US 20190172013A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2019/0172013 A1**

JACOBSON (43) **Pub. Date: Jun. 6, 2019**

(54) **CLASSIFYING A PORTION OF USER CONTACT DATA INTO LOCAL CONTACTS**

(71) Applicant: **OATH INC.**, New York, NY (US)

(72) Inventor: **Joshua Robert Russell JACOBSON**, San Francisco, CA (US)

(21) Appl. No.: **16/239,632**

(22) Filed: **Jan. 4, 2019**

Related U.S. Application Data

(63) Continuation of application No. 13/693,955, filed on Dec. 4, 2012, now Pat. No. 10,192,200.

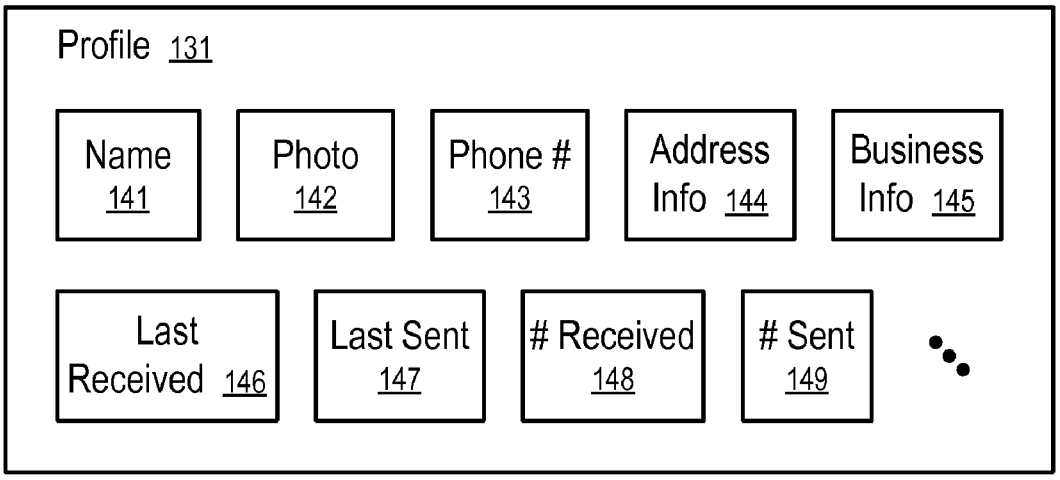
Publication Classification

(51) **Int. Cl.**
G06Q 10/10 (2006.01)

(52) **U.S. Cl.**
CPC ***G06Q 10/10*** (2013.01); *G06F 16/951* (2019.01)

(57) **ABSTRACT**

Contact data for contacts of a user is stored. Each of the contacts is ranked (e.g., based on stored profile information for each contact). In one embodiment, each of the contacts is classified as either a local contact or a directory contact based on the ranking to provide local contacts and directory contacts. The local contacts are provided for local storage on a user device of the user. A directory contact is provided to the user device from the directory contacts by the server (e.g., in response to a query from the user device).



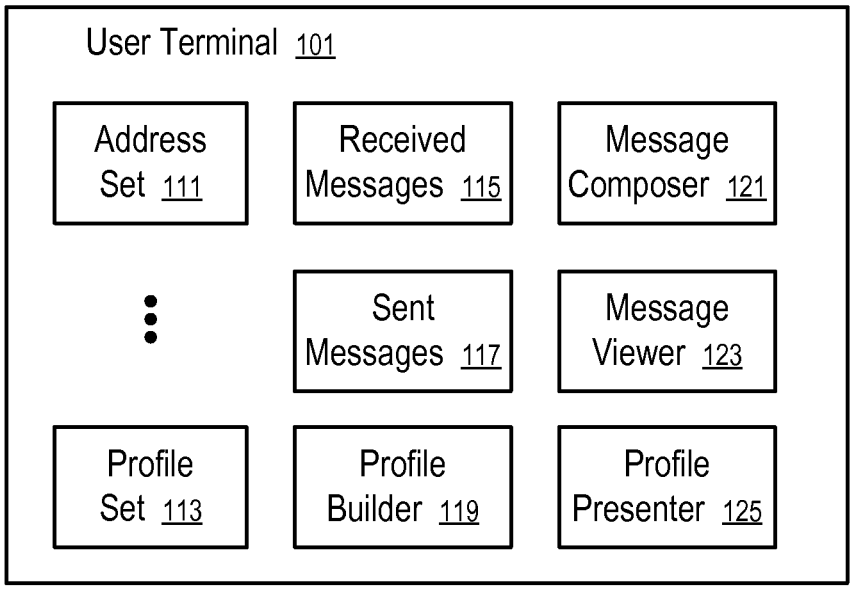

User Terminal 101
Address Set 111
Received Messages 115
Message Composer 121
Sent Messages 117
Message Viewer 123
Profile Set 113
Profile Builder 119
Profile Presenter 125

FIGURE 1

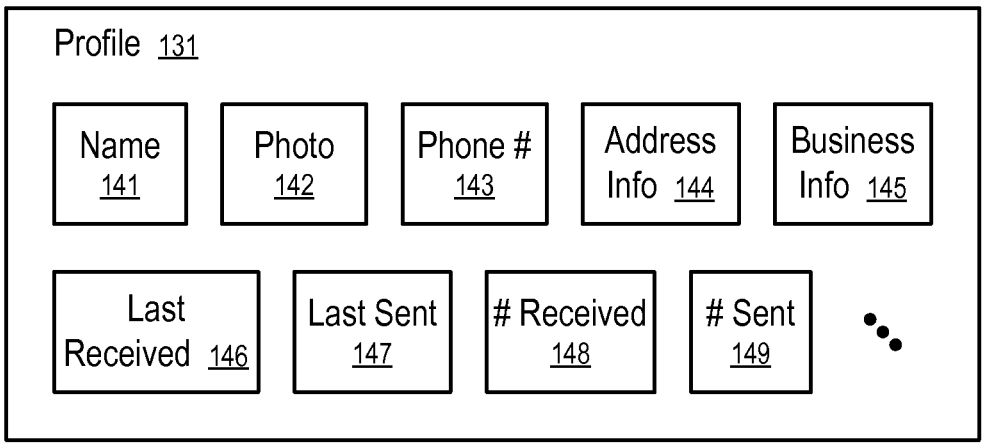

Profile 131
Name 141
Photo 142
Phone # 143
Address Info 144
Business Info 145
Last Received 146
Last Sent 147
Received 148
Sent 149

FIGURE 2

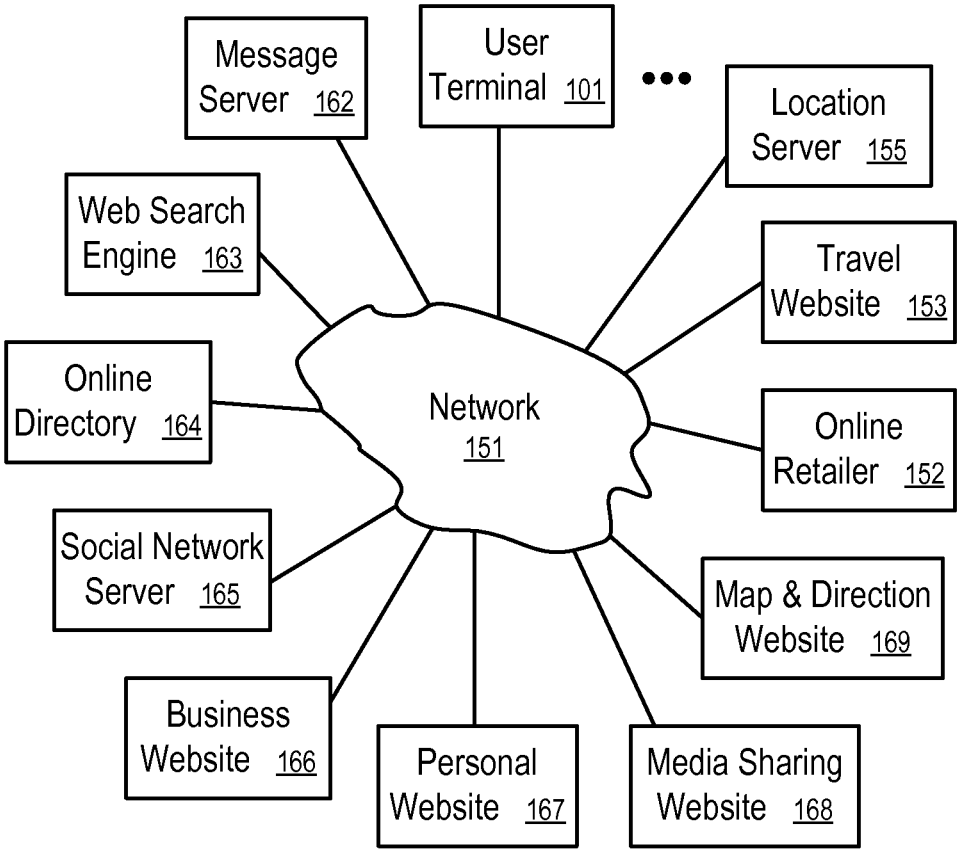

Message Server 162
User Terminal 101
Location Server 155
Web Search Engine 163
Travel Website 153
Online Directory 164
Network 151
Online Retailer 152
Social Network Server 165
Map & Direction Website 169
Business Website 166
Personal Website 167
Media Sharing Website 168

FIGURE 3

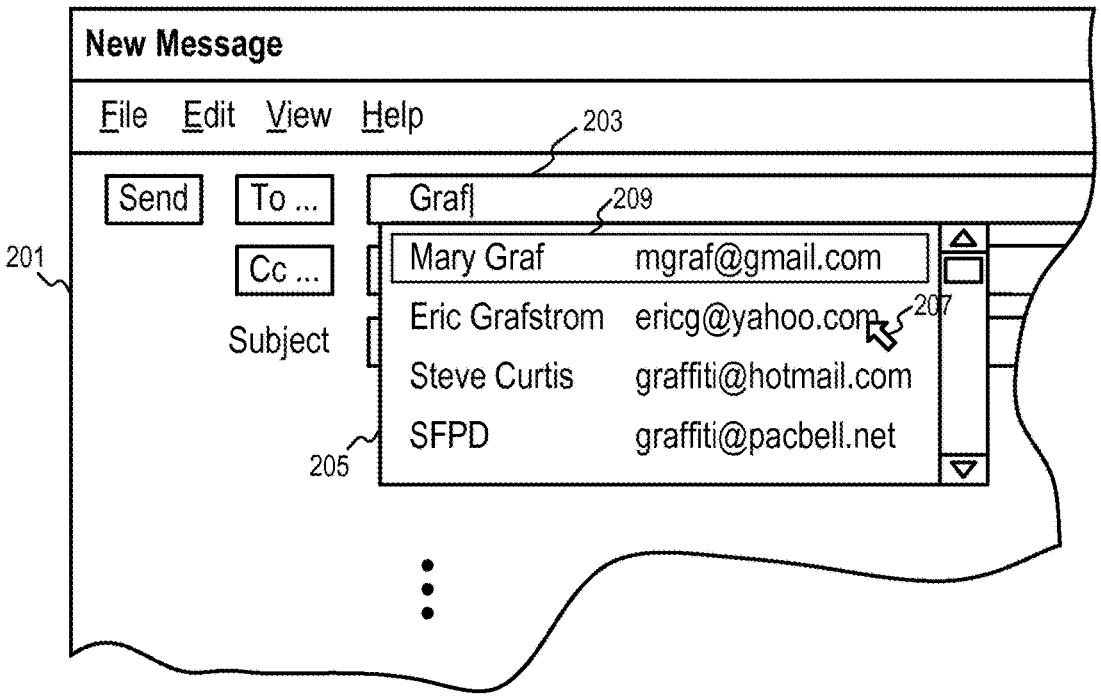

New Message
File Edit View Help
203
Send
To ...
Graf
209
201
Cc ...
Mary Graf mgraf@gmail.com
Eric Grafstrom ericg@yahoo.com
207
Subject
Steve Curtis graffiti@hotmail.com
SFPD graffiti@pacbell.net
205

FIGURE 4

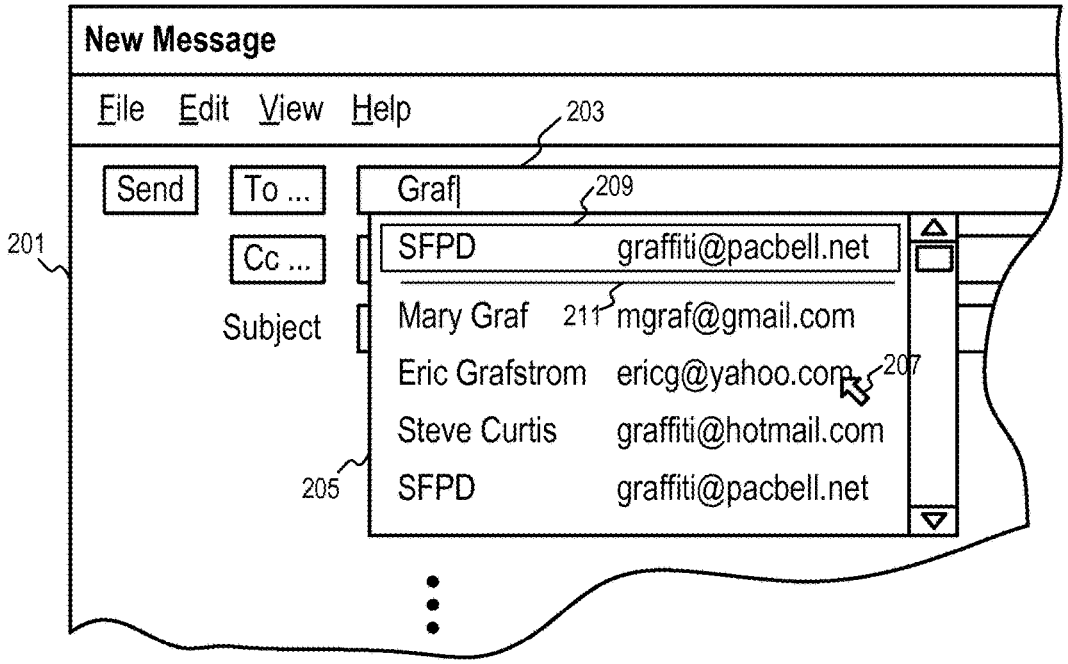

New Message
File Edit View Help
203
Send
To ...
Graf|
209
201
Cc ...
SFPD graffiti@pacbell.net
Subject
Mary Graf 211 mgraf@gmail.com
Eric Grafstrom ericg@yahoo.com
207
Steve Curtis graffiti@hotmail.com
205
SFPD graffiti@pacbell.net

FIGURE 5

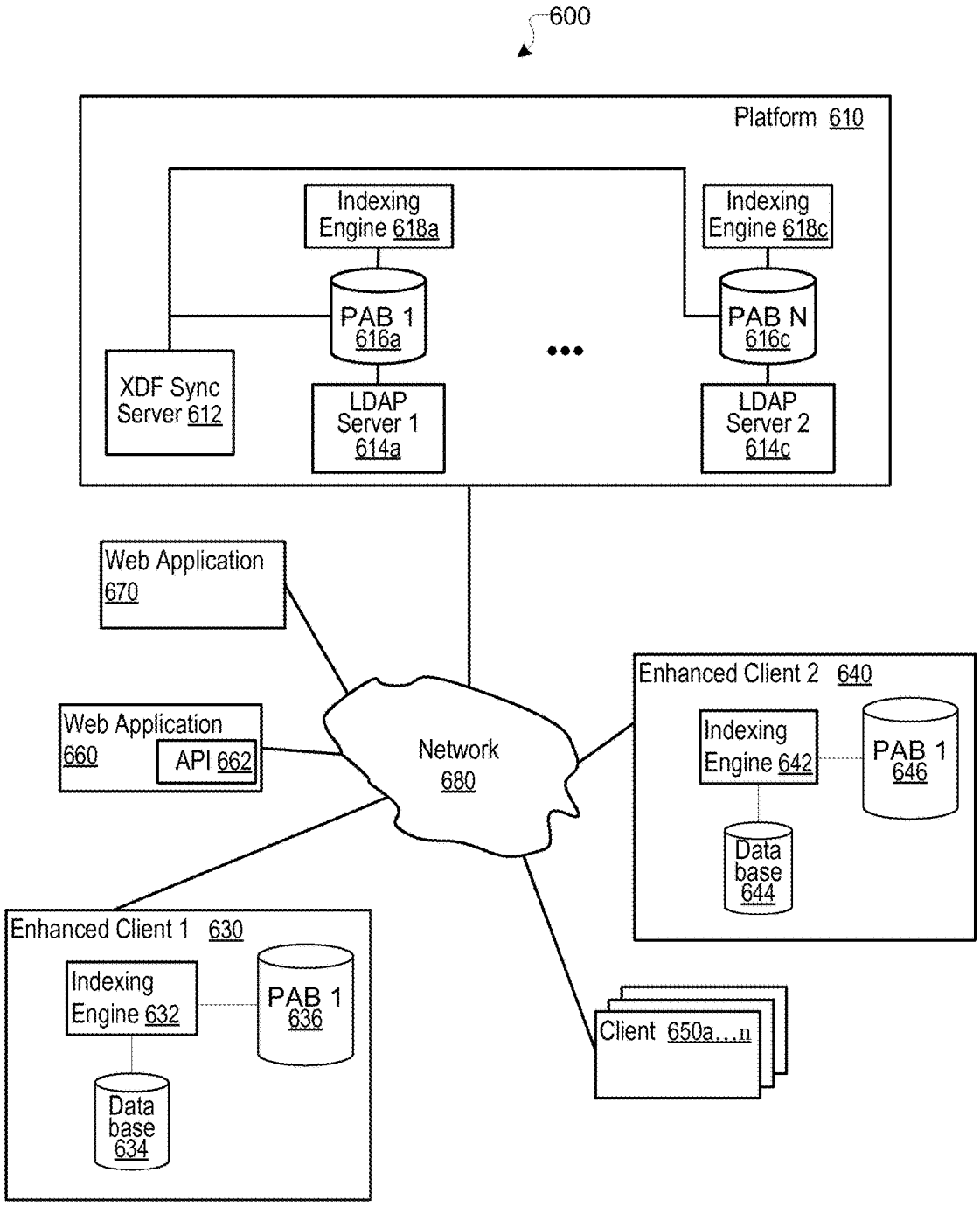

600
Platform 610
Indexing Engine 618a
Indexing Engine 618c
PAB 1 616a
PAB N 616c
XDF Sync Server 612
LDAP Server 1 614a
LDAP Server 2 614c
Web Application 670
Web Application 660
API 662
Network 680
Enhanced Client 2 640
Indexing Engine 642
PAB 1 646
Data base 644
Enhanced Client 1 630
Indexing Engine 632
PAB 1 636
Data base 634
Client 650a...n

FIGURE 6

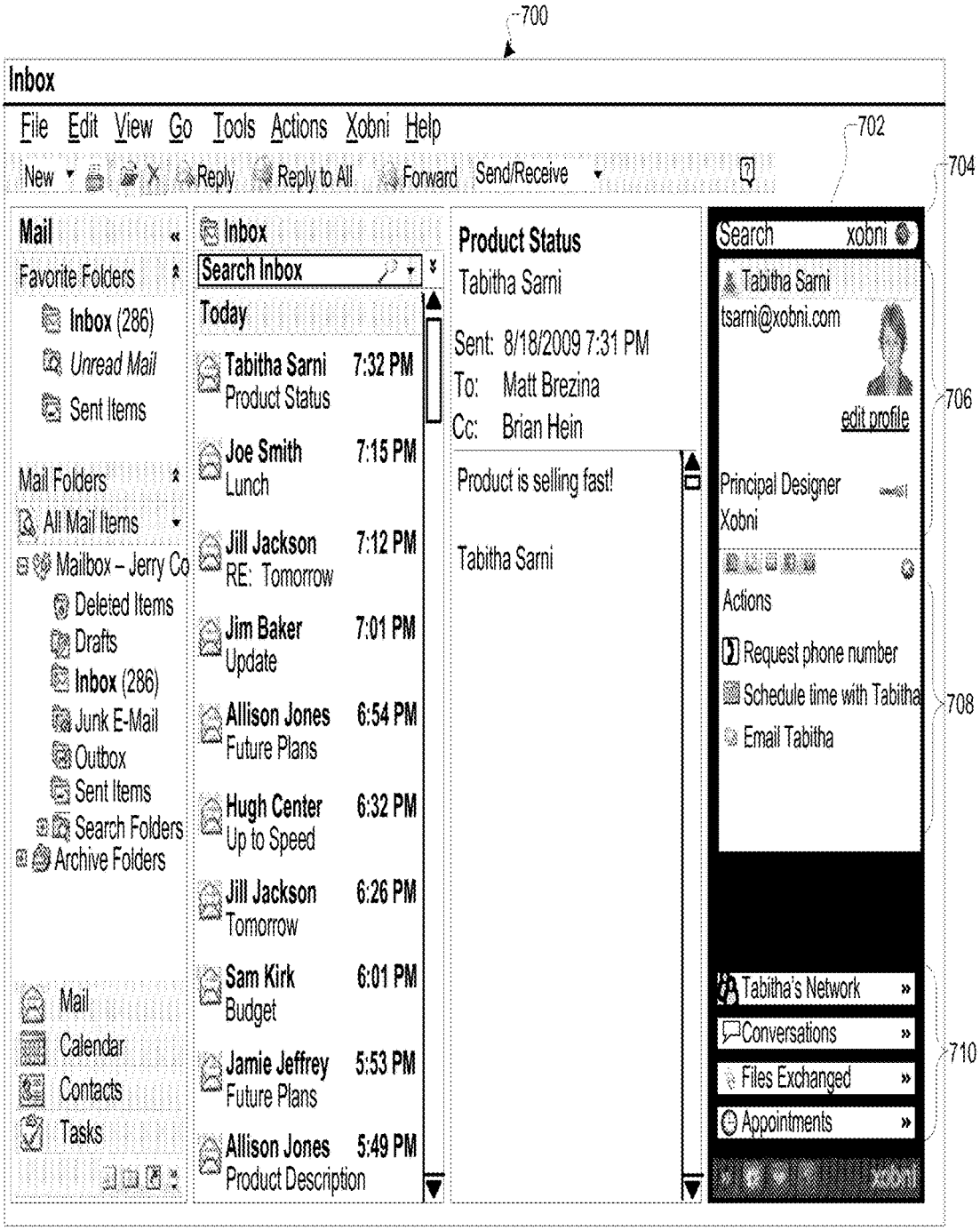

700
Inbox
File Edit View Go Tools Actions Xobni Help
New Reply Reply to All Forward Send/Receive
702
704
Mail
Favorite Folders
Inbox (286)
Unread Mail
Sent Items
Mail Folders
All Mail Items
Mailbox – Jerry Co
Deleted Items
Drafts
Inbox (286)
Junk E-Mail
Outbox
Sent Items
Search Folders
Archive Folders
Mail
Calendar
Contacts
Tasks
Inbox
Search Inbox
Today
Tabitha Sarni 7:32 PM
Product Status
Joe Smith 7:15 PM
Lunch
Jill Jackson 7:12 PM
RE: Tomorrow
Jim Baker 7:01 PM
Update
Allison Jones 6:54 PM
Future Plans
Hugh Center 6:32 PM
Up to Speed
Jill Jackson 6:26 PM
Tomorrow
Sam Kirk 6:01 PM
Budget
Jamie Jeffrey 5:53 PM
Future Plans
Allison Jones 5:49 PM
Product Description
Product Status
Tabitha Sarni
Sent: 8/18/2009 7:31 PM
To: Matt Brezina
Cc: Brian Hein
Product is selling fast!
Tabitha Sarni
Search xobni
Tabitha Sarni
tsarni@xobni.com
edit profile
706
Principal Designer
Xobni
Actions
Request phone number
Schedule time with Tabitha
Email Tabitha
708
Tabitha's Network
Conversations
Files Exchanged
Appointments
710
xobni

FIGURE 7

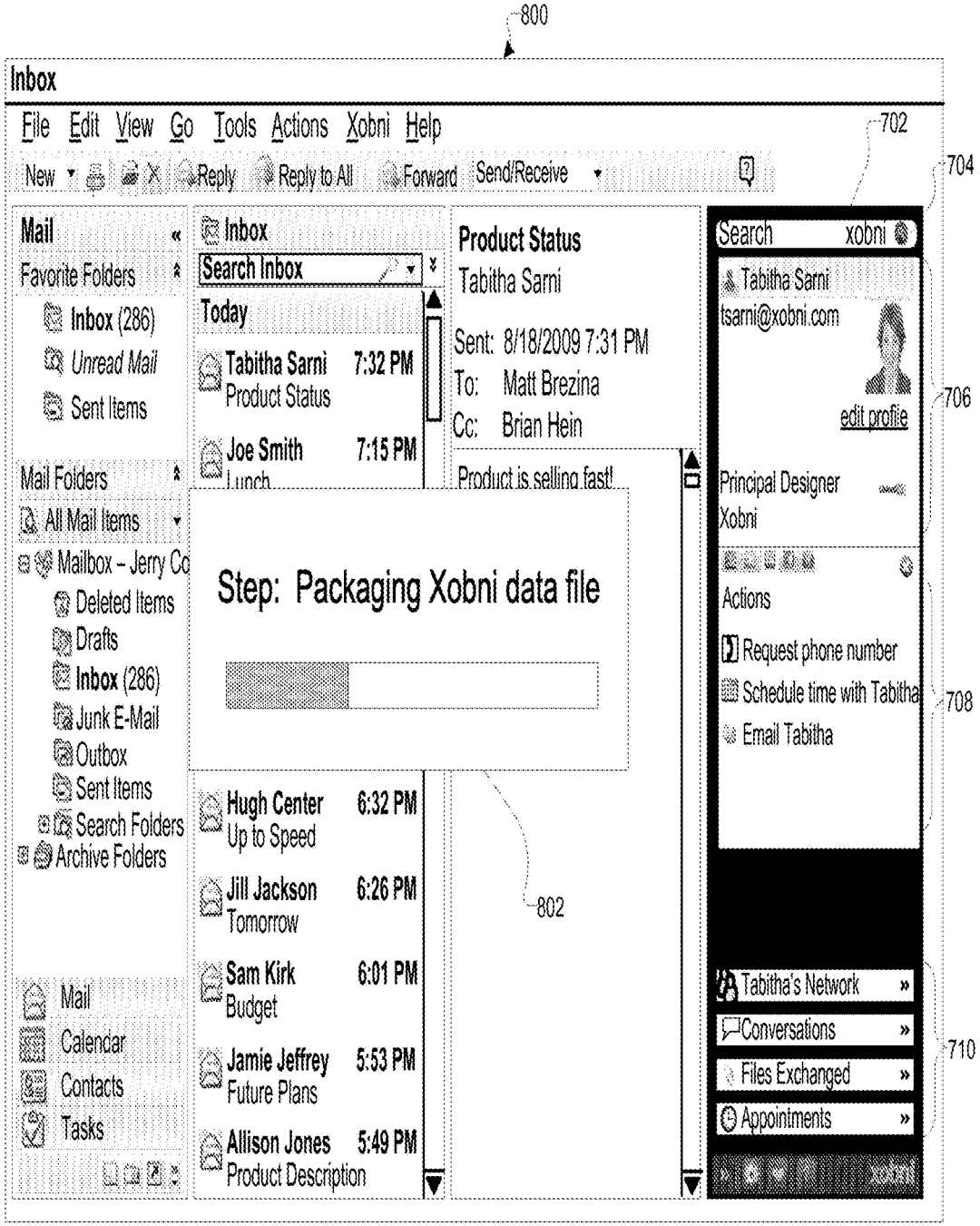

800
Inbox
File Edit View Go Tools Actions Xobni Help
New Reply Reply to All Forward Send/Receive
702
704
Mail
Favorite Folders
Inbox (286)
Unread Mail
Sent Items
Mail Folders
All Mail Items
Mailbox - Jerry Co
Deleted Items
Drafts
Inbox (286)
Junk E-Mail
Outbox
Sent Items
Search Folders
Archive Folders
Mail
Calendar
Contacts
Tasks
Inbox
Search Inbox
Today
Tabitha Sarni 7:32 PM
Product Status
Joe Smith 7:15 PM
Lunch
Hugh Center 6:32 PM
Up to Speed
Jill Jackson 6:26 PM
Tomorrow
Sam Kirk 6:01 PM
Budget
Jamie Jeffrey 5:53 PM
Future Plans
Allison Jones 5:49 PM
Product Description
Product Status
Tabitha Sarni
Sent: 8/18/2009 7:31 PM
To: Matt Brezina
Cc: Brian Hein
Product is selling fast!
Step: Packaging Xobni data file
802
Search xobni
Tabitha Sarni
tsarni@xobni.com
edit profile
706
Principal Designer
Xobni
Actions
Request phone number
Schedule time with Tabitha
Email Tabitha
708
Tabitha's Network
Conversations
Files Exchanged
Appointments
710

FIGURE 8

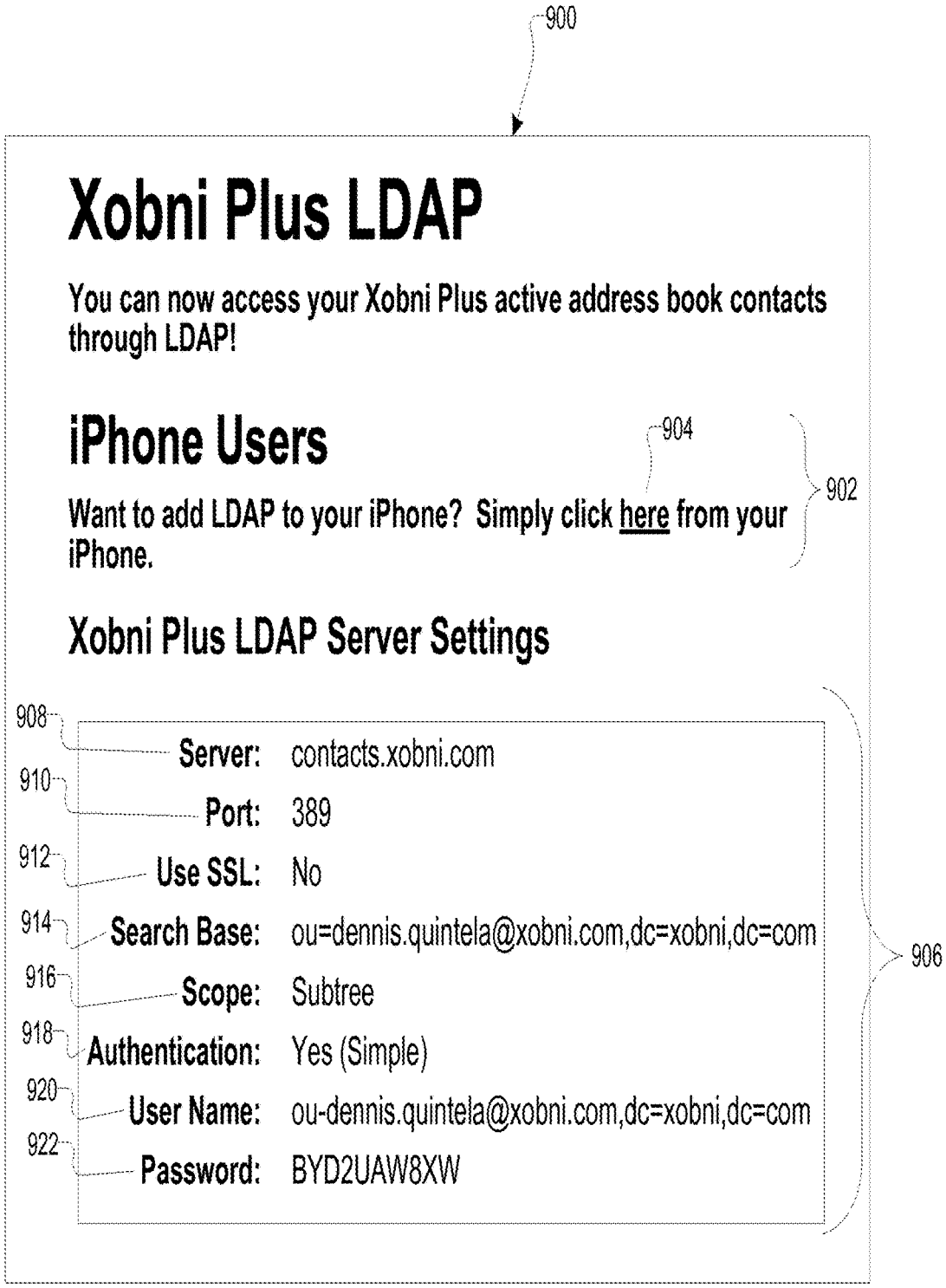

900
Xobni Plus LDAP
You can now access your Xobni Plus active address book contacts through LDAP!
iPhone Users
904
Want to add LDAP to your iPhone? Simply click here from your iPhone.
902
Xobni Plus LDAP Server Settings
908
Server: contacts.xobni.com
910
Port: 389
912
Use SSL: No
914
Search Base: ou=dennis.quintela@xobni.com,dc=xobni,dc=com
916
Scope: Subtree
918
Authentication: Yes (Simple)
920
User Name: ou-dennis.quintela@xobni.com,dc=xobni,dc=com
922
Password: BYD2UAW8XW
906

FIGURE 9

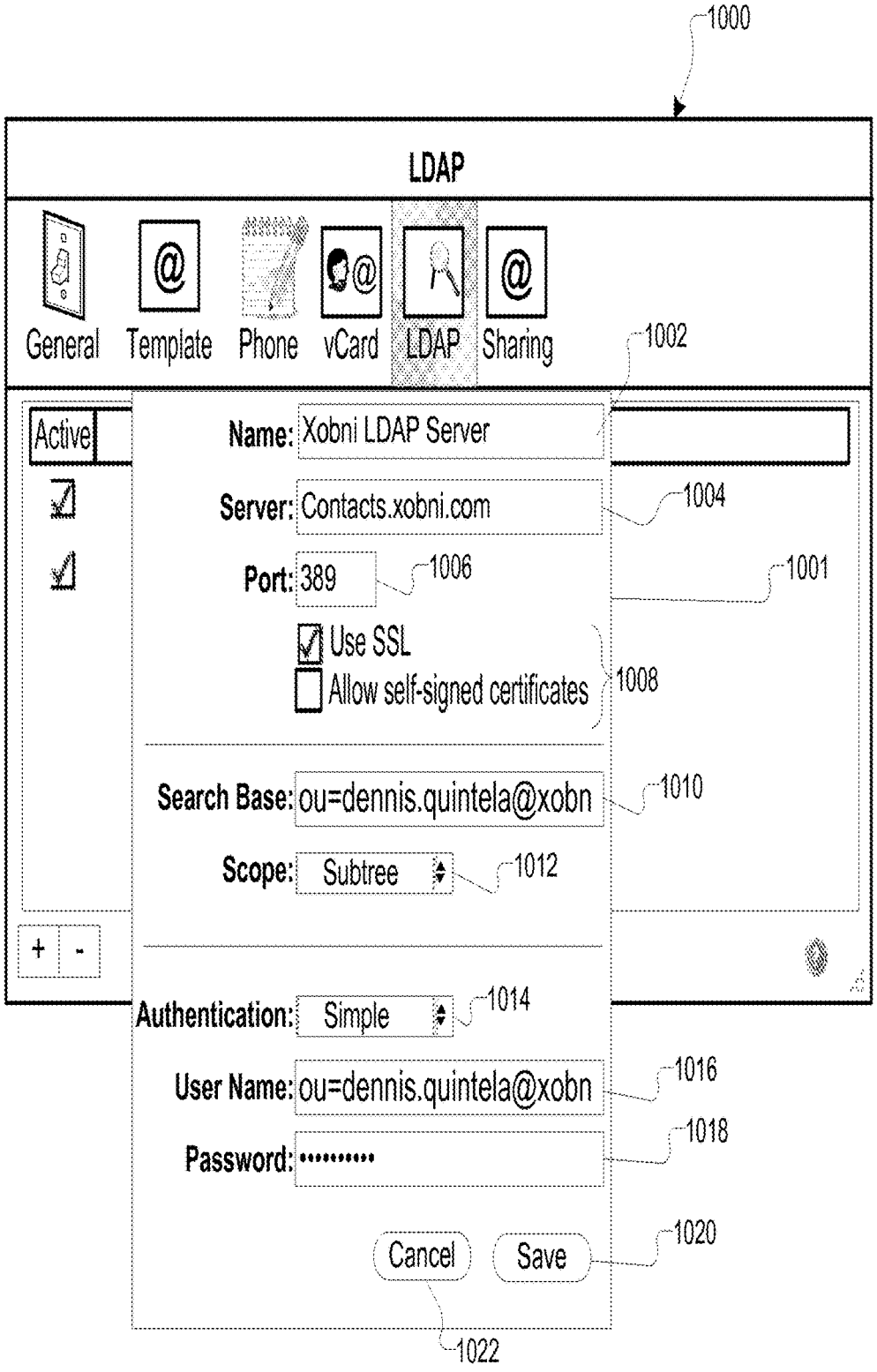

1000
LDAP
General
Template
Phone
vCard
LDAP
Sharing
1002
Active
Name: Xobni LDAP Server
Server: Contacts.xobni.com
1004
Port: 389
1006
1001
Use SSL
Allow self-signed certificates
1008
Search Base: ou=dennis.quintela@xobn
1010
Scope: Subtree
1012
+
-
Authentication: Simple
1014
User Name: ou=dennis.quintela@xobn
1016
Password: ••••••••••
1018
Cancel
Save
1020
1022

FIGURE 10

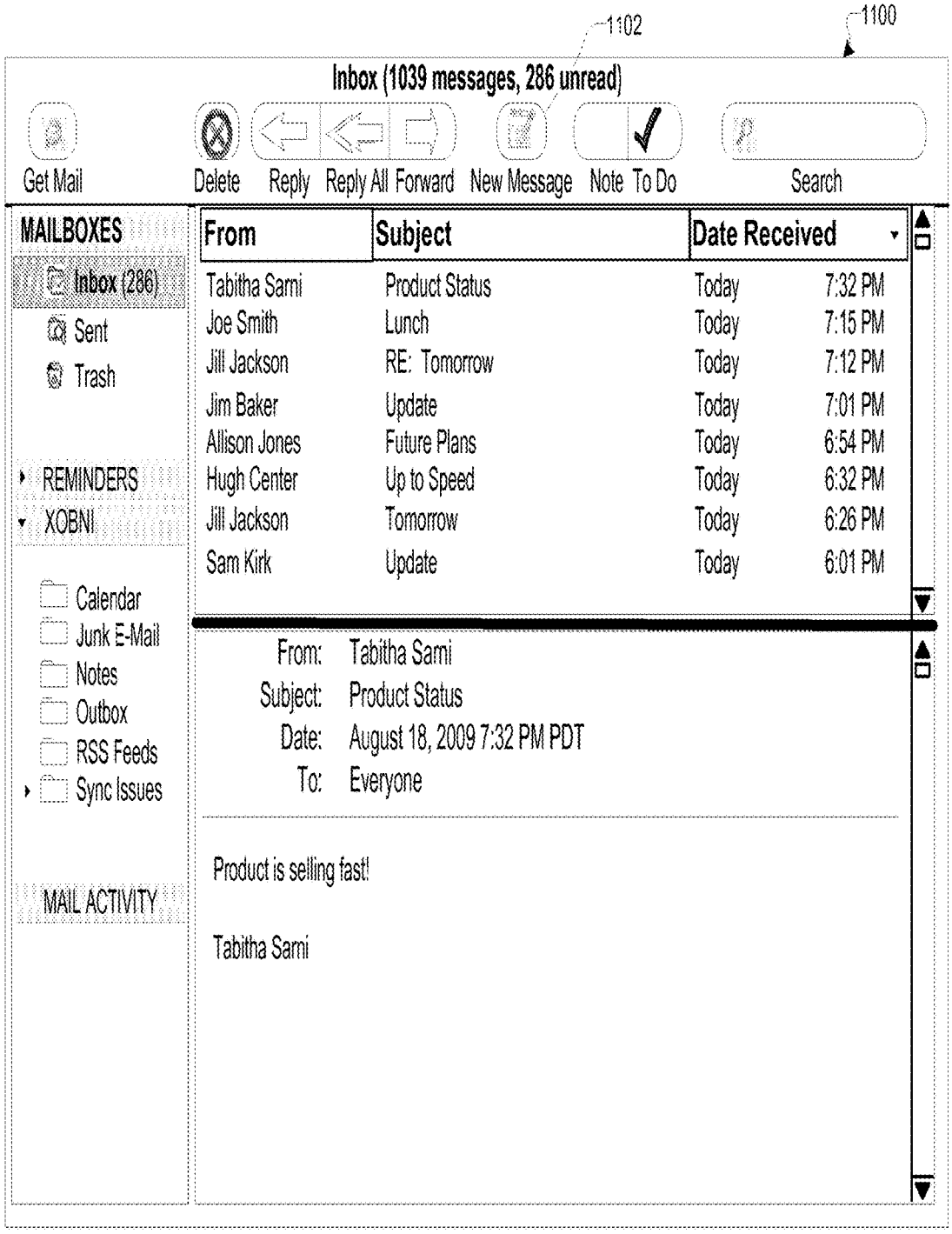

1102
1100
Inbox (1039 messages, 286 unread)
Get Mail
Delete
Reply
Reply All
Forward
New Message
Note
To Do
Search
MAILBOXES
Inbox (286)
Sent
Trash
REMINDERS
XOBNI
Calendar
Junk E-Mail
Notes
Outbox
RSS Feeds
Sync Issues
MAIL ACTIVITY
From
Subject
Date Received
Tabitha Sami Product Status Today 7:32 PM
Joe Smith Lunch Today 7:15 PM
Jill Jackson RE: Tomorrow Today 7:12 PM
Jim Baker Update Today 7:01 PM
Allison Jones Future Plans Today 6:54 PM
Hugh Center Up to Speed Today 6:32 PM
Jill Jackson Tomorrow Today 6:26 PM
Sam Kirk Update Today 6:01 PM
From: Tabitha Sami
Subject: Product Status
Date: August 18, 2009 7:32 PM PDT
To: Everyone
Product is selling fast!
Tabitha Sami

FIGURE 11

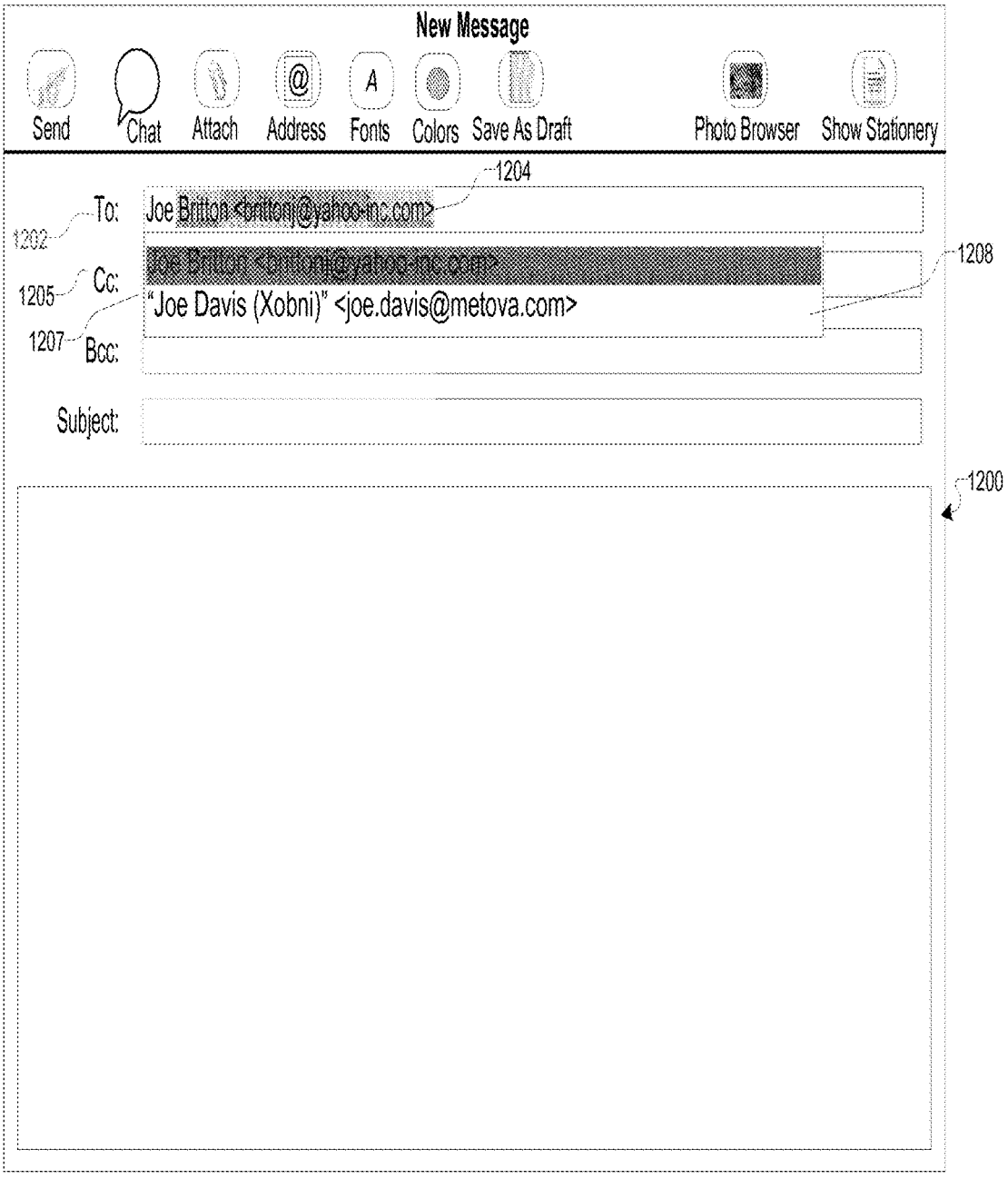

New Message
Send
Chat
Attach
Address
Fonts
Colors
Save As Draft
Photo Browser
Show Stationery
1204
To:
Joe Britton <britonj@yahoo-inc.com>
1202
Cc:
1205
Joe Britton <britonj@yahoo-inc.com>
"Joe Davis (Xobni)" <joe.davis@metova.com>
1208
Bcc:
1207
Subject:
1200

FIGURE 12

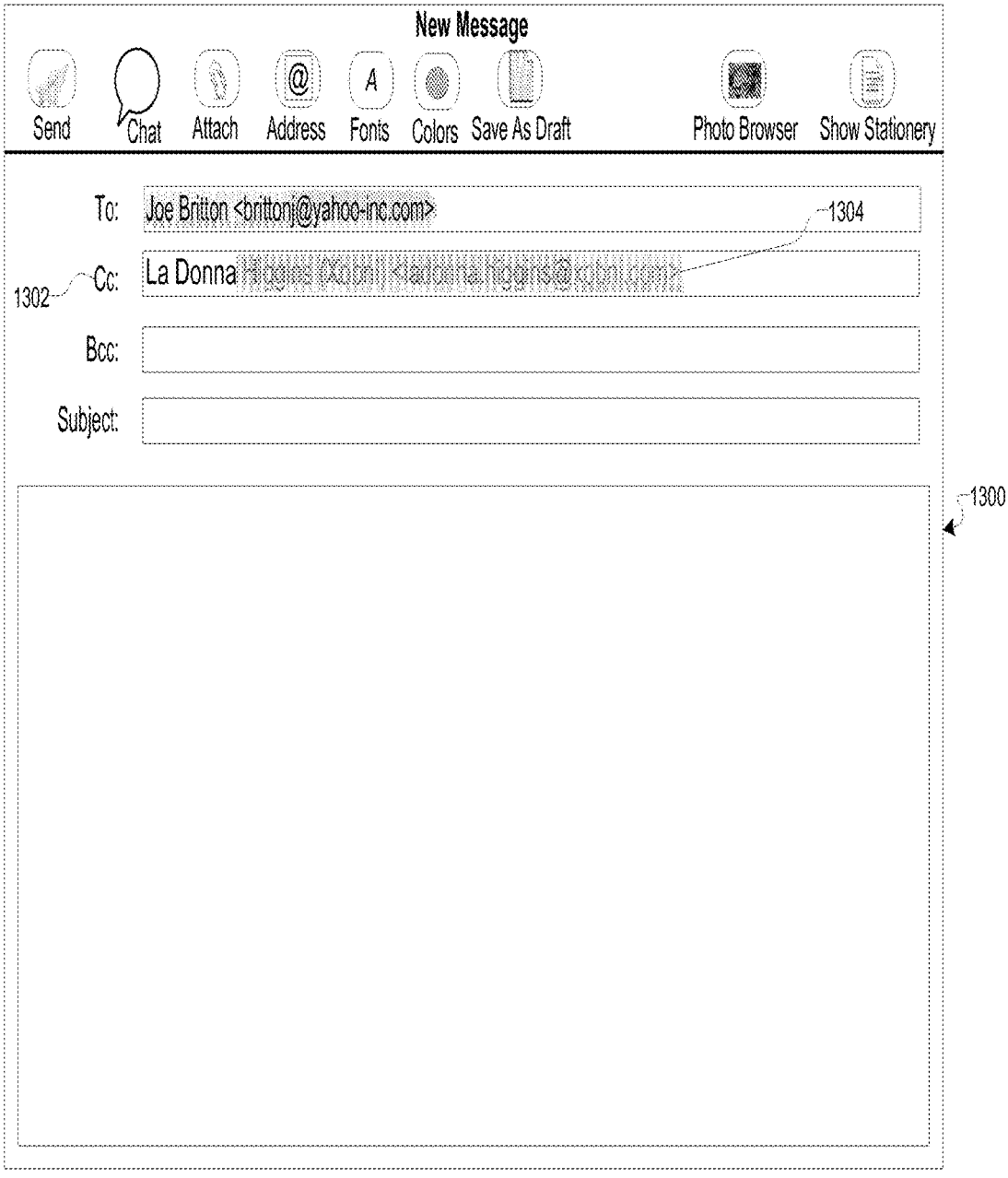

New Message
Send
Chat
Attach
Address
Fonts
Colors
Save As Draft
Photo Browser
Show Stationery
To:
Joe Britton <britonj@yahoo-inc.com>
1304
1302
Cc:
La Donna
Bcc:
Subject:
1300

FIGURE 13

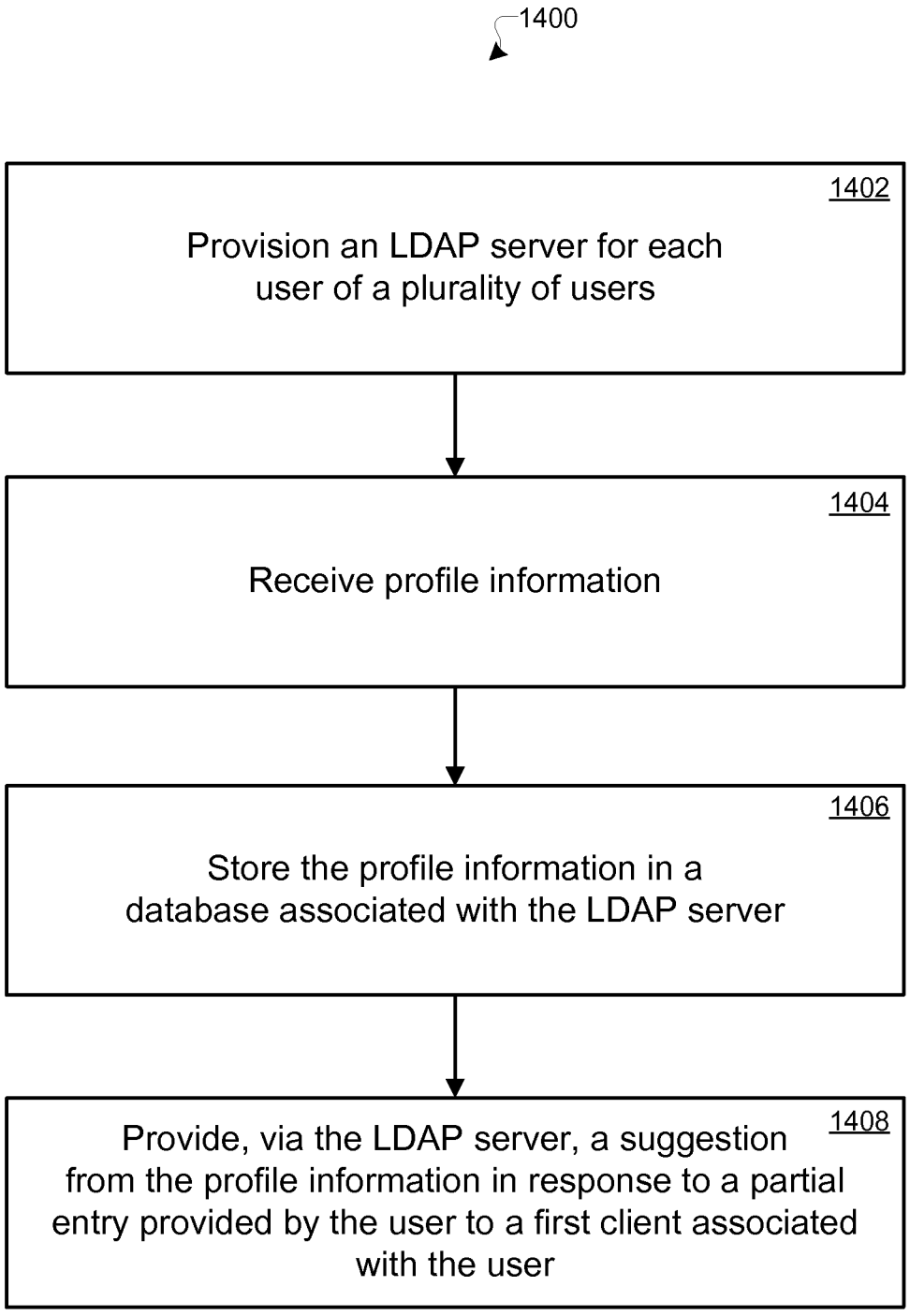

1400
1402
Provision an LDAP server for each user of a plurality of users
1404
Receive profile information
1406
Store the profile information in a database associated with the LDAP server
1408
Provide, via the LDAP server, a suggestion from the profile information in response to a partial entry provided by the user to a first client associated with the user

FIGURE 14

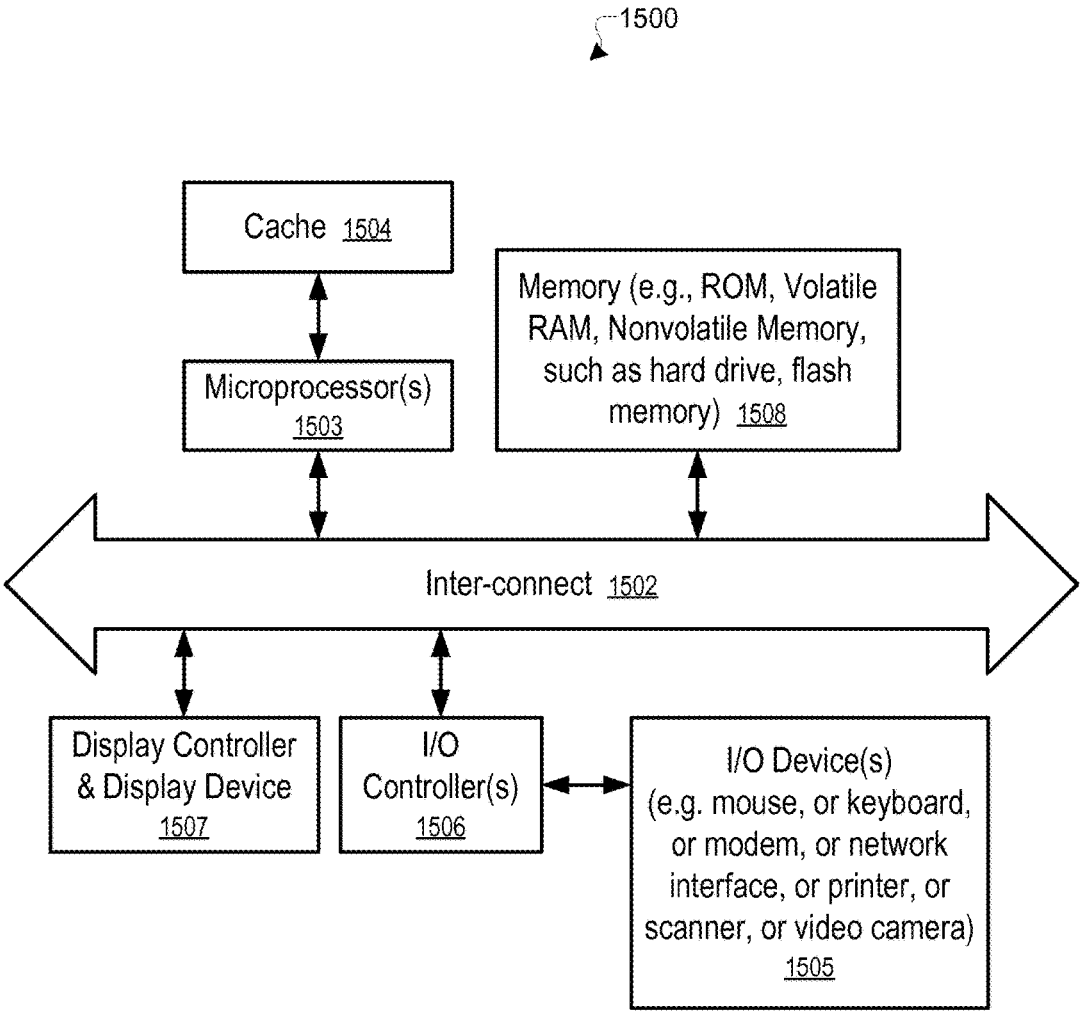

1500
Cache 1504
Microprocessor(s) 1503
Memory (e.g., ROM, Volatile RAM, Nonvolatile Memory, such as hard drive, flash memory) 1508
Inter-connect 1502
Display Controller & Display Device 1507
I/O Controller(s) 1506
I/O Device(s) (e.g. mouse, or keyboard, or modem, or network interface, or printer, or scanner, or video camera) 1505

FIGURE 15

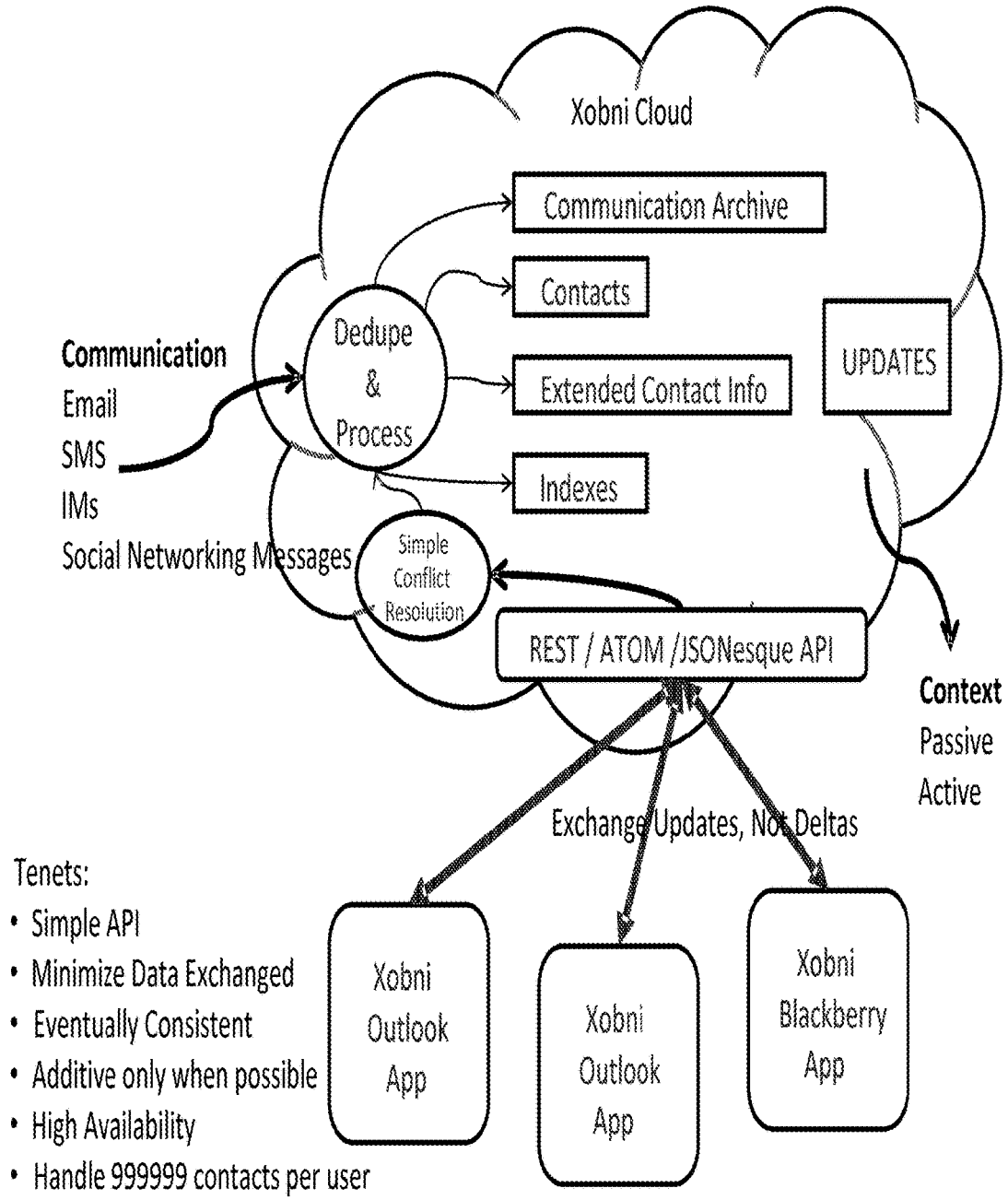

Xobni Cloud
Communication Archive
Contacts
Extended Contact Info
Indexes
Dedupe & Process
UPDATES
Simple Conflict Resolution
REST / ATOM /JSONesque API
Communication
Email
SMS
IMs
Social Networking Messages
Context
Passive
Active
Exchange Updates, Not Deltas
Xobni Outlook App
Xobni Outlook App
Xobni Blackberry App
Tenets:
• Simple API
• Minimize Data Exchanged
• Eventually Consistent
• Additive only when possible
• High Availability
• Handle 999999 contacts per user

FIGURE 16

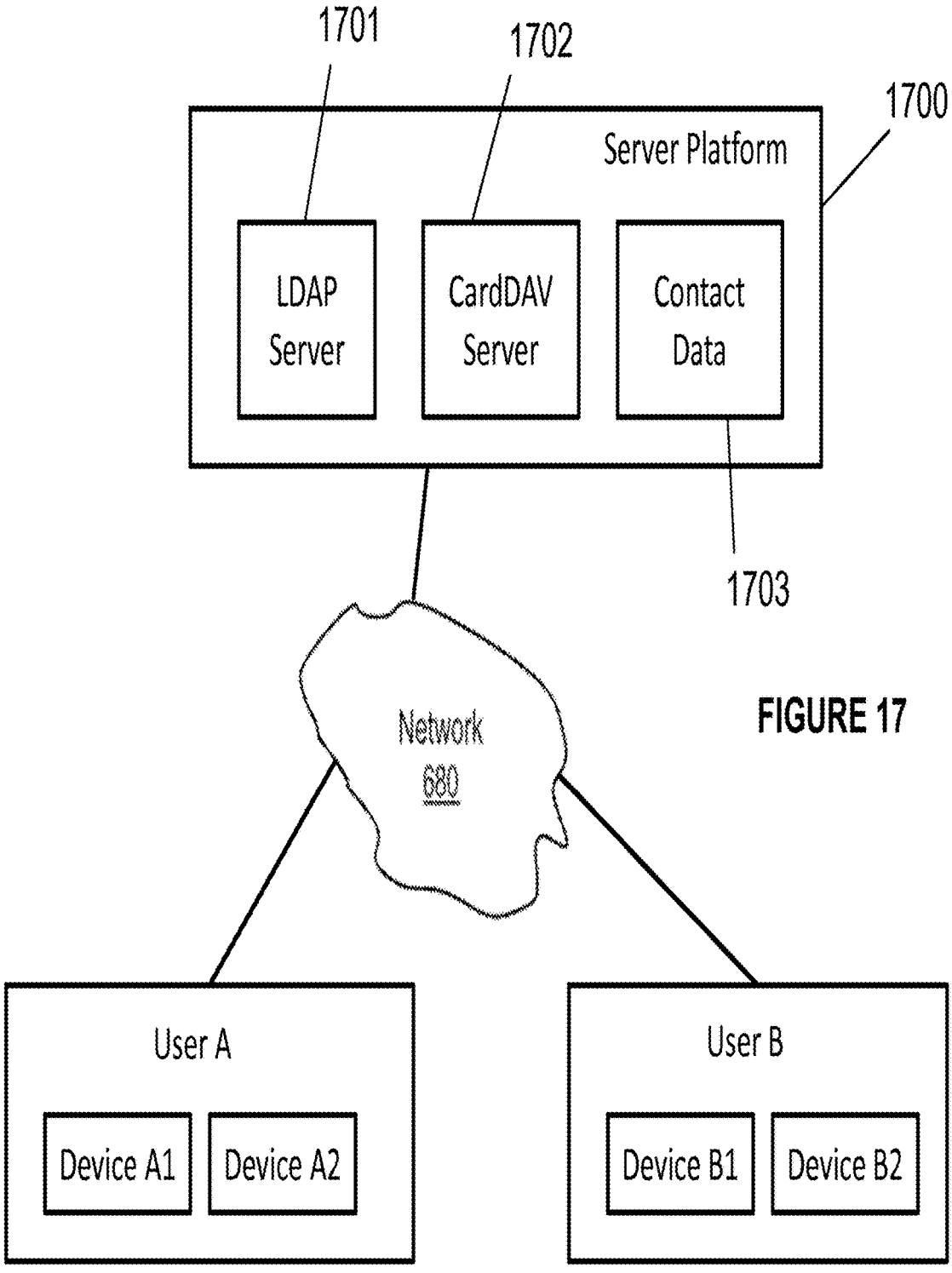

1701
1702
1700
Server Platform
LDAP Server
CardDAV Server
Contact Data
1703
Network 680
User A
Device A1
Device A2
User B
Device B1
Device B2

FIGURE 17

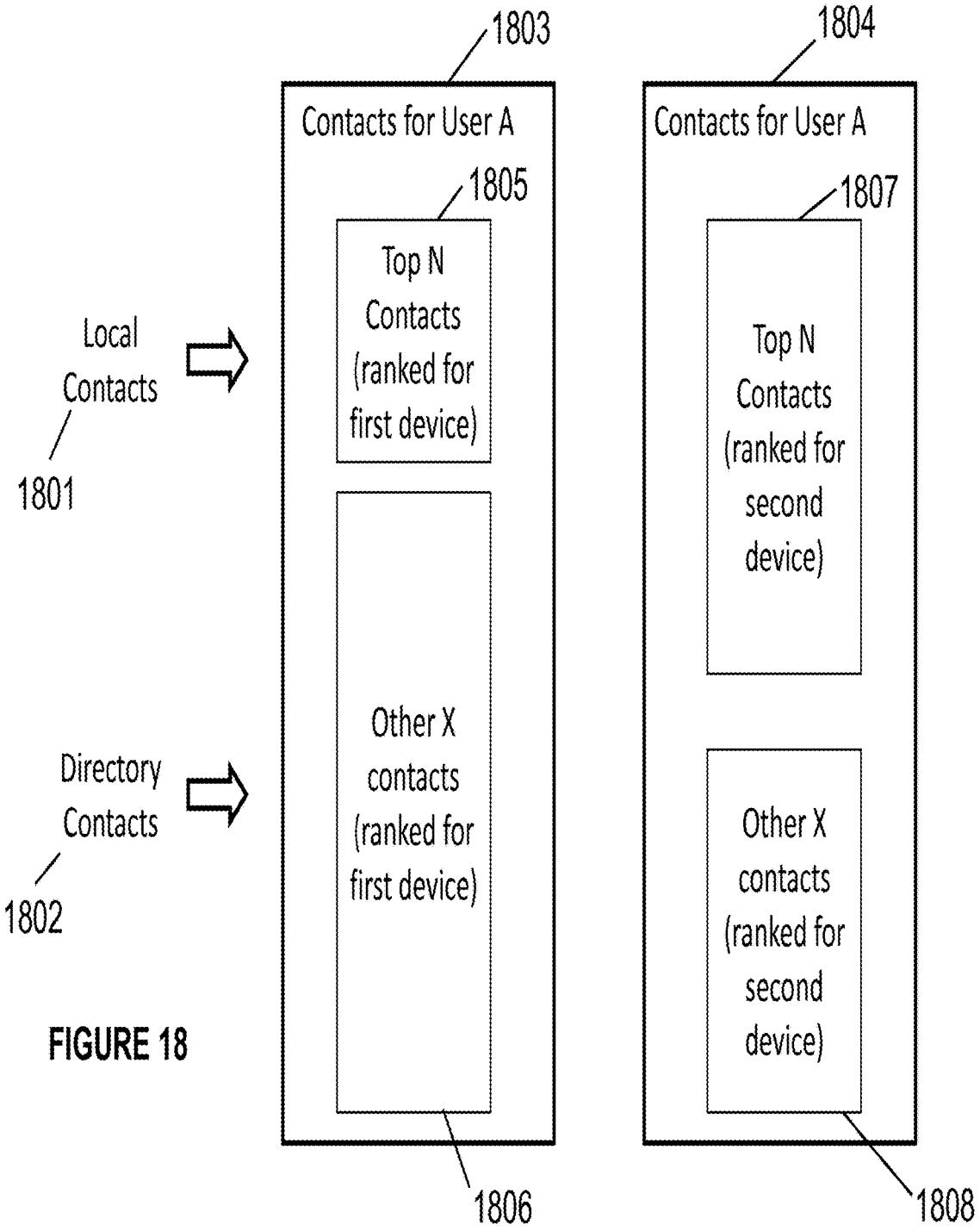

1803
1804
Contacts for User A
Contacts for User A
1805
1807
Top N Contacts (ranked for first device)
Top N Contacts (ranked for second device)
Local Contacts
1801
Directory Contacts
1802
Other X contacts (ranked for first device)
Other X contacts (ranked for second device)
1806
1808

FIGURE 18

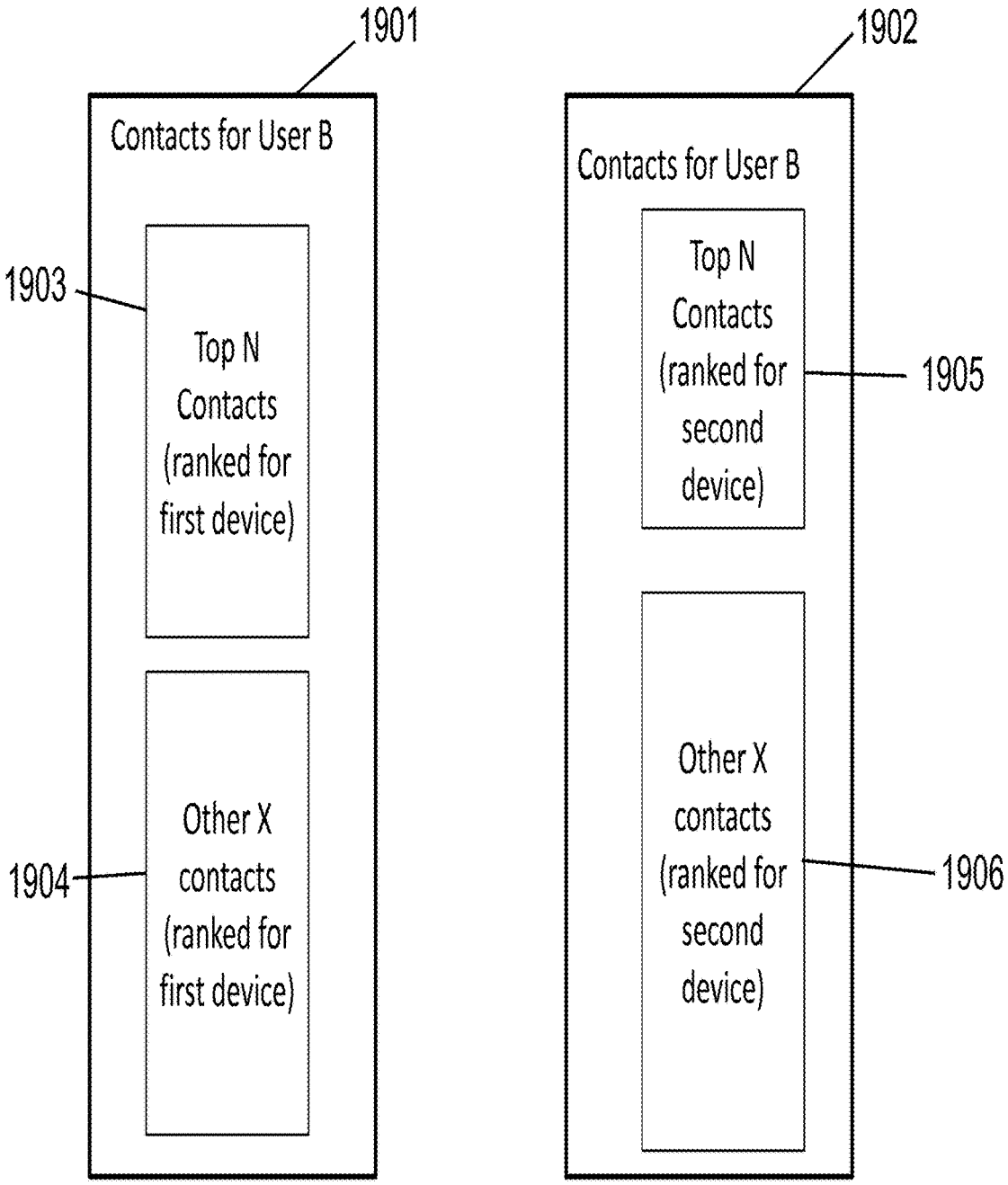

1901
Contacts for User B
1903
Top N Contacts (ranked for first device)
1904
Other X contacts (ranked for first device)
1902
Contacts for User B
Top N Contacts (ranked for second device)
1905
Other X contacts (ranked for second device)
1906

FIGURE 19

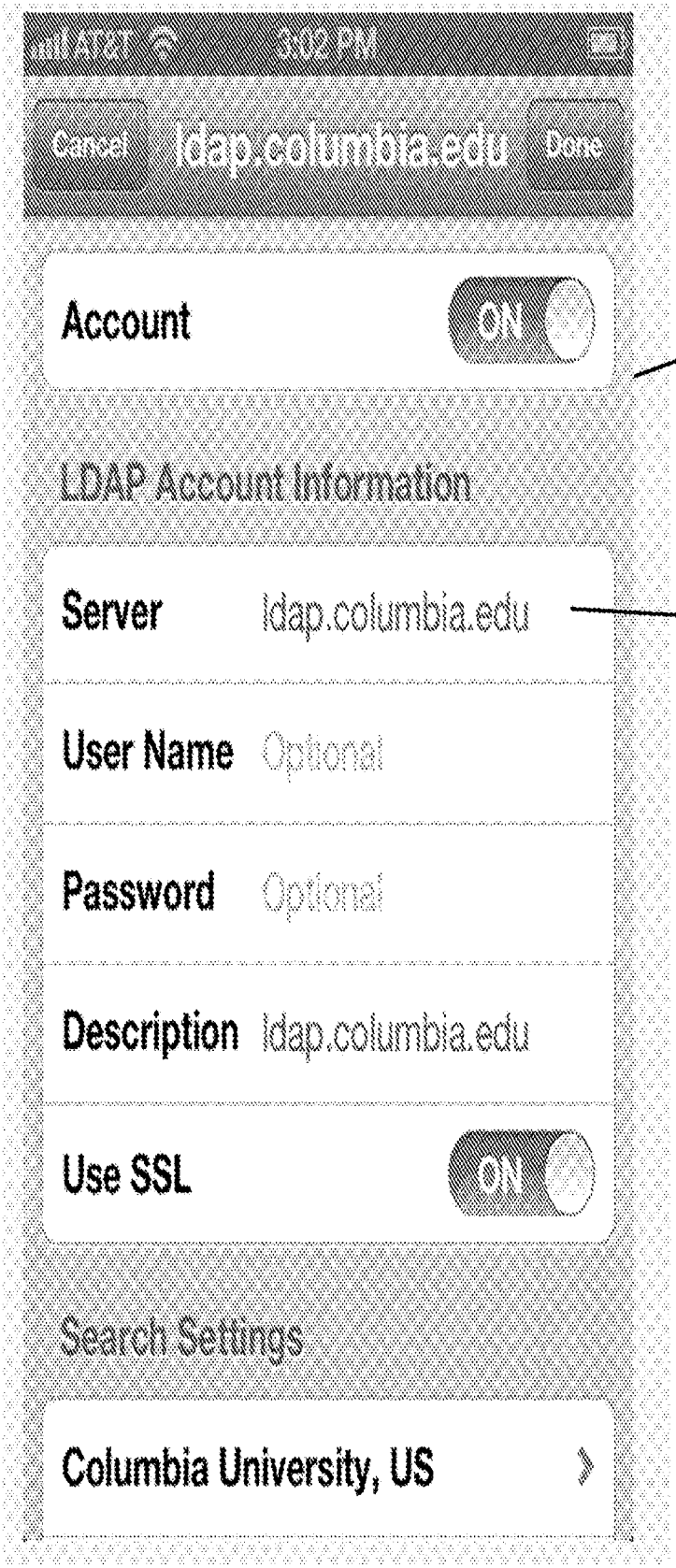

Cancel
ldap.columbia.edu
Done
Account
ON
2000
LDAP Account Information
Server
ldap.columbia.edu
2001
User Name
Optional
Password
Optional
Description
ldap.columbia.edu
Use SSL
ON
Search Settings
Columbia University, US

FIGURE 20


AT&T
3:04 PM
Cancel
CardDAV
Save
2100
Server
carddav.otherco.com
2101
User Name
andrewbuzza
Password
Description
carddav.otherco.com
Advanced Settings

FIGURE 21


AT&T
2:01 PM
Cancel
New Message
Send
To: frank
2202
Frank Cort
frank.cort@xobni.com
Frank Cort
frank.cort_xobni.com@xobni.com
2201
2200
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
space
@
.
return

FIGURE 22

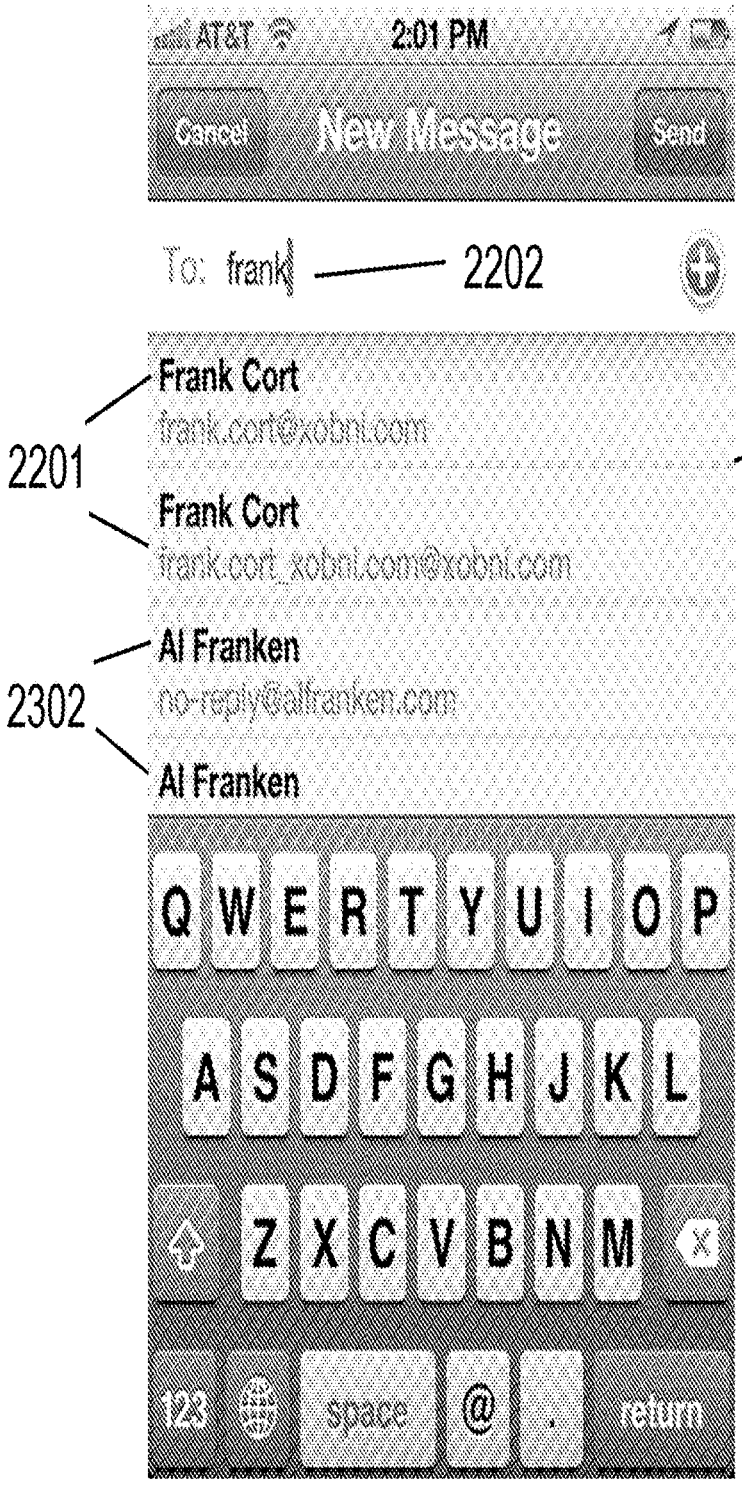

AT&T
2:01 PM
Cancel
New Message
Send
To: frank
2202
Frank Cort
frank.cort@xobni.com
Frank Cort
frank.cort_xobni.com@xobni.com
Al Franken
no-reply@alfranken.com
Al Franken
2201
2300
2302
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
space
@
.
return

FIGURE 23

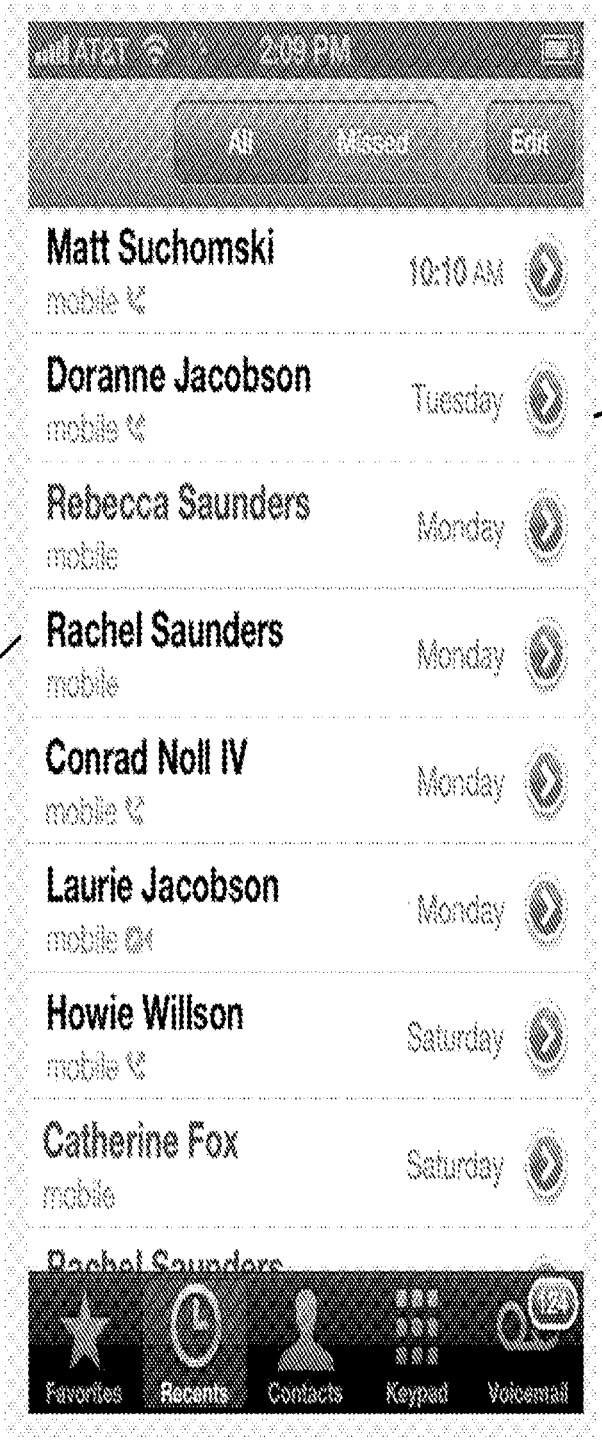

AT&T
2:09 PM
All
Missed
Edit
Matt Suchomski
mobile
10:10 AM
Doranne Jacobson
mobile
Tuesday
Rebecca Saunders
mobile
Monday
Rachel Saunders
mobile
Monday
Conrad Noll IV
mobile
Monday
Laurie Jacobson
mobile
Monday
Howie Willson
mobile
Saturday
Catherine Fox
mobile
Saturday
Favorites
Recents
Contacts
Keypad
Voicemail
2400
2401

FIGURE 24

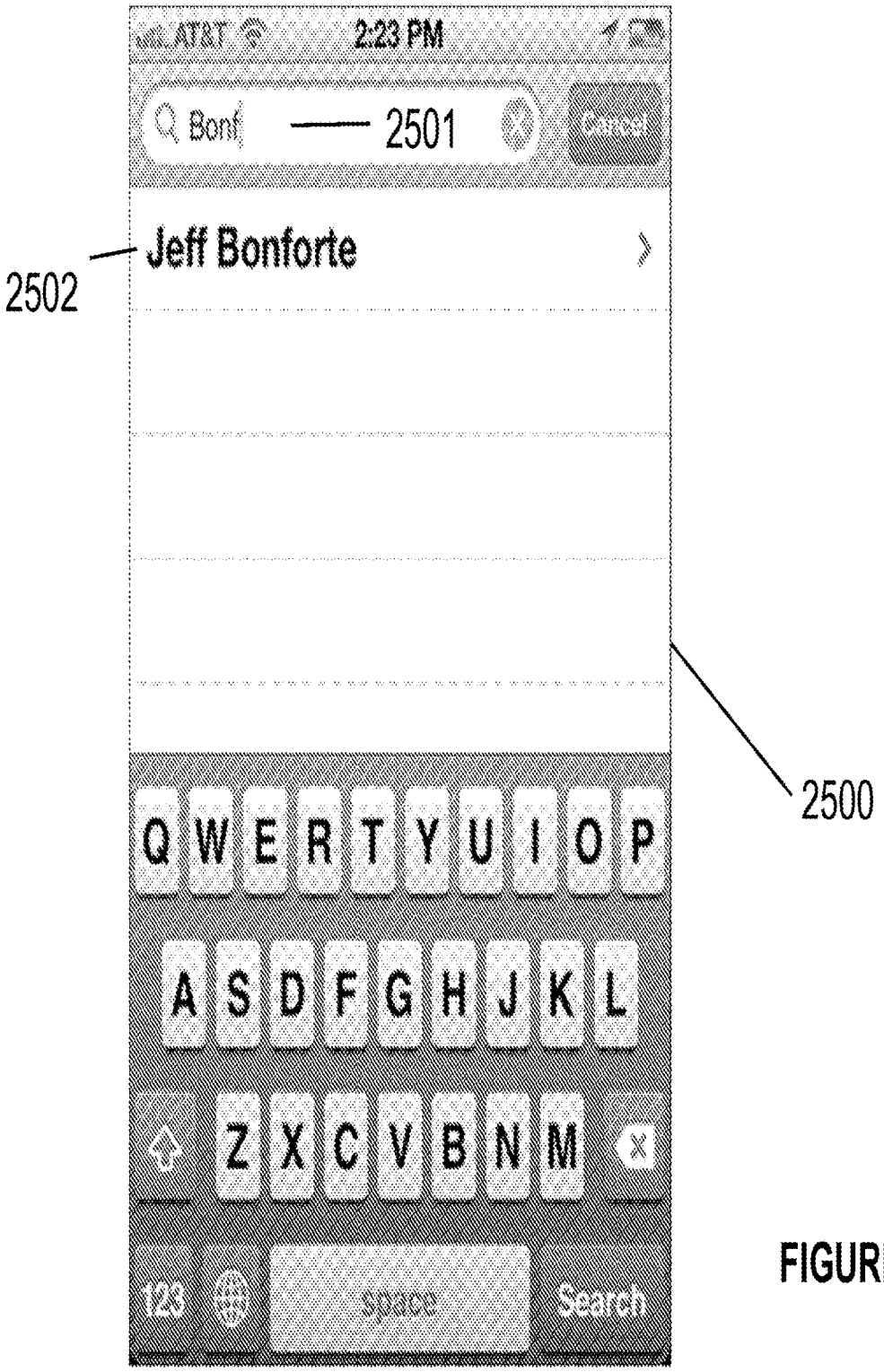

AT&T
2:23 PM
Bonf
2501
Cancel
Jeff Bonforte
2502
2500
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
space
Search

FIGURE 25

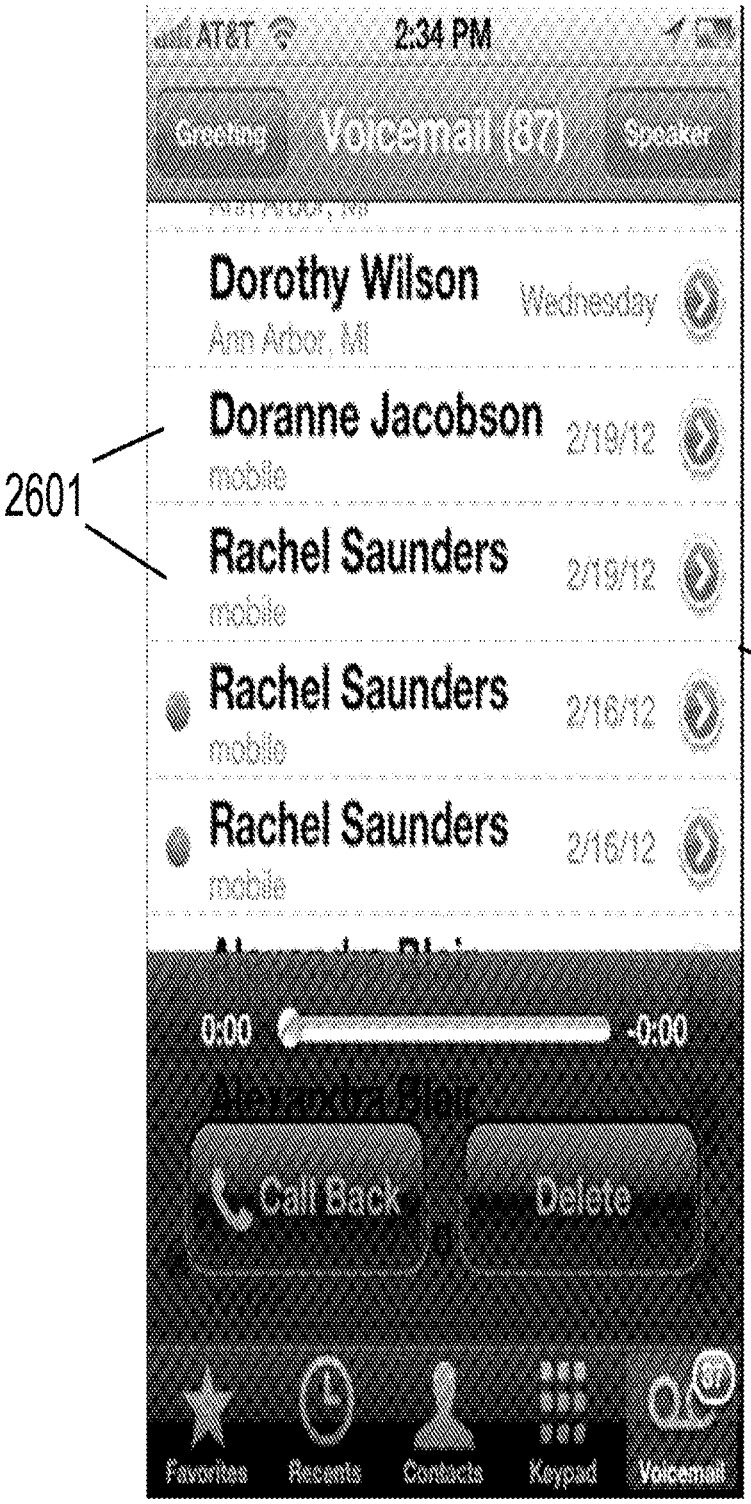

AT&T
2:34 PM
Greeting
Voicemail (87)
Speaker
Dorothy Wilson
Ann Arbor, MI
Wednesday
Doranne Jacobson
mobile
2/19/12
Rachel Saunders
mobile
2/19/12
Rachel Saunders
mobile
2/16/12
Rachel Saunders
mobile
2/16/12
0:00
-0:00
Call Back
Delete
Favorites
Recents
Contacts
Keypad
Voicemail
87
2601
2600

FIGURE 26

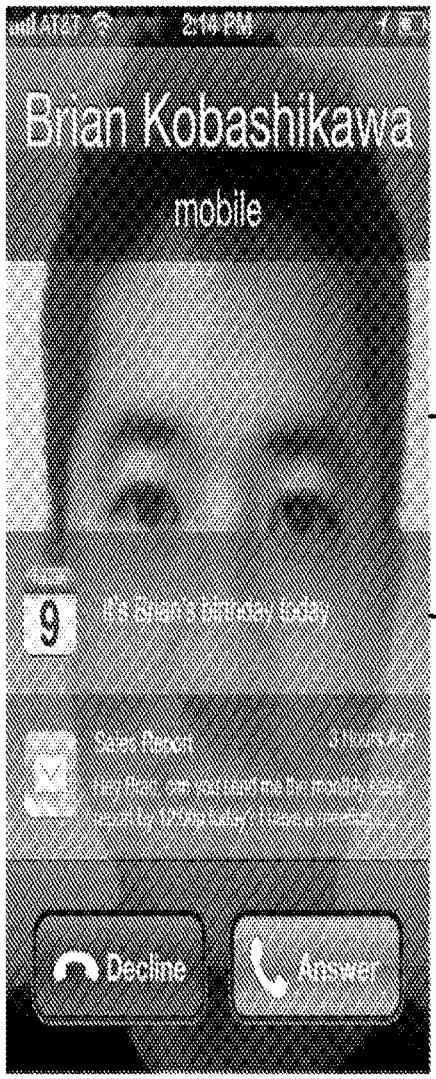


FIGURE 27

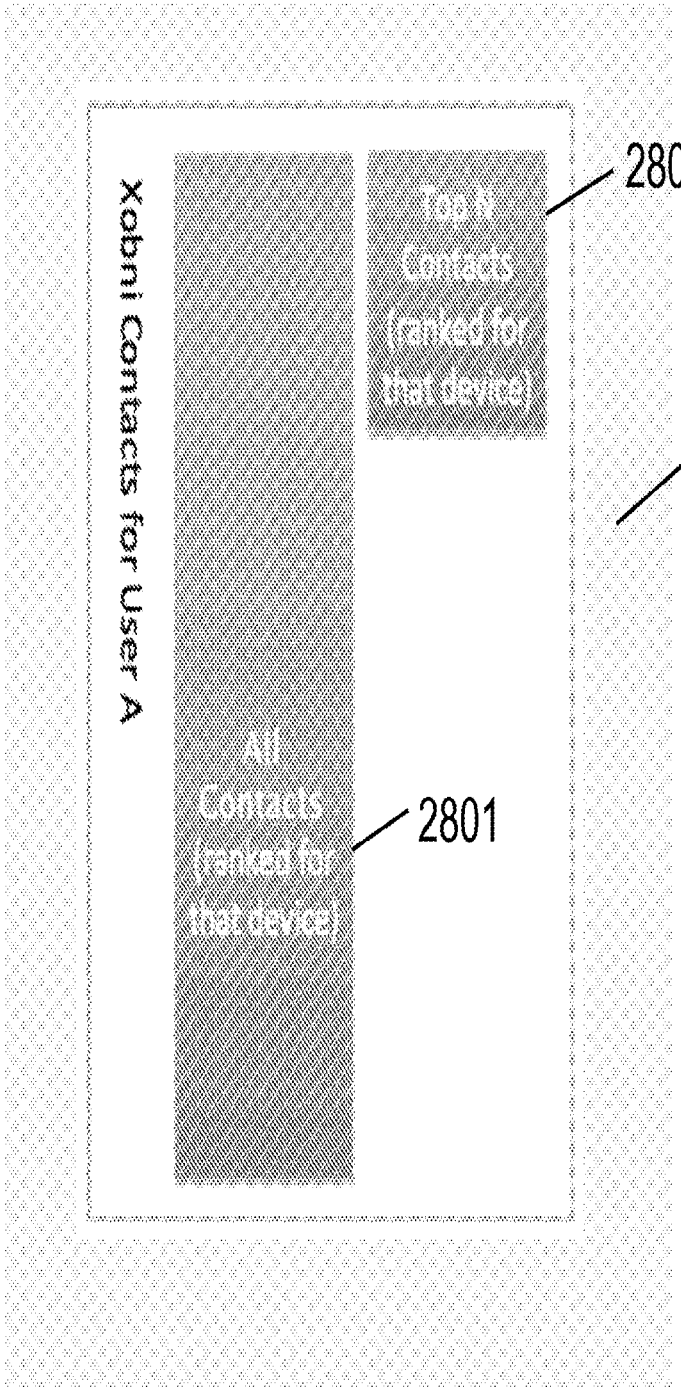

Xobni Contacts for User A
All Contacts (ranked for that device)
Top 5 Contacts (ranked for that device)
2801
2802
2800

FIGURE 28

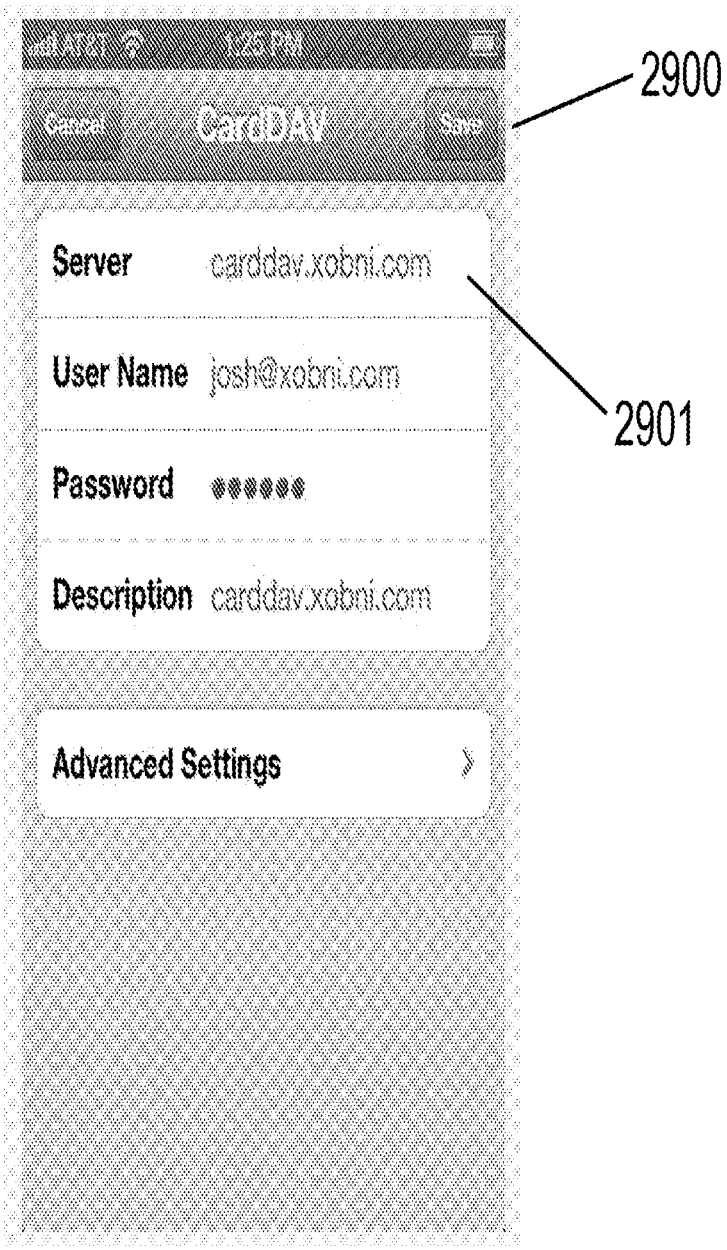

2900
Cancel
CardDAV
Save
Server
carddav.xobni.com
User Name
josh@xobni.com
Password
Description
carddav.xobni.com
Advanced Settings
2901

FIGURE 29

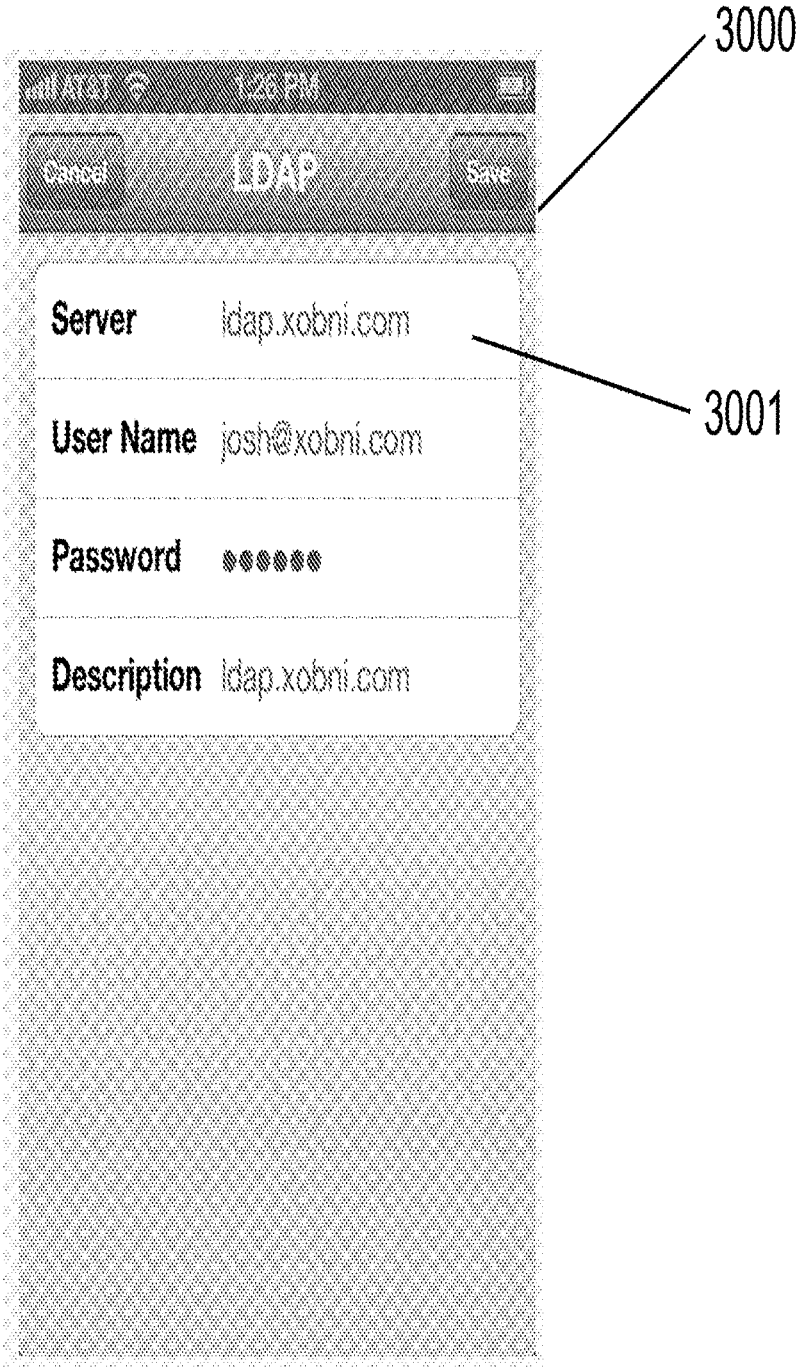

3000
Cancel
LDAP
Save
Server
ldap.xobni.com
User Name
josh@xobni.com
Password
Description
ldap.xobni.com
3001

FIGURE 30

CLASSIFYING A PORTION OF USER CONTACT DATA INTO LOCAL CONTACTS

[0001] This application is a continuation of and claims the benefit of U.S. patent application Ser. No. 13/693,955, filed Dec. 4, 2012, which is hereby incorporated by reference in its entirety.

FIELD OF THE TECHNOLOGY

[0002] At least some embodiments of the disclosure relate to contact data processing in general and, more particularly but not limited to, classifying a portion of contact data of a user of a computing device into local contacts.

BACKGROUND

[0003] Many systems and applications have been developed to allow people to communicate with each other. Such systems and applications may provide communication via emails, instant messages, text messages, web/online postings, etc.

[0004] Some applications have been developed to organize address information and other forms of contact data for users. For example, an address application may store information about a plurality of persons. For each of the persons, the address application may store the name of the person, the email address of the person, the street address of the person, the IM address of the person, the web page address of the person, phone numbers of the person, etc.

SUMMARY OF THE DESCRIPTION

[0005] Systems and methods are provided to classify a portion of the contact data of a user into local contacts. Some embodiments are summarized in this section.

[0006] In one embodiment, a method includes: storing contact data for each of a plurality of users including a first user, the contact data comprising a plurality of contacts for the first user; ranking each of the plurality of contacts to provide a ranking associated with a first user device of the first user; classifying (e.g., by a server platform) a portion of the plurality of contacts as each being a local contact based on the ranking to provide a set of local contacts; providing the set of local contacts for local storage on the first user device, wherein the set of local contacts includes a first local contact; and providing, by the server platform, to the first user device, a contact (e.g., a first directory contact) from the plurality of contacts.

[0007] In one embodiment, a computer-implemented method includes: storing contact data for each of a plurality of users including a first user, the contact data comprising a plurality of contacts for the first user; ranking each of the plurality of contacts to provide a ranking; classifying, by at least one processor, each of the plurality of contacts as either a local contact or a directory contact based on the ranking to provide a first set of local contacts and a first set of directory contacts; providing the first set of local contacts for local storage on a user device of the user, wherein the first set of local contacts includes a first local contact; and providing, by the at least one processor, to the user device, a first directory contact from the first set of directory contacts.

[0008] The disclosure includes methods and apparatuses which perform these methods, including data processing systems which perform these methods, and computer readable media containing instructions which when executed on data processing systems cause the systems to perform these methods. Other features will be apparent from the accompanying drawings and from the detailed description which follows.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] The embodiments are illustrated by way of example and not limitation in the figures of the accompanying drawings in which like references indicate similar elements.

[0010] FIG. **1** shows a user terminal to provide assistance in address input according to one embodiment.

[0011] FIG. **2** illustrates a profile of a person according to one embodiment.

[0012] FIG. **3** illustrates a system to obtain data for a profile according to one embodiment.

[0013] FIG. **4** illustrates a user interface to provide assistance in address input according to one embodiment.

[0014] FIG. **5** illustrates another user interface to provide assistance in address input according to one embodiment.

[0015] FIG. **6** shows a system to provide server side profile information according to one embodiment.

[0016] FIG. **7** shows a screen of an enhanced client according to one embodiment.

[0017] FIG. **8** shows a screen related to packaging of a data file according to one embodiment.

[0018] FIG. **9** shows a screen providing configuration settings data according to one embodiment.

[0019] FIG. **10** shows a screen to allow a user to manually enter configuration settings data according to one embodiment.

[0020] FIG. **11** shows a screen of a client according to one embodiment.

[0021] FIG. **12** shows a display of a suggestion served by an LDAP server and a suggestion provided by a client according to one embodiment.

[0022] FIG. **13** shows a display of a suggestion served by an LDAP server according to one embodiment.

[0023] FIG. **14** shows a method to provide a suggestion to a client according to one embodiment.

[0024] FIG. **15** shows a data processing system, which can be used in various embodiments.

[0025] FIG. **16** illustrates a server-client system for profile processing.

[0026] FIG. **17** shows a system for classifying user contact data for a User A and a User B according to one embodiment.

[0027] FIG. **18** shows classification of contact data for User A into local contacts and directory contacts according to one embodiment.

[0028] FIG. **19** shows classification of contact data for User B into local contacts and directory contacts according to one embodiment.

[0029] FIG. **20** shows a configuration screen of a user device to provide settings for communicating with an LDAP server according to one embodiment.

[0030] FIG. **21** shows a configuration screen of the user device of FIG. **20** to provide settings for communicating with a CardDAV server according to one embodiment.

[0031] FIG. **22** shows a screen of a user device for a user to input data to create a new message according to one embodiment.

[0032] FIG. **23** shows the screen of FIG. **22** a few seconds after the user has input a few initial characters according to one embodiment.

[0033] FIG. **24** shows a screen of a user device that displays a call history including local contacts provided via a CardDAV server according to one embodiment.

[0034] FIG. **25** shows a screen of a user device that displays a local contact provided via a CardDAV server when the user is performing a local contact search according to one embodiment.

[0035] FIG. **26** shows a screen of a user device displaying a voicemail list including local contacts provided via a CardDAV server according to one embodiment.

[0036] FIG. **27** shows a screen of a user device displaying caller ID information, previously provided to the user device from a CardDAV server, for a local contact that is calling the user device according to one embodiment.

[0037] FIG. **28** shows classification of a portion of contact data for User A into local contacts according to one embodiment.

[0038] FIG. **29** shows a configuration screen of a user device to provide settings for communicating with a CardDAV server according to one embodiment.

[0039] FIG. **30** shows a configuration screen of the user device of FIG. **29** to provide settings for communicating with an LDAP server according to one embodiment.

DETAILED DESCRIPTION

[0040] The following description and drawings are illustrative and are not to be construed as limiting. Numerous specific details are described to provide a thorough understanding. However, in certain instances, well known or conventional details are not described in order to avoid obscuring the description. References to one or an embodiment in the present disclosure are not necessarily references to the same embodiment; and, such references mean at least one.

[0041] Reference in this specification to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the disclosure. The appearances of the phrase "in one embodiment" in various places in the specification are not necessarily all referring to the same embodiment, nor are separate or alternative embodiments mutually exclusive of other embodiments. Moreover, various features are described which may be exhibited by some embodiments and not by others. Similarly, various requirements are described which may be requirements for some embodiments but not other embodiments.

[0042] In one embodiment, a user terminal is configured to provide assistance for the completion of inputting an address. While the user is typing to provide an input to specify an address for a message, the user terminal uses the incomplete input that has been provided to an address field so far by the user to identify one or more options or suggestions to complete the input, and thus allows the user to complete the input by selecting one.

[0043] In one embodiment, the user terminal is configured to search a profile set to search for a portion of the candidates that matches the incomplete input provided by the user. The profile set contains information about a plurality of persons, to whom the user may or may not have previously sent a message.

[0044] FIG. **1** shows a user terminal to provide assistance in address input according to one embodiment. The user terminal may be implemented as a personal computer, a web enabled television set, a personal digital assistant (PDA), a tablet (e.g., an iPad device) or a mobile phone, using special purpose hardware (Application-Specific Integrated Circuit (ASIC) or Field-Programmable Gate Array (FPGA)), software and a general purpose processor, or a combination of special purpose hardware and software. Thus, the disclosure is not limited to a particular implementation.

[0045] In FIG. **1**, the user terminal (**101**) is configured to store the messages (**115**) received at the user terminal (**101**) and the messages (**117**) sent from the user terminal (**101**). The user terminal (**101**) includes a message viewer (**123**) which can be used to display a message selected from the received messages (**115**) or selected from the sent messages (**117**).

[0046] In FIG. **1**, the user terminal (**101**) further includes a message composer (**121**) which is configured to present a user interface to facilitate the composition of an outgoing message.

[0047] In one embodiment, the user terminal (**101**) is configured to generate an address set (**111**) based on the addresses that have been specified in the message composer (**121**) for one or more previously sent messages (**117**). When an address is used in the message composer (**121**) to specify a recipient of a message, the address is added to the address set (**111**), if the address is not already in the address set (**111**).

[0048] In one embodiment, the message composer (**121**) uses the address set (**111**) to suggest candidates for the completion of an input when the user is typing an address. For example, when the characters typed in an address field are the same as the first characters in a number of email addresses in the address set (**111**), the message composer (**121**) displays the email addresses as possible candidates for the completion of the input.

[0049] In one embodiment, the user terminal (**101**) further includes a profile presenter (**125**), which can provide suggestions for completion based on the profile set (**113**) maintained by the profile builder (**119**).

[0050] In one embodiment, the profile builder (**119**) is configured to extract profile data of various persons based on the received messages (**115**) and the sent messages (**117**). The profile builder (**119**) can extract information about persons not only from the headers of the messages where the senders and the recipients of the messages are specified, but also from the bodies of the messages and from other online sources, such as online directories, social networking websites, web pages, result pages found by search engines, etc.

[0051] Since the profile builder (**119**) obtains the profile data for various persons referenced in the messages (**115** and **117**), the profile set (**113**) is more comprehensive than the address set (**111**).

[0052] In one embodiment, the profile builder (**119**) is configured to scan the received messages (**115**) and the sent messages (**117**) for email addresses and names associated with the email addresses. The profile builder (**119**) generates a profile for each person identified to create the profile set (**113**). When new messages are received or sent, the profile set (**113**) is updated based on the new messages.

[0053] In some embodiments, the profile builder (**119**) may further identify persons based on scanning the received messages (**115**) and the sent messages (**117**) for phone numbers, names, addresses, etc.

[0054] In some embodiments, as discussed in more detail below, some or all of the components (**111-125**) in the user terminal (**101**) may be hosted on a server or platform remote to the user terminal (**101**) (e.g., accessible via a website and a web browser) in addition to or instead of being implemented in the user terminal (**101**). For example, in one embodiment, in addition to the user terminal (**101**), the profile set (**113**) may be hosted on a remote server or platform, and accessible via a web site. As another example, in one embodiment, the received messages (**115**) and the sent messages (**117**) may be hosted on a web site; and the user can use a web browser to view a selected one of the messages (**115** and **117**).

[0055] For example, the profile builder (**119**) may be configured to communicate with the server to extract the profile set (**113**) from the messages (**115** and **117**) hosted on the server. Alternatively, the profiler builder (**119**) may also be hosted on the server to build the profile set (**113**) on the server.

[0056] FIG. **2** illustrates a profile of a person according to one embodiment. In FIG. **2**, the profile (**131**) includes various fields, such as the name (**141**) of the person, a photo image (**142**) of the person, one or more phone numbers of the person (**143**), address information (**144**) (e.g., email address, IM address, street address), business information (**145**) (e.g., employer, work address, job title), the date and time of the last message received from the person (**146**), the date and time of the last message sent to the person (**147**), the total number of messages received from the person (**148**), the total number of messages sent to the person (**149**), etc.

[0057] In one embodiment, the profile builder (**119**) obtains at least some of the information for the fields from the received messages (**115**) or the sent messages (**117**) to identify the person, and then uses other information sources to obtain the data for the other fields in an automated way to collect the profile information on behalf of the user.

[0058] For example, the profile builder (**119**) may use social networks, search engines, photo services, etc. to obtain the photo (**142**), the business information (**145**), etc.

[0059] FIG. **3** illustrates a system to obtain data for a profile according to one embodiment. In FIG. **3**, under the control of the profile builder (**119**), the user terminal (**101**) may communicate with various servers in an automated way to extract data for the profile (**131**) over the network (**151**). The network (**151**) may include a local area network, a cellular telecommunications network, a wireless wide area network, a wireless local area network, a wireless personal area network, an intranet, and/or Internet.

[0060] In one embodiment, the user terminal (**101**) communicates with the message server (**162**) to receive and send the messages (**115** and **117**).

[0061] In one embodiment, after the profile builder (**119**) extracts from the messages (**115** or **117**) certain information (e.g., an email address, an IM user name, a phone number, and/or a name) about a person, the profile builder (**119**) uses the extracted information to query various servers to obtain further information (e.g., photo (**142**), phone number (**143**), address information (**144**), and/or business information (**145**)) about the person to create the profile (**131**).

[0062] For example, the user terminal (**101**) may query a web search engine (**163**), an online directory (**164**), a social network server (**165**), a business website (**166**), a personal website (**167**), a media sharing website (**168**), a map and direction web site (**169**), an online retailer (**152**), a travel website (**153**), a location website (**155**), and/or other servers. Information that can be used as search criteria include names, screen names, social network profile names, social network profile URLs, physical addresses, website URLs, email addresses, or telephone numbers. Information that is collected as a result of these queries may be used in future searches to identify additional information that may be used to create a person profile.

[0063] For example, the user terminal (**101**) may receive an email sent by a person via the message server (**162**). The profile builder (**119**) of the user terminal (**101**) is configured to perform a search using the web search engine (**163**) with the email address of the person as the search criteria. The search engine (**163**) may return a search result that includes the phone number (**143**) of the person. The profile builder (**119**) extracts the phone number (**143**) from the search result and stores the phone number (**143**) as part of the profile (**131**) of the person. The search engine (**163**) may also return the URL for or link to a personal website (**167**) belonging to the person. The personal website (**167**) may contain additional information about the person that may be used to create a person profile, such as additional contact information or biographical information.

[0064] In another example, the email address belonging to the person may include an extension (or domain name) for a company. The profile builder (**119**) of the user terminal (**101**) is configured to perform a search using the search engine (**163**) with the email extension (or domain name) as the search criteria. A result returned by the search may be a business website (**166**). The profile builder (**119**) of the user terminal (**101**) is further configured to search the business website (**166**) to obtain information relevant to the profile (**131**) of the person. For example, a web page on the business website (**166**) may contain additional information about the person that may be used to create a person profile, such as business information (**145**), additional contact information or biographical information.

[0065] In another example, the profile builder (**119**) of the user terminal (**101**) is configured to perform a search using an online directory (**164**) (e.g., a person search directory, a yellow page directory) with the name (**141**) of the person or other contact information as the search criteria. The online directory (**164**) may return search results that include additional contact information and other information that may be used for the profile (**131**) of the person.

[0066] In another example, the user terminal (**101**) may receive an email sent by the person via the message server (**162**). The email may contain a social network user name for the person. The profile builder (**119**) of the user terminal (**101**) is configured to extract this social network user name from the email and use it to access the social network server (**165**). A webpage on the social network server (**165**) may contain additional contact information and other information that may be extracted and used by the profile builder (**119**) to create the profile (**131**) of the person. The webpage on the social network server (**165**) may also contain additional contacts that may be associated with the person in the profile (**131**). For example, persons on the friends list of the webpage on the social network server (**165**), or persons who have posted comments or messages on the social network server (**165**) may be listed as contacts in a contact network for the person.

[0067] In another example, a search performed using the search engine (**163**) may return a URL or link for a media sharing website (**168**) (e.g., for sharing photos or videos). The media sharing website (**168**) may store profile information about the person. Thus, the profile builder (**119**) of the user terminal (**101**) can extract additional contact information or biographical information from the media sharing website (**168**) for the creation of the profile (**131**) of the person. For example, a profile belonging to the person on a video sharing website may include an instant message screen name (user name) for the person. This screen name may be extracted and displayed as part of the profile (**131**) of the person.

[0068] In one embodiment, information extracted from communications between the person and other users may also be used to update profile information on a social network server (**165**) or other websites. For example, the profile builder (**119**) of the user terminal (**101**) may detect that the person has primarily used email address "david@foo.com" in recent communications, whereas the profile of the person on the social network server (**165**) shows his email address as "david@bar.com." The profile builder (**119**) of the user terminal (**101**) can share the new email address of the person with the social network server (**165**) to allow the social network server (**165**) to automatically update the corresponding information about the person, or to suggest the person to make an update based on this changed behavior detected by the profile builder (**119**) of the user terminal (**101**).

[0069] In one embodiment, the profile builder (**119**) of the user terminal (**101**) can also extract information about the person from the travel website (**153**) and the online retailer (**152**) for the creation of the profile (**131**) of the person. For example, after an email containing information about a flight itinerary is received in the user terminal (**101**), the profile builder (**119**) of the user terminal (**101**) may extract a flight number or other information about a flight from the email. The profile builder (**119**) of user terminal (**101**) then queries the travel website (**153**) using the flight number or other flight information as search criteria. Information about the flight, such as the destination city or the departure city, expected departure time, expected arrival time, expected delays, weather in the destination city, weather in the departure city, or any changes to the flight may be used as part of the profile (**131**) of the person.

[0070] In another example, an email containing information about an item or service that the second user is interested in purchasing may be received in the user terminal (**101**). The profile builder (**119**) of user terminal (**101**) is configured to query one or more search engines, websites, or on-line retailers (**152**) to determine which retailer or website has the best price or currently has the item in stock or the service available. This information is extracted by the profile builder (**119**) and displayed by the profile presenter (**125**) as part of the profile (**131**) of the person.

[0071] In one embodiment, the profile builder (**119**) of the user terminal (**101**) can also extract information from a map and direction website (**169**) and location servers (**155**) as part of the profile (**131**) of the person. For example, the person may own a GPS unit, cell phone, or other device that is capable of transmitting the person's current physical location to the location server (**155**), which allows other users to access the person's current location information. If the user of the user terminal (**101**) has permission to view the location information of the person, the profile builder (**119**) of the user terminal (**101**) may access the location server (**155**) over the network (**151**) to receive location information about the person. This location information can be displayed as part of a person profile.

[0072] The profile builder (**119**) of the user terminal (**101**) may also access the map and direction website (**169**) to create a map of the current location of the person, or to generate directions to the current location of the person. The map or directions may be displayed by the profile presenter (**125**) as part of the profile (**131**) of the person. The map and direction website (**169**) may also be used to generate a map or directions to one or more known street addresses of the person, such as a work address or home address. The map or directions can be displayed by the profile presenter (**125**) as part of the profile (**131**) of the person.

[0073] FIG. **4** illustrates a user interface to provide assistance in address input according to one embodiment. In FIG. **4**, the user interface (**201**) is configured for composing an outgoing message. The user interface (**201**) includes entries to specify the addresses of the recipients of the message. For example, the entry box (**203**) is configured to receive the email address of an outgoing email, which may be sent from the user terminal (**101**) via the message server (**162**).

[0074] In one embodiment, when the entry box (**203**) detects a change in the content received in the entry box (**203**) (e.g., a keystroke is received in the entry box (**203**), as the user types a letter on a keyboard to provide an input), the profile presenter (**125**) determines a set of suggestions for the completion of the input in the entry box (**203**). The profile presenter (**125**) uses the incomplete input in the entry box (**203**) as a search criterion to find from the profile set (**113**) a set of suggestions for the complete address input. The suggestions are presented in the window (**205**) as a selectable list.

[0075] In one embodiment, when the message composer (**121**) has an existing, native mechanism for displaying suggestions (e.g., making suggestions based on address set (**111**)), the profile presenter (**125**) can be configured as an add-on module to display these suggestions as well and to hide the native mechanism of the message composer (**121**) for displaying suggestions.

[0076] In FIG. **4**, the item (**209**) on the top of the list is highlighted; and the user may select the highlighted item (**209**) with a keystroke (e.g., pressing Tab key, Enter key, Home, End, PgUp, PgDown, etc.). The user may use the arrow keys to move the highlight up or down the list to highlight a different item, or provide additional input in the entry box (**203**) to cause the profile presenter (**125**) to update the suggestions in the window (**205**). The user may also use the cursor (**207**) to select an item using a cursor control, such as a mouse, a track ball, a touch pad, a touch screen, etc.

[0077] In one embodiment, when the message composer (**121**) has an existing, native mechanism that uses these keys for the display of suggestions, the profile presenter (**125**) configured as an add-on module can hide these keystrokes from the native mechanism.

[0078] In one embodiment, after an item is selected from the list presented in the suggestion window (**205**), the address represented by the item is inserted by the profile presenter (**125**) into the entry box (**203**) to replace the corresponding portion that is used to suggest the selected item; and the suggestion window (**205**) is closed automatically.

[0079] In some embodiments, the message composer (**121**) may be configured to perform post-processing, such as name checking, on the inserted text in the entry box (**203**) a few moments after the text has been inserted. The insertion by the profile presenter (**125**) is performed in a way that allows such post-processing by the message composer (**121**) to continue functioning. For example, if the message composer (**121**) loses focus or activation, it starts post-processing the entries in the textbox. The profile presenter (**125**) is configured to temporarily prevent this from happening while the suggestion window (**205**) is visible and causes this to happen when the suggestion window (**205**) is closed. To hide the suggestion window (**205**), the suggestion window (**205**) is configured as a topmost window that is initially not activated. The Outlook textbox window is subclassed via a code-injection mechanism to temporarily preventing it from getting "kill focus" messages (e.g., through intercepting such messages, and then throwing them away). After the suggestion window (**205**) becomes invisible, the profile presenter (**125**) sets the focus back to the textbox window and causes the post-processing to be performed.

[0080] For example, in FIG. **4**, the incomplete input "Graf" is received in the entry box (**203**). The cursor "|" in the entry box (**203**) indicates the insertion point for subsequent keystrokes received in the user terminal (**101**). While the entry box (**203**) is expecting additional input from the user, the profile presenter (**125**) uses the incomplete input "Graf" to identify a set of suggestions, such as mgraf@gmail.com for Mary Graf, ericg@yahoo.com for Eric Grafstrom, graffiti@hotmail.com for Steve Curtis, graffiti@pacbell.net for SFPD, etc., based on the profile set (**113**) established by the profile builder (**119**).

[0081] In one embodiment, the suggestions are identified by matching the incomplete input with names, addresses and/or other profile data of the persons in the profile set (**113**). The incomplete input may match the beginning of a last name, the beginning of a first name, the beginning of the name of an organization, the beginning of a domain name of an email address, the beginning of the top level domain name of an email address, the beginning of the email prefix, the beginning of a word in the job title (or the beginning of the job title), the beginning of the city of the person, etc. In some embodiments, when an email prefix contains one or more separators, such as "_" or "." or "-", the email prefix is broken down into segments or chunks; and the incomplete input may be match the beginning of any of the segments or chunks. When there is a match, the person can be selected as a candidate for the suggestions. Thus, the searching for a match is not limited to matching the beginning of an address that the user has previously typed for a previous outgoing message.

[0082] In FIG. **4**, the top item, mgraf@gmail.com for Mary Graf, is highlighted and can be selected by pressing the Tab key or the Enter key. Alternatively, the user may select the second item, ericg@yahoo.com for Eric Grafstrom, by pressing a mouse button while the cursor (**207**) is over the second item. The profile presenter (**125**) is configured to use the selected address to replace the incomplete input "Graf" received in the entry box (**203**).

[0083] In FIG. **4**, if the user further types a letter "f", the profile presenter (**125**) will use the incomplete input "Graff" to eliminate some suggestions (e.g., mgraf@gmail.com for Mary Graf, ericg@yahoo.com for Eric Grafstrom) and update the list in the suggestion window (**205**).

[0084] Thus, the profile presenter (**125**) uses the profile set (**113**) to identify suggestions based on incomplete input provided in the entry box (**203**) and to allow the user to select a suggestion to complete the input.

[0085] FIG. **4** illustrates an example in which the suggestion window (**205**) is presented to provide suggestions for the completion of an address input in the "To" field of the user interface (**201**) for composing an outgoing message. Similar suggestions can be provided when the user is typing in other fields that are configured to receive address inputs, such as the "CC" field, or "BCC" field, of a user interface for composing an outgoing email message. Further, the suggestions can be provided when the user is typing in other types of user interfaces that are configured to receive address inputs, such as a user interface to edit a contact entry, a profile page, etc. Thus, the invention is not limited to the user interface for composing an email message.

[0086] FIG. **5** illustrates another user interface to provide assistance in address input according to one embodiment.

[0087] In FIG. **5**, the suggestions are presented as two parts. The part above the separator (**211**) is identified from the address set (**111**) and the part below the separator (**211**) is identified from the profile set (**113**).

[0088] In one embodiment, the address set (**111**) is generated by collecting the addresses that have been previously specified in the address fields of the message composer (**121**) of the user terminal (**101**). In one embodiment, the suggestion above the separator (**211**) is identified by matching the incomplete input "Graf" with the starting letters of the addresses in the address set (**111**). The addresses in the address set (**111**) that have the leading characters "graf" are selected to generate the suggestions above the separator (**211**).

[0089] In one embodiment, the message composer (**121**) has an existing, native mechanism for displaying suggestions (e.g., making suggestions based on address set (**111**)). The profile presenter (**125**) is configured to obtain the suggestions from the native mechanism (e.g., via scraping, inspecting, querying, etc.) and displays the suggestions above the separator (**121**).

[0090] In one embodiment, the suggestions presented above the separator (**211**) are identified by the message composer (**121**). The profile presenter (**125**) obtains the suggestions from the message composer (**121**), presents the suggestions in the suggestion window (**205**), hide the suggestions the message composer (**121**) would have shown (and thus prevents the message composer (**121**) from presenting the suggestions in a separate window).

[0091] In another embodiment, the profile presenter (**125**) shows the suggestions based on the profile set (**113**) in one suggestion window; the message composer (**121**) shows the suggestions in a different suggestion window; and the profile presenter (**125**) aligns its suggestion window with the suggestion window of the message composer (**121**) so that the two suggestion windows appear like two panels of a large window. Alternatively, the profile presenter (**125**) presents its suggestion window over the suggestion window of the message composer (**121**) (to cover up and thus effectively disable the suggestion window of the message composer (**121**)). In one embodiment, the profile presenter (**125**) causes the suggestion window of the message composer (**121**) to be invisible on subsequent invocations to hide the suggestion window of the message composer (**121**).

[0092] In another embodiment, the profile presenter (**125**) uses the address set (**111**) to identify the suggestions presented above the separator (**211**) in the suggestion window (**205**), using the same approach the suggestion window of the message composer (**121**) would use.

[0093] In one embodiment, the profile presenter (**125**) includes an auto suggest manager. The auto suggest manager can turn on or turn off the feature of auto suggestions generated based on the profile set (**113**) in accordance with the preferences of the user. The auto suggest manager provides encapsulation, providing an easy interface for external applications to use the auto suggest feature without understanding its implementation. It provides mechanisms to add the auto suggest feature to a new window (message, contact, etc.), to "clean up" or remove the auto suggest feature from a window or all windows, to allow logging of auto suggest state and problems, and to determine any special preferences the user may have set in the native client (e.g., Outlook).

[0094] In one embodiment, the profile presenter (**125**) includes a suggestion window (**205**) that presents a dialog box or a list box to show the suggested results. The profile presenter (**125**) includes a view controller to show or hide the suggestion window (**205**). The view controller also positions the suggestion window (**205**) in the correct location (which varies as the user types in text), participates in "hiding" the suggestion window (**205**) from the native application (e.g., Outlook) so that the native application is not aware of the existence of the suggestion window (**205**), and notify other modules of navigation commands (PgUp, PgDown, etc.) and user selections. When a selection or keyboard command (e.g., arrow keys, tab return) related to the suggestions occurs, the suggestion window (**205**) (or a keyboard hook or the inspector controller, discussed below) provides messages to the view controller for processing.

[0095] In one embodiment, the profile presenter (**125**) further includes a result controller, which decides what results to show and when. After the profile presenter (**125**) detects that the user is typing in an address field (e.g., To, CC, or BCC fields of a window to compose an email), the result controller uses the incomplete input as a search criterion to search for the suggestions based on the profile set (**113**) and/or based on the address set (**111**). The view controller causes the display of the suggestion window (**205**) to show the search results. If the user selects a suggestion from the window (**205**), the address corresponding to the user selection is put into the address field.

[0096] In one embodiment, the profile presenter (**125**) is implemented as an add-on module for an existing communications client, such as Microsoft Outlook, which can make suggestions by selecting from the address set (**111**) the email addresses that start with the incomplete input typed by the user in the address field. The suggestion window (**205**) of the profile presenter (**125**) subclasses the suggestion window of the existing communications client; and the profile presenter (**125**) hides the suggestion window of the existing communications client and creates a keyboard hook as wells as a subclass mechanism using code-injection to intercept keyboard messages and other messages sent to the hidden suggestion window of the existing communications client to prevent it from processing these keys (e.g. PgUp, PgDown, Tab, Return), and also to cause it to process fake keystrokes or other messages (e.g. to implement "delete" functionality"). In one embodiment, the keyboard hook is a global WH_KEYBOARD_LL hook; in another embodiment, the keyboard hook is a WH_KEYBOARD hook. In one embodiment, the profile presenter (**125**) creates only one such keyboard hook per thread (especially for the WH_KEYBOARD hook).

[0097] In one embodiment, the result controller is configured to handle a rapidly typing user, by stopping a previous search and then starting a new search. For example, if a search for suggestions is started based on the initial input "er" and before the suggestions are displayed the user further typed "ic" to provide the input "eric", the result controller stops the search for suggestions based on "er" and starts a new search for suggestions based on "eric".

[0098] In one embodiment, an inspector controller is used to hook the functionalities of the profile presenter (**125**) with the existing communications client, which may be the message composer (**121**) in one embodiment. When a window for the message composer (**121**) is constructed and shown, the inspector controller determines whether the user is typing in a field (e.g., entry box (**203**)) that is configured to receive an address. If the user is typing in an address field, the inspector controller instantiates a keyboard hook to obtain what the user types in the address field to allow the profile presenter (**125**) to generate the suggestions based on the profile set (**113**).

[0099] In one embodiment, the inspector controller determines if the window should have auto suggest functionality. Read-only windows or unsupported type of windows should not have the auto suggest functionality. The inspector controller then searches for suitable textboxes (using a "Control Calculation mechanism") that should have auto suggest functionality attached to them. Since the native client (e.g., Outlook) has many configurations ("use Word Editor", use "RTF Editor", etc.), different mechanisms are used and a fallback mechanism in case the initial search for a suitable window fails, and in some cases, additional code injection is required because the native client (e.g., Outlook) uses a different process for different windows. The inspector controller uses a "SetProp" mechanism to "remember" which windows have already been subclassed so that auto suggest functionality is correctly added to the newest window. Additionally, the inspector controller keeps track of which textbox (To, Cc, Bcc, etc.) the user is currently using. The inspector controller also is responsible for putting the user's selection (from the dialog) into the selected textbox control. The inspector controller also watches for various windows message in the various subclassed windows (e.g. Activation, Focus, Keyboard), "hiding" (or "eating") some of these message from the native client (e.g., Outlook) so as not to break other existing functionality of the native client (e.g., Outlook), and using others to notify the view controller that it should hide or show or change the suggestion window (**205**). In addition to the subclass of the "TextBox" and "Outlook AutoSuggest" window, the main composer window is subclassed as well, and may subclass others in the future. This subclassing mechanism is implemented using code injection, but may also be implemented using other mechanisms. Code injection "injects" code into another program (e.g. Outlook or Word) while it is running.

[0100] In one embodiment, the inspector controller subclasses the textbox (and also the parent window) into which the user is typing to receive the keys typed by the user, generate requests for new suggestions in response to keys typed by the user, scroll through the suggestions (e.g., when

the user presses arrow keys), indicate a selection by the user (e.g., when the user presses Tab or Enter key) (some embodiments use subclass, some embodiments keyboard hook), the profile presenter (**125**) hide these keystrokes from the native client (e.g., Outlook window) (by "eating" or consuming the messages), and hide the suggestion window (**205**) when the keyboard focus is moved out of the textbox or user has activated another application or user has finished selection. The inspector controller instantiates the view controller to process the user input and subclasses the suggestion window of the message composer (**121**) to create the suggestion window (**205**). The inspector controller (or the keyboard hook) contacts the view controller to process the user inputs when various keys are received in the textbox in which the user is typing.

[0101] In one embodiment, a window driver is used to work with the message composer (**121**) to obtain current caret position (the position of the text insertion point indicator). In response to the requests from the inspector controller and the suggestion window (**205**), the window driver may selectively block messages from being passed on to other windows. When a selection is made, the window driver is configured to replace the incomplete input in the address field with the address corresponding to the selection.

[0102] In one embodiment, the profile presenter (**125**) is implemented as an add-on module to a native client, such as Outlook. The Outlook textbox windows are in some cases "rich edit controls" which allow other controls to be embedded in them. The profile presenter (**125**) walk both the text and these embedded COM controls using COM (also known as OLE or ActiveX programming) to determine where the user is currently entering text (ignoring any other text or embedded controls before or after the current email address) so that the profile presenter (**125**) can correctly determine where the current user's entry begins and ends, what text to use to create suggestions for, and also which part of the text (or embedded controls) to replace.

[0103] In one embodiment, the profile presenter (**125**) uses a person-centric approach to identify the suggestions. The partial input (e.g., "Graf" received in the entry box (**203**)) is used to match the names of the persons to identify the persons in the profile set (**113**) (e.g., the first name, the last name, the nickname, etc.). The profile presenter (**125**) sorts the matched persons based on a relationship score determined from the statistical data derived at least in part from the received messages (**115**) and the sent messages (**117**).

[0104] In one embodiment, after receiving one or more letters that are typed in by an end user in an entry box configured to receive an address for an outgoing message, the profile presenter (**125**) determines whether the one or more letters match part of a name in the profile set (**113**). If there is a match (or more than one match), the profile presenter (**125**) retrieves the addresses associated with the matched name(s), sorts the addresses, and presents the addresses for selection in suggestion window (**205**).

[0105] In one embodiment, if there is no name that matches the one or more letters, the profile presenter (**125**) determines whether the one or more letters match part of an address in the profile set (**113**). If there is a match (or more than one match), the profile presenter (**125**) retrieves the matched address(es), sorts the addresses, and presents the addresses for selection in suggestion window (**205**).

[0106] Thus, the user can input the names of the recipient to cause the profile presenter (**125**) to identify the persons with names matching the input to select the person and thus select the address of the person. The names of the recipient do not have to be part of the addresses to be matched and suggested.

[0107] In one embodiment, when a person has multiple addresses, the suggestion window (**205**) shows multiple suggestions for the addresses presented with the name of the person.

[0108] In another embodiment, when a person has multiple addresses, the suggestion window (**205**) shows one entry to suggest the person. If the person is selected, the suggestion window (**205**) is then updated to show a list of addresses of the person. If the user selects one or more addresses, the profile presenter (**125**) replaces the name of the person with the selected address(es) of the person; if the user does not select any of the multiple addresses, the profile presenter (**125**) replaces the name of the person with all of the multiple addresses. In one embodiment, if the user does not select any of the multiple addresses of the person, the profile presenter (**125**) replaces the name of the person with the top ranked address of the person in the To field and inserts the other addresses of the person in the CC or BCC field (or uses the top ranked address without using the other addresses of the person).

[0109] In one embodiment, the profile presenter (**125**) attempts to first match names of the persons in the profile set (**113**) and, if there is no match in names, then match addresses of the persons in the profile set (**113**).

[0110] In another embodiments, the profile presenter (**125**) may perform a search to match the beginning of a number of fields in the profile set (**113**), such as, different fields of names of the persons in the profile set (**113**), different segments of email addresses of the persons in the profile set (**113**), the names of cities (or states or countries) of the persons in the profile set (**113**), different words in the street addresses of the persons in the profile set (**113**), different words of job titles of the persons in the profile set (**113**), screen names of the persons in the profile set (**113**), etc. Examples of different fields of names include first name, last name, middle name, nick name, etc. Examples of different segments of email addresses include segments or chunks of an email prefix separated by a separator, such as "-", ".", or "_", different levels of domain names of an email address, etc. In some embodiments, the names of the persons are as part of the email addresses. In some embodiments, the names are from fields separated from the email addresses.

[0111] When the profile presenter (**125**) searches different types of fields, a match in different types of fields may be weighted differently. For example, a match in a name field may be given a first weight (e.g., 10), a match in an email prefix segment may be given a second weight (e.g., 8), a match in the top level domain name may be given a third weight (e.g., 1), and a match in other domain names (e.g., second level domain name, or lower level domain names) may be given a fourth weight (e.g., 2), etc. When a person or an address has multiple matches in different fields, the weights for the multiple matches may be added to compute the weight for the person or address. Alternatively, the highest weight for the multiple matches can be selected as the weight for the person or address. The weights for the matched persons or addresses can be used to sort the persons or addresses for selection of up to a predetermined number of suggestions for presentation to the user. For example, the weights can be applied to the relevancy scores of the

matched persons or addresses to determine relevancy scores for suggesting the persons or addresses; and the profile presenter (**125**) selects up to a predetermined number of matched persons or addresses that have the highest relevancy scores for suggesting the persons or addresses.

[0112] In FIGS. **4** and **5**, the suggestions presented based on the profile set (**113**) are shown as a list of entries, where a typical entry includes the name and the address of the person. However, in other embodiments, the entries may further include other information, such as a photo image of the person, a job title of the person, a business association of the person, etc. In some embodiments, other details of the profile of the person are shown in a separate window when the cursor is positioned and remains positioned on the entry for the person (e.g., hovering over the entry).

[0113] In one embodiment, the profile presenter (**125**) ranks the persons for the suggestions to complete the address input, based on the relevancy index between the suggested persons and the user of the user terminal (**101**). When dealing with long lists of contacts, such a relevancy index helps users find the most relevant contacts first.

[0114] In one embodiment, the profile builder (**119**) scans the messages (**115** and **117**) to capture the addresses of the senders and recipients of the messages (**115** and **117**). Each of the captured addresses is stored in an index file. When one of those addresses is seen in a header that describes the recipients and senders, the score for the address is increased by a predetermined amount (e.g., 1). In some embodiments, the profile builder (**119**) further scans the body of the messages (**115** and **117**) to identify additional addresses.

[0115] In some embodiments, different weights/amounts are given to the score when the addresses are used in different fields (e.g., From, Reply-To, To, CC, and BCC fields of emails). For example, an amount of 3 may be given to the score when an address is found in a field that identifies the sender of the message (e.g., From or Reply-To fields of emails received by the user of the user terminal (**101**)); and an amount of 1 may be given to the score when the address is found in a field that identifies the recipient of the message (e.g., To, CC, and BCC fields of emails sent from the user of the user terminal (**101**)).

[0116] In some embodiments, the same amount is given to the score regardless of the type of fields in which the address is used (e.g., From, Reply-To, To, CC, and BCC fields of emails).

[0117] In some embodiments, the length of the elapsed time period since receipt of the message is further used to determine the amount given to the score. The occurrence of an address in a recent message can be given more weight than the occurrence of the address in a message received/sent earlier than the recent message.

[0118] Thus, after the messages (**115** and **117**) are processed, each of the addresses has a relevancy score. The higher the score, the more relevant the address is to the user of the user terminal (**101**).

[0119] In some embodiments, the relevancy score is computed from the data in the profile (**131**), such as the total number of messages received from the person (**148**), the total number of messages sent to the person (**149**), etc. In some embodiments, the number of messages are counted based on the types of fields in which the addresses appear and the time periods in which the messages are sent or received.

[0120] In one embodiment, the profile builder (**119**) further groups the addresses by the names of the corresponding persons. Each person having a name in the profile set (**113**) may have one or more addresses. In one embodiment, the scores for the addresses of the person are summed to generate a score for the person. Thus, the higher the score, the more relevant the person is to the user of the user terminal (**101**).

[0121] In one embodiment, when the suggestions are presented as a list of addresses, the scores for the addresses are used to sort the addresses. When the suggestions are presented as a list of persons, the scores for the persons are used to sort the list of names of the persons.

[0122] In another embodiment, the suggestions are presented as a list of addresses grouped according to the persons. The addresses for each person are grouped together and sorted within the group based on the scores of the emails. The groups are sorted according to the scores of the persons.

[0123] In one embodiment, the profile presenter (**125**) selects up to a predetermined number (e.g., 10) of candidates, after sorting the candidates for the suggestions based on the relevancy scores. The selected candidates are presented in the suggestion window (**205**) to help the user. Less relevant candidates are not presented in the suggestion window (**205**).

[0124] While some of the embodiments are discussed here in the context of composing an email message, the techniques disclosed here can also be applied to the specifying of address information for instant messaging, text messaging, dialing a phone number, etc. Instead of matching the incomplete input to the exact address, the profile presenter (**125**) can match the incomplete input with identifiers of the persons in the profile set (**113**) to identify the persons as the basis for suggestions. Examples of identifiers include nickname, first name, last name, company name, domain name, user name, screen name, phone number, etc. In one embodiment, the suggestions are searched and presented in the form of suggested persons and thus, the suggestions are person-centric.

[0125] In some embodiments, the user terminal (**101**) may include different communication components that require address information for different types of communications, such as email, instant messages, text messages, phone calls, etc. The profile presenter (**125**) may use the profile set (**113**) to generate the suggested candidates for completing an address input for the different types of communications.

[0126] In some embodiments, the relevancy score is computed for the person without considering the type of communications. In other embodiments, the addresses having the same type as the address to be suggested are given more weight than other types of addresses. For example, when ranking the persons for suggesting addresses for instant messaging, the addresses for instant messaging may be given more weight than addresses for other types of communications, such as email, phone calls, etc. Thus, for different types of communications, the profile presenter (**125**) may suggest different lists of persons based on the same profile set (**113**) and the same incomplete input.

[0127] In one embodiment, the profile builder (**119**) scans a set of messages (**115** and **117**) to identify addresses of senders of the messages and addresses of recipients of the

messages in an automatic way, for example, when the profile builder (**119**) is installed and/or when the messages (**115** and **117**) are received or sent.

[0128] The profile builder (**119**) identifies the names of persons at the addresses to create profiles (**131**) for the persons, based on scanning the messages (**115** and **117**) and/or querying other data sources in an automated way, such as the web search engine (**163**), online directory (**164**), social network server (**165**), and other websites.

[0129] In one embodiment, the profile builder (**119**) computes a relevancy score for each of the addresses based on a number of messages in which the addresses are used and types of fields in which the addresses are used in the messages. For example, instances where an address is used to specify a recipient of a message sent from the user of the user terminal (**101**) may be given more weight than instances where the address is used to specify a sender of a message received by the user of the user terminal (**101**).

[0130] In one embodiment, the profile builder (**119**) also computes a relevancy score for each of the persons based on the relevancy score of the addresses of each person.

[0131] In one embodiment, in response to an incomplete input in an address field, the profile presenter (**125**) identifies a set of persons by matching the incomplete input with names of the persons and/or the addresses of the persons. The profile presenter (**125**) sorts the addresses of the identified persons according to the relevancy scores of the persons and/or the relevancy scores of the addresses, and selects and presents up to a predetermined number of suggestions from the sorted addresses. In response to a user selection of one of the suggestions, the profile presenter (**125**) replaces the incomplete input with the user selected address.

[0132] FIG. **6** illustrates a system (**600**) having an enhanced client **1** (**630**), an enhanced client **2** (**640**), clients (**650***a-n*), web application (**660**), web application (**670**), and a platform (**610**). The components in FIG. **6** communicate over a network (**680**). In one embodiment, the network (**680**) is the network (**151**).

[0133] The enhanced client **1** (**630**) is a client application that runs on the user terminal (**101**) of a particular user, and provides the functionality of the components (**111-125**). The enhanced client **1** (**630**) may include any software utility or client application that provides for management of messages, address books, contact information, and calendars including, for example, those provided by Microsoft (e.g., Outlook), RIM (e.g., Blackberry), and other organizations. In one embodiment, the enhanced client **1** (**630**) may include other types of software utilities and client applications.

[0134] In one embodiment, the enhanced client **2** (**640**) may include the same types of software utilities and client applications as the enhanced client **1** (**630**).

[0135] The enhanced client **1** (**630**) includes a database (**634**), an indexing engine (**632**), and a passive address book (PAB) **1** (**636**). The database (**634**) may include contact information in a native data format of the client application, including, for example, received messages (**115**), sent messages (**117**), and contact information manually entered by a user of the user terminal (**101**). In one embodiment, the database (**634**) may include information in the address set (**111**).

[0136] The indexing engine (**632**) extracts, processes, and indexes information from the database (**634**), and stores the information in the PAB **1** (**636**). In one embodiment, the indexing engine (**632**) is, or may include, the profile builder (**119**). In one embodiment, the profile information in the PAB **1** (**636**) may include information in the profile set (**113**). In this regard, the PAB **1** (**636**) contains profile information about persons obtained from the headers of sent messages and received messages that are manually entered as well as information automatically obtained from the bodies of messages and various online sources. In one embodiment, the functionality of the indexing engine (**632**) and the PAB **1** (**636**) may be provided by an add-on utility to a base client application.

[0137] In one embodiment, the enhanced client **1** (**630**), as well as the platform (**610**) via the indexing engines (**618***a*) and (**618***c*), may additionally extract profile information over the network (**680**) from a separate web application (**670**) (e.g., Google account of the user) of a remote server by a custom communications protocol between the enhanced client **1** (**630**) and the web application (**670**). In one embodiment, the enhanced client **1** (**630**)), as well as the platform (**610**) via the indexing engines (**618***a*) and (**618***c*), may extract profile information over the network (**680**) from a separate web application (**660**) (e.g., Yahoo account of the user) of a remote server via an API (**662**) with which to extract the profile information from the web application (**600**). The profile information is stored in the PAB **1** (**636**).

[0138] The enhanced client **1** (**630**) communicates with the platform (**610**) over the network (**680**). In one embodiment, the platform (**610**) is a server or server farm. The platform (**610**) includes an XDF sync server (**612**), an indexing engine (**618***a*), a PAB **1** (**616***a*), and an LDAP server **1** (**614***a*). The indexing engine (**618***a*), the PAB **1** (**616***a*), and the LDAP server **1** (**614***a*) are associated with the user of the enhanced client **1** (**630**) and the enhanced client **2** (**640**). In one embodiment, the indexing engine (**618***a*), the PAB **1** (**616***a*), and the LDAP server **1** (**614***a*) are provided for the user or group of users. An indexing engine (**618***c*), a PAB N (**616***c*), and an LDAP server N (**614***c*) of the platform (**610**) are provided for a different Nth user or group of users.

[0139] The information in the PAB **1** (**636**) of the enhanced client **1** (**630**) may be uploaded to the platform (**610**) and synchronized with the information in the PAB **1** (**616***a*) in the platform (**610**). During a synchronization process, the profile information in the PAB **1** (**636**) may be used to create and package a data file. In one embodiment, the data file is formatted in JSON. The data file may contain a plurality of entries with each entry having profile information for various individuals including, for example, name, email address, phone number, title, position, employer, social network account identification, as well as any other desirable profile information. After the data file is packaged, the enhanced client **1** (**630**) provides the data file to the XDF sync server (**612**). The XDF sync server (**612**) analyzes the profile information in the PAB **1** (**616***a*). If the PAB **1** (**616***a*) has no profile information, the XDF sync server (**612**) copies the information from the data file into the PAB **1** (**616***a*). If the PAB **1** (**616***a*) already has profile information, the XDF sync server (**612**) synchronizes the information from the data file with the profile information in the PAB **1** (**616***a*) to generate synchronized profile information and to accordingly update the PAB **1** (**616***a*). In one embodiment, the synchronization process occurs automatically and regularly. In one embodiment, the user may initiate the synchronization process.

[0140] In one embodiment, the information in the PAB **1** (**616***a*) of the platform (**610**) may be downloaded (i.e., served) to the enhanced client **1** (**630**) and synchronized with the information in the PAB **1** (**636**) in the enhanced client **1** (**630**). During a synchronization process, the information in the PAB **1** (**636**) may be used to create and package a data file. In one embodiment, the data file is formatted in JSON. The data file may contain a plurality of entries with each entry having profile information for various individuals including, for example, name, email address, phone number, title, position, employer, social network account identification, as well as any other desirable profile information. After the data file is packaged, the enhanced client **1** (**630**) provides the data file to the XDF sync server (**612**). The XDF sync server (**612**) analyzes the profile information in the PAB **1** (**616***a*) and compares it with the data file. The platform (**610**) generates a new data file reflecting the results of the comparison between the profile information in the PAB **1** (**616***a*) and the data file created from the PAB **1** (**636**). The new data file is provided to the enhanced client **1** (**630**) and the stored in the PAB **1** (**636**).

[0141] The enhanced client **2** (**640**) is another exemplary client application that runs on the user terminal (**101**) of a different device for the user, and also provides the functionality of the components (**111-125**). In one embodiment, the enhanced client **1** (**630**) and the enhanced client **2** (**640**) are different client applications that run on different user terminals (**101**). The enhanced client **2** (**640**) includes a database (**644**), an indexing engine (**642**), and a PAB **1** (**646**). With respect to the synchronization process and communication with the LDAP server **1** (**614***a*), the operation of the enhanced client **2** (**640**) is similar to the operation of the enhanced client **1** (**630**), as described herein. Thus, the description of the enhanced client **1** (**630**) and its operation applies, as is appropriate, to the enhanced client **2** (**640**) and its operation, and vice versa.

[0142] In one embodiment, updates to the PAB **1** (**636**) of the enhanced client **1** (**630**) may be propagated to the enhanced client **2** (**640**), and vice versa. For example, an update to the PAB **1** (**636**) of the enhanced client **1** (**630**) may be uploaded to the platform (**610**) during a synchronization process. During a synchronization process between the platform (**610**) and the enhanced client **2** (**640**), the update may be downloaded to the enhanced client **2** (**640**).

[0143] The clients (**650***a-n*) are client applications that run on the user terminal (**101**) of the user. In one embodiment, the clients (**650***a-n*) are all configured to have LDAP support. The clients (**650***a-n*) may include any software utility or client application that provides for management of messages, address books, contact information, and calendars including, for example, those provided by Microsoft (e.g., Outlook), Apple (e.g., mail and address book applications for iPhone; mail.app and address book for Mac; OS X, etc.), RIM (e.g., Blackberry), and other organizations. In one embodiment, the clients (**650***a-n*) may include other types of software utilities and client applications. A local database (not shown) for each of the clients (**650***a-n*) may include information in a native data format of the client application, including, for example, received messages (**115**), sent messages (**117**), and contact information manually entered by a user of the user terminal (**101**). In one embodiment, unlike the enhanced client **1** (**630**) and the enhanced client **2** (**640**), the clients (**650***a-n*) of the user do not include a PAB **1**.

[0144] In one embodiment, an LDAP server of the platform (**610**) may separately store data for multiple users supported by the platform (**610**). For example, to communicate with the clients (**650***a-n*) of the user, the platform (**610**) includes the LDAP server **1** (**614***a*) for the user or a group of users. The platform (**610**) includes the LDAP server N (**614***c*) for the Nth user or group of users supported by the platform (**610**). In one embodiment, separate LDAP servers may be partitioned in the platform (**610**) by using OpenLDAP Software and/or other LDAP or directory-service software. In one embodiment, the LDAP server **1** (**614***a*) is provisioned to the user after the enhanced client **1** (**630**) has synchronized the profile information in the PAB **1** (**636**) with the PAB **1** (**616***a*) of the platform (**610**). In one embodiment, the user may be provided access to more than one LDAP server and the profile information in the corresponding PABs. In one embodiment, the platform (**610**) may include a dedicated LDAP server for every user supported by the platform (**610**).

[0145] As discussed in more detail below, the LDAP server **1** (**614***a*) may be accessed by the user through LDAP configuration settings unique to the user. In one embodiment, login information of each user when provided to the platform (**610**) permits the user to access only the LDAP server **1** (**614***a*), and associated profile information in the PAB **1** (**616***a*), so that the profile information in the PAB **1** (**616***a*) serves as a private data repository for the user. After the LDAP configuration settings for the user are provided to the platform, the platform (**610**) may provide (i.e., serve) profile information from the PAB **1** (**616***a*) via the LDAP server **1** (**614***a*) during use by the user of any of the clients (**650***a-n*). For example, when the user provides a partial entry of contact information in a field of the client (**650**), the LDAP server **1** (**614***a*) may provide profile information to the client (**650**) as a suggestion for the partial entry.

[0146] FIG. **7** illustrates an exemplary screen (**700**) of the enhanced client **1** (**630**) of the user. The user interface (**700**) includes a panel (**702**) that reflects the ability of the enhanced client **1** (**630**) to perform the functionality of the components (**111-125**). The panel (**702**) includes a search box (**704**) to perform searches against the PAB **1** (**636**). The panel (**702**) includes a profile section (**706**) that provides information about a contact of interest to the user. The panel (**702**) includes an actions section (**708**) to allow the user to perform various actions, such as request a phone number of the contact, schedule time with the contact, and email the contact. The panel (**702**) includes a utilities section (**710**) to allow the user to, for example, see the network of the contact, and view conversations, files exchanged, and appointments of the user.

[0147] FIG. **8** illustrates a screen **800** of the enhanced client **1** (**630**) having a pop-up window (**802**) that indicates the packaging of a data file in a synchronization process between the enhanced client **1** (**630**) and the platform (**610**). In one embodiment, the synchronization process is automatically initiated. In one embodiment, the synchronization process may be initiated by the user or by the platform (**610**) or both. As part of the synchronization process, the profile information in the PAB **1** (**636**) may be used to create and package a data file to provide to the XDF sync server (**612**). In FIG. **8**, the process of packaging the data file is indicated to the user via the pop-up window (**802**). Once the packaging is complete, the data file is provided to the platform (**610**) to continue the synchronization process so that the

profile information in the PAB **1** (**636**) of the enhanced client **1** (**630**) may be provided to the PAB **1** (**616***a*) of the platform (**610**), as discussed above.

[0148] FIG. **9** illustrates an exemplary screen (**900**) of the client (**650**) to configure the client (**650**) to allow communication with the LDAP server **1** (**614***a*) that supports the user. In one embodiment, the screen (**900**) is dynamically generated for the user after log in to the platform (**610**). Before configuration of the clients (**650***a-n*), the LDAP server **1** (**614***a*) has already been provisioned to the user. The screen (**900**) includes an automatic configuration section (**902**) that allows the user to click on a link (**904**) to automatically configure the settings related to the LDAP server **1** (**614***a*) for the particular client (**650**) through use of a configuration file. In the illustrated embodiment, clicking on the link (**904**) automatically configures the client (**650**) for the LDAP server **1** (**614***a*) when the client (**650**) is the iPhone. In one embodiment, other links may be provided to automatically configure the additional or alternate clients (**605***a-n*) for the LDAP server **1** (**614***a*).

[0149] The screen (**900**) also includes a configuration settings section (**906**). The configuration settings section (**906**) includes configuration settings data to allow communications with the LDAP server **1** (**614***a*). The configuration settings data is to be manually entered by the user into the clients (**650***a-n*) that may not have an associated link to automatically perform the configuration. In one embodiment, the configurations settings data is obtained from the configuration settings section (**906**), which includes a server field (**908**), a port field (**910**), an SSL field (**912**), a search base field (**914**), a scope field (**916**), an authentication field (**918**), a user name field (**920**), and a password field (**922**). The server field (**908**) indicates a DNS name or an IP address for a particular server associated with the platform (**610**). The port field (**910**) indicates a port number to which the client (**650**) will connect. The SSL field (**912**) indicates whether SSL will be used. The search base field (**914**) indicates the search base DN at or below which entries will be matched against a search operation. The scope field (**916**) indicates level of subtree of entries that should be considered when processing the search operation. The authentication field (**918**) indicates whether the access to the LDAP server **1** (**614***a*) requires authentication. The user name field (**920**) indicates the user name assigned to the user. The password field (**922**) indicates the password of the user. In one embodiment, the user name and the password may be unique to each user.

[0150] In one embodiment, the values provided for the fields of the configuration setting section (**906**) may be different from those described and illustrated. In one embodiment, the configuration settings data and thus the configuration setting section (**906**) may contain additional or fewer fields. In one embodiment, the set of values in the fields of the configuration setting section (**906**) may be unique to each user.

[0151] FIG. **10** illustrates a window (**1001**) of a screen (**1000**) of the client (**650**) to allow the user to manually enter the configuration settings data so that the client (**650**) is configured to communicate with the LDAP server **1** (**614***a*). The window (**1001**) includes a name field (**1002**), a server field (**1004**), a port field (**1006**), an SSL field (**1008**), a search base field (**1010**), a scope field (**1012**), an authentication field (**1014**), a user name field (**1016**), and a password field (**1018**). The name field (**1002**) indicates the name of the LDAP server **1** (**614***a*) to which the client (**650**) will connect. As shown in FIG. **10**, the configuration settings data have been manually entered into their respective fields of the window (**1001**) by the user. The user may click on a save button (**1020**) to save the entered configuration settings data. The user may click on a cancel button (**1022**) to exit from the window (**1001**).

[0152] FIG. **11** illustrates a screen (**1100**) of the client (**650**). In the illustrated embodiment, the client (**650**) is a mail application (e.g., mail.app). In one embodiment, the client (**650**) is another client application. When a new message button (**1102**) is clicked on by the user, a new message screen (**1200**) is displayed by the client (**650**), as shown in FIG. **12**. The new message screen (**1200**) includes a To: field (**1202**), a Cc: field (**1205**), and a suggestion window (**1207**) that extends from the To: field (**1202**). In the illustrated embodiment, the user has provided the partial entry "Joe" in the To: field (**1202**) to address a message to a desired contact. Based on the partial entry "Joe", the client (**650**) provides a first suggestion (**1204**) of "Joe Britton <brittonj@yahoo-inc.com>" in the To: field (**1202**) as contact information potentially desired by the user. The first suggestion (**1204**) also appears in the suggestion window (**1207**). The first suggestion (**1204**) is provided by the local database of the client (**650**).

[0153] Based on the partial entry of "Joe", a second suggestion (**1208**) of "'Joe Davis (Xobni)'<joe.davis@metova.com> as profile information potentially desired by the user also appears in a second position in the suggestion window (**1207**). The second suggestion (**1208**) is provided from the LDAP server **1** (**614***a*). In this way, the suggestion of contact information as the user enters data in a field of the client (**650**) may be supplemented by the LDAP server **1** (**614***a*) even when, in contrast to the enhanced client **1** (**630**), the client (**650**) does not have a PAB **1**.

[0154] In the illustrated embodiment, contact information provided by the LDAP server **1** (**614***a*) is displayed below contact information provided by the local database of the client (**650**). In one embodiment, the profile information provided by the LDAP server **1** (**614***a*) is sometimes or always displayed above contact information provided by the local database of the client (**650**).

[0155] In one embodiment, the profile information provided by the LDAP server **1** (**614***a*), when displayed, is marked in any manner (e.g., "Xobni") to indicate the LDAP server **1** (**614***a*) as its source. In one embodiment, the contact information provided by the local database of the client (**650**), when displayed, is marked in any manner to indicate the local database as its source. In one embodiment, the source of contact information and profile information, when displayed, is not indicated.

[0156] FIG. **13** illustrates a new message screen (**1300**) of the client (**650**). The new message screen (**1300**) includes a Cc: field (**1302**). In one embodiment, the new message screen (**1300**) is the new message screen (**1200**) after the selection of a contact desired by the user in the To: field (**1202**). In the illustrated embodiment, the user has provided the partial entry "La Donna" in the Cc: field (**1302**). Based on the partial entry "La Donna", the local database of the client (**650**) does not provide any suggestions as contact information potentially desired by the user. However, based on the partial entry "La Donna", the LDAP server **1** (**614***a*) provides a suggestion (**1304**) of "La Donna Higgins (Xobni) <ladonna.higgins@xobni.com>" as profile information

potentially desired by the user. The suggestion (**1304**) appears in the Cc: field (**1302**). Thus, even when the local database of the client (**650**) fails to provide suggested contact information, the LDAP server **1** (**614***a*) may succeed in providing profile information to the client (**650**) to display as a suggestion for the user.

[0157] FIG. **14** shows a flow chart (**1400**) for the client (**650**) to receive profile information from the LDAP server **1** (**614***a*). In (**1402**), an LDAP server is provisioned for each user of a plurality of users. In (**1404**), profile information is received. In (**1406**), the profile information is stored in a database associated with the LDAP server. In (**1408**), via the LDAP server, a suggestion from the profile information is provided in response to a partial entry provided by the user to a first client associated with the user.

[0158] Some embodiments of the enhanced client **1** (**630**, the enhanced client **2** (**640**), and the clients (**650***a-n*), and screens thereof, have been described above. In other embodiments, the client (**650**) may be any other client application with functionality for messaging, mail, contacts, calendaring, and/or other utilities.

[0159] In one embodiment, the enhanced client **1** (**630**) has been described as having the database (**634**) and the PAB **1** (**636**). In one embodiment, the information in the database (**634**) and the PAB **1** (**636**) are not identical. In one embodiment, the information in the database (**634**) and the PAB **1** (**636**) overlap with at least some identity. In one embodiment, the information in the database (**634**) and the PAB **1** (**636**) may be contained in one database with a stored indication as to the source of contact information and profile information as the database (**634**) or the PAB **1** (**636**).

[0160] In one embodiment, indications about the source for particular profile information in the PAB **1** may be stored in the PAB **1** and associated with the particular profile information.

[0161] In one embodiment, the PAB **1** (**636**), the PAB **1** (**616***a*), and the PAB **1** (**646**) contain identical profile information after appropriate synchronization.

[0162] In one embodiment, the enhanced client **1** (**630**) has been described as providing the profile information of the PAB **1** (**636**) to the PAB **1** (**616***a*) via the XDF sync server (**612**) and, in turn, serving profile information via the LDAP server **1** (**614***a*) to the client (**650**). In one embodiment, the profile information from the LDAP server **1** (**614***a*) may not come from the enhanced client **1** (**630**) or the enhanced client **2** (**640**). In one embodiment, the profile information, via the indexing engine (**618***a*) in the platform (**610**), may be provided from the web application (**670**), the web application (**660**), and/or other sources that are not the enhanced client **1** (**630**) or the enhanced client **2** (**640**). The profile information may then be provided to the client (**650**) from the PAB **1** (**616***a*) via the LDAP server **1** (**614***a*).

[0163] In one embodiment, the LDAP server **1** (**614***a*) is used to "pull" profile information from the LDAP server **1** (**614***a*) to the client (**650**), as discussed above. In one embodiment, the LDAP server **1** (**614***a*) may also be used to "push" data from the enhanced client **1** (**630**) and the enhanced client **2** (**640**). Rather than communicate with the XDF sync server (**612**), the enhanced client **1** (**630**) and the enhanced client **2** (**640**) communicate with and provide profile information to the LDAP server **1** (**614***a*). The LDAP server (**614***a*) synchronizes the profile information from the enhanced client **1** (**630**) and the enhanced client **2** (**640**) with the profile information in the PAB **1** (**616***a*).

[0164] In one embodiment, the clients (**650***a-n*) support LDAP. In one embodiment, the clients (**650***a-n*) may support additional or alternate standard or open communication protocols other than LDAP. In one embodiment, another standard or open communication protocol other than LDAP may be additionally or alternately implemented in the platform (**610**) to support communications with the clients (**650***a-n*).

[0165] In one embodiment, the LDAP server **1** (**614***a*) through the LDAP server N (**614***c*) may be implemented to provide open, standard access to contact information in social networks, such as Facebook, Myspace, and Linked-In.

[0166] FIG. **16** illustrates a server-client system for profile processing.

[0167] In one embodiment, the cloud/server will generate profile data (e.g., in a way like the profile builder (**119**)) and synchronize data among different clients connected to the cloud/server. Changes made in one application can be propagated to other clients via the cloud/server. Different clients may be associated with communication clients connected to different message servers (e.g., one for work email and one for personal email). The cloud/server will bridge the data and/or message content among different devices using the clients using a same account.

[0168] Some embodiments include additional new features related to a cloud concept as discussed below.

[0169] One embodiment includes a new way to reconcile changes made on different devices. For example, instead of asking the user to make a choice to resolve the conflicts, a cloud/server may accept all changes. The ranks will be used to present the most relevant one first. Thus, no information is lost; and the user is not burdened with requests to resolve conflicts, which provides a better user experience.

[0170] In one embodiment, the ranks (e.g., in a way similar to the profile presenter (**125**) ranking persons and/or contacts) are dependent on the devices used and/or the geographical location of the current device. Thus, the same list of contacts may be sorted differently, based on the relevancy to the particular device used to access the list and/or the current location of the device used to access the list.

[0171] In one embodiment, the ranks are provided as a function to cut down data exchange between the server and clients. The parameters for the functions are communicated from the server to the clients to allow the clients to compute the ranks based on the parameters, instead of having to obtain the updated ranks. The function may be dependent on the parameters such as time, whether a message has been read or not, the level of contact in a social network.

[0172] In one embodiment, each contact has a set of values for the parameters; when a new message is processed, the parameters of the contacts relevant to the message are updated and pushed to the client; and when ranks are needed, the client computes the ranks based on the values stored locally on the device. In one example, the rank of one contact is dependent on not only the values of the parameters for this particular contact, but also the values of the parameters for other contacts (i.e., the change in one contact can impact the ranks for a set of contacts; and when the number of parameters is smaller than the number of contacts affected, transmitting the parameters involves less data communications than transmitting the updated ranks). In another example, time-based ranking is performed, in which the formula for ranking may be a function of the time since the

last message received. If the client relies upon the server to generate these values, every time it asks for "rank updates" the server would have to send back the entire set of ranks for every person assuming time had progressed since the last update requested. To reduce the data traffic, the parameters for the time-based formula can be transmitted to the client; and the client can re-evaluate the ranks based on the parameters and the formula at different time instances without having to download the entire set of ranks from the cloud/server.

[0173] In one embodiment, the clients retrieve the messages from the message servers then send them to the profile builder on the cloud servers for processing. For example, the clients may forward a copy of the message to the profile builder on the server to generate and update the profile set. In another embodiment, the Cloud Servers themselves can fetch directly from the message servers. In another example, the user terminal may be configured to use the server to get the messages and then use the client to get the messages from the server.

[0174] FIG. **15** shows a data processing system, which can be used in various embodiments. While FIG. **15** illustrates various components of a computer system, it is not intended to represent any particular architecture or manner of interconnecting the components. Some embodiments may use other systems that have fewer or more components than those shown in FIG. **15**.

[0175] In one embodiment, the user terminal (**101**) can be implemented as a data processing system, with fewer or more components, as illustrated in FIG. **15**. When one or more components of the user terminal (**101**) are implemented on one or more remote servers, the servers can be implemented as a data processing system, with fewer or more components, as illustrated in FIG. **15**.

[0176] In FIG. **15**, the data processing system (**1500**) includes an inter-connect (**1502**) (e.g., bus and system core logic), which interconnects a microprocessor(s) (**1503**) and memory (**1508**). The microprocessor (**1503**) is coupled to cache memory (**1504**) in the example of FIG. **15**.

[0177] The inter-connect (**1502**) interconnects the microprocessor(s) (**1503**) and the memory (**1508**) together and also interconnects them to a display controller, display device (**1507**), and to peripheral devices such as input/output (I/O) devices (**1505**) through an input/output controller(s) (**1506**).

[0178] Typical I/O devices include mice, keyboards, modems, network interfaces, printers, scanners, video cameras and other devices which are well known in the art. In some embodiments, when the data processing system is a server system, some of the I/O devices, such as printer, scanner, mice, and/or keyboards, are optional.

[0179] The inter-connect (**1502**) may include one or more buses connected to one another through various bridges, controllers and/or adapters. In one embodiment, the I/O controller (**1506**) includes a USB (Universal Serial Bus) adapter for controlling USB peripherals, and/or an IEEE-1394 bus adapter for controlling IEEE-1394 peripherals.

[0180] The memory (**1508**) may include ROM (Read Only Memory), volatile RAM (Random Access Memory), and non-volatile memory, such as hard drive, flash memory, etc.

[0181] Volatile RAM is typically implemented as dynamic RAM (DRAM) which requires power continually in order to refresh or maintain the data in the memory. Non-volatile memory is typically a magnetic hard drive, a magnetic optical drive, an optical drive (e.g., a DVD RAM), or other type of memory system which maintains data even after power is removed from the system. The non-volatile memory may also be a random access memory.

[0182] The non-volatile memory can be a local device coupled directly to the rest of the components in the data processing system. A non-volatile memory that is remote from the system, such as a network storage device coupled to the data processing system through a network interface such as a modem or Ethernet interface, can also be used.

Classifying of Contact Data into Local Contacts and Directory Contacts

[0183] FIG. **17** shows a system for classifying user contact data **1703** for users (e.g., User A and User B) according to one embodiment. Server platform **1700** stores contact data **1703** and includes LDAP server **1701** and CardDAV server **1702**.

[0184] The contact data **1703** may be, for example, data extracted by profile builder **119** and stored as profile set **113**. Alternatively, the contact data may be, for example, address set **111** and/or other forms or collections of data about various persons or entities with which the user communicates using her user device. In alternative embodiments, the contact data may be collected in various other ways, such as being provided by the user or by a web service or server used by the user.

[0185] Contact data **1703** may contain, for example, over 100 contacts for each user, and in some cases over 1,000 up to several thousand contacts for each user (e.g., 1,000-15,000 contacts per user). This could be a multiple of 3-10 times or greater the number of contacts for a user in contact data **1703** as compared to local or address book contacts stored on the user's device.

[0186] In this embodiment, contact data for the respective contacts of many different users (including Users A and B) is stored on server platform **1700**. Platform **1700** may be, for example, based on server platform **610** discussed above, and modified to operate with and include a CardDAV or other address book or local contact server as described herein.

[0187] Each of the contacts is ranked (e.g., based on stored profile information for each contact). In one embodiment, this ranking is intended to correspond to or estimate the probability or likelihood that a given contact will be communicated with via a user using her user device. For example, a person that is a close relative with which the user frequently sends text messages would have higher ranking than a stranger.

[0188] In one embodiment, the profile presenter (**125**), which was discussed above as ranking persons (e.g., for suggestions to complete an address input) based on a relevancy index between the suggested persons and the user of the user terminal (**101**), may be used to provide a ranking of each contact stored on server platform **1700**. Other forms of relevancy ranking or importance ranking (e.g., other correlation methods) may also be used. The ranking may also be based on a user device context as discussed below (i.e., the ranking will depend on the actual physical device being used for communication with the persons represented by the contact data).

[0189] Using the results from this ranking, each of the contacts is classified as either a local contact or a directory contact in order to provide two sets of contacts: a local contacts set, and a directory contacts set. The local contacts will be provided for local storage on a user device of the user

(e.g., via periodic updating using a CardDAV or other address book client/server protocol). A directory contact is provided (e.g., when needed or on-demand) to the user device from the directory contacts by the server (e.g., in response to a query from the user device). The directory contacts are stored using LDAP server **1701** of server platform **1700**.

[0190] User A has multiple user devices (e.g., devices A**1** and A**2**). Similarly, user B has multiple devices (e.g., devices B**1** and B**2**). These devices may be, for example, user terminals as described above. Devices A**1**, A**2**, B**1**, and B**2** each may generate or provide contact data for the respective User A or B. These devices may communicate with server platform **1700**, for example, by network **680** or another wired or wireless network.

[0191] FIG. **18** shows a classification of contact data for User A into local contacts and directory contacts according to one embodiment. In this embodiment, all of the user's contacts are put on LDAP server **1701**, but only the most-highly ranked contacts are put on CardDAV server **1702** (these contacts are used to update the user device as local contacts or address book entries). In an alternative embodiment, Microsoft's Exchange server protocol is used instead of CardDAV.

[0192] The classification is based on the ranking above. More specifically, contacts **1803** for User A have been ranked as discussed above. This ranking is used to divide contacts **1803** into two groups: a first group of local contacts **1801** and a second group of directory contacts **1802**. This first ranking of contacts **1803** is done for device A**1** and results in a top-ranked set of N contacts **1805**, and a remaining lower-ranked set of X contacts **1806**.

[0193] A second ranking is done for contacts **1804** of User A for device A**2** (which, e.g., may have greater local memory storage capacity). Contacts **1804** may be the same contacts as contacts **1803**, or may be a different set of contacts. This second ranking of contacts **1804** results in a top-ranked set of N contacts **1807**, and a remaining lower-ranked set of X contacts **1808**. It should be noted that the particular contacts in the set of local contacts **1801** may vary between the two user devices. In addition, the number of contacts selected for inclusion in local contact **1801** may vary for each user device.

[0194] The boundary (e.g., cut-off line) used to define, after the above ranking, whether a contact is a top or bottom-ranked contact for purposes of determining inclusion in local contacts **1801** or directory contacts **1802** may vary depending on the particular implementation. In one example, a predetermined percentage of top-rated contacts may be included in local contacts **1801**. In another example, a fixed number of contacts may be included in local contacts **1801**. Further, the boundary may vary depending on the physical characteristics of the particular user device, such as memory storage capacity or processing power. In one example, the point of this boundary or cut-off is determined as a function of how many contacts that User A has in total, and how strong the signals are for each contact (these signals are factors used to rank the contacts such as frequency of communications and time of last communication, or whether or not the user has sent communications to the contact, or only received communications from the contact). For example, the N value (or number of local contacts in a set) might be 50, 100, 300, or 1,000, or more.

[0195] FIG. **19** shows a classification of contact data for User B into local contacts **1801** and directory contacts **1802** according to one embodiment. In particular, contacts **1901** for User B are ranked as discussed above. This ranking provides a top-ranked set of contacts **1903**, which will be local contacts, and further provides of lower-ranked set of contacts **1904**, which will be directory contacts. This is a first ranking done for device B**1**.

[0196] A second ranking of contacts **1902**, which may be the same or differ from contacts **1901**, is done for device B**2**. This second ranking provides a top-ranked set of contacts **1905** and a lower-ranked set of contacts **1906**. It should be noted that the boundary between local and directory contacts may be different for each user A and B.

[0197] Now discussing further various embodiments, in one embodiment, a method (e.g., implemented on server platform **1700**) includes: storing contact data for each of a plurality of users including a first user, the contact data comprising a plurality of contacts for the first user; ranking each of the plurality of contacts to provide a first ranking associated with a first user device of the first user; classifying, by a server platform, each of the plurality of contacts as either a local contact or a directory contact based on the first ranking to provide a first set of local contacts and a first set of directory contacts; providing the first set of local contacts for local storage on the first user device, wherein the first set of local contacts includes a first local contact; and providing, by the server platform, to the first user device, a first directory contact from the first set of directory contacts.

[0198] In one embodiment, the providing the first directory contact is responsive to a query from the first user device. In one embodiment, the method further includes ranking each of the plurality of contacts to provide a second ranking associated with a second user device of the first user, wherein the second ranking is based at least on a likelihood that future communication with each respective contact will be performed via the second user device.

[0199] The likelihood of future communication may be determined based at least on a prior communication history with the respective contact. The first local contact may be stored on the first user device as a first option to be provided to the first user when completing an input field of a user interface of the first user device, and the first directory contact may be provided as a second option to be provided to the first user when completing the input field.

[0200] The providing the first directory contact may be responsive to a query from the first user device, and the method may further include providing a second directory contact and a third directory contact, each from the set of directory contacts, to the first user device responsive to the query, wherein the first, second, and third directory contacts are provided in a ranked order based on the first ranking. The ranking of each of the plurality of contacts may involve ranking each contact using unique profile information for the contact (e.g., profile information collected by profile builder **119** may be ranked as discussed above to provide a ranking for each contact). In one embodiment, the first set of local contacts is provided to the first user device via an address book server of the server platform, and the first directory contact is provided to the first user device via a directory server of the server platform.

[0201] In one embodiment, the first ranking is based at least on a user context of the first user device, and the

method further includes: ranking the plurality of contacts based on at least a user context of a second user device of the first user to provide a second ranking associated with the second user device; and classifying, by the server platform, each of the plurality of contacts as either a local contact or a directory contact based on the second ranking to provide a second set of local contacts and a second set of directory contacts; providing the second set of local contacts for local storage on the second user device; and providing a second directory contact, from the second set of directory contacts, to the second user device. The user context of the first user device may be based at least on one of a physical characteristic of the first user device, and a time of day.

[0202] In one embodiment, a system (e.g., a user terminal or user device) includes: a display for presenting a user interface to a user; memory storing a first set of local contacts, wherein the first set of local contacts includes a first local contact, wherein the first set of local contacts has been provided from a server platform, and wherein each of the local contacts has a rank determined by the server platform; at least one processor; and memory storing instructions configured to instruct the at least one processor to: provide the first local contact as a first option for a user when completing an input field of the user interface; send a query to the server platform when the user is completing the input field; receive, from the server platform, a first directory contact responsive to the query, wherein the first directory contact is one of a first set of directory contacts stored at the server platform, and wherein each of the directory contacts has a rank determined by the server platform; and provide the first directory contact as a second option for the user when completing the input field.

[0203] In one embodiment, the server platform: stores contact data for each of a plurality of users including the user, and the contact data comprises a plurality of contacts for the user; ranks each of the plurality of contacts to provide a ranking; classifies each of the plurality of contacts as either a local contact or a directory contact based on the ranking to define the first set of local contacts and the first set of directory contacts; provides the first set of local contacts for local storage on a user device of the user; and provides the first directory contact from the first set of directory contacts.

[0204] In one embodiment, the instructions are further configured to instruct the at least one processor to receive, from the server platform, when the user is completing the input field, a second directory contact and a third directory contact, wherein the first, second, and third directory contacts are each presented on the display in a ranked order based on a respective rank determined by the server platform.

[0205] In one embodiment, a non-transitory computer-readable storage medium stores computer-readable instructions, which when executed, cause a system to: store contact data for each of a plurality of users including a first user, the contact data comprising a plurality of contacts for the first user; rank each of the plurality of contacts to provide a ranking; classify, by at least one processor, each of the plurality of contacts as either a local contact or a directory contact based on the ranking to provide a first set of local contacts and a first set of directory contacts; provide the first set of local contacts for local storage on a user device of the user, wherein the first set of local contacts includes a first local contact; and provide, by the at least one processor, to the user device, a first directory contact from the first set of directory contacts.

[0206] The ranking may be based at least on a likelihood that future communication with each respective contact of the plurality of contacts will be performed via the user device. The first local contact may be stored on the user device as a first option to be provided to the first user when completing an input field of a user interface of the user device, and the first directory contact may be provided as a second option to be provided to the first user when completing the input field.

[0207] In one embodiment, the providing the first directory contact is responsive to a query from the user device, and the instructions further cause the system to provide a second directory contact and a third directory contact, each from the first set of directory contacts, to the user device responsive to the query, wherein the first, second, and third directory contacts are provided in a ranked order based on the ranking.

[0208] The first set of local contacts may be provided to the user device via an address book server, and the first directory contact is provided to the user device via a directory server. The user device may be a first user device, the ranking may be based at least on a user context of the first user device, and the instructions may further cause the system to: rank the plurality of contacts based at least on a user context of a second user device of the user to provide a second ranking associated with the second user device; and classify each of the plurality of contacts as either a local contact or a directory contact based on the second ranking to define a second set of local contacts and a second set of directory contacts; provide the second set of local contacts for local storage on the second user device; and provide a second directory contact, from the second set of directory contacts, responsive to a query from the second user device. The user context of the first user device may be based at least on a physical characteristic of the first user device.

[0209] FIG. **20** shows a configuration screen **2000** of a user device (e.g., a user terminal) to provide settings for communicating with LDAP server **1701** according to one embodiment. The settings illustrated in this example are for an Apple iPhone device and include a server name **2001** for LDAP server **1701**.

[0210] FIG. **21** shows a configuration screen **2100** of the user device of FIG. **20** to provide settings for communicating with CardDAV server **1702** according to one embodiment. The settings illustrated in this example are for an Apple iPhone device and include a server name **2101** for CardDAV server **1702**. In one example, the CardDAV or Exchange protocols are used by a phone or client software to ask for contacts at a specific server address. The user is given the proper settings to put into the user's phone or email client. Then, that phone or client requests the contacts from server platform **1700**, which may be set up so that for each user, the top contacts are added to the user's account from a master contact database (e.g., contact data **1703**).

[0211] FIG. **22** shows a screen **2200** of a user device for a user to input data **2202** to create a new message according to one embodiment. In particular, in this embodiment as illustrated, a user has already entered data **2202** as text characters into an input field of a user interface (e.g., a user interface of an e-mail or SMS client application) for creating a new e-mail or text message. When the user types "frank",

the user initially sees local contacts **2201** (e.g., as options that can be selected to complete the user input). These local contacts are provided from local storage (e.g., an address book) on the user device (i.e., these are contacts that were previously obtained from local contacts **1801** of server platform **1700**). These local contacts are obtained, for example, via CardDAV server **1702** (e.g., from periodic updating of the user device), or in alternative embodiments using a different local contact or address book protocol. It should be noted that local contacts stored on the user device may be used for various other functions on the user device, and are not limited solely to user input completion.

[0212] FIG. **23** shows the screen of FIG. **22** a few seconds after the user has input a few initial characters and already viewed the local contacts, according to one embodiment. More specifically, a few seconds after seeing local contacts **2201** as discussed above, the user sees directory contacts **2302**. These directory contacts are provided in response to a query sent from the user device to server platform **1700** (e.g., when the user first starts to enter data **2202** into the input field).

[0213] Further, the local contacts **2201** and/or the directory contacts **2302** typically will change (i.e., the specific contacts listed will be different) as the user enters additional characters or other data. Additional directory contacts may be sent in response to additional queries from the user device that reflect changes in the user context of the device (e.g., which may include the type of user interface being presented, and/or other user selections, and/or device activity such as location determination, or processing by one or more applications running on the user device) and/or the specific data already provided by the user. The user context may be determined by server platform **1700** based on a user device identifier along with prior received data and/or data sent with a query from the user device.

[0214] FIG. **24** shows a screen of a user device that displays a call history **2400** including local contacts provided via a CardDAV server according to one embodiment. A contact **2401** is presented in call history **2400**. Information for contact **2401** is provided from a local contact previously obtained from local contacts **1801** of server platform **1700**.

[0215] FIG. **25** shows a screen **2500** of a user device that displays a local contact **2502** (provided via a CardDAV or type of local contact/address book server as discussed above) when the user is performing a local contact search (e.g., by typing in characters **2501**) according to one embodiment.

[0216] FIG. **26** shows a screen of a user device displaying a voicemail list **2600** including local contacts **2601** provided via a CardDAV server according to one embodiment.

[0217] FIG. **27** shows a screen **2700** of a user device displaying caller ID information, previously provided to the user device via information associated with local contacts **1801** from CardDAV server **1702**, for a local contact (Brian Kobashikawa) that is calling the user device according to one embodiment. In one example, since the server platform **1700** can constantly update photos for CardDAV server **1702**, and those photos are pulled onto the user device, contact images can be modified on the server platform **1700** to include more timely and useful information than prior contact photo approaches. This presented information may include profile information of the contact and other information associated with the person, including events **2701** about or related to the contact (e.g., other information may include information regarding how the user knows the contact/person, the latest email the contact sent that the user has not yet responded to, an important update of the contact from a social networking service, important news about that person's city or company, or other photos of the person). The user device (e.g., phone) itself just gets a new image with this information included. The user device does not have to process this data any differently than the user device currently processes other address book or local contact information.

[0218] Various additional non-limiting embodiments and examples are discussed below. In a first embodiment, all contacts and their related information are put in LDAP server **1701**. This is useful because many phones and email software clients support finding LDAP contacts from a server. LDAP server **1701** can return search results in a ranked order based on the query. This is different from a typical LDAP server, which simply returns results in alphabetical order, based on the search query.

[0219] The LDAP server/protocol is treated as a search directory by most existing phones and email clients. This means that the contacts from such an LDAP server are not imported onto the user device/phone (e.g., an iPhone device). If an iPhone is set up to connect to an LDAP server, those LDAP contacts do not get added to the user's local address book, caller ID, or favorites. They only appear when the user is typing an email address or composing a text message. As the user types the name or address of a person in those contexts, the iPhone queries the LDAP server with what the user is typing to see if there are any results from the LDAP server in addition to other results already presented on the user's phone.

[0220] There are two server protocols, CardDAV and Exchange, that the iPhone imports into the phone itself and makes available throughout all of the applications on the phone (Caller ID, favorites, etc.). As discussed above, in this embodiment, all contacts are put on an LDAP server (e.g., these are contacts extracted from communications to and from a user device, as supplemented with information gathered from third party servers such as social network servers). The top-ranked contacts are made available to the user device via a CardDAV or Exchange server as described above.

[0221] These top-ranked contacts are now available throughout all of the iPhone's contact-related functions (and not just search functions). The cut-off or threshold for what is "immediately relevant" (used to determine local contacts) may be based on a "likelihood to be in touch in the next 3 months (or other predetermined time period)" score derived from previous communication history and activity on the user device. This likelihood may be determined, for example, based on the signals/factors mentioned above (for ranking) and then further using an algorithm that ranks all of that data in the context of whether the user would need to contact the specific person, or need caller ID to be displayed, for this person.

[0222] FIG. **28** shows a classification **2800** of a portion of contact data for User A into local contacts according to an alternative embodiment. This embodiment is similar to those discussed above except that all of the contacts **2801** are handled as directory contacts (i.e., there is some overlapping of classification in that the local contacts **2802** are also members of the set **2801** of directory contacts). More specifically, when server platform **1700** communicates with,

for example, an iPhone device, a subset (i.e., the top-ranked contacts) are available via CardDAV (as local contacts) as described above. However, in this embodiment, all of the contacts (e.g., all Xobni contacts) are available to the iPhone device via LDAP (as directory contacts). In other words, the local contacts do not need to be removed from the directory contacts (i.e., the directory contacts here includes all contacts). The logic described above for creating a threshold to use for selecting local contacts similarly may be used in this embodiment.

[0223] FIG. **29** shows a configuration screen **2900** of a user device to provide settings for communicating with CardDAV server **1702** according to an alternative embodiment. The settings illustrated here include a server name **2901** for CardDAV server **1702** (here the server is a Xobni server). This embodiment is otherwise similar to that of FIG. **21** above.

[0224] FIG. **30** shows a configuration screen **3000** of the user device of FIG. **29** to provide settings for communicating with LDAP server **1701** according to an alternative embodiment. The settings illustrated in this example are for an Apple iPhone device and include a server name **3001** for LDAP server **1701** (here the server is a Xobni server). This embodiment is otherwise similar to that of FIG. **20** above.

[0225] In yet another embodiment, a server as described above can be configured to be in CardDAV Directory mode. Specifically, the local contacts can be provided via CardDAV and the directory contacts can be provided via CardDAV Directory. Other directory-style API interfaces may be used in other variations.

[0226] In this description, various functions and operations may be described as being performed by or caused by software code to simplify description. However, those skilled in the art will recognize that what is meant by such expressions is that the functions result from execution of the code/instructions by a processor, such as a microprocessor. Alternatively, or in combination, the functions and operations can be implemented using special purpose circuitry, with or without software instructions, such as using Application-Specific Integrated Circuit (ASIC) or Field-Programmable Gate Array (FPGA). Embodiments can be implemented using hardwired circuitry without software instructions, or in combination with software instructions. Thus, the techniques are limited neither to any specific combination of hardware circuitry and software, nor to any particular source for the instructions executed by the data processing system.

[0227] While some embodiments can be implemented in fully functioning computers and computer systems, various embodiments are capable of being distributed as a computing product in a variety of forms and are capable of being applied regardless of the particular type of machine or computer-readable media used to actually effect the distribution.

[0228] At least some aspects disclosed can be embodied, at least in part, in software. That is, the techniques may be carried out in a computer system or other data processing system in response to its processor, such as a microprocessor, executing sequences of instructions contained in a memory, such as ROM, volatile RAM, non-volatile memory, cache or a remote storage device.

[0229] Routines executed to implement the embodiments may be implemented as part of an operating system or a specific application, component, program, object, module or sequence of instructions referred to as "computer programs." The computer programs typically include one or more instructions set at various times in various memory and storage devices in a computer, and that, when read and executed by one or more processors in a computer, cause the computer to perform operations necessary to execute elements involving the various aspects.

[0230] A machine readable medium can be used to store software and data which when executed by a data processing system causes the system to perform various methods. The executable software and data may be stored in various places including for example ROM, volatile RAM, non-volatile memory and/or cache. Portions of this software and/or data may be stored in any one of these storage devices. Further, the data and instructions can be obtained from centralized servers or peer to peer networks. Different portions of the data and instructions can be obtained from different centralized servers and/or peer to peer networks at different times and in different communication sessions or in a same communication session. The data and instructions can be obtained in entirety prior to the execution of the applications. Alternatively, portions of the data and instructions can be obtained dynamically, just in time, when needed for execution. Thus, it is not required that the data and instructions be on a machine readable medium in entirety at a particular instance of time.

[0231] Examples of computer-readable media include but are not limited to recordable and non-recordable type media such as volatile and non-volatile memory devices, read only memory (ROM), random access memory (RAM), flash memory devices, floppy and other removable disks, magnetic disk storage media, optical storage media (e.g., Compact Disk Read-Only Memory (CD ROMS), Digital Versatile Disks (DVDs), etc.), among others. The computer-readable media may store the instructions.

[0232] The instructions may also be embodied in digital and analog communication links for electrical, optical, acoustical or other forms of propagated signals, such as carrier waves, infrared signals, digital signals, etc. However, propagated signals, such as carrier waves, infrared signals, digital signals, etc. are not tangible machine readable medium and are not configured to store instructions.

[0233] In general, a tangible machine readable medium includes any apparatus that provides (i.e., stores and/or transmits) information in a form accessible by a machine (e.g., a computer, network device, personal digital assistant, manufacturing tool, any device with a set of one or more processors, etc.).

[0234] In various embodiments, hardwired circuitry may be used in combination with software instructions to implement the techniques. Thus, the techniques are neither limited to any specific combination of hardware circuitry and software nor to any particular source for the instructions executed by the data processing system.

[0235] Although some of the drawings illustrate a number of operations in a particular order, operations which are not order dependent may be reordered and other operations may be combined or broken out. While some reordering or other groupings are specifically mentioned, others will be apparent to those of ordinary skill in the art and so do not present an exhaustive list of alternatives. Moreover, it should be recognized that the stages could be implemented in hardware, firmware, software or any combination thereof.

[0236] In the foregoing specification, the disclosure has been described with reference to specific exemplary embodiments thereof. It will be evident that various modifications may be made thereto without departing from the broader spirit and scope as set forth in the following claims. The specification and drawings are, accordingly, to be regarded in an illustrative sense rather than a restrictive sense.

1-21. (canceled)

22. A method, comprising:

storing contact data for each of a plurality of users including a first user, the contact data comprising a plurality of contacts for the first user, the contact data further comprising data associated with messages sent to the first user by each of the plurality of contacts;

ranking each of the plurality of contacts to provide a first ranking associated with a first user device of the first user, the first ranking based at least in part on the data associated with the messages sent to the first user by each contact;

dividing, by a server platform, based on the first ranking, the plurality of contacts to provide a first set of local contacts for local storage on the first user device and a first set of directory contacts stored only on the server platform;

providing, by the server platform, to the first user device, the first set of local contacts for the local storage on the first user device; and

in response to a query from the first user device, providing, by the server platform, to the first user device, a first plurality of contacts in a first ranked order based on the first ranking, the first plurality of contacts including a first directory contact from the first set of directory contacts, wherein the first directory contact is provided as an option for presentation in a user interface to the first user when entering data in an input field.

23. The method of claim **22**, wherein the first set of local contacts is provided for the local storage prior to the first user device presenting the input field to the first user for entering the data.

24. The method of claim **22**, further comprising ranking each of the plurality of contacts to provide a second ranking associated with a second user device of the first user, wherein the second ranking is based at least on a likelihood that future communication with each respective contact will be performed via the second user device.

25. The method of claim **24**, wherein the likelihood of future communication is determined based at least on a prior communication history with the respective contact.

26. The method of claim **22**, further comprising providing a second directory contact and a third directory contact, each from the first set of directory contacts, to the first user device responsive to the query, wherein the first, second, and third directory contacts are presented to the first user in the user interface in a ranked order based on the first ranking.

27. The method of claim **22**, wherein the ranking of each of the plurality of contacts comprises ranking each contact using unique profile information for the contact.

28. The method of claim **22**, wherein the first set of local contacts is provided to the first user device via an address book server of the server platform, and the first directory contact is provided to the first user device via a directory server of the server platform.

29. The method of claim **22**, wherein the first ranking is further based at least on a user context of the first user device, the method further comprising:

ranking the plurality of contacts based on at least a user context of a second user device of the first user to provide a second ranking associated with the second user device; and

classifying, by the server platform, each of the plurality of contacts as either a local contact or a directory contact based on the second ranking to provide a second set of local contacts and a second set of directory contacts;

providing the second set of local contacts for local storage on the second user device; and

providing a second directory contact, from the second set of directory contacts, to the second user device.

30. The method of claim **29**, wherein the user context of the first user device is based at least on a physical characteristic of the first user device, or a time of day.

31. A system, comprising:

at least one processor; and

memory storing instructions configured to instruct the at least one processor to:

store contact data for each of a plurality of users including a first user, the contact data comprising a plurality of contacts for the first user, the contact data further comprising data associated with messages sent to the first user by each of the plurality of contacts;

rank each of the plurality of contacts to provide a first ranking associated with a first user device of the first user, the first ranking based at least in part on the data associated with the messages sent to the first user by each contact;

divide, by a server platform and based on the first ranking, the plurality of contacts to provide a first set of local contacts for local storage on the first user device and a first set of directory contacts stored only on the server platform;

provide, to the first user device, the first set of local contacts for the local storage on the first user device; and

in response to a query from the first user device, provide, to the first user device, a first plurality of contacts in a first ranked order based on the first ranking, the first plurality of contacts including a first directory contact from the first set of directory contacts, wherein the first directory contact is provided as an option for presentation in a user interface to the first user when entering data in an input field.

32. The system of claim **31**, wherein the first set of local contacts includes a first local contact to present to the first user from the local storage in response to the query from the first user device.

33. The system of claim **31**, wherein the instructions are further configured to instruct the at least one processor to provide, from the server platform, when the first user is entering the data into the input field, a second directory contact and a third directory contact, wherein the first, second, and third directory contacts are each presented on a display in a ranked order based on a respective rank determined by the server platform.

34. The system of claim **31**, wherein the contact data for a respective contact of the plurality of contacts further

comprises data extracted from the messages sent to the first user by the respective contact.

35. The system of claim **31**, wherein each of the first set of local contacts has a higher rank than any of the first set of directory contacts.

36. A non-transitory computer-readable storage medium storing computer-readable instructions, which when executed, cause a system to:

store contact data for each of a plurality of users including a first user, the contact data comprising a plurality of contacts for the first user, the contact data further comprising data associated with messages sent to the first user by each of the plurality of contacts;

rank each of the plurality of contacts to provide a first ranking associated with a first user device of the first user, the first ranking based at least in part on the data associated with the messages sent to the first user by each contact;

divide, based on the first ranking, the plurality of contacts to provide a first set of local contacts for local storage on the first user device and a first set of directory contacts stored only on a server platform;

provide, to the first user device, the first set of local contacts for the local storage on the first user device; and

in response to a query from the first user device, provide, to the first user device, a first plurality of contacts in a first ranked order based on the first ranking, the first plurality of contacts including a first directory contact from the first set of directory contacts, wherein the first directory contact is provided for presentation in the user interface to the first user when entering data in an input field.

37. The non-transitory computer-readable storage medium of claim **36**, wherein the ranking is based at least on a likelihood that future communication with each respective contact of the plurality of contacts will be performed via the first user device.

38. The non-transitory computer-readable storage medium of claim **36**, wherein the instructions further cause the system to provide a second directory contact and a third directory contact, each from the first set of directory contacts, to the first user device responsive to the query, wherein the first, second, and third directory contacts are to be presented to the first user in a ranked order based on the ranking.

39. The non-transitory computer-readable storage medium of claim **36**, wherein the first set of local contacts is provided to the first user device via an address book server, and the first directory contact is provided to the first user device via a directory server.

40. The non-transitory computer-readable storage medium of claim **36**, wherein the ranking is further based at least on a user context of the first user device, and the instructions further cause the system to:

rank the plurality of contacts based at least on a user context of a second user device of the first user to provide a second ranking associated with the second user device; and

classify each of the plurality of contacts as either a local contact or a directory contact based on the second ranking to define a second set of local contacts and a second set of directory contacts;

provide the second set of local contacts for local storage on the second user device; and

provide a second directory contact, from the second set of directory contacts, responsive to a query from the second user device.

41. The non-transitory computer-readable storage medium of claim **40**, wherein the user context of the first user device is based at least on a physical characteristic of the first user device.

* * * * *