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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
21 June 2012 (21.06.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/079650 A1

(51) International Patent Classification:
H04L 29/06 (2006.01) *H04L 29/08* (2006.01)

(21) International Application Number:
PCT/EP2010/070112

(22) International Filing Date:
17 December 2010 (17.12.2010)

(25) Filing Language: English

(26) Publication Language: English

(71) **Applicant** *(for all designated States except US)*: **NOKIA SIEMENS NETWORKS OY** [FI/FI]; Karaportti 3, FI-02610 Espoo (FI).

(72) **Inventors; and**
(75) **Inventors/Applicants** *(for US only)*: **MARTON, Gabor** [HU/HU]; Jagelló út 42/A, H-1124 Budapest (HU). **GESZTESI, Gabor** [HU/HU]; Szlávy u. 50, H-1103 Budapest (HU).

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

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

Published:

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

(54) **Title**: USER INTERACTION FOR WEB RESOURCES

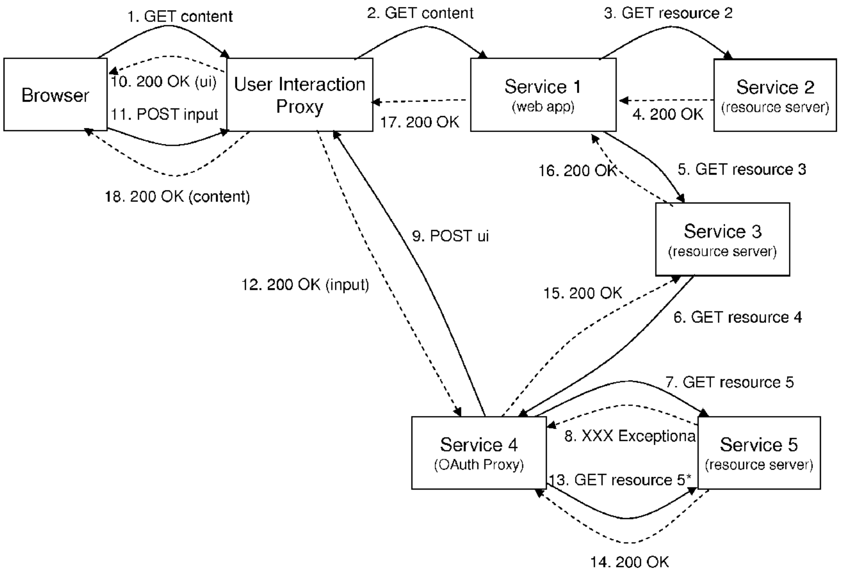


Fig. 6

(57) **Abstract**: It is provided an apparatus (AuthProxy), comprising first receiving means for receiving a service request from a first party (resource server); detecting means for detecting a requirement for user interaction in order to comply with said service request; requesting means for requesting said user interaction from a second party (User Interaction Proxy) different from the first party; and second receiving means for receiving said user interaction as a response from said second party.

WO 2012/079650 A1

Description

Title

User interaction for web resources

Field of the invention

The present invention relates to apparatuses, a methods, a system, and a computer program product related to web resources. More particularly, the present invention relates to apparatuses, methods, a system, and a computer program product for user interaction when accessing web resources.

Background of the invention

More and more Web applications are composed from simpler services operated by different parties. Many of the contributing services are "UI-less" (UI: user interface) services by their nature, i.e. they don't have a user interface as they do not need a direct interaction with the end-user (e.g.: weather information service, media file store, currency exchange converter, search engine). We call such contributing services *Web resources*. In a general setting, we have a web of services connected to each other, some playing the role of a client, others being resource servers and yet others appearing in both roles.

In the context of this application, the terms "service" and "resource" are used synonymously, except if a differentiation between these two terms is explicitly made. That is, a "resource" according to the present application may be e.g. a service or a resource, and a "service" according to the present application may be e.g. a resource or a service, too.

In a narrower sense, the invention is related to identity management, because the problem statement originates from a related issue called cross-service authorization. *Cross-service authorization* – also called as *Web API authorization* – is a process in which a provider service determines if a consumer service is allowed to access some of its resources or operations owned by a user (the resource owner). A *provider service* exposes *resources* – e.g. photos uploaded by users – to other services via an API (Application Programming Interface). A *consumer service* – e.g. a printing service – accesses such a resource via an API.

HTTP (Hypertext Transfer Protocol) is "an application-level protocol for distributed, collaborative, hypermedia information systems. It is used for retrieving inter-linked resources led to the establishment of the World Wide Web".

HTTP is a *request-response* protocol where "a requestor sends a request message to a replier system which receives and processes the request, ultimately returning a message in response".

OAuth is "an open protocol to allow secure API authorization in a simple and standard method from desktop and web applications" (http://oauth.net/), based on HTTP. Provider services require the presence of a valid OAuth token in the resource request (HTTP request). Speaking in terms of OAuth 2.0 terminology, the client (consumer service) first obtains an *access grant* from the *resource owner*, then – in exchange of authenticating itself and submitting the access grant – it obtains an *access token* from an authorization server, and finally – in exchange of submitting the access token – it retrieves the resource from the resource server (see Fig. 1). (In OAuth 2.0, the functions of Resource Owner, Authorization Server and Resource Server are clearly separated, whereas in OAuth 1.0 they are all tied to the Service Provider by means of the three types of token: unauthorized request token, authorized request token and access token.)

An *OAuth Proxy* is a utility component supporting OAuth clients (consumer services) in accessing resources at OAuth-enabled resource servers (provider services) by hiding the details of the OAuth protocol. All the client has to know is the URL of the resource to be retrieved; the rest is done by the OAuth Proxy residing between the client and the resource server (for more details, see, as an example, the Google OAuth Proxy explained below).

When a service "deep inside" the web of interconnected services "suddenly" requires end-user interaction, a problem arises.

Fig. 2 shows an example of such a situation. A browser requests a map of the present environment from a location tracker (1.). The location tracker requests the location information from a location source via an OAuth Proxy (2., 3.). That is, the location source is configured such that it requires an authentication of the requestor. Since authentication information is not available, the Location Source returns (4.) a 401 Unauthorized response to the OAuth Proxy, because the request (3.) did not contain a valid access token. At this point, the OAuth Proxy needs end-user interaction (has to redirect the user to the Location Source for authorizing the access to their personal data), but it is not in a direct contact with the end-user.

One option is to involve the client service (Location Tracker) and make it return the necessary UI. For this purpose, however, the Location Tracker service must "understand" the situation i.e. the service logic must be prepared for the user interaction case as well, whereas this interaction – between the user and the Location Source – is not a business of the Location Tracker at all.

So a better solution may be one that does not involve the client service.

Fig. 3 shows a more complex example with one more level of indirection or distance between the user and the interacting service. In this case, a web application ("My Desktop") is inserted between the browser and the location tracker.

The current practice does not provide a generic solution to the problem.

The *Google OAuth Proxy* (http://code.google.com/intl/hu/apis/gadgets/docs/oauth.html) is a service which – in combination with a client-side software (JavaScript) – supports developers in "writing gadgets that run on iGoogle and access a user's private data from any website that supports the OAuth protocol" and "is designed to make it easier for gadgets to use the OAuth standard". The sample gadget code (http://gadget-doc-examples.googlecode.com/svn/trunk/opensocial-gadgets/yahoo-presence.xml) provided by Google accesses Yahoo's API and the OAuth-related configuration is a simple as:

```
<OAuth>
 <Service name="yahoo">
```

```
  <Request param_location="uri-query"
url="https://api.login.yahoo.com/oauth/v2/get_request_token"
/>
  <Access param_location="uri-query"
url="https://api.login.yahoo.com/oauth/v2/get_token" />
  <Authorization
url="https://api.login.yahoo.com/oauth/v2/request_auth" />
 </Service>
</OAuth>
```

Accessing the user's status message – i.e. invoking an OAuth-enabled API method – is done by the following function:

```
var url = getPresenceUrl();
var params = getDefaultParams();
gadgets.io.makeRequest(url, function(response) {
  makeRequestCallback(fetchStatusDone, response);
}, params);
```

where the makeRequestCallback function is implemented by the gadget for performing the steps necessary for authorization – i.e. directing the user to the authorization page at the service provider – like for example:

```
function makeRequestCallback(nextCallback, response) {
  if (response.oauthApprovalUrl) {
    var onOpen = function() {
      showOneSection('waiting');
    };
    var onClose = function() {
      completeAuthorization();
    };
    var popup = new gadgets.oauth.Popup(
      response.oauthApprovalUrl,
      "width=800,height=600",
      onOpen,
      onClose
    );
    ...
  }
}
```

That is, the Google OAuth Proxy relies on its *immediate* client for handling the user interaction. It cannot cope with the situation shown in Fig.3.

SiteMinder (http://www.ca.com/us/collateral/product-briefs/na/ca-siteminder.aspx) "enables users to authenticate only once and have access both to
web applications protected by CA SiteMinder WAM [Web Access Management] and non-web applications with access controlled by CA SSO [Single Sign-On]". The solution authenticates the user before they access the protected service. It cannot cope with the situation where the need for the user interaction arises after the service has been invoked.

The *captive portal technique* (http://en.wikipedia.org/wiki/Captive portal) — used e.g. for controlling the WiFi access in hotels — "forces an HTTP client on a network to see a special web page (usually for authentication purposes) before using the Internet normally. A captive portal turns a Web browser into an authentication device. This is done by intercepting all packets, regardless of address or port, until the user opens a browser and tries to access the Internet. At that time the browser is redirected to a web page which may require authentication and/or payment, or simply display an acceptable use policy and require the user to agree". Similarly to the previous two examples, this technique cannot cope with "lately arising" user interaction requests.

WS-Coordination (https://www.ibm.com/developerworks/webservices/library/ws-transjta/) "is a coordination framework to enable distributed participants to agree on a universal outcome over their individual activities". "Essentially this means that when distributed participants (two application servers on different machines for example), that would otherwise be unable to complete in a controlled manner, would be able to use WS-Coordination to group the actions of each participant together, and further manage them by ensuring that they all agree to a singular outcome for all the actions that they have individually performed under this coordination context".

The basic WS-Coordination flow is illustrated by Fig.5.
"1. App1 makes a request to the activation service on a Coodinator.
2. The Coordinator begins a new activity and responds to App1 with its CoordinationContext (XML information of the Coordinator).
3. App1 makes a request to the registration service to register to use coordination protocol X.
4. App1 invokes App2 in whatever way it wishes, passing across the CoordinationContext for the Coordinator.
5. App2 makes a request to the registration service (using parameters such as port information found in the CoordinationContext passed by App1) to register to use coordination protocol Y.
6. App2 finishes its work and control returns back to App1, and the activity is called to complete.
7. The Coordinator responds to App1 using protocol X style messages.
8. The Coordinator responds to App2 using protocol Y style messages".

The relation of WS-Coordination to the present invention is distant: WS-Coordination does not contain the concept of an intercepting proxy at all.

Summary of the invention

It is an object of the present invention to improve the prior art.

According to a first aspect of the invention, there is provided an apparatus, comprising first receiving means for receiving a service request from a first party; detecting means for detecting a requirement for user interaction in order to comply with said service request; requesting means for requesting said user interaction from a second party different from the first party; and second receiving means for receiving said user interaction as a response from said second party.

In the apparatus, said request for user interaction may comprise an information element for provisioning a user interface for executing said user interaction.

The apparatus may further comprise resource requesting means for requesting a second resource from a resource device in order to comply with said service request; and the detecting means may be adapted to detect the requirement for user interaction based on an exception information received from the resource device in response to the request for the resource.

In the apparatus, the user interaction may be requested from a user who has caused the service request received from the first party.

In the apparatus, said request for user interaction may comprise an identification of said service request.

In the apparatus, the requesting means may be adapted to provide, together with said request for said user interaction, a transaction identifier, wherein the transaction identifier is comprised in the service request received from the first party.

The apparatus may further comprise response means for responding to the first party, wherein the response may comprise a trigger for terminating service execution, compliant to the service request, in the first party.

The apparatus may further comprise handle generating means for generating a handle which is related to the service request received from the first party, and wherein the handle is unique in the realm of the apparatus; handle detecting means for detecting if a handle message comprising the handle is received by the apparatus from the second party; and the requesting means may be adapted to request the user interaction if the handle message is detected.

The apparatus may further comprise identifying means for identifying the second party based on the handle message.

In the apparatus, the service request may comprise a first user identification, and the apparatus may further comprise correlating means for correlating a second service request from the first party with the received user interaction based on a second identification comprised in the second service request.

According to a second aspect of the invention, there is provided an apparatus, comprising first receiving processor for receiving a service request from a first party; detecting processor for detecting a requirement for user interaction in order to comply with said service request; requesting processor for requesting said user interaction from a second party different from the first party; and second receiving processor for receiving said user interaction as a response from said second party.

In the apparatus, said request for user interaction may comprise an information element for provisioning a user interface for executing said user interaction.

The apparatus may further comprise resource requesting processor for requesting a second resource from a resource device in order to comply with said service request; and the detecting processor may be adapted to detect the requirement for user interaction based on an exception information received from the resource device in response to the request for the resource.

In the apparatus, the user interaction may be requested from a user who has caused the service request received from the first party.

In the apparatus, said request for user interaction may comprise an identification of said service request.

In the apparatus, the requesting processor may be adapted to provide, together with said request for said user interaction, a transaction identifier, wherein the transaction identifier is comprised in the service request received from the first party.

The apparatus may further comprise response processor for responding to the first party, wherein the response may comprise a trigger for terminating service execution, compliant to the service request, in the first party.

The apparatus may further comprise handle generating processor for generating a handle which is related to the service request received from the first party, and wherein the handle is unique in the realm of the apparatus; handle detecting processor for detecting if a handle message comprising the handle is received by the apparatus from the second party; and the requesting processor may be adapted to request the user interaction if the handle message is detected.

The apparatus may further comprise identifying processor for identifying the second party based on the handle message.

In the apparatus, the service request may comprise a first user identification, and the apparatus may further comprise correlating processor for correlating a second service request from the first party with the received user interaction based on a second identification comprised in the second service request.

According to a third aspect of the invention, there is provided a resource access proxy comprising an apparatus according to any of the first and second aspects.

According to a fourth aspect of the invention, there is provided an apparatus comprising sending means for sending a service request to a first device; first receiving means for receiving a request for user interaction from a second device in order to comply with said service request, wherein the second device is different from the first device; requesting means for requesting said user interaction from a user interaction device; second receiving means for receiving said user interaction from said user interaction device; and responding means for responding, to said request for user interaction, based on said user interaction received from said user interaction device.

In the apparatus, said service request may comprise an identification of the device sending said service request.

The apparatus may further comprise third receiving means for receiving the service request from said user interface.

In the apparatus, said response to said request for user interaction may be adapted to trigger a second sending of said service request to said first device.

The apparatus may further comprise fourth receiving means for receiving a retry message comprising a handle in response to the service request; handle message providing means for providing a handle message comprising the handle to the second device, and the second receiving means may be adapted to receive the request for user interaction in response to the provision of the handle message.

The apparatus may further comprise identifying means for identifying the second device based on the received retry message comprising the handle.

In the apparatus, said request for user interaction may comprise an identification of said service request.

The apparatus may further comprise identifier generating means for generating an identifier for the service request, wherein said response to said request for user interaction may comprise said identifier of said service request.

The apparatus may further comprise fifth receiving means for receiving an error message in response to the service request; correlating means for correlating the error message with the request for user interaction based on said identifier.

According to a fifth aspect of the invention, there is provided an apparatus comprising sending processor for sending a service request to a first device; first receiving processor for receiving a request for user interaction from a second device in order to comply with said service request, wherein the second device is different from the first device; requesting processor for requesting said user interaction from a user interaction device; second receiving processor for receiving said user interaction from said user interaction device; and responding processor for responding, to said request for user interaction, based on said user interaction received from said user interaction device.

In the apparatus, said service request may comprise an identification of the device sending said service request.

The apparatus may further comprise third receiving processor for receiving the service request from said user interface.

In the apparatus, said response to said request for user interaction may be adapted to trigger a second sending of said service request to said first device.

The apparatus may further comprise fourth receiving processor for receiving a retry message comprising a handle in response to the service request; handle message providing processor for providing a handle message comprising the handle to the second device, and the second receiving processor may be adapted to receive the request for user interaction in response to the provision of the handle message.

The apparatus may further comprise identifying processor for identifying the second device based on the received retry message comprising the handle.

In the apparatus, said request for user interaction may comprise an identification of said service request.

The apparatus may further comprise identifier generating processor for generating an identifier for the service request, wherein said response to said request for user interaction may comprise said identifier of said service request.

The apparatus may further comprise fifth receiving processor for receiving an error message in response to the service request; correlating processor for correlating the error message with the request for user interaction based on said identifier.

According to a sixth aspect of the invention, there is provided a user interaction proxy comprising an apparatus according to any of the fourth and fifth aspects.

According to a seventh aspect of the invention, there is provided an apparatus, comprising first receiving means for receiving a service request from a first party; requesting means for requesting, in order to fulfill the service request, a resource from a second party; second receiving means for receiving a response to the request for the resource; detecting means for detecting if the response comprises a first retry message comprising a handle and a trigger for terminating service execution, compliant to the request for the resource; responding means for responding to the service request by a second retry message comprising the handle and a trigger for terminating service execution, compliant to the service request, if the detecting means detects that the response is the first retry message.

According to an eighth aspect of the invention, there is provided an apparatus, comprising receiving means for receiving a service request from a first party; requesting means for requesting, in order to fulfill the service request, a resource from a second party; wherein the service request comprises an identifier; and the request for the resource comprises the identifier.

According to a ninth aspect of the invention, there is provided an apparatus, comprising first receiving processor for receiving a service request from a first party; requesting processor for requesting, in order to fulfill the service request, a resource from a second party; second receiving processor for receiving a response to the request for the resource; detecting processor for detecting if the response comprises a first retry message comprising a handle and a trigger for terminating service execution, compliant to the request for the resource; responding processor for responding to the service request by a second retry message comprising the handle and a trigger for terminating service execution, compliant to the service request, if the detecting processor detects that the response is the first retry message.

According to an tenth aspect of the invention, there is provided an apparatus, comprising receiving processor for receiving a service request from a first party; requesting processor for requesting, in order to fulfill the service request, a resource from a second party; wherein the service request comprises an identifier; and the request for the resource comprises the identifier.

According to an eleventh aspect of the invention, there is provided a resource server, comprising an apparatus according to any of the seventh to tenth aspects.

According to a twelfth aspect of the invention, there is provided a System, comprising a first resource apparatus according to any of the first to third aspects; and a user interaction apparatus according to any of the fourth to sixth aspects; wherein the second party of the first resource apparatus comprises the user interaction apparatus; the second device of the user interaction apparatus comprises the first resource apparatus; the request for said user interaction of the first resource apparatus is the received request for user interaction of the user interaction apparatus; and the response to said user request for user interaction of the user interaction apparatus is the user interaction received by the first resource apparatus.

The system may further comprise second resource apparatus according to any of the seventh to eleventh aspects; wherein the first party of the first resource apparatus may comprise the second resource apparatus; the first device of the user interaction apparatus may comprise the second resource apparatus; the service request sent by the user interaction apparatus may be the service request received by the second resource apparatus; and the request for the resource of the second resource apparatus may be the service request received by the first resource apparatus.

According to a thirteenth aspect of the invention, there is provide a method, comprising receiving a service request from a first party; detecting a requirement for user interaction in order to comply with said service request; requesting said user interaction from a second party different from the first party; and receiving said user interaction as a response from said second party.

The method may be a method of user interaction.

In the method, said request for user interaction may comprise an information element for provisioning a user interface for executing said user interaction.

The method may further comprise requesting a second resource from a resource device in order to comply with said service request; and the detecting may be adapted to detect the requirement for user interaction based on an exception information received from the resource device in response to the request for the resource.

In the method, the user interaction may be requested from a user who has caused the service request received from the first party.

In the method, said request for user interaction may comprise an identification of said service request.

In the method, the requesting may be adapted to provide, together with said request for said user interaction, a transaction identifier, wherein the transaction identifier may be comprised in the service request received from the first party.

The method may further comprise responding to the first party, wherein the response may comprise a trigger for terminating service execution, compliant to the service request, in the first party.

The method may further comprise generating a handle which is related to the service request received from the first party, and wherein the handle is unique in the realm of an apparatus performing the method; detecting if a handle message comprising the handle is received by the apparatus from the second party; and the requesting may be adapted to request the user interaction if the handle message is detected.

The method may further comprise identifying the second party based on the handle message.

In the method, the service request may comprise a first user identification, the method may further comprise correlating a second service request from the first party with the received user interaction based on a second identification comprised in the second service request.

According to a fourteenth aspect of the invention, there is provided a method comprising sending a service request to a first device; receiving a request for user interaction from a second device in order to comply with said service request, wherein the second device is different from the first device; requesting said user interaction from a user interaction device; receiving said user interaction from said user interaction device; and responding, to said request for user interaction, based on said user interaction received from said user interaction device.

The method may be a method of user interaction.

In the method, said service request may comprise an identification of the device sending said service request.

The method may further comprise receiving the service request from said user interface.

In the method, said response to said request for user interaction may trigger a second sending of said service request to said first device.

The method may further comprise receiving a retry message comprising a handle in response to the service request; providing a handle message comprising the handle to the second device, and the request for user interaction may be received in response to the provision of the handle message.

The method may further comprise identifying the second device based on the received retry message comprising the handle.

In the method, said request for user interaction may comprise an identification of said service request.

The method may further comprise generating an identifier for the service request, wherein said response to said request for user interaction may comprise said identifier of said service request.

The method may further comprise receiving an error message in response to the service request; correlating the error message with the request for user interaction based on said identifier.

According to a fifteenth aspect of the invention, there is provided a method, comprising first receiving means for receiving a service request from a first party; requesting means for requesting, in order to fulfill the service request, a resource from a second party; second receiving means for receiving a response to the request for the resource; detecting means for detecting if the response comprises a first retry message comprising a handle and a trigger for terminating service execution, compliant to the request for the resource; responding means for responding to the service request by a second retry message comprising the handle and a trigger for terminating service execution, compliant to the service request, if the detecting means detects that the response is the first retry message.

According to a sixteenth aspect of the invention, there is provided a method, comprising receiving means for receiving a service request from a first party; requesting means for requesting, in order to fulfill the service request, a resource from a second party; wherein the service request comprises an identifier; and the request for the resource comprises the identifier.

The method according to any of the fifteenth and sixteenth aspects may be a method of service requesting.

According to a seventeenth aspect of the invention, there is provided a computer program product including a program comprising software code portions being arranged, when run on a processor of an apparatus, to perform the method according to any one of the thirteenth to sixteenth aspects.

The computer program product may comprise a computer-readable medium on which the software code portions are stored, and/or the program may be directly loadable into a memory of the processor.

By the apparatuses, methods, system, and computer program product, a smooth user experience may be provided without compromising security because it is avoided that services in the middle of the chain from the browser (or a related user interaction proxy) and the resource requesting a user information (or a related authentication proxy) are involved in the transmission of user credentials. That is, the credentials are kept as a secret between the browser (or its related user interaction proxy) and the service in question (or its related authentication proxy), and hidden from all other parties taking part in the service session.

It is to be understood that any of the above modifications can be applied singly or in combination to the respective aspects to which they refer, unless they are explicitly stated as excluding alternatives.

Brief description of the drawings

Further details, features, objects, and advantages are apparent from the following detailed description of the preferred embodiments of the present invention which is to be taken in conjunction with the appended drawings, wherein

Fig. 1 shows an OAuth sequence;

Fig. 2 shows a simple example of a system comprising an OAuth proxy;

Fig. 3 shows a more complex example of a system comprising an OAuth proxy;

Fig. 4 shows a basic WS-Coordination flow;

Fig. 5 shows an example system with a workflow on which embodiments of the present invention are based;

Fig. 6 shows a system with a workflow according to an embodiment of the invention;

Fig. 7 shows a system with a workflow according to an embodiment of the invention;

Fig. 8 shows a system with a workflow according to an embodiment of the invention;

Fig. 9 shows an apparatus according to an embodiment of the invention;

Fig. 10 shows a method according to an embodiment of the invention;

Fig. 11 shows an apparatus according to an embodiment of the invention;

Fig. 12 shows a method according to an embodiment of the invention;

Fig. 13 shows an apparatus according to an embodiment of the invention;

Fig. 14 shows a method according to an embodiment of the invention;

Fig. 15 shows a system according to an embodiment of the invention; and

Fig. 16 shows a system according to an embodiment of the invention.

Detailed description of certain embodiments

Herein below, certain embodiments of the present invention are described in detail with reference to the accompanying drawings, wherein the features of the embodiments can be freely combined with each other unless otherwise described. However, it is to be expressly understood that the description of certain embodiments is given for by way of example only, and that it is by no way intended to be understood as limiting the invention to the disclosed details.

Moreover, it is to be understood that the apparatus is configured to perform the corresponding method, although in some cases only the apparatus or only the method are described.

Fig. 5 shows a system with a workflow on which embodiments of the present invention may be based. The system is not limiting the invention, and other systems, e.g. involving more or less servers, and other workflows are within the scope of the invention. The system according to Fig. 5 is exemplarily used hereinafter to explain the present invention.

The system comprises a browser. The browser may be any browser such as MS Internet Explorer, Mozilla Firefox, Opera, etc., that allows user interaction.

For example, based on user interaction, the browser requests some content from Service 1 (1.). Service 1 may be a Web application such as iGoogle.

In order to fulfill the request, Service 1 may have to retrieve resources from Service 2 and Service 3 (2., 4.). Service 2 may response by providing the requested resource such as specific data (3.).

In order to fulfill the request of Service 1, Service 3 may have to request a resource from Service 4 (5.), which may be an OAuth proxy.

In order to fulfill the request of Service 3, Service 4 may have to request a resource from Service 5 (6.). Service 5 requires further information (such as a valid OAuth access token) in order to provide the requested resource. Since the information is not available, it responds to Service 4 with an exception message (7.). Service 4 (the OAuth proxy) needs to find a way how to get the required information. Getting the required information requires interaction with the end-user (e.g. redirecting the browser to Service 5 for authorizing an OAuth token). In general, there are two potential approaches (8.) to achieve this: Service 4 may request the required information following the way back via Service 3 and Service 1 to the browser, or it may directly contact the browser. The prior art does not provide a solution to this problem.

According to embodiments of the invention, an architecture and a workflow are as follows:

- A *User Interaction Proxy (UIP)* resides between the browser (user agent) and the realm of cooperating services. All communication — requests and responses — between the browser to the services flow through the Proxy.
- When a service such as Service 4 of Fig. 5 requires user interaction (or, more detailed, requires an information form the user that may be potentially obtained by user interaction), it requests the Proxy to insert a dialog (user interface, UI) into the communication flow. The way a service requests the Proxy to insert the dialog can vary. Some exemplary embodiments are outlined further below.
- As the UIP is aware of all the pending requests from the browser to the services, it is able to insert the UI

into the communication flow as a response to the pending request that initially triggered the cooperation of the services.

Note that the concept of relying on a proxy naturally fits into "controlled" service delivery environments – as opposed to the open Internet – where access to the services from the outside world happens in some controlled way. One example is the service Marketplace in the SERVERY project (http://www.celtic-initiative.org/projects/servery/) where access to the services is controlled by a gateway which is the single entry point to the services on the Marketplace. Another example is a group of service components deployed to a computing cloud where all the services reside behind an HTTP reverse proxy which is responsible for end-user authentication and access authorization. In such cases, the User Interaction Proxy is an additional function of the otherwise already present service access gateway.

In the following, some embodiments of the invention are described at greater detail, with reference to the exemplary architecture outlined in Fig. 5, wherein a user interaction proxy (UIP) as described hereinabove is inserted between the browser and Service 1.

A) Synchronous variant:

In this variant, the service which needs user interaction synchronously contacts the User Interaction Proxy and waits until the user interaction is completed.

Fig. 6 illustrates the embodiment by means of HTTP message exchanges. Steps 1 to 8 are the same as Steps 1 to 7 of Fig. 5, wherein Step 1 of Fig. 5 is split into Steps 1 and 2 of

Fig. 6 because of insertion of the UIP. Step 8 indicates the delivery of a 401 Unauthorized message returned by Service 5 (corresponding to the Location Source of Fig. 2 and Fig. 3) in response to the resource request in step 7.

Service 4 "suddenly" needs end-user interaction, so it contacts (9) the User Interaction Proxy (UIP) and passes a UI to it. The UIP returns (10) the UI to the Browser, the user inputs the required data, the Browser submits (11) the response to the Proxy, and eventually the Proxy returns (12) to Service 4 with the requested input, i.e. user information such as user credentials.

After having received the user input, Service 4 requests again the resource from Service 5 using the user input (13.). Then, Service 5 provides the requested resource to Service 4 (14.). Thereafter, Services 4, 3, and 1 propagate the respective resources back to the browser (via the UIP) in the usual way (15.-18.).

There may be two options how the authentication proxy (Service 4) may know the address of the UIP: The address of the UIP may be preconfigured in the authentication proxy, or the address is delivered with the request through the service chain. This may be achieved in that the UIP inserts a special HTTP header into the forwarded HTTP request telling its own address. E.g.:

```
GET .../my-desktop HTTP/1.1
X-User-Interaction-Proxy: http://.../uip
Host: ...
```

In this variant, neither Service 1 nor Service 2 or Service 3 is aware that an interaction with the user is required and ongoing. In particular, the user information is not passed through Services 1 to 3, avoiding a potential security threat.

This solution may cause a problem following from the synchronous nature of the call (9) from Service 4 to the User Interaction Proxy. All the services in the call chain (2, 5, 6) from the Proxy to Service 4 – including Service 4 itself – are waiting for the user interaction to be completed. In other words, the threads of execution assigned to the requests in the chain are blocked. In real deployments, web servers and servlet containers may use thread pools with a fixed number of threads, the number typically falling into the range of hundreds (200-800; see http://www.springsource.com/files/uploads/tomcat/tomcatx-performance-tuning.pdf, for example). So a few hundred pending requests – waiting for end-user response – may block the server ("thread pool exhaustion"). The response time of web servers (without user interaction) is in the order of milliseconds, whereas the response time of users is around a second in the best case. However, users sometimes do not answer immediately if ever (e.g. doing something else or being away from the computer).

This problem may be overcome by asynchronous variants of embodiments of the invention as outlined below:

B) Asynchronous variant with retry handle:

In this variant, the synchronous call to the User Interaction Proxy from the service (or its related authentication proxy such as Service 4 in Fig. 5) – which needs user interaction – is replaced with retry and callback (the Proxy calling the service).

An embodiment of this variant is shown in Fig. 7. Steps 1 to 8 are the same as the corresponding steps in Fig. 6.

Service 4 now returns (9.) immediately with a special response to Service 3, and this special response is propagated (10., 11.) by the services in the call chain (in the example: Services 1 and 3) back to the User Interaction Proxy. The special response also contains a handle which uniquely identifies the case in the realm of Service 4.

The Proxy recognizes the special response and does not propagate it back to the Browser; instead, it extracts the handle and contacts (12.) Service 4 directly, passing the handle, to retrieve (13.) the necessary UI. Then it sends (14.) the UI to the Browser, receives (15.) the input, and sends (16.) the input (user information) to Service 4. Then the user interaction proxy repeats (18.) the original request (i.e. the request of step 2.), resulting in a similar call chain (19.-22.) as before.

After having received the user information (16.) and the repeated request (22.), Service 4 requests again the resource from Service 5 using the user input (23.). Then, Service 5 provides the requested resource to Service 4 (24.). Thereafter, Services 4, 3, and 1 propagate the respective resources back to the browser (via the UIP) in the usual way (25.-28.).

In this variant, the UIP may contact the authentication proxy taking the address of the authentication proxy from the www-authenticate header returned by the authentication proxy in the special response (e.g. a special HTTP 401 response). Thus, the authentication proxy does not have to know the address of the UIP.

Service 4 may correlate the received user information and the repeated request e.g. by a user identification if is transported from the UIP through the chain of Services (Services 1 to 4), and if it is comprised in the user information.

This way, the user information is not passed through Services 1 to 3, avoiding a potential security threat. In addition, the open threads of the first request chain (steps 2. to 7.) are closed within the usual response time for webservices, i.e. milliseconds, such that the servers are not blocked.

If the user information has a sufficient long lifetime, it may be used for several requests of the same user. Thus, a user need not to input its user information once for each request, and user convenience is achieved without compromising security, if the authentication proxy (Service 4) is secure. If, on the other side, the user information is valid for one request only, it may be correspondingly marked in the message from the UIP to the authentication proxy.

In some embodiments, the correlation required for the asynchronous variants between the user information received at the authentication proxy (Service 4 in Fig. 7) and the repeated request from Service 3 may be based on an identifier which preferably should be unique in the realm of the authentication proxy. For example, the handle may be comprised in the repeated requests (steps 18. to 25. in Fig. 7). The handle may be but need not be included in the message transmitting the user information from the UIP to the authentication proxy (step 16. in Fig. 7) because this dialogue between UIP and authentication proxy is triggered by the transmission of the handle in step 12.

Thus, an unambiguous correlation of the repeated request and the user information may be achieved. However, the service chain has to support the transmission of the identifier from the UIP to the authentication proxy.

The following sequence illustrates one possible implementation of variant B by means of HTTP messages. It shows an embodiment where variant B (Fig. 7) is applied to the OAuth Proxy problem (Fig. 3); the numbering of the steps is in line with the message numbering in Fig. 7. The correspondence of the Services in Fig. 7 and the notations in the sequence are as follows:

- User Interaction proxy: my desktop;
- Service 1: My Desktop
- Service 3: LocationTracker
- Service 5: LocationSource

1. The Browser issues an HTTP request to My Desktop. But since the User Interaction Proxy resides between the Browser and the service, it is actually received by the Proxy:
   ```
   GET .../my-desktop HTTP/1.1
   Host: ...
   ```
2. The User Interaction Proxy forwards the request to the My Desktop:
   ```
   GET .../my-desktop HTTP/1.1
   Host: ...
   ```
3. … (intentionally left blank to stay in line Fig. 7).
4. …
5. My Desktop requests the map from Location Tracker:
   ```
   GET .../location-tracker HTTP/1.1
   Host: ...
   ```
6. Location Tracker → OAuth Proxy:
   ```
   GET .../location-source?id=user1234 HTTP/1.1
   Host: ...
   ```
7. OAuthProxy → LocationSource:
   ```
   GET .../location-source?id=user1234 HTTP/1.1
   Host: ...
   ```

8. LocationSource --> OAuthProxy (as the request did not contain a valid access token; the Token authentication scheme is defined by the OAuth specification):

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Token realm="..." error="..."
...
```

9. OAuthProxy → LocationTracker:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Proxy realm="..."
url="http://.../oauth-proxy/case2345"
...
```

Notice the Proxy authentication scheme which is a fictitious one, just serving as an example; in combination with the 401 response code, it implements the "Retry" response shown in Fig. 7; the case2345 identifier in the url field implements the "handle" shown in Fig. 7.

10. Location Tracker → My Desktop (just propagating the Unauthorized response):

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Proxy realm="..."
url="http://.../oauth-proxy/case2345"
...
```

11. My Desktop → User Interaction Proxy:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Proxy realm="..."
url="http://.../oauth-proxy/case2345"
...
```

12. The User Interaction Proxy recognizes the Proxy authentication scheme, retrieves the URL of the OAuth Proxy from the WWW-Authenticate field, and contacts the OAuth Proxy:

```
GET .../oauth-proxy/case2345 HTTP/1.1
Host: ...
```

13. OAuth Proxy → User Interaction Proxy:

```
HTTP/1.1 200 OK
Content-Type: text/plain
Content-Length: ...

HTTP/1.1 302 Found
Location: https://.../location-
source/authorize?type=web_server&client_id=client34
56&redirect_uri=https%3A%2F%2F...%2Foauth-
proxy/case2345
...
```

Notice that the HTTP response (200) contains another HTTP response (302) in its body. This latter HTTP response is in fact the implementation of the "ui" indicated Fig. 7.

14. The User Interaction Proxy extracts the redirection (302) response from the received message and returns it to the Browser:

```
HTTP/1.1 302 Found
Location: https://.../location-source/authorize?type=web_server&client_id=client3456&redirect_uri=https%3A%2F%2F...%2Foauth-proxy/case2345
...
```

The user interaction will now take place, which – due to the nature of the OAuth protocol – takes a more complicated form than the simple "input" indicated in Fig. 7.

a. Browser → User Interaction Proxy → Location Source (the request also being proxied by the User Interaction Proxy, but not shown in separate steps here):

```
GET .../location-source/authorize?type=web_server&client_id=client3456&redirect_uri=https%3A%2F%2F...%2Foauth-proxy/case2345 HTTP/1.1
Host: ...
```

b. …

c. … (according to OAuth, the Location Source first authenticates the user; steps not shown)

d. …

e. Browser → User Interaction Proxy → Location Source (user authorizes the access):

```
POST .../location-source/authorize?type=web_server&client_id=client3456&redirect_uri=https%3A%2F%2F...%2Foauth-proxy/case2345 HTTP/1.1
Host: ...
```

f. Location Source → User Interaction Proxy → Browser :

```
HTTP/1.1 302 Found
Location: https://.../oauth-proxy/case2345?code=code4567
...
```

15. Browser → User Interaction Proxy:

```
GET .../oauth-proxy/case2345?code=code4567 HTTP/1.1
Host: ...
```

16. User Interaction Proxy → OAuth Proxy:

```
GET .../oauth-proxy/case2345?code=code4567 HTTP/1.1
Host: ...
```

Two more OAuth-specific steps take place; the OAuth Proxy obtains an access token from the Location Source in exchange to the access code (code4567):

a. OAuth Proxy → Location Source:

```
POST .../location-source/token HTTP/1.1
```

```
Host: ...
Content-Type: application/x-www-form-urlencoded
Content-Length: ...

grant_type=authorization_code&client_id=...&client_
secret=...&code=code4567&redirect_uri=https%3A%2F%2
F...%2Foauth-proxy/case2345
```

b. Location Source → OAuth Proxy:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: ...
Cache-Control: no-store

{
  "access_token":"token56789",
  "expires_in":3600
}
```

The OAuth Proxy caches the received access token for later use (step 23).

17. OAuth Proxy → User Interaction Proxy:

```
HTTP/1.1 200 OK
...
```

18. User Interaction Proxy → My Desktop (repeating request of step 2):

```
GET .../my-desktop HTTP/1.1
Host: ...
```

19. … (repeating request of step 3; intentionally left blank to stay in line with Fig. 7).

20. …

21. My Desktop → Location Tracker (repeating request of step 5):

```
GET .../location-tracker HTTP/1.1
Host: ...
```

22. Location Tracker → OAuth Proxy (repeating request of step 6):

```
GET .../location-source?id=user1234 HTTP/1.1
Host: ...
```

23. OAuth Proxy → Location Source (repeating request of step 7, but now with a valid access token included):

```
GET .../location-source?id=user1234 HTTP/1.1
Host: ...
Authorization: Token realm="..." token="token56789"
...
```

24. Location Source → OAuth Proxy (return geo-location coordinates):

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: ...
Cache-Control: no-store
```

```
               {
                 "lat": 57.502098,
                 "lng": 18.057159,
               }
        25.    OAuth Proxy → Location Tracker:
               HTTP/1.1 200 OK
               Content-Type: application/json
               Content-Length: ...
               Cache-Control: no-store

               {
                 "lat": 57.502098,
                 "lng": 18.057159,
               }
        26.    Location Tracker → My Desktop (return map with
          user’s location indicated):
               HTTP/1.1 200 OK
               ...
        27.    My Desktop → User Interaction Proxy (return
          desktop with map embedded):
               HTTP/1.1 200 OK
               ...
        28.    User Interaction Proxy → Browser:
               HTTP/1.1 200 OK
               ...
```

C) Asynchronous variant with transaction Id:

This variant of some embodiments of the invention avoids the “thread pool exhaustion” problem (as variant B does) and does not need a special response message ("retry") as according to variant B. Furthermore, an unambiguous correlation between the repeated request and the user information may be reached.

Steps 1. to 8. are the same as those of Fig. 7. However, the requests in steps 2. to 6. comprise a transaction Id generated by the UIP and unique in the realm of the UIP. When the authentication proxy (Service 4) receives the exception message (8.), it directly contacts the UIP (9.), whose address is known as in the synchronous variant, and passes a user interface and the transaction ID to the UID. Only then, Service 4 returns an error message to Service 3 (10.), which is propagated back to the UIP (11., 12.). The error message may be a conventional one.

The UIP receives both the message from the authentication proxy and the error message and may correlate them by the transaction Id. Thus, it knows that a user interaction is required, requests the user input from the browser as in variants A and B (13., 14.), and sends it to the authentication proxy in response to its message comprising the user interface and transaction Id (15.).

Then, it repeats the original request to Service 1 (the request of step 2.) including the same transaction Id (16.-20.).

When the authentication proxy receives both the user information and the repeated request from the UIP through the Service chain, it may correlate them based on the transaction Id. Then, Service 4 requests again the resource from Service 5 using the user input (21.). Service 5 provides the requested resource to Service 4 (22.). Thereafter, Services 4, 3, and 1 propagate the respective resources back to the browser (via the UIP) in the usual way (23.-26.).

In this variant, neither Service 1 nor Service 2 or Service 3 is aware that an interaction with the user is required and ongoing. In particular, the user information is not passed through Services 1 to 3, avoiding a potential security threat. In addition, the open threads of the first request chain (steps 2. to 7.) are closed within the usual response time for webservices, i.e. milliseconds, such that the servers are not blocked.

During its lifetime, the transaction Id should be unique in the realm of the UIP. In order to make the transaction Id unique in the realm of the authentication proxy, too, which is preferable in order to achieve an unambiguous correlation of the repeated request and the user information, the transaction Id may preferably comprise an identification of the UIP which is unique in the network, such as a MAC address.

In some embodiments of variant C, it may be preferred that the user information may be used for several requests of the same user. If user identification is provided with the requests, this may be achieved by the same correlation as in variant B. Alternatively, it may be achieved in that the correlation of the first request using user information is correlated based on the transaction Id. Then, a user identification of this request is retrieved and associated to the user information. Subsequent requests of the same user may then be correlated to the user information during its lifetime.

Adaptation to AJAX-based Web applications

An option of insertion of the UI according to some embodiments of the invention, as described in the previous sections (messages 10-11 in Fig. 6 and 14-15 in Fig. 7) is described in detail hereinafter. Many modern Web UIs are based on AJAX (Asynchronous JavaScript and XML; http://en.wikipedia.org/wiki/Ajax (programming)), which is "a group of interrelated web development techniques used on the client-side to create interactive web applications. With Ajax, web applications can retrieve data from the server asynchronously in the background without interfering with the display and behavior of the existing page. The use of Ajax techniques has led to an increase in interactive or dynamic interfaces on web pages. Data is usually retrieved using the XMLHttpRequest object".

For AJAX-based Web UIs, the insertion of the UI may be done in the following way. The Web UI keeps an HTTP connection continuously open to the User Interaction Proxy, corresponding to message 1 in Fig. 6 or Fig. 7). Then whenever the User Interaction Proxy is requested to insert a UI, it does it in response to the pending HTTP request. To avoid the "thread pool exhaustion" problem, a preferable solution may be using a polling technique: instead of keeping the HTTP connection continuously open, the client periodically checks for a potential UI request.

Fig. 9 shows an apparatus according to an embodiment of the invention. The apparatus may be an authentication proxy, such as Service 4 of Figs. 6 to 8. Fig. 10 shows a method according to an embodiment of the invention. The apparatus according to Fig. 9 may perform the method of Fig. 10 but is not limited to this method. The method of Fig. 10 may be performed by the apparatus of Fig. 9 but is not limited to being performed by this apparatus.

The apparatus comprises a first receiving means 10, a detecting means 20, a requesting means 30, and a second receiving means 40.

According to step S10 which may be performed by the first receiving means 10, a service request is received from a first party such as Service 3 of Figs. 6 to 8. According to step S20, which may be performed by detecting means 20, it is detected if a user interaction is required in order to comply with said service request. An example for such detection is the detection of the error message from Service 5 in Figs. 6 to 8. More in detail, the need for user interaction may be a need for a user related information which may be possibly obtained by user interaction.

If no user interaction is needed, the method ends (step S25).

If user interaction is required, the requesting means 30 may request user information from a second party such as the user interaction proxy of Figs. 6 to 8 (step S30). The second party is different from the first party. The request may comprise e.g. a user interface.

In step S40, which may be performed by the second receiving means 40, the user interaction (more detailed: the user related information) is received from the second party.

Fig. 11 shows an apparatus according to an embodiment of the invention. The apparatus may be a user interaction proxy, such as the one of Figs. 6 to 8. Fig. 12 shows a method according to an embodiment of the invention. The apparatus according to Fig. 11 may perform the method of Fig. 12 but is not limited to this method. The method of Fig. 12 may be performed by the apparatus of Fig. 11 but is not limited to being performed by this apparatus.

The apparatus comprises a sending means 60, a first receiving means 70, a requesting means 80, a second receiving means 90, and a responding means 100.
According to step S60 which may be performed by the resource requesting means 60, a service request is sent to a first device such as Service 1 of Figs. 6 to 8.

According to step S70, which may be performed by the first receiving means 70, a request for user interaction may be received from a second device in order to comply with the service request sent according to step S60. The second device is different from the first device.

Upon having received the request, the user interaction is requested from a user interaction device such as the browser of Figs. 6 to 8 (step S80). Step S80 may be performed by requesting means 80.

According to step S90, which may be performed by the second receiving means 90, the user interaction is received in response to the request. The responding means 100 provides a response to the second device (step S100). The response is based on the received user interaction.

Fig. 13 shows an apparatus according to an embodiment of the invention. The apparatus may be a resource server, such as Service 1, Service 2, and Service 3 of Figs. 6 and 7. Fig. 14 shows a method according to an embodiment of the invention. The apparatus according to Fig. 13 may perform the method of Fig. 14 but is not limited to this method. The method of Fig. 14 may be performed by the apparatus of Fig. 13 but is not limited to being performed by this apparatus.

The apparatus comprises a first receiving means 110, a requesting means 120, a second receiving means 130, a detecting means 140, and a responding means 150.

According to step S110 which may be performed by the first receiving means 110, a service request is received from a first party. In order to fulfill the service request, according to step S120 which may be performed by requesting

means 120, a resource is requested from a second party. The second party may be different from the first party.
The second receiving means 130 may receive, according to step S130, a response to the request for the resource.

In step S140, the detecting means 140 detects if, the response to the request for the resource comprises a retry message. In some embodiments, the apparatus may close the thread related to the request for the resource upon receipt of the retry message. Furthermore, the detecting means 140 checks in step S140 whether the retry message includes a handle.

If no retry message is detected or the retry message does not comprise the handle, the method is terminated (step S145).

Otherwise, in response to the service request, a retry message comprising the handle is provided to the first party (S150). This step may be performed by the responding means 150. In some embodiments, the thread related to the service request may be closed upon responding with the retry message.

Fig. 15 shows a system according to an embodiment of the invention. The system comprises a first resource apparatus 210 such as Service 4 in Figs. 6 to 8 and the apparatus shown in Fig. 9, and a user interaction apparatus 200 such as the user interaction proxy in Figs. 6 to 8 and the apparatus shown in Fig. 11.

The user interaction apparatus 200 provides a user interaction to the first resource apparatus 210 in response to a corresponding request from the first resource apparatus 210.

The system according to an embodiment of the invention shown in Fig. 16 corresponds to that of Fig. 15. That is, it comprises a first resource apparatus 210 such as Service 4 in Figs. 6 to 8 and the apparatus shown in Fig. 9, and a user interaction apparatus 200 such as the user interaction proxy in Figs. 6 to 8 and the apparatus shown in Fig. 11. Furthermore, it comprises a second resource apparatus 230. The second resource apparatus may comprise one or more resources or services, such as Services 1 to 3 in Figs. 6 to 8.

In the system of Fig. 16, a service request corresponding to a request for a resource is passed from the user interaction apparatus 200 via the second resource apparatus 220 to the first resource apparatus 210. The user interaction apparatus 200 provides a user information to the first resource apparatus 210 in response to a request for a user interaction received from the first resource apparatus 210.

The system of Fig. 16 corresponds e.g. to the systems shown in Figs. 6 to 8.

The leap from variant A to variant B (or variant C) may be interpreted as „trading execution thread for retry+handle (transaction ID)”, or in other words „externalizing” the (internal) execution state (call stack) to „ordinary” data i.e. the handle (transaction Id) and the data associated with the handle (transaction Id) at Service 4 and at the User Interaction Proxy.

In some embodiments, based on this consideration, a Web Services choreography may be employed: In these embodiments, there is a set of cooperating services, one invocing the other. Whenever a service “gets stuck”, i.e. does not know how to handle a situation (e.g. because an information is missing), it returns a "retry" response with a handle in it. The service receiving the "retry" response either knows how to deal with it, or not. In the latter case, it blindly propagates the response back to its caller. In the former case, it handles the case and then helps the "stuck" service resume its computation, as outlined in detail for the case of missing user information according to variant B herein.

One class of possible use cases – besides user interaction – is where a service "deep inside" the service web requires some additional context information. For example, it may miss an assertion from a trusted source that the user has enough credit for the requested service to be performed (a composite service may involve sending an SMS). Another example is where the service expects the operation to be performed inside a transaction environment, but there is no such. Such use cases are expected to arise more and more frequently as end-users will be able to compose new services from already available service elements or enablers, or other users' services, which is one key driver of the SERVERY project.

Some embodiments are described with respect to a http protocol. However, in other embodiments other protocols may be used as long as they are request-response protocols, where "a requestor sends a request message to a replier system which receives and processes the request, ultimately returning a message in response" (http://en.wikipedia.org/wiki/Request-response).

In the embodiments of Figs. 6 to 8, Service 4 requests a user interaction from the user interaction proxy by providing a user interface. In some embodiments, the request may comprise a reference to a user interface or another information element for provisioning a user interface. An example is given in the message flow outlined with respect to Fig. 7 above: In step 13 of the message flow, the HTTP response (200) contains another HTTP response (302) in its body. This latter HTTP response is an implementation of an information element for provisioning the user interface (UI). Thus, the returned content is not the UI itself but an HTTP redirect message which, after the UIP propagates it to the browser, will redirect the browser to Service 5 which in turn will return the UI itself (i.e. the question: "do you authorize [Service 3] to access your data?").

Instead or in addition of triggering the request for authentication by an exception message as according to the embodiments above, other conditions may trigger a workflow according to embodiments of the invention: For example, Service 4 may comprise an indication that Service 5 requires user interaction. Thus, it may not need to request the resource from Service 5 but immediately take care of obtaining the user information, and only then request the resource from Service 5 using the obtained user information.

The user information passed from the UID to the authentication proxy may have an infinite or predefined finite lifetime, and/or may be usable for a predefined number of requests.

If not otherwise stated or otherwise made clear from the context, the statement that two entities are different means that they are differently addressed in the communication network. It does not necessarily mean that they are based on different hardware. That is, each of the entities described in the present description may be based on a different hardware, or some or all of the entities may be based on the same hardware.

According to the above description, it should thus be apparent that exemplary embodiments of the present invention provide, for example an authentication proxy such as an OAuth proxy, or a component thereof, an apparatus embodying the same, a method for controlling and/or operating the same, and computer program(s) controlling and/or operating the same as well as mediums carrying such computer program(s) and forming computer program product(s). Further exemplary embodiments of the present invention provide, for example a user interaction proxy, or a component thereof, an apparatus embodying the same, a method for controlling and/or operating the same, and computer program(s) controlling and/or operating the same as well as mediums carrying such computer program(s) and forming computer program product(s) controlling and/or operating the same as well as mediums carrying such computer program(s) and forming computer program product(s). Still further exemplary embodiments of the present invention provide, for example a resource server, or a component thereof, an apparatus embodying the same, a method for controlling and/or operating the same, and computer program(s) controlling and/or operating the same as well as mediums carrying such computer program(s) and forming computer program product(s) controlling and/or operating the same as well as mediums carrying such computer program(s) and forming computer program product(s).

Implementations of any of the above described blocks, apparatuses, systems, techniques or methods include, as non limiting examples, implementations as hardware, software, firmware, special purpose circuits or logic, general purpose hardware or controller or other computing devices, or some combination thereof.

The proposed solution is capable of coping with user interaction triggered by any of the services inside a web of cooperating services. In embodiments according to variants A and C — unlike in the solutions listed in the prior art section — none of the other services notice that a user interaction is taking place, taking this burden off of the service developers' shoulder, hence simplifying service development. According to variant A, a price of this elegance is the "thread pool exhaustion" problem, according to variant C it is the requirement of transmitting the transaction Id. Embodiments of variants B and C counter the "thread pool exhaustion" problem; the price of this enhancement is that the services in the call chain must be able to cope with a special response type which we denoted by "retry" (variant B) or with the transmission of the transaction Id (variant C).

It is to be understood that what is described above is what is presently considered the preferred embodiments of the present invention. However, it should be noted that the description of the preferred embodiments is given by way of example only and that various modifications may be made without departing from the scope of the invention as defined by the appended claims.

Claims:

1. Apparatus, comprising

first receiving means for receiving a service request from a first party;

detecting means for detecting a requirement for user interaction in order to comply with said service request;

requesting means for requesting said user interaction from a second party different from the first party; and

second receiving means for receiving said user interaction as a response from said second party.

2. Apparatus according to claim 1, wherein

said request for user interaction comprises an information element for provisioning a user interface for executing said user interaction.

3. The apparatus according to any of claims 1 to 2, further comprising

resource requesting means for requesting a second resource from a resource device in order to comply with said service request; and wherein

the detecting means is adapted to detect the requirement for user interaction based on an exception information received from the resource device in response to the request for the resource.

4. The apparatus according to any of claims 1 to 3, wherein the user interaction is requested from a user who has caused the service request received from the first party.

5. The apparatus according to any of claims 1 to 4, wherein said request for user interaction comprises an identification of said service request.

6. The apparatus according to any of claims 1 to 5, wherein

the requesting means is adapted to provide, together with said request for said user interaction, a transaction identifier, wherein the transaction identifier is comprised in the service request received from the first party.

7. The apparatus according to any of claims 1 to 6, further comprising

response means for responding to the first party, wherein the response comprises a trigger for terminating service execution, compliant to the service request, in the first party.

8. The apparatus according to any of claims 1 to 4, further comprising

handle generating means for generating a handle which is related to the service request received from the first party, and wherein the handle is unique in the realm of the apparatus;

handle detecting means for detecting if a handle message comprising the handle is received by the apparatus from the second party; and wherein

the requesting means is adapted to request the user interaction if the handle message is detected.

9. The apparatus according to claim 8, further comprising

identifying means for identifying the second party based on the handle message.

10. The apparatus according to any of claims 8 to 9, wherein

the service request comprises a first user identification, the apparatus further comprising

correlating means for correlating a second service request from the first party with the received user interaction based on a second identification comprised in the second service request.

11. Resource access proxy comprising an apparatus according to any of claims 1 to 10.

12. Apparatus comprising

sending means for sending a service request to a first device;

first receiving means for receiving a request for user interaction from a second device in order to comply with said service request, wherein the second device is different from the first device;

requesting means for requesting said user interaction from a user interaction device;

second receiving means for receiving said user interaction from said user interaction device; and

responding means for responding, to said request for user interaction, based on said user interaction received from said user interaction device.

13. The apparatus according to claim 12, wherein

said service request comprises an identification of the device sending said service request.

14. The apparatus according to any of claims 12 to 13, further comprising

third receiving means for receiving the service request from said user interface.

15. The apparatus according to any of claims 12 to 14, wherein

said response to said request for user interaction is adapted to trigger a second sending of said service request to said first device.

16. The apparatus according to any of claims 12 to 15, further comprising

fourth receiving means for receiving a retry message comprising a handle in response to the service request;

handle message providing means for providing a handle message comprising the handle to the second device, and wherein

the second receiving means is adapted to receive the request for user interaction in response to the provision of the handle message.

17. The apparatus according to claim 16, further comprising

identifying means for identifying the second device based on the received retry message comprising the handle.

18. The apparatus according to any of claims 12 to 15, wherein

said request for user interaction comprises an identification of said service request.

19. The apparatus according to claim 18, further comprising

identifier generating means for generating an identifier for the service request, wherein said response to said request for user interaction comprises said identifier of said service request.

20. The apparatus according to claim 19, further comprising

fifth receiving means for receiving an error message in response to the service request;

correlating means for correlating the error message with the request for user interaction based on said identifier.

21. User interaction proxy comprising an apparatus according to any of claims 12 to 20.

22. Apparatus, comprising

first receiving means for receiving a service request from a first party;

requesting means for requesting, in order to fulfill the service request, a resource from a second party;

second receiving means for receiving a response to the request for the resource;

detecting means for detecting if the response comprises a first retry message comprising a handle and a trigger for terminating service execution, compliant to the request for the resource;

responding means for responding to the service request by a second retry message comprising the handle and a trigger for terminating service execution, compliant to the service request, if the detecting means detects that the response is the first retry message.

23. Apparatus, comprising

receiving means for receiving a service request from a first party;

requesting means for requesting, in order to fulfill the service request, a resource from a second party; wherein

the service request comprises an identifier; and

the request for the resource comprises the identifier.

24. Resource server, comprising an apparatus according to any of claims 22 and 23.

25. System, comprising

a first resource apparatus according to any of claims 1 to 10; and

a user interaction apparatus according to any of claims 12 to 20; wherein

the second party of the first resource apparatus comprises the user interaction apparatus;

the second device of the user interaction apparatus comprises the first resource apparatus;

the request for said user interaction of the first resource apparatus is the received request for user interaction of the user interaction apparatus; and

the response to said user request for user interaction of the user interaction apparatus is the user interaction received by the first resource apparatus.

26. System according to claim 25, further comprising

second resource apparatus according to any of claims 22 to 23; wherein

the first party of the first resource apparatus comprises the second resource apparatus;

the first device of the user interaction apparatus comprises the second resource apparatus;

the service request sent by the user interaction apparatus is the service request received by the second resource apparatus; and

the request for the resource of the second resource apparatus is the service request received by the first resource apparatus.

27. Method, comprising

receiving a service request from a first party;

detecting a requirement for user interaction in order to comply with said service request;

requesting said user interaction from a second party different from the first party; and

receiving said user interaction as a response from said second party.

28. Method according to claim 27, wherein

said request for user interaction comprises an information element for provisioning a user interface for executing said user interaction.

29. The method according to any of claims 27 to 28, further comprising

requesting a second resource from a resource device in order to comply with said service request; and wherein

the detecting is adapted to detect the requirement for user interaction based on an exception information received from the resource device in response to the request for the resource.

30. The method according to any of claims 27 to 29, wherein the user interaction is requested from a user who has caused the service request received from the first party.

31. The method according to any of claims 27 to 30, wherein said request for user interaction comprises an identification of said service request.

32. The method according to any of claims 27 to 31, wherein

the requesting is adapted to provide, together with said request for said user interaction, a transaction identifier, wherein the transaction identifier is comprised in the service request received from the first party.

33. The method according to any of claims 27 to 32, further comprising

responding to the first party, wherein the response comprises a trigger for terminating service execution, compliant to the service request, in the first party.

34. The method according to any of claims 27 to 30, further comprising

generating a handle which is related to the service request received from the first party, and wherein the handle is unique in the realm of an apparatus performing the method;

detecting if a handle message comprising the handle is received by the apparatus from the second party; and wherein the requesting is adapted to request the user interaction if the handle message is detected.

35. The method according to claim 34, further comprising
identifying the second party based on the handle message.

36. The method according to any of claims 34 to 35, wherein
the service request comprises a first user identification, the method further comprising
correlating a second service request from the first party with the received user interaction based on a second identification comprised in the second service request.

37. Method comprising
sending a service request to a first device;
receiving a request for user interaction from a second device in order to comply with said service request, wherein the second device is different from the first device;
requesting said user interaction from a user interaction device;
receiving said user interaction from said user interaction device; and
responding, to said request for user interaction, based on said user interaction received from said user interaction device.

38. The method according to claim 37, wherein
said service request comprises an identification of the device sending said service request.

39. The method according to any of claims 37 to 38, further comprising
receiving the service request from said user interface.

40. The method according to any of claims 37 to 39, wherein

said response to said request for user interaction triggers a second sending of said service request to said first device.

41. The method according to any of claims 37 to 40, further comprising

receiving a retry message comprising a handle in response to the service request;

providing a handle message comprising the handle to the second device, and wherein

the request for user interaction is received in response to the provision of the handle message.

42. The method according to claim 41, further comprising

identifying the second device based on the received retry message comprising the handle.

43. The method according to any of claims 37 to 40, wherein

said request for user interaction comprises an identification of said service request.

44. The method according to claim 43, further comprising

generating an identifier for the service request, wherein said response to said request for user interaction comprises said identifier of said service request.

45. The method according to claim 44, further comprising

receiving an error message in response to the service request;

correlating the error message with the request for user interaction based on said identifier.

46. Method, comprising

first receiving means for receiving a service request from a first party;

requesting means for requesting, in order to fulfill the service request, a resource from a second party;

second receiving means for receiving a response to the request for the resource;

detecting means for detecting if the response comprises a first retry message comprising a handle and a trigger for terminating service execution, compliant to the request for the resource;

responding means for responding to the service request by a second retry message comprising the handle and a trigger for terminating service execution, compliant to the service request, if the detecting means detects that the response is the first retry message.

47. Method, comprising

receiving means for receiving a service request from a first party;

requesting means for requesting, in order to fulfill the service request, a resource from a second party; wherein

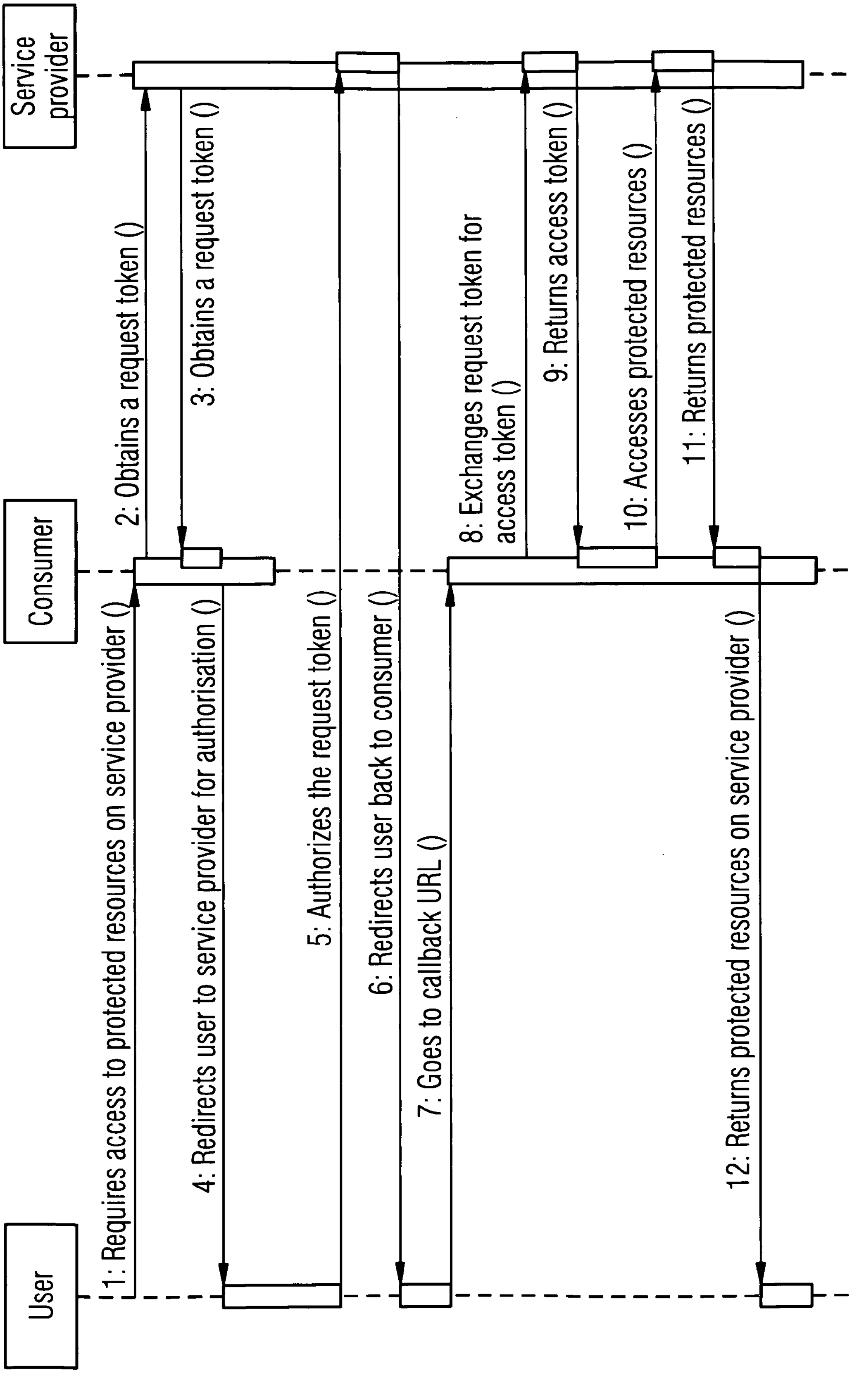
the service request comprises an identifier; and

the request for the resource comprises the identifier.

48. A computer program product including a program comprising software code portions being arranged, when run on a processor of an apparatus, to perform the method according to any one of claims 27 to 47.

49. The computer program product according to claim 48, wherein the computer program product comprises a computer-readable medium on which the software code portions are stored, and/or wherein the program is directly loadable into a memory of the processor.

FIG 1


User
Consumer
Service provider
1: Requires access to protected resources on service provider ()
2: Obtains a request token ()
3: Obtains a request token ()
4: Redirects user to service provider for authorisation ()
5: Authorizes the request token ()
6: Redirects user back to consumer ()
7: Goes to callback URL ()
8: Exchanges request token for access token ()
9: Returns access token ()
10: Accesses protected resources ()
11: Returns protected resources ()
12: Returns protected resources on service provider ()

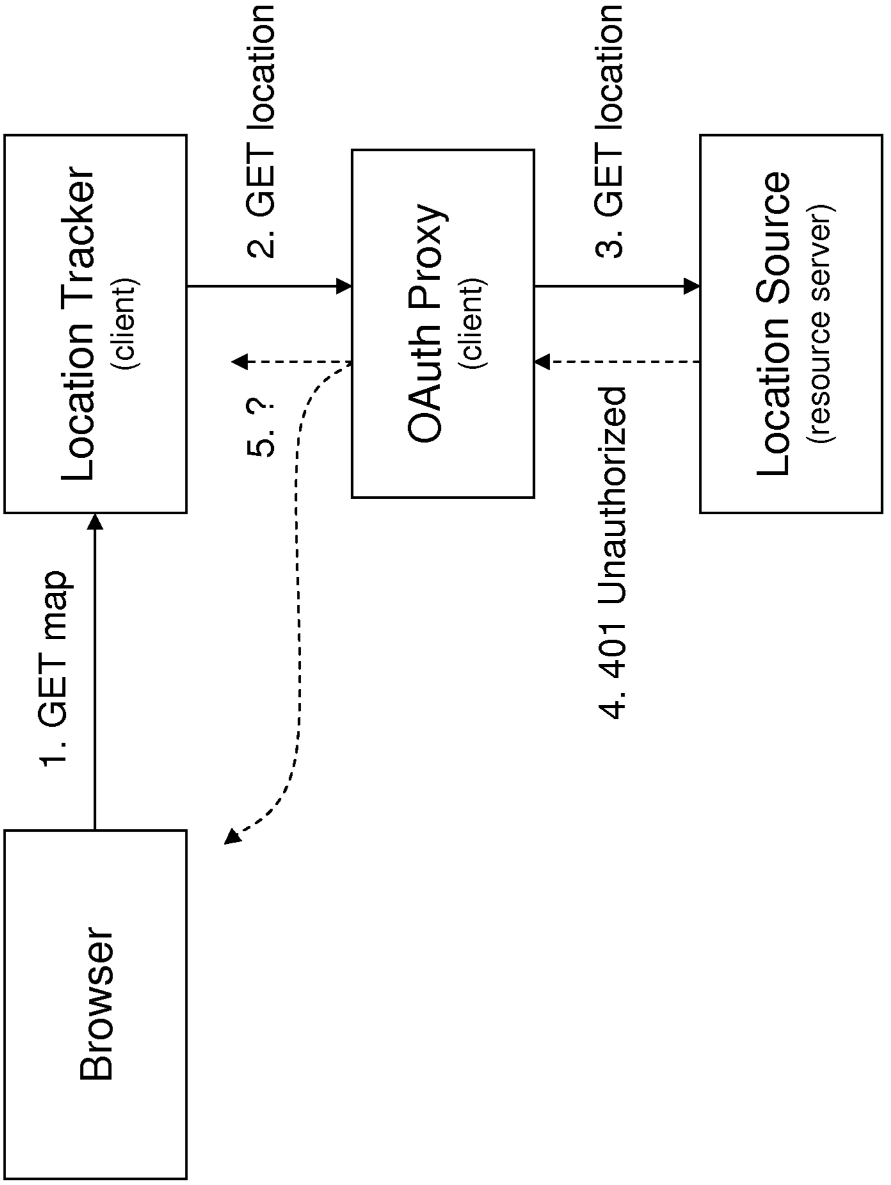

Browser
1. GET map
Location Tracker
(client)
2. GET location
5. ?
OAuth Proxy
(client)
3. GET location
4. 401 Unauthorized
Location Source
(resource server)

Fig. 2

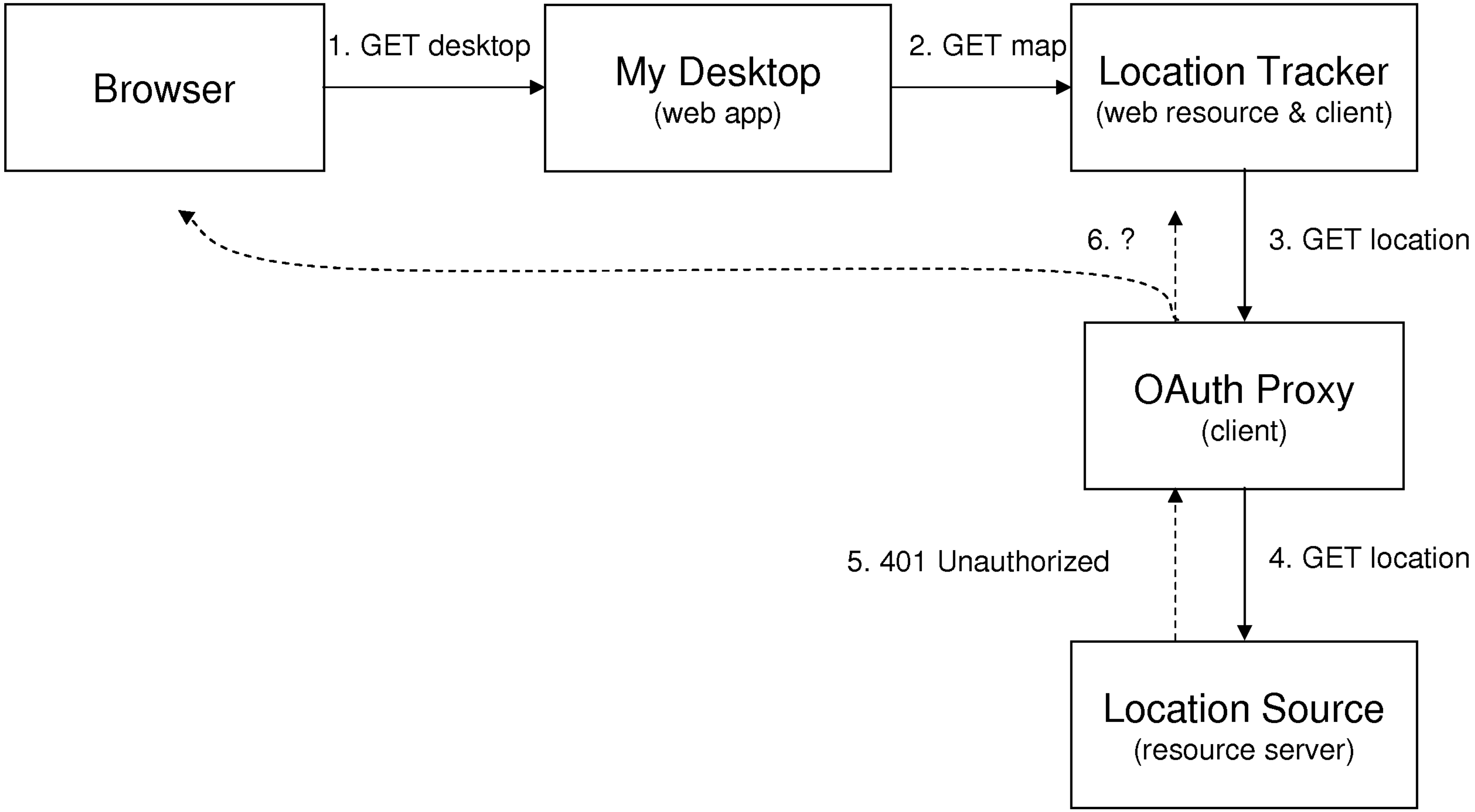
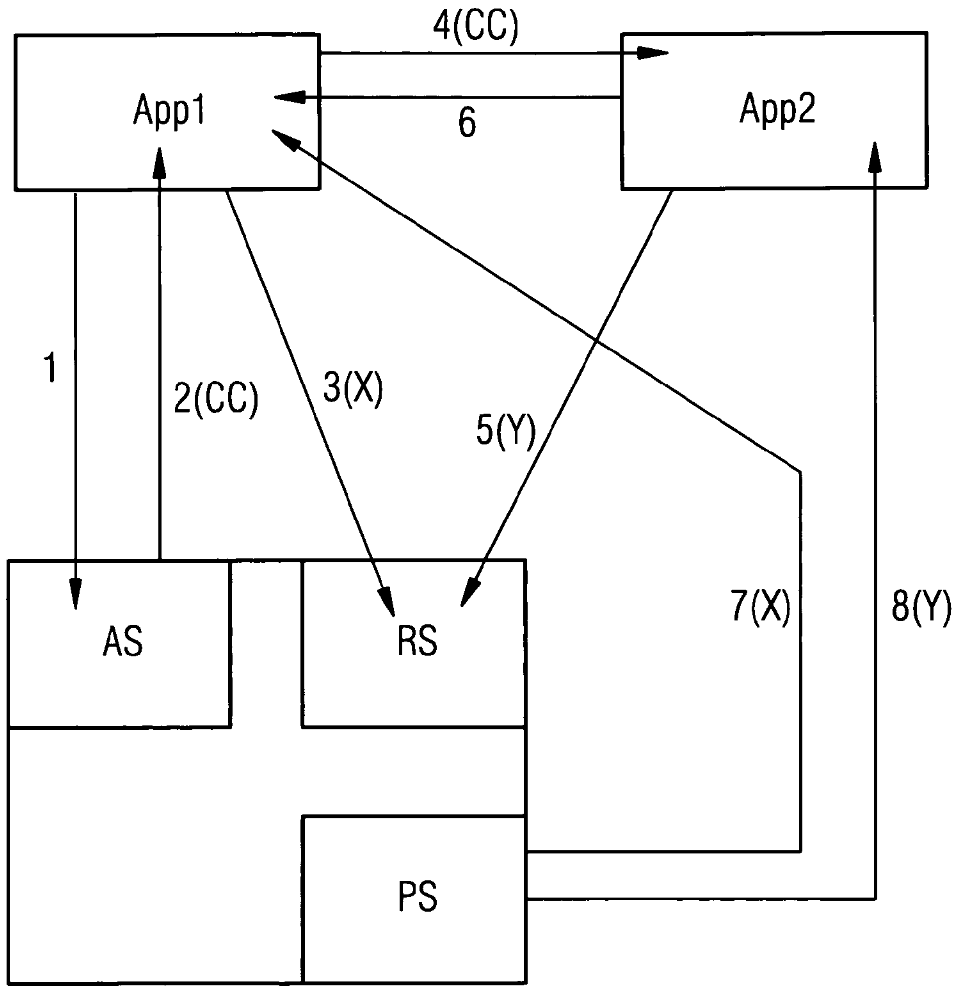

Browser
1. GET desktop
My Desktop
(web app)
2. GET map
Location Tracker
(web resource & client)
6. ?
3. GET location
OAuth Proxy
(client)
5. 401 Unauthorized
4. GET location
Location Source
(resource server)

Fig. 3

FIG 4



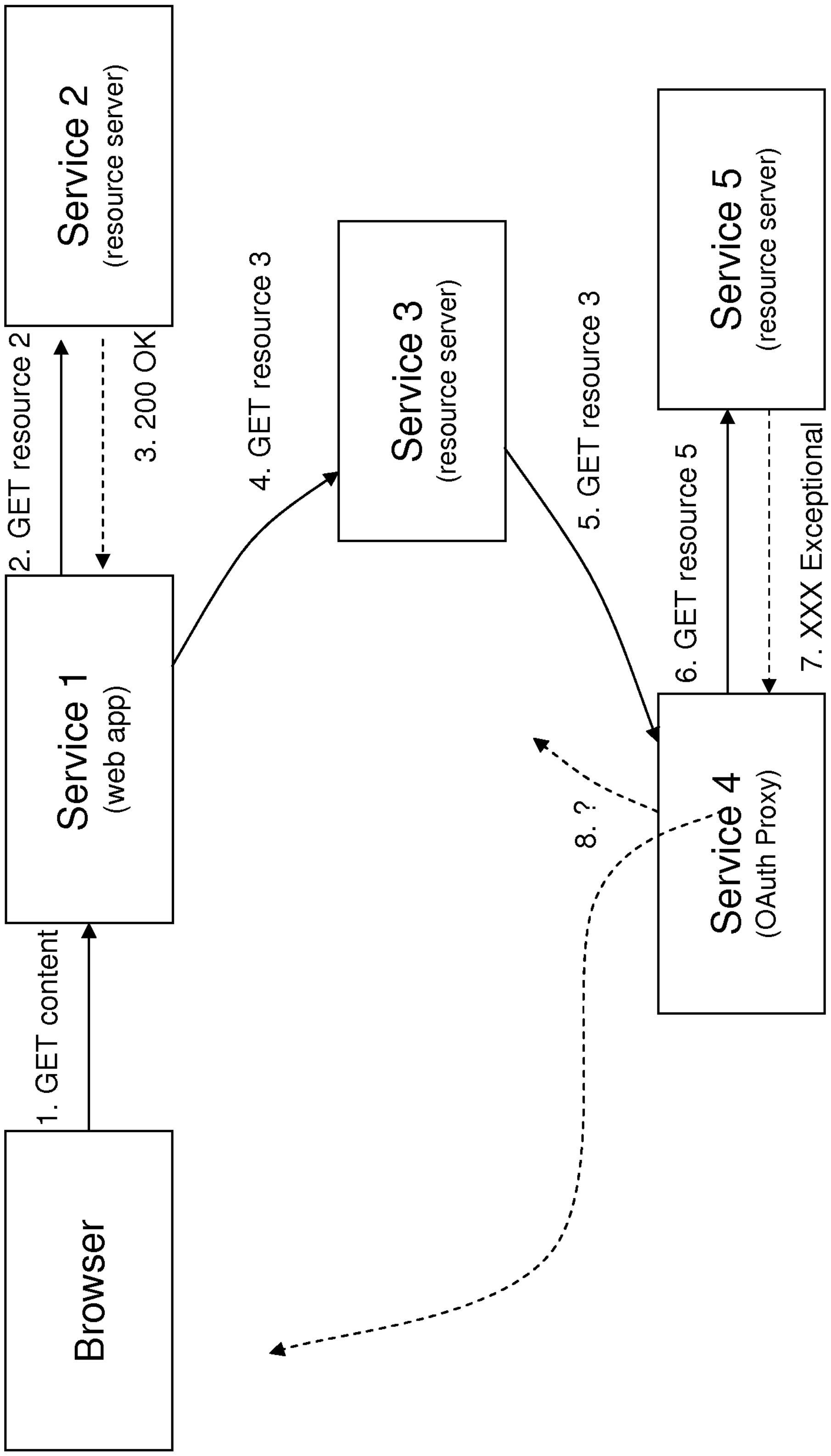

Browser
1. GET content
Service 1
(web app)
2. GET resource 2
3. 200 OK
Service 2
(resource server)
4. GET resource 3
Service 3
(resource server)
5. GET resource 3
Service 4
(OAuth Proxy)
6. GET resource 5
7. XXX Exceptional
Service 5
(resource server)
8. ?

Fig. 5

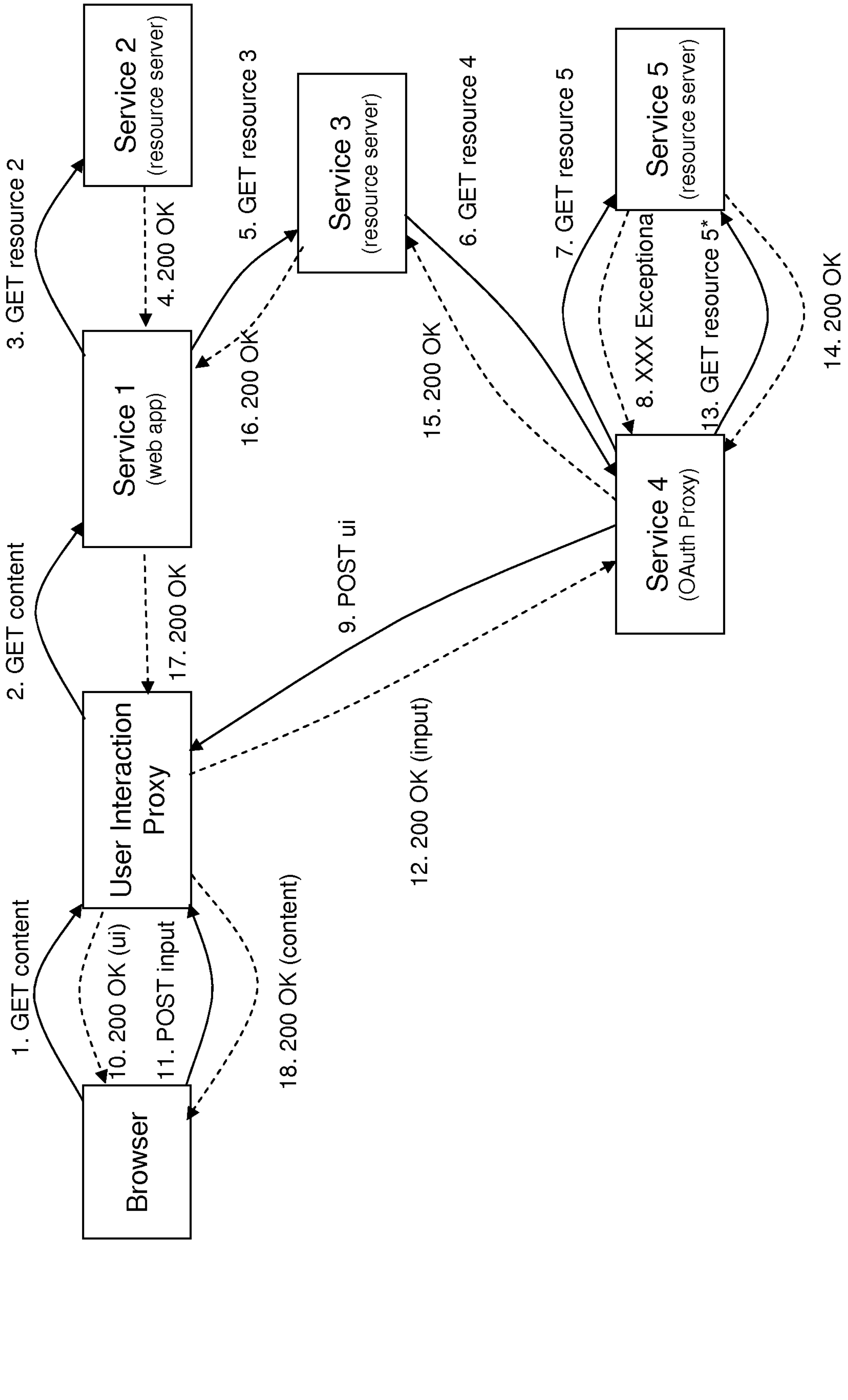

Browser
User Interaction Proxy
Service 1 (web app)
Service 2 (resource server)
Service 3 (resource server)
Service 4 (OAuth Proxy)
Service 5 (resource server)
1. GET content
2. GET content
3. GET resource 2
4. 200 OK
5. GET resource 3
6. GET resource 4
7. GET resource 5
8. XXX Exceptiona
9. POST ui
10. 200 OK (ui)
11. POST input
12. 200 OK (input)
13. GET resource 5*
14. 200 OK
15. 200 OK
16. 200 OK
17. 200 OK
18. 200 OK (content)

Fig. 6

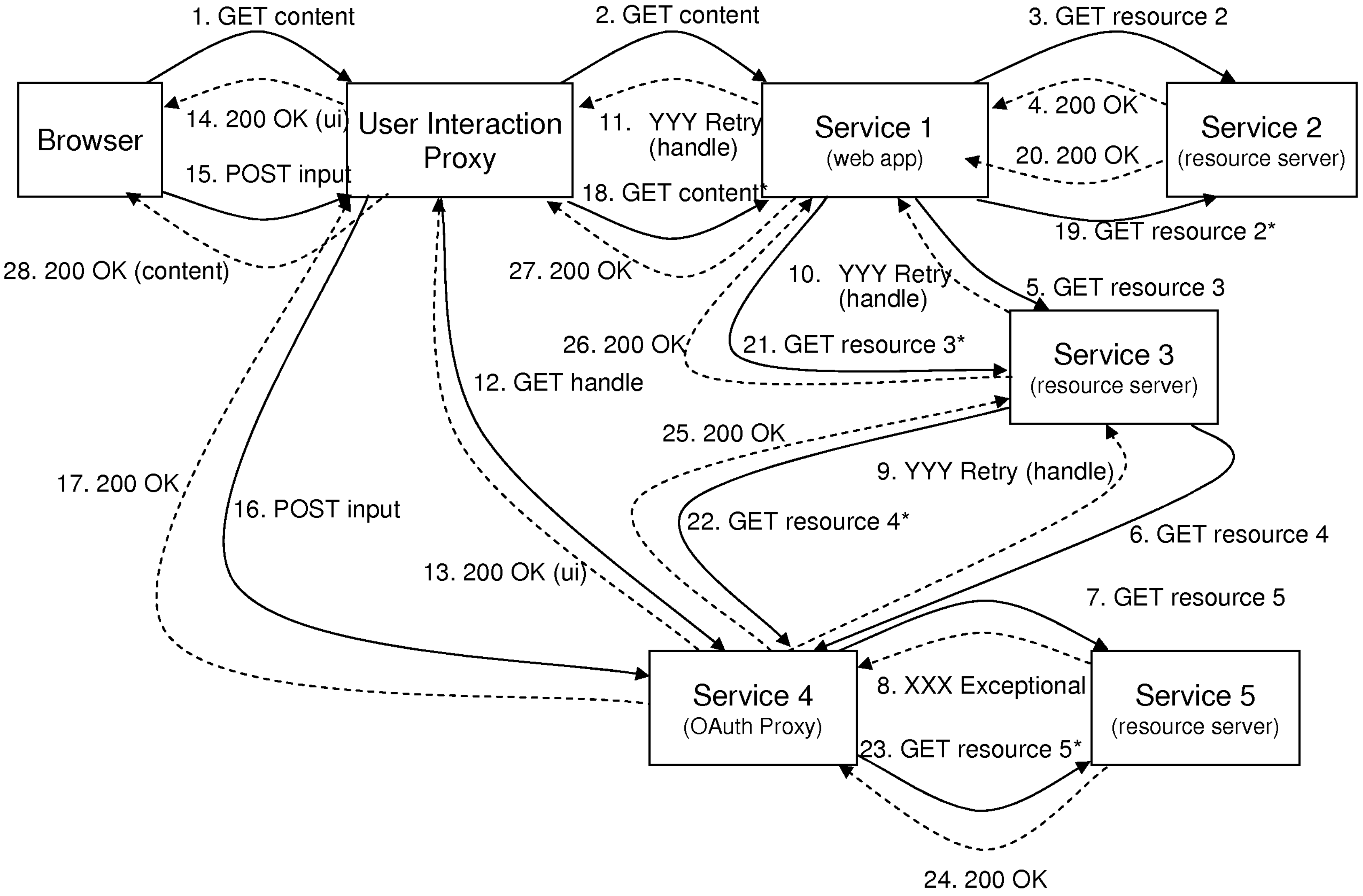

Browser
User Interaction Proxy
Service 1
(web app)
Service 2
(resource server)
Service 3
(resource server)
Service 4
(OAuth Proxy)
Service 5
(resource server)
1. GET content
2. GET content
3. GET resource 2
4. 200 OK
5. GET resource 3
6. GET resource 4
7. GET resource 5
8. XXX Exceptional
9. YYY Retry (handle)
10. YYY Retry (handle)
11. YYY Retry (handle)
12. GET handle
13. 200 OK (ui)
14. 200 OK (ui)
15. POST input
16. POST input
17. 200 OK
18. GET content*
19. GET resource 2*
20. 200 OK
21. GET resource 3*
22. GET resource 4*
23. GET resource 5*
24. 200 OK
25. 200 OK
26. 200 OK
27. 200 OK
28. 200 OK (content)

Fig. 7

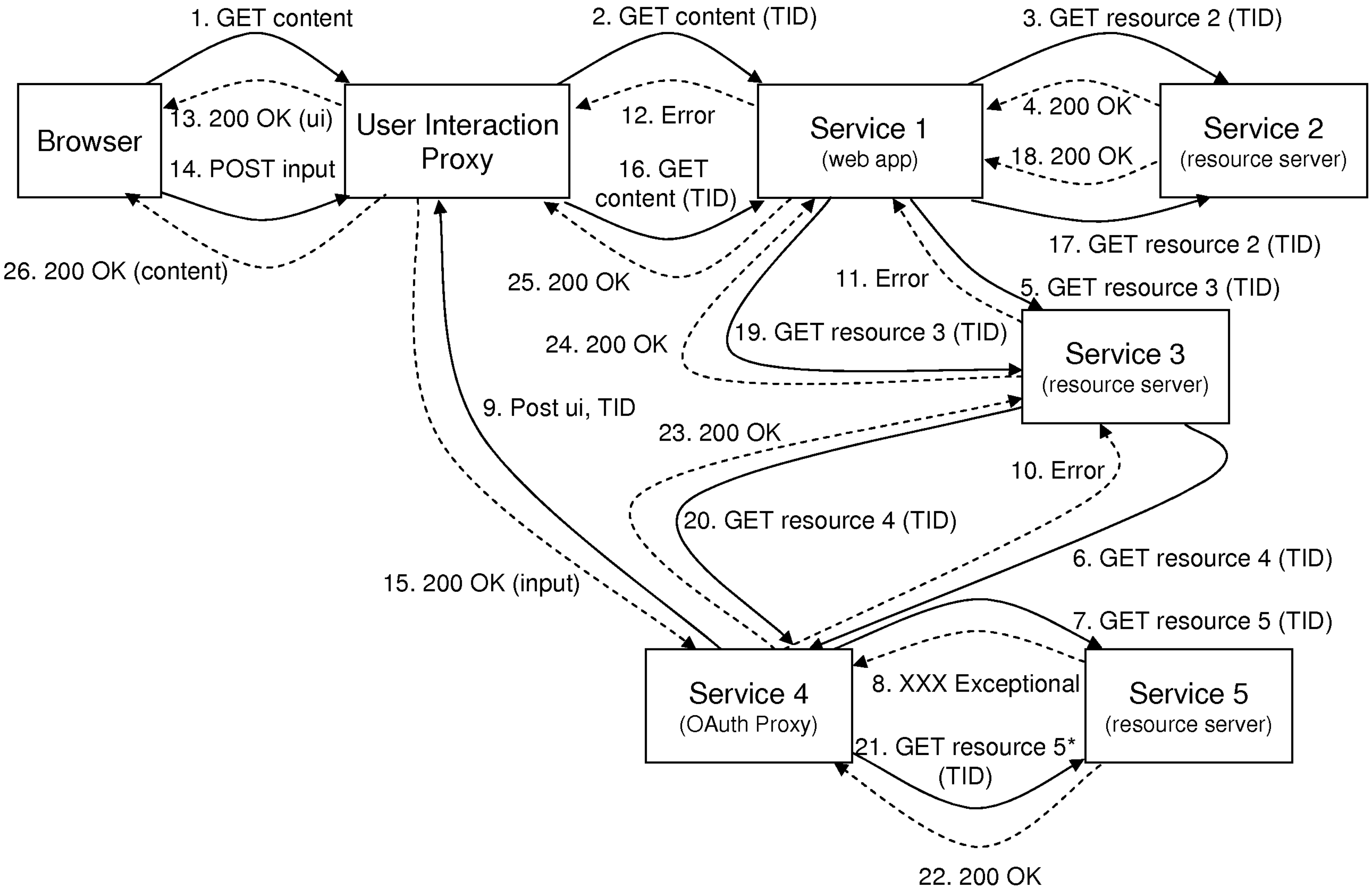

1. GET content
2. GET content (TID)
3. GET resource 2 (TID)
Browser
13. 200 OK (ui)
14. POST input
User Interaction Proxy
12. Error
16. GET content (TID)
Service 1
(web app)
4. 200 OK
18. 200 OK
Service 2
(resource server)
17. GET resource 2 (TID)
26. 200 OK (content)
25. 200 OK
11. Error
5. GET resource 3 (TID)
24. 200 OK
19. GET resource 3 (TID)
Service 3
(resource server)
9. Post ui, TID
23. 200 OK
10. Error
20. GET resource 4 (TID)
6. GET resource 4 (TID)
15. 200 OK (input)
7. GET resource 5 (TID)
Service 4
(OAuth Proxy)
8. XXX Exceptional
Service 5
(resource server)
21. GET resource 5*
(TID)
22. 200 OK

Fig. 8

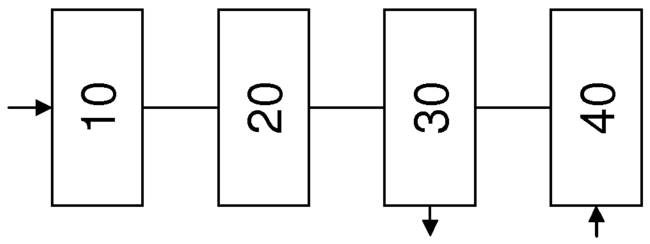

10
20
30
40

Fig. 9

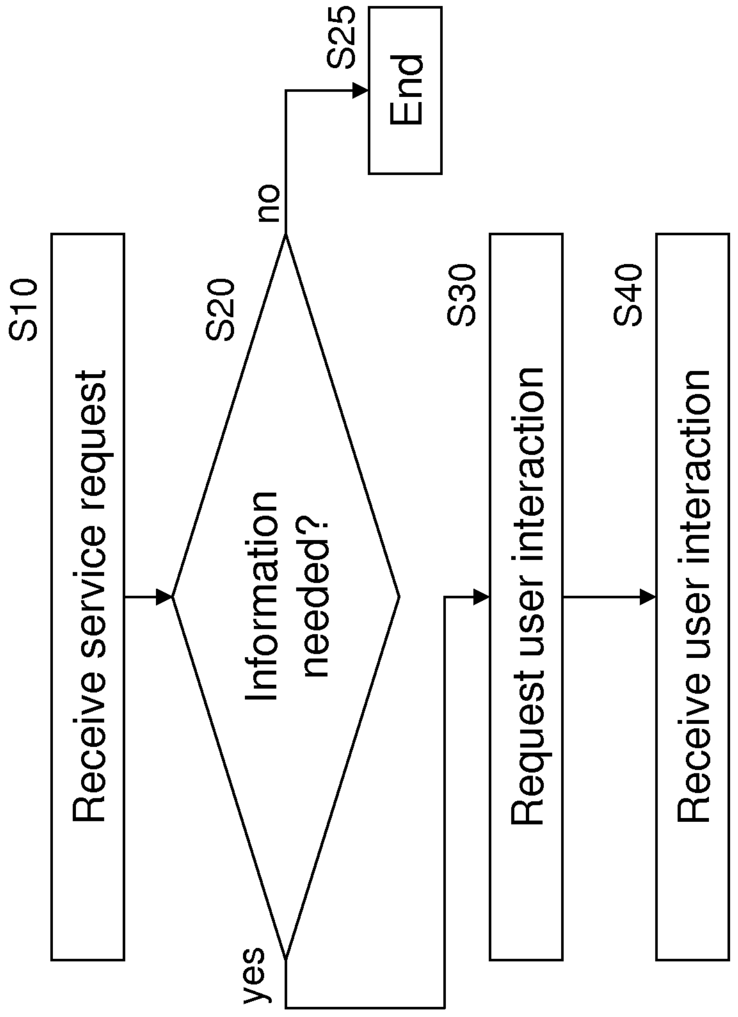

S10
Receive service request
S20
Information needed?
yes
no
S25
End
S30
Request user interaction
S40
Receive user interaction

Fig. 10

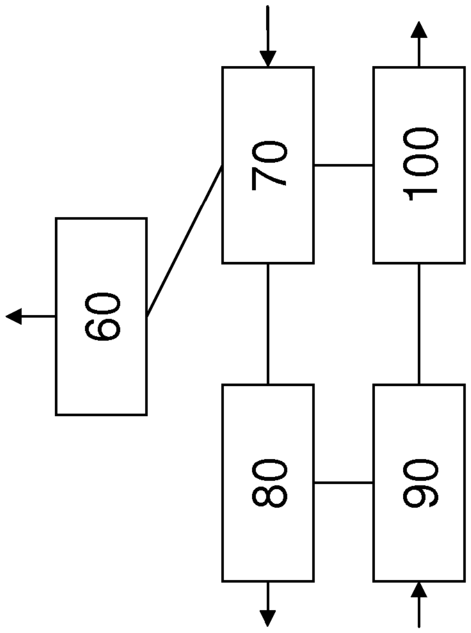

60
70
80
90
100

Fig. 11

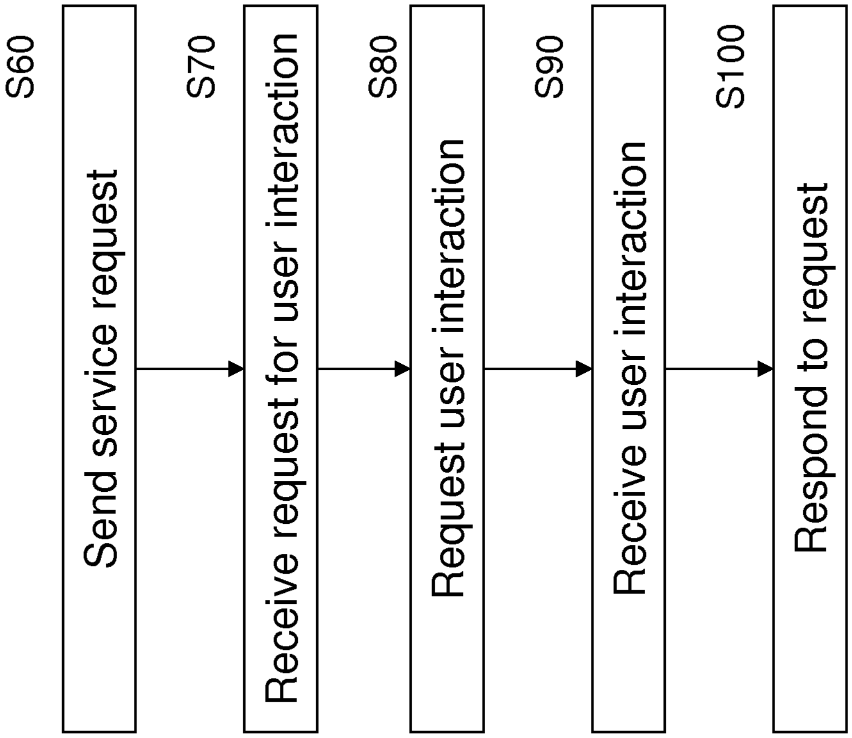

S60
Send service request
S70
Receive request for user interaction
S80
Request user interaction
S90
Receive user interaction
S100
Respond to request

Fig. 12

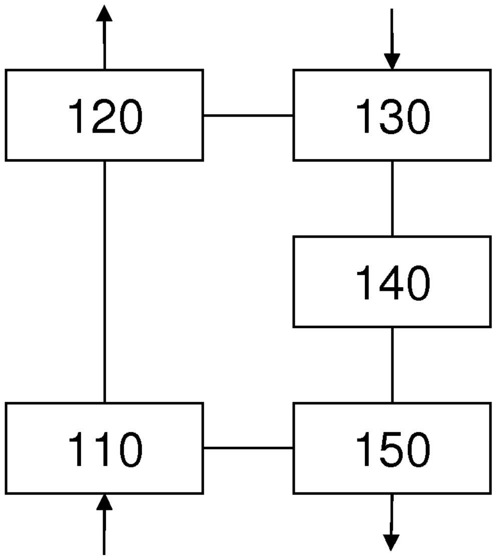

120
130
140
110
150

Fig. 13

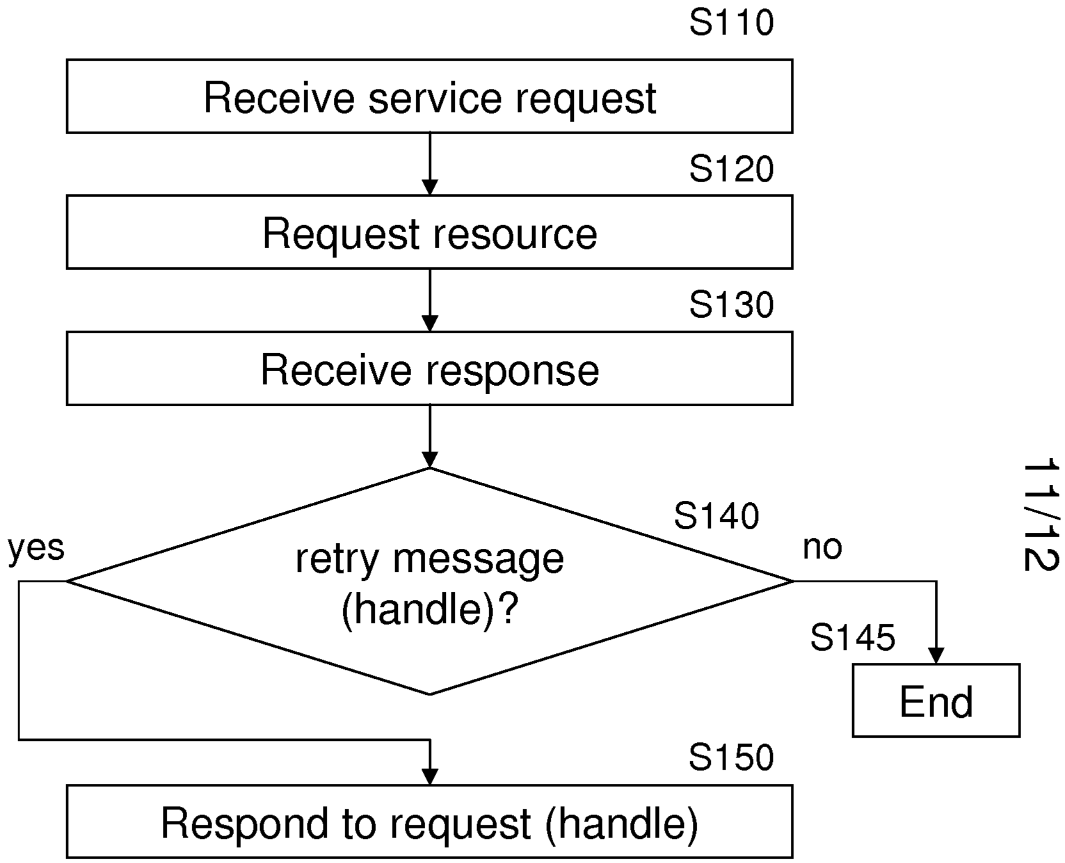

S110
Receive service request
S120
Request resource
S130
Receive response
S140
retry message (handle)?
yes
no
S145
End
S150
Respond to request (handle)

Fig. 14

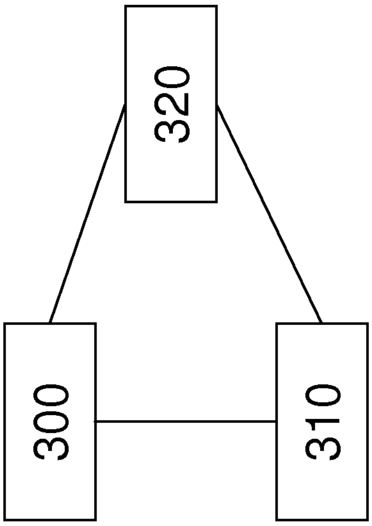


Fig. 16

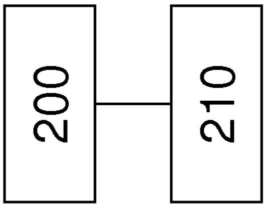


Fig. 15

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2010/070112

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L29/06 H04L29/08
ADD.

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L

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

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

EPO-Internal, WPI Data, COMPENDEX, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 03/073242 A1 (ERICSSON TELEFON AB L M [SE]; BARRIGA LUIS [SE]; PARDO-BLAZQUEZ AVELIN) 4 September 2003 (2003-09-04) paragraphs [0042] - [0048], [0062] figures 2,3B -----	1-21, 25-45, 47-49
X	TAKEDA Y ET AL: "Avoidance of Performance Bottlenecks Caused by HTTP Redirect in Identity Management Protocols", ACM, 2 PENN PLAZA, SUITE 701 - NEW YORK USA, 3 November 2006 (2006-11-03), XP040050485, section 4 figures 7-11 ----- -/--	1-21, 25-45, 47-49

☒ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

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

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

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search 25 October 2011	Date of mailing of the international search report 08/11/2011
Name and mailing address of the ISA/ European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Authorized officer Bengi, Kemal

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2010/070112

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 2010/094331 A1 (NOKIA SIEMENS NETWORKS OY [FI]; SEIDL ROBERT [DE]; MARTON GABOR [HU];) 26 August 2010 (2010-08-26) page 16, line 25 - page 20, line 26 figures 4,5 -----	1-21, 25-45, 47-49
X	EP 2 257 026 A1 (ALCATEL LUCENT [FR]) 1 December 2010 (2010-12-01) paragraphs [0045], [0049] figures 1,4,7 -----	1-21, 25-45, 47-49
X	US 2007/094401 A1 (GAGNE FRANCOIS [CA] ET AL) 26 April 2007 (2007-04-26) paragraphs [0027] - [0030] figures 1,2 -----	22-24,46
X	WANG BIN ET AL: "Open Identity Management Framework for SaaS Ecosystem", E-BUSINESS ENGINEERING, 2009. ICEBE '09. IEEE INTERNATIONAL CONFERENCE ON, IEEE, PISCATAWAY, NJ, USA, 21 October 2009 (2009-10-21), pages 512-517, XP031571864, ISBN: 978-0-7695-3842-6 sections III, IV.D figures 3,4,6 -----	22-24,46

INTERNATIONAL SEARCH REPORT

International application No.
PCT/EP2010/070112

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. □ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. □ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. □ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

see additional sheet

1. ☒ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. □ As all searchable claims could be searched without effort justifying an additional fees, this Authority did not invite payment of additional fees.

3. □ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. □ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

□ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

□ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☒ No protest accompanied the payment of additional search fees.

FURTHER INFORMATION CONTINUED FROM PCT/ISA/ 210

This International Searching Authority found multiple (groups of) inventions in this international application, as follows:

1. claims: 1-21, 25-45, 47-49

   Apparatuses and methods for synchronously calling a user interaction proxy in a Web Services system.

   ---

2. claims: 22-24, 46

   Apparatuses and methods for asynchronously calling a user interaction proxy in a Web Services system.

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/EP2010/070112

Patent document cited in search report		Publication date	Patent family member(s)			Publication date
WO 03073242	A1	04-09-2003	AU	2003212742	A1	09-09-2003
			EP	1497705	A1	19-01-2005
			HK	1080658	A1	05-11-2010
			JP	4579546	B2	10-11-2010
			JP	2005519365	A	30-06-2005
			US	2003163733	A1	28-08-2003
			US	2005154913	A1	14-07-2005
WO 2010094331	A1	26-08-2010	WO	2010094578	A1	26-08-2010
EP 2257026	A1	01-12-2010	WO	2010136323	A1	02-12-2010
US 2007094401	A1	26-04-2007	NONE			