(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
27 February 2003 (27.02.2003)

PCT

(10) International Publication Number
WO 03/017089 A2

(51) International Patent Classification[7]: G06F 9/00

(21) International Application Number: PCT/GB02/03121

(22) International Filing Date: 8 July 2002 (08.07.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
01124795.9 13 August 2001 (13.08.2001) CN

(71) Applicant *(for all designated States except US)*: **INTERNATIONAL BUSINESS MACHINES CORPORATION** [US/US]; New Orchard Road, Armonk, NY 10504 (US).

(71) Applicant *(for MG only)*: **IBM UNITED KINGDOM LIMITED** [GB/GB]; PO Box 41, North Harbour, Portsmouth, Hampshire PO6 3AU (GB).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **CAI, Hong** [CN/CN]; 4-402, Building 19, MoShiKouXiLi, ShiJingShan District, PRC Beijing (CN).

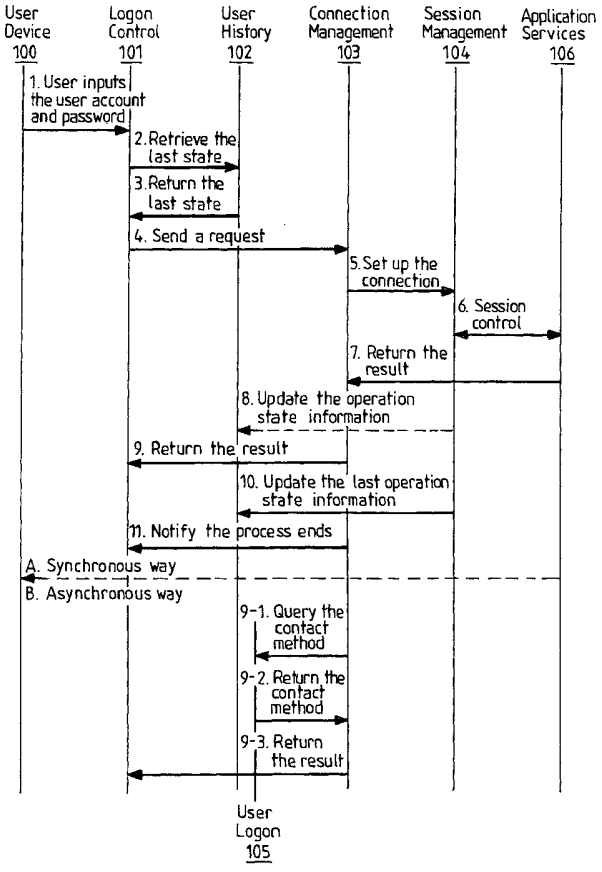
(74) Agent: **LITHERLAND, David, Peter**; IBM United Kingdom Limited, Intellectual Property Law, Hursley Park, Winchester, Hampshire SO21 2JN (GB).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZM, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW),

[Continued on next page]

(54) Title: KEEPING PERSISTENCY WHILE SWITCHING BETWEEN MODALITIES



(57) Abstract: Disclosed is a method and device for keeping persistency while switching between modalities, which include: (1) searching out the last operation state of the application service accessed by the user from the user history , in response to a request for accessing an application service from a user; (2) connecting to the application service and continuing to execute the application service from the last operation state; (3) updating the last operation state in the user history based on the execution result of the application service. A user could flexibly selects different favorable terminal to continue with unfinished application service by adopting the invention.

WO 03/017089 A2

Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

KEEPING PERSISTENCY WHILE SWITCHING BETWEEN MODALITIES

Field of the Invention

The invention relates to a method and apparatus for accessing an application service in a computer network.

Background of the Invention

With the Internet becoming pervasive in every field of society, there appear more and more intelligent devices capable of processing information. These devices, for example, include PCs, mobile phones, palmtop computers, etc., most of which can be connected with a network to process information. Such a network is often referred to as a pervasive network. When a user accesses an application service, he/she prefers using one of those devices in one specific environment, for example using a PC to access the application service in the office, while using another different device in another specific environment, for example using a mobile phone to continue executing the application service in the car. That is to say, "keeping persistency" is required for the same application service even after switching from one device to another different device. However, for the current technology in accessing the same application service via multiple devices, a user can only be allowed to use one device to access the same application service from the beginning to the end. And if the user switches to another device to access the application service, he has to access the application service from scratch. The current technology does not solve the problem on how to keep persistency in this situation.

Nowadays, applications tend to become more and more modularized, but not a big packed module any more. An application may consist of a plurality of highly modularized independent components (hereinafter named application logic components). So the whole procedure to execute an application becomes to execute all the independent application logic components one by one. The present invention makes use of this modularization to solve the "persistency maintenance" issue.

Summary of the Invention

An aim of the invention is to provide a method and device for keeping persistency while switching between modalities, so that a user

using multiple terminal devices to access an application service can continue to execute the application service using device B from the last operation state point when using device A, instead of from scratch.

Users can use different terminal devices flexibly to continue to execute an unfinished application service by implementing the invention.

In a first aspect, the invention provides a method offering continuous service, which includes the following steps: (1) searching out the last operation state of the application service accessed by the user from the user history, in response to a request for accessing an application service from a user; (2) connecting to the application service and continuing to execute the application service from the last operation state; (3) updating the last operation state in the user history based on the execution result of the application service.

A second aspect of the invention also provides an apparatus for offering continuous service, which includes: means for searching out the last operation state of the application service accessed by the user from the user history in response to a request for accessing an application service from a user; means for connecting to the application service and continuing to execute the application service from the last operation state; means for updating the last operation state in the user history based on the execution result of the application service.

In a third aspect, the invention also provides a device offering continuous service, which includes a user history module for recording the historical call information of at least one user for at least one application service based on the user ID; a logon control module for identifying or verifying the user requesting for accessing an application service, retrieving the historical call information of the user for the application service from the user history module, and extracting the call ID of the last operation of the user for the application service from the historical call information; a session management module for connecting to the application service and executing the application service based on the historical call information, and for updating the historical call information recorded in the user history module based on the execution result; and a connection management module for setting up the connection to the session management module based on the extracted call ID.

In one embodiment, the start point for the next time to execute the same application will be determined based on the last operation state

information recorded for all the application logic components of the application, when a modularized application is executed.

Embodiments of the invention will now be described, by way of example only, with reference to the accompanying drawings:

Brief Description of the Drawings

Fig. 1 is a block diagram showing the device for keeping persistency according to an embodiment of the invention; and

Fig. 2 is a processing flow chart of the above device.

Detailed Description of the Invention

As shown in figure 1, the device for keeping persistency comprises a logon control module 101 for identifying and verifying the user identification (user ID) of a user requesting an application service 106 from the user device 100. A user history module 102 records the historical call information of each of the users for their application services, respectively. The user history module 102 may record the historical call information of the users based on their user Ids. In addition, the logon control module 101 may search out the historical call information of certain user for a specific application service, from the user history module 102. And then the module 101 may extract the call ID of the last connection from the historical call information for the user, the call ID corresponding to the serial number of each application logic component in the application service 106. Furthermore, the logon control module 101 may accept the logon requests from various user devices 100 adopting various protocols, for example the device may be any device like mobile phone, palmtop computer, PC etc., the protocols may be WML, HTML, Notes client etc. A session management module 104 connects to the background application service 106 and executes the application logic components in the corresponding background application service continuously based on the call ID of the last connection extracted by the logon control module 101. A connection management module 103 sets up the connection to the session management module 104 based on the call ID extracted by the logon control module 101, and gets the information of the devices which the user usually uses to receive messages from the user register module 105 when required. In addition, the session management

module 104 updates the historical call information recorded in the history module according to the execution result of the background application service 106, so that the call information can be used for the next access to the background application service. A user register module 105 stores each of the users' devices registering information, for example, in the form of script or database. The user register module 105 may register utility information about a user's several devices as follows:

800AM~5:00PM, telephone xxxxxxx is preferable;
other time, mobile phone xxxxxx is preferable;
if not found, try to use Instant Message;
if still not found, and the service priority is high, contact the user's manager or college via enterprise name service (e.g. bluepage of IBM).

How to use the device to keep persistency will now be described in detail by reference to figure 2.

A. When a user uses one device to access an application service for the first time or access the application service again after completing an access to the same application service.

Suppose that user U uses a WAP mobile phone 100 to access an application service 106 via WML language, the process is as follows:

1. User U inputs his user account and password via WAP mobile phone 100, connects to the logon control module 101 and chooses the application service 106 that the system offers.

2. The logon control module 101 queries the user history module 102 to search out the last operation state of the application service 106 for user U.

3. The user history module 102 returns the logon control module 101 the information indicating the user U has not accessed the application service 106 before or the user has finished his access and needs to resume.

4. The logon control module 101 sends out a request to the connection management module 103, with the request including the user (i.e. user U) ID, the call ID, the application ID included in the call ID for the application service 106, and the call progress information (the

call progress information here is initialized to be 0 or 1 because it's the first call). The request here is a general request independent of the user's device. For example, the information included in a typical request may have the format:

SERVICE http://myservice.com/office/
From: caihong@cn.ibm.com
To: office@myservice.com
CallID: 0_office@myservice.com
Content Type: application/xml
ContentLength:

5. Once the connection management module 103 receives the above request, it will extract the information about the user U (for example, the user ID)and the operation state information of the application service (for example, the call ID, the application ID and the call progress information), and sets up a new connection to the session management module 104.

6. The session management module 104 sets up a connection to the application service 106, executes the application service 106 from the initial state for the user U, and monitors the state of each of the application logic components of the application service 106, so as to get the operation state of the application service 106.

7. Each of the application logic components of the application service 106 returns its execution result to the connection management module 103.

8. The session management module 104 updates the call progress information of the application service 106 for the user U in the user history module 102, when one application logic component of the application service 106 is completed and another application logic component is going to be started.

9. The connection management module 103 then returns the execution result of application service 106 to the logon control module 101, and further returns to the user U.

The processes in the above steps 7-9 are based on an assumption, that is, this sort of application can be split into small application logic components which can be executed recursively until the user U issues

an interrupt request or the execution of the application service 106 is finished. In addition, the persons skilled in the art will understand that the operation of the above step 8 is a sort of temporary operation, and that in order to save space and time, this step can be omitted.

10. When the user U sends out an interrupt request or the execution of the application service 106 is finished, the session management module 104 updates the call progress information in the user history module with the latest state of this operation, so as to make the user history module to always keep the latest operation state of the application service 106 for the user U.

11. When the connection management module 103 detects the interrupt request sent by the user U or detects that the execution of application service 106 is finished, it will notify the logon control module 101 that this operation ends.

The above description has shown the working process of the user U accessing the application service 106 for the first time. It can be seen from the above description that the operation state of the application service 106 for the user U is stored in the user history module 102. Next time the user U may continue to access the application service 106 from the point of the last operation state which is recorded in the user history module, and therefore the persistency can be kept.

In addition, in the above embodiment, the request in step 4 is a general request independent of the user device 100. So the session management module 104 executes the corresponding application logic component only based on the call progress information therein, which makes it possible for the user to use various terminal devices to access the same application service.

B. In the case of A, when the user stops using WAP mobile phone to access the application service 106 and switches to another device, for example a Lotus Notes Client to continue executing the application service, the working process is as follows:

1. The User U inputs the user account and password via Lotus Notes client 100, connects to the logon control module 101 and chooses the application service 106 offered by the system.

2. The logon control module 101 queries the user history module 102 to search out the last operation state of the application service 106 for the user U.

3. The user history module 102 returns the logon control module 101 the information indicating that the user has accessed the application service 106 before, as well as the last operation state of the application service 106 for the user, i.e., the operation progress information.

4. The logon control module 101 sends a request to the connection management module 103, the request comprising user (i.e. user U) ID, call ID, the ID included in the call ID for the application service 106, and the call progress information. The request here is a general request independent of the user device. The information for a typical request, for example, may have the format below:

SERVICE http://myservice.com/office/
From: caihong@cn.ibm.com
To: office@myservice.com
CallID: 01_office@myservice.com
Content Type: application/xml
ContentLength:

5. Once the connection management module 103 receives the above request, it will extract the information for the user (user U) and the last operation state of the application service, and sets up a new connection to the session management module 104.

6. The session management module 104 sets up a connection to the application service 106, continues to execute the application service 106 from the point of the last operation state for the user U, and monitors the state of each of the application logic components of the application service 106, so as to get the latest operation state of the application service 106.

7. Each of the application logic components of the application service 106 returns its execution result to the connection management module 103.

8. The session management module 104 updates the call progress information of the application service 106 for the user U in the user history module 102, when one application logic component of the

application service 106 is ended up and another application logic component is going to be started.

9. The connection management module 103 then returns the execution result of application service 106 to the logon control module 101, and further returns the result to the user U.

In the above steps 7-9, the application service 106 can be split into application logic components which can be executed recursively until the user U sends out an interrupt request or the execution of the application service 106 is finished. In addition, since the operation of the above step 8 is a temporary operation, this step can be omitted in order to save space and time.

10. When the user U sends out an interrupt request or the execution of the application service 106 is finished, the session management module 104 updates the call progress information in the user history module with the latest operation state.

11. When the connection management module 103 detects an interrupt request sent by the user U or detects that the execution of application service 106 is finished, it will notify the logon control module 101 that this operation ends.

The above description has shown the working process of the user U continuing to access the application service. It can be seen from the above description that next time when the user U continues to access the application service 106, the application service 106 can be executed continuously based on the operation state stored in the user history module 102, and therefore the persistency can be kept. In addition, by changing the latest progress information in the user history module 102, the user history module always keeps the latest operation state of the application service for the user to start the next access. Therefore the persistency can be kept.

In addition, the request in step 4 is a device independent request, so the session management module 104 executes the corresponding application logic components only based on the call progress information therein, which makes it possible for the user to use various terminal devices to access the same application service.

In addition, although not illustrated, it is still understandable that the logon control module 101 returns the execution result of the application service in the communication way through which user U sends out the request in the above cases of A and B. For example the logon control module 101 returns the result to the user's WAP mobile phone in case A, and returns the result to the user's Lotus Notes Client in case B.

Another case will be described now, as the following

C. The asynchronous request from the user

When a user or an application service requires the successive procedure to be executed asynchronously, without specifying what kind of device to be used for receiving the execution result, the user register module 105 can be used to determine the right user device used to receive the execution result.

Specifically, the difference between the asynchronous and synchronous procedures lies in step 9.

As shown in figure 2, in asynchronous way:

9-1. The connection management module 103 calls the users register module 105 to determine the proper contact method to communicate with the user U.

9-2. The user register module 105 returns the proper contact method for the user U now to the connection management module 103.

9-3. The connection management module 103 sends out the execution result via the detected proper contact method to communicate with the user U.

Thus in asynchronous way, the invention also can always return the information to the user U in time.

The invention has been described with reference to the embodiments and the drawings which are not used to limit the invention. The invention can be modified and improved without departing from the scope of the claims.

CLAIMS

1. A method for accessing an application service in a pervasive network comprising:

in response to a user request to access the application service, recalling the last operation state of the application service from a user history;

connecting to the application service and continuing to execute the application service from the last operation state; and

updating the last operation state in the user history based on the execution result of the application service.

2. The method according to claim 1 further comprising:

returning the execution result to the user using a contact method appropriate to the user.

3. The method according to claim 2, characterized in that the contact method is that which is used by the user when requesting access to the application service.

4. The method according to claim 2, characterized in that the contact method is selected from a plurality of contact methods pre-registered by the user.

5. Apparatus for providing access to an application service in a pervasive network comprising:

means for recalling the last operation state of the application service accessed by the user from the user history responsive to a request for accessing an application service from a user;

means for connecting to said application service and continuing to execute said application service from said last operation state; and

means for updating the last operation state in the user history based on said execution result of said application service.

6. Apparatus according to claim 5, further comprising:

means for presenting said execution result to said user using a contact method appropriate to the user.

7. Apparatus according to claim 6, wherein the contact method is that which is used by said user when requesting access to said application service.

8. Apparatus according to claim 6, wherein the contact method is selected from a plurality of methods pre-registered by said user.

9. Apparatus for keeping persistency in a computer network comprising:

a user history module (102) for recording the historical call information of at least one user for at least one application service based on the user ID;

logon control module (101)for identifying or verifying the user requesting access to an application service, retrieving the historical call information of the user for the application service from the user history module (102), and extracting the call ID of the last operation of the user for the application service from the historical call information;

a session management module (104) for connecting to the application service and executing the application service based on the historical call information, and for updating the historical call information recorded in the user history module(102) based on the execution result; and

a connection management module (103), for setting up the connection to the session management module(104) based on the extracted call ID.

10. The apparatus according to claim 9, further comprising:

a user registrar module (105) for registering information on at least one device used by a user for connection, the connection management module (103) being operable to select one device from the user register module (105) for the user to receive the execution result of the application service.

11. The apparatus according to claim 10, further comprising:

means for presenting the execution result to the selected device.

12. The apparatus according to claim 11, wherein said terminal is the one used by the user when requesting access to the application service.

13. The apparatus according to claim 11, wherein the device is selected from a plurality of terminals pre-registered by the user.

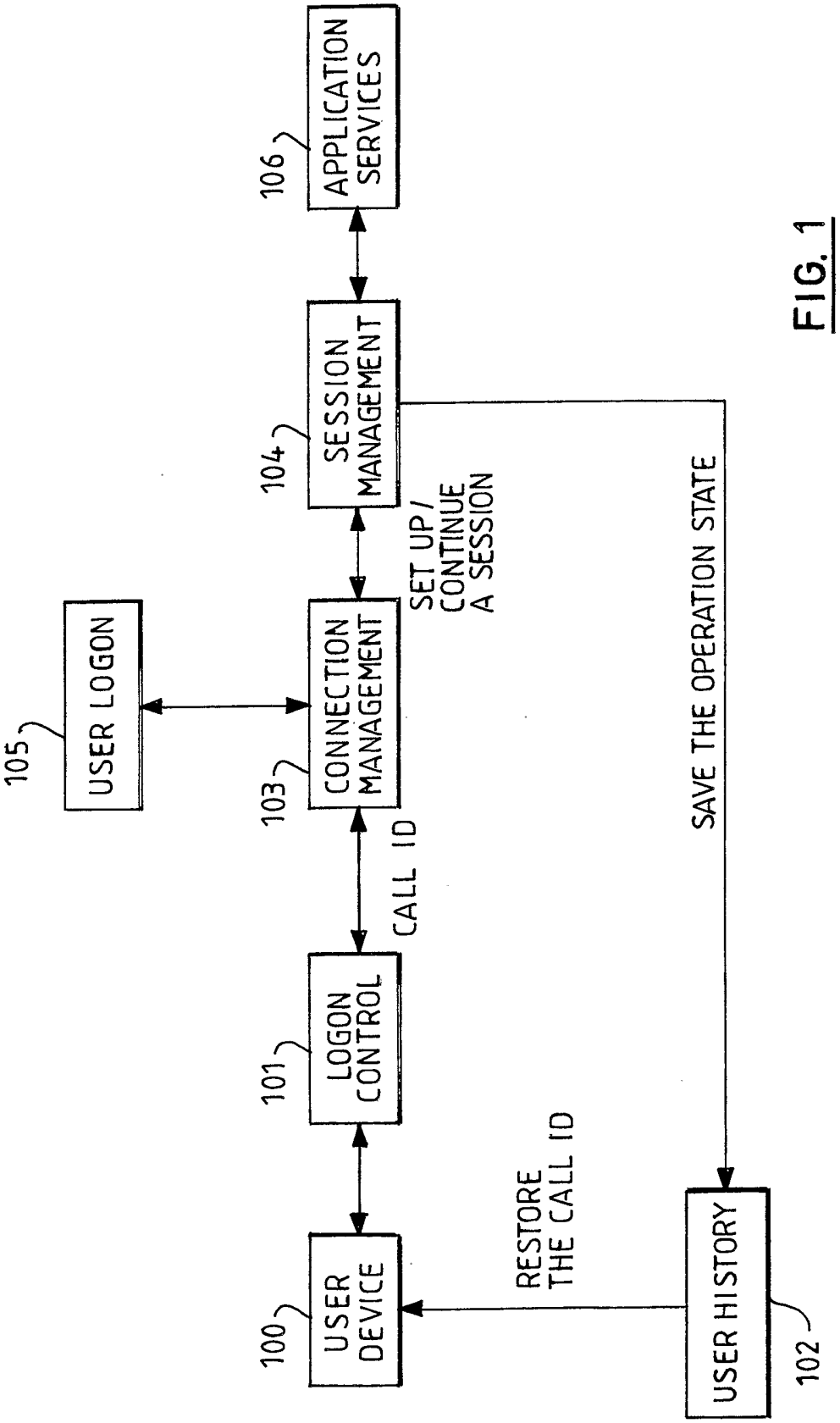

APPLICATION SERVICES
106
SESSION MANAGEMENT
104
SET UP / CONTINUE A SESSION
USER LOGON
105
CONNECTION MANAGEMENT
103
CALL ID
SAVE THE OPERATION STATE
LOGON CONTROL
101
RESTORE THE CALL ID
USER DEVICE
100
USER HISTORY
102

FIG. 1

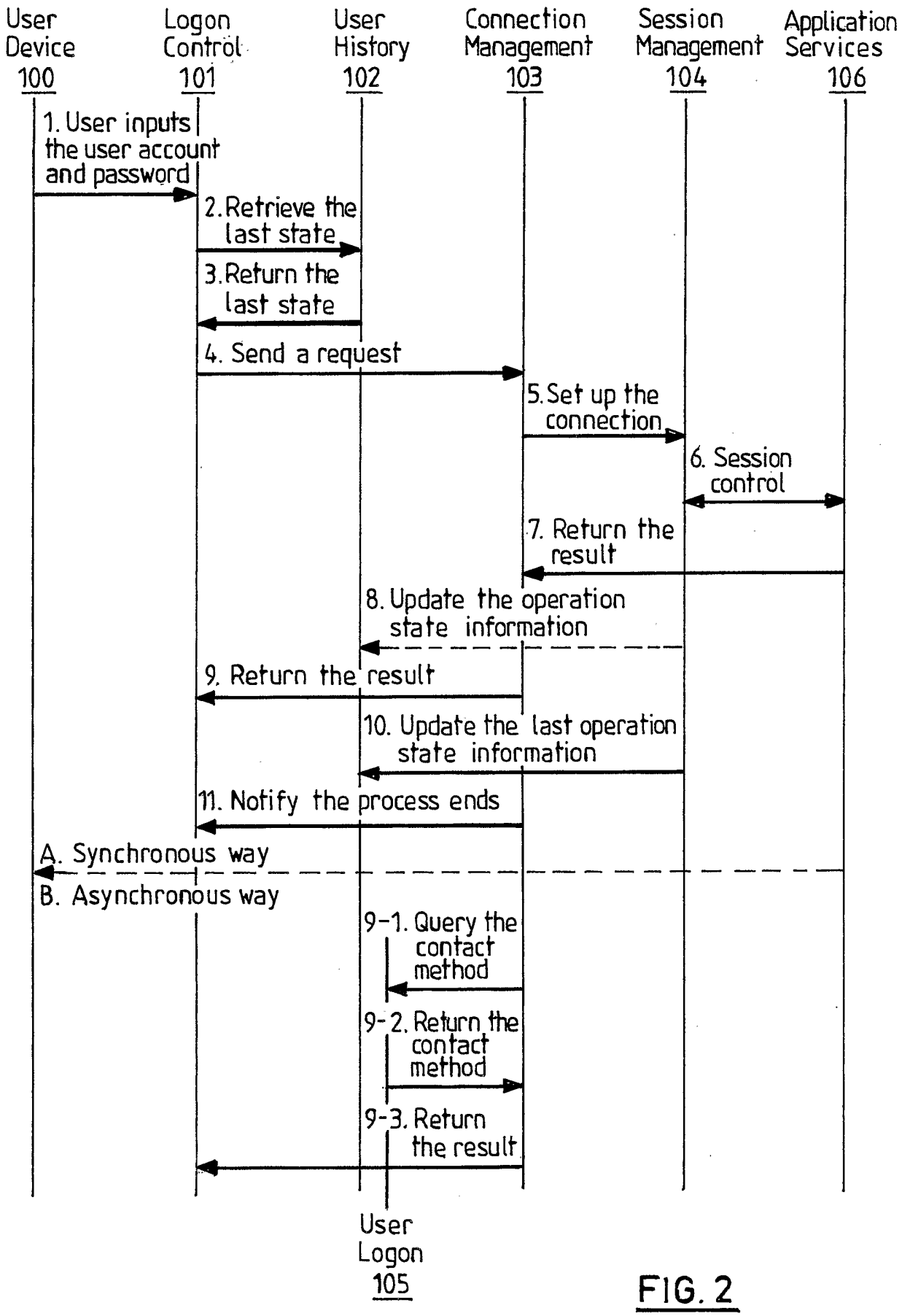

User Device 100
Logon Control 101
User History 102
Connection Management 103
Session Management 104
Application Services 106
1. User inputs the user account and password
2. Retrieve the last state
3. Return the last state
4. Send a request
5. Set up the connection
6. Session control
7. Return the result
8. Update the operation state information
9. Return the result
10. Update the last operation state information
11. Notify the process ends
A. Synchronous way
B. Asynchronous way
9-1. Query the contact method
9-2. Return the contact method
9-3. Return the result
User Logon 105

FIG. 2