

Remote Control Using Instant Messaging

Field of Invention

[0001] The present invention relates to remote control using instant messaging. In particular, the invention relates to a plug-in, which is operable at an automated instant messaging client for receiving and carrying out an instruction or instructions set, the operation including sending position data, upon request, to an IM user.

Background

[0002] An Instant Messaging ("IM") system is a client-server system. The IM server tracks the presence information of IM clients that an IM client has subscriptions to. The IM system thus links these IM clients together and allows the IM clients to exchange messages. FIG. 1 depicts typical IM Clients 11, 12, 13 connected to an IM Server 10 through an Internet Service Provider (ISP) 40 and internet 30 with other IM clients 14,15,16 through the same or another ISP 42. The IM Server 10 is connected to a Directory/Authentication Server 20 and the IM Server 10 runs a roster engine 22. The roster engine 22 manages transactions and associated roster subscription details of the IM clients connected to the IM Server 10. The roster subscription details of the IM clients are stored in a Repository 24. For example, as shown in FIG. 1, the IM Client 11 connects up with the IM Server 10 and notifies the IM Server 10 that it is connected. The IM Server 10 returns a list of contacts to IM Client 11 together with their presence information from the Repository 24. The contact list contains the contact details of those clients who have subscribed to both the IM Server 10 and the IM Client 11. The IM Client 11 then sends out its presence information to the IM Server 10 stating that it is "available". The IM Server 10 updates its Repository 24 and simultaneously sends the presence information of IM Client 11 to all clients that have subscribed to the IM Client 11. For example, as shown in a roster 26 in FIG. 1, IM Client 12 has IM Client 11 in its contact list (but not IM Client 13), the IM Server 10 would send the presence information of IM Client 11 to IM Client 12 only.

[0003] Conventional Instant Messenger (IM) systems, such as MSN, Yahoo, AOL and so on, can therefore provide computer users who have subscribed to and are connected

on-line to a service provider's server to "chat" or exchange messages/files instantaneously. Some of these instant messages may not be routed through the service provider's server but directly with other IM clients by communicating instantaneously through the use of Peer-to-Peer protocols after the IM server has/servers have authenticated and linked the IM clients together. These conventional IM systems also provide fully asynchronous communication. In addition, some of these systems now enable live audio or video conferencing.

[0004] US patent publication no. 2008/0140796, assigned to Microsoft Corp., describes a system for communicating with devices through instant messaging system. As shown in FIG. 2, a user 17 and an IM client 18 are connected to a device 70 through an automated IM client 50. The automated IM client 50 has a host user identity 52 and a contact list 54. The device 70 may be a plurality of devices 70a, 70b, etc. The automated IM client 50 is connected to the devices 70, 70a, 70b by a dispatcher 60. The IM system thus provides authentication and security for communication between an IM client 18 and an automated IM client 50. Various types of devices are mentioned but this document does not disclose details of the dispatcher 30.

[0005] In another approach, one may resort to plug-and-play (or PnP) protocol to control different devices that are connected to the automated IM client 50. PnP defines a set of protocols for devices to join a network and describe themselves so that other devices and users can use them without setup or configuration. However, PnP protocol may not be suitable for IM.

[0006] It can thus be seen that there exists a need for a plug-in to allow a registered user to control a device remotely via an automated IM client by instant messaging.

Summary

[0007] The following presents a simplified summary to provide a basic understanding of the present invention. This summary is not an extensive overview of the invention, and is not intended to identify key features of the invention. Rather, it is to present

some of the inventive concepts of this invention in a generalised form as a prelude to the detailed description that is to follow.

[0008] The present invention describes an automated instant messaging remote control system. In the system, a device to be controlled remotely by an IM user is connected to an automated IM client (machine). A plug-in operating in the IM client (machine) facilitates such remote control. Advantageously, the plug-in accepts strings of commands, which conventional IM systems do not permit.

[0009] In one embodiment, the present invention provides an automated Instant Messaging (IM) client. The automated IM client comprises: an IM client (machine) that is configurable to connect to a network and an associated IM server once power to the IM client (machine) is turned on; a device for connection to the IM client (machine); and a plug-in that is operable in the IM client (machine) to facilitate remote control of the device by an IM client (user).

[0010] In another embodiment, the present invention provides an instant messaging system. The IM system comprises: a communication machine operable to connect to an instant messaging service; and a plug-in operable in the communication machine; wherein the communication machine has a communication port to which a device to be controlled remotely by an IM user is connected, and the plug-in is operable to accept a string of commands, which are then sorted into a queue for execution.

[0011] In yet another embodiment, the present invention provides a process for remote control of a device via instant messaging. The process comprises: configuring an IM client (machine) to connect to a network and an associated IM service; connecting the device to the IM client (machine); inputting a supported device list in the IM client (machine); and operating a plug-in to facilitate remote control of the device by an IM user; wherein the plug-in is operable to sort a string of commands into a queue, which comprises a synchronous loop and/or an asynchronous loop.

[0012] In one embodiment of the process for remote control of a device via IM, it further comprises: retrieving the device's identity by an auto-detect component of the

plug-in; comparing the retrieved device's identity against those in the supported device list; if there is a match in the comparison, adding an associated device driver to a device mapping list in a device listing component of the plug-in, and adding function calls associated with the device driver to a function mapping list in a function component of the plug-in; adding a token of the retrieved device's identity into each command in the string of commands; interpreting the string of commands by an interpreter, which is a component of the plug-in, and sorting the commands into a queue before sending the commands to the executing component of the plug-in; the executing component identifies synchronous loop and/or asynchronous loop in the commands and accordingly initiates a timer or counter for each loop; and carrying out the function calls associated with the device and drivers, before passing control to a device linker, which is a part of a device connection component of the plug-in, so that each command is sent to the device to perform a task.

Brief Description of the Drawings

[0013] This invention will be described by way of non-limiting embodiments of the present invention, with reference to the accompanying drawings, in which:

[0014] FIG. 1 shows a conventional instant messaging system;

[0015] FIG. 2 shows an instant messaging system disclosed in US 2008/0140796;

[0016] FIG. 3A illustrates a remote control system according to an embodiment of the present invention; whilst FIG. 3B illustrates a plug-in operable in an automated IM client; and

[0017] FIG. 4 illustrates a device registration process for use with the remote control system shown in FIG. 3A;

[0018] FIG. 5 illustrates instructions flow process in the plug-in shown in FIG. 3B; and

[0019] FIG. 6 shows a robotic arm for use with the remote control system.

Detailed Description

[0020] One or more specific and alternative embodiments of the present invention will now be described with reference to the attached drawings. It shall be apparent to one skilled in the art, however, that this invention may be practised without such specific details. Some of the details may not be described at length so as not to obscure the invention. For ease of reference, common reference numerals or series of numerals will be used throughout the figures when referring to the same or similar features common to the figures.

[0021] FIG. 3A shows an instant messaging (IM) remote control system 100 according to an embodiment of the present invention. As shown in FIG. 3A, the IM remote control system 100 is simplified for illustration purposes; the system 100 shows an IM server 10, an IM client (user) 11 and an automated IM client (machine) 150. A device 180 to be controlled remotely by the IM client (user) 11 is connected to the automated IM client (machine) 150. For purposes of configuring the IM client (machine) 150, a user management station 152 is provided. The IM client (machine) 150 is setup so that it connects to a network, such as the internet 30, and IM server 10 once its power is turned on. In the IM client (machine) 150 is a plug-in 160 software, which is operable to facilitate remote control of the device 180.

[0022] FIG. 3B shows the functional components of the plug-in 160. As shown in FIG. 3B, the plug-in 160 includes a channel component 162, an interpreter 164, instructions executing component 166, a device connection component 168, an auto-connect component 170, a device listing component 172 and a function component 174. Connected to the device connection component 168 is a driver component 176, which allows a driver associated with the device 180 to be connected to the device connection component 168.

[0023] The device 180 to be controlled remotely via IM may be a machine, an instrument, a camera, a recorder, an air-conditioning unit, a heater, etc. that are parts of an automated factory, intelligent building, surveillance system, etc. For illustration purposes, the IM remote control system 100 will be described with the device 180 being a robotic arm 400.

[0024] When the plug-in 160 is turned on and the device 180 is connected to the IM client (machine) 150 for the first time, some initializations take place. The device listing component 172 contains a list of the devices 180, their function calls and locations of the associated drivers. The channel component 162 then detects the type of communication port 154 that the device 180 is connected to. The auto-detect component 170 detects the device 180 that is connected to a communication port 154 of the automated IM client (machine) 150. The auto-detect component 170 retrieves the device's identity and passes control to the device listing component 172 for a match against a device mapping list 173, whilst the function component 174 retrieves the available/associated function calls. When an in-coming command is received by the automated IM client (machine) 150, the channel component 162 transmits the in-coming command to the interpreter 164. The interpreter 164 converts the command or groups of command into a queue 165 and the executing component 166 then carries out the command, one at a time, from the queue 165. The device connection component 168 then makes a connection with the driver component 176, which facilitates connection to the associated device driver so that the device 180 is controllable remotely by an IM client (user) 11.

[0025] During initial setup of the automated IM client (machine) 150, for example, from a user management station 152, the device 180 is registered for the plug-in 160 to recognize the device 180 and a supported device list 156 is generated. FIG. 4 shows the registration process 200 of a device 180. When the device 180 is connected to the automated IM client (machine) 150 via the communication port 154 (such as a USB, serial or parallel port) and the plug-in 160 is turned on, the auto-detect component 170 retrieves the device's unique identity in step 210. For example, the identity of the device 180 is obtained from a pin of a parallel port; from a byte from a handshake signal in a serial port; or a USB function call when the USB port is connected. With the

supported device list 156 input in step 220, a decision is made, in step 225, whether there is a match between the retrieved device identity and any one identity in the support device list 156. If there is no match in step 225, the registration process proceeds to step 230 and returns a message that the device 180 is not supported by the plug-in 160.

[0026] If the decision in step 225 is positive, the registration process proceeds to step 235. In step 235, the associated device driver is added to the device mapping list 173 in the device listing component 172. The associated function calls are then added or populated, in step 240, into a function mapping list 175 in the function component 174. Proceeding to step 245, the device mapping list 173 is updated. Once the entire device registration process 200 is complete, the device 180 is registered in the device mapping list 173 and is then ready, in step 250, to accept any command.

[0027] When the automated IM client (machine) 150 is connected to the IM server 10 and a command or group of commands is received, processing 300 of the command or group of commands is shown in FIG. 5. As shown in FIG. 5, the process 300 starts with receipt of the command or group of commands in step 305. In step 310, the command or group of commands is tokenized with the device 180 identity and is then sent to the interpreter 164. Together with the function mapping list 175 from the function component 174, the interpreter 164 sorts the command or group of commands into a queue 165 before sending it to the executing component 166. In step 330, the executing component 166 carries out each command, one at a time. Before executing the command, the device's parameters are inputted, in step 335, from a parameter storage whilst positions of elements of the device 180 are obtained, in step 340, from a sensor monitor. As illustrated in FIG. 5, the sensor monitor is connected to sensors of each element of the device 180, such as the robot's gripper, wrist, elbow, shoulder and base.

[0028] Execution of the command is carried out in step 330. After executing step 330, the process 300 identifies the synchronous and asynchronous portions in step 345. The process 300 then proceeds to step 350, in which counters and trackers are set. Upper and lower limits of each sensor in the sensor monitor are also inputted into the counters/trackers, in step 350.

[0029] From step 350, the process 300 executes the synchronous and/or asynchronous portions of the command. The synchronous portion of the command proceeds to step 360, in which a timer is set for each synchronous loop and the associated functions are called out, in step 355. On the other hand, the asynchronous portion of the command proceeds to step 365. In step 365, a counter is set for each conditional loop and the associated functions are called out, again in step 355. From the function calls in step 355, the process 300 proceeds to a device linker 177, in step 370; the device linker 177 is part of the driver component 176.

[0030] With inputs of the device identity from step 310 and device mapping list 173 from step 375, the device linker 177, in step 370, channels the command to the appropriate device driver in step 380 for the associated device 180 to perform a task.

[0031] There are a number of functionalities of the plug-in 160 that are different from the known art. For example, the interpreter 164 accepts a group of commands and sorts them into a queue 165 for the executing component 166 to carry out the command; in contrast, the conventional IM system only allows a command to be received at the automated IM client (machine) 150 one at a time. With this feature, a user 11 is able to send an entire instructions set or subset to remotely re-program the device 180, instead of from only the user management station 152.

[0032] Moreover, the instruction set or subset can be carried out synchronously and/or asynchronously in a loop. The asynchronous loop may be conditional, such as, a do-until loop, an if-then-else loop, and so on. The conditions may be defined in the instructions or by the results of execution.

[0033] In another example, auto-detect component 170 of the plug-in 160 allows a device 180 to be connected to the automated IM client (machine) 150 regardless of the communication port 154. The communication port 154 may be a USB, a serial, parallel, RS, and so on. An advantage of the auto-detect component 170 is that it allows backward compatibility of the device 180 in relation to the automated IM client (machine) 150. With the auto-detect component 170 and device listing component 172, an IM

client (user) 11 is able to remotely view the device mapping list 173 and the states of the devices 180 that are connected to the automated IM client (machine) 150.

[0034] An implementation of the above IM remote control system 100 according to the present invention is illustrated with a robotic arm 400 in FIG. 6. As shown in FIG. 6, the robotic arm 400 has a base 410, an upper arm 420, a fore arm 430, a wrist 440 and a gripper 450. Between the base 410 and the upper arm 420 is a shoulder joint J1. Between the upper arm 420 and the fore arm 430 is an elbow joint J2. Each axis of motion or rotation of the robotic arm 400 has a sensor 412,422,432,442 to feedback to the IM client (user) of the position/orientation of the base 410, upper arm 420, fore arm 430 and wrist 440, respectively. Preferably, the gripper 450 may have another sensor to feedback the degree of its opening or closing.

[0035] As an example of the above implementation, a command structure may be as follow:

```
device1 {s = RotateBase(90,1); MoveElbow(50,0); if (s==90)<MoveWrist(30,0)>;}
```

where: device1 is the device identity; in this example, the device is the robotic arm 400;

{ represents start of an instruction;

} represents end of the instruction;

s represents return status from a function call;

; represents end of a command;

RotateBase(90,1) represents a function call to rotate the base 410 by 90 degrees; parameter 1 in the second argument of the function call denotes a synchronous call type; a parameter 0 would denote an asynchronous call.

This particular command instructs the robot arm 400 to rotate the base 410 by 90 degrees; when the base 410 reaches the 90-degree angle, the wrist would rotate 30 degrees asynchronously. This is then followed by rotation of the elbow J2 by 50 degrees.

[0036] As another example, a request for all the positions of elements of the robotic arm 400 may be as follow:

```
TestRobot {GetAllAxisPositionStatusToUser();}
```

[0037] While specific embodiments have been described and illustrated, it is understood that many changes, modifications, variations and combinations thereof could be made to the present invention without departing from the scope of the invention. For example, the IM client (machine) may be a wireless enabled communication device, such as a PDA or mobile telephone, operable to send and receive instant and text messaging.

CLAIMS:

1. An automated Instant Messaging (IM) client comprising:
 - an IM client (machine) that is configurable to connect to a network and an associated IM server once power to the IM client (machine) is turned on;
 - a device for connection to the IM client (machine); and
 - a plug-in comprising an interpreter to convert and to sort an IM message containing a list of commands into a command queue for execution, so that said plug-in is operable in the IM client (machine) to facilitate remote control of the device by an IM client (user).
2. An automated IM client according to claim 1, wherein the IM client (machine) comprises a user management station.
3. An automated IM client according to claim 2, wherein the user generates a supported device list to define the devices that are connectable to the IM client (machine).
4. An automated IM client according to any one of claims 1-3, wherein the plug-in comprises:
 - a channel component;
 - an auto-detect component;
 - a device connection component;
 - a device listing component;
 - a function component; and
 - an executing component;wherein after initial registration of a device at the automated instant messaging client, a device mapping list is created in the device listing component whilst a function mapping list is created in the function component so that a command is sent from the command queue to the executing component, device connection component and a driver component, such that the latter connects a driver associated with the device to perform a task associated with the relevant command.

5. An automated IM client according to claim 4, wherein the command comprises an instruction for the device to perform or a request for position data from the device.
6. An automated IM client according to any one of the preceding claims, wherein the device comprises any one of the following: a machine, an instrument, a camera, a recorder, an air-conditioning unit, a heater and so on, that forms part of an automated factory, intelligent building or surveillance system.
7. An automated IM client according to any one of the preceding claims, wherein the command queue comprises a synchronous loop and/or asynchronous loop, each type of loop is identified by an argument in a function call of the device.
8. An automated IM client according to any one of claims 4-7, wherein the auto-detect component is operable to distinguish the following communication port to which the device is connected: USB, serial, parallel or RS.
9. An instant messaging (IM) system comprising:
 - a communication machine operable to connect to an instant messaging service;
 - and
 - a plug-in operable in the communication machine;wherein the communication machine has a communication port to which a device to be controlled remotely by an IM user is connected, and the plug-in is operable to accept an IM message containing a string of commands, which are then converted and sorted by an interpreter into a command queue for execution.
10. An IM client according to claim 9, wherein the queue comprises one or more synchronous loops and/or asynchronous loops.
11. A command structure for use in the plug-in according to any one of the preceding claims, the command structure comprising:
 - { to represent start of a command string;
 - } to represent end of a command string;
 - s to represent return status from a function call;

; to represent end of each command; and

function(x,y) to represent a function call, with y=1 to denote a synchronous call and y=0 to denote an asynchronous call.

12. A process for remote control of a device via instant messaging, the process comprising:

configuring an IM client (machine) to connect to a network and an associated IM service;

connecting the device to the IM client (machine);

inputting a supported device list in the IM client (machine); and

operating a plug-in to facilitate remote control of the device by an IM user;

wherein the plug-in is operable by an interpreter to convert and to sort an IM message containing a string of commands into a command queue, which comprises a synchronous loop and/or an asynchronous loop.

13. A process according to claim 12, further comprises registering the device, said registration comprising:

retrieving the device's identity by an auto-detect component of the plug-in;

comparing the retrieved device's identity against those in the supported device list; and

if there is a match in the comparison, adding an associated device driver to a device mapping list in a device listing component of the plug-in, and adding function calls associated with the device driver to a function mapping list in a function component of the plug-in.

14. A process according to claim 12 or 13, further comprises adding a token of the retrieved device's identity into each command in the string of commands.

15. A process according to claim 14, further comprises sending the commands from the command queue to the executing component of the plug-in.

16. A process according to claim 15, wherein the executing component identifies synchronous loop and/or asynchronous loop in the commands and accordingly initiates a timer or counter for each loop.
17. A process according to claim 16, further comprises carrying out the function calls associated with the device and drivers, before passing control to a device linker, which is a part of a device connection component of the plug-in, so that each command is sent to the device to perform a task.
18. A process according to any one of claims 12-17, wherein the plug-in is operable to remotely program the device instead of executing the command queue.