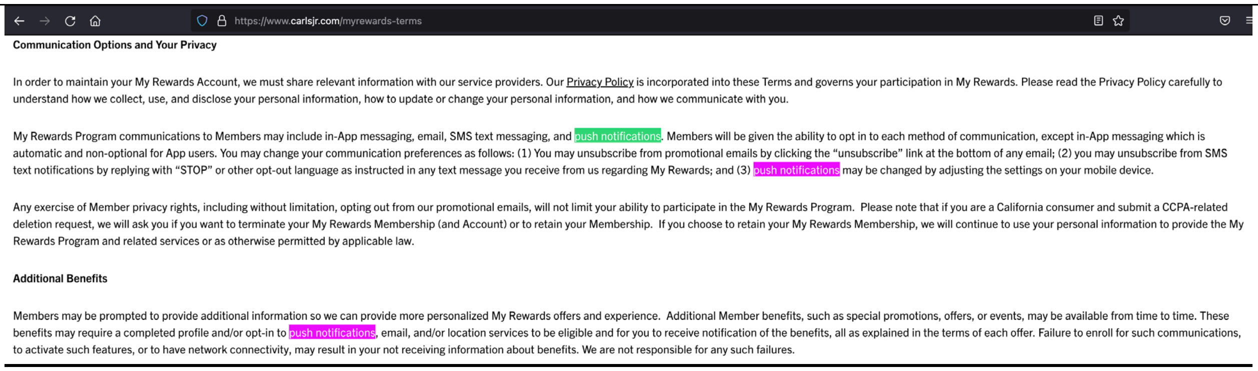
COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 6,574,239 – Defendant / CKE Restaurants Holdings, Inc.
Claim 7

	Claim 7	**Carl's Jr. Service**  https://play.google.com/store/apps/developer?id=CKE+Restaurants gle Play Games Apps Movies & TV Books Kids CKE Restaurants Carl's Jr.® CKE Restaurants 4.2 ★ Hardee's® CKE Restaurants 4.3 ★
7	For use in controlling a virtual session on a server, a method comprising:	A method is specified for controlling a virtual session on a server. The definition of a virtual session is described section 7a below.
7a	establishing a **virtual session** with a remote unit, the virtual session being instantiated to support at least one application layer program;	Wireless push notification messages are sent over Transport Layer Security (TLS) sessions. Each Push message includes an encrypted push token as per **Endnote #1**. The push token is sent in a Push Notification message over TLS sessions from the **Carl's Jr.** Server backend to the **Carl's Jr.** App (application program, application layer program) running on a user's smartphone or tablet (remote unit).

		In the Carl's Jr. application, for example, a push notification contains information related to products and/or services.  https://www.carlsjr.com/myrewards-terms **Communication Options and Your Privacy** In order to maintain your My Rewards Account, we must share relevant information with our service providers. Our Privacy Policy is incorporated into these Terms and governs your participation in My Rewards. Please read the Privacy Policy carefully to understand how we collect, use, and disclose your personal information, how to update or change your personal information, and how we communicate with you. My Rewards Program communications to Members may include in-App messaging, email, SMS text messaging, and push notifications. Members will be given the ability to opt in to each method of communication, except in-App messaging which is automatic and non-optional for App users. You may change your communication preferences as follows: (1) You may unsubscribe from promotional emails by clicking the "unsubscribe" link at the bottom of any email; (2) you may unsubscribe from SMS text notifications by replying with "STOP" or other opt-out language as instructed in any text message you receive from us regarding My Rewards; and (3) push notifications may be changed by adjusting the settings on your mobile device. Any exercise of Member privacy rights, including without limitation, opting out from our promotional emails, will not limit your ability to participate in the My Rewards Program. Please note that if you are a California consumer and submit a CCPA-related deletion request, we will ask you if you want to terminate your My Rewards Membership (and Account) or to retain your Membership. If you choose to retain your My Rewards Membership, we will continue to use your personal information to provide the My Rewards Program and related services or as otherwise permitted by applicable law. **Additional Benefits** Members may be prompted to provide additional information so we can provide more personalized My Rewards offers and experience. Additional Member benefits, such as special promotions, offers, or events, may be available from time to time. These benefits may require a completed profile and/or opt-in to push notifications, email, and/or location services to be eligible and for you to receive notification of the benefits, all as explained in the terms of each offer. Failure to enroll for such communications, to activate such features, or to have network connectivity, may result in your not receiving information about benefits. We are not responsible for any such failures. Also, the Server Application and the client-side App establish a separate TLS connection for traditional client-server communications. For example the **Carl's Jr.** Application Server program establishes a TLS session with the **Carl's Jr.** App. The TLS session used for client-server communications between the **Carl's Jr.** Application Server and the **Carl's Jr.** App correspond to the recited **virtual session**. TLS session use a full handshake sequence that is used to establish connection parameters, and an abbreviated handshake sequence that is used to resume the TLS session from an inactive or dormant state to an active state whereby new payload data can be sent via the virtual session once again. See **Endnote#2** for a discussion of the virtual session aspects of TLS. Mobile applications communicate with their application server via TLS connections. These TLS connections are established at the time the app is installed or launched and can be resumed at a later time

		using a session token. See **Endnote#2** for a discussion of TLS. https://threema.ch/press-files/cryptography_whitepaper.pdf - Android uses TLS 1.2 resumable sessions. https://www.icir.org/johanna/papers/conext17android.pdf - Android supports TLS and secure connections between client app and server are ubiquitously used. https://developer.android.com/training/articles/security-ssl. – "The Secure Sockets Layer (SSL)—now technically known as Transport Layer Security (TLS)—is a common building block for encrypted communications between clients and servers." - Note that certain certificates are used, and these are used to help create Android Device Tokens used in Push Notification messages. **See Endnotes #1, and #2**.
7b	placing the virtual session in an inactive state;	See **Endnote #2**. Note when the application data phase is finished, the TLS client-server data session is placed back into the inactive state. Hence the end of application data marker is the signal used to place the virtual session into the inactive state.
7c	sending a signal indicative of an **incoming communication request** and an **application-program identifying packet** to said remote unit, said application-program identifying packet identifying an application program that needs to resume a virtual session and communicate with said remote unit; and	**Carl's Jr.** Application server causes a push notification message (incoming communication) to be sent to the **Carl's Jr.** App running on the user's smartphone or tablet device (remote unit). **See Endnote #1**. In the Carl's Jr. application, for example, a push notification contains information related to products and/or services.

		 https://www.carlsjr.com/myrewards-terms **Communication Options and Your Privacy** In order to maintain your My Rewards Account, we must share relevant information with our service providers. Our Privacy Policy is incorporated into these Terms and governs your participation in My Rewards. Please read the Privacy Policy carefully to understand how we collect, use, and disclose your personal information, how to update or change your personal information, and how we communicate with you. My Rewards Program communications to Members may include in-App messaging, email, SMS text messaging, and push notifications. Members will be given the ability to opt in to each method of communication, except in-App messaging which is automatic and non-optional for App users. You may change your communication preferences as follows: (1) You may unsubscribe from promotional emails by clicking the "unsubscribe" link at the bottom of any email; (2) you may unsubscribe from SMS text notifications by replying with "STOP" or other opt-out language as instructed in any text message you receive from us regarding My Rewards; and (3) push notifications may be changed by adjusting the settings on your mobile device. Any exercise of Member privacy rights, including without limitation, opting out from our promotional emails, will not limit your ability to participate in the My Rewards Program. Please note that if you are a California consumer and submit a CCPA-related deletion request, we will ask you if you want to terminate your My Rewards Membership (and Account) or to retain your Membership. If you choose to retain your My Rewards Membership, we will continue to use your personal information to provide the My Rewards Program and related services or as otherwise permitted by applicable law. **Additional Benefits** Members may be prompted to provide additional information so we can provide more personalized My Rewards offers and experience. Additional Member benefits, such as special promotions, offers, or events, may be available from time to time. These benefits may require a completed profile and/or opt-in to push notifications, email, and/or location services to be eligible and for you to receive notification of the benefits, all as explained in the terms of each offer. Failure to enroll for such communications, to activate such features, or to have network connectivity, may result in your not receiving information about benefits. We are not responsible for any such failures. The signal indicative of an incoming communication request will request the application program running on the remote unit to resume the TLS session used for client-server communications between the **Carl's Jr.** Application server and the **Carl's Jr.** App running on the user's smartphone or tablet device (remote unit). The application-program identifying packet will include information used to identify the App) on the remote unit. The **Carl's Jr.** server and the Carl's Jr. App will resume a TLS session so that the server and the remote unit can resume communications. To do so the **Carl's Jr.** App will invoke a protocol stack within the remote unit to communicate back to the server via the remote unit. See **Endnote #1** for a discussion of how each new set of data payloads coming into the Carl's Jr. application includes an **app-specific device token**. The **app-specific device token** is indicative of the Carl's Jr. application running on the remote unit. Each incoming wireless push notification message contains the **app-specific device token** which is part of the **application-program identifying packet.**
7d	placing the virtual session back into the active state and transferring data between	Based upon user selection of information associated with the above-mentioned push notification, the Transport Layer Security (TLS) session between the **Carl's Jr.** Application Server and the **Carl's Jr.** App is

	the application and the remote unit via the virtual session in response to said step of sending.	resumed as discussed in **Endnote #2**. TLS session resumption means that the TLS session is placed back into the active state using an abbreviated handshake sequence so that new application data can be passed between the **Carl's Jr.** Application Server and the App) running on the user's smartphone or tablet. See **Endnote#2** for a discussion of TLS session resumption. Also see, https://docs.microsoft.com/en-us/windows/desktop/secauthn/tls-handshake-protocol.

Caveat: The notes and/or cited excerpts utilized herein are set forth for illustrative purposes only and are not meant to be limiting in any manner. For example, the notes and/or cited excerpts, may or may not be supplemented or substituted with different excerpt(s) of the relevant reference(s), as appropriate. Further, to the extent any error(s) and/or omission(s) exist herein, all rights are reserved to correct the same.

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 6,574,239 – Defendant / CKE Restaurants Holdings, Inc.
Claim 7

Endnote#1 - App-Specific Device Token

https://help.pushwoosh.com/hc/en-us/articles/360000364923-What-is-a-Device-token-

Question:

What is a Device token?

Answer:

Push token (device token) - is a unique key for the app-device combination which is issued by the Apple or Google push notification gateways. It allows gateways and push notification providers to route messages and ensure the notification is delivered only to the unique app-device combination for which it is intended.

iOS device push tokens are strings with 64 hexadecimal symbols. Push token example:
`03df25c845d460bcdad7802d2vf6fc1dfde97283bf75cc993eb6dca835ea2e2f`
Make sure that iOS push tokens you use when targeting specific devices in your API requests are in lower case.

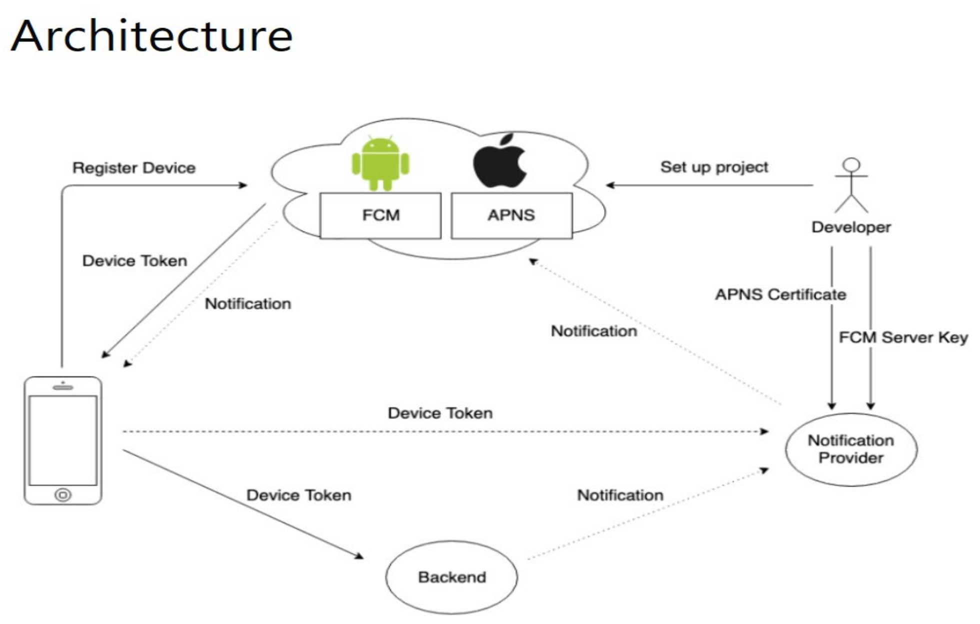
Android device push tokens can differ in length (usually below 255 characters), and usually start with APA…Push token example:

```
APA91bFoi3lMMre9G3XzR1LrF4ZT82_15MsMdEICogXSLB8-
MrdkRuRQFwNI5u8Dh0cI90ABD3BOKnxkEla8cGdisbDHl5cVIkZah5QUhSAxzx4Roa7b4xy9tvx9iNSYw-eXBYYd8k1XKf8Q_Qq1X9-
x-U-Y79vdPq
```

Note: The Android device push tokens correspond to the app-specific device token terminology used in the claim charts.

https://dev.to/jakubkoci/react-native-push-notifications-313i

U.S. Patent No. 6,574,239 – Defendant / CKE Restaurants Holdings, Inc.
Claim 7



Note: In the above Architecture, the "backend" corresponds to the backend of the Application Server. The Device token corresponds to a specific App running on a specific device. That is what is meant by the app-specific device token in the claim charts.

https://firebase.google.com/docs/cloud-messaging/android/first-message

Access the registration token

To send a message to a specific device, you need to know that device's registration token. Because you'll need to enter the token in a field in the Notifications console to complete this tutorial, make sure to copy the token or securely store it after you retrieve it.

On initial startup of your app, the FCM SDK generates a registration token for the client app instance. If you want to target single devices or create device groups, you'll need to access this token by extending FirebaseMessagingService and overriding on NewToken.

This section describes how to retrieve the token and how to monitor changes to the token. Because the token could be rotated after initial startup, you are strongly recommended to retrieve the latest updated registration token.

The registration token may change when:

- The app deletes Instance ID
- The app is restored on a new device
- The user uninstalls/reinstall the app
- The user clears app data."

https://firebase.google.com/docs/cloud-messaging/concept-options

For example, here is a JSON-formatted notification message in an IM app. The user can expect to see a message with the title "Portugal vs. Denmark" and the text "great match!" on the device:

```
{
 "message":{
   "token":"bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1...", ←-- App-specific token
   "notification":{
     "title":"Portugal vs. Denmark",
     "body":"great match!"
   }
 }
}
```

https://firebase.google.com/docs/cloud-messaging/android/client

Retrieve the current registration token

When you need to retrieve the current token, call FirebaseInstanceId.getInstance().getInstanceId():

```
FirebaseInstanceId.getInstance().getInstanceId()
        .addOnCompleteListener(new OnCompleteListener<InstanceIdResult>() {
            @Override
            public void onComplete(@NonNull Task<InstanceIdResult> task) {
                if (!task.isSuccessful()) {
                    Log.w(TAG, "getInstanceId failed", task.getException());
                    return;
                }
                // Get new Instance ID token
                String token = task.getResult().getToken();

                // Log and toast
                String msg = getString(R.string.msg_token_fmt, token);
                Log.d(TAG, msg);
                Toast.makeText(MainActivity.this, msg, Toast.LENGTH_SHORT).show();
            }
        });
```

MainActivity.java

Monitor token generation

The onNewToken callback fires whenever a new token is generated.

```
/**
 * Called if InstanceID token is updated. This may occur if the security of
 * the previous token had been compromised. Note that this is called when the InstanceID token
 * is initially generated so this is where you would retrieve the token.
 */
@Override
public void onNewToken(String token) {
   Log.d(TAG, "Refreshed token: " + token);

   // If you want to send messages to this application instance or
   // manage this apps subscriptions on the server side, send the
   // Instance ID token to your app server.
```

```
    sendRegistrationToServer(token);
}
```

After you've obtained the token, you can send it to your app server and store it using your preferred method. See the Instance ID API reference [https://firebase.google.com/docs/reference/android/com/google/firebase/iid/FirebaseInstanceId] for full detail on the API.

Endnote#2 - Transport Layer Security and Virtual Sessions

Transport Layer Security (TLS) is used by both Apple iOS and Android based devices. The handshake diagrams in this endnote use Apple iOS as an example but apply equally to Android type implementations.

https://developer.android.com/training/articles/security-ssl

"The Secure Sockets Layer (SSL)—now technically known as Transport Layer Security (TLS)—is a common building block for encrypted communications between clients and servers."

https://android-developers.googleblog.com/2018/04/protecting-users-with-tls-by-default-in.html

"Android is committed to keeping users, their devices, and their data safe. One of the ways that we keep data safe is by protecting all data that enters or leaves an Android device with Transport Layer Security (TLS) in transit.

A. Razaghpanah et al., *Studying TLS Usage in Android Apps,* CoNEXT '17, Dec.12-15, 2017, Incheon, Republic of Korea
http://abbas.rpanah.ir/publications/conext2017_tls_paper.pdf

"**A History of TLS Support in Android:** Android has supported TLS 1.0 since its first version released in 2008 and TLS 1.1 and TLS 1.2 since 2012."

"However, other protocols such as secure email (42 apps) and Google's Cloud Messaging service for push notifications (9 apps) [11, 47] also use TLS."

https://developer.ibm.com/customer-engagement/docs/watson-marketing/ibm-engage-2/tls-1-2-migration-for-mobile-push-clients/

What will happen on devices that are unable to support TLS 1.2?

Devices which do not support TLS 1.2 will be unable to connect to our WCA servers. This will prevent users of those devices from:

• Registering new mobile user IDs
• Updating push tokens
• Receiving inbox messages
• Receiving In-app messages

Note: As the above link shows, the creation of the App IDs of Endnote #1 are linked to the TLS protocol being run on the TLS-enabled Push-Notification channel.

https://tools.ietf.org/html/rfc5246

F.1.4. Resuming Sessions

When a connection is established by resuming a session, new ClientHello.random and ServerHello.random values are hashed with the session's master_secret. Provided that the master_secret has not been compromised and that the secure hash operations used to produce the encryption keys and MAC keys are secure, the connection should be secure and effectively independent from previous connections. Attackers cannot use known encryption keys or MAC secrets to compromise the master_secret without breaking the secure hash operations.

Sessions cannot be resumed unless both the client and server agree. If either party suspects that the session may have been compromised, or that certificates may have expired or been revoked, it should force a full handshake. An upper limit of 24 hours is suggested for session ID lifetimes, since an attacker who obtains a master_secret may be able to impersonate the compromised party until the corresponding session ID is retired. Applications that may be run in relatively insecure environments should not write session IDs to stable storage.

https://tools.ietf.org/html/rfc5077

Abstract

This document describes a mechanism that enables the Transport Layer Security (TLS) server to resume sessions and avoid keeping per-client session state. The TLS server encapsulates the session state into a ticket and forwards it to the client. The client can subsequently resume a session using the obtained ticket.

3. Protocol

This specification describes a mechanism to distribute encrypted session-state information in the form of a ticket. The ticket is created by a TLS server and sent to a TLS client. The TLS client presents the ticket to the TLS server to resume a session.

EXHIBIT 5

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,266,296 – Defendant / CKE Restaurants Holdings, Inc.
Claims 1 & 5

Communication Interface Technologies LLC ("CIT") provides evidence of infringement of claim 1 of U.S. Patent No. 8,266,296 (hereinafter "the '296 patent") by CKE Restaurants Holdings, Inc. ("CKE"). In support thereof, CIT provides the following claim charts.

"Accused Instrumentalities" as used herein refers to at least CKE's application ("App") in systems and methods, including hardware and software products including, but not limited to the CKE App as developed for mobile electronic devices (*see, e.g.,* https://play.google.com/store/apps/details?id=com.carlsjr.ordering&hl=en_US&gl=US), along with any hardware and/or software for provisioning the App. Upon information and belief, the exemplary version herein and previous versions of the Accused Instrumentalities distributed prior to expiration of the patents in suit operated materially in the same manner.

These claim charts demonstrate CKE's infringement, and provide notice of such infringement, by comparing each element of the asserted claim to corresponding components, aspects, and/or features of the Accused Instrumentalities. These claim charts are not intended to constitute an expert report on infringement. These claim charts include information provided by way of example, and not by way of limitation.

The analysis set forth below is based only upon information from publicly available resources regarding the Accused Instrumentalities, as CKE has not yet provided any non-public information. An analysis of CKE's (or a third parties') technical documentation and/or software source code may assist in fully identifying all infringing features and functionality. Accordingly, CIT reserves the right to supplement this infringement analysis once such information is made available to CIT. Furthermore, CIT reserves the right to revise this infringement analysis, as appropriate, upon issuance of a court order construing any terms recited in the asserted claims.

CIT provides this evidence of infringement and related analysis without the benefit of claim construction or expert reports or discovery. CIT reserves the right to supplement, amend or otherwise modify this analysis and/or evidence based on any such claim construction or expert reports or discovery.

Unless otherwise noted, CIT contends that CKE has directly infringed the '296 patent in violation of 35 U.S.C. § 271(a) by selling, offering to sell, making, using, and/or importing the Accused Instrumentalities. The following exemplary analysis demonstrates that infringement.

Communication Interface Technologies LLC's First Infringement Analysis

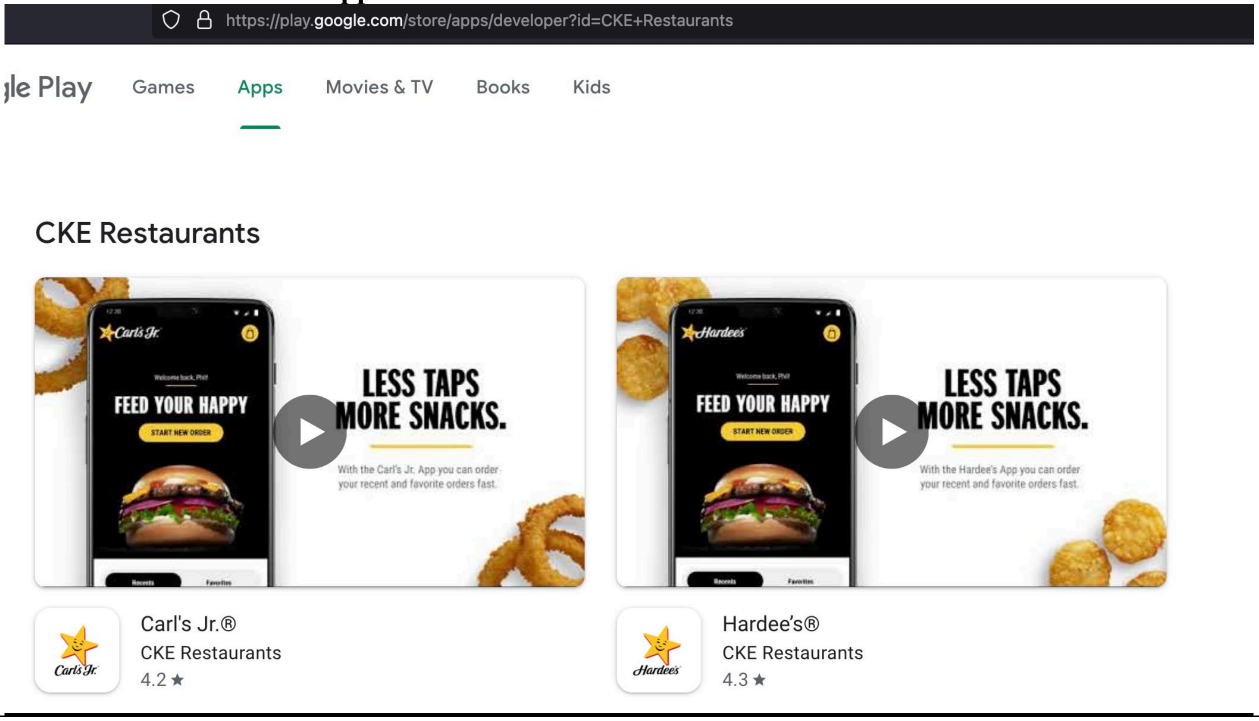
U.S. Patent No. 8,266,296 – Defendant / CKE Restaurants Holdings, Inc.
Claims 1 & 5

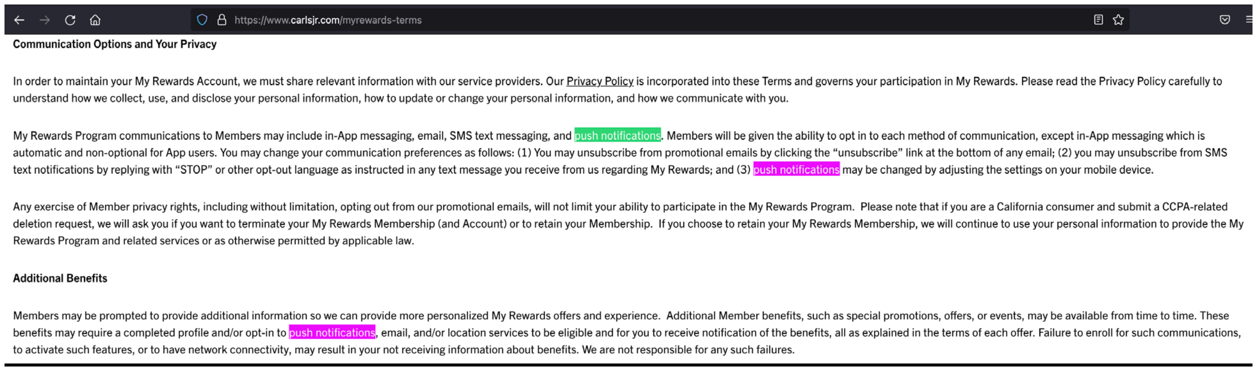
Unless otherwise noted, CIT believes and contends that each element of each claim asserted herein is literally met through CKE's provision of the Accused Instrumentalities. However, to the extent that CKE attempts to allege that any asserted claim element is not literally met, CIT believes and contends that such elements are met under the doctrine of equivalents. More specifically, in its investigation and analysis of the Accused Instrumentalities, CIT did not identify any substantial differences between the elements of the patent claims and the corresponding features of the Accused Instrumentalities, as set forth herein. In each instance, the identified feature of the Accused Instrumentalities performs at least substantially the same function in substantially the same way to achieve substantially the same result as the corresponding claim element.

To the extent the chart of an asserted claim relies on evidence about certain specifically identified Accused Instrumentalities, CIT asserts that, on information and belief, any similarly functioning instrumentalities also infringes the charted claim. CIT reserves the right to amend this infringement analysis based on other products made, used, sold, imported, or offered for sale by CKE. CIT also reserves the right to amend this infringement analysis by citing other claims of the '296 patent, not listed in the claim chart, that are infringed by the Accused Instrumentalities. CIT further reserves the right to amend this infringement analysis by adding, subtracting, or otherwise modifying content in the "Accused Instrumentalities" column of each chart.

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,266,296 – Defendant / CKE Restaurants Holdings, Inc.
Claims 1 & 5

	Claim 1	**Carl's Jr. Downloadable App Service**  https://play.google.com/store/apps/developer?id=CKE+Restaurants gle Play Games Apps Movies & TV Books Kids LESS TAPS MORE SNACKS. FEED YOUR HAPPY With the Carl's Jr. App you can order your recent and favorite orders fast. With the Hardee's App you can order your recent and favorite orders fast. CKE Restaurants Carl's Jr.® CKE Restaurants 4.2 ★ Hardee's® CKE Restaurants 4.3 ★
1	A method comprising:	A method is specified for controlling a virtual session on a user device like a mobile handset such as smartphone or tablet.
1a (i)	receiving, at a control program executing on a mobile handset, a first communication initiated by a remote entity,	Wireless push notification messages are sent over Transport Layer Security (TLS) sessions. Each Push message includes an encrypted push token as per **Endnote #1**. The push token is sent in a Push Notification message over a TLS session from the **Carl's Jr.** Server backend to the **Carl's Jr.** App (application program, application layer program) running on a user's smartphone (mobile handset) or tablet. See Endnote #1 and Endnote #3. The **Carl's Jr.** App executes a registration process which includes

		instructions that will be executed to receive the specific push notifications coming into the **Carl's Jr.** App. In the Carl's Jr. application, for example, a push notification contains information related to products and/or services.  https://www.carlsjr.com/myrewards-terms **Communication Options and Your Privacy** In order to maintain your My Rewards Account, we must share relevant information with our service providers. Our Privacy Policy is incorporated into these Terms and governs your participation in My Rewards. Please read the Privacy Policy carefully to understand how we collect, use, and disclose your personal information, how to update or change your personal information, and how we communicate with you. My Rewards Program communications to Members may include in-App messaging, email, SMS text messaging, and push notifications. Members will be given the ability to opt in to each method of communication, except in-App messaging which is automatic and non-optional for App users. You may change your communication preferences as follows: (1) You may unsubscribe from promotional emails by clicking the "unsubscribe" link at the bottom of any email; (2) you may unsubscribe from SMS text notifications by replying with "STOP" or other opt-out language as instructed in any text message you receive from us regarding My Rewards; and (3) push notifications may be changed by adjusting the settings on your mobile device. Any exercise of Member privacy rights, including without limitation, opting out from our promotional emails, will not limit your ability to participate in the My Rewards Program. Please note that if you are a California consumer and submit a CCPA-related deletion request, we will ask you if you want to terminate your My Rewards Membership (and Account) or to retain your Membership. If you choose to retain your My Rewards Membership, we will continue to use your personal information to provide the My Rewards Program and related services or as otherwise permitted by applicable law. **Additional Benefits** Members may be prompted to provide additional information so we can provide more personalized My Rewards offers and experience. Additional Member benefits, such as special promotions, offers, or events, may be available from time to time. These benefits may require a completed profile and/or opt-in to push notifications, email, and/or location services to be eligible and for you to receive notification of the benefits, all as explained in the terms of each offer. Failure to enroll for such communications, to activate such features, or to have network connectivity, may result in your not receiving information about benefits. We are not responsible for any such failures. As per **Endnote #1**, the remote server causes a push notification message to be sent to the mobile handset. Part of this push notification message (first communication) will be forwarded to the **Carl's Jr.** App running on the user's smartphone or tablet device.
1a (ii)	wherein the first communication includes a **set of information identifying an application layer program** that is installed on the mobile handset, and	**See Endnote #1** for a discussion of how each Push Notification message coming into the Carl's Jr. application includes an **app-specific device token**. **The app-specific device token** is indicative of the Carl's Jr. App running on the user's smartphone or tablet. Each incoming wireless push notification message contains an **app-specific device token** which is a set of information that identifies the push-service reception process portion of the **Carl's Jr.** App**.**
1a (iii)	wherein initiation of the first communication by the remote entity was not in response to a request sent by the mobile handset;	The message sent by the server is called a push message or a push notification. Push notifications are call-out type messages, and as such, are not sent in response to pull requests sent by the mobile handset. See **Endnote #1**.

1b	the control program **causing** the mobile handset to **evaluate** the set of information included in the first communication; and	The control program is connected to the phone's OS that evaluates the first communication by looking at the **app-specific device token**. This lets the system know to which App on the device to activate or to send the new incoming information to. See **Endnote #1** for a discussion of how each Push Notification message coming into the **Carl's Jr.** App includes an **app-specific device token**. The app-specific device token is indicative of the **Carl's Jr.** App running on the user's smartphone or tablet. When the push notification has been received by the **Carl's Jr.** App, the **Carl's Jr.** App provides user interface capabilities that allow the user to click application data information received in the push message payload. When the user clicks this information, the **Carl's Jr.** App evaluates this information and causes the **Carl's Jr.** App to launch.
1c (i)	in response to determining, based on the evaluating, that the set of information identifies the application layer program, the control program **causing** the mobile handset to:	Determining is performed, for example, when a user clicks on a banner notification icon or a notification icon in the notifications tray. This determining is based on the evaluating, because the evaluating looks at the **app-specific device token** and identifies the incoming push notification message with the Carl's Jr. application program on the handset.
1c (ii)	launch the application layer program; and	Upon this determining, the notification icon in the banner or the notifications tray is displayed in accordance to system configuration commands issued by the **Carl's Jr.** App. The **Carl's Jr.** App issues commands to control how the notification will be displayed in the notifications tray upon receipt of the push message. When these notification display-control commands are executed, this is launching the App in the sense of executing notification display commands that are part of the **Carl's Jr.** App itself. Upon user click, the rest of the **Carl's Jr.** App was launched during testing. See Endnote #3 for further details. Also, if a data message is sent while the **Carl's Jr.** App is in the background, the **Carl's Jr.** App will launch and execute the onMessageReceived() function. See Endnote #4 for further details.

1c (iii)	reactivate, from an inactive state, a communication session between the mobile handset and the remote entity.	The Server Application and the client-side App have already established a separate TLS connection for traditional client-server communications. For example the **Carl's Jr.** Application Server program uses a TLS session to communicate application data with the **Carl's Jr.** App. Also in response to the user-clicking of the notification message and launching of the App, and possibly other user interface selections provided in response thereto, the TLS session between the **Carl's Jr.** Application Server program and the **Carl's Jr.** App is resumed. See **Endnote#2** for a discussion of TLS session resumption. Also see, https://docs.microsoft.com/en-us/windows/desktop/secauthn/tls-handshake-protocol. The remote server and the Carl's Jr. application will resume their client-server TLS session so that the server and the remote unit can resume communications. To do so the application program will invoke a protocol stack within the remote unit to communicate back to the server via the remote unit. TLS session use a full handshake sequence that is used to establish connection parameters, and an abbreviated handshake sequence that is used to resume the TLS session from an inactive or dormant state to an active state whereby new payload data can be sent via the virtual session once again.

5	The method of claim 1, wherein the first communication was initiated by the remote entity in response to information received by the remote entity from a server coupled to the remote entity via the Internet.	The message (first communication) sent by the server is called a push message or a push notification or a data message. **Carl's Jr.** push notifications are call-out type messages, and as such, are initiated and sent by the remote entity (**Carl's Jr.** push server) in response to information received by the remote entity from the **Carl's Jr.** backend server that is coupled to the remote entity via the Internet. *See* Endnote #1.

Communication Interface Technologies LLC's First Infringement Analysis

U.S. Patent No. 8,266,296 – Defendant / CKE Restaurants Holdings, Inc.
Claims 1 & 5

Caveat: The notes and/or cited excerpts utilized herein are set forth for illustrative purposes only and are not meant to be limiting in any manner. For example, the notes and/or cited excerpts, may or may not be supplemented or substituted with different excerpt(s) of the relevant reference(s), as appropriate. Further, to the extent any error(s) and/or omission(s) exist herein, all rights are reserved to correct the same.

Communication Interface Technologies LLC's First Infringement Analysis

U.S. Patent No. 8,266,296 – Defendant / CKE Restaurants Holdings, Inc.
Claims 1 & 5

Endnote#1 - App-Specific Device Token

https://help.pushwoosh.com/hc/en-us/articles/360000364923-What-is-a-Device-token-

Question:

What is a Device token?

Answer:

Push token (device token) - is a unique key for the app-device combination which is issued by the Apple or Google push notification gateways. It allows gateways and push notification providers to route messages and ensure the notification is delivered only to the unique app-device combination for which it is intended.

iOS device push tokens are strings with 64 hexadecimal symbols. Push token example:
`03df25c845d460bcdad7802d2vf6fc1dfde97283bf75cc993eb6dca835ea2e2f`
Make sure that iOS push tokens you use when targeting specific devices in your API requests are in lower case.

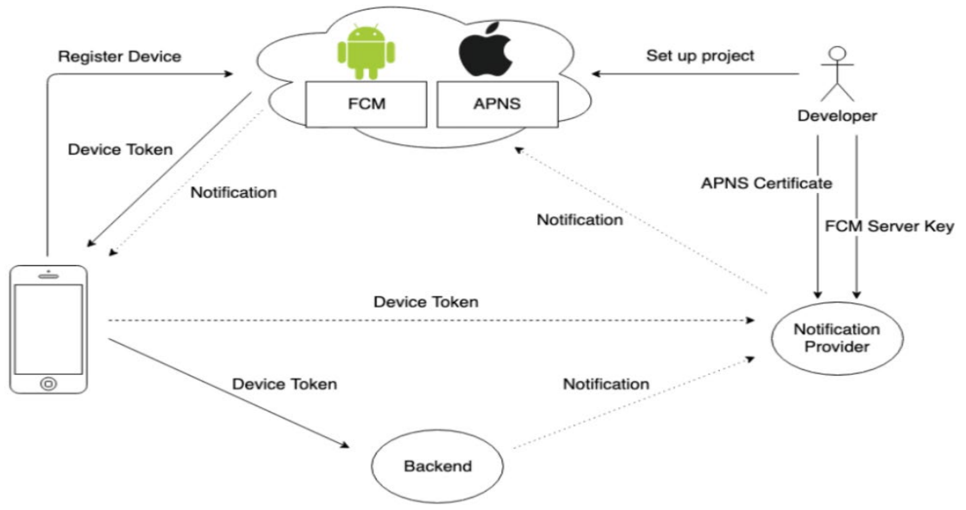
Android device push tokens can differ in length (usually below 255 characters), and usually start with APA…Push token example:

```
APA91bFoi3lMMre9G3XzR1LrF4ZT82_15MsMdEICogXSLB8-
MrdkRuRQFwNI5u8Dh0cI90ABD3BOKnxkEla8cGdisbDHl5cVIkZah5QUhSAxzx4Roa7b4xy9tvx9iNSYw-eXBYYd8k1XKf8Q_Qq1X9-
x-U-Y79vdPq
```

Note: The Android device push tokens correspond to the app-specific device token terminology used in the claim charts.

https://dev.to/jakubkoci/react-native-push-notifications-313i

Architecture



Note: In the above Architecture, the "backend" corresponds to the backend of the Application Server. The Device token corresponds to a specific App running on a specific device. That is what is meant by the app-specific device token in the claim charts.

https://firebase.google.com/docs/cloud-messaging/android/first-message

Access the registration token

To send a message to a specific device, you need to know that device's registration token. Because you'll need to enter the token in a field in the Notifications console to complete this tutorial, make sure to copy the token or securely store it after you retrieve it.

On initial startup of your app, the FCM SDK generates a registration token for the client app instance. If you want to target single devices or create device groups, you'll need to access this token by extending FirebaseMessagingService and overriding on NewToken.

This section describes how to retrieve the token and how to monitor changes to the token. Because the token could be rotated after initial startup, you are strongly recommended to retrieve the latest updated registration token.

The registration token may change when:

- The app deletes Instance ID
- The app is restored on a new device
- The user uninstalls/reinstall the app
- The user clears app data."

https://firebase.google.com/docs/cloud-messaging/concept-options

For example, here is a JSON-formatted notification message in an IM app. The user can expect to see a message with the title "Portugal vs. Denmark" and the text "great match!" on the device:

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,266,296 – Defendant / CKE Restaurants Holdings, Inc.
Claims 1 & 5

```
{
 "message":{
   "token":"bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1...", ←-- App-specific token
   "notification":{
     "title":"Portugal vs. Denmark",
     "body":"great match!"
   }
 }
}
```

https://firebase.google.com/docs/cloud-messaging/android/client

Retrieve the current registration token

When you need to retrieve the current token, call FirebaseInstanceId.getInstance().getInstanceId():

```
FirebaseInstanceId.getInstance().getInstanceId()
        .addOnCompleteListener(new OnCompleteListener<InstanceIdResult>() {
            @Override
            public void onComplete(@NonNull Task<InstanceIdResult> task) {
                if (!task.isSuccessful()) {
                    Log.w(TAG, "getInstanceId failed", task.getException());
                    return;
                }
                // Get new Instance ID token
                String token = task.getResult().getToken();

                // Log and toast
                String msg = getString(R.string.msg_token_fmt, token);
                Log.d(TAG, msg);
                Toast.makeText(MainActivity.this, msg, Toast.LENGTH_SHORT).show();
            }
        });
```

MainActivity.java

Monitor token generation

The onNewToken callback fires whenever a new token is generated.

```
/**
 * Called if InstanceID token is updated. This may occur if the security of
 * the previous token had been compromised. Note that this is called when the InstanceID token
 * is initially generated so this is where you would retrieve the token.
 */
@Override
public void onNewToken(String token) {
   Log.d(TAG, "Refreshed token: " + token);

   // If you want to send messages to this application instance or
   // manage this apps subscriptions on the server side, send the
   // Instance ID token to your app server.
```

```
    sendRegistrationToServer(token);
}
```

After you've obtained the token, you can send it to your app server and store it using your preferred method. See the Instance ID API reference [https://firebase.google.com/docs/reference/android/com/google/firebase/iid/FirebaseInstanceId] for full detail on the API.

<u>Endnote#2 - Transport Layer Security and Virtual Sessions</u>

Transport Layer Security (TLS) is used by both Apple iOS and Android based devices. The handshake diagrams in this endnote use Apple iOS as an example but apply equally to Android type implementations.

https://developer.android.com/training/articles/security-ssl

"The Secure Sockets Layer (SSL)—now technically known as <u>Transport Layer Security (TLS)</u>—is a <u>common building block for encrypted communications between clients and servers."</u>

https://android-developers.googleblog.com/2018/04/protecting-users-with-tls-by-default-in.html

"<u>Android</u> <u>is committed to keeping users, their devices, and their data safe</u>. One of the ways that we keep data safe is <u>by protecting all data that enters or leaves an Android device with Transport Layer Security (TLS) in transit</u>.

A. Razaghpanah et al., *Studying TLS Usage in Android Apps,* CoNEXT '17, Dec.12-15, 2017, Incheon, Republic of Korea
http://abbas.rpanah.ir/publications/conext2017_tls_paper.pdf

"**A History of TLS Support in Android:** Android has supported TLS 1.0 since its first version released in 2008 and TLS 1.1 and TLS 1.2 since 2012."

"However, other protocols such as secure email (42 apps) and Google's Cloud Messaging service for push notifications (9 apps) [11, 47] also use TLS."

https://developer.ibm.com/customer-engagement/docs/watson-marketing/ibm-engage-2/tls-1-2-migration-for-mobile-push-clients/

What will happen on devices that are unable to support TLS 1.2?

Devices which do not support TLS 1.2 will be unable to connect to our WCA servers. This will prevent users of those devices from:

• Registering new mobile user IDs
• Updating push tokens
• Receiving inbox messages
• Receiving In-app messages

Note: As the above link shows, the creation of the App IDs of Endnote #1 are linked to the TLS protocol being run on the TLS-enabled Push-Notification channel.

https://tools.ietf.org/html/rfc5246

F.1.4. Resuming Sessions

When a connection is established by resuming a session, new ClientHello.random and ServerHello.random values are hashed with the session's master_secret. Provided that the master_secret has not been compromised and that the secure hash operations used to produce the encryption keys and MAC keys are secure, the connection should be secure and effectively independent from previous connections. Attackers cannot use known encryption keys or MAC secrets to compromise the master_secret without breaking the secure hash operations.

Sessions cannot be resumed unless both the client and server agree. If either party suspects that the session may have been compromised, or that certificates may have expired or been revoked, it should force a full handshake. An upper limit of 24 hours is suggested for session ID lifetimes, since an attacker who obtains a master_secret may be able to impersonate the compromised party until the corresponding session ID is retired. Applications that may be run in relatively insecure environments should not write session IDs to stable storage.

https://tools.ietf.org/html/rfc5077

Abstract

This document describes a mechanism that enables the Transport Layer Security (TLS) server to resume sessions and avoid keeping per-client session state. The TLS server encapsulates the session state into a ticket and forwards it to the client. The client can subsequently resume a session using the obtained ticket.

3. Protocol

This specification describes a mechanism to distribute encrypted session-state information in the form of a ticket. The ticket is created by a TLS server and sent to a TLS client. The TLS client presents the ticket to the TLS server to resume a session.

EXHIBIT 6

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc. Claim 1

Communication Interface Technologies LLC ("CIT") provides evidence of infringement of claim 1 of U.S. Patent No. 8,291,010 (hereinafter "the '010 patent") by CKE Restaurants Holdings, Inc. ("CKE"). In support thereof, CIT provides the following claim charts.

"Accused Instrumentalities" as used herein refers to at least CKE's application ("App") in systems and methods, including hardware and software products including, but not limited to the CKE App as developed for mobile electronic devices (*see, e.g.,* https://play.google.com/store/apps/details?id=com.carlsjr.ordering&hl=en_US&gl=US), along with any hardware and/or software for provisioning the App. Upon information and belief, the exemplary version herein and previous versions of the Accused Instrumentalities distributed prior to expiration of the patents in suit operated materially in the same manner.

These claim charts demonstrate CKE's infringement, and provide notice of such infringement, by comparing each element of the asserted claim to corresponding components, aspects, and/or features of the Accused Instrumentalities. These claim charts are not intended to constitute an expert report on infringement. These claim charts include information provided by way of example, and not by way of limitation.

The analysis set forth below is based only upon information from publicly available resources regarding the Accused Instrumentalities, as CKE has not yet provided any non-public information. An analysis of CKE's (or a third parties') technical documentation and/or software source code may assist in fully identifying all infringing features and functionality. Accordingly, CIT reserves the right to supplement this infringement analysis once such information is made available to CIT. Furthermore, CIT reserves the right to revise this infringement analysis, as appropriate, upon issuance of a court order construing any terms recited in the asserted claims.

CIT provides this evidence of infringement and related analysis without the benefit of claim construction or expert reports or discovery. CIT reserves the right to supplement, amend or otherwise modify this analysis and/or evidence based on any such claim construction or expert reports or discovery.

Unless otherwise noted, CIT contends that CKE has directly infringed the '010 patent in violation of 35 U.S.C. § 271(a) by selling, offering to sell, making, using, and/or importing the Accused Instrumentalities. The following exemplary analysis demonstrates that infringement.

Communication Interface Technologies LLC's First Infringement Analysis

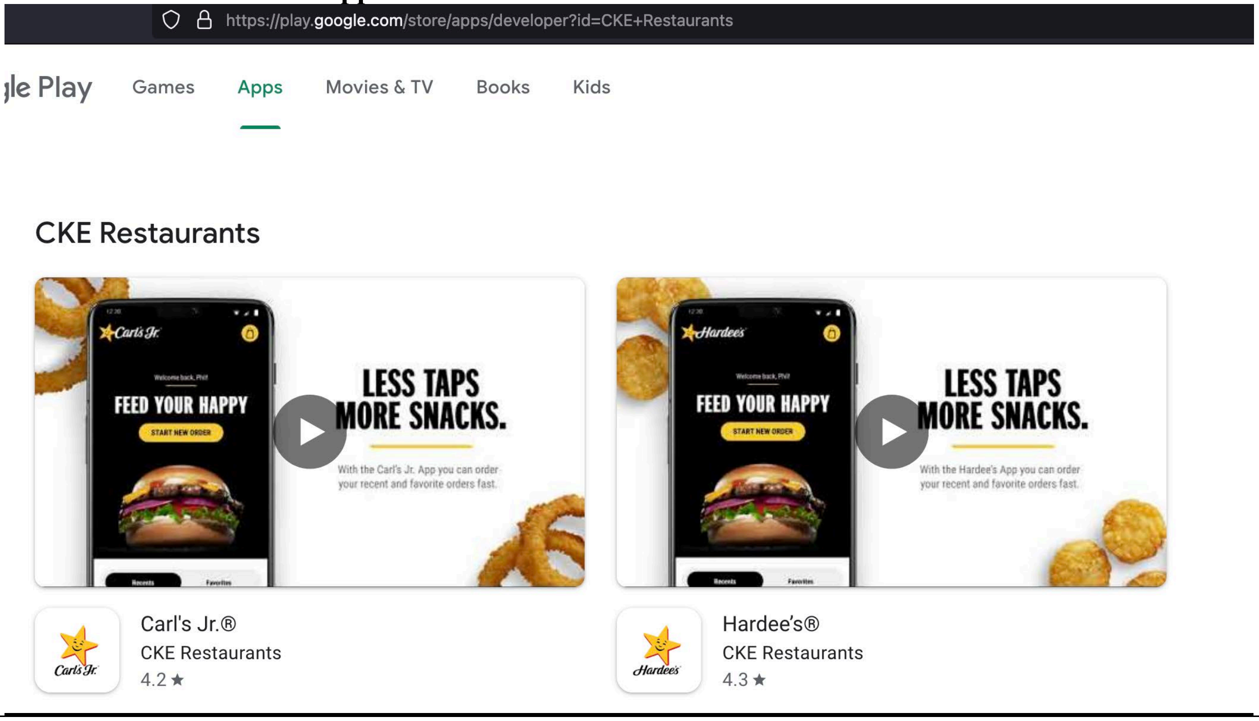
U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 1

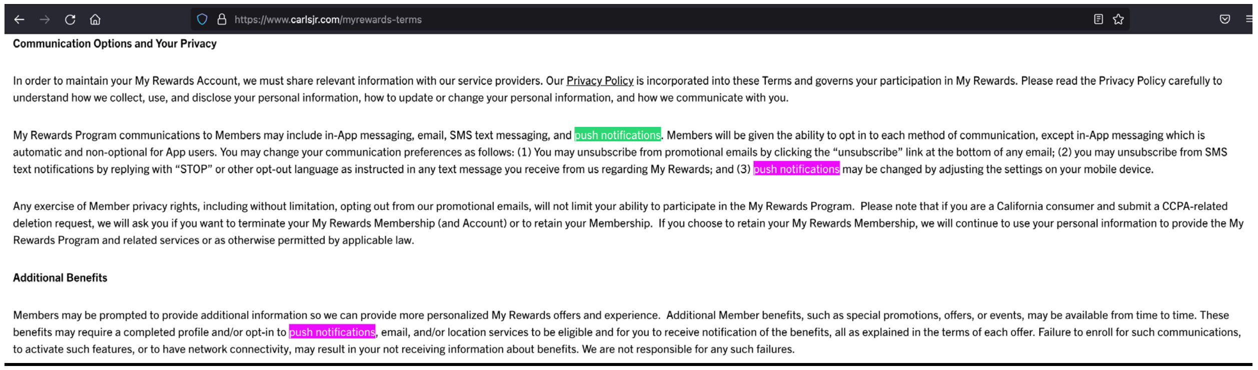
Unless otherwise noted, CIT believes and contends that each element of each claim asserted herein is literally met through CKE's provision of the Accused Instrumentalities. However, to the extent that CKE attempts to allege that any asserted claim element is not literally met, CIT believes and contends that such elements are met under the doctrine of equivalents. More specifically, in its investigation and analysis of the Accused Instrumentalities, CIT did not identify any substantial differences between the elements of the patent claims and the corresponding features of the Accused Instrumentalities, as set forth herein. In each instance, the identified feature of the Accused Instrumentalities performs at least substantially the same function in substantially the same way to achieve substantially the same result as the corresponding claim element.

To the extent the chart of an asserted claim relies on evidence about certain specifically identified Accused Instrumentalities, CIT asserts that, on information and belief, any similarly functioning instrumentalities also infringes the charted claim. CIT reserves the right to amend this infringement analysis based on other products made, used, sold, imported, or offered for sale by CKE. CIT also reserves the right to amend this infringement analysis by citing other claims of the '010 patent, not listed in the claim chart, that are infringed by the Accused Instrumentalities. CIT further reserves the right to amend this infringement analysis by adding, subtracting, or otherwise modifying content in the "Accused Instrumentalities" column of each chart.

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 1

	Claim 1	**Carl's Jr. Downloadable App Service**
		 https://play.google.com/store/apps/developer?id=CKE+Restaurants gle Play Games Apps Movies & TV Books Kids LESS TAPS MORE SNACKS. FEED YOUR HAPPY CKE Restaurants Carl's Jr.® CKE Restaurants 4.2 ★ Hardee's® CKE Restaurants 4.3 ★
1	A method comprising:	A method is specified for use on a user device like a handheld or tablet.
1a	establishing, at a computing device, a communication session supporting communication between a first program executing at an application layer of the computing device and a remote server;	Wireless push notification messages are sent over Transport Layer Security (TLS) sessions. Each Push message includes an encrypted push token as per **Endnote #1**. The push token is sent in a Push Notification message over TLS sessions from the **Carl's Jr.** Server backend (remote server) to the **Carl's Jr.** App (application program, application layer program) running on a user's smartphone (mobile handset) or tablet. In the Carl's Jr. application, for example, a push notification contains information related to products and/or

		services.  https://www.carlsjr.com/myrewards-terms **Communication Options and Your Privacy** In order to maintain your My Rewards Account, we must share relevant information with our service providers. Our Privacy Policy is incorporated into these Terms and governs your participation in My Rewards. Please read the Privacy Policy carefully to understand how we collect, use, and disclose your personal information, how to update or change your personal information, and how we communicate with you. My Rewards Program communications to Members may include in-App messaging, email, SMS text messaging, and push notifications. Members will be given the ability to opt in to each method of communication, except in-App messaging which is automatic and non-optional for App users. You may change your communication preferences as follows: (1) You may unsubscribe from promotional emails by clicking the "unsubscribe" link at the bottom of any email; (2) you may unsubscribe from SMS text notifications by replying with "STOP" or other opt-out language as instructed in any text message you receive from us regarding My Rewards; and (3) push notifications may be changed by adjusting the settings on your mobile device. Any exercise of Member privacy rights, including without limitation, opting out from our promotional emails, will not limit your ability to participate in the My Rewards Program. Please note that if you are a California consumer and submit a CCPA-related deletion request, we will ask you if you want to terminate your My Rewards Membership (and Account) or to retain your Membership. If you choose to retain your My Rewards Membership, we will continue to use your personal information to provide the My Rewards Program and related services or as otherwise permitted by applicable law. **Additional Benefits** Members may be prompted to provide additional information so we can provide more personalized My Rewards offers and experience. Additional Member benefits, such as special promotions, offers, or events, may be available from time to time. These benefits may require a completed profile and/or opt-in to push notifications, email, and/or location services to be eligible and for you to receive notification of the benefits, all as explained in the terms of each offer. Failure to enroll for such communications, to activate such features, or to have network connectivity, may result in your not receiving information about benefits. We are not responsible for any such failures. Also, the Server Application and the client-side App establish a separate TLS connection for traditional client-server communications. For example the **Carl's Jr.** Application Server program establishes a TLS session with the **Carl's Jr.** App. The TLS session used for client-server communications between the **Carl's Jr.** Application Server and the **Carl's Jr.** App correspond to the recited communication session. TLS sessions use a full handshake sequence that is used to establish connection parameters, and an abbreviated handshake sequence that is used to resume the TLS session from an inactive or dormant state to an active state whereby new payload data can be sent via the virtual session once again. See **Endnote#2** for a discussion of the virtual session aspects of TLS. Mobile applications communicate with their application server via TLS connections. These TLS connections are established at the time the app is installed or launched and can be resumed at a later time using a session token. See **Endnote#2** for a discussion of TLS.

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 1

		https://threema.ch/press-files/cryptography_whitepaper.pdf - Android uses TLS 1.2 resumable sessions. https://www.icir.org/johanna/papers/conext17android.pdf - Android supports TLS and secure connections between client app and server are ubiquitously used. https://developer.android.com/training/articles/security-ssl. – "The Secure Sockets Layer (SSL)—now technically known as Transport Layer Security (TLS)—is a common building block for encrypted communications between clients and servers." - Note that certain certificates are used, and these are used to help create Android Device Tokens used in Push Notification messages. **See Endnotes #1, and #2**.
1b (i)	subsequent to deactivation of the established communication session,	See **Endnote #2**. Note when the application data phase is finished, the TLS client-server data session is placed back into the inactive state. Hence the end of application data marker is the signal used to place this TLS session into the inactive state.
1b (ii)	the computing device receiving an incoming communication from the remote server, wherein the incoming communication is not in response to a request sent by the computing device;	As per **Endnote #1**, the remote server causes a push notification message to be sent to the computing device. Part of this push notification message (incoming communication) will be forwarded to the **Carl's Jr.** App running on the user's smartphone or tablet device. The push notification message is asynchronously initiated by the remote server(**Carl's Jr.** Application Server) as opposed to being sent in response to a request sent by the user computing device.
1c	at the application layer, the computing device reading a set of information included in the incoming communication;	An **app-specific device token** included in the incoming communication lets the system know to which App on the device to activate or to send the new incoming information to. See **Endnote #1** for a discussion of how each Push Notification message coming into the **Carl's Jr.** App includes an **app-specific device token**. The app-specific device token is indicative of the **Carl's Jr.** App running on the user's smartphone or tablet. When the push notification has been received by the **Carl's Jr.** App, the **Carl's Jr.** App provides user interface capabilities that allow the user to click application data information received in the push message payload. When the user clicks this information, the **Carl's Jr.** App evaluates this information and causes the **Carl's Jr.** App to launch.

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 1

1d	in response to determining that the set of information read at the application layer includes information identifying the first program executing at the computing device, the computing device reactivating the communication session between the first program and the remote server.	The remote server sends a push notification message to the Carl's Jr. application running in the smartphone or tablet operated by a specified user. As per **Endnote #1**, this push notification message is processed by the operating system using specific instructions supplied by the app during registration. The **app-specific device token** and related information (incoming communication) is forwarded from the OS to the **Carl's Jr.** App. If a data message is sent, or if the push notification is received when the **Carl's Jr.** App is already in the foreground state, then the data message or the push notification is handled directly from within the App as opposed to being routed to the notifications tray. The **Carl's Jr.** App has an onMessageReceive() function and this automatically handles the incoming push message. The push message includes URI related data and causes the TLS session to be resumed in order for the App itself to handle and display information related to the incoming push notification. See **Endnote#2** for a discussion of TLS session resumption. Also see, https://docs.microsoft.com/en-us/windows/desktop/secauthn/tls-handshake-protocol. The remote server and the Carl's Jr. application will resume their client-server TLS session so that the server and the remote unit can resume communications. To do so the application program will invoke a protocol stack within the remote unit to communicate back to the server via the remote unit. TLS session use a full handshake sequence that is used to establish connection parameters, and an abbreviated handshake sequence that is used to resume the TLS session from an inactive or dormant state to an active state whereby new payload data can be sent via the virtual session once again.

Caveat: The notes and/or cited excerpts utilized herein are set forth for illustrative purposes only and are not meant to be limiting in any manner. For example, the notes and/or cited excerpts, may or may not be supplemented or substituted with different excerpt(s) of the relevant reference(s), as appropriate. Further, to the extent any error(s) and/or omission(s) exist herein, all rights are reserved to correct the same.

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 1

Endnote#1 - App-Specific Device Token

https://help.pushwoosh.com/hc/en-us/articles/360000364923-What-is-a-Device-token-

Question:

What is a Device token?

Answer:

Push token (device token) - is a unique key for the app-device combination which is issued by the Apple or Google push notification gateways. It allows gateways and push notification providers to route messages and ensure the notification is delivered only to the unique app-device combination for which it is intended.

iOS device push tokens are strings with 64 hexadecimal symbols. Push token example:
`03df25c845d460bcdad7802d2vf6fc1dfde97283bf75cc993eb6dca835ea2e2f`
Make sure that iOS push tokens you use when targeting specific devices in your API requests are in lower case.

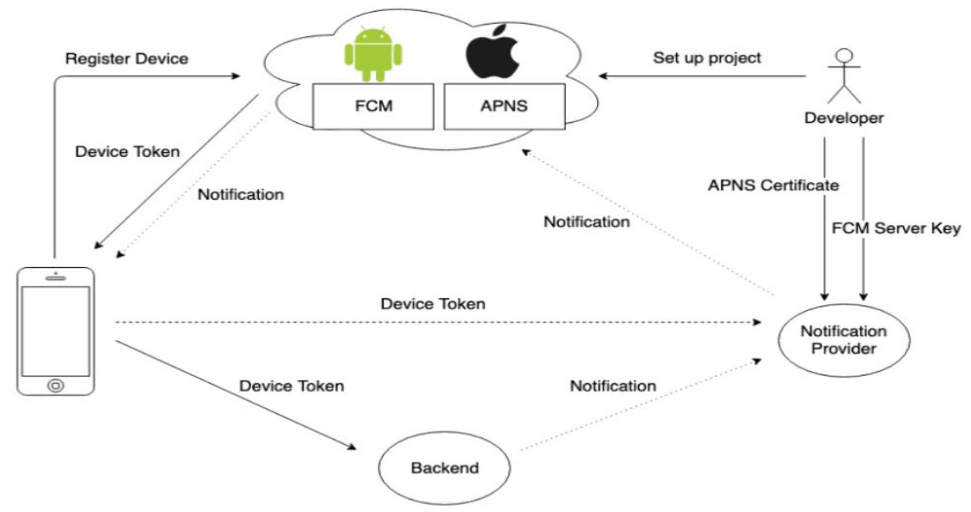
Android device push tokens can differ in length (usually below 255 characters), and usually start with APA…Push token example:

```
APA91bFoi3lMMre9G3XzR1LrF4ZT82_15MsMdEICogXSLB8-
MrdkRuRQFwNI5u8Dh0cI90ABD3BOKnxkEla8cGdisbDHl5cVIkZah5QUhSAxzx4Roa7b4xy9tvx9iNSYw-eXBYYd8k1XKf8Q_Qq1X9-
x-U-Y79vdPq
```

Note: The Android device push tokens correspond to the app-specific device token terminology used in the claim charts.

https://dev.to/jakubkoci/react-native-push-notifications-313i

Architecture



Note: In the above Architecture, the "backend" corresponds to the backend of the Application Server. The Device token corresponds to a specific App running on a specific device. That is what is meant by the app-specific device token in the claim charts.

https://firebase.google.com/docs/cloud-messaging/android/first-message

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 1

Access the registration token

To send a message to a specific device, you need to know that device's registration token. Because you'll need to enter the token in a field in the Notifications console to complete this tutorial, make sure to copy the token or securely store it after you retrieve it.

On initial startup of your app, the FCM SDK generates a registration token for the client app instance. If you want to target single devices or create device groups, you'll need to access this token by extending FirebaseMessagingService and overriding on NewToken.

This section describes how to retrieve the token and how to monitor changes to the token. Because the token could be rotated after initial startup, you are strongly recommended to retrieve the latest updated registration token.

The registration token may change when:

- The app deletes Instance ID
- The app is restored on a new device
- The user uninstalls/reinstall the app
- The user clears app data."

https://firebase.google.com/docs/cloud-messaging/concept-options

For example, here is a JSON-formatted notification message in an IM app. The user can expect to see a message with the title "Portugal vs. Denmark" and the text "great match!" on the device:

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 1

```
{
 "message":{
   "token":"bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1...", ←-- App-specific token
   "notification":{
     "title":"Portugal vs. Denmark",
     "body":"great match!"
   }
 }
}
```

https://firebase.google.com/docs/cloud-messaging/android/client

Retrieve the current registration token

When you need to retrieve the current token, call FirebaseInstanceId.getInstance().getInstanceId():

```
FirebaseInstanceId.getInstance().getInstanceId()
        .addOnCompleteListener(new OnCompleteListener<InstanceIdResult>() {
            @Override
            public void onComplete(@NonNull Task<InstanceIdResult> task) {
                if (!task.isSuccessful()) {
                    Log.w(TAG, "getInstanceId failed", task.getException());
                    return;
                }

                // Get new Instance ID token
                String token = task.getResult().getToken();

                // Log and toast
                String msg = getString(R.string.msg_token_fmt, token);
                Log.d(TAG, msg);
                Toast.makeText(MainActivity.this, msg, Toast.LENGTH_SHORT).show();
            }
        });
```

MainActivity.java

Monitor token generation

The onNewToken callback fires whenever a new token is generated.

```
/**
 * Called if InstanceID token is updated. This may occur if the security of
 * the previous token had been compromised. Note that this is called when the InstanceID token
 * is initially generated so this is where you would retrieve the token.
 */
@Override
public void onNewToken(String token) {
   Log.d(TAG, "Refreshed token: " + token);

   // If you want to send messages to this application instance or
   // manage this apps subscriptions on the server side, send the
   // Instance ID token to your app server.
   sendRegistrationToServer(token);
}
```

After you've obtained the token, you can send it to your app server and store it using your preferred method. See the Instance ID API reference [https://firebase.google.com/docs/reference/android/com/google/firebase/iid/FirebaseInstanceId] for full detail on the API.

Endnote#2 - Transport Layer Security and Virtual Sessions

Transport Layer Security (TLS) is used by both Apple iOS and Android based devices. The handshake diagrams in this endnote use Apple iOS as an example but apply equally to Android type implementations.

https://developer.android.com/training/articles/security-ssl

"The Secure Sockets Layer (SSL)—now technically known as Transport Layer Security (TLS)—is a common building block for encrypted communications between clients and servers."

https://android-developers.googleblog.com/2018/04/protecting-users-with-tls-by-default-in.html

"Android is committed to keeping users, their devices, and their data safe. One of the ways that we keep data safe is by protecting all data that enters or leaves an Android device with Transport Layer Security (TLS) in transit.

A. Razaghpanah et al., *Studying TLS Usage in Android Apps,* CoNEXT '17, Dec.12-15, 2017, Incheon, Republic of Korea *http://abbas.rpanah.ir/publications/conext2017_tls_paper.pdf*

"**A History of TLS Support in Android:** Android has supported TLS 1.0 since its first version released in 2008 and TLS 1.1 and TLS 1.2 since 2012."

"However, other protocols such as secure email (42 apps) and Google's Cloud Messaging service for push notifications (9 apps) [11, 47] also use TLS."

https://developer.ibm.com/customer-engagement/docs/watson-marketing/ibm-engage-2/tls-1-2-migration-for-mobile-push-clients/

What will happen on devices that are unable to support TLS 1.2?

Devices which do not support TLS 1.2 will be unable to connect to our WCA servers. This will prevent users of those devices from:

• Registering new mobile user IDs
• Updating push tokens
• Receiving inbox messages
• Receiving In-app messages

Note: As the above link shows, the creation of the App IDs of Endnote #1 are linked to the TLS protocol being run on the TLS-enabled Push-Notification channel.

https://tools.ietf.org/html/rfc5246

F.1.4. Resuming Sessions

When a connection is established by resuming a session, new ClientHello.random and ServerHello.random values are hashed with the session's master_secret. Provided that the master_secret has not been compromised and that the secure hash operations used to produce the encryption keys and MAC keys are secure, the connection should be secure and effectively independent from previous connections. Attackers cannot use known encryption keys or MAC secrets to compromise the master_secret without breaking the secure hash operations.

Sessions cannot be resumed unless both the client and server agree. If either party suspects that the session may have been compromised, or that certificates may have expired or been revoked, it should force a full handshake. An upper limit of 24 hours is suggested for session ID lifetimes, since an attacker who obtains a master_secret may be able to impersonate the compromised party until the corresponding session ID is retired. Applications that may be run in relatively insecure environments should not write session IDs to stable storage.

https://tools.ietf.org/html/rfc5077

Abstract

This document describes a mechanism that enables the Transport Layer Security (TLS) server to resume sessions and avoid keeping per-client session state. The TLS server encapsulates the session state into a ticket and forwards it to the client. The client can subsequently resume a session using the obtained ticket.

3. Protocol

This specification describes a mechanism to distribute encrypted session-state information in the form of a ticket. The ticket is created by a TLS server and sent to a TLS client. The TLS client presents the ticket to the TLS server to resume a session.

EXHIBIT 7

Communication Interface Technologies LLC's First Infringement Analysis

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 17

Communication Interface Technologies LLC ("CIT") provides evidence of infringement of claim 17 of U.S. Patent No. 8,291,010 (hereinafter "the '010 patent") by CKE Restaurants Holdings, Inc. ("CKE"). In support thereof, CIT provides the following claim charts.

"Accused Instrumentalities" as used herein refers to at least CKE's application ("App") in systems and methods, including hardware and software products including, but not limited to the CKE App as developed for mobile electronic devices (*see, e.g.,* https://play.google.com/store/apps/details?id=com.carlsjr.ordering&hl=en_US&gl=US), along with any hardware and/or software for provisioning the App. Upon information and belief, the exemplary version herein and previous versions of the Accused Instrumentalities distributed prior to expiration of the patents in suit operated materially in the same manner.

These claim charts demonstrate CKE's infringement, and provide notice of such infringement, by comparing each element of the asserted claim to corresponding components, aspects, and/or features of the Accused Instrumentalities. These claim charts are not intended to constitute an expert report on infringement. These claim charts include information provided by way of example, and not by way of limitation.

The analysis set forth below is based only upon information from publicly available resources regarding the Accused Instrumentalities, as CKE has not yet provided any non-public information. An analysis of CKE's (or a third parties') technical documentation and/or software source code may assist in fully identifying all infringing features and functionality. Accordingly, CIT reserves the right to supplement this infringement analysis once such information is made available to CIT. Furthermore, CIT reserves the right to revise this infringement analysis, as appropriate, upon issuance of a court order construing any terms recited in the asserted claims.

CIT provides this evidence of infringement and related analysis without the benefit of claim construction or expert reports or discovery. CIT reserves the right to supplement, amend or otherwise modify this analysis and/or evidence based on any such claim construction or expert reports or discovery.

Unless otherwise noted, CIT contends that CKE has directly infringed the '010 patent in violation of 35 U.S.C. § 271(a) by selling, offering to sell, making, using, and/or importing the Accused Instrumentalities. The following exemplary analysis demonstrates that infringement.

Communication Interface Technologies LLC's First Infringement Analysis

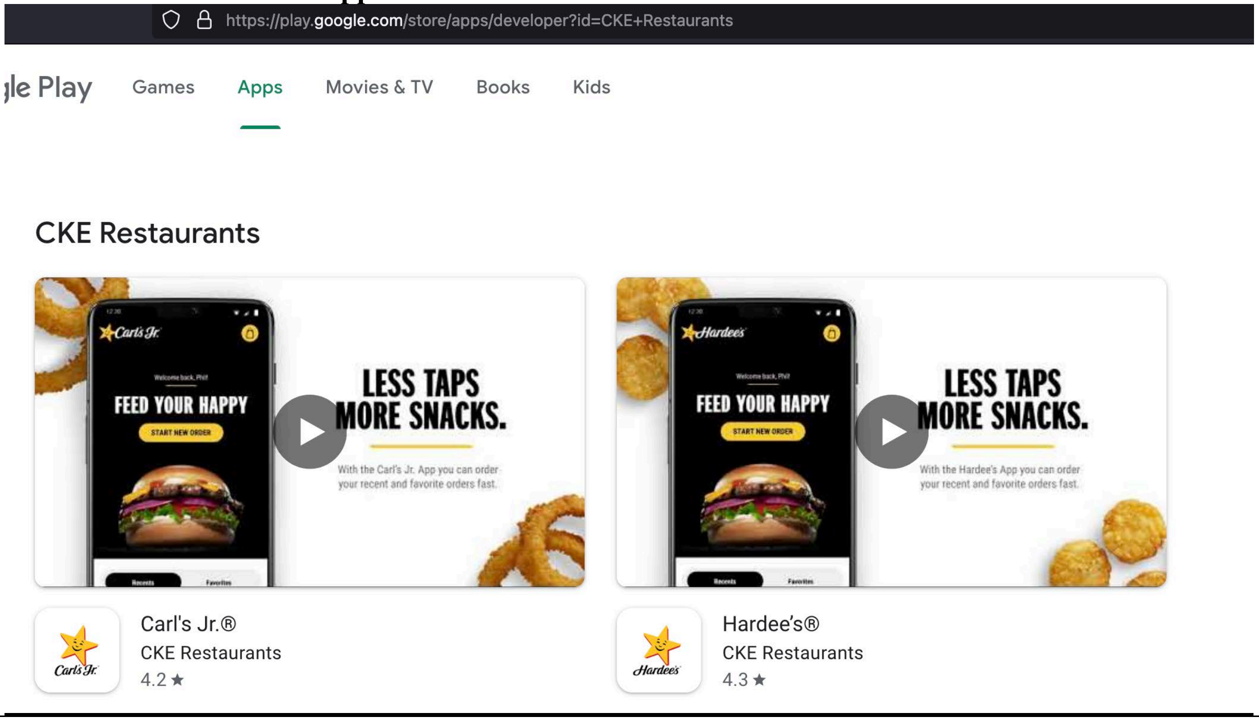
U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 17

Unless otherwise noted, CIT believes and contends that each element of each claim asserted herein is literally met through CKE's provision of the Accused Instrumentalities. However, to the extent that CKE attempts to allege that any asserted claim element is not literally met, CIT believes and contends that such elements are met under the doctrine of equivalents. More specifically, in its investigation and analysis of the Accused Instrumentalities, CIT did not identify any substantial differences between the elements of the patent claims and the corresponding features of the Accused Instrumentalities, as set forth herein. In each instance, the identified feature of the Accused Instrumentalities performs at least substantially the same function in substantially the same way to achieve substantially the same result as the corresponding claim element.

To the extent the chart of an asserted claim relies on evidence about certain specifically identified Accused Instrumentalities, CIT asserts that, on information and belief, any similarly functioning instrumentalities also infringes the charted claim. CIT reserves the right to amend this infringement analysis based on other products made, used, sold, imported, or offered for sale by CKE. CIT also reserves the right to amend this infringement analysis by citing other claims of the '010 patent, not listed in the claim chart, that are infringed by the Accused Instrumentalities. CIT further reserves the right to amend this infringement analysis by adding, subtracting, or otherwise modifying content in the "Accused Instrumentalities" column of each chart.

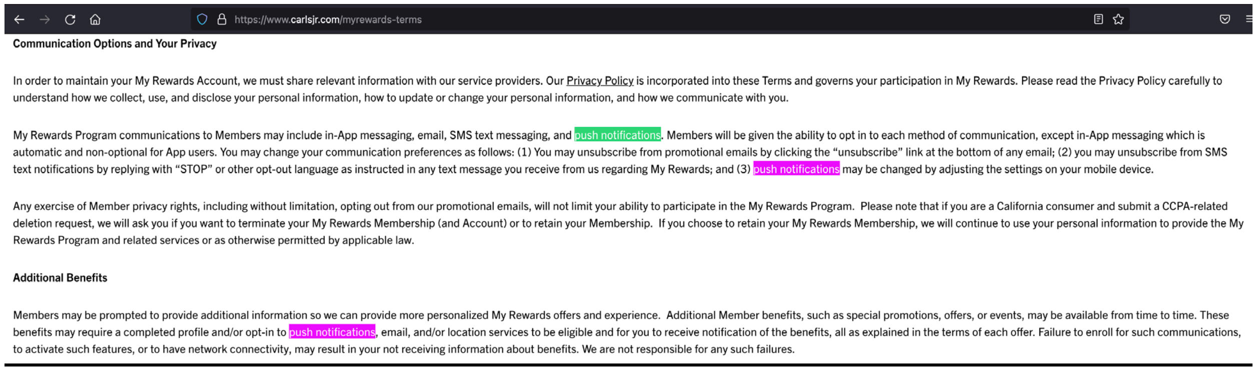
COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 17

	Claim 17	**Carl's Jr. Downloadable App Service**  https://play.google.com/store/apps/developer?id=CKE+Restaurants gle Play Games Apps Movies & TV Books Kids LESS TAPS MORE SNACKS. FEED YOUR HAPPY CKE Restaurants Carl's Jr.® CKE Restaurants 4.2★ Hardee's® CKE Restaurants 4.3★
17	A method performed by a computing device, the method comprising:	The computing device is Carl's Jr. App (application which controls a computing device such as a smartphone or tablet those carriers out the instructions of the application.)
17a (i)	during a communication session that has previously been established with a remote entity and that is currently inactive,	Wireless push notification messages are sent over Transport Layer Security (TLS) sessions. Each Push message includes an encrypted push token as per **Endnote #1**. The push token is sent in a Push Notification message over TLS sessions from the **Carl's Jr.** Server backend (remote entity) to the **Carl's Jr.** App (application program, application layer program) running on a user's smartphone (mobile handset) or tablet.

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 17

		In the Carl's Jr. application, for example, a push notification contains information related to products and/or services.  https://www.carlsjr.com/myrewards-terms **Communication Options and Your Privacy** In order to maintain your My Rewards Account, we must share relevant information with our service providers. Our Privacy Policy is incorporated into these Terms and governs your participation in My Rewards. Please read the Privacy Policy carefully to understand how we collect, use, and disclose your personal information, how to update or change your personal information, and how we communicate with you. My Rewards Program communications to Members may include in-App messaging, email, SMS text messaging, and push notifications. Members will be given the ability to opt in to each method of communication, except in-App messaging which is automatic and non-optional for App users. You may change your communication preferences as follows: (1) You may unsubscribe from promotional emails by clicking the "unsubscribe" link at the bottom of any email; (2) you may unsubscribe from SMS text notifications by replying with "STOP" or other opt-out language as instructed in any text message you receive from us regarding My Rewards; and (3) push notifications may be changed by adjusting the settings on your mobile device. Any exercise of Member privacy rights, including without limitation, opting out from our promotional emails, will not limit your ability to participate in the My Rewards Program. Please note that if you are a California consumer and submit a CCPA-related deletion request, we will ask you if you want to terminate your My Rewards Membership (and Account) or to retain your Membership. If you choose to retain your My Rewards Membership, we will continue to use your personal information to provide the My Rewards Program and related services or as otherwise permitted by applicable law. **Additional Benefits** Members may be prompted to provide additional information so we can provide more personalized My Rewards offers and experience. Additional Member benefits, such as special promotions, offers, or events, may be available from time to time. These benefits may require a completed profile and/or opt-in to push notifications, email, and/or location services to be eligible and for you to receive notification of the benefits, all as explained in the terms of each offer. Failure to enroll for such communications, to activate such features, or to have network connectivity, may result in your not receiving information about benefits. We are not responsible for any such failures. Also, the Server Application and the client-side App establish a separate TLS connection for traditional client-server communications. For example the **Carl's Jr.** Application Server program establishes a TLS session with the **Carl's Jr.** App. The TLS session used for client-server communications between the **Carl's Jr.** Application Server and the **Carl's Jr.** App correspond to the recited communication session. TLS sessions use a full handshake sequence that is used to establish connection parameters, and an abbreviated handshake sequence that is used to resume the TLS session from an inactive or dormant state to an active state whereby new payload data can be sent via the virtual session once again. See **Endnote#2** for a discussion of the virtual session aspects of TLS. Mobile applications communicate with their application server via TLS connections. These TLS connections are established at the time the app is installed or launched and can be resumed at a later time using a session token. See **Endnote#2** for a discussion of TLS. When the App is inactive and a Push

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 17

		Notification is received for the App or is retrieved or selected by the user from the Notifications tray, then this corresponds to a time during the client-server TLS communication session that has previously been established with a remote entity and that is currently inactive. https://threema.ch/press-files/cryptography_whitepaper.pdf - Android uses TLS 1.2 resumable sessions. https://www.icir.org/johanna/papers/conext17android.pdf - Android supports TLS and secure connections between client app and server are ubiquitously used. https://developer.android.com/training/articles/security-ssl. – "The Secure Sockets Layer (SSL)—now technically known as Transport Layer Security (TLS)—is a common building block for encrypted communications between clients and servers." - Note that certain certificates are used, and these are used to help create Android Device Tokens used in Push Notification messages. **See Endnotes #1, and #2**.
17a (ii)	receiving a first communication initiated by the remote entity, wherein the first communication has not been initiated by the remote entity in response to a previous communication from the computing device;	As per **Endnote #1**, the remote entity causes a push notification message to be sent to the computing device. Part of this push notification message (first communication) will be forwarded to the **Carl's Jr.** App running on the user's smartphone or tablet device. The push notification message is asynchronously initiated by the remote entity (**Carl's Jr.** Application Server) as opposed to being sent in response to a previous communication sent by the user computing device. The receiving is performed using instructions supplied by the **Carl's Jr.** App.
17b (i)	in response to determining that the received first communication includes information identifying an application at the computing device,	See **Endnote #1** for a discussion of how each Push Notification message coming into the **Carl's Jr.** App includes an **app-specific device token**. The app-specific device token is indicative of the **Carl's Jr.** App running on the user's smartphone or tablet. The receiving is performed using instructions supplied by the **Carl's Jr.** App. When the push notification has been received by the **Carl's Jr.** App, the **Carl's Jr.** App provides user interface capabilities that allow the user to click application data information received in the push message payload. When the user clicks this information, the **Carl's Jr.** App evaluates this information and causes the **Carl's Jr.** App to launch.

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 17

17b (ii)	reactivating the communication session with the remote entity.	If a data message is sent, or if the push notification is received when the **Carl's Jr.** App is already in the foreground state, then the data message or the push notification is handled directly from within the App as opposed to being routed to the notifications tray. The **Carl's Jr.** App has an onMessageReceive() function and this automatically handles the incoming push message. The push message includes URI related data and causes the TLS session to be resumed in order for the App itself to handle and display information related to the incoming push notification. See **Endnote#2** for a discussion of TLS session resumption. Also see, https://docs.microsoft.com/en-us/windows/desktop/secauthn/tls-handshake-protocol. The remote server and the Carl's Jr. application will resume their client-server TLS session so that the server and the remote unit can resume communications. To do so the application program will invoke a protocol stack within the remote unit to communicate back to the server via the remote unit. TLS session use a full handshake sequence that is used to establish connection parameters, and an abbreviated handshake sequence that is used to resume the TLS session from an inactive or dormant state to an active state whereby new payload data can be sent via the virtual session once again.

Caveat: The notes and/or cited excerpts utilized herein are set forth for illustrative purposes only and are not meant to be limiting in any manner. For example, the notes and/or cited excerpts, may or may not be supplemented or substituted with different excerpt(s) of the relevant reference(s), as appropriate. Further, to the extent any error(s) and/or omission(s) exist herein, all rights are reserved to correct the same.

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 17

Endnote#1 - App-Specific Device Token

https://help.pushwoosh.com/hc/en-us/articles/360000364923-What-is-a-Device-token-

Question:

What is a Device token?

Answer:

Push token (device token) - is a unique key for the app-device combination which is issued by the Apple or Google push notification gateways. It allows gateways and push notification providers to route messages and ensure the notification is delivered only to the unique app-device combination for which it is intended.

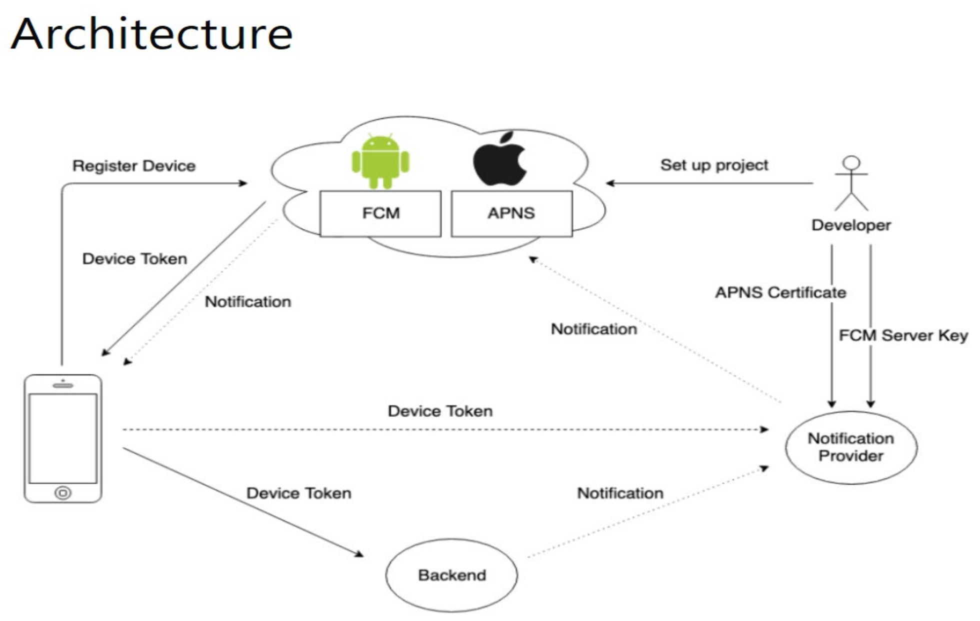
iOS device push tokens are strings with 64 hexadecimal symbols. Push token example:
03df25c845d460bcdad7802d2vf6fc1dfde97283bf75cc993eb6dca835ea2e2f
Make sure that iOS push tokens you use when targeting specific devices in your API requests are in lower case.

Android device push tokens can differ in length (usually below 255 characters), and usually start with APA…Push token example:

```
APA91bFoi3lMMre9G3XzR1LrF4ZT82_15MsMdEICogXSLB8-
MrdkRuRQFwNI5u8Dh0cI90ABD3BOKnxkEla8cGdisbDHl5cVIkZah5QUhSAxzx4Roa7b4xy9tvx9iNSYw-eXBYYd8k1XKf8Q_Qq1X9-
x-U-Y79vdPq
```

Note: The Android device push tokens correspond to the app-specific device token terminology used in the claim charts.

https://dev.to/jakubkoci/react-native-push-notifications-313i



Note: In the above Architecture, the "backend" corresponds to the backend of the Application Server. The Device token corresponds to a specific App running on a specific device. That is what is meant by the app-specific device token in the claim charts.

https://firebase.google.com/docs/cloud-messaging/android/first-message

COMMUNICATION INTERFACE TECHNOLOGIES LLC'S FIRST INFRINGEMENT ANALYSIS

U.S. Patent No. 8,291,010 – Defendant / CKE Restaurants Holdings, Inc.
Claim 17

Access the registration token

To send a message to a specific device, you need to know that device's registration token. Because you'll need to enter the token in a field in the Notifications console to complete this tutorial, make sure to copy the token or securely store it after you retrieve it.

On initial startup of your app, the FCM SDK generates a registration token for the client app instance. If you want to target single devices or create device groups, you'll need to access this token by extending FirebaseMessagingService and overriding on NewToken.

This section describes how to retrieve the token and how to monitor changes to the token. Because the token could be rotated after initial startup, you are strongly recommended to retrieve the latest updated registration token.

The registration token may change when:

- The app deletes Instance ID
- The app is restored on a new device
- The user uninstalls/reinstall the app
- The user clears app data."

https://firebase.google.com/docs/cloud-messaging/concept-options

For example, here is a JSON-formatted notification message in an IM app. The user can expect to see a message with the title "Portugal vs. Denmark" and the text "great match!" on the device:

```
{
 "message":{
   "token":"bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1...", ←-- App-specific token
   "notification":{
     "title":"Portugal vs. Denmark",
     "body":"great match!"
   }
 }
}
```

https://firebase.google.com/docs/cloud-messaging/android/client

Retrieve the current registration token

When you need to retrieve the current token, call FirebaseInstanceId.getInstance().getInstanceId():

```
FirebaseInstanceId.getInstance().getInstanceId()
        .addOnCompleteListener(new OnCompleteListener<InstanceIdResult>() {
            @Override
            public void onComplete(@NonNull Task<InstanceIdResult> task) {
                if (!task.isSuccessful()) {
                    Log.w(TAG, "getInstanceId failed", task.getException());
                    return;
                }

                // Get new Instance ID token
                String token = task.getResult().getToken();

                // Log and toast
                String msg = getString(R.string.msg_token_fmt, token);
                Log.d(TAG, msg);
                Toast.makeText(MainActivity.this, msg, Toast.LENGTH_SHORT).show();
            }
        });
```

MainActivity.java

Monitor token generation

The onNewToken callback fires whenever a new token is generated.

```
/**
 * Called if InstanceID token is updated. This may occur if the security of
 * the previous token had been compromised. Note that this is called when the InstanceID token
 * is initially generated so this is where you would retrieve the token.
 */
@Override
public void onNewToken(String token) {
   Log.d(TAG, "Refreshed token: " + token);

   // If you want to send messages to this application instance or
   // manage this apps subscriptions on the server side, send the
   // Instance ID token to your app server.
   sendRegistrationToServer(token);
}
```

After you've obtained the token, you can send it to your app server and store it using your preferred method. See the Instance ID API reference [https://firebase.google.com/docs/reference/android/com/google/firebase/iid/FirebaseInstanceId] for full detail on the API.

Endnote#2 - Transport Layer Security and Virtual Sessions

Transport Layer Security (TLS) is used by both Apple iOS and Android based devices. The handshake diagrams in this endnote use Apple iOS as an example but apply equally to Android type implementations.

https://developer.android.com/training/articles/security-ssl

"The Secure Sockets Layer (SSL)—now technically known as Transport Layer Security (TLS)—is a common building block for encrypted communications between clients and servers."

https://android-developers.googleblog.com/2018/04/protecting-users-with-tls-by-default-in.html

"Android is committed to keeping users, their devices, and their data safe. One of the ways that we keep data safe is by protecting all data that enters or leaves an Android device with Transport Layer Security (TLS) in transit.

A. Razaghpanah et al., *Studying TLS Usage in Android Apps,* CoNEXT '17, Dec.12-15, 2017, Incheon, Republic of Korea
http://abbas.rpanah.ir/publications/conext2017_tls_paper.pdf

"**A History of TLS Support in Android:** Android has supported TLS 1.0 since its first version released in 2008 and TLS 1.1 and TLS 1.2 since 2012."

"However, other protocols such as secure email (42 apps) and Google's Cloud Messaging service for push notifications (9 apps) [11, 47] also use TLS."

https://developer.ibm.com/customer-engagement/docs/watson-marketing/ibm-engage-2/tls-1-2-migration-for-mobile-push-clients/

What will happen on devices that are unable to support TLS 1.2?

Devices which do not support TLS 1.2 will be unable to connect to our WCA servers. This will prevent users of those devices from:

• Registering new mobile user IDs
• Updating push tokens
• Receiving inbox messages
• Receiving In-app messages

Note: As the above link shows, the creation of the App IDs of Endnote #1 are linked to the TLS protocol being run on the TLS-enabled Push-Notification channel.

https://tools.ietf.org/html/rfc5246

F.1.4. Resuming Sessions

When a connection is established by resuming a session, new ClientHello.random and ServerHello.random values are hashed with the session's master_secret. Provided that the master_secret has not been compromised and that the secure hash operations used to produce the encryption keys and MAC keys are secure, the connection should be secure and effectively independent from previous connections. Attackers cannot use known encryption keys or MAC secrets to compromise the master_secret without breaking the secure hash operations.

Sessions cannot be resumed unless both the client and server agree. If either party suspects that the session may have been compromised, or that certificates may have expired or been revoked, it should force a full handshake. An upper limit of 24 hours is suggested for session ID lifetimes, since an attacker who obtains a master_secret may be able to impersonate the compromised party until the corresponding session ID is retired. Applications that may be run in relatively insecure environments should not write session IDs to stable storage.

https://tools.ietf.org/html/rfc5077

Abstract

This document describes a mechanism that enables the Transport Layer Security (TLS) server to resume sessions and avoid keeping per-client session state. The TLS server encapsulates the session state into a ticket and forwards it to the client. The client can subsequently resume a session using the obtained ticket.

3. Protocol

This specification describes a mechanism to distribute encrypted session-state information in the form of a ticket. The ticket is created by a TLS server and sent to a TLS client. The TLS client presents the ticket to the TLS server to resume a session.

EXHIBIT 8

Seth W. Wiener, CSBN No. 203747
LAW OFFICES OF SETH W. WIENER
609 Karina Court
San Ramon, CA 94582
Telephone: (925) 487-5607
Facsimile (925) 828-8648
Email: seth@sethwienerlaw.com

Attorneys for Plaintiff
COMMUNICATION INTERFACE
TECHNOLOGIES LLC

UNITED STATES DISTRICT COURT

FOR THE CENTRAL DISTRICT OF CALIFORNIA

WESTERN DIVISION

COMMUNICATION INTERFACE TECHNOLOGIES LLC,

Plaintiff,

v.

CKE RESTAURANTS HOLDINGS, INC.,

Defendant.

Case No.:

DECLARATION OF ERIC MORGAN DOWLING IN SUPPORT OF COMPLAINT

I, Eric Dowling, declare as follows:

I. BACKGROUND INFORMATION

1. My name is Eric Morgan Dowling. I reside at Ave 30, Calle 154, Cond. Cartagena Real #4, Escazu, San Jose, Costa Rica.

2. I am currently an independent consultant.

3. This declaration is based on my own personal knowledge.

4. I am one of the named inventors of U.S. Patent Nos. 6,574,239 ("the '239 patent"), 8,266,296 ("the '296 patent"), and 8,291,010 ("the '010 patent"), (collectively the "patents in suit").

5. Each of the patents in suit has the same figures and the same substantive written description, although column and line number citations may differ slightly due to the U.S. Patent and Trademark Office's clerical data. Hence in this Declaration I will focus on discussing the '239 patent.

6. I understand that Communication Interface Technologies LLC's ("CIT" or "Plaintiff") intends to file a Complaint for infringement of the patents in suit against GENERIC DEFENDANT (LONG) ("DEF (SHORT)" or "Defendant").

7. I make this declaration in support of CIT's Complaint against Defendant in the above-captioned action.

II. QUALIFICATIONS AND EXPERIENCE

8. I hold three degrees from the University of Florida: (1) a Bachelor of Science degree in Electrical Engineering earned in 1984; and (2) a Master of Science degree in Electrical Engineering earned in 1986; and (3) a Doctor of Philosophy (Ph.D.) in Electrical Engineering earned in 1989.

9. From 2011–2018, I acted as President of Texas Trellis Phase, LLC, and general partner of Trellis Phase Communications, LP. As such, I was responsible for leading a research and development company developing next generation physical layer communication technologies for DSL, Cable Modems,

WiFi, Bluetooth, Digital CATV Transmission, Terrestrial HDTV Transmission, and Cellular Communications.

10. Since 1995, I have acted as a consultant in patent-related projects, providing consulting services related to, for example, patent infringement, noninfringement, and validity analysis, product/standards research, and litigation and cross licensing support. A partial list of these past clients include: Microsoft, AT&T, Tellular, Telesys, Harris Corp, Hughes, Thompson, GE, Raytheon, Paradyne, Samsung, J2 Communications, LSI Logic, ARC International, Enterprise Partners, Solar Products, EKMS, BTG, VSI, Core Enterprises, Baker and Botts; McKool Smith; Adduci, Manstriani and Schamuberg; Knobbe Martens Olson & Bear; Fulbright and Jarwoski; Merchant and Gould; Weil Gotshal, and Manges; Lerner and Greenberg; Klarquist Sparkman; Workman Nydegger; Gazdzinski and Associates; Kaplan & Gilman; Hamman and Benn; Marshal, Denehey, Warner, Coleman & Goggin; Carstens, Yee & Cahoon, others not disclosed due to NDAs.

11. From 1989 to 2001, I was employed as a professor at the University of Texas, Dallas. First, from 1989-1995, I served as an assistant professor of Electrical Engineering. Then, from 1995-2001, I served as an associate professor of Electrical Engineering, with tenure.

12. In addition, from 1985-2001, I acted as a consultant in electrical engineering projects, focusing on design. For example, some clients during these years include: DGI Technologies Inc, focusing on DSP Microprocessor based Real Time T1 Network Echo Canceller; Micron Technology Inc., wherein I consulted on the design of processors and embedded DRAM systems; Pinpoint communications Inc., focusing on radio direction finding for RF identification; DTex Inc., focusing on design of algorithms, DSP processing board, and DSP56L811 DSP software to implement an advanced ultra-sensitive metal detector; Raytheon / E-Systems, focusing on design of Digital Signal Processing

Software for a Military Mobile Telecommunications Switch; INET Corp., focusing on Design of Equalizers for T-Carrier Wireline Transmission Systems; US Naval Undersea Warfare Center, focusing on Adaptive Sonar Signal Processing Software; Ultimate Technologies Inc., focusing on Research and Design of Active Noise Control Systems; Alcatel Network Systems, focusing on Microprocessor Systems Design - On Site Training; Nortel, focusing on Digital Signal Processing - On Site Training; University of Texas Southwestern Medical Center, focusing on design of algorithms for Biomedical Signal Processing, Electrospace Systems Inc., focusing on design of DSP algorithms for noise cancellation; and Athena Group, Inc., focusing on NSF SBIR Contracting and DSP Software.

III. MATERIALS CONSIDERED

13. I have considered the materials listed in this declaration as well as the ’239 patent, the ’296 patent, and the ’010 patent, and their prosecution histories, as well as related patents.

14. I am being compensated at my regular and established rate of $250 per hour. My compensation is not dependent on the outcome of this proceeding.

IV. BACKGROUND OF THE INVENTIONS OF THE PATENTS IN SUIT

15. In the late 1990s the most widely implemented technology in client-server communications involved sessions that could be instantiated (started), and then torn down. If communications between a client and server were needed after the previous session had been torn down, the widely implemented technology as that time would have been to instantiate a brand-new session between the same client and server. Creating a new session required the renegotiation of a set of session keys that included computation of new cryptographic keys. This process required significant start up times and computational resources.

16. My business associate and co-inventor, Mark Anastasi, and I developed the inventions that resulted in the ’239, ’296, and the ’010 patents to shorten the connection time of the dialup modems in use in the 1990s. Each time a

new dialup modem connection needed to be reestablished, there would be a several-second period (typically around 10-12 seconds) during which the user would hear audio modem tones and hissing sounds while the modems reconnected and negotiated a new data session. Our inventions allowed the modems to reconnect by remembering the previously negotiated modem parameters, thereby greatly shortening this renegotiation time to being almost unnoticeable. ('239 patent at 13:42-43, 17:50-58.)

17. While developing these inventions, we also contemplated virtual sessions would be very useful in wireless applications to allow a client-side remote unit to maintain a virtual presence with a remote server. For example, we taught in the '239 patent that virtual sessions could be layered over wireless connections to allow remote units such as wireless Internet devices to be virtually connected to one or more server-side application programs, which were running on one or more remote server systems. ('239 patent at 9:28-60.) This avoided wasting wireless physical layer resources to maintain the session layer connections by allowing the physical layer to be inactive, while the virtual session layer connections could be maintained without using wireless resources. ('239 patent at 3:45-49, 8:56-58, 9:7-10.) When the client-side remote unit needed to communicate with the server, or when the server needed to send newly received information to the remote unit, the virtual session could be reactivated without the need to tediously set up and authenticate a new secure cryptographic session with the server. ('239 patent at Fig. 1A, 9:53-60, 13:48-14:17.)

18. The virtual sessions in the '239, '296, and '010 patents are disclosed with reference to the definition of "session" provided in the open systems interconnect ("OSI") reference model from the International Standards Organization. ('239 patent at 8:17-21.) The OSI model is "a model of a layered software structure used in computer communications" and a software system implemented such a layered model is known as a "protocol stack." ('239 patent at

8:21-24.) The various layers in a representative protocol stack are used in the invention are shown in Figure 1A of the ’239 patent. This protocol stack includes the application session layer, the virtual session layer, and the physical layer, among others. (’239 patent at Fig. 1A.) The physical layer represents a physical layer communication connection such as a cellular wireless or network connection. (’239 patent at 9:32-35.) The application session layer involves communications between the application software on the client-side and the server-side. (’239 patent at 8:35-45.) The virtual session layer links the various application sessions. (’239 patent at 8:61-9:3.) While OSI was previously known, the ’239, ’296, and ’010 patents include novel methods to resume the virtual sessions for specific application programs through the use of application-program identifying packets (which can be embedded into a larger communication packet).

19. Claim 7 of the ’239 patent, for example, recites a particular improvement in the relevant technology, because the signal sent from the server contains an application-program identifying packet. This application-program identifying packet allows the virtual session to resume at the application session layer of the protocol stack (as discussed above) for a particular application program. Claim 7 refers to the relevant program as “at least one application layer program.” The application layer program of claim 7 operates at the application session layer of the protocol stack in Figure 1A. Resuming a disconnected communication session between the server and the remote unit for a particular application program at the application session layer was not known in the prior art. Therefore, the inventions resulted in a more efficient use of computational resources and provide for a more seamless client-server experience over extended periods of usage.

20. The ’239, ’296, and ’010 patents also disclose how these particular technological improvements can be implemented. For example, the ’239 patent discloses that a virtual session server maintains a table linking one or more

application sessions to a virtual session. ('239 patent at 8:61-63.) Once an application session has been established over a virtual session, the table can be used to maintain a set of parameters to maintain the application session even without a physical layer connection. ('239 patent at 10:57-11:5.) The '239 patent discloses multiple ways that the table structure can link application sessions such as user identification parameters, a user password, a set of application session parameters, a communication session identification parameter and an encryption key. ('239 patent at 11:17-23.) In contrast, the prior art required all the different sessions to be tediously and manually established and torn down each time they were separately needed. Therefore, the inventions created a more seamless experience for a consumer during extended periods of usage of the communication sessions.

V. IMPROVEMENTS OF THE '239, '296, AND THE '010 PATENTS

21. The '239, '296, and the '010 patents resolve technical problems related to client-server computing architecture. The claims of the '239 Patent, the '296 Patent, and the '010 Patent represent a major departure from conventional methods of client-server communications, which relied on instantiating a brand-new session between the client and server for each communication session. Resuming a virtual session at the application-session layer as well as at the physical layer represented an improvement in computer capabilities with respect to client-server communications technology. Each of these benefits is grounded in computerized improvements. For example, the '239 patent explains how virtual sessions supporting at least one application session layer program can be resumed through the sending of application identifying information. ('239 patent at 10:32-33.) This allows the virtual session server to maintain a plurality of virtual sessions with remote units for specific application programs through the use of a table structure. ('239 patent at 8:63-9:1.) The '239 patent further demonstrates how the reassociation involving the application identifying information is

performed such as through the use of communication session parameters and application session parameters. (’239 patent at 10:7-8; 11:13-23; 20:45-52; 22:13-21.) Figures 5-8 demonstrate specific algorithms by which these processes are performed. (’239 patent at Figs. 5-8; 20:19-25:14.)

22. The claims of the ’239, ’296, and the ’010 patents recite inventive concepts rooted in computerized client-server computing. The ’239 Patent, the ’296 Patent, and the ’010 Patent are not directed to any generic device to implement conventional functionality. On the contrary, the specific manner of sending application-program identifying packets to resume virtual sessions between a remote unit and servers at the application session layer of the protocol stack was unconventional. For example, claim 7 of the ’239 patent recites how a server controls a virtual session between a server and a remote unit by sending a signal with particular characteristics such as “sending a signal indicative of . . . an application-program identifying packet . . . said application-program identifying packet identifying an application program that needs to resume a virtual session.” (’239 patent at 25:52-56.) This method represents an improvement in computer capabilities with respect to client-server communications technology.

23. Claim 7 of the ’239 patent includes the following features:

- Establishing a virtual session with a remote unit, the virtual session being instantiated to support at least one application layer program;
- Sending a signal indicative of an incoming communication request and an application-program identifying packet to said remote unit, said application-program identifying packet identifying an application program that needs to resume a virtual session and communicate with said remote unit; and
- Placing the virtual session back into the active state and transferring data between the application and the remote unit via the virtual session in response to said step of sending.

('239 patent at 26:47-61.) Claim 1 of the '296 patent and claims 1 and 17 of the '010 patent include similar features as those recited above from claim 7 of the '239 patent.

24. The features from claim 7 of the '239 patent set forth above are unconventional in that they represent a major departure from the then contemporaneous conventional methods of client-server communications. The '239, '296 and '010 patents claim the inventive concept of sending a particular signal from the server with an application-program identifying packet to allow a specific virtual session associated with a specific one of a plurality of different possible application programs to be resumed at the application session layer. This was unconventional because it allowed a virtual session to resume communication with a specific one of a plurality of different possible application programs at the application session layer, and to do so in an efficient manner. The prior art did not disclose the use of sending an application-program identifying packet to identify which application program should resume its application-session layer communication channel back to the server who sent the application program identifying packet to that particular application program.

25. A benefit of the system described and claimed in the '239, '296 and '010 patents is that it enables users such as remote workers to stay connected to one or more central servers without the need to continuously remain connected by the physical layer of the protocol stack. This allows users of remote devices to be reconnected to a virtual session without the need for long and tedious reestablishment procedures. This also allows a secure connection to be maintained during long periods of inactivity. The patents explain how to accomplish resuming a connection according to public key cryptography techniques with for example a logon session, authorization sequence, authentication parameters, encryption keys, digital signatures, or use of a table structure for maintaining session parameters such as a password. ('239 patent at 3:2-5, 3:55-60, 4:22-25, 8:45-53, 10:2-15,

10:51-55, 10:57-62, 11:15-21, 14:32-33, 18:61-66, 20:40-43, 20:50-55, 21-49-55, 22:1-7; Figs. 6, 7.)

26. Each of these benefits is grounded in computerized improvements. The use of a signal including an application-program identifying packet to identify an application program that needs to resume a virtual session at the application session layer, allows for a much shorter renegotiation sequence through the use of saved session parameters. Prior systems used separate sessions for each communication and did not have the capability for reactivation of a previous virtual session at the application session layer after the session had been deactivated. Prior art systems therefore did not rely on an application-program identifying packet sent from the server to place a specific virtual session back into an active state.

***

I declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct. Executed in Escazu, Costa Rica on September 27, 2022.

Respectfully submitted,

[illegible]

Eric Dowling