



- (51) **International Patent Classification:**
G06F 21/30 (2013.01)
- (21) **International Application Number:**
PCT/US2014/035030
- (22) **International Filing Date:**
22 April 2014 (22.04.2014)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (30) **Priority Data:**
13/913,232 7 June 2013 (07.06.2013) US
- (71) **Applicant:** APPLE INC. [US/US]; 1 Infinite Loop, Cupertino, CA 95014 (US).
- (72) **Inventors:** HUHNS, Derrick, S.; 1 Infinite Loop, Cupertino, CA 95014 (US). WERNER, Jeremy, M.; 1 Infinite Loop, Cupertino, CA 95014 (US). PATTEKAR, Amol, V.; 1 Infinite Loop, Cupertino, CA 95014 (US).
- (74) **Agents:** FERRAZANO, Michael, J. et al.; Womble Carlyle Sandridge & Rice LLP, 10050 North Wolfe Road, Suite 260, Cupertino, CA 95014 (US).

(81) **Designated States** (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) **Designated States** (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— with international search report (Art. 21(3))

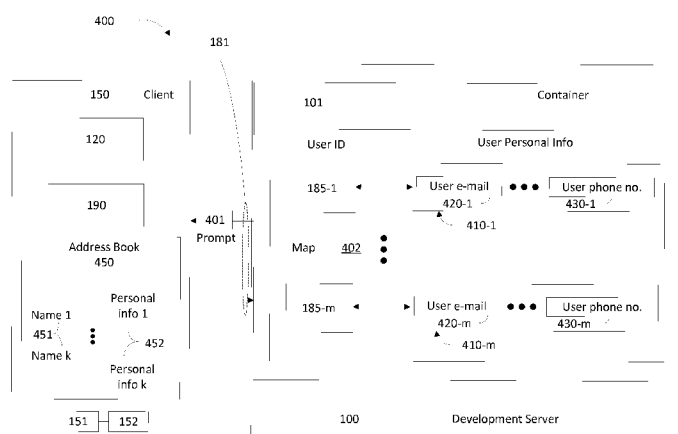
(54) **Title:** METHODS AND SYSTEMS FOR SINGLE SIGN-ON WHILE PROTECTING USER PRIVACY

FIG. 4

(57) **Abstract:** A method of enabling applications to reference user information is provided, including receiving a request for a user identifier that references a user of the application and sending a second request for the user identifier to a server. The second request may include a second user identifier that references the user and a second authentication token for the second user identifier. Furthermore, the second user identifier and the second authentication token are not accessible by the user. The method includes receiving the user identifier and an authentication token for the first user identifier. The user identifier corresponds to the second identifier; and providing the user identifier and authentication token to the application. A method of enabling an application to identify users associated with a user of the application is provided; the method may include receiving, from the server, user identifiers that reference one or more users scoped to the application.



METHODS AND SYSTEMS FOR SINGLE SIGN-ON WHILE PROTECTING USER PRIVACY

FIELD OF THE DESCRIBED EMBODIMENTS

5 [0001] The described embodiments relate generally to methods, devices, and systems for developing applications in datacenters. More particularly, embodiments disclosed herein relate to methods and systems for single sign-on while protecting user privacy and address book discovery for users of network-based software applications.

10 BACKGROUND

[0002] The proliferation of client computing devices—such as smart phones and tablets—has drastically changed the manner in which software applications are designed and executed. Software applications rely on accessing server computing devices such as a development server that are designed to interact with the software applications. When a software application contacts the server, a user authentication takes place in order for the server to release and provide data and services. In many instances, the application provider or developer handles the user personal information in the sign-on process with the development server. While this configuration simplifies operation of the software application, it exposes the privacy of the users to a third party (at least the developer). As developers take advantage of the information they may handle, users become bombarded with unwanted advertisements, solicitation, and e-mail spam. Eventually, the software application becomes unpopular and loses user subscriptions.

25 [0003] Therefore, what is desired is a sign-on configuration for a network-based software application that protects user privacy from third parties.

SUMMARY OF THE DESCRIBED EMBODIMENTS

[0004] In a first embodiment, a method for enabling applications to reference user information is provided. The method may include receiving, from an application, a first request for a first user identifier that references a user of the application and sending a second request for the first user identifier to a server. Accordingly, the second request includes a second user identifier that references the user and a second authentication token for the second user identifier, where the second user identifier and the second authentication token are not accessible by the user. The method may

30

also include receiving, from the server, the first user identifier and a first authentication token for the first user identifier, where, in some embodiments, the first user identifier corresponds to the second identifier. Finally, the method includes providing the first user identifier and the first authentication token to the application.

5 [0005] In a second embodiment, a method for enabling an application to identify one or more users associated with a first user of the application is provided. The method includes receiving, from an application, contact information that can be used to identify one or more users. The method further includes receiving, from the application, a request for one or more user identifiers that reference the one or more
10 users, and sending, to a server, the contact information and the request for one or more user identifiers that reference the one or more users. In some embodiments, the method includes receiving, from the server, the one or more user identifiers that reference one or more users. Furthermore, the method may include receiving, from the server, a correlation between the one or more user identifiers and the one or more
15 users, and, further, providing the one or more user identifiers and the correlation between the one or more user identifiers and the one or more users to the application.

[0006] In a third embodiment, a method for enabling an application to identify one or more users associated with a first user of the application is provided. The method includes receiving, from an application, a request for one or more user
20 identifiers that reference one or more users in a contact list of the first user, wherein the contact list is not accessible to the application. Also, the method may include sending, to a server, the request for one or more user identifiers that reference one or more users in the contact list, receiving, from the server, the one or more user identifiers that reference one or more users, and, finally, providing the one or more
25 user identifiers to the application.

[0007] Other aspects and advantages of the invention will become apparent from the following detailed description taken in conjunction with the accompanying drawings which illustrate, by way of example, the principles of the described embodiments

30 BRIEF DESCRIPTION OF THE DRAWINGS

[0008] The described embodiments may be better understood by reference to the following description and the accompanying drawings. Additionally, advantages of the described embodiments may be better understood by reference to the following description and accompanying drawings. These drawings do not limit any changes in

form and detail that may be made to the described embodiments. Any such changes do not depart from the spirit and scope of the described embodiments.

[0009] FIG. 1 illustrates a block diagram of a development server and a client device in a single sign-on configuration, according to some embodiments.

5 [0010] FIG. 2 illustrates a block diagram of a client device adapted for a single sign-on configuration, according to some embodiments.

[0011] FIG. 3 illustrates a block diagram of a container in a development server adapted for a single sign-on configuration, according to some embodiments.

10 [0012] FIG. 4 illustrates a block diagram of a client and a container in an address book discovery configuration, according to some embodiments.

[0013] FIG. 5 illustrates a flow chart including steps in a method for a single sign-on configuration, according to some embodiments.

[0014] FIG. 6 illustrates a flow chart including steps in a method for a single sign-on configuration including multiple user identifiers, according to some embodiments.

15 [0015] FIG. 7 illustrates a flow chart including steps in a method for discovering address books in network-based software applications, according to some embodiments.

[0016] FIG. 8 illustrates a flow chart including steps in a method for discovering address books in network-based software applications, including receiving
20 information selected from an address book, according to some embodiments.

[0017] FIG. 9 illustrates a flow chart including steps in a method for discovering address books in network-based software applications, according to some embodiments.

25 [0018] FIG. 10 illustrates a flow chart including steps in a method for discovering address books in network-based software applications as above, including discovering users with given addresses, according to some embodiments.

[0019] FIG. 11 illustrates a flow chart including steps in a method for creating e-mail user lists, according to some embodiments.

30 [0020] In the figures, elements referred to with the same or similar reference numerals include the same or similar structure, use, or procedure, as described in the first instance of occurrence of the reference numeral.

DETAILED DESCRIPTION OF SELECTED EMBODIMENTS

[0021] Representative applications of methods and apparatus according to the present application are described in this section. These examples are being provided

solely to add context and aid in the understanding of the described embodiments. It will thus be apparent to one skilled in the art that the described embodiments may be practiced without some or all of these specific details. In other instances, well known process steps have not been described in detail in order to avoid unnecessarily obscuring the described embodiments. Other applications are possible, such that the following examples should not be taken as limiting.

[0022] In the following detailed description, references are made to the accompanying drawings, which form a part of the description and in which are shown, by way of illustration, specific embodiments in accordance with the described embodiments. Although these embodiments are described in sufficient detail to enable one skilled in the art to practice the described embodiments, it is understood that these examples are not limiting; such that other embodiments may be used, and changes may be made without departing from the spirit and scope of the described embodiments.

[0023] Use of network-based software applications has rapidly increased due to the availability of a large variety of portable or handheld electronic devices capable of coupling to a network. A client device, such as a smart phone, a tablet computer, or a laptop computer may run the software application by using remote resources accessible through a network server. The network server may be a development server providing storage and data processing capabilities to the software application (hereinafter referred to as 'application'). To run the application, the client device presents a unique authentication token to sign-on with the development server. With the authentication token, the development server may determine the attributes and privileges of the user handling the client device as regards to the software application. Thus, in embodiments disclosed herein, the user personal information is kept private from a third party.

[0024] In embodiments consistent with the present disclosure, the application logs-in the user without asking the user for private information or any other registration protocol. The authentication process is transparent to the user handling the client device, thus creating a pleasant experience. Any authenticated information or private information from the user is hidden from the application itself. The application sees a stable user ID consistent with data lists stored in the development server. Thus, according to embodiments disclosed herein, a single sign-on configuration protecting the user privacy is able to create stable user IDs for use

across multiple applications. No private information from the user is shared with the application or the application developers. This is all the more relevant for users executing multiple applications, some of which may link the client device to a third party network server. The user handling the client device may desire to keep personal information private, and not share it with the third party network servers.

[0025] FIG. 1 illustrates a block diagram of a development server 100 and a client device 150 in a single sign-on configuration 10, according to some embodiments. In single sign-on configuration 10, a client device 150 interacts with a development server 100 through a network link 181. Each of the client device 150, the development server 100, and the user electronic device 150 may include a memory circuit and a processor circuit. The memory circuits store commands and data that, when executed, cause the processor circuits to perform operations in accordance with embodiments disclosed herein. For example, client device 150 includes processor circuit 151 and memory circuit 152; and development server 100 includes processor circuit 111 and memory circuit 112.

[0026] Development server 100 includes a plurality of containers 101-1, 101-2, and 101-3 (collectively referred to hereinafter as containers 101). Containers 101 include data structures associated with Applications 120-1 (Application A), 120-2 (Application B), and 120-3 (Application C), generically referred to hereinafter as application 120. Containers 101 include data structures that are manipulated by processor circuit 111 upon request by client device 150. The specific correlation between containers 101 and Applications 120-1, 120-2, and 120-3 may not be one-to-one. For example, FIG. 1 illustrates Application A associated with containers 101-1 and 101-2, Application B associated with container 101-3, and Application C also associated with container 101-3. Containers 101 may be shared by multiple applications and multiple client devices. One of skill in the art will recognize that the specific number of containers 101 and applications in development server 100 is not limiting of embodiments consistent with the present disclosure. In other words, any number of applications and containers 101 may be included in development server 100. Furthermore, the number of applications 120 may be different from the number of containers 101.

[0027] Client device 150 includes an application 120 and a daemon 190. Application 120 may be any type of network-based executable software code, including any one of applications 120-1, 120-2, and 120-3. Daemon 190 may include

an independent operating system process generated by applications executing on the client device. In some embodiments, single sign-on configuration 10 involves implementing the daemon 190 on each of the client computing devices to interact with development server 100.

5 [0028] Accordingly, in some embodiments, development server 100 receives a request from client device 150 to create or lookup a given user identifier (ID) 185. Development server 100 then determines whether a user handling client device 150 is already registered with development server 100. If the user is not registered, then development server 100 creates user ID 185 associated with a Destination Signaling
10 ID (DSID) 192. Client device 150 including DSID 191 may be associated with an authentication token 192. Authorization token 191 may be an encrypted code such as a password so that server 100 securely identifies client device 150. In some embodiments, authentication token 192 may also be associated to application 120. In that regard, when the user handling client device 150 taps on application 120,
15 application 120 may prompt the user to enter authentication token 192. Development server 100 then returns user ID 185 associated with DSID 191 and authorization token 191 scoped to a container in the development server. DSID 191 may be an identifier for a client device, and authentication token 192 may be a password or some other encrypted code associated with DSID 191 and the application name. For example, a
20 given application may have a password for access by a user handling the client device.

[0029] FIG. 2 illustrates a block diagram of a client device 150 adapted for single sign-on configuration 10, according to some embodiments. Client device 150 includes application 120, daemon 190, processor circuit 151, and memory circuit 152.
25 In some embodiments, application 120 further includes an application name 271 and a container name 272 that client device 150 provides daemon 190 to use in single sign-on configuration 10. Daemon 190 includes DSID 191 and authentication token 191, as discussed above. In some embodiments, authentication token 192 is provided to daemon 190 by the user, through a password or some other code, signature, or
30 biometric measure.

[0030] According to some embodiments, daemon 190 provides DSID 191, authorization token 192, and application name 271 to development server 100. Daemon 190 may receive user ID 185 from development server 100 to execute application 120 in client device 150. Daemon 190 may transmit requests from

development server 100 to client device 150 during execution of application 120. Such requests may be related to security procedures and maintenance operations by development server 100.

5 [0031] Client device 150 may be a portable electronic device such as a laptop or a tablet computer, a handheld electronic device such as a smart phone or any other cellular phone. In some embodiments, client device 150 may be a web site located in a network server operated by the client. Further, in some embodiments client device 150 may be a website located in a network server operated by a third party vendor, and a user of application 120 logs in to the website.

10 [0032] FIG. 3 illustrates a block diagram of a container 101 in development server 100 adapted for single sign-on configuration 10, according to some embodiments. Container 101 may include a table 300 having lines associating each of a plurality of DSIDs 191-1 through 191-n and each of a plurality of authentication tokens 192-1 through 192-n, with a plurality of user IDs 185-1 through 185-n. The value 'n' can be
15 any number of lines in table 300, as one of ordinary skill will recognize.

[0033] In some embodiments of single sign-on configuration 10, development server 100 receives a request including application name 271 from client device 150. Then, development server 100 identifies any one of containers 101 (cf. FIG. 1) associated with application name 271. Development server 100 then determines
20 whether there is an association (mapping) between DSID 191 and a user ID 185 in the identified container. If no association exists between DSID 191 and a user ID 185, then development server 100 creates user ID 185. Development server then adds a line to table 300 in the container. The line associates the newly created user ID 185 and DSID 192 of client device 150 requesting access to application 120.
25 Accordingly, development server 100 returns newly created user ID 185 and authentication token 192 to client device 150. If there is an association between DSID 191 and user ID 185 in table 300, then development server 100 returns existing user ID 185 and authentication token 192 to client device 150. Thus, embodiments consistent with the present disclosure enable network-based software applications to
30 be executed from client device 150 using a single sign-on procedure while protecting user privacy. For example, client device 150 may sign-on to development server 100 only once and be ready to execute a plurality of applications 120. Since table 300, including user IDs 185 associated with DSIDs 191 and authentication tokens 192, is

stored in a container in the development server, a smooth and secure operation is obtained.

[0034] FIG. 4 illustrates a block diagram of a client device 150 and a container 101 in an address book discovery configuration 400, according to some embodiments.

5 Client device 150 includes application 120, daemon 190, processor circuit 151, and memory circuit 152, as described in detail above (cf. FIGS. 1- 3). Client device 150 may also include an address book 450. In some embodiments address book 450 may be stored in memory circuit 152. Accordingly, address book 450 includes a plurality of names 451 (names 1 through k). Associated with each of the plurality of names

10 451, address book 450 includes a plurality of lists 452 (lists 1 through k). Lists 452 include personal information for each name 451. In some embodiments, names 451 correspond to users subscribed to application 120. Further according to some embodiments, names 451 may correspond to users subscribed to an application contained in development server 100. Personal information 452 may include a user e-

15 mail, a user phone number, address, mailing address, and other personal information associated with the name. Address book discovery configuration 400 may include a map 402 in container 101. Map 402 includes a list of user IDs 185-1 through 185-m. Each of user IDs 185-1 through 185-m is associated with a user personal information 410- through 410-m. User personal information 410 includes a user e-mail 420, a

20 user phone no. 430, and any other personal information associated with user ID 185.

[0035] In some embodiments of address book discovery configuration 400, developer server 100 provides a prompt 401 to client device 150 to allow discovery of address book 450. Upon acceptance by the user handling client device 150, developer server 100 may perform a discovery of address book 450. The discovery may include

25 searching for matches between entries in address book 450 with any of the entries in map 402 or in table 300 in container 101. For example, a match between a name 451 and a user ID 185 in table 300 (cf. FIG. 3) may be found during discovery. Accordingly, server 100 searches for matches in address book 450 with maps 400 and tables 300 in a plurality of containers 101. For example, some embodiments of an

30 address book discovery configuration include search for matches of address book 450 in containers 101 not necessarily associated with application 120.

[0036] FIG. 5 illustrates a flow chart including steps in a method 500 for single sign-on configuration, according to some embodiments. A single sign-on configuration as illustrated in FIG. 5 may include a development server coupled to a

client device through a network link (e.g., single sign-on configuration 10, development server 100, and client device 150, cf. FIG. 1 above). The client device may be executing a network-based software application (e.g., application 120, cf. FIG. 1). Steps in method 500 may be performed partially or completely by a processor circuit in the development server, where the processor circuit executes commands stored in a memory circuit (e.g., processor circuit 111 and memory circuit 112, cf. FIG. 1). In some embodiments, steps in method 500 may be performed partially or completely by a processor circuit in the client device, the processor circuit executing commands stored in a memory circuit (e.g., processor circuit 151, and memory circuit 152, cf. FIG. 1).

[0037] Step 510 includes receiving a first request for a first user identifier (e.g., user ID 185). The request may be provided by a server upon a user attempt to execute the network-based software application. Step 520 includes sending a second request for the first user identifier to a server (e.g., the developer server). For example, in some embodiments the request may be generated by a second network-based software application executed by the user handling the client device. In some embodiments step 520 may include providing the server a DSID an authentication token (e.g., DSID 191 and authentication token 192, cf. FIG. 1). Step 530 includes receiving from the server the first user identifier and the authentication token (e.g., user ID 185, and authentication token 192, cf. FIG. 1). According to some embodiments, steps 520 and 530 may be partially or totally performed by a daemon in the client device (e.g., daemon 190, cf. FIG. 1). Step 540 includes providing the first user identifier and the first authentication token to the application. Accordingly, in some embodiments step 540 is performed by the daemon in the client device.

[0038] FIG. 6 illustrates a flow chart including steps in a method 600 for single sign-on configuration including multiple user identifiers, according to some embodiments. A single sign-on configuration as illustrated in FIG. 6 may include a development server coupled to a client device through a network link (e.g., single sign-on configuration 10, development server 100, and client device 150, cf. FIG. 1 above). The client device may be executing a network-based software application (e.g., application 120, cf. FIG. 1). Steps in method 600 may be performed partially or completely by a processor circuit in the development server, where the processor circuit executes commands stored in a memory circuit (e.g., processor circuit 111 and memory circuit 112, cf. FIG. 1). In some embodiments, steps in method 600 may be

performed partially or completely by a processor circuit in the client device, the processor circuit executing commands stored in a memory circuit (e.g., processor circuit 151, and memory circuit 152, cf. FIG. 1).

5 [0039] Step 610 may include receiving from the application a request for one or more user IDs (e.g., user ID 185, cf. FIG. 1). Step 620 may include sending to a server (e.g., the development server) the request for one or more user IDs from step 610. Step 630 may include receiving from the server one or more user IDs. Accordingly, in some embodiments steps 610 through 630 are performed partially or totally by the development server coupled to a daemon process in the client device
10 daemon 190, cf. FIG. 1).

[0040] FIG. 7 illustrates a flow chart including steps in a method 700 for discovering address books in network-based software applications, according to some embodiments. Method 700 may include a development server coupled to a client device through a network link (e.g., development server 100, and client device 150,
15 cf. FIG. 1 above). Steps in method 700 may be performed partially or completely by a processor circuit in the development server, where the processor circuit executes commands stored in a memory circuit (e.g., processor circuit 111 and memory circuit 112, cf. FIG. 1). In some embodiments, steps in method 700 may be performed partially or completely by a processor circuit in the client device, the processor circuit
20 executing commands stored in a memory circuit (e.g., processor circuit 151, and memory circuit 152, cf. FIG. 1).

[0041] Step 710 includes querying an address book in the client device. In some embodiments, step 710 may include providing a prompt requesting an address book query from a user handling the client device, by the server (e.g., prompt 401, cf. FIG.
25 4). Step 720 includes providing the address book to a server. Step 730 includes selecting vetted addresses from the server. Accordingly, step 730 may include comparing addresses in the address book with addresses in a container stored in the server (e.g., one of containers 101, cf. FIG. 1). A vetted address may be an address of a registered user included in the container. Step 740 includes querying user IDs for e-mail and phone numbers in the address book. Step 750 includes providing a contact-to-user ID mapping to the client device. Accordingly, step 750 may include creating a map in the container, or adding a line to an already existing map in the container (e.g., map 402, cf. FIG. 4).
30

[0042] FIG. 8 illustrates a flow chart including steps in a method 800 for discovering address books in network-based software applications, including receiving information selected from an address book, according to some embodiments. Method 800 may include a development server coupled to a client device through a network link (e.g., development server 100, and client device 150, cf. FIG. 1 above). The address book in method 800 may be stored in the client device (e.g., address book 450, cf. FIG. 4). The client device may be executing a network-based software application (e.g., application 120, cf. FIG. 1). Steps in method 800 may be performed partially or completely by a processor circuit in the development server, where the processor circuit executes commands stored in a memory circuit (e.g., processor circuit 111 and memory circuit 112, cf. FIG. 1). In some embodiments, steps in method 800 may be performed partially or completely by a processor circuit in the client device, the processor circuit executing commands stored in a memory circuit (e.g., processor circuit 151, and memory circuit 152, cf. FIG. 1). Furthermore, steps in method 800 may be performed by a daemon process coupling the client device with the server through a network link (e.g., daemon 190 and link 181, cf. FIG. 1).

[0043] Step 810 includes receiving from the application contact information that can be used to identify one or more users. Accordingly, step 810 may include personal information retrieved from the address book in the client device (e.g., personal information 452, FIG. 4). Step 820 includes receiving from the application a request for one or more user identifiers. The one or more user identifiers may be associated with one or more network-based software applications being executed by one or more client devices. Step 830 includes sending to a server (e.g., development server 100) the contact information and the request for the one or more user identifiers. Step 840 includes receiving from the server one or more user identifiers that reference one or more users. For example, in some embodiments a single user handling a client device may be associated with one or more user identifiers for one or more applications. Step 850 includes receiving from the server a correlation between a user identifier and a user. Accordingly, step 850 may include receiving a map from the server (e.g., map 402, cf. FIG. 4). Step 860 includes providing to the application a user identifier and a correlation between a user identifier and a user. Accordingly, step 860 may be performed by the daemon interacting with the network-based software application in the client device.

[0044] FIG. 9 illustrates a flow chart including steps in a method 900 for discovering address books in network-based software applications, according to some embodiments. Method 900 may include a development server coupled to a client device through a network link (e.g., development server 100, and client device 150, cf. FIG. 1 above). The address book in method 900 may be stored in the client device (e.g., address book 450, cf. FIG. 4). Steps in method 900 may be performed partially or completely by a processor circuit in the development server, where the processor circuit executes commands stored in a memory circuit (e.g., processor circuit 111 and memory circuit 112, cf. FIG. 1). In some embodiments, steps in method 900 may be performed partially or completely by a processor circuit in the client device, the processor circuit executing commands stored in a memory circuit (e.g., processor circuit 151, and memory circuit 152, cf. FIG. 1). Steps in method 900 may be performed by the server interacting with a daemon process running in the client device (daemon 190, cf. FIG. 1).

[0045] Step 910 includes requesting user IDs and an address book to the client device. Step 920 includes querying an address book in the client device. Step 920 may include searching in the address book for names that may be stored in a container included in the server (e.g., names 450 and container 101, cf. FIG. 4). Step 930 includes receiving the address book in the server (e.g., the development server). Accordingly, step 930 may include receiving the entire address book from the client device. In some embodiments step 930 includes receiving in the server only the portions of the address book that have a match between a name in the address book and a name stored in the container. Step 940 includes mapping contacts to network-based software applications scoped to the user IDs. Accordingly, step 940 may include adding lines in a map included in a container. The added lines may include user e-mail values and user phone number values retrieved from the address book (e.g., user e-mails 420 and user phone nos. 430, cf. FIG. 4). Step 950 includes providing the application scoped user IDs to a user registered in the server. Accordingly, the user registered in the server may be a different user than the user handling the client device from which the address book has been queried.

[0046] FIG. 10 illustrates a flow chart including steps in a method 1000 for discovering address books in network-based software applications, including discovering users with given addresses, according to some embodiments. Method 1000 may include a development server coupled to a client device through a network

link (e.g., development server 100, and client device 150, cf. FIG. 1 above). Address books in method 1000 may be stored in the client device (e.g., address book 450, cf. FIG. 4). Steps in method 1000 may be performed partially or completely by a processor circuit in the development server, where the processor circuit executes commands stored in a memory circuit (e.g., processor circuit 111 and memory circuit 112, cf. FIG. 1). In some embodiments, steps in method 1000 may be performed partially or completely by a processor circuit in the client device, the processor circuit executing commands stored in a memory circuit (e.g., processor circuit 151, and memory circuit 152, cf. FIG. 1). Steps in method 900 may be performed by the server interacting with a daemon process running in the client device (daemon 190, cf. FIG. 1).

[0047] Step 1010 includes retrieving an address book from a user registered to a server. Step 1020 includes discovering users with given addresses in the address book. For example, step 1020 may include finding a match for a known address stored in a container included in the server with an address in the address book included in the client device. Step 1030 includes receiving an address book in the server. Accordingly, step 1030 may include receiving the entire address book from the client device. In some embodiments step 1030 includes receiving in the server only the portions of the address book that have a match between an address in the address book and an address stored in a container (e.g., container 101, cf. FIG. 4). Step 1040 includes mapping contacts to network-based software applications scoped to the user IDs. Accordingly, step 1040 may include associating a user ID and user personal information in a map (e.g., map 402) with a plurality of applications included in the container. Step 1050 includes providing the scoped user IDs to a user registered in the server. Accordingly, the user registered in the server may be a different user than the user handling the client device from which the address book has been queried.

[0048] FIG. 11 illustrates a flow chart including steps in a method 1100 for creating e-mail user lists, according to some embodiments. Method 1100 may include a development server coupled to a client device through a network link (e.g., development server 100, client device 150, and network link 181, cf. FIG. 1 above). Steps in method 1100 may be performed partially or completely by a processor circuit in the development server, where the processor circuit executes commands stored in a memory circuit (e.g., processor circuit 111 and memory circuit 112, cf. FIG. 1). In

some embodiments, steps in method 1100 may be performed partially or completely by a processor circuit in the client device, the processor circuit executing commands stored in a memory circuit (e.g., processor circuit 151, and memory circuit 152, cf. FIG. 1).

5 **[0049]** Step 1110 includes requesting user permission for discovery scope in an address book. The address book in method 1100 may be stored in the client device (e.g., address book 450, cf. FIG. 4). If the user does not accept in step 1120, then step 1125 includes making the address book not discoverable. If the user accepts in step 1120, step 1130 includes making the address book discoverable. For example, step 10 1130 may include providing a daemon in the client device with a configuration that allows the server to perform a method such as methods 700, 800, and 900, above. Step 1140 determines if the user accepts e-mails from the server. If the user rejects e-mails from the server, step 1145 includes removing the user from an e-mail list. If the user accepts e-mails from the server, step 1150 includes adding the user to the e-mail 15 list. Step 1160 includes receiving a user interface from the server (e.g., developer server 100) to generate e-mails. Step 1170 includes providing the server a center to send e-mails to users in the e-mail list. Step 1180 includes allowing users to toggle discoverability settings and e-mail alert settings. In some embodiments, the toggles may be grouped according to a plurality of containers in the server (e.g., containers 20 101, cf. FIG. 1).

[0050] The various aspects, embodiments, implementations or features of the described embodiments can be used separately or in any combination. Various aspects of the described embodiments can be implemented by software, hardware or a combination of hardware and software. The described embodiments can also be 25 embodied as computer readable code on a computer readable medium for controlling manufacturing operations or as computer readable code on a computer readable medium for controlling a manufacturing line. The computer readable medium is any data storage device that can store data which can thereafter be read by a computer system. Examples of the computer readable medium include read-only memory, 30 random-access memory, CD-ROMs, HDDs, DVDs, magnetic tape, and optical data storage devices. The computer readable medium can also be distributed over network-coupled computer systems so that the computer readable code is stored and executed in a distributed fashion.

[0051] The foregoing description, for purposes of explanation, used specific nomenclature to provide a thorough understanding of the described embodiments. However, it will be apparent to one skilled in the art that the specific details are not required in order to practice the described embodiments. Thus, the foregoing

5 descriptions of specific embodiments are presented for purposes of illustration and description. They are not intended to be exhaustive or to limit the described embodiments to the precise forms disclosed. It will be apparent to one of ordinary skill in the art that many modifications and variations are possible in view of the above teachings.

What is claimed is:

1. A method for enabling applications to reference user information, the method comprising:
 - 5 receiving, from an application, a first request for a first user identifier that references a user of the application;
 - sending a second request for the first user identifier to a server, the second request including:
 - a second user identifier that references the user, and
 - 10 a second authentication token for the second user identifier, wherein the second user identifier and the second authentication token are not accessible by the user;
 - receiving, from the server, the first user identifier and a first authentication token for the first user identifier, wherein the first user identifier
 - 15 corresponds to the second identifier; and
 - providing the first user identifier and the first authentication token to the application.
2. The method of claim 1, wherein the second request further includes a container identifier that identifies a container associated with the application, and the
- 20 first user identifier is accessible by applications when the applications are associated with the container.
3. The method of claim 1, further comprising:
 - receiving, at the server from a computing device, the second request, including
 - the second user identifier and the second authentication token;
 - 25 determining whether an association exists between the second user identifier and an application-accessible user identifier; and
 - sending the application-accessible user identifier and an associated application-accessible authentication token to the computing device, wherein the first user identifier is the application-accessible user
 - 30 identifier.
4. The method of claim 3, wherein:
 - receiving the second request includes receiving a container identifier that
 - identifies a container associated with the application, and identifying a
 - container association with the container identifier; and

determining whether an association exists further comprises determining whether an association exists between the container identifier and the second user identifier, wherein the first user identifier is accessible by applications associated with the container when the association exists.

5 5. The method of claim 3, further comprising:

in response to determining that an association does not exist between the second user identifier and an application-accessible user identifier: generating a new application-accessible user identifier, and creating an association between the second user identifier and the new application-accessible user identifier.

10

6. The method of claim 5, wherein the association is represented by a mapping table, and creating an association between the second user identifier and the new application-accessible user identifier comprises creating an entry in the mapping table that maps the second user identifier to the new application-accessible user identifier.

15

7. The method of claim 3, wherein the association is represented by a mapping table, and determining whether an association exists between the second user identifier and an application-accessible user identifier comprises searching the mapping table for an entry that includes the second user identifier.

20

8. The method of claim 1, wherein the authentication token can be used to verify an identity of the user.

9. A method for enabling an application to identify one or more users associated with a first user of the application, the method comprising:

receiving, from an application, contact information that can be used to identify one or more users;

25

receiving, from the application, a request for one or more user identifiers that reference the one or more users;

sending, to a server, the contact information and the request for one or more user identifiers that reference the one or more users;

30

receiving, from the server, the one or more user identifiers that reference one or more users, and a correlation between the one or more user identifiers and the one or more users; and

providing the one or more user identifiers and the correlation between the one or more user identifiers and the one or more users to the application.

10. The method of claim 9, wherein the contact information comprises a list of email addresses associated with the one or more users, and the correlation between the one or more user identifiers and the one or more users includes a mapping between the one or more user identifiers and email addresses in the list of email addresses that
5 correspond to the one or more user identifiers.
11. A method of enabling an application to identify one or more users associated with a first user of the application, the method comprising:
receiving, from an application, a request for one or more user identifiers that
reference one or more users in a contact list associated with the first
10 user, wherein the contact list is not accessible to the application;
sending, to a server, the request for one or more user identifiers that reference one or more users in the first user's contact list;
receiving, from the server, the one or more user identifiers that reference one or more users; and
15 providing the one or more user identifiers to the application.
12. The method of claim 11, further comprising sending, to the server, the first user's contact list.
13. A method for discovering address books in network-based software applications, the method comprising:
20 at a client device:
querying an address book in the client device;
providing the address book to a server;
selecting at least one vetted address from the server;
querying user identifiers (IDs) for e-mail and phone numbers in the
25 address book; and
providing a contact-to-user ID mapping to the client device.
14. The method of claim 13, further comprising providing a prompt to a user handling the client device.
15. The method of claim 13, wherein selecting the at least one vetted address from
30 the server comprises comparing addresses in the address book with addresses in a container that is stored in the server.
16. The method of claim 15, wherein selecting the at least one vetted address from the server comprises selecting an address of a registered user included in the container.

17. The method of claim 15, further comprising creating a map in the container.
18. The method of claim 17, wherein creating a map in the container comprises adding a line to an already existing map in the container, the line including a user ID associated with personal information, the personal information comprising a user e-mail and a user phone number.
19. A method for creating e-mail user lists, the method comprising:
 - issuing a request, to a user, for discovery scope in an address book;
 - making the address book discoverable;
 - adding the user to an e-mail list stored in a server; and
 - receiving a user interface from the server to generate e-mails.
20. The method of claim 19, further comprising allowing a user in the e-mail list to toggle discoverability settings and e-mail alert settings.

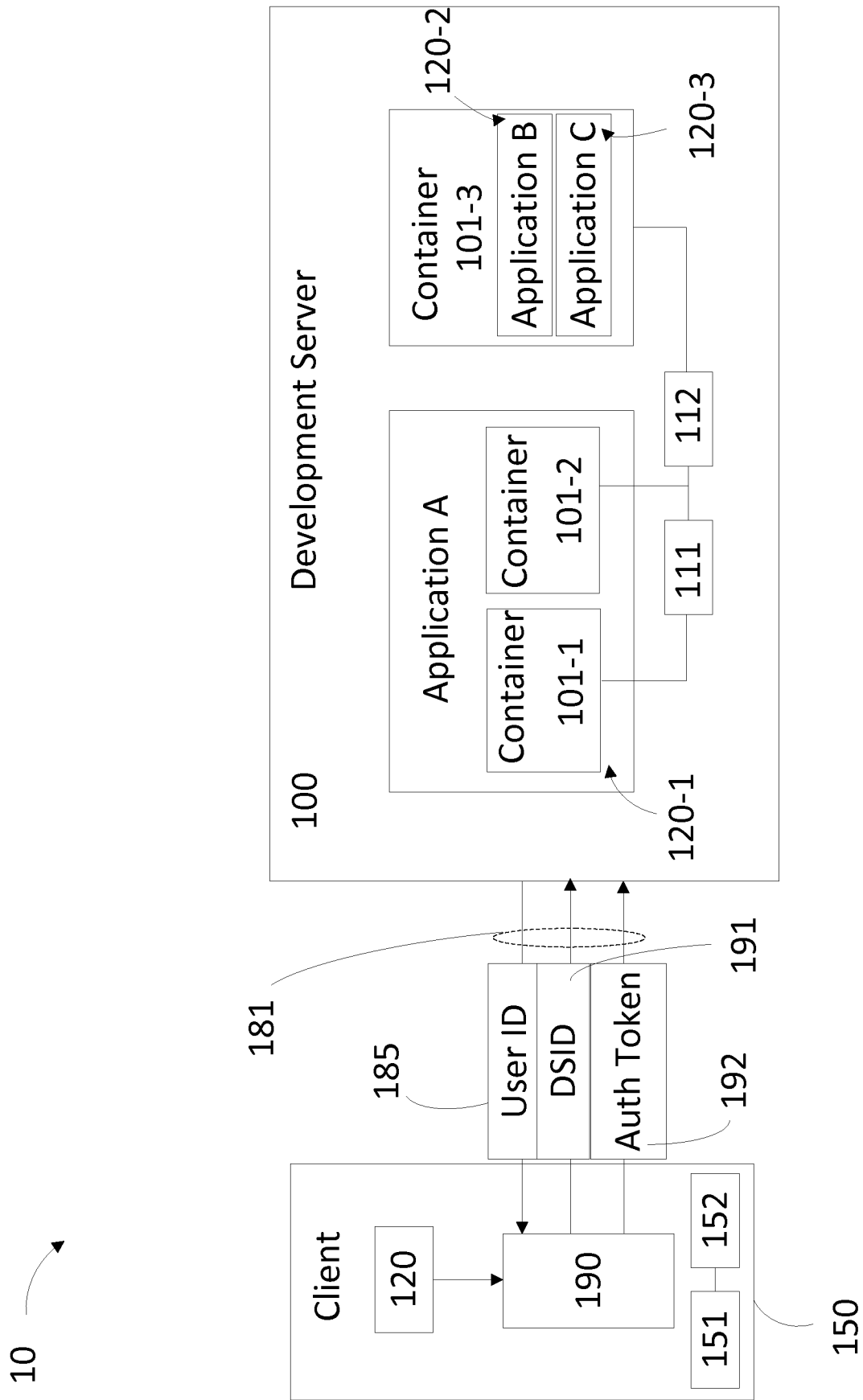


FIG. 1

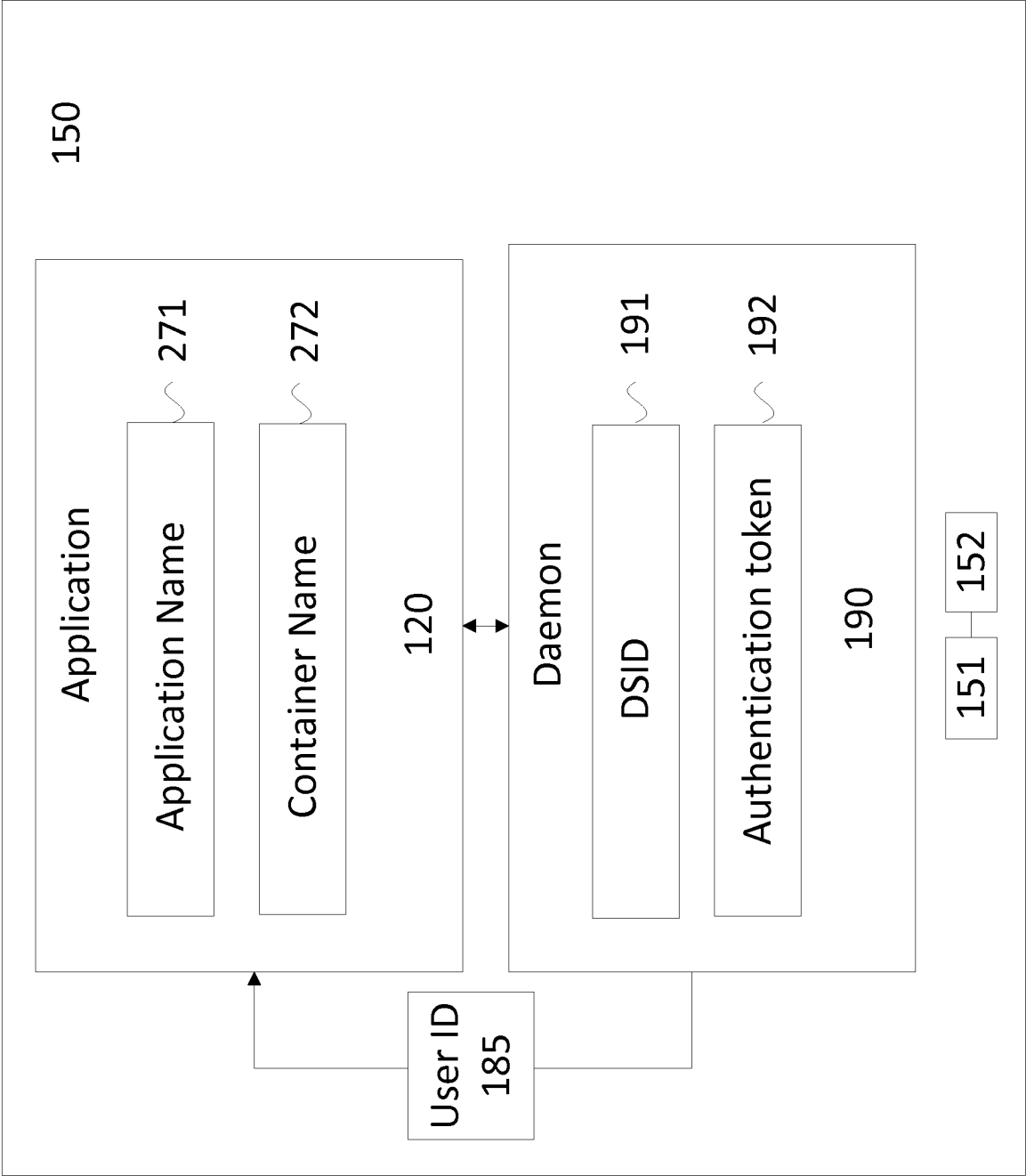


FIG. 2

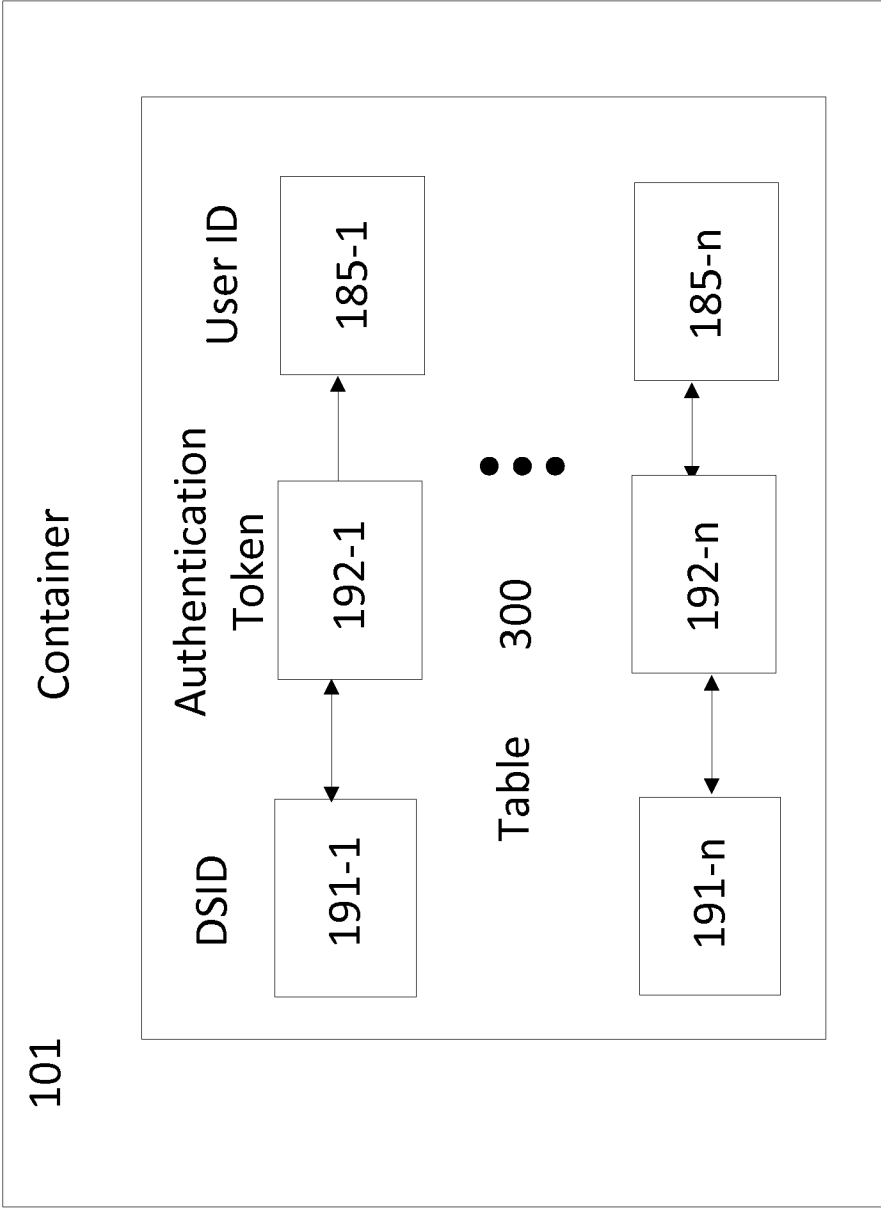


FIG. 3

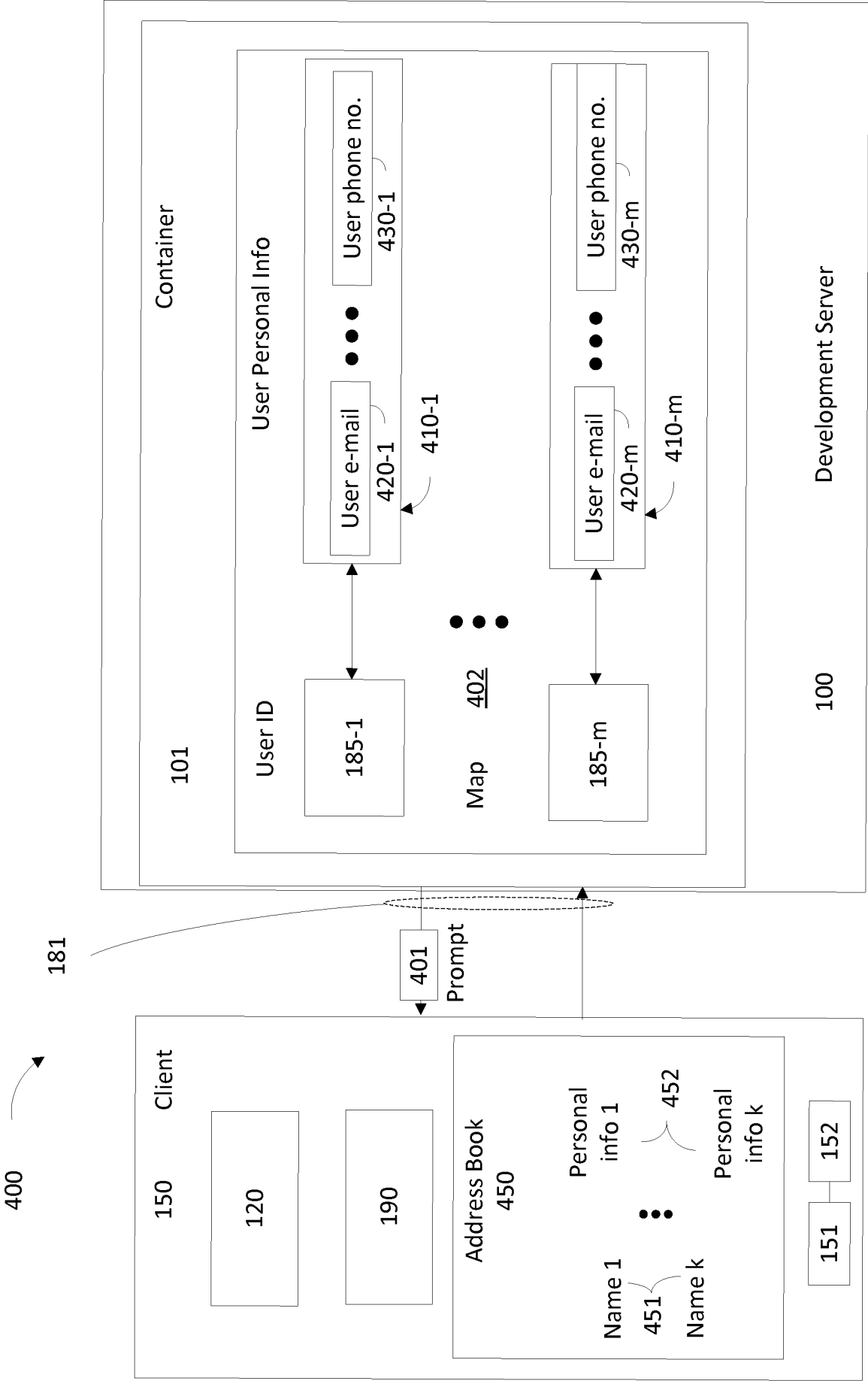


FIG. 4

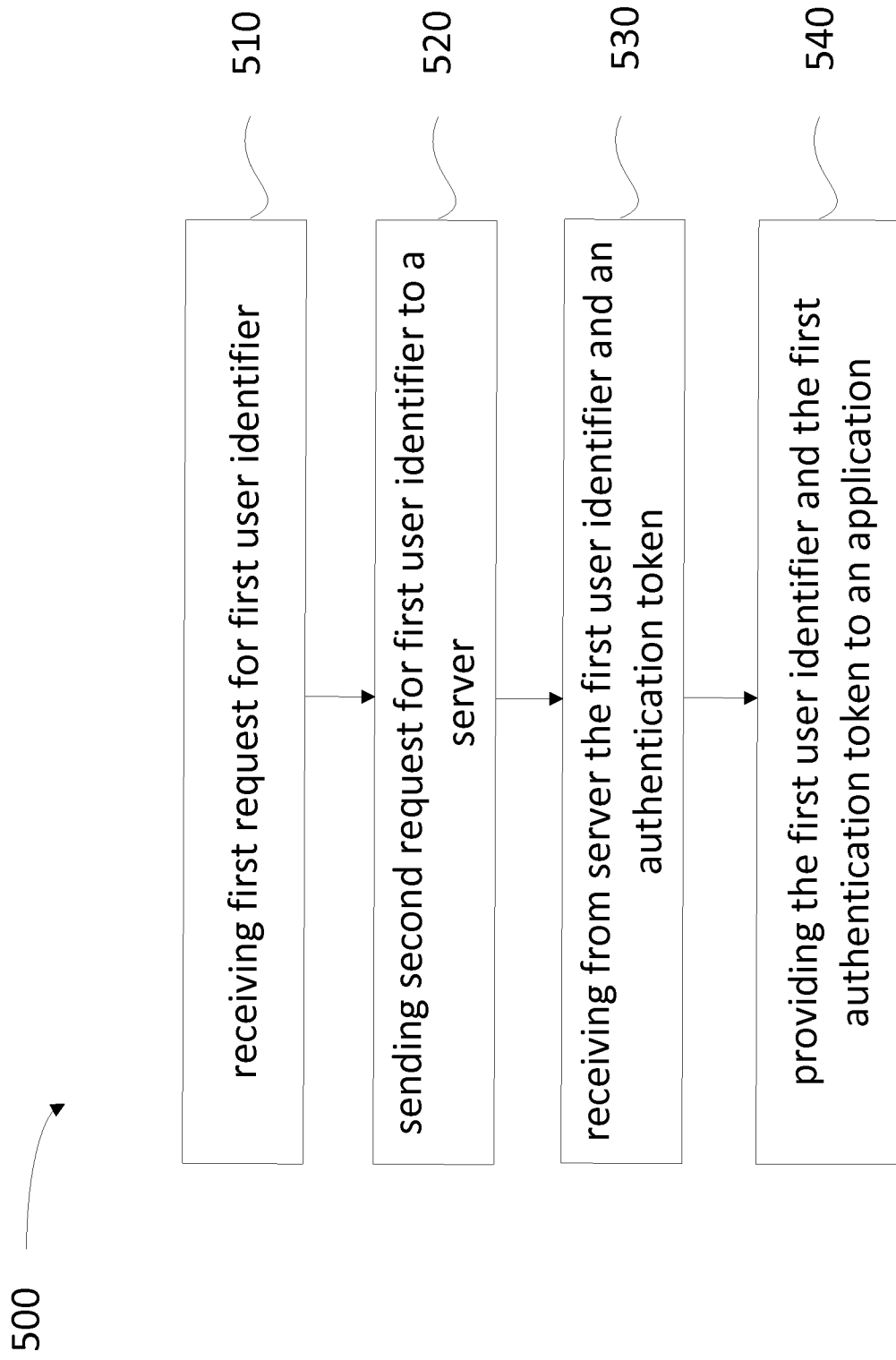


FIG. 5

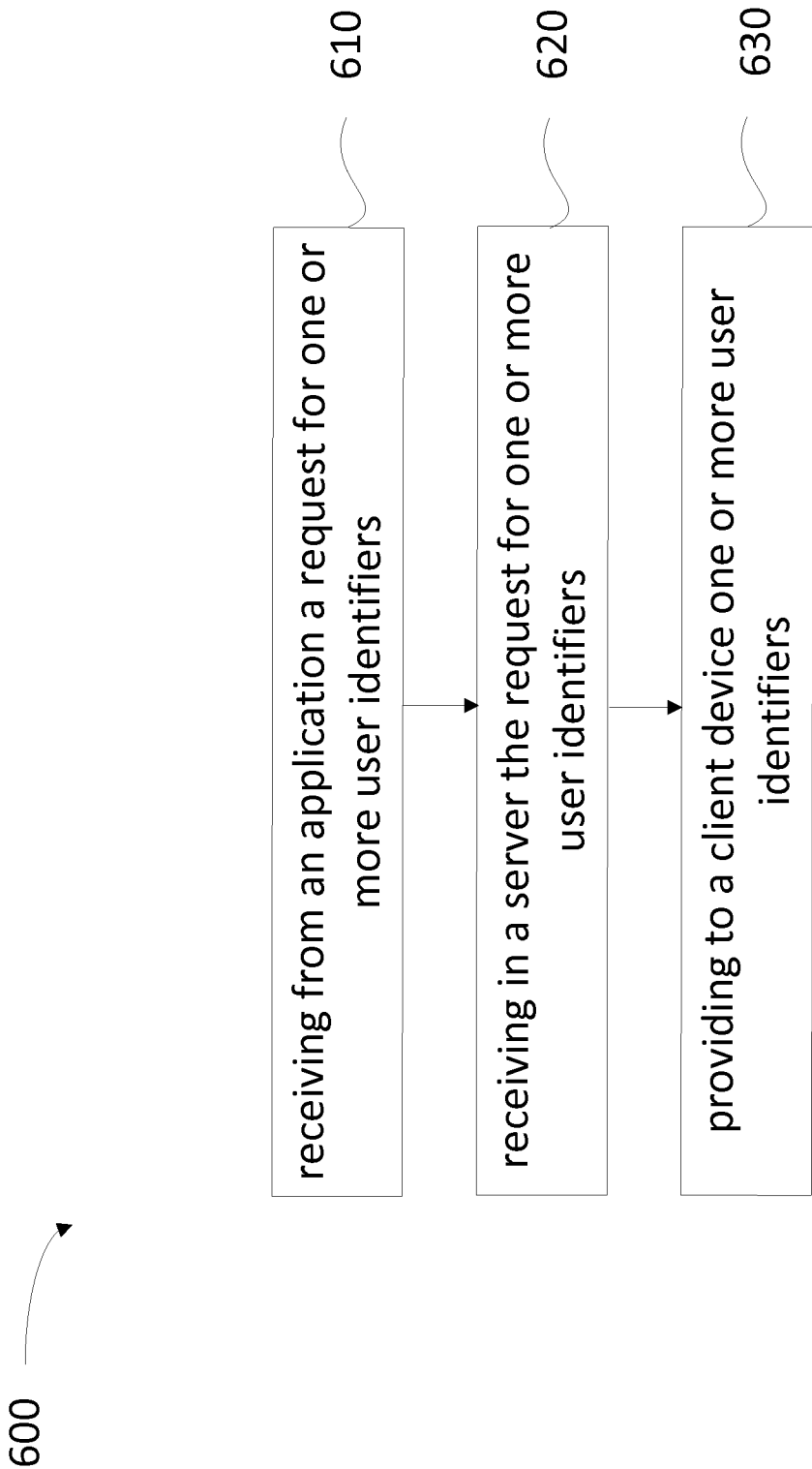


FIG. 6

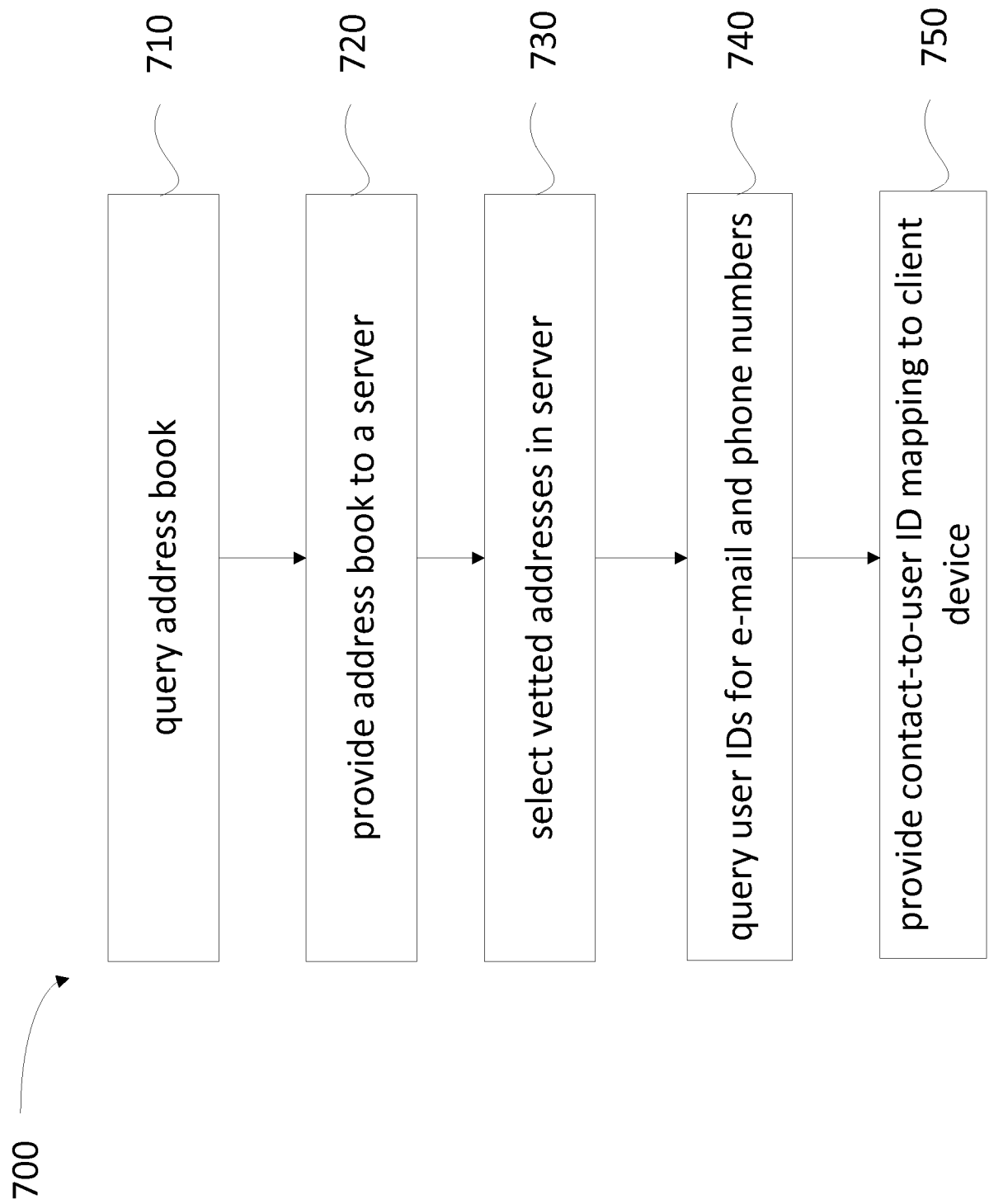


FIG. 7



FIG. 8

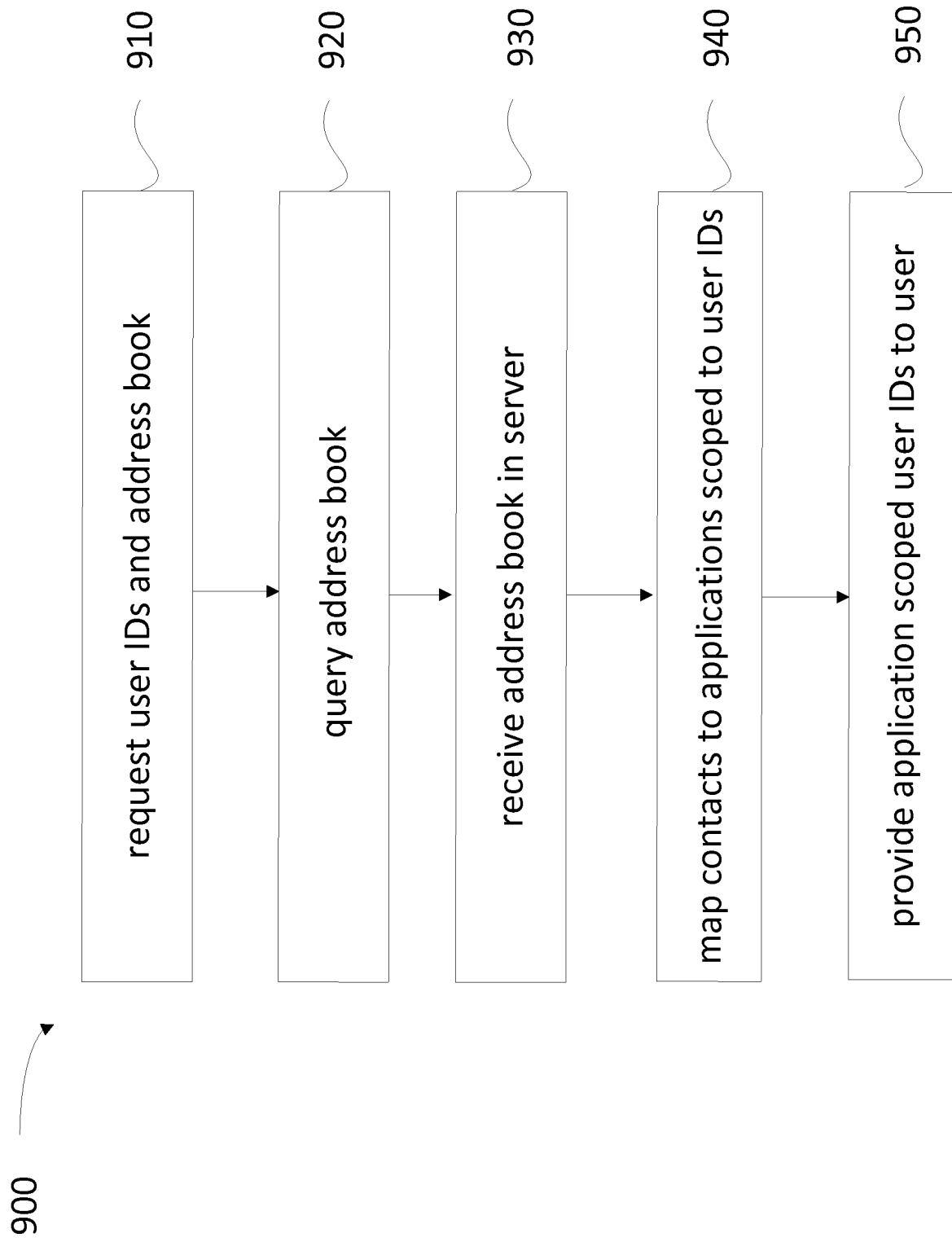


FIG. 9

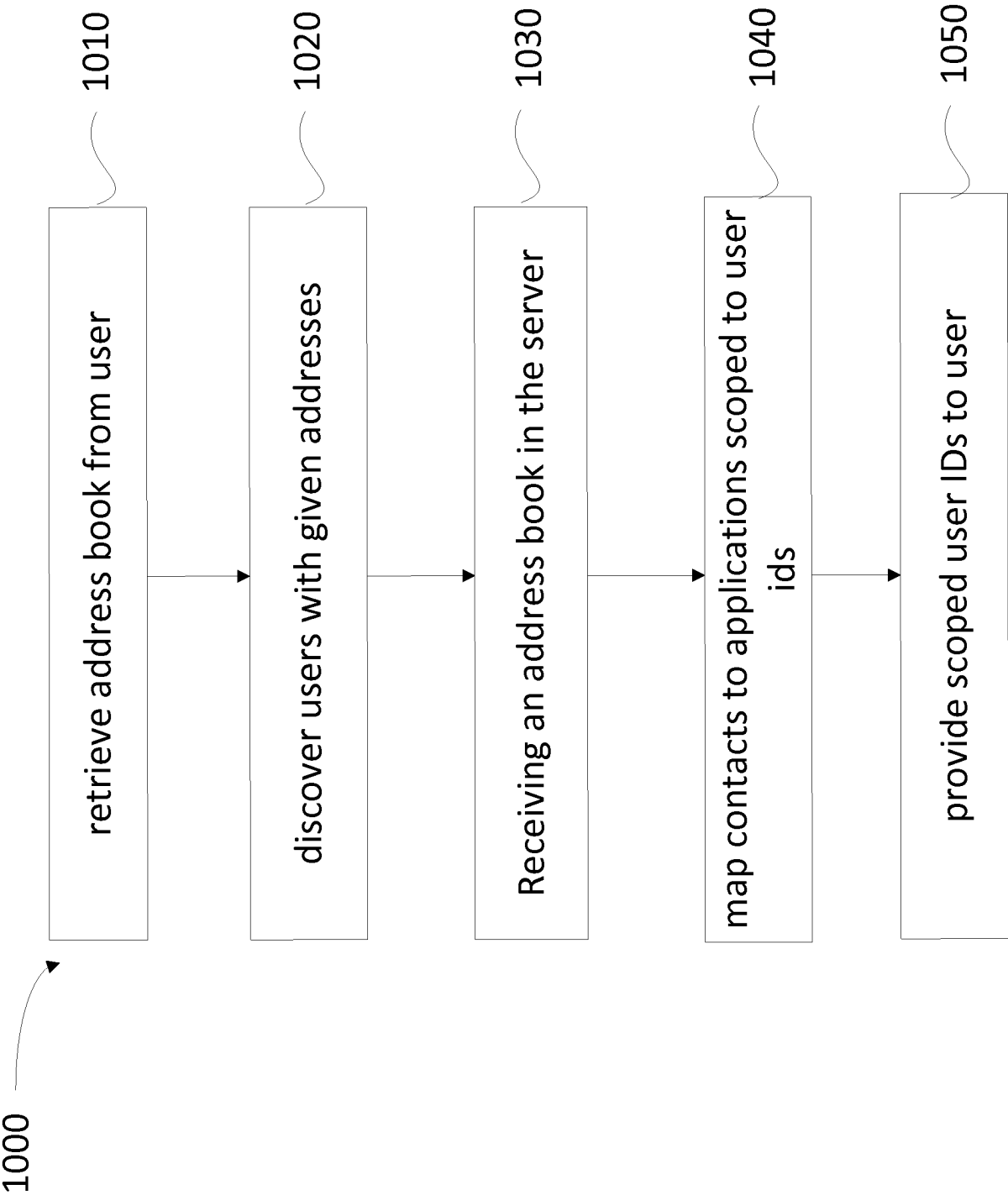


FIG. 10

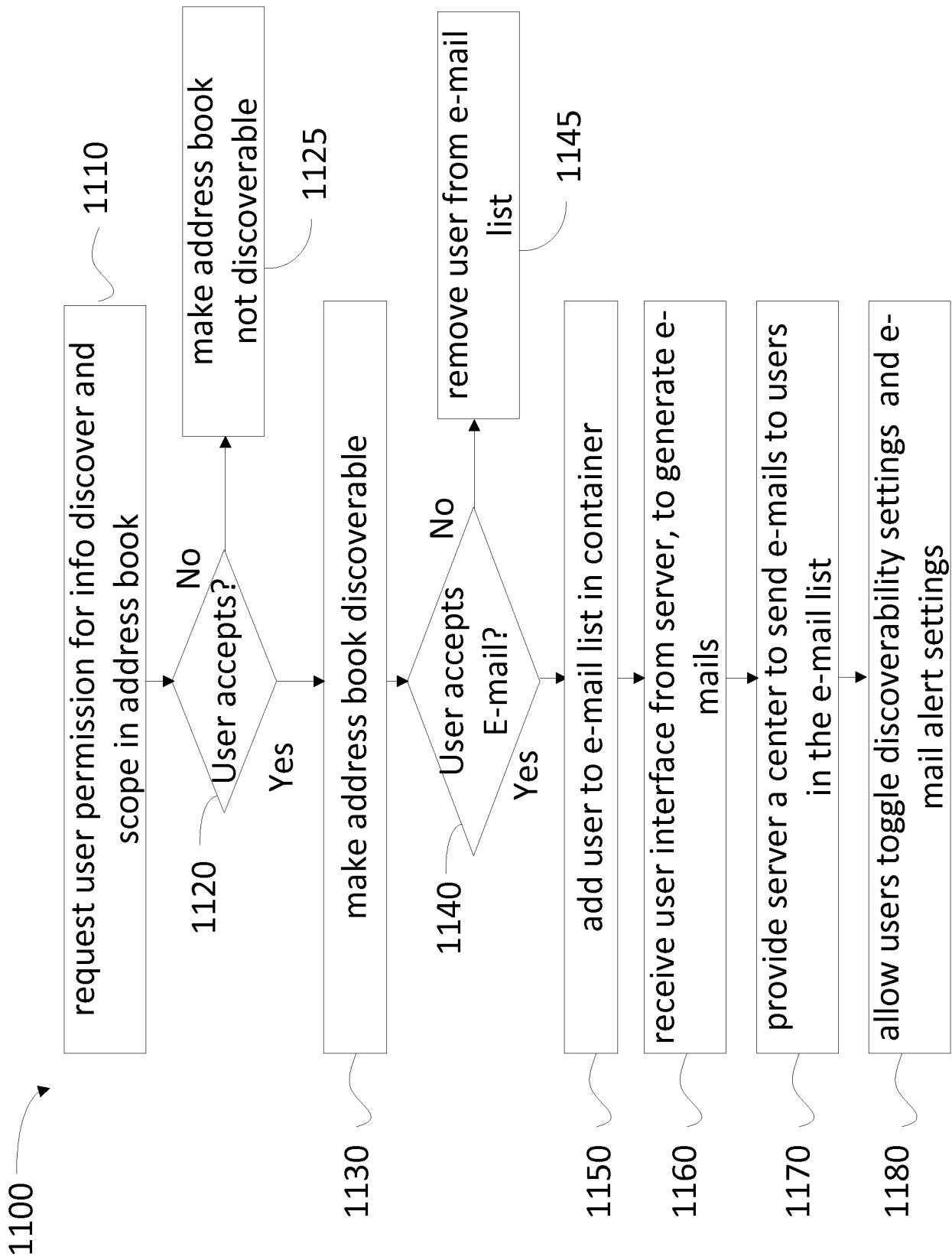


FIG. 11

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/035030**A. CLASSIFICATION OF SUBJECT MATTER****G06F 21/30(2013.01)i**

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F 21/30; G06F 15/16; G06F 17/60; H04L 29/06; H04L 9/00; G06F 21/00

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Korean utility models and applications for utility models

Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKOMPASS(KIPO internal) & Keywords: sigle sign-on, private information, second user identifier, reject, access, discoverable, address

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2005-0165698 A1 (KU GON CHO et al.) 28 July 2005 See paragraphs [0021]-[0024]; claim 1; and figure 2.	1-20
A	US 2004-0267870 A1 (JOHN ROZMUS et al.) 30 December 2004 See paragraphs [0017]-[0021]; claim 16; and figure 2.	1-20
A	US 2006-0155993 A1 (AXEL BUSBOON) 13 July 2006 See paragraphs [0077]-[0083]; and figure 3.	1-20
A	WO 01-72009 A2 (AT & T CORP.) 27 September 2001 See page 17, line 13 - page 18, line 21; and figure 2.	1-20
A	US 2012-0159588 A1 (BIN BENJAMIN ZHU et al.) 21 June 2012 See paragraphs [0022]-[0027]; claim 1; and figure 2.	1-20



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

17 September 2014 (17.09.2014)

Date of mailing of the international search report

17 September 2014 (17.09.2014)

Name and mailing address of the ISA/KR

International Application Division
Korean Intellectual Property Office
189 Cheongsu-ro, Seo-gu, Daejeon Metropolitan City, 302-701,
Republic of Korea

Facsimile No. +82-42-472-7140

Authorized officer

AHN, Jeong Hwan

Telephone No. +82-42-481-8440



INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2014/035030

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2005-0165698 A1	28/07/2005	AU 2003-235308 A1 EP 1512080 A1 JP 04413774 B2 JP 2005-527909 A KR 10-0464755 B1 KR 10-2003-0091237 A US 8056122 B2 WO 03-100629 A1	12/12/2003 09/03/2005 10/02/2010 15/09/2005 06/01/2005 03/12/2003 08/11/2011 04/12/2003
US 2004-0267870 A1	30/12/2004	None	
US 2006-0155993 A1	13/07/2006	AU 2003-212261 A1 EP 1595190 A1 EP 1595190 B1 WO 2004-075035 A1	09/09/2004 16/11/2005 27/09/2006 02/09/2004
WO 01-72009 A2	27/09/2001	CA 2400623 A1 EP 1264463 A2	27/09/2001 11/12/2002
US 2012-0159588 A1	21/06/2012	US 2010-131755 A1 US 2012-144465 A1 US 2012-159589 A1 US 2012-174207 A1 US 8151333 B2	27/05/2010 07/06/2012 21/06/2012 05/07/2012 03/04/2012