PRESENCE MONITORING IN A SERVERLESS PEER-TO-PEER SYSTEM

Background

[0001] Server based communication services such as the Messenger service provided by MSN® communication services permit users to sign into a server-based network and then use the services of the network (e.g., e-mail, text messaging, etc.). A server may store a contact list for the user and the user can add and delete persons from the contact list. When the user signs in, a server or servers may notify persons in the user's contact list that the user is "online." Similarly, the server or servers may notify the user of persons in the user's contact list that are "online."

[0002] The MICROSOFT® Corporation also provides Peer-to-Peer Networking software for use with its WINDOWS® operating systems. With this system, users can create a network of peer computers and can communicate with one another without having to sign into a central server. For example, users can create a peer-to-peer group and then create a chat room in which all members of the group can post messages and see messages posted by others in the group. The chat room is maintained using the peer computers and without the need for a central server.

Summary

[0003] Systems and methods are described for facilitiating collaboration and/or communication in a peer-to-peer serverless system. The system may transmit to other computing systems associated with other entities information regarding presence information associated with a user entity. Also, the system may request of other computing systems associated with other entities information regarding presence information associated with the other entities. Presence information may generally indicate the willingness and/or ability of an entity to communicate and/or collaborate with other entities, for example.

Drawings

[0004] Fig. 1 is a block diagram of a computing system that may operate in accordance with the claims;

[0005] Fig. 2 is a block diagram of an example system that may facilitate peer-to-peer, serverless collaboration and/or communications;

[0006] Fig. 3 is a flow diagram of an example method related to monitoring presence information of an entity;

[0007] Fig. 4 is a flow diagram of an example method related to permitting an entity to monitor presence of a user;

[0008] Fig. 5 is a flow diagram of an example method related to monitoring presence information of one or more entities;

[0009] Fig. 6 is a flow diagram of an example method related to providing presence information to one or more entities;

[0010] Fig. 7 is a flow diagram of an example method related to providing presence information regarding an entity; and

[0011] Fig. 8 is a flow diagram of an example method related to providing presence information regarding a user to another entity.

Description

[0012] Although the following text sets forth a detailed description of numerous different embodiments, it should be understood that the legal scope of the description is defined by the words of the claims set forth at the end of this patent. The detailed description is to be construed as exemplary only and does not describe every possible embodiment since describing every possible embodiment would be impractical, if not impossible. Numerous alternative embodiments could be implemented, using either current technology or technology developed after the filing date of this patent, which would still fall within the scope of the claims.

[0013] It should also be understood that, unless a term is expressly defined in this patent using the sentence "As used herein, the term '_____' is hereby defined to mean..." or a similar sentence, there is no intent to limit the meaning of that term, either expressly or by implication, beyond its plain or ordinary meaning, and such term should not be interpreted to be limited in scope based on any statement made in any section of this patent (other than the language of the claims). To the extent that any term recited in the claims at the end of this patent is referred to in this patent in a manner consistent with a single meaning, that is done for sake of clarity only so as to not confuse the reader, and it is not intended that such claim term by limited, by implication or otherwise, to that single meaning. Finally, unless a claim element is defined by reciting the word "means" and a function without the recital of any structure, it is not intended that the scope of any claim element be interpreted based on the application of 35 U.S.C. § 112, sixth paragraph.

[0014] Fig. 1 illustrates an example of a suitable computing system environment 100 on which a system for the steps of the claimed method and apparatus may be implemented. The

computing system environment 100 is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the method of apparatus of the claims. Neither should the computing environment 100 be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment 100.

[0015] The steps of the claimed method and apparatus are operational with numerous other general purpose or special purpose computing system environments or configurations. Examples of well known computing systems, environments, and/or configurations that may be suitable for use with the methods or apparatus of the claims include, but are not limited to, personal computers, server computers, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or devices, and the like.

[0016] The steps of the claimed method and apparatus may be described in the general context of computer-executable instructions, such as program modules, being executed by a computer. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. The methods and apparatus may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote computer storage media including memory storage devices.

[0017] With reference to Fig. 1, an exemplary system for implementing the steps of the claimed method and apparatus includes a general purpose computing device in the form of a computer 110. Components of computer 110 may include, but are not limited to, a processing unit 120, a system memory 130, and a system bus 121 that couples various system components including the system memory to the processing unit 120. The system bus 121 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus also known as Mezzanine bus.

[0018] Computer 110 typically includes a variety of computer readable media. Computer readable media can be any available media that can be accessed by computer 110 and

includes both volatile and nonvolatile media, removable and non-removable media. By way of example, and not limitation, computer readable media may comprise computer storage media and communication media. Computer storage media includes both volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules or other data. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical disk storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can accessed by computer 110. Communication media typically embodies computer readable instructions, data structures, program modules or other data in a modulated data signal such as a carrier wave or other transport mechanism and includes any information delivery media. The term "modulated data signal" means a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media includes wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, RF, infrared and other wireless media. Combinations of the any of the above should also be included within the scope of computer readable media.

[0019] The system memory 130 includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) 131 and random access memory (RAM) 132. A basic input/output system 133 (BIOS), containing the basic routines that help to transfer information between elements within computer 110, such as during start-up, is typically stored in ROM 131. RAM 132 typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit 120. By way of example, and not limitation, Fig. 1 illustrates operating system 134, application programs 135, other program modules 136, and program data 137.

[0020] The computer 110 may also include other removable/non-removable, volatile/nonvolatile computer storage media. By way of example only, Fig. 1 illustrates a hard disk drive 140 that reads from or writes to non-removable, nonvolatile magnetic media, a magnetic disk drive 151 that reads from or writes to a removable, nonvolatile magnetic disk 152, and an optical disk drive 155 that reads from or writes to a removable, nonvolatile optical disk 156 such as a CD ROM or other optical media. Other removable/non-removable, volatile/nonvolatile computer storage media that can be used in the exemplary operating environment include, but are not limited to, magnetic tape cassettes, flash memory cards, digital versatile disks, digital video tape, solid state RAM, solid state ROM, and the like. The

hard disk drive 141 is typically connected to the system bus 121 through a non-removable memory interface such as interface 140, and magnetic disk drive 151 and optical disk drive 155 are typically connected to the system bus 121 by a removable memory interface, such as interface 150.

[0021] The drives and their associated computer storage media discussed above and illustrated in Fig. 1, provide storage of computer readable instructions, data structures, program modules and other data for the computer 110. In Fig. 1, for example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146, and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer 20 through input devices such as a keyboard 162 and pointing device 161, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the system bus 121 via an interface, such as a video interface 190. In addition to the monitor, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 190.

[0022] The computer 110 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 180. The remote computer 180 may be a personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer 110, although only a memory storage device 181 has been illustrated in Fig. 1. The logical connections depicted in Fig. 1 include a local area network (LAN) 171 and a wide area network (WAN) 173, but may also include other networks. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

[0023] When used in a LAN networking environment, the computer 110 is connected to the LAN 171 through a network interface or adapter 170. When used in a WAN networking

environment, the computer 110 typically includes a modem 172 or other means for establishing communications over the WAN 173, such as the Internet. The modem 172, which may be internal or external, may be connected to the system bus 121 via the user input interface 160, or other appropriate mechanism. In a networked environment, program modules depicted relative to the computer 110, or portions thereof, may be stored in the remote memory storage device. By way of example, and not limitation, Fig. 1 illustrates remote application programs 185 as residing on memory device 181. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

[0024] Fig. 2 is a block diagram of an example system 200 that may be used to implement example methods described herein. The system 200 may facilitate peer-to-peer, serverless collaboration and/or communications via a communication network 202, and may be implemented using a computing system such as the computing system 100 of Figure 1. The communication network 202 may comprise a LAN and/or a WAN, for example. In some implementations, a communication network 202 may be omitted and communications with other computing systems may occur in a point-to-point manner, for example.

[0025] The system 200 may include a presence system 204 that monitors the presence of other entities on the communication network 202. An entity may be, for example, a particular person, a position in an organization (e.g., "manager," "customer service representative," etc.), an organization, a device (e.g., a printer, a copier, a scanner, a computer, etc.), etc. Presence may generally refer to a current status of an entity with regard to their willingness or ability to communicate with other entities, but may also refer to additional or alternative information regarding the entity such as a current activity of the entity. Presence of an entity may be represented by presence information. Examples of presence information may include one or more of an indication that an entity is "online," an indication that an entity is "offline," an indication that an entity is "out to lunch," an indication that an entity is "away," an indication that an entity will "be right back," an indication that an entity is "idle," an indication that an entity is "busy," an indication that an entity is "on the phone," an indication that an entity is "watching a movie," an indication that an entity is "playing Halo®," an indication that an entity is "helping another customer," etc. The indications described above could comprise identifiers associated with presence states (e.g., number 7 indicates presence is "online"), one or more strings (e.g., the string "online"), etc. Also, presence information may be selectable from a set of allowable presence states and/or a user entity may be able to define custom presence states that can be represented, for example, by a string. For example, a user entity could define a custom presence state as "I'm

out of the office. Will be back tomorrow." Presence information obtained by the presence system 204 may be stored in a presence store 208.

[0026] The presence system 204 may facilitate a user entity to monitor (or "subscribe") to presence information of other entities. This may comprise the presence system 204 polling other computing systems periodically, for example. Additionally or alternatively, other computing systems corresponding to other user entities may transmit event indications to the system 200 that notify the presence system 204 of events such as a change in presence state. For example, an event may occur when a user's presence changes from "offline" to "online," and the presence system 204 may detect this event. The presence system 204 could then notify other applications or software modules (e.g., such as the application 280) that the event occurred.

[0027] The presence system 204 may also monitor capabilities of other entities published on the network. Capabilities of an entity may include, for example, static capabilities such as whether a computing system of the entity is configured to execute a particular software application, whether a computing system of the entity has a particular hardware device, etc. Capabilities of an entity may also include, for example, dynamic capabilities such as real-time capabilities of an entity with respect to a game software application currently being executed on the entity's computing system, etc. An entity publishing capabilities on the network may refer to permitting other entities to be able to monitor the capabilities via the network. Capability information obtained by the presence system 204 may be stored in a capability store 212.

[0028] The presence system 204 may also monitor objects of other entities published on the network 202. Objects of an entity may include, for example, data objects such as files, structures, pictures, sounds, a description such as meta-data, a name-value pair, etc. An entity publishing objects on the network may refer to permitting other entities to be able to monitor the objects via the network. As just one example, publishing an object may permit an entity to provide other entities with information specific to an application being executed by a computing system of the entity and/or real-time information. With respect to a game application, for instance, a published object could include information regarding a player's current score, a weapon currently in possession of the player, etc. Objects information obtained by the presence system 204 may be stored in an objects store 216.

[0029] The presence system 204 may also provide (or "publish") presence information associated with a user entity (i.e., the entity associated with the system 200) to other entities on the network 202. The presence information associated with the user entity may be stored

in the presence store 208 or some other storage. Similarly, the presence system 204 may also provide (or "publish") information regarding capabilities of the user entity to other entities on the network 202. The capability information associated with the user entity may be stored in the capability store 208 or some other storage. Further, the presence system 204 may also provide (or "publish") information regarding objects of the user entity to other entities on the network 202. The object information associated with the user entity may be stored in the objects store 216 or some other storage.

[0030] The presence system 204 may interface with a contact store 240 that stores information regarding other entities. The contact store 240 may store information for an entity such as one or more of a secure identifier, a human readable alias, an indicator of whether presence information for this entity is to be monitored, and an indicator of whether to allow this entity to obtain presence information regarding the user entity. An entity as represented in the contact store 240 may be referred to as a contact.

[0031] The presence system 204 may access the contact store 240 via a contact manager 250. The contact manager 250 may provide a set of application programming interfaces (APIs) that permit the presence system 204 to retrieve information from the contact store 240 and to optionally modify the contact store 240. For example, the contact manager 250 may provide APIs that permit adding a contact, updating contact information, deleting a contact, getting contact information, getting an enumeration of contacts stored in the contact store, etc.

[0032] Each entity may have one or more communication endpoints with which it is associated. Generally, different communication endpoints associated with an entity may comprise different communication termination points associated with the entity, such as different computing systems. As an example, endpoints for a particular entity may comprise a desktop computer at work, a desktop computer at home, a personal digital assistant (PDA), etc. Optionally, different communication endpoints associated with an entity may also comprise different software applications being executed by a single computing system.

[0033] The presence system 204 may also interface with a communication module 260, which is coupled to the communication network 202. The communication module 260 may establish connections between the system 200 and other peer computing systems associated with other entities. Establishing a connection may comprise, for example, one or more of determining an endpoint associated with an entity, resolving an address of the endpoint, authenticating communications, encrypting and decrypting communications, etc. In one implementation, the communication module 260 may interface with an authentication system

270 that is itself coupled to the contact store 240. In attempting to establish a connection with another computing system, the communication module 260 may receive from the other computing system an indication of an identifier associated with an entity. The authentication system 270 may then check whether information associated with a user entity corresponding to the identifier is stored in the contact store 240. As just one example, the authentication system 270 could check whether the identifier is stored in the contact store 240. If information associated with the user entity corresponding to the identifier is not found in the contact store 240, the connection may be refused.

[0034] A connection may be secured. Establishing a connection and communicating over a connection may comprise, for example, one or more of utilizing a secure channel, utilizing secure socket layer (SSL) techniques, utilizing transport layer security (TLS) techniques, utilizing public/private key pairs, utilizing authentication techniques (e.g., X.509 certificates, encrypted signatures utilizing a pretty good privacy (PGP) program, etc.), utilizing a peer name resolution protocol (PNRP), transmission control protocol (TCP), internet protocol (IP), internet protocol version six (IPv6), etc. Resolving an address of an endpoint may comprise, for example, resolving a PNRP identifier to an IP address and a port.

[0035] A software application 280 or some other software module may communicate with the presence system 204 to obtain presence information, capabilities information, and/or objects information associated with other user entities on the communication network 202. For example, the presence system 204 may provide a set of APIs that permit software applications and other software modules to request and receive information regarding presence, capabilities, and/or objects associated with other user entities. The presence system 204 may retrieve the requested information from the presence store 208, capabilities store 212, and/or the objects store 216. Additionally or alternatively, the presence system 204 could obtain requested information from the other user entities via the communication module 260 and the communication network 202.

[0036] Similarly, the software application 280 or some other software module may communicate with the contact manager 250 to modify the contact store 240 and/or get information from the contact store 240. The software application 280 or some other software module may utilize APIs provided by the contact manager 250 to modify the contact store 240 and/or get information from the contact store 240. Some of the blocks in Fig. 2 may communicate with other blocks using remote procedure call (RPC) techniques, although other techniques for inter-process communication can be used as well.

Contact Store

[0037] As discussed above, the contact store 240 may comprise a store of information regarding other entities or contacts. Some of the information stored in the contact store may comprise information that can be used to authenticate information received from others. For example, a contact may provide a user with an encrypted version of a unique identifier for the contact (e.g., an X.509 certificate, a digital signature encrypted using PGP, etc.). The encrypted version of the unique identifier may be stored in the contact store. In such an implementation, at least some information in the contact store may be retrieved, updated, deleted, etc., via cryptographic application programming interfaces (crypto APIs), for example. Also, in such an implementation, the contact store 240 may be access control list (ACL) protected such that only the user can read from or write to the contact store 240, for example. Optionally, others could be given access rights as well such as an administrator, a supervisor, someone the user permits, etc.

[0038] In one implementation, the contact store 240 may include, for each contact, a secure unique identifier, a human readable alias for the contact, an indicator of whether the presence of the contact is to be monitored, and an indicator of whether the contact is authorized to monitor the presence of the user. The unique identifier may be secured via a digital signature such as, for example, an X.509 certificate, or the like. The X.509 certificate may be a third-party certificate or, optionally, a self-signed certificate. The unique identifier may be stored in the contact store 240 by storing the X.509 certificate, for example. The unique identifier may comprise any of a variety of identifiers that may permit the contact to be located on a network. For example, the unique identifier may comprise a peer name resolution protocol (PNRP) identifier, an internet protocol (IP) address, etc. In one implementation, the unique identifier cannot be edited by a user in order to maintain security. The alias may comprise, for example, a human recognizable string such as "John Smith," or "Mom." In some implementations, a user may be able to modify the alias if desired.

[0039] The indicator of whether the presence of the contact is to be monitored may comprise, for example, a Boolean variable that may be set by the user to "TRUE" or "FALSE." Alternatively, the indicator of whether the presence of the contact is to be monitored may comprise a variable that can take on a larger range of values such as a value indicating that the presence of the contact is to be monitored during business hours, the presence of the contact is to be monitored depending upon some other variable, etc. The indicator of whether the presence of the contact is to be monitored may comprise a variable that can assume various values indicating, for example, that the contact has not yet been

granted permission to monitor the user's presence, the contact is not allowed to monitor the user's presence, or the contact is allowed to monitor the user's presence. The variable may be allowed assume other values additionally or alternatively such as values indicating that the contact may be able to monitor the user's presence during business hours, the contact may be able to monitor the user's presence depending upon some other variable, etc.

[0040] Other information associated with a contact may be stored in the contact store 240 as well such as a mailing address, an e-mail address, a telephone number, etc. Also, classification information for the contact may be stored such as whether the contact is a personal contact, a business contact, a family contact, a friend contact, etc.

[0041] The indicator of whether a contact is authorized to monitor the presence of the contact may also comprise, for example, a variable that indicates a category of which the contact is a member as well, as a variable that indicates whether members of the category are authorized to monitor the presence of the contact. Similarly, the indicator of whether the presence of the contact is to be monitored may comprise, for example, a variable that indicates a category of which the contact is a member, as well as a variable that indicates whether the presence of members of the category are to be monitored.

[0042] Optionally, the contact store could also include indicators of whether contacts are authorized to monitor capabilities and/or objects of the user, as well as indicators of whether capabilities and/or objects of contacts are to be monitored. For example, a contact may have associated with it an indicator or indicators of whether the contact is authorized to monitor capabilities and/or objects of the user. Similarly, a contact may have associated with it an indicator or indicators of whether capabilities and/or objects of the contact should be monitored. Alternatively, determination of whether capabilities and/or objects of contacts should be monitored and/or if contacts are authorized to monitor capabilities and/or objects of the user may be based on the indicators associated with presence discussed above, for example.

[0043] The contact store 240 may be stored in a non-volatile memory (e.g., a hard disk, a magnetic disk, an optical disk, a FLASH memory, a memory stick, etc.) so that the contact information may persist when the computing system is shut down. Similarly, each of a plurality of computing systems of a user may store a version of the contact store 240. The contact stores on the plurality of computing systems could be synchronized using any of a variety of techniques, including known techniques, to help ensure that updates made to one version of the contact store 240 on one computing system may be propagated to, duplicated on, etc., to another version of the contact store 240 on another computing system.

Contact Manager and Contact Manager APIs

[0044] As discussed above, applications and software modules such as the presence system 204 may access the contact store 240 via the contact manager 250. Also as discussed above, the contact manager 250 may provide a set of APIs that permit applications and software modules to read or modify information in the contact store 240. Examples of such APIs will be discussed below. It will be understood by those of ordinary skill in the art that other APIs may be used additionally and/or alternatively, and that the APIs discussed herein may be modified. Some applications or software modules may be permitted to use only some APIs. For example, it may be desirable in a particular implementation to permit only some applications or software modules to modify information in the contact store 240, and to permit other applications or software modules to only retrieve information from the contact store 240. One or more of the following example functions and/or other similar functions may be made available to applications and/or software modules via one or more dynamic link libraries (DLLs). Alternatively, the functions may be provided using any other type of technique known to those of ordinary skill in the art.

[0045] One example is an "addcontactfromXML" function to permit adding a contact to the contact store 240 based on information in an extensible markup language (XML) format. This function may be used, for example, to import contact information from another computing system. In one implementation, the addcontactfromXML function may be passed XML data that includes an X.509 certificate and optionally other information. Then the XML data may be parsed and the X.509 certificate extracted. Next, the X.509 certificate may be parsed to extract the unique identifier of the contact and optionally other information such as an alias. Then, it may be determined if a contact having the unique identifier is already stored in the contact store 240. If a contact having the unique identifier is already stored in the contact store 240, the contact is not added an error notification may be returned. If a contact having the unique identifier is not already stored in the contact store 240, the X.509 certificate may be stored in the contact store. Default values, for example, may be stored for the indicator of whether the presence of the contact is to be monitored and the indicator of whether the presence of the contact is to be monitored. Other information in the XML data could be stored as well such as a mailing address, e-mail address, phone number, etc. In other similar functions, the contact information could be provided in a format other than XML.

[0046] Another example function is a "deletecontact" function to permit deleting a contact from the contact store 240. In one implementation, the deletecontact function may be passed

the unique identifier of the contact. Then, it may be determined if a contact having the unique identifier is stored in the contact store 240. If a contact having the unique identifier is not stored in the contact store 240, an error notification may be returned. If a contact having the unique identifier is stored in the contact store 240, the X.509 certificate associated with the contact may be deleted from the contact store.

[0047] Yet another example function is an "updatecontact" function to permit modifying information in the contact store 240 associated with a contact. For example, a function such as this function could be used to change the indicator of whether the contact's presence is to be monitored and the indicator of whether the contact may monitor the user's presence. Similarly, a function such as this function could be used to change contact information such as the contact's alias, a mailing address, an e-mail address, a telephone number, a categorization of the contact, etc. In one implementation, the updatecontact function may be passed a data structure that includes the unique identifier of the contact and information that is to be updated. Then, it may be determined if a contact having the unique identifier is stored in the contact store 240. In other implementations, the function may be passed information other than the unique identifier and the contact may be located in the contact store 240 using this information. If a contact having the unique identifier is not stored in the contact store 240, an error notification may be returned. If a contact having the unique identifier is stored in the contact store 240, the information passed in the data structure may be used to update the information in the contact store 240. In this implementation, the updatecontact function cannot be used to modify the unique identifier of a contact. In other implementations, it may be possible to modify the unique identifier.

[0048] Another example function is a "getcontact" function to permit retrieving information associated with a contact from the contact store 240. This function may be used, for example, to retrieve contact information from the contact store 240. In one implementation, the function is passed the unique identifier of the contact. Then, it may be determined if a contact having the unique identifier is stored in the contact store 240. In other implementations, the function may be passed information other than the unique identifier and the contact may be located in the contact store 240 using this information. If a contact having the unique identifier is not stored in the contact store 240, an error notification may be returned. If a contact having the unique identifier is stored in the contact store 240, some or all information associated with the contact may be returned in a data structure, for example. For instance, the data may be stored in the data structure, and the function may return a pointer to the data structure. In one implementation, if the unique identifier passed to the function is a value that indicates that the user's information is desired (e.g., the unique

identifier is a NULL value, the user's unique identifier, etc.), then the user's contact information (e.g., including a X.509 certificate) may be returned (i.e., a "Me" contact).

[0049] A further example function is a "getcontactXML" function to permit retrieving information associated with a contact from the contact store 240. This function may be used, for example, to export contact information to another computing system. In one implementation, the function is passed the unique identifier of the contact. Then, it may be determined if a contact having the unique identifier is stored in the contact store 240. In other implementations, the function may be passed information other than the unique identifier and the contact may be located in the contact store 240 using this information. If a contact having the unique identifier is not stored in the contact store 240, an error notification may be returned. If a contact having the unique identifier is stored in the contact store 240, some or all information associated with the contact may be stored in a string variable in an XML format. Then, the function may return the string variable or a pointer to the string variable, for example. In one implementation, if the unique identifier passed to the function is a value that indicates that the Me contact is desired (e.g., the unique identifier is a NULL value, the user's unique identifier, etc.), then the Me contact information (e.g., including a X.509 certificate) may be returned as XML formatted data.

[0050] Another example function is an "enumcontacts" function to permit obtaining an indication of the contacts stored in the contact store 240. In one implementation, when this function is called, a list of all the contacts in the contact store 240 is created. Then, an object is created that contains the list. Next, the function returns a handle to the object. This handle may then be used to retrieve the list of contacts in the contact store 240.

[0051] Yet another example function is a "getcontactfromXML" function to permit obtaining contact information from XML formatted data. This function may be used, for example, by an application or software module to display contact information received as XML formatted data prior to storing the contact information in the contact store. In one implementation, the getcontactfromXML function may be passed XML data that includes an X.509 certificate and optionally other information. Then the XML data may be parsed and the X.509 certificate extracted. Next, the X.509 certificate may be parsed to extract the unique identifier of the contact and optionally other information such as an alias. Then, the unique identifier and optionally some or all of the other information such as the alias may be stored as an object. Next, the function may return the object or a pointer to the object, for example. In other similar functions, the contact information could be provided in a format other than XML.

[0052] Other functions could be provided as well. For example, the contact manager 250 could provide a function to determine for which contacts presence is to be monitored. As another example a function could be provided to determine which contacts are allowed to monitor the presence of the user.

[0053] The contact manager 250 may notify other applications and software modules of changes related to the contact store 240. For instance, the contact manager 250 may notify one or more other applications and software modules when, for example, a new contact has been added to the contact store 240, when a contact has been deleted, when a contact has been updated, when a contact whose presence was being monitored was deleted or when an indicator of whether the presence of a contact should be monitored has been changed, when a contact who had been marked as being allowed to monitor the user's presence was deleted or when an indicator of whether the contact is allowed to monitor the presence of the user has been changed, etc. The contact manager 250 could, for example, send an indication that a particular type of event has occurred (e.g., a contact has been deleted) to multiple applications and/or software modules directly or indirectly. Then, the contact manager 250 could, for example, present more information regarding the event (e.g., the particular contact that was deleted) in an accessible location such that other applications and/or software modules that would like to obtain more information regarding the event can access the information. Alternatively, the contact manager 250 could send information to applications and/or software modules indicating that the event occurred and also providing the additional information regarding the event. For example, the contact manager 250 could send information to applications and/or software modules that had previously indicated that they would like to receive such information. One of ordinary skill in the art will recognize many other techniques in which the contact manager 250 can notify other applications and/or software modules regarding changes to the contact store 240.

[0054] Fig. 3 is a flow diagram of an example method 300 for determining if the presence of a contact in the contact store 240 should be monitored, and if so, getting presence information for the contact. The method 300 could be implemented by a system such as the system 200 of Fig. 2, for example, and will be described with reference to Fig. 2. At a block 304, the presence system 204 may request from the contact manager 250 information regarding a contact. For example, the presence system could utilize the "getcontact" function, or a similar technique. At a block 308, the contact manager 250 may retrieve the contact information for the specified contact from the contact store 240. The contact information may include an identifier of the contact that can be used to locate the contact on the network 202. The contact information may also include an indication of whether

presence of the contact is to be monitored. Then, the requested contact information is provided to the presence manager 204.

[0055] At a block 312, it may be determined if the presence of the contact is to be monitored. For example, the contact information retrieved at the block 308 may include an indicator of whether presence of the contact is to be monitored, and this indicator may be examined to determine if presence of the contact is to be monitored. As another example, the contact information retrieved at the block 308 may include an indication of a category of which the contact is a member. In some implementations, determining if presence is to be monitored may comprise determining if presence of contacts in the category indicated by the contact information is to be monitored. For example, a user may choose to monitor the presence of all contacts in a "Friends" category.

[0056] If it is determined that presence of the contact is to be monitored, the presence system 204 may attempt to get presence information for the contact at a block 316. If it is determined that presence of the contact is not to be monitored, the flow may end.

[0057] Fig. 4 is a flow diagram of an example method 350 for determining if a contact is authorized to monitor presence of a user entity, and if so, providing presence information to the contact. The method 350 could be implemented by a system such as the system 200 of Fig. 2, for example, and will be described with reference to Fig. 2. At a block 354, the presence system 204 may receive, via the network 202, a request from a contact to monitor the presence of the user entity. The request may include an authenticated identifier of the contact, for example. At a block 358, the presence system 204 may send a request to the contact manager 250 for information regarding a contact. For example, the presence system could utilize the "getcontact" function, or a similar technique.

[0058] At a block 362, the contact manager 250 may retrieve the contact information for the specified contact from the contact store 240. The contact information may include an indication of whether the contact is authorized to monitor presence of the user entity. Then, the requested contact information is provided to the presence manager 204.

[0059] At a block 366, it may be determined if the contact is authorized to monitor presence of the user entity. For example, the contact information provided at the block 362 may include an indicator of the contact is authorized to monitor presence of the user entity, and this indicator may be examined to determine if the contact is authorized to monitor presence of the user entity. As another example, the contact information retrieved at the block 362 may include an indication of a category of which the contact is a member. In some implementations, determining if the contact is authorized to monitor presence of the user

entity may comprise determining if contacts in the category indicated by the contact information authorized to monitor presence of the user entity. For example, a user may choose to authorize contacts in a "Friends" category to monitor presence of the user.

[0060] If it is determined that the contact is authorized to monitor presence of the user entity, the presence system 204 may attempt to send presence information to the contact via the network 202 at a block 370. If it is determined that the contact is not authorized to monitor presence of the user entity, a denial may be sent to the contact via the network 202 at a block 374.

Presence System

[0061] As discussed above, the presence system 204 may monitor the presence of other entities on the communication network 202 and may publish the presence of the user to other entities. Additionally, the presence system 204 may monitor capabilities and/or objects of other entities, and may publish capabilities and/or objects of the user. Also as described above, the presence system 204 may store presence information, capability information, and/or objects information regarding the user entity and other entities in the presence store 208, the capabilities store 212, and the objects store 216, respectively.

[0062] In some implementations, an entity may have multiple endpoints associated with it (e.g., a computer at home, a computer at work, a PDA, etc.). In these implementations, the presence system 204 may comprise an endpoint manager that may determine one or more endpoints associated with a contact. For each endpoint of a contact, the presence system 204 may store information such as an address and/or a port number, for example, to enable establishing a connection and communication with the endpoint and, optionally, a human readable name of the endpoint (e.g., "Home," "Work," "PDA," etc.). If an address and/or a port number, for example, of an endpoint changes, the presence system 204 may detect such a change and update its information regarding the contact.

[0063] The presence system 204 may provide APIs that permit applications and software modules to read or modify information associated with endpoints. One example function is an "enumendpoints" function to permit obtaining information regarding the endpoints of a contact. In one implementation, the enumendpoints function may be passed an indication of the contact (e.g., a unique identifier) for which endpoint information is desired. Then the endpoints associated with the contact may be assembled into an array, for example. Next, the function may return a pointer or handle to the array, for example.

[0064] Another example function is a "getendpointname" function to return a human readable name of the endpoint associated with the computing system on which the presence

system is being implemented. For example, the function may return the human readable name, a pointer to the human readable name, or some other indicator of the endpoint. As discussed above, a particular computing system may have multiple endpoints associated with it. For example, separate applications running on the computing system may be considered separate endpoints. As another example, two or more users could use the same computing system, thus the computing system may have multiple endpoints associated with it corresponding to different login accounts, for example. Thus, in some implementations, the getendpointname function optionally may be passed an argument that indicates the particular endpoint for which a name is requested.

[0065] Yet another example function is a "setendpointname" function to set a human readable name for an endpoint. The function may be passed a variable that includes text of the desired name to be assigned to the endpoint. In implementations in which a computing system may have multiple endpoints with which it is associated, the function optionally may be passed an indicator of the endpoint.

[0066] The presence system 204 may maintain a list of contacts and/or endpoints that the presence system 204 is monitoring for presence information. The presence system 204 may provide an API to permit applications and software modules to get information regarding the contacts and/or endpoints that the presence system 204 is monitoring. As an example, a "getaddresses" function may enable applications and software modules to obtain the list of contacts and/or endpoints maintained by the presence system 204. The function may return an array containing a list of contacts or endpoints, a pointer to the array, a handle, etc., and may also return a number of endpoints/contacts in the array.

[0067] Another example function is a "getpresenceinfo" function to get presence information for a contact. In one implementation, the getpresenceinfo function may be passed an indication of the contact (e.g., a unique identifier) for which presence information is to be retrieved. Optionally, the function may also be passed an indication of an endpoint of the contact for which presence information is to be retrieved. In response, the presence system 204 may attempt to retrieve presence information regarding the contact. For example, the presence system 204 could utilize the communication module 260 to establish a connection with one or more computing systems associated with the contact, and then retrieve presence information from the computing system(s). If presence at a particular endpoint is desired, the presence system 204 could utilize the communication module 260 to establish a connection with a computing system associated with the endpoint, and then obtain presence information from the computing system. As another example, the presence system

204 could first attempt to retrieve presence information from the presence store 208. If presence information for the contact is in the presence store 208, the presence system 204 could return that presence information rather than retrieving presence information from a computing system (or systems) associated with the contact. If the presence system 204 is unable to obtain presence information for the contact (e.g., cannot obtain presence information from a computing system of the contact), the getpresenceinfo function may return an indication that the presence system 204 was unable to obtain presence information for the contact.

[0068] In implementations in which the presence system 204 may determine respective presence information at a plurality of endpoints associated with a contact, the presence system 204 may generate aggregated presence information for the contact. For example, if the presence system 204 determines presence for one endpoint of a contact is "online" and that presence for the remaining endpoints of the contact is "offline," the presence system 204 may determine the aggregated presence information for the contact is "online." After receiving the aggregated presence of the contact, an application or software module could then utilize a function such as the getpresenceinfo function to determine presence at particular endpoints of the contact. Generate aggregated presence information for a contact based on presence information corresponding to a plurality of endpoints of the contact could implemented using any of a variety of techniques. As just one example, presence states could be prioritized, and the aggregated presence information could be set to the presence state having the highest priority in the plurality of presence states corresponding to endpoints of the contact.

[0069] Still another example function is an "enumcapabilities" function to get capability information for a contact. In one implementation, the enumcapabilities function may be passed an indication of the contact (e.g., a unique identifier) for which capability information is to be retrieved. Optionally, the function may also be passed an indication of an endpoint of the contact for which capability information is to be retrieved. In response, the presence system 204 may attempt to retrieve capability information regarding the contact. For example, the presence system 204 could utilize the communication module 260 to establish a connection with one or more computing systems associated with the contact, and then retrieve capability information from the computing system(s). If capabilities at a particular endpoint are desired, the presence system 204 could utilize the communication module 260 to establish a connection with a computing system associated with the endpoint, and then obtain capability information from the computing system. As another example, the presence system 204 could first attempt to retrieve capability information from the capabilities store 212. If

capability information for the contact is in the capabilities store 212, the presence system 204 could return that capability information rather than retrieving capability information from a computing system (or systems) associated with the contact. If the presence system 204 is unable to obtain capability information for the contact (e.g., cannot obtain capability information from a computing system of the contact), the enumcapabilities function may return an indication that the presence system 204 was unable to obtain capability information for the contact. If the presence system 204 is able to obtain capability information for the contact, the enumcapabilities function may return an array that lists the capabilities, a pointer to the array, a handle to the array, etc.

[0070] Capabilities of a contact could be identified using a variety of techniques. In one implementation, capabilities could be identified using a unique identifier, such as a globally unique identifier (GUID). In this implementation, the enumcapabilities function could return a list of GUIDs corresponding to the capabilities of a contact. Additionally or alternatively, other information associated with capabilities could be stored in the capabilities store 212 (e.g., a descriptive name, version identifier, etc.). The enumcapabilities function could return some or all (or none) of this other information as well.

[0071] In one implementation, the enumcapabilities function may also be passed an indication of a particular capability. For example, the function could be passed a GUID corresponding to the capability. In this implementation, the enumcapabilities function may return an indication of whether the contact has the specified capability.

[0072] Yet another example function is an "enumobjects" function to get capability information for a contact. In one implementation, the enumobjects function may be passed an indication of the contact (e.g., a unique identifier) for which object information is to be retrieved. Optionally, the function may also be passed an indication of an endpoint of the contact for which object information is to be retrieved. In response, the presence system 204 may attempt to retrieve object information regarding the contact. For example, the presence system 204 could utilize the communication module 260 to establish a connection with one or more computing systems associated with the contact, and then retrieve object information from the computing system(s). If objects at a particular endpoint are desired, the presence system 204 could utilize the communication module 260 to establish a connection with a computing system associated with the endpoint, and then obtain object information from the computing system. As another example, the presence system 204 could first attempt to retrieve object information from the objects store 216. If object information for the contact is in the objects store 212, the presence system 204 could return that object information rather

than retrieving object information from a computing system (or systems) associated with the contact. If the presence system 204 is unable to obtain object information for the contact (e.g., cannot obtain object information from a computing system of the contact), the enumobjects function may return an indication that the presence system 204 was unable to obtain object information for the contact. If the presence system 204 is able to obtain object information for the contact, the enumobjects function may return an array that lists the objects, a pointer to the array, a handle to the array, etc.

[0073] Objects of a contact could be identified using a variety of techniques. In one implementation, objects could be identified using a unique identifier, such as a GUID. In this implementation, the enumobjects function could return a list of GUIDs corresponding to the objects of a contact. Additionally or alternatively, other information associated with objects could be obtained from other user entities and/or stored in the objects store 216 (e.g., a descriptive name). The enumobjects function could return some or all (or none) of this other information as well.

[0074] In one implementation, the enumobjects function may also be passed an indication of a particular object. For example, the function could be passed a GUID, an object name, etc., corresponding to the object. In this implementation, the enumobjects function may return an indication of whether the contact has the specified object.

[0075] Another example function is a "setpresenceinfo" function to set the presence of the user. In implementations in which a contact may only have one endpoint with which it is associated, this function may be used to set the presence for the user entity. In implementations in which a contact may have multiple endpoints with which it is associated, this function may be used to set the presence for the user entity for a particular endpoint (e.g., the computing system on which the presence system 204 is being implemented). The function may be passed an indication of a presence value. In response, the presence system 204 may set the presence of the user entity or endpoint to that value. Afterwards, if the presence of the user entity or endpoint is published to other contacts, it will reflect the new presence value.

[0076] Another example function is a "setobject" function to publish an object of the user. In implementations in which a contact may only have one endpoint with which it is associated, this function may be used to publish an object associated with the user entity. In implementations in which a contact may have multiple endpoints with which it is associated, this function may be used to publish an object associated with a particular endpoint (e.g., the computing system on which the presence system 204 is being implemented). The function

may be passed an indication of the object (e.g., a GUID, an object name, etc.). In response, the presence system 204 may determine if an object corresponding to the indication passed with the function has already been published. For example, the presence system 204 may examine the objects store 216 for an indication of the object. If it has not been published, the presence system may store the object or an indication of the object in the objects store 216, and may publish the object to those contacts authorized to monitor the user entity, those authorized and who have requested object information of the user entity, etc. If it has been published, the presence system may store an updated version of the object or an indication of the updated object in the objects store 216, and may publish the updated object to those contacts authorized to monitor the user entity, those authorized and who have requested object information of the user entity, etc.

[0077] Yet another example function is a "deleteobject" function to stop publishing an object of the user. The function may be passed an indication of the object (e.g., a GUID). In response, the presence system 204 may delete the object or an indication of the object from the objects store 216. Afterwards, the presence system 204 will no longer publish the object to other contacts.

[0078] Similar functions to the setobject function and the deleteobject function may be used to publish and unpublish capabilities of a user entity and/or an endpoint of the user entity. With these functions, capabilities and/or indications of capabilities may be added, updated, and/or deleted from the capabilities store 212.

[0079] The presence system 204 may notify other applications and software modules of changes related to presence, capabilities, objects, publishing presence, and monitoring others' presence. For instance, the presence system 204 may notify one or more other applications and software modules when, for example, presence status of a contact/endpoint currently being monitored has changed, when a written description of an endpoint has changed (e.g., "Home PC" changed to "Xbox"), when presence information about a new endpoint is available, when presence information regarding an endpoint is no longer available, etc. Also, the presence system 204 may notify one or more other applications and software modules when, for example, an indicator of whether a contact's presence should be monitored has been changed, an indicator of whether a contact is authorized to monitor the user's presence has changed, a contact whose presence was being monitored was deleted from the contact store 240, a contact authorized to monitor the user's presence was deleted from the contact store, etc.

[0080] Additionally, the presence system 204 may notify one or more other applications and software modules when, for example, an object of the user or an object of a contact that is being monitored has been changed (e.g., added, deleted, or updated). Similarly, the presence system 204 may notify one or more other applications and software modules when, for example, a capability of the user or a capability of a contact that is being monitored has been changed (e.g., added or deleted. Further, the presence system 204 may notify one or more other applications and software modules when, for example, a contact has requested to monitor the presence of the user and such a request from the contact was not previously denied and the contact has not been marked as not authorized to monitor presence of the user in the contact store 240.

[0081] The presence system 204 and/or the contact manager 250 may notify other applications and software modules of changes related to the contact store 240. For instance, the presence system 204 and/or the contact manager 250 may notify one or more other applications and software modules when, for example, when information in the contact store 240 regarding a contact has been modified, when a contact has been added to the contact store 240, when a contact has been deleted from the contact store 240, etc..

[0082] The presence system 204 and/or the contact manager 250 could, for example, send an indication that a particular type of event has occurred (e.g., presence of a new contact is to be monitored) to multiple applications and/or software modules directly or indirectly. Then, the presence system 204 could, for example, present more information regarding the event (e.g., the particular contact whose presence is to be monitored) in an accessible location such that other applications and/or software modules that would like to obtain more information regarding the event can access the information. Alternatively, the presence system 204 could send information to applications and/or software modules indicating that the event occurred and also providing the additional information regarding the event. For example, the presence system 204 could send information to applications and/or software modules that had previously indicated that they would like to receive such information. If the contact manager 250 is to notify other applications and software modules of changes related to the contact store 240, it could utilize similar techniques. One of ordinary skill in the art will recognize many other techniques in which the presence system 204 and/or the contact manager 250 can notify other applications and/or software modules regarding events such as those described above.

[0083] Fig. 5 is a flow diagram of an example method 400 for retrieving presence information of one or more contacts. The method 400 could be implemented by a system

such as the system 200 of Fig. 2, for example, and will be described with reference to Fig. 2. At a block 404, entities for which presence information is desired are determined. Determining entities for which presence information is desired may comprise, for example, examining information in the contact store 240 via the contact manager 250, for example. Determining entities for which presence information is desired may also comprise, for example, examining a list of contacts and/or endpoints for which presence information is desired.

[0084] At a block 408, connections with computer systems associated with the entities determined at the block 404 may be established. For example, the presence system 204 could utilize the communication module 260 to establish connections with appropriate computing systems. Establishing connections may comprise determining one or more endpoints associated with an entity. Then, at a block 412, presence information may be requested from the computing systems with which connections were established. For example, the computing systems could be implementing systems the same as or similar to the system 200 of Fig. 2, and thus could provide requested presence information. At a block 416, the presence information requested at the block 412 may be received by the system 200. Then, the presence information may be stored in the presence store 208.

[0085] At least some of the blocks 404, 408, 412, 416, and 420 may be repeated periodically (e.g., ever 5 minutes or at a rate that is suitable for a particular implementation). In this way, a contact that went "offline" but did not announce it was doing so may be detected. Additionally or alternatively, at least some of the blocks 404, 408, 412, 416, and 420 may be repeated upon an occurrence of an event, such as the addition of a contact for which presence is to be monitored.

[0086] Fig. 6 is a flow diagram of an example method 450 for publishing presence information to one or more contacts. The method 450 could be implemented by a system such as the system 200 of Fig. 2, for example, and will be described with reference to Fig. 2. At a block 454, entities to which presence information is to be published are determined. Determining entities to which presence information is to be published may comprise, for example, examining information in the contact store 240 via the contact manager 250, for example. Determining entities to which presence information is to be published may also comprise, for example, examining a list of contacts and/or endpoints to which presence information is to be published.

[0087] At a block 458, connections with computer systems associated with the entities determined at the block 454 may be established. For example, the presence system 204 could

utilize the communication module 260 to establish connections with appropriate computing systems. Establishing connections may comprise determining one or more endpoints associated with an entity. Then, at a block 462, presence information may be sent to the computing systems with which connections were established.

[0088] At least some of the blocks 454, 458, and 462 may be repeated periodically (e.g., ever 5 minutes or at a rate that is suitable for a particular implementation). Additionally or alternatively, at least some of the blocks 454, 458, and 462 may be repeated upon an occurrence of an event, such as the addition of a contact to which presence is to be provided.

[0089] Referring now to Figs. 5 and 6, the method 400 may be used to monitor the presence of a first set of entities, and the method 500 may be used to publish presence information regarding the user to a second set of entities. As will be understood by those of ordinary skill in the art, the first set of entities may be different than the second set of entities because the system 200 permits a user to separately select the first and second sets. For instance, in the implementation described above, the user can separately select for each contact in the contact store 240 whether to authorize the contact to monitor the presence of the user and whether the presence of the contact should be monitored. As another example, the user could separately select whether a category of contacts is authorized to monitor the presence of the user, and whether presence information for the category of contacts should be monitored.

[0090] Fig. 7 is a flow diagram of an example method 500 for getting presence information of a contact in response to a request from an application or other software module. The method 500 could be implemented by a system such as the system 200 of Fig. 2, for example, and will be described with reference to Fig. 2. At a block 504, the presence system 200 may receive a request from an application or software module for presence information for a contact. At a block 508, it may be determined if presence information corresponding to the contact is in the presence store 208. If presence information corresponding to the contact is in the presence store 208, then the presence information may be retrieved from the presence store 208 at a block 512.

[0091] If presence information corresponding to the contact is not in the presence store 208, the flow may proceed to a block 516. At the block 516, a connection with one or more computing systems corresponding to the contact may be established. For example, the presence system 204 could utilize the communication module 260 to establish a connection with the computing system(s). Establishing connections may comprise determining one or more endpoints associated with the contact.

[0100] At a block 520, presence information may be requested from the computing systems with which connections were established. For example, the computing systems could be implementing systems the same as or similar to the system 200 of Fig. 2, and thus could provide requested presence information. At a block 524, the presence information requested at the block 520 may be received by the system 200. Optionally, the presence information may be stored in the presence store 208 at a block 528. Next, at a block 532, the presence information received at the block 524 is provided to the application or software module that requested the presence information at the block 504.

[0101] Optionally, the contact and/or endpoints associated with the contact may be added to a list of contacts and/or endpoints of which presence information is to be retrieved, for example. Then, a method the same as or similar to the method 400 of Fig. 5 could be used to monitor the presence of the contact.

[0102] Fig. 8 is a flow diagram of an example method 550 for publishing presence information to a contact in response to a request from the contact. The method 550 could be implemented by a system such as the system 200 of Fig. 2, for example, and will be described with reference to Fig. 2. At a block 554, a request for presence information may be received from a contact. For example, a computing system associated with a contact could implement a system the same as or similar to the system 200 of Fig. 2, and thus could send requests for presence information to a computing system of the user.

[0103] At a block 558, an identifier of the entity requesting the presence information may be determined. For example, an X.509 certificate or the like of the entity obtained while establishing a connection with the entity may be analyzed to determine a unique identifier of the entity. Optionally, an endpoint of the contact may also be determined. For example, the endpoint that issued the request received at the block 554 may be identified.

[0104] At a block 562, the identifier determined at the block 558 may be used to determine if the entity is authorized to receive presence information. For example, the identifier determined at the block 558 may be used to retrieve information in the contact store 240 regarding the contact. For instance, information in the contact store 240 could be obtained via the contact manager 250. As another example, a list of contacts and/or endpoints to which presence information is to be published could be examined to determine if the contact and/or an endpoint associated with the contact is in this list. It may be assumed, for instance, that contacts/endpoints in the list are authorized to receive presence information.

[0105] If it is determined that the contact/endpoint is not authorized to receive presence information, then at a block 566, the request received at the block 554 may be denied. On the

other hand, if it is determined that the contact/endpoint is authorized to receive presence information, then at a block 570, the presence information may be transmitted to the entity. Transmitting the presence information may comprise utilizing the communication module 260 to establish a connection or connections, if not currently established, with appropriate computing systems. Establishing connections may comprise determining one or more endpoints associated with an entity.

[0106] Optionally, the contact and/or endpoints associated with the contact may be added to a list of contacts and/or endpoints to which presence information is to be transmitted, for example. Then, a method the same as or similar to the method 450 of Fig. 6 could be used to publish to the contact the presence information.

[0107] With regard to capabilities, methods similar to the method 400 of Fig. 5 and the method 500 of Fig. 7 could be used to obtain capabilities of one or more contacts. Also, methods similar to the method 450 of Fig. 6 and the method 550 of Fig. 8 could be used to publish capabilities of the user to one or more contacts. With regard to objects, methods similar to the method 400 of Fig. 5 and the method 500 of Fig. 7 could be used to obtain objects of one or more contacts. Also, methods similar to the method 450 of Fig. 6 and the method 550 of Fig. 8 could be used to publish objects of the user to one or more contacts.

[0108] Referring again to Fig. 2, the system 200 or some other system may additionally interface with server-based systems to monitor others' presence, publish an entity's presence, monitor capabilities, etc., via the server-based system. For example, the contact store 240 could include contacts whose presence may be monitored via a server-based system, as well as peer-to-peer contacts. Similarly, the presence store 208, the capabilities store 212 and/or the objects store 216 could include information regarding contacts that was obtained via a server-based system.

[0109] Many modifications and variations may be made in the techniques and structures described and illustrated herein without departing from the spirit and scope of the present claims. Accordingly, it should be understood that the methods and apparatus described herein are illustrative only and are not limiting upon the scope of the claims.

CLAIMS:

1. A method for facilitating peer-to-peer collaboration in a serverless system, the method comprising:

transmitting, using a peer computing system associated with a user entity, presence information regarding the user entity to peer computing systems associated with a first set of other entities, the first set of other entities indicated by contact information in a contact store stored on the peer computing system associated with the user entity; and

transmitting, using the peer computing system associated with the user entity, requests for presence information associated with a second set of other entities indicated by contact information in the contact store, the requests being transmitted to peer computing systems associated with the second set of other entities indicated by contact information in the contact store;

wherein the first set of other entities is capable of being different than the second set of other entities.

2. A method according to claim 1, further comprising at least one of:

providing indications of the other entities in the second set of other entities;

providing presence information associated with a particular other entity;

providing capabilities information associated with a particular other entity;

providing objects information associated with a particular other entity; and

providing endpoint information associated with a particular other entity;

providing endpoint information associated with the user entity.

3. A method according to claim 2, wherein providing presence information associated with the particular other entity comprises at least one of:

providing presence information associated with one endpoint associated with the other entity; or

providing presence information associated with a plurality of endpoints associated with the other entity.

4. A method according to claim 2, wherein providing capabilities information associated with the particular other entity comprises at least one of:

providing capabilities information associated with one endpoint associated with the other entity; or

providing capabilities information associated with a plurality of endpoints associated with the other entity.

5. A method according to claim 2, wherein providing objects information associated with the particular other entity comprises at least one of:

providing objects information associated with one endpoint associated with the other entity; or

providing objects information associated with a plurality of endpoints associated with the other entity.

6. A method according to claim 1, further comprising:

changing the presence information regarding the user entity to a specified presence state; and

transmitting, using the peer computing system associated with the user entity, the changed presence information regarding the user entity to peer computing systems associated with the first set of other entities.

7. A method according to claim 1, wherein the peer computing system associated with the user entity is one endpoint of a plurality of endpoints associated with the user further comprising:

transmitting, using the peer computing system associated with the user entity, an endpoint name associated with the peer computing system associated with the user entity to at least some peer computing systems associated with at least some other entities in the first set of other entities.

8. A method according to claim 7, further comprising setting the endpoint name associated with the peer computing system associated with the user entity.

9. A method according to claim 1, further comprising transmitting, using the peer computing system associated with the user entity, an indication of capabilities associated with the user entity to peer computing systems associated with a third set of other entities.

10. A method according to claim 1, further comprising transmitting, using the peer computing system associated with the user entity, an indication of objects associated with the user entity to peer computing systems associated with a third set of other entities.

11. A method according to claim 10, further comprising determining the objects corresponding to which the indication of the objects is to be transmitted.

12. A method according to claim 10, wherein determining the objects corresponding to which the indication of the objects is to be transmitted comprises at least one of:

determining that an object has been added to the objects corresponding to which the indication of the objects is to be transmitted; or

determining that an object has been removed from the objects corresponding to which the indication of the objects is to be transmitted.

13. A method according to claim 10, wherein the second set of other entities is the same as the third set of other entities.

14. A method according to claim 1, further comprises at least one of:

generating an indication that a composition of the second set of other entities has changed;

generating an indication that presence of an other entity in the second set of entities has changed;

generating an indication that objects information regarding an other entity in the second set of entities has changed;

generating an indication that capabilities information regarding an other entity in the second set of entities has changed;

generating an indication that a name of an endpoint associated with an other entity in the second set of entities has changed; or

generating an indication that an other entity not in the first set of other entities has sent a request to monitor presence of the user entity.

15. A method for facilitating peer-to-peer collaboration and/or collaboration in a serverless system using a peer computing system associated with a user entity, the method comprising:

establishing a connection with a peer computing system associated with an other entity;

receiving a request from the peer computing system associated with the other entity to monitor presence of the user entity;

determining an identifier of the other user entity;

determining whether the identifier is in a contact store of the computing system associated with a user entity;

if the identifier is in the contact store, determining if an indicator in the contact store associated with the identifier of the other user entity indicates the other user entity may monitor the presence of the user entity; and

if the indicator associated with the identifier of the other user entity indicates the other user entity may monitor the presence of the user entity, transmitting, using the peer computing system associated with the user entity, presence information associated with the user entity to the peer computing system associated with the other entity.

16. A method according to claim 15, further comprising authenticating the identifier of the other user entity.

17. A method according to claim 16, wherein authenticating the identifier of the other user entity comprises utilizing at least one of a self-signed certificate or a third-party certificate.

18. A method according to claim 15, further comprising at least one of:

if the indicator associated with the identifier of the other user entity indicates the other user entity may monitor the presence of the user entity, transmitting, using the peer computing system associated with the user entity, capabilities information associated with the user entity to the peer computing system associated with the other entity; or

if the indicator associated with the identifier of the other user entity indicates the other user entity may monitor the presence of the user entity, transmitting, using the peer computing system associated with the user entity, objects information associated with the user entity to the peer computing system associated with the other entity.

19. A peer computing system comprising:

a contact store to store contact information, the contact store capable of indicating a first set of entities to which presence information regarding a user entity is to be provided and capable of indicating a second set of other entities from which presence information regarding the second set other entities is to be received;

a presence system coupled to the contact store, the presence system configured to transmit presence information associated with the user entity to peer computing systems associated with the first set of entities and configured to request from peer computing systems associated with the second set of entities presence information regarding the second set other entities.

20. A peer computing system according to claim 20, further comprising at least one of:

a presence store coupled to the presence system to store presence information regarding the second set other entities;

a capabilities store coupled to the presence system to store capabilties information regarding the second set other entities; or

an objects store coupled to the presence system to store objects information regarding the second set other entities.