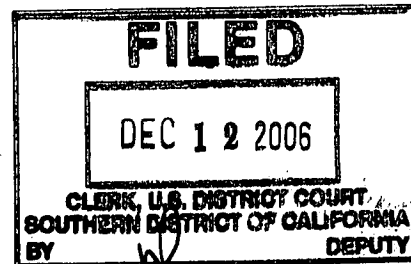


ORIGINAL

1 John E. Gartman (SBN 152300)
Christopher S. Marchese (SBN 170239)
2 FISH & RICHARDSON P.C.
12390 El Camino Real
3 San Diego, California 92130
Telephone: (858) 678-5070
4 Facsimile: (858) 678-5099

5 Stephen P. McGrath (SBN 202696)
One Microsoft Way
6 Redmond, WA 98052
Telephone: (425) 882-8080
7 Facsimile: (425) 936-7329

8 Attorneys for Plaintiff
MICROSOFT CORPORATION



9
10 UNITED STATES DISTRICT COURT
11 SOUTHERN DISTRICT OF CALIFORNIA

12 MICROSOFT CORPORATION,

13 Plaintiff,

14 v.

15 ALCATEL-LUCENT,

16 Defendant.

Case No. '06CV 2696

LAB CAB

17
18 MICROSOFT'S COMPLAINT
AGAINST ALCATEL-LUCENT FOR
PATENT INFRINGEMENT

JURY TRIAL DEMANDED

19
20 COMPLAINT

21 This is Microsoft's complaint against Alcatel-Lucent for infringing ten Microsoft patents.

22 PARTIES

23 1. Microsoft Corporation ("Microsoft") is a Delaware Corporation with its principal
24 place of business at One Microsoft Way, Redmond, WA 98052.

25 2. Alcatel-Lucent is a French corporation with its principal place of business at 54, rue
26 La Boetie, 75008, Paris, France.

JURISDICTION AND VENUE

3. This patent infringement case arises under Title 35 of the United States Code. This Court has subject matter jurisdiction under 28 U.S.C. §§ 1331 and 1338(a).

4. This Court has personal jurisdiction over Alcatel-Lucent. Alcatel-Lucent has done and continues to do business in California and in this district. Through intermediaries or by itself, Alcatel-Lucent imports, makes, uses, sells, offers to sell, ships, distributes, and/or advertises its products and services in California and in this district.

5. Venue is proper under 28 U.S.C. §§ 1391 and 1400.

PATENT INFRINGEMENT

6. Microsoft owns and has always owned each of the following lawfully issued U.S. Patents: 6,438,217; 6,339,794; 5,438,433; 5,941,947; 6,412,004; 5,838,319; 5,977,971; 6,565,608; 5,764,913; and 5,917,499. A copy of each patent is attached.

7. Alcatel-Lucent has infringed and continues to infringe each patent, either directly, or indirectly by inducement or contribution.

8. Alcatel-Lucent has known about the patents and its infringement is willful. This case is an exceptional case that warrants treble damages and attorneys' fees under 35 U.S.C. § 285.

9. Alcatel-Lucent's infringement has damaged Microsoft.

10. Alcatel-Lucent will continue to infringe, causing irreparable harm to Microsoft for which there is no adequate remedy at law, unless preliminarily and permanently enjoined.

PRAYER FOR RELIEF

Microsoft requests the following relief:

A. A judgment that Alcatel-Lucent has infringed each patent, and requiring Alcatel-Lucent to pay damages, including treble damages, pre- and post-judgment interest, and all the costs of this action and attorneys' fees as provided by 35 U.S.C. § 285,

1 B. A judgment that Alcatel-Lucent, its officers, agents, servants, directors, employees,
2 subsidiaries, parents, attorneys, and those persons acting in active concert, on behalf of, in joint
3 venture, or in participation with Alcatel-Lucent, be preliminarily and permanently enjoined from
4 further infringement, and

5 C. Such further relief as the Court deems just and equitable.

6 **JURY DEMAND**

7 Under Local Rule CV-38.1 and Fed. R. Civ. P. 38, Microsoft demands a trial by jury.

8
9 Dated: December 12, 2006

Respectfully submitted,

10 FISH & RICHARDSON P.C.

11
12 By: 

13 John E. Gartman (SBN 152300)
14 FISH & RICHARDSON P.C.

15 Attorneys for Plaintiff
16 MICROSOFT CORPORATION

17
18
19
20
21
22
23
24
25
26
27
28
10691263.doc



US006438217B1

(12) **United States Patent**
Huna

(10) Patent No.: **US 6,438,217 B1**
(45) Date of Patent: **Aug. 20, 2002**

(54) **APPARATUS AND METHOD FOR FUTURE TRANSMISSION OF DEVICE-INDEPENDENT MESSAGES**

(75) Inventor: **Emmanuel L. Huna, Daly City, CA (US)**

(73) Assignee: **Microsoft Corporation, Redmond, WA (US)**

(*) 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.: **09/268,524**

(22) Filed: **Mar. 11, 1999**

(51) Int. Cl.⁷ **H04M 1/64**

(52) U.S. Cl. **379/88.14; 379/88.23**

(58) Field of Search **379/67.1, 88.11, 379/88.12, 88.13, 88.14, 88.17, 88.18, 88.22, 88.23, 90.01, 93.01, 93.15, 93.24; 709/217, 218, 219**

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,608,786 A	3/1997	Gordon	
5,630,060 A	5/1997	Tang et al.	
5,646,982 A	7/1997	Hogan et al.	
5,652,789 A *	7/1997	Miner et al.	379/201
5,675,507 A *	10/1997	Bobo, II	364/514
5,742,905 A	4/1998	Pepe et al.	
5,870,454 A *	2/1999	Dahlen	379/88.14
5,872,926 A *	2/1999	Levac et al.	379/93.15 X
5,951,638 A *	9/1999	Hoss et al.	709/206
5,987,100 A *	11/1999	Fortman et al.	379/88.14
6,055,240 A *	4/2000	Tunncliffe	370/428

6,134,454 A *	10/2000	Foladare et al.	455/556
6,157,924 A *	12/2000	Austin	707/10
6,181,781 B1 *	1/2001	Porter et al.	379/88.17
6,203,192 B1 *	3/2001	Fortman	379/88.14
6,233,318 B1 *	5/2001	Picard et al.	379/88.17

FOREIGN PATENT DOCUMENTS

EP 0854655 A2 7/1998

* cited by examiner

Primary Examiner—Scott L. Weaver

(74) Attorney, Agent, or Firm—Senniger, Powers, Leavitt & Roedel

(57) **ABSTRACT**

An apparatus and method are provided for entering and transmitting a message at a future delivery time to a receiving device that is coupled either to a telephony-centric network or to a data-centric network. The apparatus includes a message server, a data-centric network server, and a telephony-centric network server. The message server translates the message into a format compatible with the receiving device and initiates delivery of the message at the future delivery time. The data-centric network server transmits the message over the data-centric network and, if the receiving device is addressable over the data-centric network, then said data-centric network server delivers the message directly to the receiving device. The telephony-centric network server provides an interface between the data-centric network server and the telephony-centric network. If the receiving device is addressable by the telephony-centric network, then the telephony-centric network server receives the message from said data-centric network server and delivers the message to the receiving device over the telephony-centric network.

59 Claims, 8 Drawing Sheets

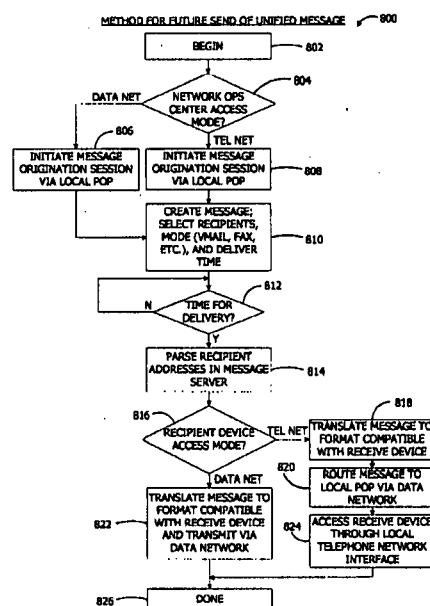
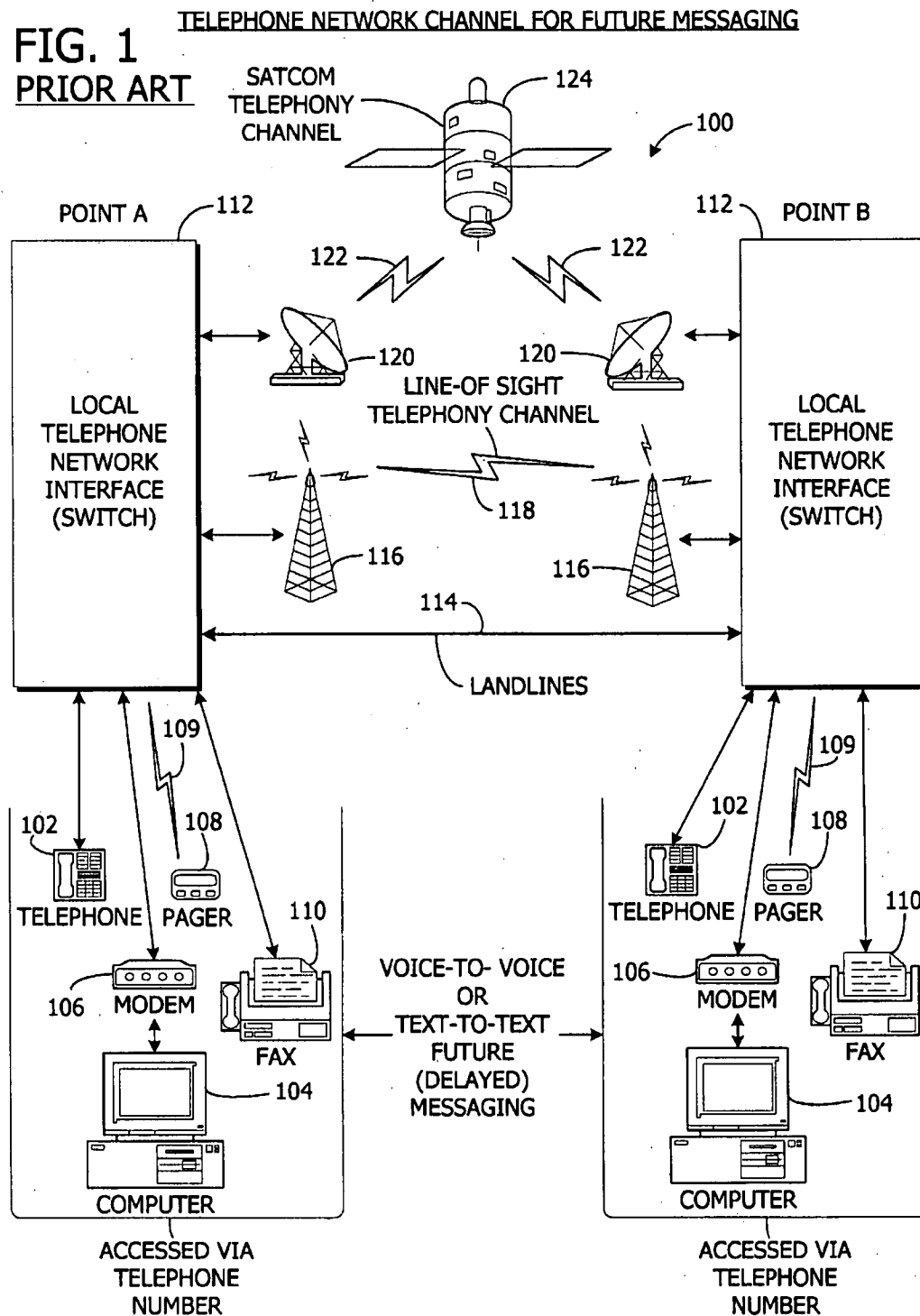


FIG. 1
PRIOR ART

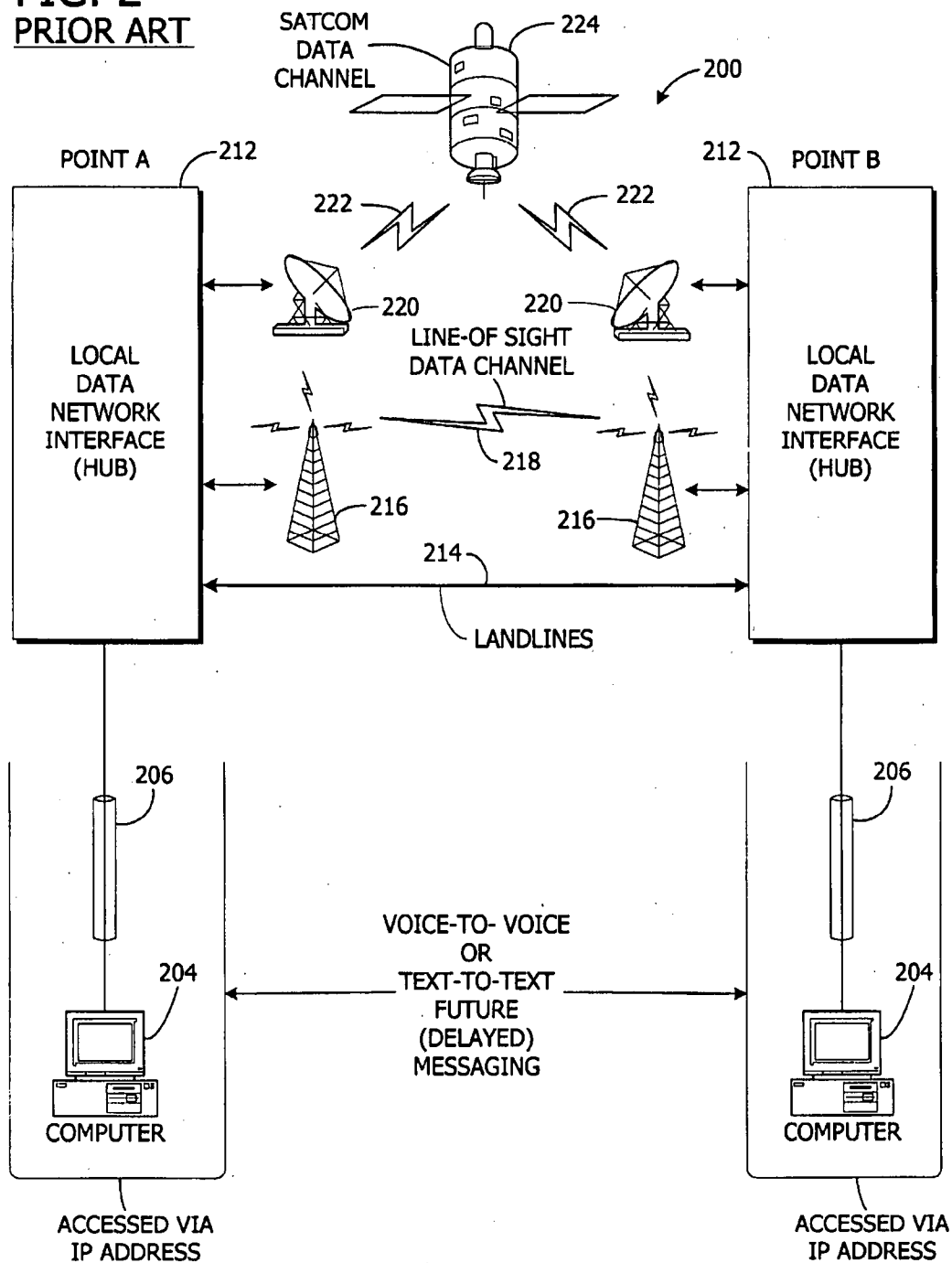


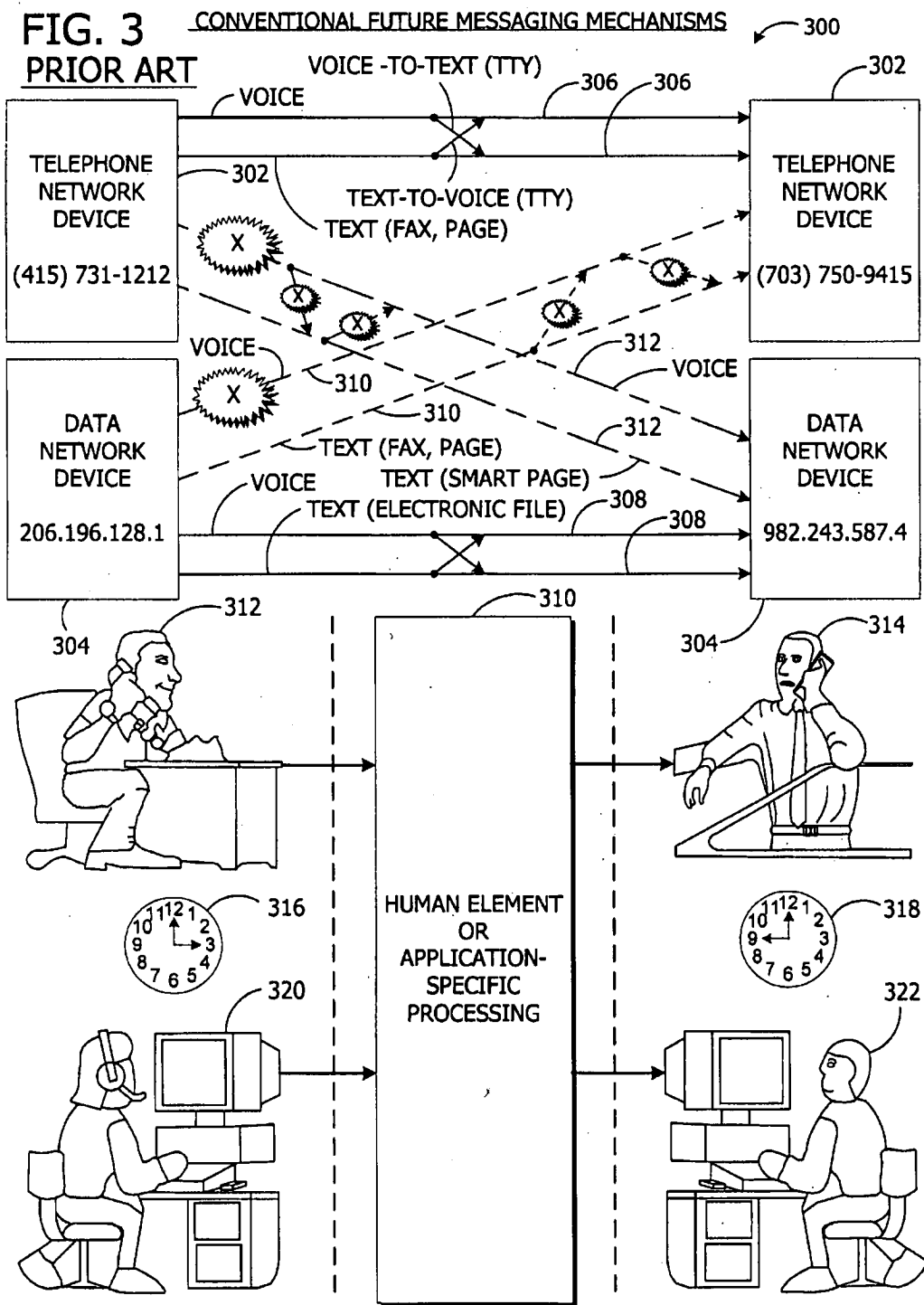
U.S. Patent

Aug. 20, 2002

Sheet 2 of 8

US 6,438,217 B1

FIG. 2
PRIOR ART**DATA NETWORK CHANNEL FOR FUTURE MESSAGING**



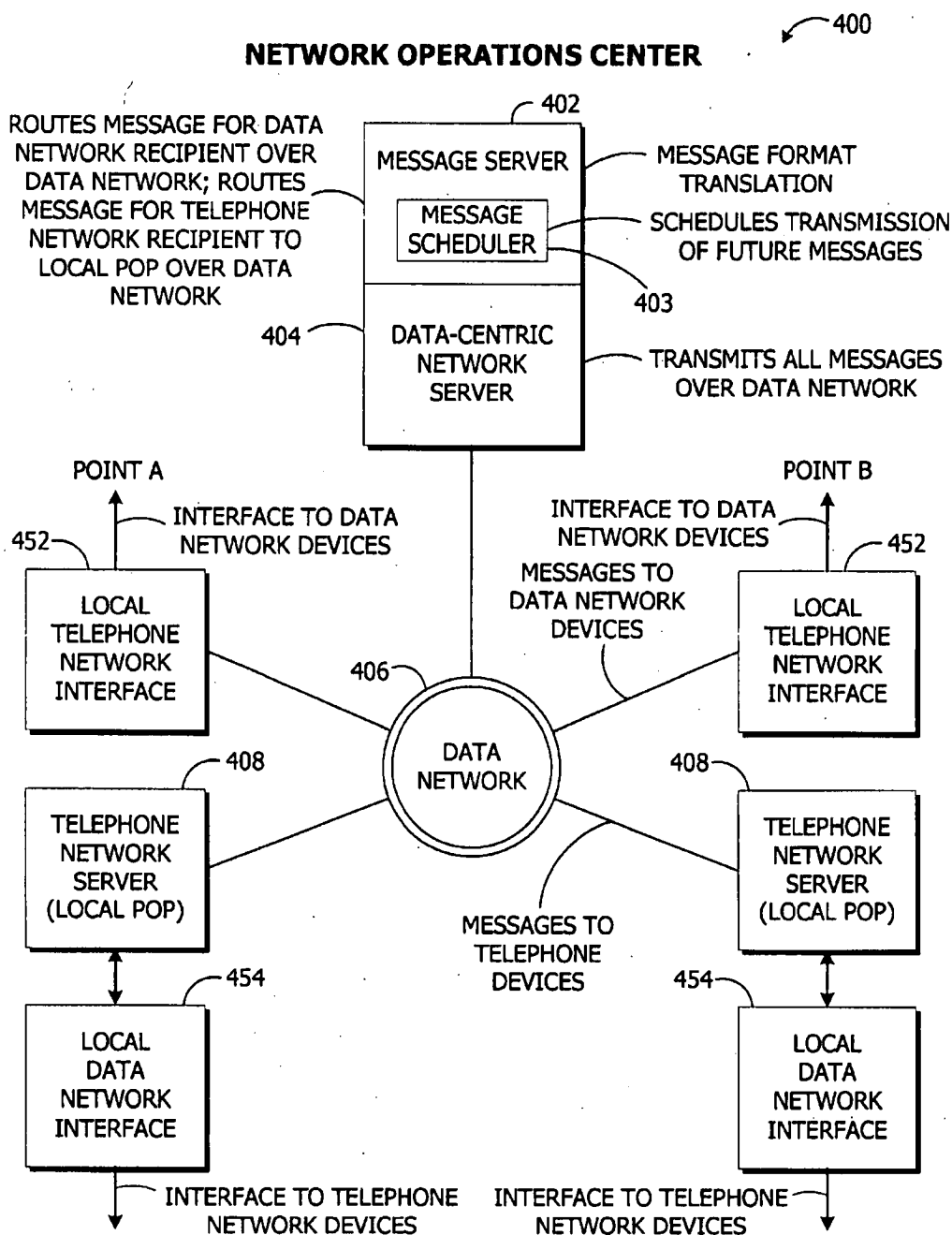
U.S. Patent

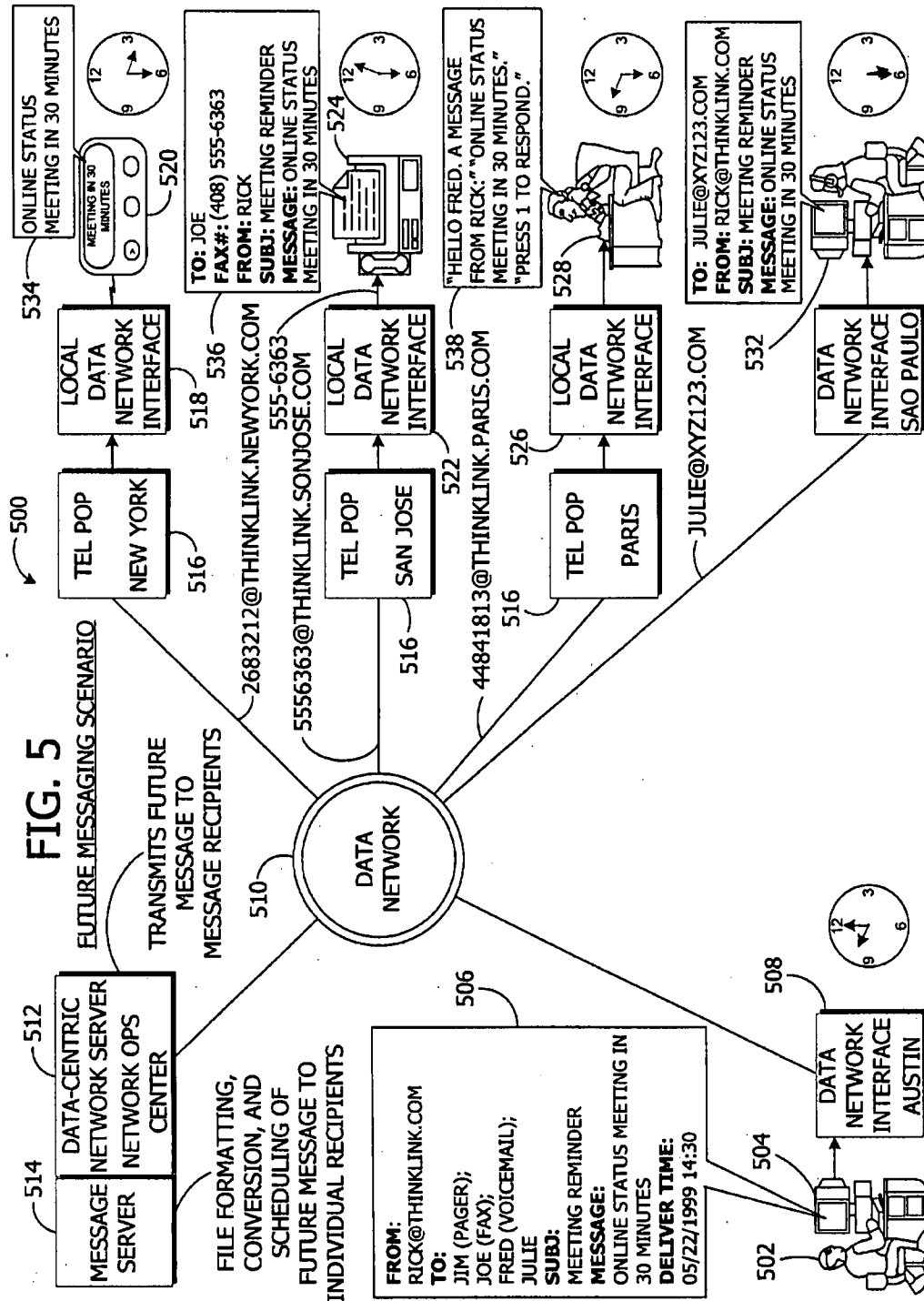
Aug. 20, 2002

Sheet 4 of 8

US 6,438,217 B1

FIG. 4

CHANNEL-TRANSPARENT FUTURE MESSAGING**NETWORK OPERATIONS CENTER**



U.S. Patent

Aug. 20, 2002

Sheet 6 of 8

US 6,438,217 B1

FIG. 6

UNIFIED MESSAGE ENTRY WINDOW FOR FUTURE TRANSMISSION

600

602

HEADER

610

COMPOSE EMAIL, FAX OR VOICEMAIL

612

SEND SAVE DRAFT

PRIORITY NORMAL CANCEL

TO: JIM (PAGE), JOE (FAX), FRED (VOICEMAIL), JULIE

USE A COMMA BETWEEN ADDRESSES.
TO SEND A FAX: 123-456-7890 (FAX)
OR VOICEMAIL: 123-456-7890 (VOICEMAIL)

SUBJECT: STATUS MEETING REMINDER

Cc: BOSS (EMAIL)

Bcc:

ADDRESS BOOK

FOUR11 DIRECTORY

HELLO FOLKS:

614

THIS IS AN AUTOMATED MESSAGE TO REMIND YOU THAT WE HAVE SCHEDULED AN ONLINE PROJECT STATUS MEETING IN 30 MINUTES.

AGENDA IS ATTACHED. PLEASE BE PREPARED TO DISCUSS.

RICK

616

DELIVER TIME: 05/22/1999 14:30 (mm/dd/yyyy hh:mm)

PLAIN TEXT HTML PREVIEW SAVE A COPY OF OUTGOING MESSAGE USE SIGNATURE

MENU

HOME

COMPOSE

FOLDERS

ADDRESSES

OPTIONS

MY

ACCOUNT

HELP DESK

LOG OUT

U.S. Patent

Aug. 20, 2002

Sheet 7 of 8

US 6,438,217 B1

FIG. 7

CONFIGURATION FOR A WAKE UP CALL

MENU

HOME

COMPOSE

FOLDERS

ADDRESSES

OPTIONS

MY ACCOUNT

HELP DESK

LOG OUT

COMPOSE EMAIL, FAX OR VOICEMAIL

710

HELP

SEND

SAVE DRAFT

712

TO: 713-555-1212

SUBJECT: WAKE UP CALL

Cc:

Bcc:

PRIORITY

NORMAL

CANCEL

USE A COMMA BETWEEN ADDRESSES.

TO SEND A FAX: 123-456-7890 (FAX)

OR VOICEMAIL: 123-456-7890 (VOICEMAIL)

ADDRESS BOOK

FOUR11 DIRECTORY

714

GOOD MORNING RICHARD. TIME TO WAKE UP. ARE YOU AWAKE?
WAKE UP. WAKE UP. WAKE UP. WAKE UP. WAKE UP.

REMEMBER THAT YOU HAVE A CLIENT MEETING AT THE HOTEL IN TWO HOURS.

HAVE A GOOD DAY.

☒ PLAIN TEXT
 ☐ HTML PREVIEW

☒ SAVE A COPY OF OUTGOING MESSAGE

☒ USE SIGNATURE

DELIVER TIME: 05/23/1999 06:00

(mm/dd/yyyy hh:mm)

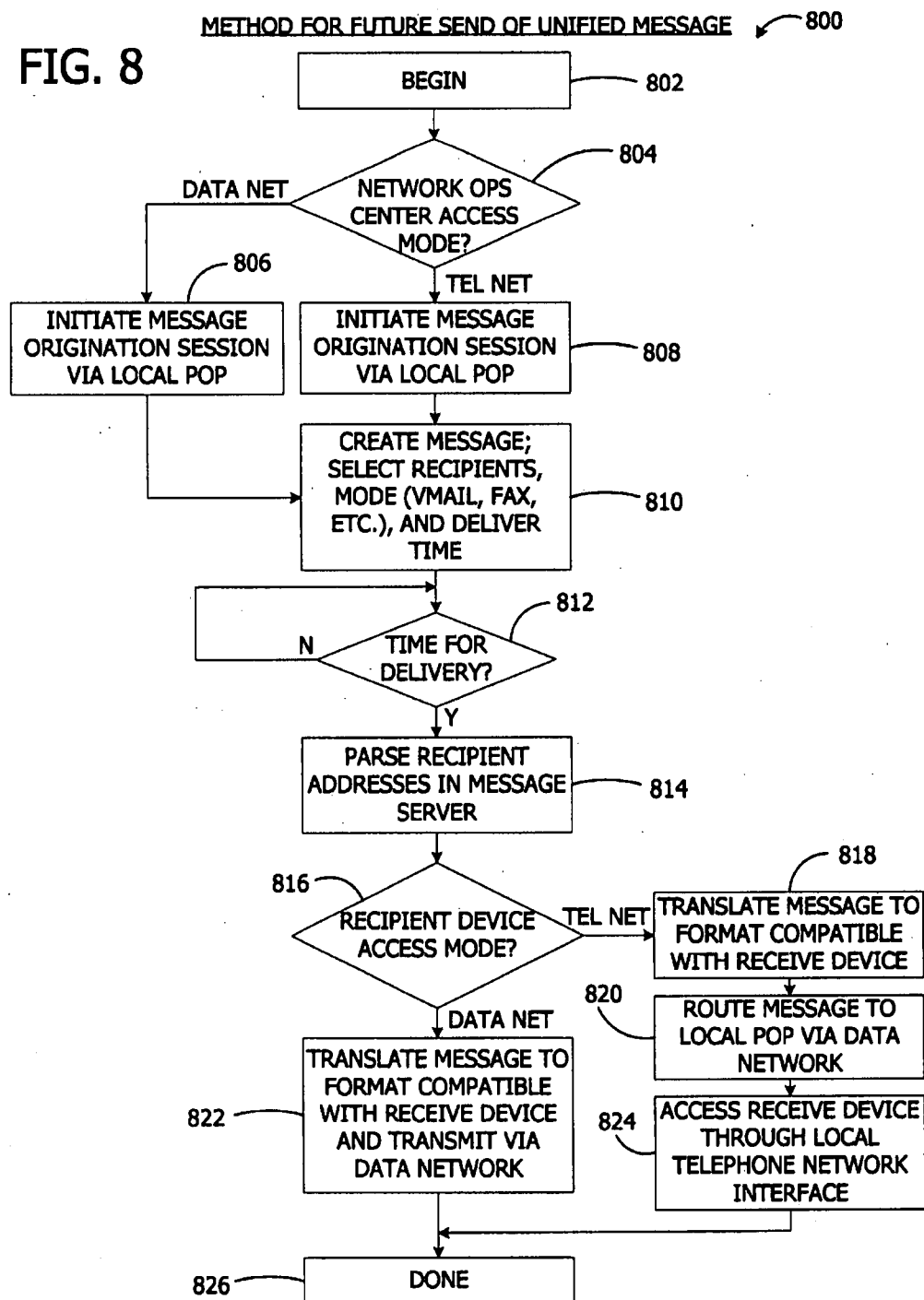
U.S. Patent

Aug. 20, 2002

Sheet 8 of 8

US 6,438,217 B1

FIG. 8



US 6,438,217 B1

1

APPARATUS AND METHOD FOR FUTURE TRANSMISSION OF DEVICE-INDEPENDENT MESSAGES

This application is related to the following co-pending U.S. Patent Applications which are hereby incorporated by reference:

Application No.	Filing Date	Title
09/239,560	1/29/99	Integrated Message Storage And Retrieval System Distributed Over A Large Geographical Area
09/240,367	1/29/99	A System And Method For Providing Unified Messaging To A User With A Thin Web Browser
09/239,585	1/29/99	Centralized Communication Control Center For Visually And Audibly Updating Communication Options Associated With Communication Services Of A Unified Messaging System And Methods Therefor
09/239,584	1/29/99	Computer-Implemented Call Forwarding Options And Methods Therefor In A Unified Messaging System
09/240,893	1/29/99	Interactive Billing System Utilizing A Thin Web Client Interface
09/240,368	1/29/99	System And Method To Manage Phone Sourced Messages Using A User Modifiable Field Associated With The Message
09/240,434	1/29/99	Method And Apparatus For Network Independent Initiation Of Telephony
09/240,435	1/29/99	Apparatus And Method For Device Independent Messaging Notification
09/240,436	1/29/99	Apparatus And Method For Channel-Transparent Multimedia Broadcast Messaging
09/239,589	1/29/99	Voice Access Through A Data-Centric Network To An Integrated Message Storage And Retrieval System

DEFINITION OF TERMS

Data-centric network: a network that carries digital data, primarily to facilitate information exchange among computers and computer peripherals. Examples include distributed computer networks such as the Internet.

Telephony-centric network: a network that carries telephony information such as voice, fax, page messages, and the like, primarily to facilitate information exchange among telephony devices.

Message: a communication which may be transmitted via either the data-centric network or the telephony-centric network. Examples include voicemail, electronic mail (email), facsimile (fax), page, and the like.

Telecommunication device: POTS telephone, cellular telephone, satellite telephone, web telephone, PC (desktop and laptop), web surfer, personal digital assistant (PDAs), facsimile machine, teletype, modem, video telephone, set top telephone.

Web telephone: a telephone implemented via a computer that is coupled to the data-centric network. An example is a PC with microphone, speaker and internet connection.

Set top telephone: a telephone set coupled to a cable-based set top box, bypassing the local telco provider. The cable-based system may be provided by, for example, WebTV, TCI cablevision.

Web surfer: an Internet-ready PC with a network connection and pre-installed web browser.

2

PDA: personal digital assistant, e.g., Palm Pilot available from 3 COM.

Thin Web Client: A commonly employed web browser such as Internet Explorer or Netscape Navigator—JAVA enabled.

PSIN: Public Service Telephony-centric network, e.g., AT&T, MCI, Sprint-owned telco.

GUI: graphic user interface

POTS: plain old telephone service

NOC: Network Operations Center

POP: point of presence, e.g., co-location at a local telco switch or at a company controlled area with T1 connections to a local switch.

WPOP: Web POP

VPOP: Voice POP

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates in general to the field of messaging, and more particularly to an apparatus and method for utilizing a data-centric network to deliver a message to a receiving device at a scheduled time, where the receiving device is coupled either to the data-centric network or to a telephony-centric network.

2. Description of the Related Art

The advent of the telephone at the turn of the century presented a new form of communication to the population at large. Whereas prior messages were delivered directly to a message recipient either by mail, telegraph, or personal courier, telephones introduced a new option for delivering a message. More urgent matters were treated immediately with a telephone message while less important matters were relegated to the mail. The option to call or write notwithstanding, a message originator still knew that the message itself was being delivered to a particular geographic location, presumably in the presence of a message recipient. One of the most elementary systems for delivering a message was known as a messaging service. To utilize such a service, the message originator selected message recipients and either wrote or dictated a message to an operator who, in turn, distributed the broadcast message serially to each message recipient over the telephony-centric network. If the nature of the message required that it be delivered at a particular time, then the operator simply held the message until it was time to deliver it.

The advent of automated voicemail systems provided the means to take the human element out of the loop. Using a voicemail system, a message originator could record a message from his/her telephone and subsequently enter the telephone numbers of recipients for the message. More advanced voice mail systems provided to capability to specify a delivery time for the message.

Facsimile (fax) machines expanded the messaging options for a user. Now with a machine connected to the telephony-centric network, the user could convert a written page into electrical signals to be transmitted to a like machine over the telephony-centric network. The like machine would then translated the electrical signals back to text form and print out the transmitted page, thus delivering a textual message in a textual format. Higher end fax machines allow the user to prescribe a delivery time for an entered fax, which is a very useful feature since long distance rates are typically lower during the night. By delaying transmission of faxes until nighttime, a user can save money.

Though perhaps the most revolutionary series of events to advance the art of messaging has been the development of

US 6,438,217 B1

3

computer networking technologies resulting in what is now known as the Internet, along with related audio recording, storage, and transmission techniques. Today a user can access the Internet from virtually anywhere in the world and retrieve electronic mail (email) in text form or in voice form. Delayed messaging, or future messaging, can be implemented on a desktop or laptop computer by purchasing special-purpose software that allows the user to additionally prescribe a delivery time for a created message. Delayed email messages are extremely useful tools for scheduling a task management applications.

Cellular phones and pagers also provide a user with the ability to send and receive messages from other than a fixed location. Cellular phone and pager technologies are on the verge of providing worldwide coverage. It will soon be possible to call or page a message recipient anywhere in the world.

But in spite of the above noted advances, several problems still exist. A first problem relates to restricted distribution of a message. More specifically, a message that is entered from a device connected to the telephony-centric network, i.e., a device having an assigned telephone number, is restricted for delivery to devices that are also connected to the telephony-centric network. A device with a telephone number is designed to distribute messages similar devices having telephone numbers.

Likewise, a message that is entered from a device connected to the Internet or similar data-centric network, i.e., a device having an assigned Internet Protocol (IP) address, is restricted for delivery to devices that are also connected to the Internet. A device with an IP address is designed to distribute messages to similar devices having IP addresses.

A second problem is that future messaging features are not viable from a cost standpoint for the average consumer. A sole proprietor or small business entrepreneur is most often not in the position to retain a messaging service or to acquire high end capitol equipment, for the sole purpose of obtaining future messaging capabilities. He/She chooses rather to live without the capability and depend upon whatever capabilities exist in the competitive marketplace. Cost-competitive voice mail systems do not provide future messaging capabilities. To obtain delayed fax and email capabilities he/she is required to purchase special-purpose hardware and/or software.

A third problem regards the format translation for messages. Since techniques are now available to transmit both voice and text messages over both the telephony-centric network and the Internet, it is essential that messages for a particular recipient be provided in a format that is compatible with the particular recipient's receiving device. For example, a computer can function as a facsimile machine, but to provide fax capability on the computer, special-purpose application software is required to translate facsimile format to a format that can be viewed on a computer monitor.

Consequently, if a user today desires to send a message at a specified delivery time to recipients, the user must enter, schedule, and transmit the message on an originating device that is compatible with the receiving device, that possesses future messaging features, and that is part of the same network (i.e., telephony-centric network or data-centric network) as the receiving device. The situation is exacerbated when the message has multiple recipients. A first message must be broadcast to recipients on the telephony-centric network and a second message must be broadcast to recipients over the Internet.

4

Therefore, what is needed is an apparatus for sending a message to a receiving device at a future delivery time, where the message format and transmission network are transparent to the message originator.

In addition, what is needed an apparatus providing the capability to enter a future message in email format and to have the message delivered to a telephone in voicemail format.

Furthermore, what is needed is an apparatus for transmitting a future message to a number of recipients having receiving devices that are addressable over both the telephony-centric network and a data-centric network.

Moreover, what is needed is a method for transmitting a future message that permits a message originator to specify a delivery time and recipients, where the recipients can be addressed by a telephone number or a data-centric network address.

SUMMARY

To address the above-detailed deficiencies, it is a feature of the present invention to provide a messaging system for sending a message to a receiving device at a future delivery time, where the receiving device is coupled to either a telephony-centric network or to a data-centric network.

Accordingly, the present invention provides an apparatus for sending a message to a receiving device, where the receiving device is coupled to either a data-centric network or a telephony-centric network. The apparatus includes a message server, a data-centric network server, and a telephony-centric network server. The message server translates the message into a format compatible with the receiving device and initiates delivery of the message at a delivery time. The data-centric network server is coupled to the message server. The data-centric network server transmits the message over the data-centric network. If the receiving device is addressable over the data-centric network, then the data-centric network server delivers the message to the receiving device. The telephony-centric network server is coupled to the data-centric network server. The telephony-centric network server interfaces the data-centric network server to the telephony-centric network. If the receiving device is addressable by the telephony-centric network, then the telephony-centric network server receives the message from the data-centric network server and delivers the message to the receiving device over the telephony-centric network.

A benefit of the present invention is that a user is not required to retain a messaging service or to acquire special-purpose hardware/software to obtain future messaging capabilities.

In another aspect, the present invention provides a mechanism for sending a message to a receiving device, where the receiving device is coupled to either a data-centric network or a telephony-centric network. The mechanism has a message server and a data-centric network server. The message server translates the message into a format compatible with the receiving device and initiates delivery of the message. The message server has a message scheduler that causes the message server to initiate delivery of the message at a delivery time, the delivery time being specified within the message. The data-centric network server is coupled to the message server and transmits the message over a data-centric network for delivery to the receiving device.

Another benefit of the present invention is that a user is provided with device-independent future messaging capabilities. No longer is he/she required to restrict delivery of

US 6,438,217 B1

5

messages only to those receiving devices that are addressable over the same network the device used to originate a message.

In a further aspect, the present invention provides a system for sending a message at a specified delivery time to a receiving device. The system includes a message scheduler, a message server, a data-centric network server, and a data-centric network. The message scheduler initiates delivery of the message at the specified delivery time. The message server is coupled to the message scheduler and translates the message into a format that is compatible with the receiving device. The data-centric network server is coupled to the message server and transmits the message. The data-centric network is coupled to the data-centric network server. The data-centric network routes the message from the data-centric network server to either the receiving device or a telephony-centric network server, wherein, if the receiving device is addressable over a telephony-centric network, then the data-centric network routes the message to the telephony-centric network server.

A further benefit of the present invention is that a user can configure and execute voicemail messages to himself/herself throughout the day to provide administrative prompts and reminders.

In yet another aspect, the present invention provides a method for sending a message at a delivery time to a receiving device that is coupled either to a data-centric network or a telephony-centric network. The method includes generating the message from an originating device, the message prescribing the receiving device and the delivery time; translating the message into a format that is compatible with the receiving device; at the delivery time, transmitting the message over a data-centric network; and delivering the message to the receiving device.

Yet another benefit of the present invention is that a message originator can enter a message for future transmission to a recipient, without having to be concerned about the network over which the message is transmitted.

BRIEF DESCRIPTION OF THE DRAWINGS

These and other features, and benefits of the present invention will become better understood with regard to the following description, and accompanying drawings where:

FIG. 1 is a block diagram illustrating related art mechanisms for delivering a future message to a recipient over a telephony-centric network.

FIG. 2 is a block diagram illustrating related art mechanisms for delivering a future message to a recipient over a data-centric network.

FIG. 3 is a diagram illustrating related art future messaging capabilities for delivery to recipients over the telephony-centric network and the data-centric network.

FIG. 4 is a block diagram of a channel-transparent future messaging system according to the present invention.

FIG. 5 is a diagram depicting a future message transmission according to the present invention to recipients having disparate receiving devices.

FIG. 6 is a diagram illustrating a future message data entry window web page according to the present invention.

FIG. 7 is a diagram illustrating configuration of a wake-up call according to the present invention.

FIG. 8 is a flow chart of a method according to the present invention for configuring and delivering a message at a future time to a receiving device that is addressable over a telephony-centric network or to a receiving device that is addressable over a data-centric network.

6

DETAILED DESCRIPTION

In light of the above background on messaging techniques and mechanisms, several related art examples will now be discussed with reference to FIGS. 1 through 3. These examples illustrate how present day messaging systems confine delivery of a future, or delayed, message to recipients having receiving devices that are connected to the same communication network as the messaging system, a specific obstacle being the inability to direct a future message to either a data-centric network addressee (e.g., electronic mail) or a telephony-centric network addressee (e.g., voicemail). Following this discussion, a detailed description of the present invention will be provided with reference to FIGS. 4 through 8. The present invention permits a user to address a future message for delivery to either data-centric network devices or telephony-centric network devices, or both types of devices. Moreover, the present invention delivers the future message at a specified delivery time to all recipients over communication channels that are transparent to the message originator. The message is delivered to each recipient in a form (e.g., email, facsimile, voicemail) that matches his/her receiving device.

Referring to FIG. 1, a block diagram 100 is presented illustrating related art mechanisms for delivering a future message to a recipient over a telephony-centric network. The block diagram 100 shows two local telephony-centric network interfaces 112, one 112 at POINT A and one 112 at POINT B. The telephony-centric network interface 112 is also referred to as a local switch 112. The block diagram 100 additionally depicts various devices connected to the local telephony-centric network interfaces 112: a telephone 102, a pager 108, a facsimile (fax) machine 110, and a modem 106 that provides connectivity for a computer 104. The block diagram 100 shows three channels for transmitting telecommunication signals: a hardwired channel 114, a radio frequency (RF) line-of-sight (LOS) channel 118, and a satellite communications (SATCOM) channel 122.

In operation, each of the devices 102, 108, 110, 106/104 are provided with a unique address, or telephone number, so that they may be readily accessed by the local telephony-centric network interface 112 for message transmission and receipt. The local switch 112 is the point where local telecommunication devices 102, 108, 110, 106/104 interface to the telephony-centric network communication channels 114, 118, 122. A transmitting local device, say a telephone 102 at POINT A, sends a message to a compatible receiving device, say a telephone 102 at POINT B, by providing a telephone number assigned for the receiving device 102 at POINT B to the local switch 112 at POINT A. The local switch 112 at POINT A then transmits the message to the local switch 112 at POINT B via the hardwired channel 114, the RF LOS channel 118, the SATCOM channel 122, or a combination thereof. The local switch 112 at POINT B then delivers the message to the receiving device 102. For a given message, routing logic (not shown) within the local switch 112 at POINT A determines which telephony channel 114, 118, 122 or combination of channels 114, 118, 122 to use for the transmission. This determination is based upon a number of factors to include the geographic separation of POINT A and POINT B and the availability of a channel 114, 118, 122 at the time the given message is transmitted. For example, a first message from San Francisco to San Jose, because the two points are only a few miles apart, may be transmitted over the hardwired, or landline, channel 114. This channel 114 modulates electrical signals over wires or fiber-optic cables to communicate the first message between San Fran-

US 6,438,217 B1

7

cisco and San Jose. A second message from San Francisco to Los Angeles, because the two points are separated by hundreds of miles, may be transmitted over the RF LOS channel 118. This channel translates electrical signals provided by the local switch 112 in San Francisco to RF signals and transmits the second message between a number of RF LOS antennas 116 for delivery to Los Angeles. The RF signals are then translated back to electrical signals compatible with receiving devices in Los Angeles. A third message from San Francisco to Paris, because the two points are separated by thousands of miles, may be transmitted over the SATCOM channel 122. This channel translates electrical signals provided by the local switch 112 in San Francisco to RF signals and transmits the third message between a transmitting satellite antenna 120 to a satellite 124 above the Earth. The satellite 124 relays the third message to a receiving satellite antenna 120 near Paris. The RF signals are then translated back to electrical signals compatible with receiving devices in Los Angeles. One skilled in the art will appreciate that many factors influence the channel medium 114, 118, 122 chosen for transmission of a given message over the telephony-centric network and that the choice of medium 114, 118, 122 is transparent to both the message originator and the message recipient. A message between San Francisco and Paris could just as well be transmitted by landlines 114 as by a satellite 124—what the originator and recipient hear is words spoken over a telephone 102.

Regardless of which channel 114, 118, 122 is provided for transmission of a message, it is important to note that the local switch 112 is the point of interface to the telephony-centric network and that each device 102, 108, 110, 106/104 connected to the network is accessed, or addressed, by a unique telephone number. To be accessed, that is, to transmit and receive messages, a device 102, 108, 110, 106/104 must be connected to the telephony-centric network and must have an assigned telephone number. Use of the telephone number is the only way to address a device 102, 108, 110, 106/104. Moreover, a device 102, 108, 110, 106/104 connected to the telephony-centric network may not be accessed via any other network except through a corresponding local switch 112.

Although messages via the telephony-centric network are modulated for reliable transmission over a particular telephony channel 114, 118, 122, the format of such messages can differ. For instance, digitized voice files are normally transmitted between two telephones 102, thus providing voice-to-voice messaging. One skilled in the art will appreciate that there are several digitized voice file formats in use today and that off-the-shelf products are available for translation from one file format to the next. But voice-to-voice messaging is not the only form of messaging over the telephony-centric network; text-to-text messaging is also available. A textual item may be encoded by a fax machine 110 at POINT A and then transmitted to a compatible fax machine 110 at POINT B, thus achieving text-to-text messaging. Yet, file formats for a fax are different than for digitized voice. And like digitized voice, a number of file formats are in use today. Virtually all present day fax machines 110 provide transparent translation between all of the fax file formats.

A pager 108 provides the capability to receive a textual message that has been created via a telephone keypad or similar alphanumeric entry device. The pager 108 is actually addressed over an RF paging channel 109 rather than a landline 114. As a result, a recipient within range of the RF paging channel 109 can be provided with a message in written form that has been entered from a telephone 102.

8

Like the telephone 102 and fax machine 110, several message file formats are in use today for pagers 108.

The computer 104 provides the capability to send either a voice format message or a text format message over the telephony-centric network. The modem 106 interfaces the computer 104 to a local switch 112 and modulates voice and text files provided by the computer 104 into electrical signals suited for transmission over the telephony-centric network. Most present day computers 104 that are connected to the telephony-centric network provide the capability to transmit and receive a voice message, or voice mail, and to transmit and receive a text message in the form of an electronic file or a fax. Readily available software programs for computer communications applications provide the capability to translate a fax file format to a format for display on a computer monitor (not shown).

It is also possible to send a message over the telephony-centric network to more than one receiving device 102, 106/104, 108, 110. A message addressed to more than one receiving device 102, 106/104, 108, 110 is called a broadcast message. In essence, the broadcast message is entered once from an originating device 102, 106/104, 108, 110 and is then broadcast to selected recipients. In practice, however, the broadcast message is actually translated into individual messages corresponding to each message recipient and the individual messages are subsequently transmitted to each message recipient over the telephony-centric network. For example, it is trivial to address a fax to several receiving fax machines 110 or computers 104 by providing the corresponding telephone numbers to the broadcast fax machine 110. The broadcast fax machine 110 then contacts each recipient individually over the telephony-centric network. A more complex case is illustrated by a message originator broadcasting a voicemail to both a receiving computer 104 and a receiving telephone 102 by entering a voice message into an originating computer 104 and selecting the telephone numbers corresponding to the intended recipients. When a voice message is immediately transmitted to more than one telephone or equivalent receiving device, the call session is referred to as a conference call. When the voice message is not immediately transmitted, but rather delayed for transmission at a future time, what transpires is known as a voice mail broadcast.

In general, messages that are not immediately delivered, but rather are specified to be delivered at a later time, are referred to as future messages, or delayed messages. Future messaging is prevalently used today both as a cost savings technique and as an administrative mechanism. With regard to cost savings, several facsimile programs for desktop computers provide the capability to specify a delivery time for generated faxes, thereby enabling a message originator to take advantage of nighttime long distance rates, which are most often lower than daytime rates. Regarding administrative uses of delayed messaging, the wake-up call is a good example: A businessperson checks into a hotel for the evening and calls the front desk to request a wake-up call for a specified time the next morning. The following morning, a hotel operator or special-purpose voicemail system places a call to the businessperson at the specified time, thus providing an administrative prompt to the businessperson to initiate a new task, i.e., to wake up.

One skilled in the art will appreciate that other variations exist that are not discussed above for future messaging over the telephony-centric network to include voice-to-text and text-to-voice translations. Such translations are provided for the hearing impaired via a TTY device connected to the telephony-centric network. However, in all cases, to address

US 6,438,217 B1

9

a message to any device 102, 106/104, 108, 110 over the telephony-centric network requires the provision of telephone numbers for each receiving device.

The telephony-centric network has been predominately used for commercial messaging since the early 1900's. But as of the mid-1980's commercial messaging practices began to change with the advent of the internet. The internet is a worldwide network of computers connected together via a data-centric network, the data-centric network being distinct from the telephony-centric network. And the primary factor that distinguishes devices connected to the internet from those connected to the telephony-centric network is that the devices connected to the internet are not addressed by telephone numbers; they are addressed with a designator called an internet protocol (IP) address. A more detailed description of messaging over a data-centric network is provided with reference to FIG. 2.

Referring to FIG. 2, a block diagram 200 is presented illustrating related art mechanisms for delivering a future message to a recipient over a data-centric network. The block diagram 200 shows two local data-centric network interfaces 212, one 212 at POINT A and one 212 at POINT B. The block diagram 100 also depicts computers 204 connected to the local data-centric network interfaces 112 via data-centric network medium 206. The data-centric network medium 206 can be a local area network, a wide area network, a cable modem, or any of a number of present day mechanisms for connecting a computer 204 to a data-centric network. Like the telephony-centric network of FIG. 1, three data communication channels are shown: a hardwired channel 214, a radio frequency (RF) line-of-sight (LOS) channel 218, and a satellite communications (SATCOM) channel 222.

In operation, each computer 204 connected to the data-centric network is provided with a unique IP address so that it may be readily accessed by the local data-centric network interface 212. The local data-centric network interface 212 is also called a local hub 212. The local hub 212 is the point where a computer 204 interfaces to the data-centric network communication channels 214, 218, 222. A transmitting computer 204 at POINT A sends a message to a receiving computer 204 at POINT B by providing an IP address assigned to the receiving computer 204 at POINT B to the local hub 212 at POINT A. The local hub 212 at POINT A then transmits the message to the local hub 212 at POINT B via the hardwired channel 214, the RF LOS channel 218, the SATCOM channel 222, or a combination thereof. The local hub 212 at POINT B subsequently delivers the message to the receiving computer 204. For a given message, routing logic (not shown) within the local hub 212 at POINT A determines which data-centric network channel 214, 218, 222 or combination of channels 214, 218, 222 to use for transmission. This determination is based upon a number of factors to include the geographic separation of POINT A and POINT B and the availability of a channel 214, 218, 222 at the time the given message is transmitted. Operation of each of the data-centric network channels 214, 218, 222 and their associated elements 216, 220, 224 is similar to like telephony-centric network elements of FIG. 1, the hundreds digit being replaced with a 2. One skilled in the art will likewise appreciate that many factors influence the medium 214, 218, 222 chosen for transmission of a given message over the data-centric network and that the choice of medium 214, 218, 222 is transparent to both the message originator and the message recipient. A message between San Francisco and Paris could just as well be transmitted by landlines 214 as by a satellite 224.

10

Regardless of the data channel 214, 218, 222 provided for communication of a message, it is important to note that the local hub 212 is the point of interface to the data-centric network and computers 204 connected to the network are addressed by unique IP addresses. To be accessed, that is, to transmit and receive messages, a computer 204 must be connected to the data-centric network and must have an assigned IP address. Use of the IP address is the only way to address a computer 204 connected to the data-centric network. Moreover, a computer 204 connected to the data-centric network may not be accessed via any other network except that the access occur through a local hub 212.

Although messages via the data-centric network are modulated for reliable transmission in accordance with a particular data-centric network channel 214, 218, 222, the format of such messages can differ. For instance, technology advances permit digitized voice files to be transmitted between two computers 204, thus providing voice-to-voice messaging. One skilled in the art will appreciate that there are several digitized voice file formats in use today and that off-the-shelf products are available for translation from one file format to the next. These formats include wave format (i.e., .wav files) and real-audio format (.ra files). Yet the most prevalent form of messaging exercised by computers 204 today is text-to-text messaging. A textual item is entered into a computer 204 at POINT A and then transmitted to a computer at POINT B, thus achieving text-to-text messaging, principally in email format. And though several email formats are used today, most present day computer communication provide transparent translation and presentation of both voice and text messages for virtually all of the various formats.

Like messaging over the telephony-centric network, it is possible to send a broadcast message over the data-centric network to more than one receiving computer 204. In fact, present day email software applications make it possible to enter one textual message, or to record a voice message, and then broadcast the message to numerous recipients simply by specifying their corresponding addresses. Yet, similar to broadcasting over the telephony-centric network, a data-centric network broadcast message is actually translated into individual messages corresponding to each message recipient and the individual messages are subsequently transmitted to each message recipient over the data-centric network.

Future messaging has been employed over a data-centric network, yet only in a limited sense. This is because common desktop computer communications programs do not provide the capability for a user to prescribe a time-to-transmit, or delivery time, for a message. Some commercial computing entities maintain special-purpose application programs that do provide the capability to prescribe delivery times for large batches of messages, such as solicitations and the like, but on the whole, delayed messaging capabilities are not provided for an end user.

Many software applications known as personal information managers (PIMs) provide a user with the ability to link a calendar event with a programmed response such as transmission of an email message. To schedule a delivery time for a message using this approach, one would create a calendar event to occur at the delivery time, say an event entitled "SEND MESSAGE." Then the message to be transmitted would be linked to the calendar event. Consequently, when the delivery time occurs, the email message would be sent. Although it is possible to schedule future delivery of a message in this manner, it is not very practicable. First, the user is required to purchase PIM software because such software is not common to desktop computer systems.

US 6,438,217 B1

11

Furthermore, each time the user desired to send a future message, he/she would be required to create a corresponding calendar event.

One skilled in the art will appreciate that a number of variations exist for broadcast messaging over the data-centric network that are not discussed above, to include voice-to-text and text-to-voice translations. And there are a number of special-purpose devices that can be connected to the data-centric network for special communication scenarios. These devices are referred to in the larger sense in this application simply as computers 204 because to address a broadcast message to such devices 204 over the data-centric network requires the provision of an IP address for each message recipient.

It is also well understood that telephonic communication channels 114, 118, 122 are the primary channels for transmission of information over the internet and other private data-centric networks. Nevertheless, though telephone channels 114, 118, 122 function as the backbone of a data-centric network, the distinction cited above remains: to address a message to a computer 204 or other device connected to a data-centric network, the recipient's IP address must be provided to a local interface 212 to the data-centric network.

Enabling technologies have proliferated in more recent years to the extent that the lines between telephony-centric network messaging and data-centric network messaging are becoming blurred, particularly from the standpoint of a user. A message originator desires to enter a message one time, in a format compatible with his/her data entry device, and then transmit this message to one or more recipients, without regard to the unique characteristics of receiving devices. The message originator furthermore has no interest in whether the receiving device is a telephone 102 connected to the telephony-centric network, or a computer 204 connected to the data-centric network, or some other device having either an IP address or a telephone number assigned to it. His/Her chief desire is to communicate information to recipients, not to interact with a host of disparate receiving devices. In addition, for cost control or administrative purposes, the message originator desires to specify a delivery time for each message that is generated. However, in spite of recent advances in the art, a number of obstacles have yet to be overcome that allow such seamless future messaging to occur. FIG. 3 summarizes the present day capabilities and limitations of future messaging between telephony-centric network devices and data-centric network devices.

Referring to FIG. 3, a diagram 300 is presented illustrating related art future messaging capabilities for delivery to recipients over the telephony-centric network and the data-centric network. The diagram 300 shows two telephony-centric network devices 302: a first device 302 having a telephone number of (415)731-1212 and a second device 302 having a telephone number of (705)750-9415. The diagram 300 also depicts two data-centric network devices 304: a first data-centric network device 304 having an IP address of 206.196.128.1 and a second data-centric network device 304 having an IP address of 982.243.587.4. The diagram 300 also depicts two telephony-centric network communication channels 306: a voice form channel 306 and a text form channel 306. Two data-centric network communication channels 308, a voice form channel 308 and a text form channel 308 are also presented. Two data-to-telephony-centric network channels 310 are depicted in the diagram 300 along with two telephone-to-data-centric network channels 312. The inter-network channels 310, 312 are depicted as dotted lines in the diagram 300 because such channels 310, 312 are presented only for the purposes of discussion;

12

present day implementations of inter-network channels 310, 312 exist only at a rudimentary level. The diagram 300 also depicts two message originators 312, 320: one 312 coupled to a telephony-centric network and another 320 coupled to a data-centric network. The diagram 300 additionally shows two message recipients 314, 322: one 314 coupled to a telephony-centric network and another 322 coupled to a data-centric network. The diagram moreover shows a first clock 316 depicting an origination time and a second clock 318 depicting a delivery time. The diagram 300 also shows an intervention apparatus 310 that must be provided to enable future transmission of messages. The intervention apparatus 310 may be a special-purpose piece of equipment, special-purpose software running on a desktop computer, or a human being.

Operationally, as was discussed with reference to FIG. 1, it is possible today to send future messages in both voice form and text form over telephony-centric network devices 302. For future transmission of voice messages, i.e., voicemail, telephones 302 are most often used to originate and receive. For future transmission of fax messages, fax machines 302 are most often used to originate and receive. A computer equipped with a modem 302 or fax modem 302 can be used to originate or receive either voicemail or facsimiles. Both voice form messages and text form future messages are transmitted over the telephone communication channels 306. Recipients of such messages are addressed by their corresponding telephone number.

But a computer 302 connected to the telephony-centric network must have special-purpose software installed to translate voice-to-text format or text-to-voice format. Hence, without special-purpose software, a voicemail received by a computer 302 must be heard, not read. Furthermore, a fax must be read, not heard. Special-purpose software is available to provide voice-to-text and text-to-voice translation, for example as a TTY aid for the deaf, but such software is rarely found in desktop computing systems.

As was discussed with reference to FIG. 2, it is possible today to broadcast both voice form and text form messages over data-centric network devices 304. For broadcasting of voice messages, i.e., broadcast voicemail, computers 304 are most often used to originate and receive. Digitized voice file formats are used by the computers 304 to send and receive streaming audio. Email is the most common embodiment of text form messaging over data-centric network channels 308 today. And although email addressing mechanisms are presented to a user in the form of an email address like joe@pto.gov, one skilled in the art will comprehend that the email itself is routed to a specific computer 304 having a unique IP address. To translate voice to text or text to voice, as was discussed above, requires special-purpose software not commonly found in desktop systems 304.

Present day techniques do not allow a user to direct a voice format message from a telephone 302 connected to the telephony-centric network to a receiving device 304 connected to a data-centric network. But it is possible to direct a text format message from a device 302 called a smart pager 302 to a receiving device connected to a data-centric network-however only with human intervention. For example, a smart pager user types in a message on his/her smart pager 304 and selects an email address of a recipient. The message is broadcast over the telephony-centric network to a paging center, where a technician intercepts and forwards the message to a recipient over the data-centric network. It is also not possible to enter a voice message over a telephony-centric network device 302 and deliver it as a text format message to a data-centric network device 304.

US 6,438,217 B1

13

Furthermore, a user cannot enter a text message from a telephony-centric network device 302 and deliver it as a voice message to a data-centric network device, without the employment of special translation software.

Mechanisms do exist today to send a fax or a page from a data-centric network device 304 to a telephony-centric network device 302. The fax and page are entered and received in text format. Special-purpose translation software is required to perform text-to-voice translation. Present day mechanisms do not provide the capability for a user to enter a voice message over a data-centric network device and deliver it to a telephony-centric network device.

With regard to scheduling and transmitting future messages, any of the available techniques discussed with reference to FIG. 3 can be used, however, to do so proves cumbersome or costly to a message originator because the intervention apparatus 310 is required to delay transmission of a message until a specified delivery time. In the simple, yet costly case, the intervention apparatus 310 takes the form of a paid message operator. At the origination time shown on the origination clock 316, the telephone message originator 312 calls the paid message operator 310 and prescribes the telephone number of the telephone recipient 314, the message content, and the specified delivery time. At the specified delivery time shown on the delivery clock 318, the paid message operator places a call to the telephone recipient 314 to deliver the message. With a paid message operator 310, it is also possible to transmit a future message from the telephone originator 312 to the data-centric network recipient 322.

Future messaging over a data-centric network typically requires intervention apparatus 310 in the form of special-purpose software. And with custom intervention apparatus 310, it is possible to send a future message from a data-centric network originator 320 to a telephony-centric network receiver 314, the smart pager being one example of how this is done.

In summary, both the telephony-centric network and data-centric network extend virtually all around the world. And it is standard practice to send both voice format and text format messages over either network. But present day messaging systems do not provide a user with the capability to seamlessly transmit a message to a receiving device that is coupled either to the data-centric network or to the telephony-centric network, without intervention apparatus 310. Either an expensive messaging service is required, or the message originator is required to send his message twice, once from a data-centric network device 304 and again from a telephony-centric network device 302. Furthermore, even if a recipient is within the same network as an originator, to schedule transmission of a message at a future delivery time requires an intervention apparatus 310, thus further limiting a user's ability to communicate.

The problems associated with future messaging are further exacerbated by the fact that some receiving devices 302, 304 are primarily voice based, that is, they 302, 304 provide no capability to receive text format messages. A telephone is an example of a voice based device. Other receiving devices 302, 304 are primarily text based, such as a fax machine. One skilled in the art will appreciate that any of the above limitations can be overcome through use of special intervention apparatus 310, either in the form of a human, or a special-purpose computer, or special application software. But such capabilities are not common, and as a result, obtaining these capabilities requires a significant investment.

14

The present invention overcomes the above noted obstacles to both inter-network future messaging and message format translation by providing channel-transparent future messaging and format translation via servers that are connected to both a data-centric network and the telephony-centric network. Routing information, translation software (voice-to-text and text-to-voice translators), and message transmission scheduling logic are resident in the servers according to the present invention rather than being resident in originating and receiving devices. Moreover, the channel-transparent future messaging system according to the present invention can be accessed from either the telephony-centric network or the data-centric network. The present invention is more specifically described with reference to FIGS. 4 through 8.

Now referring to FIG. 4, a block diagram is presented of a channel-transparent future messaging system 400 according to the present invention. The future messaging system 400 includes a message server 402 and a data-centric network server 404, both servers 402, 404 located at a network operations center. The data-centric network server 404 is connected to a data-centric network 406. In one embodiment, the data-centric network is the Internet, also known as the World Wide Web. In an alternative embodiment, the data-centric network 406 is a private packet-switched network. The future messaging system 400 also includes telephony-centric network servers 408 that interface to the data-centric network 406 and that communicate with corresponding local telephony-centric network interfaces 454, or local switches 454. The block diagram also depicts local data-centric network interfaces 452 that are connected to the data-centric network 406. For the ensuing discussion, the Internet embodiment is specifically described, however, one skilled in the art will appreciate that elements and techniques similar to those discussed for the Internet embodiment may be substituted for a private data-centric network embodiment.

Operationally, a user (not shown) may access the future messaging system 400 either via a telephony-centric network device (not shown) or a data-centric network device (not shown). By dialing a local telephone number, the user accesses the messaging system 400 from his/her local switch 454. The telephony-centric network server 408, or local point-of-presence (POP) 408, in one embodiment, is collocated with a corresponding local telephone switch 454. The local POP 408 converts electrical signals that are modulated for communication over the telephony-centric network into data packets for communication over the data-centric network 406. The data packets are then sent by the local POP 408 over the data-centric network 406 to the data-centric network server 404 in the Network Operations Center. The data-centric network server 404 receives the data packets transmitted over the data-centric network and provides them to the message server 402. The future messaging system 400 can also be directly accessed from a data-centric network device such as a computer. In one embodiment, a common desktop computer equipped with a thin web client (i.e., a web browser) such as Netscape® Communicator or Microsoft® Internet Explorer is used to access a web site at the network operations center. Thus the data-centric network server 404 provides network packet transmission and reception for access to the message server 402 in the network operations center.

The message server 402 maintains account and messaging information for registered users. Each registered user is issued a telephone number, an email address, and is provided with a universal resource locator (URL) corresponding to

US 6,438,217 B1

15

the network operations center. For example, a user in Denver is issued a telephone number having area code 303 corresponding to the local POP in Denver, an email address of, say richardh@thinklink.com, and <http://www.thinklink.com> as a URL. The registered user can access his/her account by dialing the local telephone number or by using his web browser to access the URL.

For entry of messages in text form, the message server 402 provides a data entry web page via the web site so that the registered user can enter a message from a computer keyboard or other device connected to the data-centric network that is hypertext markup language (HTML) compatible. Message recipients can be manually entered via the data entry form or they may be selected from an address book resident in the message server. Using the address book, a recipient, say Rick, can be aliased to several receiving addresses to include a telephone number, pager number, fax number, and email address. Multiple recipients may be specified for delivery of the message. The message server 402 provides the capability to address Rick via any combination of aliased receiving devices. The message entry web page contains a field for specifying a delivery time so that a message originator can individually specify delivery times for each message that is generated.

The message server also includes translation logic and special-purpose software to translate voice-to-text and text-to-voice so that a message can be seamlessly entered, transmitted, and received. For example, if a message is entered in text form and it is addressed to a recipient having a voice-only receiving device, then the message server 402 translates the message into a format compatible for reception by the voice-only device prior to transmission. If the receiving device is a fax machine, then the message routing logic 402 translates the message into a format compatible with the fax machine prior to transmission. And because translation is performed within the message server 402, no special-purpose translation logic of software is required to be resident in an originating or receiving device. In addition, a message scheduler 403 within the message server 402 determines when to send messages that have future delivery times specified. Thus, no special intervention apparatus is required to send a future message. In one embodiment, the message scheduler 403 is a software program that reads a real-time clock (not shown) within the message server 402 to determine when it is time to deliver a message.

The message server 402 routes messages designated for receiving devices connected to the data-centric network 406 directly to the IP address of a recipient. For recipients having receiving devices connected to the telephony-centric network, the message router 1) embeds the telephone number of a receiving device into the message along with contact protocol for the receiving device, and 2) routes the message to the IP address of the local POP 408 corresponding to the embedded telephone number. Upon reception of the message, the local POP 408 directs the local switch 454 to call the receiving device over the telephony-centric network. Once the call session is secured, then the local POP 408 delivers the message over the telephony-centric network in the format provided by the message server 402.

Now referring to FIG. 5, a diagram 500 is presented depicting future message transmission according to the present invention to recipients having disparate receiving devices. The diagram 500 shows a message originator 502, or registered user 502, in Austin, accessing the network operations center from a computer 504, composing a message 506 for transmission to four recipients, at approximately 10:00. A first recipient is in New York and has a pager

16

520 as a receiving device; a second recipient is in San Jose and has a fax machine 524 as a receiving device; a third recipient is in Paris and has a telephone 528 as a receiving device; and a fourth recipient is in Sao Paulo and has a computer 532 as a receiving device. The diagram 500 shows both a data-centric network server 512 and a message server computer 514 in the network operations center. The diagram 500 depicts transmission of the message over a data-centric network 510 to either a data-centric network interface 530 or to local POPs 516. The local POPs 516 are connected to corresponding local telephone switches 518, 522, 526 for interface to receiving devices 520, 524, 528 over the telephony-centric network.

In one embodiment, the message originator 502 directs his/her computer 504 to access the network operations center web site by entering the URL of the website, <http://www.thinklink.com>, into an address field of a web browser on the computer 504. Data packets are then routed over the data-centric network 510 via the data-centric network interface 508 in Austin to the data-centric network server 512 in the network operations center. The message server 514 issues appropriate web pages to the user for message entry, recipient addressing, and specification of delivery time by sending packets to the data-centric network address of the user's computer 504. Using the data entry web page, the user 502 composes a message 506 in email format to jim's pager, joe's fax machine, fred's voicemail, and julie's default receiving device, her email address. For the purposes of message composition, it is of no concern to the user 502 what transmission channel is used to contact the recipients; what the user 502 values is that the message 506 need be entered only once. In addition, the user specifies a delivery time of 2:30 Austin time. The content of the message is created to remind Jim, Joe, Fred, and Julie that they are required to participate in an online meeting with Rick and that the meeting will occur at 3:00 Austin time, which is 4:00 New York time, 1:00 San Jose time, 10:00 Paris time, and 6:00 Sao Paulo time.

The future message 506 is provided to the network operations center over the data-centric network 510. The data-centric network server 512 receives the message packets and provides them to the message server 514. The message server computer 514 then translates the message 506 into four messages 534, 536, 538, 540, each of the four messages 534, 536, 538, 540 corresponding to each of the message recipients.

Translation of the future message 506 into a first message 534 requires that 1) the telephone number of Jim's pager 520, 268-3212, be embedded, 2) the IP address of the local POP 516 in New York be used as an address for the first message 534, and 3) that the future message 506 be translated to pager-compatible format. The message server 514 accomplishes these tasks and maintains it until the specified delivery time. At 3:30 New York time, the message server 514 provides the first message 534 to the data-centric network server 512 for delivery to the local POP 516 in New York.

Translation of the future message 506 into a second message 536 requires that 1) the telephone number of Joe's fax machine 524, 555-6363, be embedded, 2) the IP address of the local POP 516 in San Jose be used as an address for the second message 536, and 3) that the future message 506 be translated into facsimile-compatible format. The message server 514 accomplishes these tasks and maintains it until the specified delivery time. At 12:30 San Jose time, the message server 514 provides the second message 536 to the data-centric network server 512 for delivery to the local POP 516 in San Jose.

US 6,438,217 B1

17

Translation of the future message 506 into a third message 538 requires that 1) the telephone number of Fred's telephone 528, 44 84 18 13, be embedded, 2) the IP address of the local POP 516 in Paris be used as an address for the third message 538, and 3) that the future message 506 be translated from textual email format to voice format. The message server 514 accomplishes these tasks and maintains it until the specified delivery time. At 9:30 Paris time, the message sever 514 provides the third message 538 to the data-centric network server 512 for delivery to the local POP 516 in Paris.

Translation of the future message 506 into a fourth message 540 requires only that Julie's email address be provided as a address. This is because the future message 506 is already compatible with Julie's receiving device 532 and because Julie's receiving device 532, a computer 532, is addressable over the data-centric network 510. The message server 514 supplies Julie's email address and maintains the fourth message until the specified delivery time. At 5:30 Sao Paulo time, the message sever 514 provides the fourth message 540 to the data-centric network server 512 for delivery to a local data-centric network interface 530 in Sao Paulo.

FIG. 5 depicts Internet-based email addresses for transmission of each of the four messages 534, 536, 538, 540. For the three messages 534, 536, 538 requiring access through a local switch 518, 522, 526, telephone numbers for corresponding receiving devices 520, 524, 528 have been embedded into their associated email address. Although such and embodiment is shown in FIG. 5 for transmission of the future message 506, one skilled in the art will understand that the telephone numbers and local POP information can be embedded within the message 506 itself, or provided separately.

At the specified delivery time, the data-centric network server 512 transmits packets corresponding to each of the four messages 534, 536, 538, 540 over the data-centric network 510. The local POP 516 in New York intercepts the first message 534 and directs the local switch 518 to contact Jim's pager 520 using the embedded telephone number 268-3212. The first message 534 is then delivered to Jim's pager 520. The local POP 516 in San Jose intercepts the second message 536 and directs the local switch 522 to call Joe's fax machine 524 using the embedded telephone number 555-6363. Once the call is established, the local POP 516 provides the second message 536 in fax format to Joe's fax machine 524. The local POP 516 in Paris intercepts the third message 538 and directs the local switch 526 to call Fred's telephone 528 using the embedded telephone number 44 84 18 13. Once the call is established, the local POP 516 provides the third message 538 in recorded voice format to Fred's telephone 528. The local data-centric network interface 530 in Sao Paulo simply routes the fourth message 540 to Julie's computer 532 using her data-centric network address, julie@xyz123.com.

Now referring to FIG. 6, a diagram 600 is presented illustrating a future message data entry window web page 610 according to the present invention. The web page 610 is provided over a data-centric network to an originator's computer that is executing a thin web client program. The diagram 600 depicts a display 602 provided by the thin web client. The message data entry web page 610 includes a recipient field 612, a message field 614, and a delivery time field 616.

Operationally, a user accesses the message entry web page 610 by providing a universal resource locator (URL) to

18

his/her web browser corresponding to the data-centric network server at the network operations center. In turn, the user is provided with a web page (not shown) enabling he/she to select from several options, one of which is to compose a message. The message data entry web page 610 is sent to the user's computer as a result of selecting to compose a message. The user creates the message by entering addresses of recipients within the recipient field 612, message text within the message field 614, and a time to transmit the future message within the delivery time field 616. The completed web page 610 is transmitted back to the data-centric network server and a message server in turn translates the message into formats compatible with the recipients' receiving devices. The message scheduler then transmits the message to the recipients at the scheduled delivery time, in this case May 22, 1999 at 14:30.

Future messaging as depicted in FIG. 6 has many useful applications. For example, suppose that a salesperson is traveling to a client's facility to present a proposal. If the proposal is being prepared by an outside source in voice form while the salesperson is in transit, the present invention can be employed to, say, provide a fax form of the proposal on a fax machine at the client's facility shortly after the salesperson is scheduled to arrive. Another useful application of the present invention is described with reference to FIG. 7.

Referring to FIG. 7, a diagram 700 is presented illustrating configuration of a wake-up call through completion of a future message data entry window web page 710 like that shown in FIG. 6. Like elements have like descriptions, the hundreds digit being replaced by a 7.

Operationally, a wake up call is configured by entering a telephone number within the recipient field 712, a message to be read in voice form in the message field 714, and a the time that the user desires to be called in the delivery time field 716. Consequently, the wake-up call will occur at the scheduled delivery time, in this case May 23, 1999 at 06:00.

The examples of FIGS. 4 through 7 show how a future message is generated and transmitted according to the present invention to disparate receiving devices at a prescribed delivery time. The receiving devices can be coupled to either the telephony-centric network or the data-centric network and selection of communication channel for transmission of the message to recipients is transparent to a message originator. The examples furthermore show how format incompatibilities between receiving devices and transmission scheduling are overcome by the present invention without a requirement for human intervention, special-purpose hardware, or special client applications.

Now referring to FIG. 8, a flow chart 800 is presented of a method according to the present invention for configuring and delivering a message at a future time to a receiving device that is addressable over a telephony-centric network or to a receiving device that is addressable over a data-centric network.

Flow begins at block 802 where a user initiates a session at the network operations center to enter a message for future transmission. Flow then proceeds to decision block 804.

At decision block 804, it is determined whether the user is originating the message via a device connected to the telephony-centric network or a device connected to a data-centric network. If the origination device is connected to the data-centric network, then flow proceeds to block 806. If the origination device is connected to the telephony-centric network, then flow proceeds to block 808.

At block 806, a data-centric network server at the network operations center intercepts packets from the data-centric

US 6,438,217 B1

19

network originated by the user and establishes a session for entry of the future message over the data-centric network. A message entry web page is provided over the data-centric network to the user. Flow then proceeds to block 810.

At block 808, a local POP corresponding to the user's telephone number directs packets over the data-centric network to the data-centric network server. A message entry session is established via the local POP in a format compatible with the user's telephony-centric network device. Flow then proceeds to block 810.

At block 810, the user creates a message in the format obtained via block 806 or 808. The user also selects recipients for the broadcast message either directly or by using his/her address book stored at the network operations center. For recipients having multiple receiving devices, the user specifies a particular receiving device for delivery. In addition, the user specifies a future delivery time for the message. Flow then proceeds to decision block 812.

At decision block 812, a message server at the network operations center periodically reads the current time and determine if the prescribed delivery time is present. If it is time to transmit the message, then flow proceeds to block 814. If not, then flow proceeds to this decision block 812.

At block 814, the message server at the network operations center parses the broadcast message according to each specified recipient into corresponding individual messages. Flow then proceeds to decision block 816.

At decision block 816, the message server evaluates how to access each receiving device corresponding to each message recipient. If a message recipient's receiving device is connected to the telephony-centric network, then flow proceeds to block 818. If a message recipient's receiving device is connected to the data-centric network, then flow proceeds to block 822.

At block 818, individual messages for receiving devices connected to the telephony-centric network are translated, if necessary, from the format in which the future message was entered in block 810 into a format compatible with a designated receiving device. Flow then proceeds to block 820.

At block 820, individual messages designated for delivery to receiving devices connected to the telephony-centric network are transmitted over the data-centric network to a local POP corresponding to a telephone number embedded in each individual message. Flow then proceeds to block 824.

At block 824, the local POP accesses a receiving device by providing the number to a local telephone switch and subsequently delivers the future message in the format provided by block 818. Flow then proceeds to block 826.

At block 816, individual messages for receiving devices connected to the data-centric network are translated, if necessary, from the format in which the future message was entered in block 810 into a format compatible with designated receiving devices and the individual messages are transmitted over the data-centric network and delivered to the designated receiving devices. Flow then proceeds to block 826.

At block 826, the method completes.

Although the present invention and its objects, features, and advantages have been described in detail, other embodiments are encompassed by the invention. For example, the present invention has been particularly characterized by transmission of messages over the Internet data-centric network. Although the Internet is widely used today for

20

transmission of messages between communication devices, the present invention is not dependent upon such capability being provided. The data-centric network element according to the present invention can be embodied as a private network utilizing proprietary or leased communication channel assets.

In addition, the present invention has been specifically discussed with reference to commonly found receiving devices such as telephones, fax machines, computers, and pagers, however, such devices do not restrict application of the present invention. Any device having a telephone number or data-centric network address that provides either voice or text format communication capability may be applied to the present invention.

Furthermore, the present invention has been characterized in terms of voice format and text format messaging because such formats are commonly employed at present. In the near future enabling technologies may permit the proliferation of video telephones or video data-centric network communication devices. The present invention comprehends incorporation of video-based formats and devices into all aspects of future messaging.

Those skilled in the art should appreciate that they can readily use the disclosed conception and specific embodiments as a basis for designing or modifying other structures for carrying out the same purposes of the present invention without departing from the spirit and scope of the invention as defined by the appended claims.

What is claimed is:

1. An apparatus for sending a message to a receiving device, the receiving device coupled to either a data-centric network or a telephony-centric network, the apparatus comprising:

a message server, configured to translate the message into a format compatible with the receiving device, and to initiate delivery of the message at a delivery time;

a data-centric network server, coupled to said message server, configured to transmit the message over the data-centric network, wherein, if the receiving device is addressable over the data-centric network, then said data-centric network server delivers the message to the receiving device; and

a telephony-centric network server, coupled to said data-centric network server, configured to interface said data-centric network server to the telephony-centric network, wherein, if the receiving device is addressable by the telephony-centric network, then said telephony-centric network server receives the message from said data-centric network server and delivers the message to the receiving device over the telephony-centric network.

2. The apparatus as recited in claim 1, wherein the receiving device comprises a telephone, a facsimile machine, a pager, a smart pager, a cellular telephone, a personal digital assistant, or a desktop computer.

3. The apparatus as recited in claim 2, wherein the receiving device is addressed by dialing a telephone number over the telephony-centric network.

4. The apparatus as recited in claim 2, wherein the receiving device is addressed by accessing an internet protocol (IP) address over the data-centric network.

5. The apparatus as recited in claim 4, wherein said IP address is accessed using TCP/IP protocol.

6. The apparatus as recited in claim 5, wherein the message is supplied to the message server by an originator.

7. The apparatus as recited in claim 6, wherein said originator supplies the message in text or voice form from an

US 6,438,217 B1

21

originating device that is addressable over the telephony-centric network.

8. The apparatus as recited in claim 6, wherein said originator supplies the message in text or voice form from an originating device that is addressable over the data-centric network.

9. The apparatus as recited in claim 6, wherein said data-centric network server transmits a message entry web page to said originator for configuration of the message.

10. The apparatus as recited in claim 9, wherein said originating device comprises a computer that is executing a thin web client to receive said message entry web page.

11. The apparatus as recited in claim 10, wherein said thin web client comprises Netscape® Communicator or Microsoft® Internet Explorer.

12. The apparatus as recited in claim 11, wherein said originator prescribes the receiving device, the message, and said delivery time by completing fields within said message entry web page.

13. The apparatus as recited in claim 2, wherein the message is delivered to the receiving device in text format as an email, or as a fax, or as a text file.

14. The apparatus as recited in claim 2, wherein the message is delivered to the receiving device in voice format as an electronic voice file.

15. The apparatus as recited in claim 1, wherein said message server selects said format for delivery according to receiving capabilities of the receiving device.

16. The apparatus as recited in claim 15, wherein, if the receiving device is addressable over the telephony-centric network, said message server includes a telephone number within the message that corresponds to the receiving device.

17. The apparatus as recited in claim 16, wherein said telephony-centric network server extracts said telephone number from within a field of the message for initiating a call over the telephony-centric network to the receiving device for delivery of the message.

18. The apparatus as recited in claim 15, wherein said message server comprises message scheduling logic, for determining when said delivery time occurs.

19. A mechanism for sending to a receiving device a message having a field prescribing a delivery time, the receiving device being coupled to either a data-centric network or a telephony-centric network, the mechanism comprising:

a message server, for translating the message into a format compatible with the receiving device, and for initiating delivery of the message, said message server comprising:

a message scheduler, for causing said message server to initiate delivery of the message at the delivery time prescribed within the field of the message; and

a data-centric network server, coupled to said message server, for transmitting the message over a data-centric network for delivery to the receiving device.

20. The mechanism as recited in claim 19, wherein the receiving device is addressed by dialing a telephone number over the telephony-centric network.

21. The mechanism as recited in claim 20, wherein the receiving device comprises a telephone, a facsimile machine, a pager, a smart pager, or a cellular telephone.

22. The mechanism as recited in claim 19, wherein the receiving device is addressed by accessing an internet protocol (IP) address over the data-centric network.

23. The mechanism as recited in claim 22, wherein said IP address is accessed using TCP/IP protocol.

24. The mechanism as recited in claim 23, wherein the receiving device comprises a computer.

22

25. The mechanism as recited in claim 19, wherein the message is supplied to the message server by an originator.

26. The mechanism as recited in claim 25, wherein said originator supplies the message in voice form from an originating device that is addressable over the telephony-centric network.

27. The mechanism as recited in claim 26, wherein said originator supplies the message in text or voice form from an originating device that is addressable over the data-centric network.

28. The mechanism as recited in claim 27, wherein said data-centric network server transmits a message entry web page to said originator for configuration of the message.

29. The mechanism as recited in claim 28, wherein said originating device comprises a computer that is executing a thin web client to receive said message entry web page.

30. The mechanism as recited in claim 29, wherein said thin web client comprises Netscape® Communicator or Microsoft® Internet Explorer.

31. The mechanism as recited in claim 30, wherein said originator prescribes the recipient, the receiving device, the message, and said delivery time by completing fields within said message entry web page.

32. The mechanism as recited in claim 19, wherein the message is delivered to the receiving device in text format as an email, or as a fax, or as a text file.

33. The mechanism as recited in claim 19, wherein the message is delivered to the receiving device in voice format as an electronic voice file.

34. The mechanism as recited in claim 19, wherein said message server selects said format for delivery according to receiving capabilities of the receiving device.

35. The mechanism as recited in claim 34, wherein, if the receiving device is addressable over the telephony-centric network, said message server includes a telephone number within the message that corresponds to the receiving device.

36. The mechanism as recited in claim 34, further comprising:

a telephony-centric network server, coupled to said data-centric network server, for extracting said telephone number from within the message and for initiating a call over the telephony-centric network to the receiving device for delivery of the message.

37. A system for sending a message at a specified delivery time to a receiving device, the system comprising:

a message scheduler, configured to initiate delivery of the message at the specified delivery time;

a message server, coupled to said message scheduler, configured to translate the message into a format that is compatible with the receiving device;

a data-centric network server, coupled to said message server, configured to transmit the message;

a data-centric network, coupled to said data-centric network server, configured to route the message from said data-centric network server to either the receiving device or a telephony-centric network server, wherein, if the receiving device is addressable over a telephony-centric network, then said data-centric network routes the message to said telephony-centric network server.

38. The system as recited in claim 37, wherein, if said receiving device is addressable over said data-centric network, then the message is routed directly to said receiving device.

39. The system as recited in claim 38, wherein the receiving device comprises a telephone, a facsimile machine, a pager, a smart pager, a computer, or a cellular telephone.

US 6,438,217 B1

23

40. The system as recited in claim 39, wherein the receiving device is addressed by accessing an internet protocol (IP) address over said data-centric network or by dialing a telephone number over said telephony-centric network.

41. The system as recited in claim 40, wherein an originator supplies the message in voice form from an originating device that is addressable over the telephony-centric network.

42. The system as recited in claim 40, wherein said originator supplies the message in text or voice form from an originating device that is addressable over the data-centric network.

43. The system as recited in claim 42, wherein said originating device comprises a computer that is executing a web browser.

44. The system as recited in claim 43, wherein said web browser comprises Netscape® Communicator or Microsoft® Internet Explorer.

45. The system as recited in claim 44, wherein said data-centric network server transmits a message entry web page to said originating device for configuration of the message.

46. The system as recited in claim 30, wherein said originator prescribes the receiving device, the message, and the specified delivery time by completing fields within said message entry web page.

47. The system as recited in claim 46, wherein the message is delivered to the receiving device in text format as an email, or as a fax, or as a text file.

48. The system as recited in claim 47, wherein the message is delivered to the receiving device in voice format as an electronic voice file.

49. The system as recited in claim 48, wherein said message server selects said format for delivery according to receiving capabilities of the receiving device.

50. The system as recited in claim 49, wherein said telephony-centric network server extracts said telephone number from within a field of the message and initiates a call over said telephony-centric network to the receiving device for delivery of the message.

51. A method for sending a message at a delivery time to a receiving device that is coupled either to a data-centric network or a telephony-centric network, the method comprising:

- a) generating the message from an originating device, the message having a recipient field prescribing a receiving

24

device and having a delivery time field prescribing a delivery time;

- b) translating the message into a format that is compatible with the receiving device;

- c) at or before the delivery time prescribed in the delivery time field of the message, transmitting the message over the data centric network to the receiving device prescribed in the recipient field;

- d) if the receiving device is coupled to the data-centric network, routing the message directly to the receiving device; and

- e) if the receiving device is coupled to the telephony-centric network, routing the message directly to a telephony-centric network server over the data-centric network, and causing the telephony-centric network server to deliver the message to the receiving device.

52. The method as recited in claim 51, wherein the originating device is addressable via a telephony-centric network.

53. The method as recited in claim 51, wherein the originating device is addressable via the data-centric network.

54. The method as recited in claim 53, wherein the originating device is a computer that is executing a thin web client.

55. The method as recited in claim 54, wherein the message is generated by completing fields within a message entry web page that has been received over the data-centric network.

56. The method as recited in claim 55, wherein the receiving device comprises a telephone, a facsimile machine, a pager, a smart pager, a computer, or a cellular telephone.

57. The method as recited in claim 56, wherein e) comprises:

- i) embedding a telephone number in the message;
- ii) causing the telephony-centric network server to call the telephone number to access the receiving device.

58. The method as recited in claim 57, wherein the message is delivered to the receiving device in text format as an email, or as a fax, or as a text file.

59. The method as recited in claim 57, wherein the message is delivered to the receiving device in voice format as an electronic voice file.

* * * * *



US006339794B2

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

(10) Patent No.: **US 6,339,794 B2**
(45) Date of Patent: ***Jan. 15, 2002**

(54) **WIRE PROTOCOL FOR A MEDIA SERVER SYSTEM**

(75) Inventors: **William J. Bolosky**, Issaquah; **Craig M. Dowell**, Redmond; **Robert P. Fitzgerald**, Redmond; **Steven P. Levi**, Redmond; **Jan de Rie**, Redmond; **Richard F. Rashid**, Woodinville, all of WA (US)

(73) Assignee: **Microsoft Corporation**, Redmond, WA (US)

(*) Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

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.: **08/569,380**

(22) Filed: **Dec. 8, 1995**

(51) Int. Cl.⁷ **G06F 15/173**

(52) U.S. Cl. **709/233; 709/227**

(58) Field of Search **395/200.12, 200.09, 395/200.04, 680, 200.57, 200.6, 200.3, 200.58, 200.33, 200.34; 370/464; 709/200, 203, 204, 228, 227, 230, 233**

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,319,353 A * 3/1982 Alvarez, III et al. 370/321

5,404,523 A * 4/1995 DellaFera et al. 395/671
5,446,846 A * 8/1995 Lennartsson 395/280
5,517,645 A * 5/1996 Stutz et al. 395/680
5,544,320 A * 8/1996 Konrad 395/200.09
5,555,375 A * 9/1996 Sudama et al. 395/200.03
5,603,091 A * 2/1997 Linquist et al. 455/56.1
5,621,734 A * 4/1997 Mann et al. 395/200.12
5,822,524 A * 10/1998 Chen et al. 395/200.33
5,854,893 A * 12/1998 Ludwig et al. 395/200.34

OTHER PUBLICATIONS

Digital Audio-Visual Council, *Davic 1.1 Specification Part 07—High And Mid Layer Protocols (Technical Specification)*, Sep. 15, 1995.

* cited by examiner

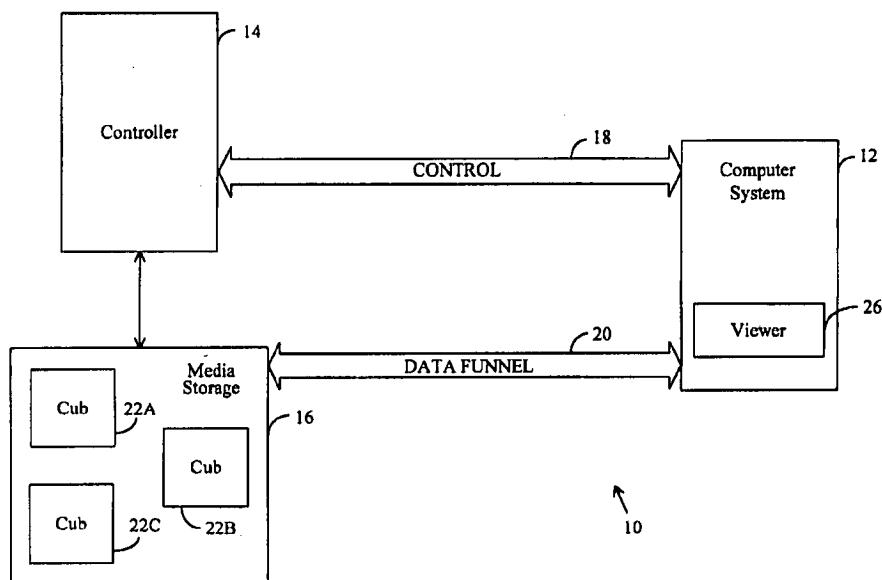
Primary Examiner—John A. Follansbee

(74) *Attorney, Agent, or Firm*—Lee & Hayes, PLLC

(57) **ABSTRACT**

A wire protocol provides message formats for creating multiple network connections between a media server and a client. These multiple network connections may include a control link connection for passing control information and a data funnel connection for passing data of multiple media. The data funnel connection may be a multipoint-to-point connection that connects multiple data servers with the client. The protocol facilitates multiple requests being concurrently outstanding and asynchronous processing of requests. The protocol is designed to exist on top of a transport protocol layer.

39 Claims, 8 Drawing Sheets



U.S. Patent

Jan. 15, 2002

Sheet 1 of 8

US 6,339,794 B2

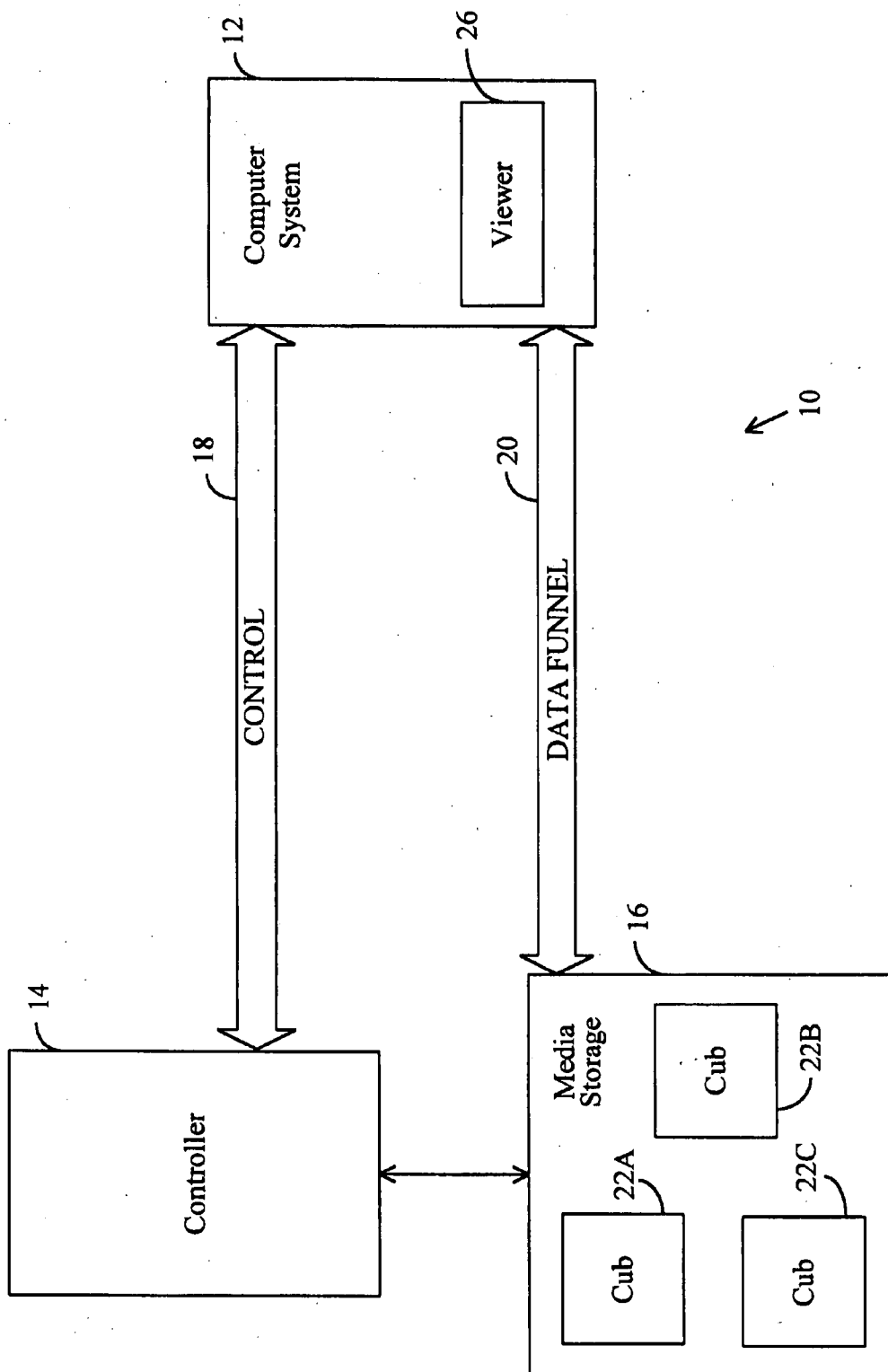


FIG. 1

U.S. Patent

Jan. 15, 2002

Sheet 2 of 8

US 6,339,794 B2

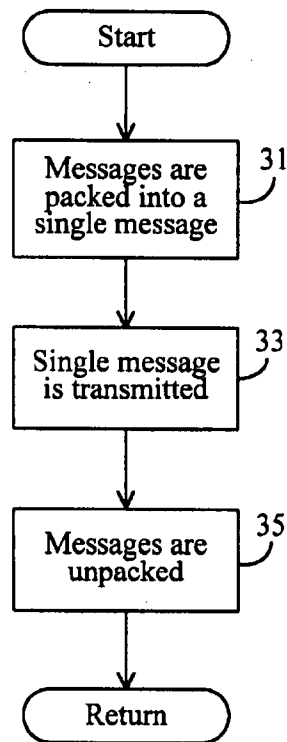


FIG. 2

U.S. Patent

Jan. 15, 2002

Sheet 3 of 8

US 6,339,794 B2

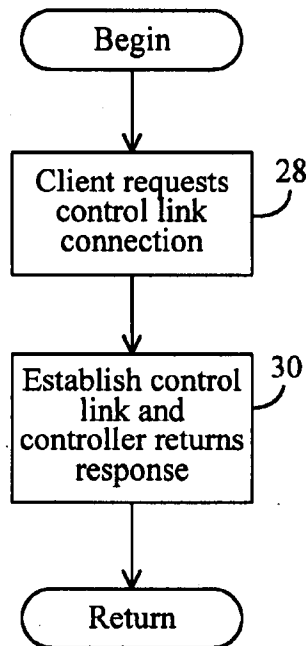


FIG. 3

U.S. Patent

Jan. 15, 2002

Sheet 4 of 8

US 6,339,794 B2

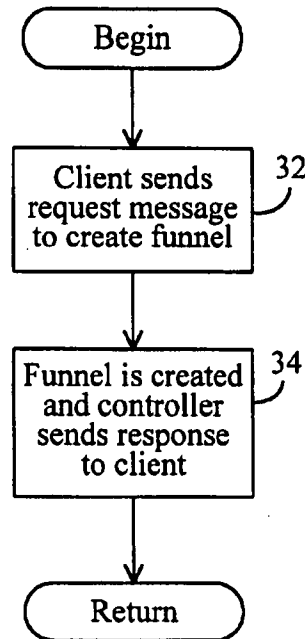


FIG. 4

U.S. Patent

Jan. 15, 2002

Sheet 5 of 8

US 6,339,794 B2

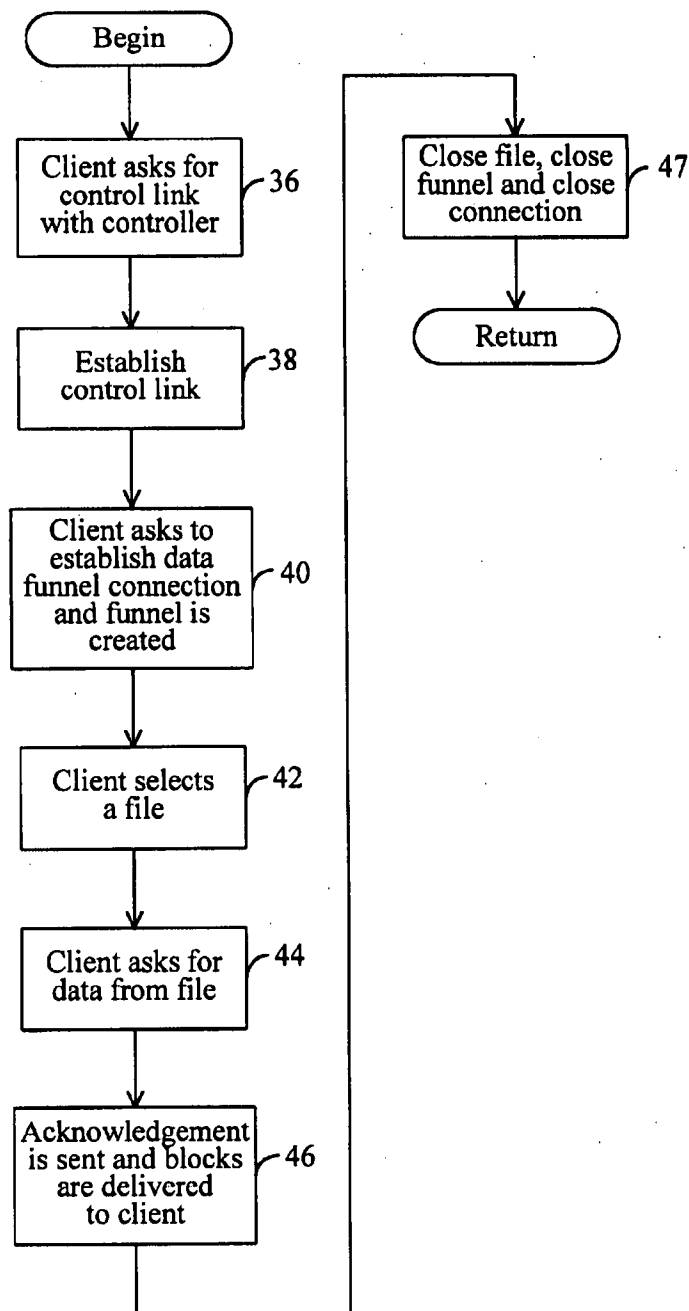


FIG. 5

U.S. Patent

Jan. 15, 2002

Sheet 6 of 8

US 6,339,794 B2

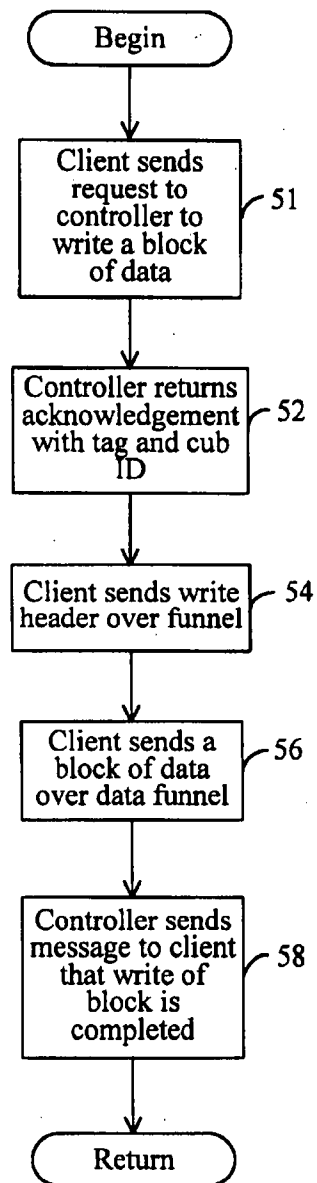


FIG. 6

U.S. Patent

Jan. 15, 2002

Sheet 7 of 8

US 6,339,794 B2

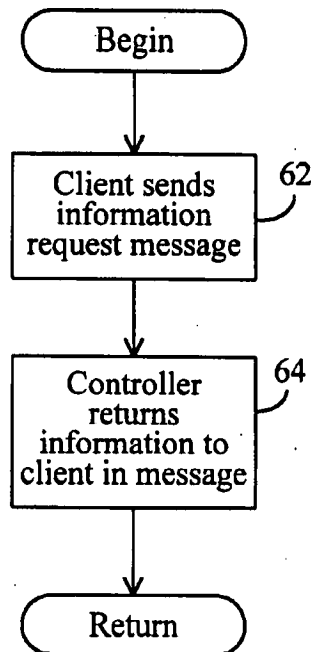


FIG. 7

U.S. Patent

Jan. 15, 2002

Sheet 8 of 8

US 6,339,794 B2

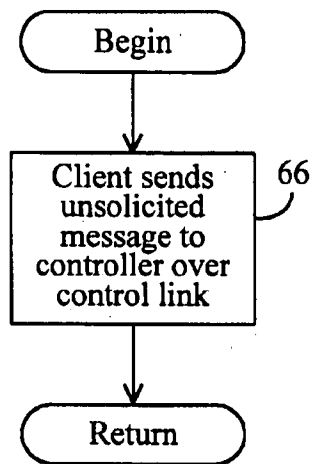


FIG. 8

US 6,339,794 B2

1

WIRE PROTOCOL FOR A MEDIA SERVER SYSTEM

TECHNICAL FIELD

The present invention relates generally to computer systems and more particularly to a wire protocol for communications between a media server system and a client.

BACKGROUND OF THE INVENTION

The use of computer networks has been gaining popularity. Local area networks have become commonplace in business environments, and residential users have begun to connect to computer networks, such as the Internet. Multimedia applications that generate multiple media output, such as audio output and video output, have also been gaining popularity. As such, it is not surprising that there has been an increase in the number of multimedia applications available on computer networks. In general, multimedia data has been transported across computer networks using transport protocols such as TCP/IP, but there has been no protocol present on top of such transport protocols for facilitating efficient and useful communications between clients and multimedia servers.

SUMMARY OF THE INVENTION

The present invention overcomes the limitations of the prior art by adding an additional layer on top of a transfer protocol layer to facilitate communications between a client on a first computer and a media server on a second computer. In accordance with a first aspect of the present invention, a method is practiced in a computer network that has a media server for storing data and a client. Per this method, a wire protocol is provided that facilitates creation of connections between the media server and the client. The wire protocol is utilized to create a control connection between the media server and the client to facilitate exchange of control information. The wire protocol is also used to create a data connection between the media server and the client that facilitates the exchange of data between the media server and the client at a rate substantially equal to a rate at which the data is consumed by the client.

In accordance with another aspect of the present invention, a control connection is created to enable control information to pass between a media server and a client computer in a distributed system that are on separate computers. A data funnel connection is created to enable data to be transferred between the media server and the client at a rate substantially equal to the rate at which the client consumes data.

In accordance with an additional aspect of the present invention, a first request for service is sent from a client to a media server. The first request includes a first identifier that uniquely identifies the first request. A second request for service is also sent from the client to the media server. The second request includes a second identifier that uniquely identifies the second request and that differs from the first identifier. The media server asynchronously services the first request and returns an acknowledgment to the client. The acknowledgment includes the first identifier. The media server asynchronously services the second request and returns an acknowledgment that includes the second identifier.

In accordance with a further aspect of the present invention, a method of decreasing network traffic is practiced in a computer network that has a media server con-

2

nected to a client via a network connection. Multiple messages are batched into a single message at the client. A single message is then sent from the client to the media server. The media server unbatches the multiple messages and processes each of the multiple messages.

In accordance with another aspect of the present invention, a method is practiced in a distributed system that has a media server for storing files holding data of multiple media, and a client for requesting service from the media server. A control connection connects the media server and the client to pass control information, and a data connection connects the media server and the client to pass data. Per the method of this aspect of the present invention, a read request message is sent from the client to the media server over the control connection. The read request message requests that data in a file of multiple media data stored at the media server be read and output to the client. A read request acknowledgment message is sent from the media server to the client over the control connection to acknowledge the read request message. The requested data is then forwarded from the media server to the client over the data connection.

In accordance with yet another aspect of the present invention, a write request message is sent from a client to a media server over a control connection. The write request message requests that data from the client be written into a file at the media server. A write request acknowledgment message is sent from the media server to the client over the control connection to acknowledge the write request message. The data to be written is forwarded from the client to the media server over the data connection, and the forwarded data is written into a file at the media server.

In accordance with a further aspect of the present invention, a computer system is part of a distributed system that has a media server for storing files that hold data of multiple media. The computer system includes a control connection generator for generating a bidirectional control connection between the media server and the computer system. The control connection enables control information to be passed between the media server and the computer system. The computer system also includes a data connection generator for creating a bidirectional data connection between the media server and the computer system. The data connection enables data to be passed between the media server and the computer system.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be described in more detail below relative to the following figures.

FIG. 1 is a block diagram of a distributed environment that is suitable for practicing the preferred embodiment of the present invention.

FIG. 2 is a flowchart illustrating the steps that are performed to send batched messages in accordance with the preferred embodiment of the present invention.

FIG. 3 is a flowchart that illustrates the steps that are performed to establish a control link between a client and a controller.

FIG. 4 is a flowchart that illustrates the steps that are performed to establish a data funnel connection between a client and data servers in a media storage.

FIG. 5 is a flowchart illustrating the steps that are performed for a client to read a file of data stored on a media server system in the preferred embodiment of the present invention.

FIG. 6 is a flowchart illustrating the steps that are performed for a client to write data into a file that is stored on

US 6,339,794 B2

3

the media storage system in the preferred embodiment of the present invention.

FIG. 7 is a flowchart illustrating the steps that are performed to obtain requested information for a client in accordance with the preferred embodiment of the present invention.

FIG. 8 is a flowchart illustrating the step that is performed for a client to unilaterally initiate an action via the wire protocol in accordance with the preferred embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

The preferred embodiment of the present invention provides a wire protocol on top of a transport layer to facilitate communications between a media server system and a client. The wire protocol of the preferred embodiment provides a number of messages that simplify communication between the client and the server and provide functionality that is well-suited for interaction with the media server system. For example, the wire protocol enables multiple network connections to be established between a client and the media server system. In particular, a control link connection may be established to facilitate the communication of control information between the media server system and the client, and a data link connection may be established to facilitate the transfer of data between the client and the server. The wire protocol facilitates multiple requests for service from the server to be concurrently outstanding. These requests are handled in an asynchronous fashion. A unique identification, denoted as an "incarnation," is included with each request and response to disambiguate responses to requests. In other words, the incarnation enables a response to be matched with a request. The wire protocol also enables multiple messages to be batched together in a single message that may be transmitted over the network in a single packet rather than in separate packets, thus reducing network traffic. The preferred embodiment is well adapted for use with data files that contain data of different media. Nevertheless, the present invention may also be used with single medium data files.

FIG. 1 is a block diagram depicting a networked environment 10 that is suitable for practicing the preferred embodiment of the present invention. The networked environment 10 includes a computer system 12 that is connected to a controller 14 for a media server system. The computer 14 may be one of numerous controllers in the system that are provided to enhance fault tolerance and to help in load balancing. The controller 14 controls access to media storage 16 which stores files holding data of multiple media. The computer system 12 is connected to the controller 14 via control link 18. The control link 18 is a bidirectional logical connection that facilitates messages being passed between the controller 14 and the computer system 12. The computer system 12 is also connected to the media storage 16 via a data funnel 20. The data funnel 20 is a bidirectional logical connection that connects the respective media storage managers, denoted as cubs, 22A, 22B and 22C, with the computer system 12. These logical connections are established on top of one or more physical connectors, such as an "ETHERNET" wire, a phone line or fiber optic line. The wire protocol facilitates the creation of the control link 18 and the data funnel 20, as will be described in more detail below. The computer system 12 runs code 26 that constitute a viewer 26 for viewing output that is read from media storage 16. The viewer 26 acts as a client of the multimedia

4

system formed by the controller 14 and the media storage 16. Those skilled in the art will appreciate that the viewer 26 may be part of an application program, part of an operating system, or, alternatively, part of a dynamic link library (DLL) module. The viewer 26 includes support for the wire protocol of the preferred embodiment.

As was mentioned above, the wire protocol of the preferred embodiment of the present invention facilitates multiple network connections to be established to service requests. The first connection is the control link 18, and the second connection is the data funnel 20. The control link 18 uses the TCP/IP protocol to send commands in the form of messages between the viewer 26 and the controller 14, and the data funnel 20 relies upon the UDP protocol to transfer data between the viewer and the controller (although the TCP/IP protocol may be used as well). Nevertheless, those skilled in the art will appreciate that different transport layer protocols may be utilized. The controller 14 and viewer 26 use the UDP datagram protocol to package blocks of data that are sent over the data funnel 20. Other datagram protocols may also be used by the present invention. It should be appreciated that the data funnel 20 is a multipoint-to-point connection that connects each of the cubs 22A, 22B and 22C to the computer system 12. It should be also appreciated that the present invention may include multiple clients and multiple media server systems. A single viewer and a single multimedia server system are depicted in FIG. 1 for purposes of clarity and simplicity.

In the preferred embodiment of the present invention, the cubs 22A, 22B and 22C hold multimedia data that may be played upon a request by a subscriber who uses the computer system 12. A more detailed description of such a multimedia on demand system is described in U.S. Pat. No. 5,473,362, which is explicitly incorporated by reference herein.

Multiple messages may be batched into a single message structure for transmission over the control link 18. A batch of messages starts with a header that contains the length of the batch of messages. The header is followed by a list of messages that are concatenated. Each of the messages begins with a header that describes the size of the message and the type of message. Each message that is sent over the control link 18 has the following format:

```

struct      LinkMessage {
    int      chunkLen;
    int      MID;
}

```

The chunkLen field specifies the length of the message in 8-byte units. The MID field is a message identifier. A message identifier is a numerical value that specifies a message type, where each message type has a unique numerical value associated with it.

FIG. 2 is a flowchart illustrating the steps are performed to batch messages that are sent to the control link 18. First the messages are packed into a single message (step 31 in FIG. 2). The single message is then transmitted over the control link 18 (step 33 in FIG. 2). The recipient of the message then unpacks the message (step 35 in FIG. 2). As noted above, the batch of messages starts with a header that contains the length of the batch. This header is followed by a concatenated list of messages, each of which contains its own header. Each message header identifies the size of the message, and, thus, these headers may be utilized in conjunction with the batch header to unpack the respective messages until no messages remain to be unpacked.

FIG. 3 is a flowchart that illustrates the steps that are performed to realize control link 18 between the viewer 26

US 6,339,794 B2

5

and the controller 14 using the wire protocol of the preferred embodiment of the present invention. The viewer 26 initiates the creation of the control link 18 by requesting a control link connection (step 28 in FIG. 3). In particular, the viewer 26 sends a message to the controller 14 that has the following format.

```

struct LinkViewerToMacConnectMessage
: public LinkMessage {
    int          MacToViewerProtocolRevision;
    int          ViewerToMacProtocolRevision;
    int          blockHole;
    char         subscriberName[]; //length as required
};

```

The protocol revision fields of this message specify protocol revision numbers that identify which version of a protocol is being used. The fields of the message also specify the subscriber name. The controller 14 receives the request message from the viewer, establishes the control link and returns a response to the viewer to inform the viewer of the successful creation of the control link (step 30 in FIG. 3). The message that is returned to the viewer 26 has the following format.

```

struct LinkMacToViewerReportConnectedMessage
: public LinkMessage {
    int          MacToViewerProtocolRevision;
    int          ViewerToMacProtocolRevision;
    Time         blockGroupPlayTime;
    unsigned     blockGroupBlocks;
    unsigned     nMaxOpenFiles;
    unsigned     nBlockMaxBytes;
    unsigned     maxBitRate;
};

```

The blockGroupPlayTime field is of the Time data type and specifies the amount of time it takes a consumer (e.g., viewer 26) of the block of data to render the block of data. It should be noted that the Time data type is a double precision floating point value. Each file of multiple media data is divisible into fixed size blocks. The blockGroupBlocks field specifies the number of blocks in a group. The nMaxOpenFiles field specifies the maximum number of files that may be concurrently opened by a single client on the multimedia server system formed by the controller 14 in media storage 16. The nBlockMaxBytes field specifies the maximum block size in bytes. Lastly, the maxBitRate field specifies the maximum bit rate of transmission of the blocks.

The data funnel 20 is created by passing messages in accordance with the wire protocol of the preferred embodiment of the present invention. FIG. 4 is a flowchart illustrating the steps that are performed to create the funnel connection 20. Initially, the viewer 26 sends a request message to create a funnel to the controller 14 (step 32 in FIG. 4). The request message has the following format.

```

struct LinkViewerToMacConnectFunnelMessage
: public LinkMessage {
    unsigned     maxBlockBytes;
    unsigned     maxFunnelBytes;
    unsigned     funnelMode;
    char         funnelName[]; //length as required
};

```

The maxBlockBytes field specifies the maximum number of bytes in a block that the viewer desires. The maxFunnel-

6

Bytes field specifies the maximum number of bytes per network datagram that is sent across the funnel connection. The funnelMode field identifies the current mode as "read," "write" or "read/write." Lastly, the funnelName field holds the characters and the name of the funnel specifies the type of transport being used.

The controller 14 receives the viewer request message and creates the appropriate data funnel connection (step 34 in FIG. 4). The controller 14 sends a response message back to the viewer 26 to indicate that the funnel has been successfully created. The response message has the following format.

```

struct LinkMacToViewerReportConnectedFunnelMessage
: public LinkMessage {
    char         funnelName[]; //length as required
};

```

As can be seen, the message specifies the name of the funnel. If for some reason, the controller 14 is unable to create the data funnel connection 20, the controller sends a Report-DisconnectedFunnelMessage (which is described in more detail below).

The wire protocol also enables the viewer 26 to request the playing of a data sequence by the multimedia server system on behalf of the viewer so that the multimedia output is delivered from the media storage 16 to the viewer 26. FIG. 5 is a flowchart illustrating the steps that are performed to initiate such playing of a multimedia sequence. Initially, the viewer 26 asks for the creation of a control link with the controller 14 by sending the LinkViewerToMacConnectMessage (described above) to the controller (step 36 in FIG. 5). The controller 14 then establishes the control link 18 (step 38 in FIG. 5). The controller 14 then sends the LinkMacToViewerReportConnectedMessage (described above) to the viewer 26. If the control link 18 is already established, these steps are not necessary. The viewer 26 next asks the controller 14 to establish a data funnel connection 20 by sending the LinkViewerToMacConnectFunnelMessage (described above) to the controller 14. The data funnel connection is created and the LinkMacToViewerReportConnectedFunnelMessage (described above) is sent from the controller 14 to the viewer 26 (step 40 in FIG. 5). These steps need not be repeated if a data funnel connection already exists.

The viewer 26 subsequently selects a file (step 42 in FIG. 5). The selected file must be opened. In order to open the file, the viewer 26 sends a request to open the file to the controller 14. The request message has the following format.

```

struct LinkViewerToMacOpenFileMessage
: public LinkMessage {
    int          playIncarnation;
    char         completeFileName[];
};

```

The playIncarnation field specifies the incarnation for this request. As was discussed above, multiple requests may be concurrently outstanding and the requests are asynchronously handled. As a result, there must be a mechanism in place for matching responses with requests. The incarnation serves as a basis for identifying each request so that responses may be matched with requests.

The controller 14 receives the request to open the file, opens the file and sends a response message. The response message has the following format.

US 6,339,794 B2

7

8

```

struct LinkMacToViewerReportOpenFileMessage
: public LinkMessage {
    Win32Error      dwError;
    int             playIncarnation;
    unsigned        openFileId;
    unsigned        tigerFileId;
    unsigned        blockODiskId;
    unsigned        blockOCubId;
    char            *name;
    MmsFileEntry    entry[1];
};

```

The dwError field specifies an error code which identifies which error occurred, if any, in opening the file. The playIncarnation field holds a play incarnation value. The openFileId field specifies a handle for the file that has been opened. The tigerFileId field specifies an ID associated with the file that has been opened. Different clients may receive different openFileId's for a same file but each will receive the same tigerFileId. The blockODiskId field identifies the Id of the storage disk that holds the first block of the file that has been opened. Similarly, the blockOCubId holds the Id of the cub 22A, 22B and 22C which holds the first block of the file that has been opened. The name field holds the name of the file and the entry field holds a file entry having information about the file.

The viewer 26 must then identify that the file is to serve as the "current file." The "current file" is a variable value that is maintained by the controller 14 to determine to which file subsequent play/stop messages should refer. Thus, as part of the selection of a file, the viewer 26 sends a message that sets a value for the current file variable to be the file of interest. In particular, the viewer 26 sends a message with the following format.

```

struct LinkViewerToMacSetCurrentFileMessage
: public LinkMessage {
    unsigned        openFileId;
};

```

The openFileId field holds a handle that uniquely identifies the file of interest.

Once these steps have been completed, the viewer may ask for data from the file to be played (step 44 in FIG. 5). This viewer 26 sends a message with the following format.

```

struct LinkViewerToMacStartPlayingMessage
: public LinkMessage {
    Time            position;
    int             frameOffset;
    int             playIncarnation;
};

```

The position field specifies a position in the file where the viewer 26 is to begin playing. The frameOffset field is reserved and the playIncarnation field identifies the incarnation value for the play request.

The controller 14 returns a message to specify that playing has begun (see step 46 in FIG. 5). This message takes the following format.

```

struct LinkMacToViewerReportStartedPlayingMessage
: public LinkMessage {
    Win32Error      dwError;
    int             playIncarnation;
    unsigned        tigerFileId;
    unsigned        numFileBlocks;
    unsigned        fileBlockId;
    unsigned        nextCubId;
    unsigned        numCubs;
    char            fileName[ ]; //length as required
};

```

The dwError field specifies which error, if any, has occurred in initiating the playing of the file. The playIncarnation field specifies the play incarnation, and the tigerFileId field holds the tigerFileId for the controller 14. The numFileBlocks field specifies how many blocks are in the file. The fileBlockId field holds the Id for the block at which playing is initiated. The nextCubId field holds the Id of the cub 22A, 22B or 22C which will start playing the first block of the file. The numCubs field specifies the number of cubs 22A, 22B and 22C in the media storage 16. Lastly, the fileName field holds the name for the file that is being played.

All of the above-described messages are passed over the control link 18. The read data from the media storage 16 is passed over the data funnel 20 (see step 46 in FIG. 5). The media storage 16 begins forwarding blocks of the file of data to the viewer 26 over the data funnel 20 (step 46 in FIG. 5). The blocks are delivered asynchronously from the cubs 22A, 22B and 22C of the media storage 16 over the data funnel 20 to the viewer 26.

The messages are transferred as datagrams, where a datagram is a group of one or more packets that logically represent a single message. A packet is a single unit that is transmitted by the network hardware. The size of packets may vary. For example, in an "ETHERNET" network, packets may range in size from about 20 bytes to about 1500 bytes, whereas in an ATM network, the packets may range from 48 bytes to about 64 kilobytes. When a datagram contains more than one packet it is said to be "fragmented" and the packets that make up the datagram constitute "fragments."

In a heterogeneous network, it is possible that different pieces of the network have different maximum packet sizes. The use of datagrams helps to bridge between such networks. An application may control the datagram size, which is logically independent of the packet size. In one embodiment of the present invention, a 528 byte datagram size is chosen because it works well with a given file format on the Internet. Blocks of data are transmitted across the network in frames. For certain transport protocols, a frame is a datagram. In other transport protocols, a frame is an arbitrary size that correlates with the maxFunnelBytes value.

The blocks are transmitted in frames, and each frame includes a compressed funnel header having the following format.

```

struct CompressedFunnelFrameHeader {
    unsigned        frameOffset;
    unsigned        frameLength;
    int             playIncarnation;
    unsigned short  playSequence;
    unsigned short  fileBlockId;
    unsigned        chunkLength;
};

```

60

65

US 6,339,794 B2

9

The frameOffset field specifies the offset at which the data begins relative to the beginning of the block. The frameLength field specifies the length of the frame. The playIncarnation field holds an incarnation for a play request. The playSequence field holds a value that identifies where the block fits into the playing sequence. The fileBlockId field holds a numerical identifier for the block that is held in the payload of the frame and the chunkLength field holds the total amount of data to be sent for the block.

An example helps to illustrate how these fields are utilized. Suppose that a block of size 200 kilobytes is to be sent over the data funnel. The maximum datagram size is 128 kilobytes. The block of data is sent in two datagrams. The first datagram has a frame offset of 0, a frame length of 128 kilobytes, and a chunk length of 200 kilobytes. The first datagram contains the first 128 kilobytes of data in the block. The second frame has a frame offset of 128 kilobytes, a frame length of 72 kilobytes, and a chunk length of 200 kilobytes. The second datagram contains the remaining 72 bytes of the block.

The cubs 22A, 22B and 22C cause the blocks of the file to be delivered at a particular frequency based upon a datagram size being used for the data funnel 20 and the block play time. The blocks are delivered until end of file is reached, assuming no errors or other intervening requests (step 46 of FIG. 5).

The system must then clean up by first closing the file and then closing the funnel and control link connections, respectively (step 47). The file is closed by sending the following message from the viewer 26 to the controller 14.

```

struct LinkViewerToMacCloseFileMessage
: public LinkMessage {
    unsigned    openFileId;
};

```

The message holds the file Id for the file to be closed.

The funnel 20 is closed by sending a disconnect message from the viewer 26 to the controller 14. The disconnect message has the following format.

```

struct LinkViewerToMacDisconnectFunnelMessage
: public LinkMessage {
};

```

The controller 14 receives the disconnect message, disconnects the funnel and returns the following message.

```

struct LinkMacToViewerReportDisconnectedFunnelMessage
: public LinkMessage {
    Win32Error    dwError;
};

```

The dwError field specifies whether an error occurred in disconnecting the data funnel 20. If for some reason, such as a problem in the underlying network, the data funnel 20 closes, the controller 14 generates a disconnected funnel message without a recipient request to close the data funnel from the viewer 26.

The control link 18 is disconnected using mechanisms provided by the TCP/IP protocol. Those skilled in the art will appreciate that the control link 18 and funnel 20 need not be disconnected immediately after the file is no longer playing; rather, these connections may remain intact to be used further.

10

Typically, a file is played until end of file is reached. However, the viewer 26 may terminate the playing of a file by sending the following message.

```

struct LinkViewerToMacStopPlayingMessage
: public LinkMessage {
};

```

When the controller 14 receives this message, the controller terminates the playing of the file so that the blocks of the file are no longer transmitted over the funnel 20.

A file may also stop playing in situations where an error or other event forces the termination of the playing of the file.

The preferred embodiment of the present invention is not limited to playing the whole file but rather facilitates the playing of blocks of a file on a block-by-block basis. In particular, the viewer 26 may request that a particular block or portion of a block of a file be played. The viewer 26 makes a request to play a block by sending the following message to the controller 14 (step 51 in FIG. 6).

```

struct LinkViewerToMacReadBlockMessage
: public LinkMessage {
    unsigned    openFileId;
    unsigned    fileBlockId;
    unsigned    offset;
    unsigned    length;
    unsigned    flags;
    Time        tEarliest;
    Time        tDeadline;
    int         playIncarnation;
    int         playSequence;
};

```

The openFileId field holds the Id for the file from which the block is to be read. The fileBlockId field holds the Id of the block that is to be read. The offset field specifies the offset within the block of the portion requested to be played. The length field specifies the number of bytes to send. If the viewer 26 requests that data beyond the end of the block be sent (because offset length is greater than or equal to block size), the controller 14 reduces the length field so that only valid data will be sent. part of the flags field is used for system debugging information and the remainder is reserved for future use. The tEarliest field specifies the earliest time at which the block may be scheduled to be read, and the tDeadline field specifies the latest time at which the block may be read. The playIncarnation field specifies the incarnation for the request, and the playSequence field specifies where the read block request fits into a sequence of block read requests.

In response to the viewer 26 request, the controller 14 sends an acknowledgment message to the viewer 26 and reads the requested block of information. The acknowledgment message takes the following form.

```

struct LinkMacToViewerReportReadBlockMessage
: public LinkMessage {
    Win32Error    dwError;
    int         playIncarnation;
    int         playSequence;
};

```

US 6,339,794 B2

11

The dwError field specifies an error code which indicates which error, if any, occurred during the reading of the block of the file. The playIncarnation field holds the play incarnation value, and the playSequence field holds the value that identifies where the block fits within a play sequence.

The viewer 26 may also explicitly request the cancellation of one or more read block requests by sending the following message.

```

struct LinkViewerToMacCancelReadBlockMessage
: public LinkMessage {
    int                playIncarnation;
};

```

The cancellation request message specifies the play incarnation associated with the request.

The preferred embodiment of the present invention also enables a viewer 26 to write data to the media storage 16. FIG. 6 is a flowchart illustrating the steps that are performed in such a writing operation. Initially, the viewer 26 sends a request to the controller 14 to write a block of data (step 51 in FIG. 6). It is assumed that the viewer 26 has already allocated a file on the storage media 16. In order to allocate a file, the viewer 26 sends the following message.

```

struct LinkViewerToMacAllocateFileMessage
: public LinkMessage {
    int                playIncarnation;
    char               newName;
    MmsFileEntry       newMmsFileEntry[1];
};

```

The playIncarnation field specifies a play incarnation for allocating the file. The newName field identifies the file name for the new file and the newMmsFileEntry field holds file information.

The controller 14 receives the request to allocate a file, attempts to allocate the file, and sends the following response message:

```

struct LinkMacToViewerReportAllocatedFileMessage
: public LinkMessage {
    Win32Error         dwError;
    int                playIncarnation;
    unsigned            openFileId;
    unsigned            tigerFileId;
    unsigned            block0DiskId;
    unsigned            block0CubId;
};

```

The dwError holds an error code that specifies whether an error occurred and identifies any error that did occur. The playIncarnation field specifies a play incarnation value. The openFileId field holds a handle for the file that has been allocated. The tigerFileId holds an id for the file. The block0DiskId field holds the id of the disk that holds the first block of the file that has been allocated. Lastly, the block0CubId field holds the value of the id for the cub that holds block 0 of the allocated file.

Once the file is allocated and opened, the viewer 26 may request that a block be written to the file by sending the following message.

12

```

struct LinkViewerToMacWriteBlockRequestMessage
: public LinkMessage {
    unsigned            openFileId;
    unsigned            fileBlockId;
    unsigned            numBlockBytes;
    int                playIncarnation;
};

```

The openFileId field specifies the file Id for the file to which the block of data is to be written. The fileBlockId field holds an Id for the file block that is to be written. The numBlockBytes field specifies the number of bytes in the block, and the playIncarnation field holds the incarnation value for the write operation. The controller 14 sends a message to the cubs 22A, 22B and 22C to prepare for the data to be written. The controller 14 returns an acknowledgment to the viewer 26 that contain the tag and an identifier for the cub that holds the file to which the block of data is to be written (step 52 in FIG. 6). The acknowledgment message has the following format.

```

struct LinkMacToViewerReportWriteBlockRequestedMessage
: public LinkMessage {
    Win32Error         dwError;
    int                playIncarnation;
    unsigned            operationTag;
    unsigned            cubId;
    Time               wbStamps [1+WBStampRequestedOnTiger];
};

```

The dwError field holds an error code that either identifies an error or indicates that no error occurred during the request for the write block. The playIncarnation field holds a value for the play incarnation. The operationTag identifies the operation being performed because multiple operations may be performed at the same time. The CubId field identifies the cub to which the block is to be sent, and the final field is a set of time stamps.

The viewer 26 then sends a funnel write data header over the funnel 20 to the appropriate cub (step 54 in FIG. 6). The funnel write data header has the following format.

```

struct FunnelWriteDataHeader {
    unsigned            operationTag;
    int                playIncarnation;
    unsigned            numBlockBytes;
};

```

The operationTag field specifies a tag that identifies the operation to differentiate it from other operations. The playIncarnation field holds the current play incarnation value and the numBlockBytes field holds the number of blocks being sent in the write block. The viewer sends the block of data over the funnel to the appropriate cub (step 56 in FIG. 6). When the writing is completed, the controller 14 sends a message to the viewer 26 indicating that the write of the block is completed (step 58 in FIG. 6). This acknowledgment message has the following format.

US 6,339,794 B2

13

```

struct LinkMacToViewerReportWriteBlockCompletedMessage
: public LinkMessage {
    Win32Error      dwError;
    int             playIncarnation;
    Time            wbStamps [1+WBSstampWrittenOnTiger];
};

```

The acknowledgment specifies an error code, a play incarnation and a set of time stamps.

The protocol also facilitates the viewer 26 sending a request to obtain information from the controller 14. FIG. 7 is a flowchart of the basic steps that are performed. Initially, the viewer 26 sends an information request message to the controller 14 over the control link 18 (step 62 in FIG. 7). The controller 14 then returns information to the viewer in the form of a response message (step 64 in FIG. 7).

In order to understand the utility of the steps shown in FIG. 7, it is helpful to review some of the messages that may be sent to request information and to provide requested information. For example, the viewer 26 may request information about a particular controller 14 by sending the following message.

```

struct LinkViewerToMacTigerInfoMessage
: public LinkMessage {
    char            tigerName[ ]//length as required
};

```

The viewer 26 may also request information about the funnel 20 by sending the following message.

```

struct LinkViewerToMacFunnelInfoMessage
: public LinkMessage {
};

```

The controller 14 receives the request from the viewer 26 and returns the following message.

```

struct LinkMacToViewerReportFunnelInfoMessage
: public LinkMessage {
    unsigned        transportMask;
    unsigned        nBlockFragments;
    unsigned        fragmentBytes;
    unsigned        nCubs;
    unsigned        failedCubs;
    unsigned        nDisks;
    unsigned        decluster;
    unsigned        cubddDatagramSize;
};

```

The transportMask field is a bit mask that specifies which transports are supported. The nBlockFragments field specifies the number of fragments that may be in a block (in this context "fragments" refers to portions of the data block on secondary storage). The fragmentBytes field specifies the number of bytes in each fragment. The nCubs field specifies the number of cubs in the media storage 16 that are connected via the data funnel 20, and the failedCubs field is a bit mask that specifies whether any of the cubs have failed or not. The nDisks field specifies the number of disks in the media storage 16. The decluster field specifies how information is mirrored in the media storage 16. Lastly, the

14

cubddDatagramSize field specifies the datagram size that is utilized by the cubs 22A, 22B and 22C.

The viewer 26 may request information about a particular file by sending the following message.

```

struct LinkViewerToMacFileInfoMessage
: public LinkMessage {
    int             playIncarnation;
    char            completeFileName[ ]//length as required
};

```

The message specifies a playIncarnation and a FileName. The controller 14 responds by returning the following report message.

```

struct LinkMacToViewerReportFileInfoMessage
: public LinkMessage {
    Win32Error      dwError;
    int             playIncarnation;
    MmsFileEntry    entry[1];
};

```

The report message includes an entry that holds information about the file as well as a playIncarnation value and an error code.

In addition to obtaining information about a file, the viewer 26 may also obtain directory information from the controller 14 by sending a request for directory information having the following format.

```

struct LinkViewerToMacDirectoryEntriesMessage
: public LinkMessage {
    char            *tigerName;
    int             incarnation;
    unsigned        nPatterns;
    unsigned        startingFileId;
    char            *patterns [nPatterns];
};

```

The message specifies the controller (i.e., tiger) in which the directory entries are maintained. An incarnation value for the request is included in the message, and the number of patterns to be searched is specified in the nPatterns field. It should be appreciated that this request does a textual search to look for certain textual patterns among the directory entries. The *patterns[nPatterns] field specifies the patterns that are to be sought. The starting file ID specifies where in a list of files the search is to begin.

The controller 14 receives the request for directory information and returns a report. The report message has the following format.

```

struct LinkMacToViewerReportDirectoryEntriesMessage
: public LinkMessage {
    int             incarnation;
    unsigned        nFiles;
    unsigned        nValid;
    unsigned        nInitialized;
    unsigned        nBlocks;
    unsigned        nFree;
    unsigned        nEntries;
};

```

US 6,339,794 B2

15

-continued

```

int
TigerDirectoryEntry    complete;
                        entries [nEntries];
};

```

The report message includes the incarnation value to delineate this response from other responses and to match up the response with the request. The nFile field specifies the number of files in the system. The nValid field specifies the number of files that are valid, and the nInitialized field specifies the number of files that have been initialized. The nBlocks field specifies the number of blocks on the media server system. The nFree field specifies the number of free blocks and the nEntries field specifies the number of directory entries in this message that match the patterns that were requested. The complete field specifies whether the end of the response to the request for directory entries is contained in the message. Oftentimes the response is too large for one message and must be broken into multiple messages. Lastly, the entries[nEntries] field is an array for each matching directory entry that describes the directory entry.

The viewer 26 may also request a number of administrative functions be performed at the controller 14. For example, the viewer 26 may request that a file be removed from the storage on one of the cubs 22A, 22B and 22C. The viewer initiates a request by sending a message with the following format.

```

struct LinkViewerToMacRemoveFileMessage
: public LinkMessage {
    int                playIncarnation;
    char               fileName[]; //length as required
};

```

This format specifies a play incarnation and a file name. The controller 14 receives the request and attempts to perform the request. The controller 14 then returns a message with the following format.

```

struct LinkMacToViewerReportRemovedFileMessage
: public LinkMessage {
    Win32Error         dwError;
    int                playIncarnation;
};

```

The report message indicates whether the removal of the file was successful and also returns to the play incarnation so as to disambiguate this report message from other report messages.

A viewer may request that a file be renamed. Specifically, the viewer sends a message with the following format.

```

struct LinkViewerToMacRenameFileMessage
: public LinkMessage {
    int                playIncarnation;
    char               *newName;
    char               oldName[]; //length as required
};

```

This message includes the old name of the file, the new name of the file to which the file is to be renamed and a play incarnation value. The controller 14, in response, attempts to

16

rename to the file and sends a report message having the following format.

```

struct LinkMacToViewerReportRenamedFileMessage
: public LinkMessage {
    Win32Error         dwError;
    int                playIncarnation;
};

```

The report message specifies whether an error occurred and returns a play incarnation value.

A viewer 26 may also request that a file be initialized so that the file is at a state that is ready to be played. The viewer 26 sends such a request by sending a message with the following format.

```

struct LinkViewerToMacInitializeFileMessage
: public LinkMessage {
    int                playIncarnation;
    unsigned           openFileId;
};

```

The request message includes a play incarnation value and a file ID for the file that is to be initialized. The controller 14 responds by attempting to initialize the file and returning a request message that specifies whether the initialization was successful or not. The response message has the following format.

```

struct LinkMacToViewerReportInitializedFileMessage
: public LinkMessage {
    Win32Error         dwError;
    int                playIncarnation;
};

```

As shown in FIG. 8 the viewer may also send messages over their control link 18 that prompt no report message in return (step 66 in FIG. 8). One example of such a message is a message the viewer 26 sends to the controller 14 to indicate that the viewer did not receive a block that was transmitted over the data funnel 20. This message is especially useful because many protocols, such as UDP, cannot guarantee arrival of a data block. The message the viewer 26 sends has the following format.

```

struct LinkViewerToMacReportLostBlockMessage
: public LinkMessage {
    int                scheduled;
    BufferDataHeader    header[1];
};

```

The scheduled field specifies whether the block was scheduled as a block read or scheduled as part of a file read. The header field identifies the block.

The viewer may also send a message indicating that the block that was transmitted was damaged. The viewer 26 indicates such a damaged block by sending the following message to the controller 14.

US 6,339,794 B2

17

```

struct LinkViewerToMacReportDamagedBlockMessage
: public LinkMessage {
    BufferDataHeader    header[1];
};

```

This message includes the data header for the block that was damaged.

While the present invention has been described with reference to a preferred embodiment thereof, those skilled in the art will appreciate that various changes in form and detail may be made without departing from the intended scope of the present invention as defined in the appended claims.

What is claimed is:

1. In a computer network having a client on a first computer and a media server for storing data on a second computer, a method comprising the steps of:

providing a wire protocol that facilitates creation of connections between the media server and the client; using the wire protocol to create a control connection between the media server and the client to facilitate exchange of control information between the media server and the client, the control connection utilizing a first transport protocol; and

using the wire protocol to create a data connection between the media server and the client to facilitate the exchange of data between the media server and the client at a rate based upon a rate at which the client consumes the data, the data connection using a second transport protocol distinct from the first transport protocol.

2. The method of claim 1 wherein the step of providing the wire protocol includes providing formats for messages in the wire protocol.

3. The method of claim 1, further comprising the step of using the control connection to exchange control information between the media server and the client.

4. The method of claim 1 wherein the media server includes multiple data servers and wherein the step of using the wire protocol to create the data connection includes creating a multipoint-to-point connection between the data servers and the client.

5. The method of claim 1, further comprising the step of using the data connection to transfer data of multiple media between the media server and the client.

6. The method of claim 1, further comprising the step of using the wire protocol to cause data to be read from the media server and forwarded to the client over the data connection.

7. The method of claim 1 wherein the media server includes storage and wherein the method further comprises the step of using the wire protocol to cause data from the client to be passed over the data connection to the media server to be written on the storage at the media server.

8. The method of claim 1 wherein the step of using the wire protocol to create a control connection includes creating a bidirectional control connection that allows control information to flow from the client to the media server and from the media server to the client.

9. The method of claim 1, further comprising the steps of: sending a request message for service over the control connection from the client to the media server; and asynchronously processing the request message at the media server.

10. The method of claim 9, further comprising the step of asynchronously generating at the media server and sending

18

over the control connection to the client a response message that responds to the request message.

11. The method of claim 1 wherein the first transport protocol provides delivery verification, and wherein the second transport protocol does not provide delivery verification.

12. The method of claim 1 wherein the first transport protocol is TCP, and wherein the second transport protocol is UDP.

13. The method of claim 1 wherein the steps of creating a control connection and creating a data connection create distinct logical connections carried on the same physical connection.

14. In a distributed system having a media server on a first computer for supplying media output and a client on a second computer for requesting the media output from the media server, a method of interconnecting the media server and the client, comprising the steps of:

creating a control connection for enabling control information to pass between the media server and the client, the control connection utilizing a first transport protocol; and

creating a data funnel connection between the media server and the client for transferring data between the media server and the client at a rate based upon a rate at which the client consumes data, the data funnel connection using a second transport protocol distinct from the first transport protocol.

15. The method of claim 14 wherein the media server includes multiple data servers and wherein the data funnel connection is a multipoint-to-point connection that connects at least some of the data servers with the client.

16. The method of claim 14 wherein the media output includes multiple media.

17. The method of claim 14 wherein the step of creating the control connection includes passing a message from the client to the media server requesting the control connection.

18. The method of claim 14 wherein the step of creating the data funnel connection includes passing a message from the client to the media server requesting the data funnel connection.

19. The method of claim 14 wherein the control connection is bidirectional.

20. The method of claim 14 wherein the data funnel connection is bidirectional.

21. The method of claim 14, further comprising the steps of:

sending multiple requests for service from the client over the control connection to the media server such that the multiple requests are concurrently outstanding; and asynchronously servicing the multiple requests for service at the media server.

22. The method of claim 14 wherein the first transport protocol provides delivery verification, and wherein the second transport protocol does not provide delivery verification.

23. The method of claim 14 wherein the steps of creating a control connection and creating a data funnel connection create distinct logical connections carried on the same physical connection.

24. In a distributed environment that includes a media server for providing multiple media output to a client wherein said client is connected to the media server via a network connection, a method comprising the steps of:

sending a first request for service from the client to the media server wherein said first request includes a first identifier that uniquely identifies the first request;

US 6,339,794 B2

19

sending a second request for service from the client to the media server wherein said second request includes a second identifier that uniquely identifies the second request and wherein the second identifier differs from the first identifier;

at the media server, asynchronously servicing the first request to provide multiple media output specified by the first request to the client and returning an acknowledgment to the client that includes the first identifier to specify that servicing of the first request has begun; and

at the media server, asynchronously servicing the second request to provide multiple media output specified by the second request to the client concurrently with providing multiple media output specified by the first request and returning an acknowledgment to the client that includes the second identifier to specify that processing of the second request has begun, such that, for a period of time, the media server is providing multiple media output as specified by both the first and second requests.

25. In a distributed system having a media server for storing files holding data of multiple media, a client for requesting service from the media server, a control connection between the media server and the client for passing control information and a data connection for passing data between the media server and the client, a method comprising the steps of:

sending a read request message from the client to the media server over the control connection using a transport protocol that provides delivery verification, wherein said read request message contains a request identifier uniquely identifying the request requests that data in a file of multiple media data stored at the media server be read and input to the client;

sending a read request acknowledgment message containing the request identifier from the media server to the client over the control connection to acknowledge receipt of the read request message; and

forwarding the requested data from the media server to the client over the data connection at a rate based upon a rate at which the client consumes the data using a transport protocol that does not provide delivery verification.

26. The method of claim 25 wherein the requested data is a single fixed size block of data.

27. The method of claim 25 wherein the requested data is an entire file of data.

28. The method of claim 25 wherein the transport protocol used to send the read request message over the control connection is TCP, and wherein the transport protocol used to forward the requested data over the data connection is UDP.

29. The method of claim 25 wherein the control connection over which the read request message is sent and the data connection over which the requested data is forwarded are distinct logical connections that share the same physical connection.

30. In a distributed system having a media server for storing files holding data of multiple media, a client for requesting service from the media server, a control connection between the media server and the client for passing control information between the media server and the client and a data connection for passing data between the media server and the client, a method comprising the steps of:

sending a write request message from the client to the media server over the control connection using a first

20

transport protocol, said write request message requesting that data from the client be written into a file at the media server;

sending a write request acknowledgment message from the media server to the client over the control connection to acknowledge the write request message;

forwarding the data to be written from the client to the media server over the data connection using a second transport protocol distinct from the first transport protocol; and

writing the forwarded data into the file at the media server.

31. The method of claim 30 wherein the first transport protocol provides delivery verification, and wherein the second transport protocol does not provide delivery verification.

32. The method of claim 30 wherein the control connection over which the write request message is sent and the data connection over which the data to be written is forwarded are distinct logical connections that share the same physical connection.

33. In a distributed system having a media server storing files holding data of multiple media, a computer system comprising:

a control connection generator for creating a bidirectional control connection between the media server and the computer system to enable control information to be passed between the media server and the computer system, the control connection utilizing a first transport protocol; and

a data connection generator for creating a bidirectional data connection between the media server and the computer system to enable data to be passed between the media server and the computer system, the data connection using a second transport protocol distinct from the first transport protocol.

34. The computer system of claim 33, further comprising a request generator for generating requests for service from the media server that are passed over the control connection wherein each request includes a unique identifier.

35. The computer system of claim 34 wherein the request generator further comprises a read request generator for generating requests to read data from the files of the media server so that the read data is output over the data connection to the computer system.

36. The computer system of claim 34 wherein the request generator further comprises a write generator for generating requests to write data from the computer system to the media server so that the data written is forwarded over the data connection to the media server and written into a file at the media server.

37. The computer system of claim 33, further comprising a message generator for generating a message that holds multiple messages for transmission over the control connection to the media server.

38. The computer system of claim 33 wherein the first transport protocol provides delivery verification, and wherein the second transport protocol does not provide delivery verification.

39. The method of claim 33 wherein the control connection generator and the data connection generator generates distinct logical connections that are carried on the same physical connection.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE
CERTIFICATE OF CORRECTION

PATENT NO. : 6,339,794 B2
DATED : January 15, 2002
INVENTOR(S) : Bolosky 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 10,

Line 11, replace "he" with -- the --.

Column 12,

Line 41, replace "Cubld" with -- cubld --.

Column 18,

Line 20, replace the second "the" with -- a --.

Line 24, replace the first "the" with -- a --.

Line 25, replace "the" with -- a --.

Line 26, insert -- the -- between "consumes" and "data,".

Column 19,

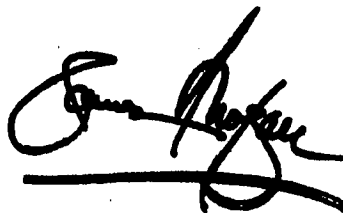
Line 60, replace "for" with -- or --.

Line 62, delete the first "the".

Line 62, replace the second "the" with -- a --.

Signed and Sealed this

Eleventh Day of February, 2003

A handwritten signature in black ink, appearing to read "James E. Rogan", written over a horizontal line.

JAMES E. ROGAN
Director of the United States Patent and Trademark Office