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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
2 February 2017 (02.02.2017)

WIPO | PCT

(10) International Publication Number
WO 2017/019047 A1

(51) International Patent Classification:
G06F 3/048 (2006.01) *G06F 3/14* (2006.01)

(21) International Application Number:
PCT/US2015/042534

(22) International Filing Date:
28 July 2015 (28.07.2015)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT PACKARD ENTERPRISE DEVELOPMENT LP** [US/US]; 11445 Compaq Center Drive West, Houston, TX 77070 (US).

(72) Inventors: **CURLAND, Sebastian**; Shabazi Street No. 26, 56100 Yehud (IL). **FRIEMAN, Omer**; Shabazi 19, 56100 Yehud (IL). **MIZRAHI, Avigad**; Shabazi 19, 56100 Yehud (IL).

(74) Agents: **WOODS, Ariana** et al.; Hewlett Packard Enterprise, 3404 E. Harmony Road, Mail Stop 79 , Fort Collins, CO 80528 (US).

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

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

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

[Continued on next page]

(54) Title: COMPONENT PROFILE DISPLAY

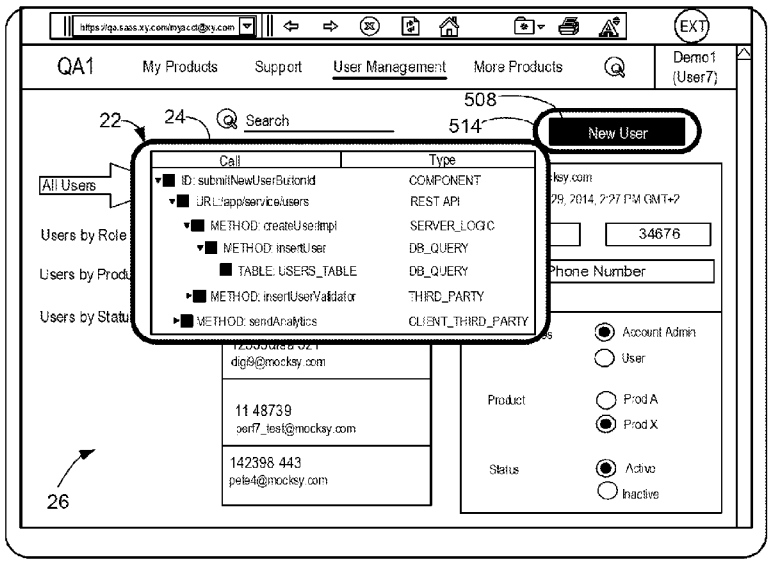

FIG. 6

(57) Abstract: A non-transitory machine-readable storage medium is described in which selection instructions receive a selection of a component in a graphical user interface of an application. Identification instructions associate the component with a component identifier, and query instructions identify, in an analysis database, at least one web service call linked to the component identifier. Profile display instructions cause a component profile to be displayed within the graphical user interface, with the component profile comprising the component identifier and the web service call.

WO 2017/019047 A1

Published:

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

COMPONENT PROFILE DISPLAY

BACKGROUND

[0001] When working with client-server applications, analyzing and understanding complete client-side and server-side code flows can be challenging and time-consuming. Full stack developers, QA personnel and other users who develop and test such applications may need to acquire a detailed understanding of the application's logic, such as methods utilized, databases accessed, third party servers called, etc. In some cases a developer or tester is required to manually review the code from client to server to understand each and every portion of the code that is relevant to her task. In some cases relevant code may be inaccessible.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] FIG. 1 is a block diagram of a system for displaying a component profile according to an example of the present disclosure.

[0003] FIG. 2 is a block diagram illustrating a system for displaying a component profile according to another example of the present disclosure.

[0004] FIG. 3 is an illustration of a client component map according to an example of the present disclosure.

[0005] FIG. 4 is an illustration of a web service map according to an example of the present disclosure.

[0006] FIG. 5 is an illustration of a graphical user interface according to an example of the present disclosure.

[0007] FIG. 6 is an illustration of the graphical user interface of FIG. 5 including a component profile according to an example of the present disclosure.

[0008] FIG. 7 is an illustration of the graphical user interface of FIG. 5 including a component profile according to another example of the present disclosure.

[0009] FIG. 8 is an illustration of the graphical user interface of FIG. 5 including a filtered component profile according to another example of the present disclosure.

[0010] FIG. 9 is a block diagram of a non-transitory machine-readable storage medium containing instructions according to an example of the present disclosure.

[0011] FIGS. 10A and 10B are a block diagram of a non-transitory machine-readable storage medium containing instructions according to another example of the present disclosure.

[0012] FIG. 11 is a flow chart of a method for displaying a component profile according to an example of the present disclosure.

[0013] FIG. 12 is a flow chart of a method for displaying a component profile according to an example of the present disclosure.

DETAILED DESCRIPTION

[0014] Analyzing and understanding both client-side code flows and server-side code flows for a client-server application can be a challenging and time-consuming endeavor. For example, a developer may be tasked with implementing a new user story in a client-server application that changes an application's behavior and creates modified client-side and server-side code flows. In one example, the user story may include adding a new "user phone" field to the application's graphical user interface (GUI), adding a server validation to validate a phone number inputted by a user, and saving the field in a database.

[0015] In this example, the developer needs to add the new phone field to a "new user' form in the client-side code of the application, find the location of the server validations via web services and add a new validation for the phone number, and find the database table name and the location (method) of the insert query to the database. Currently, exiting solutions do not provide a convenient way to aggregate and present all of this information to the developer in a useful format. Accordingly, the developer is left to separately scan and analyze the client-server application code and the relevant server code, and attempt to locate those particular web service calls, database queries, and other portions of the code flow that are relevant to the developer's task. In some cases, the developer may not have access to source code that is relevant to a particular component or function of interest. For example, where a third party

server call made by a web service is of interest, the source code of the third party server may not be available to the developer.

[0016] In other examples, quality assurance (QA) personnel may be tasked with creating a testing plan for a feature of a client-server application. The QA personnel may have a limited understanding of the feature they are testing and the logic underlying the feature. Thus, it may be difficult to create a testing plan that addresses the full functionality of the feature. Additionally, with just a superficial understanding of the relevant application logic, it may be challenging and sometimes impossible for QA personnel to determine the root cause of a failed test. As in the example described above, to obtain a complete understanding of the pertinent application logic, such QA personnel would need to manually scan and analyze both the client-side code and the server-side code.

[0017] In some existing solutions, code call graphs for only server-side code of an application may be generated. In other examples, code call graphs for only client-side code of an application may be generated. Such solutions do not support cross application code flows, i.e. client to server code flows. Thus, none of these solutions enable a user to see aggregated information of complete code flows of an application from client-side to server-side. Further, such solutions are unable to provide a developer or other user with a complete client-to-server code flow as viewed from the user interface of the actual application.

[0018] The present disclosure describes examples of a computing device, method and non-transitory machine-readable storage media for displaying a component profile within a graphical user interface (GUI) of an application. In one example and with reference to FIG. 1, a block diagram of a system 10 for displaying a component profile according to an example of the present disclosure is provided. As described in more detail below, the system 10 may comprise a computing device 14 that includes a processor 18 that executes instructions for displaying a component profile 22 within a graphical user interface (GUI) 26 of an application. In some examples the processor 18 may comprise a plurality of processors.

[0019] The processor 18 may include at least one physical device configured to execute at least one instruction. For example, the processor 18

may be configured to execute instructions that are stored on a non-transitory machine-readable storage medium. Such instructions may be part of at least one application, agent, module, service, program, routine, library, object, component, data structure, or other logical construct. Such instructions may be implemented to perform methods and functions described herein, or to perform a task, implement a data type, transform the state of at least one device, or otherwise arrive at a desired result.

[0020] Storage 30 may store instructions executable by the processor 18. As described in more detail below, in some examples storage 30 may include non-transitory machine-readable storage media such as removable media and/or built-in devices, optical memory devices such as CD, DVD, HD-DVD, Blu-Ray Disc, and the like, semiconductor memory devices such as RAM, EPROM, EEPROM, and the like, and/or magnetic memory devices such as hard disk drive, floppy disk drive, tape drive, MRAM, and the like, among others. Storage 40 may include memory devices with at least one of the following characteristics: volatile, nonvolatile, dynamic, static, read/write, read-only, random access, sequential access, location addressable, file addressable, and content addressable.

[0021] In some examples, the processor 18 and storage 30 may be components of at least one computing device, such as computing device 14. In some examples, such computing device may take the form of a server, network computing device, desktop computing device, and/or other suitable type of computing device.

[0022] Storage 30 may store instructions comprising a client-server application 38, with the application comprising code for generating GUI 26 and code for generating component 42. A web browser 44 may be configured to display the GUI 26 and the component profile 22 via client-server application 38. In other examples, another computing device may include a web browser that displays the GUI 26 of the client-server application 38 and component profile 22. In some examples the client-server application 38 may reside on another computing device, such as a server. As described in more detail below, the GUI 26 may include a component 42 which may be selectable via user input. Examples of selectable components include buttons, sliders, drop-down menus,

data input regions, etc. In some examples, selection of component 42 may trigger a web service call 50.

[0023] The processor 18 of computing device 14 may execute instructions to scan the client-server application 38 to identify at least one web service call 50 and at least one component 42 in the GUI 26 that calls the web service call. As described in more detail below, a client component map 56 that links a component identifier of the component 42 to the web service call 50 may be generated. The processor 18 also may execute instructions to scan server code 60 to identify at least one web service 64 that is called by the web service call 50. In some examples the server code 60 may be located on another computing device, such as a server. In other examples, server code 60 may be located in storage 30 of computing device 14. A web service map 68 comprising the web service call 50 may be generated. In some examples and as described in more detail below, the web service map 68 may include at least one sub-call from the web service call 50.

[0024] The processor 18 also may execute instructions to combine the client component map 56 and the web service map 68 into a component profile 22. The component profile 22 may be sent to the web browser 44 for display within the GUI 26 presented by the browser. As described and illustrated in the examples discussed below, upon selection of the component 42 in the GUI 26 by a user, the component profile 22 may be displayed in a manner visually linking the component profile to the component 42. In this manner, with a single click or selection of the component 42, a user such as a developer may conveniently visualize the corresponding component profile 22 in context with the component.

[0025] Further, such component profile 22 may comprise a combination of the client component map 56 and the web service map 68, beginning with the component identifier calling the web service call. In this manner, the component profile 42 may be conveniently displayed to the developer from the point of view of the corresponding component 42 that is of interest to the developer. Accordingly, the developer may be provided with a complete client-to-server code flow for a component of interest, with such information displayed within the GUI 26 of the application under development.

[0026] Turning now to FIGS. 2-7, example systems and use cases of displaying a component profile according to examples of the present disclosure are provided. FIG. 2 illustrates a block diagram showing one example of a system 200 for displaying a component profile according to an example of the present disclosure. In this example, a client computing device 204 may comprise a storage 208 and a processor 212 that executes instructions for displaying a component profile 22 within a GUI 26 of an application. In some examples the processor 212 may comprise a plurality of processors. The client computing device 204 may be communicatively coupled to a display 220 that may display the GUI 26.

[0027] The processor 212 may include at least one physical device configured to execute at least one instruction. For example, the processor 212 may be configured to execute instructions that are stored on a non-transitory machine-readable storage medium. Such instructions may be part of at least one application, agent, service, program, routine, library, object, component, data structure, or other logical construct. Such instructions may be implemented to perform methods and functions described herein, or to perform a task, implement a data type, transform the state of at least one device, or otherwise arrive at a desired result.

[0028] Storage 208 may store instructions executable by the processor 212. In some examples storage 208 may include non-transitory machine-readable storage media such as removable media and/or built-in devices, optical memory devices such as CD, DVD, HD-DVD, Blu-Ray Disc, and the like, semiconductor memory devices such as RAM, EPROM, EEPROM, and the like, and/or magnetic memory devices such as hard disk drive, floppy disk drive, tape drive, MRAM, and the like, among others. Storage 208 may include memory devices with at least one of the following characteristics: volatile, nonvolatile, dynamic, static, read/write, read-only, random access, sequential access, location addressable, file addressable, and content addressable.

[0029] In some examples, the processor 212 and storage 208 may be components of at least one computing device, such as client computing device 204. In other examples, such computing device may take the form of a network computing device, desktop computing device, and/or other suitable type of computing device.

[0030] As described above for the example computing device 14 of FIG. 1, storage 208 may store instructions comprising a client-server application 38, with the application comprising code for generating GUI 26 and code for generating at least one component 42, which may be a selectable component. A web browser 226 may be configured to display the GUI 26 via client-server application 38. As described in more detail below, a plug-in 230 to the browser may cause the component profile 22 to be displayed within the GUI 26.

[0031] With reference to FIG. 5, in one use case example the client-server application 38 may comprise a quality assurance (QA) application "QA1". The QA application may generate a GUI 26 that includes a plurality of user-selectable components, such as radio buttons 504, New User button 508, Search field 512, checkbox 516, selectable menu items 518, etc. A user of the QA application may desire to understand the logic underlying one or more of the selectable components in the GUI 26. As described in more detail below, when a component is selected by the user, a component profile comprising the component identifier and at least one web service call linked to the component may be generated and displayed within the GUI 26.

[0032] While the following description is provided with reference to an example QA application, the principles of the present disclosure may be utilized with any application that utilizes a client-server model, such as for example email applications, networking applications, gaming applications, etc.

[0033] With reference again to FIG. 2, to enable the generation of a component profile 22 for each component of the client-server application 38, storage 208 may further include an analysis module 234 containing instructions executable by the processor 212 to, in a first phase, scan the client-server application 38 to identify each web service that is called by the application. In some examples, the analysis module 234 may perform a static analysis of the client-server application files to identify all server calls to a web service. A web service may comprise a software system designed to support interoperable machine-to-machine interaction over a network. Examples of web services include, but are not limited to, REST-compliant web services that manipulate representations of web resources using a uniform set of stateless operations, and arbitrary web services, such as SOAP and WSDL-based web services, in which the service may expose an arbitrary set of operations.

[0034] In some examples, to identify each web service that is called by the application, the analysis module 234 may utilize a database of known server calls, such as REST-compliant calls, SOAP calls, WSDL calls, etc. With reference to an example client-server application 38 that may generate the GUI 26 shown in FIG. 5, the analysis module 234 may scan the client-server application code and find the following REST-compliant web service call that is used for creating a new user for the application:

```
Function saveNewUser (newUserObject) {
        $http.post("/app/service/users", newUserObject);
}
```

[0035] For each web service call that is identified, the analysis module 234 scans the client-server application 38 to find which GUI component(s) and code flow(s) generate this call to the web service. With reference to the example GUI 26 shown in FIG. 5, the analysis module 234 may scan the client-server application code and find the following selectable submit button (corresponding to New User button 508) in an "add new user" form which calls the above call:

```
<button id='submitNewUserButtonId' ng-click='saveNewUser();'
class='Btn__action'>
    Create
</button>
```

[0036] The analysis module 234 may determine that the component identifier of this component is 'submitNewUserButtonId'. With reference now to FIG. 3, the analysis module 234 may then generate and store a client component map 56 that includes the component identifier in the form of a client component ID (submitNewUserButtonId), the corresponding application (QA1), the web service call (/app/service/users), and the web service type or method (POST). Additional web service calls and corresponding component identifiers of other components also may be discovered, and corresponding additional client component maps 56 may be generated and stored.

[0037] In some examples, a selectable component in the GUI 26 may call a chain of functions that end with a call to the web service. For example, the New User button 508 may call function A, function A may call function B, and function B may call the 'saveNewUser' function discussed above.

[0038] With reference again to FIG. 2, in some examples the client component maps 56 generated by analysis module 234 may be stored in an analysis database 250 located in server 244 or in another storage or other computing device. The client computing device 204 may be communicatively coupled to the server 244 via a network, such as a local area network (LAN), wide area network (WAN), wired network, wireless network, or a combination thereof, and which may include the Internet. In other examples, the client component maps 56 may be stored in storage 208 of client computing device 204. In some examples, the analysis module 234 may be located in storage 240 of server 244 and executed by processor 250 to perform the functions described herein.

[0039] With continued reference to FIG. 2, the analysis module 234 may contain instructions executable by the processor 212 in a second phase to scan the server code 60 to identify web services 64. In some examples the analysis module 234 may identify web services 64 that are called by the components of the client-server application 38 that were found in the first phase discussed above. As described in more detail below, component profiles may be generated that map the client-to-server code flows from a selected component to the web service called by that component, and any sub-calls from the web service.

[0040] For each web service 64 that is located, the analysis module 234 may programmatically build a web service map 68 that may comprise a code call graph showing code flows and any sub-calls from the web service. For a particular web service 64 that is identified, the analysis module 234 may search for other web services and/or other related functions and methods called by the identified web service, such as database queries, third party calls, other server calls, etc. In some examples, such code call graphs may be generated via static analysis of the server code files using static analysis tools or platforms.

[0041] With reference to the example above in which the web service call /app/service/users was located in the client server application code, the analysis

module 234 may identify a web service (for example, /createUser) that is called by this web service call. Once the web service /createUser is identified, the analysis module 234 may identify additional information related to the web service, such as a corresponding method name and a category, and may search for sub-calls to other functions and methods called by such service.

[0042] With reference to FIG. 4, in this example the web service map 68 may comprise the web service (/createUser) called by the web service call app/service/users in the example above, and a database query and a third party server validation call made by the web service. As shown in this example, the web service method createUserImpl calls the insertUser method to make a database query. The web service method createUserImpl also makes a third party server call via the insertUserValidator method to a URL having the value /validationServer/validateNewUser to validate data related to a new user.

[0043] For example and with reference again to FIG. 2, a web service 64 hosted on server 244 may make a third party server call to another web service 260 hosted on a different server 264. In this manner, another function performed by web service 260 and utilized by web service 64 may be identified. In other examples, server 264 may call other third party server(s) in a server-to-server chain, and such code flow may be captured in the web service map 68. In other examples of web services, fewer or more server calls, data base queries, and other third party calls may be made. In this manner, the web service map 68 may start with the method createUserImpl and include all code flows from this starting point.

[0044] In some examples, the client component map 56 for the component identifier (in this example, submitNewUserButtonId) and the corresponding web service map 68 for the web service (/createUser) called by this component may be combined to create a component profile 22. In some examples, the component profile 22 comprises a call flowchart that begins with the component identifier calling the web service call for the web service, and includes any sub-calls from the web service. Additionally and as described in more detail below, the component profile 22 may be conveniently displayed within the GUI 26, thereby providing a user with a complete client-to-server code graph from a selected component in the GUI 26 to all server-side functions and methods related to that component.

[0045] As noted above, in some examples a plug-in 230 to the web browser 226 may cause the component profile 22 to be displayed within the GUI 26. In some examples, generation of the client component maps 56 of the client-server application 38, the web service maps 68 of the server code 60, and/or the component profile 22 may be performed prior to the plug-in 230 being executed in the web browser 226. For example, upon initial installation of the plug-in 230 and analysis module 234, the client-server application 38 and the server code 60 may be analyzed as described above, and corresponding client component maps 56 and web service maps 68 may be generated. In this example, when the plug-in 230 is enabled in web browser 226, the relevant client component maps 56 and the corresponding web service maps 68 have been previously generated and are available for constructing component profiles 22 of selected components in the GUI 26.

[0046] In some examples, component profiles 22 for each of the components in the client component maps 56 may be generated upon such installation or at another point prior to execution of the plug-in 230. In other examples, and as described in more detail below, upon a user selection of a component 42 in the GUI 26, a component profile 22 may be generated using the previously-generated client component maps 56 and web service maps 68.

[0047] With reference again to FIG. 5, an example use case of the present disclosure will be provided. In one example, a developer user of the QA1 application may be testing functionality associated with adding a new user via the New User button 508. The developer would like to understand how the New User button 508 behaves, and see the full client-to-server logic underlying the New User button 508, including all of the web services and other methods and functions associated with the button. In some examples, the developer may not have access to all of the source code associated with such web services, methods and functions, such as source code of the server code 60 and/or other web services and code located on other servers or in other third party storage.

[0048] In one example, the user may select the EXT button 520 to activate the plug-in 230 and enable the generation and display of component profiles 22 for components in the GUI 26. Once the plug-in 230 is activated, the user may select any selectable component in the GUI 26 to trigger the generation and display of a component profile 22 for that component.

[0049] In the present example, the user may select the New User button 508. Upon receiving the selection of the New User button 508, the plug-in 230 may associate this component with its component identifier, such as the client component ID (submitNewUserButtonId) from the client component map 56 of FIG. 3. With the component identifier of the New User button 508 identified, the analysis database 250 may be queried to identify the web service call(s) 64 that are linked to this component identifier. For example, the plug-in 230 may identify a web service map 68 for the web service call(s) 64 linked to the component identifier. As noted above, in some examples the plug-in 230 may combine the client component map 56 for the New User button 508 with the web service map 68 for the web service call(s) 64 linked to the component identifier to generate a component profile 22 for this component. In some examples, the component profile 22 may be generated at the server 244, stored in the analysis database 250 and received by the plug-in 230.

[0050] With reference now to FIG. 6, the plug-in 230 may cause the component profile 22 to be displayed within the GUI 26 of the QA1 application. In this example, the component profile 22 comprises a sequenced listing of all methods, functions and other calls related to the selected component (New User button 508). In some examples, the component profile 22 may comprise an ordered combination of information from the client component map 56 and the web service map 68 for the selected component. As shown in the component profile 22 of FIG. 6, the ordered combination may begin with the component ID and may illustrate the complete client-to-server code flow associated with this component. In this manner, the user can conveniently visualize the complete code flow for the selected component, with such flow presented from the point of view of the component.

[0051] In the example of FIG. 6, the component profile 22 includes the information from the client component map 56 for the submitNewUserButtonID component as discussed above, including the web service call to URL: /app/service/users. Following this web service call is the information from the web service map 68 for the web service called by /app/service/users, namely the Method createUserImpl. In this example, the component profile 22 includes the database query and third party validation sub-calls identified in the web service map 68 discussed above. In this example, the component profile 22

also includes a sendAnalytics third party call, which may have been identified. Accordingly, the component profile 22 presents a complete client-to-server, call-by-call code graph of the logic underlying the selected component.

[0052] As shown in the example of FIG. 6, this example of a component profile 22 includes two columns of information: a Call column identifying the particular call, and a Type column identifying a type of the call. In some examples of component profiles, the Type column may not be displayed. In other examples of component profiles, additional column(s) of information from the web service map 68 may be displayed in addition to the Type column. For example, a component profile 22 may include the Call column, the Type column and a Value column from a corresponding web service map 68. In still other examples, additional column(s) of information from the web service map 68 may be displayed while the Type column is not displayed.

[0053] As shown in the example of FIG. 6, by displaying the component profile 22 within the actual GUI 26 in which the component is utilized, a developer may conveniently and quickly visualize the entire client-to-server code flow of the component of interest. Additionally and in some examples, the component profile 22 may be displayed in a manner that visually links the component profile to the component to which it corresponds. In this manner, the association between the selected component and its component profile may be further enhanced.

[0054] In some examples the component profile 22 may be displayed in close proximity to the corresponding component, including but not limited to touching the component. As illustrated in the example of FIG. 6, a border 24 of the component profile 22 may be displayed as touching a border 514 of the New User button 508. In another example, the border 24 of the component profile 22 and the border 514 of the New User button 508 may be displayed adjacent to one another without touching.

[0055] In some examples, the component profile 22 may be displayed in a manner that visually links the component profile to the component to which it corresponds by displaying the component profile and the component in a coordinated visual style. For example, in the example of FIG. 6 both the border 24 of the component profile 22 and the border 514 of the New User button 508 may be displayed in a similarly highlighted style, such as by displaying the

borders with a similar darkened line. In other examples, other manners of visually coordinating the component profile 22 and its corresponding component may be utilized, such as displaying the profile and component in the same color, with a similar distinctive font, as animated in a similar fashion, etc.

[0056] With reference to FIG. 7, in some examples the component profile 22 may be displayed in a manner visually linking it to its corresponding component by visually connecting the profile and component. In the example of FIG. 7, while the component profile 22 and New User button 508 are not displayed in close proximity to one another, the profile and this component are visually linked by an arrow 700. Many other examples of visually linking a component profile 22 to its corresponding component also may be utilized and are within the scope of the present disclosure

[0057] With reference now to FIG. 8, in some examples a user may desire to see only certain types of information in the component profile 22 for a component. Accordingly, the plug-in 230 may filter the component profile by a type to yield a type-filtered component profile 22'. In some examples, the type may comprise a function or category of a call. As shown in the example of FIG. 8, when the user selects to filter the component profile 22 of FIG. 7 to display only database queries, a corresponding type-filtered component profile 22' that includes only the database queries is generated and displayed in GUI 26.

[0058] With reference now to FIG. 9, a block diagram of a non-transitory machine-readable storage medium 900 containing instructions according to an example of the present disclosure is provided. In some examples and with reference also to FIG. 2, the non-transitory machine-readable storage medium may comprise instructions for performing the functions and methods described above. When executed by at least one processor, such as processor 212 of the computing device 204, such instructions may create client component maps 56 and web service maps 68, and may generate and display component profiles 22 in a manner consistent with the following example and other examples described herein.

[0059] In the example of FIG. 9 the instructions of non-transitory machine-readable storage medium 900 may include selection instructions 904 to receive a selection of a component in a graphical user interface of an application. The instructions may include identification instructions 908 to

associate the component with a component identifier. The instructions may include query instructions 912 to identify, in an analysis database, at least one web service call linked to the component identifier. The instructions may include profile display instructions 916 to cause a component profile to be displayed within the graphical user interface, the component profile comprising the component identifier and the web service call.

[0060] With reference now to FIGS. 10A and 10B, a block diagram of a non-transitory machine-readable storage medium 1000 containing instructions according to another example of the present disclosure is provided. In some examples and with reference also to FIG. 2, the non-transitory machine-readable storage medium 1000 may comprise instructions for performing the functions and methods described above. When executed by at least one processor, such as processor 212 of the computing device 204, such instructions may create client component maps 56 and web service maps 68, and may generate and display component profiles 22 in a manner consistent with the following example and other examples described herein.

[0061] In the example of FIGS. 10A and 10B, the instructions of non-transitory machine-readable storage medium 1000 may include selection instructions 1004 to receive a selection of a component in a graphical user interface of an application. The instructions may include identification instructions 1008 to associate the component with a component identifier. The instructions may include query instructions 1012 to identify, in an analysis database, at least one web service call linked to the component identifier. The instructions may also include query instructions 1016 to also identify at least one sub-call from the web service call.

[0062] The instructions may include profile display instructions 1020 to cause a component profile 22 to be displayed within the graphical user interface, the component profile comprising the component identifier and the web service call. The instructions may also include profile display instructions 1024 to display the component profile in a manner visually linking the component profile to the component. The instructions may include profile display instructions 1028 to display the component profile in close proximity to the selected component. The instructions may include profile display instructions 1032 to display the component profile and the selected component

in a coordinated visual style. The instructions may include filter instructions 1036 to filter the component profile by a type to yield a type-filtered component profile. The instructions may include profile display instructions 1040 to cause only the type-filtered component profile to be displayed within the graphical user interface display.

[0063] With reference now to FIG. 10B, at 1044 the non-transitory machine-readable storage medium 1000 may comprise a plug-in for a web browser. At 1048, where the web service call identifies a first function executed at a first server, the component profile may further comprise a third party call from the first server that identifies a second function executed at a second server. At 1052 the component profile may further comprise at least one database query. At 1056 the component profile may further comprise a combination of a client component map that links the component identifier to the web service call and a web service map that is called by the web service call.

[0064] Turning now to FIG. 11, a flow chart of a method 1100 for displaying a component profile within a graphical user interface according to another example of the present disclosure is provided. The following description of method 1100 is provided with reference to the software and hardware components described above and shown in FIGS. 1-10. The method 1100 may be executed in the form of instructions encoded on a non-transitory machine-readable storage medium that is executable by a processor. It will be appreciated that method 1100 may also be performed in other contexts using other suitable hardware and software components.

[0065] With reference to FIG. 11, at 1104 the method 1100 may include associating a component in a graphical user interface of an application with a component identifier. At 1108 the method 1100 may include querying a database in a storage to identify at least one web service call linked to the component identifier. At 1112 the method 1100 may include receiving a component profile from the storage, the component profile comprising a call flowchart that begins with the component identifier calling the web service call. At 1116 the method 1100 may include displaying the component profile within the graphical user interface of the application.

[0066] It will be appreciated that method 1100 is provided by way of example and is not meant to be limiting. Therefore, it is to be understood that

method 1100 may include additional and/or other steps than those illustrated in FIG. 11. Further, it is to be understood that method 1100 may be performed in any suitable order. Further still, it is to be understood that at least one step may be omitted from method 1100 without departing from the scope of this disclosure.

[0067] Turning now to FIG. 12, a flow chart of a method 1200 for displaying a component profile within a graphical user interface according to another example of the present disclosure is provided. The following description of method 1200 is provided with reference to the software and hardware components described above and shown in FIGS. 1-10. The method 1200 may be executed in the form of instructions encoded on a non-transitory machine-readable storage medium that is executable by a processor. It will be appreciated that method 1200 may also be performed in other contexts using other suitable hardware and software components.

[0068] With reference to FIG. 12, at 1204 the method 1200 may include associating a component in a graphical user interface of an application with a component identifier. At 1208 the method 1200 may include querying a database in a storage to identify at least one web service call linked to the component identifier. At 1212 the method 1100 may include receiving a component profile from the storage, the component profile comprising a call flowchart that begins with the component identifier calling the web service call. At 1216 the method 1200 may include displaying the component profile within the graphical user interface of the application. At 1220 the method 1200 may include displaying the component profile in a manner visually linking the component profile to the component. At 1224 the method 1200 may include displaying the component profile in a manner visually linking the component profile to the component by at least one of displaying the component profile in close proximity to the selected component and displaying the component profile and the component in a coordinated visual style.

[0069] At 1228 the method 1200 may include filtering the component profile by a type to yield a type-filtered component profile. At 1232 the method 1200 may include displaying only the type-filtered component profile within the graphical user interface of the application. At 1236 the method 1200 may include the storage located on a first server, and the call flowchart comprising at

least one third party call from the first server that identifies a second function executed at a second server.

[0070] It will be appreciated that method 1200 is provided by way of example and is not meant to be limiting. Therefore, it is to be understood that method 1200 may include additional and/or other steps than those illustrated in FIG. 12. Further, it is to be understood that method 1200 may be performed in any suitable order. Further still, it is to be understood that at least one step may be omitted from method 1200 without departing from the scope of this disclosure.

CLAIMS:

1. A non-transitory machine-readable storage medium encoded with instructions executable by a processor, the storage medium comprising:

selection instructions to receive a selection of a component in a graphical user interface of an application;

identification instructions to associate the component with a component identifier;

query instructions to identify, in an analysis database, at least one web service call linked to the component identifier; and

profile display instructions to cause a component profile to be displayed within the graphical user interface, the component profile comprising the component identifier and the web service call.

2. The non-transitory machine-readable storage medium of claim 1, the profile display instructions to display the component profile in a manner visually linking the component profile to the component.

3. The non-transitory machine-readable storage medium of claim 2, wherein displaying the component profile in a manner visually linking the component profile to the component comprises at least one of displaying the component profile in close proximity to the selected component and displaying the component profile and the selected component in a coordinated visual style.

4. The non-transitory machine-readable storage medium of claim 1, the query instructions to also identify at least one sub-call from the web service call.

5. The non-transitory machine-readable storage medium of claim 1, comprising:

filter instructions to filter the component profile by a type to yield a type-filtered component profile; and

the profile display instructions to cause only the type-filtered component profile to be displayed within the graphical user interface display.

6. The non-transitory machine-readable storage medium of claim 1, wherein the non-transitory machine-readable storage medium comprises a plug-in for a web browser.

7. The non-transitory machine-readable storage medium of claim 1, wherein the web service call identifies a first function executed at a first server, and the component profile further comprises a third party call from the first server that identifies a second function executed at a second server.

8. The non-transitory machine-readable storage medium of claim 1, wherein the component profile further comprises at least one database query.

9. The non-transitory machine-readable storage medium of claim 1, wherein the component profile comprises a combination of a client component map that links the component identifier to the web service call and a web service map that is called by the web service call.

10. A method, comprising:

associating a component in a graphical user interface of an application with a component identifier;

querying a database in a storage to identify at least one web service call linked to the component identifier;

receiving a component profile from the storage, the component profile comprising a call flowchart that begins with the component identifier calling the web service call; and

displaying the component profile within the graphical user interface of the application.

11. The method of claim 10, wherein displaying the component profile comprises displaying the component profile in a manner visually linking the component profile to the component.

12. The method of claim 11, wherein displaying the component profile in a manner visually linking the component profile to the component comprises at

least one of displaying the component profile in close proximity to the selected component and displaying the component profile and the component in a coordinated visual style.

13. The method of claim 10, comprising:

filtering the component profile by a type to yield a type-filtered component profile; and

displaying only the type-filtered component profile within the graphical user interface of the application.

14. The method of claim 10, wherein the storage is located on a first server, and the call flowchart comprises at least one third party call from the first server that identifies a second function executed at a second server.

15. A computing device, comprising:

a storage; and

a processor coupled to the storage, the processor to:

scan a client-server application to identify at least one web service call and at least one component in a graphical user interface that calls the web service call;

generate a client component map that links a component identifier of the component to the web service call;

scan server code to identify a web service that is called by the web service call;

generate a web service map comprising the web service call and at least one sub-call from the web service call;

combine the client component map and the web service map into a component profile; and

send the component profile to a web browser for display within the graphical user interface presented by the web browser.

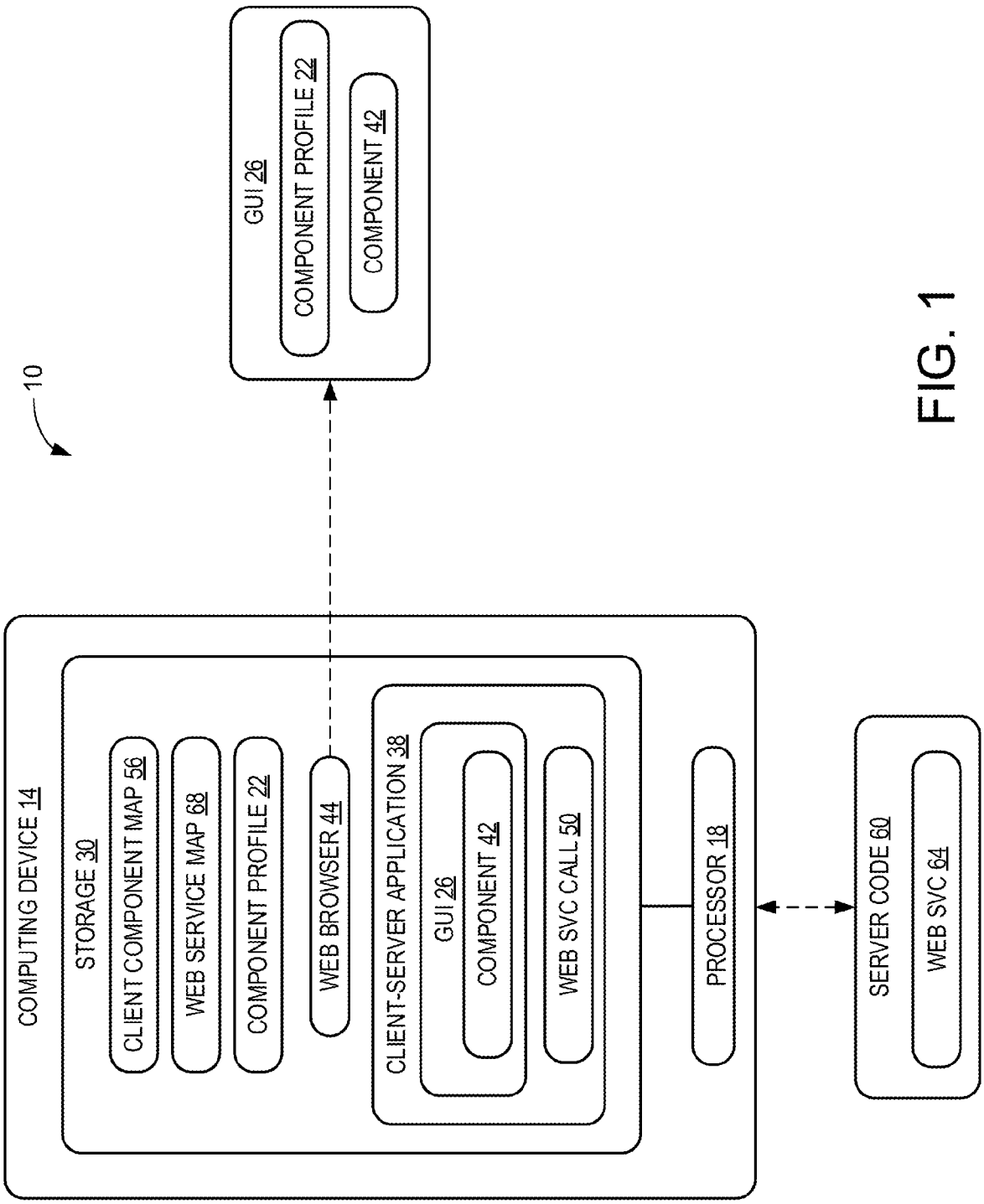
10
COMPUTING DEVICE 14
STORAGE 30
CLIENT COMPONENT MAP 56
WEB SERVICE MAP 68
COMPONENT PROFILE 22
WEB BROWSER 44
CLIENT-SERVER APPLICATION 38
GUI 26
COMPONENT 42
WEB SVC CALL 50
PROCESSOR 18
SERVER CODE 60
WEB SVC 64
GUI 26
COMPONENT PROFILE 22
COMPONENT 42

FIG. 1

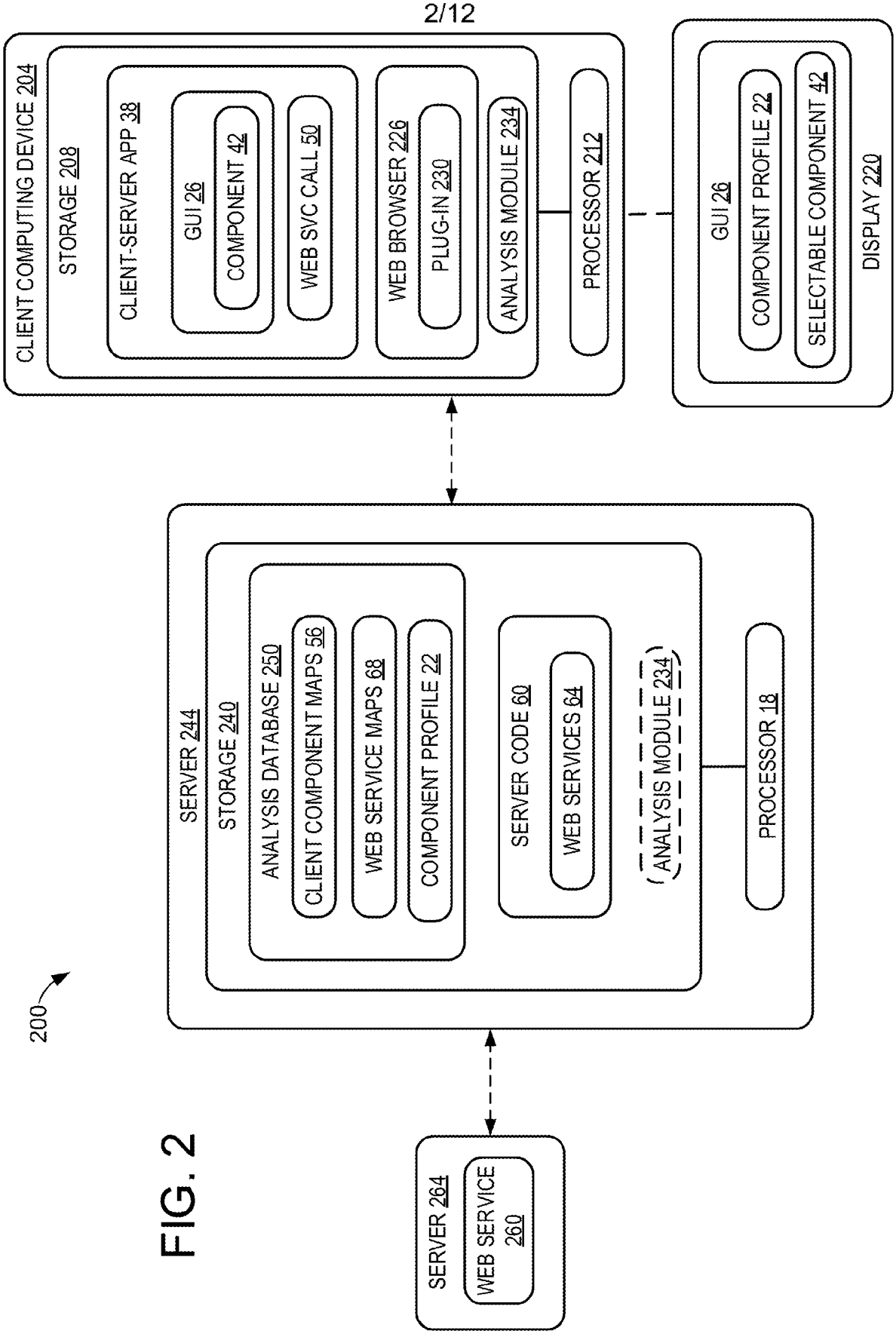
200
CLIENT COMPUTING DEVICE 204
STORAGE 208
CLIENT-SERVER APP 38
GUI 26
COMPONENT 42
WEB SVC CALL 50
WEB BROWSER 226
PLUG-IN 230
ANALYSIS MODULE 234
PROCESSOR 212
GUI 26
COMPONENT PROFILE 22
SELECTABLE COMPONENT 42
DISPLAY 220
SERVER 244
STORAGE 240
ANALYSIS DATABASE 250
CLIENT COMPONENT MAPS 56
WEB SERVICE MAPS 68
COMPONENT PROFILE 22
SERVER CODE 60
WEB SERVICES 64
ANALYSIS MODULE 234
PROCESSOR 18
SERVER 264
WEB SERVICE 260

FIG. 2

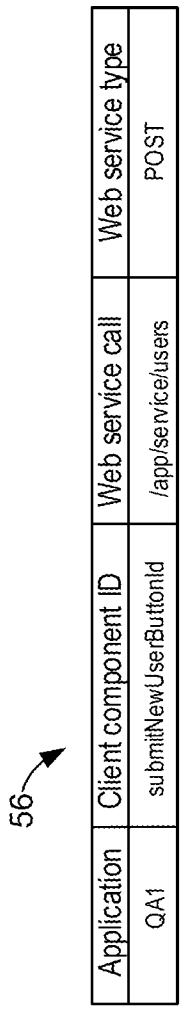
56
Application
Client component ID
Web service call
Web service type
QA1
submitNewUserButtonId
/app/service/users
POST

FIG. 3

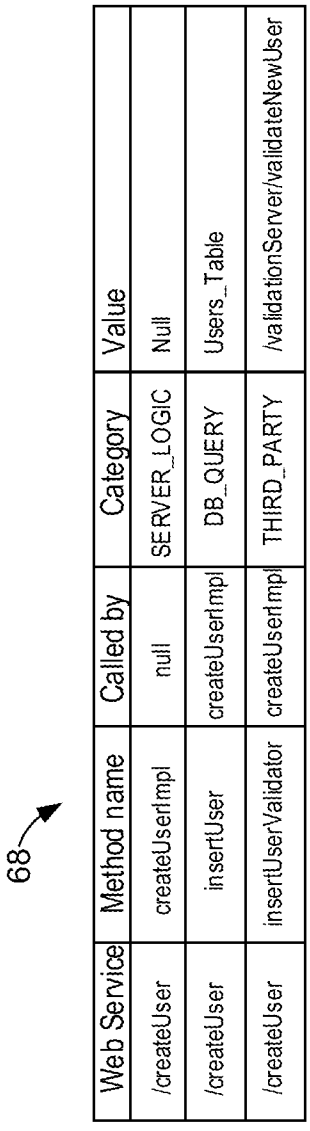
68
Web Service
Method name
Called by
Category
Value
/createUser
createUserImpl
null
SERVER_LOGIC
Null
/createUser
insertUser
createUserImpl
DB_QUERY
Users_Table
/createUser
insertUserValidator
createUserImpl
THIRD_PARTY
/validationServer/validateNewUser

FIG. 4

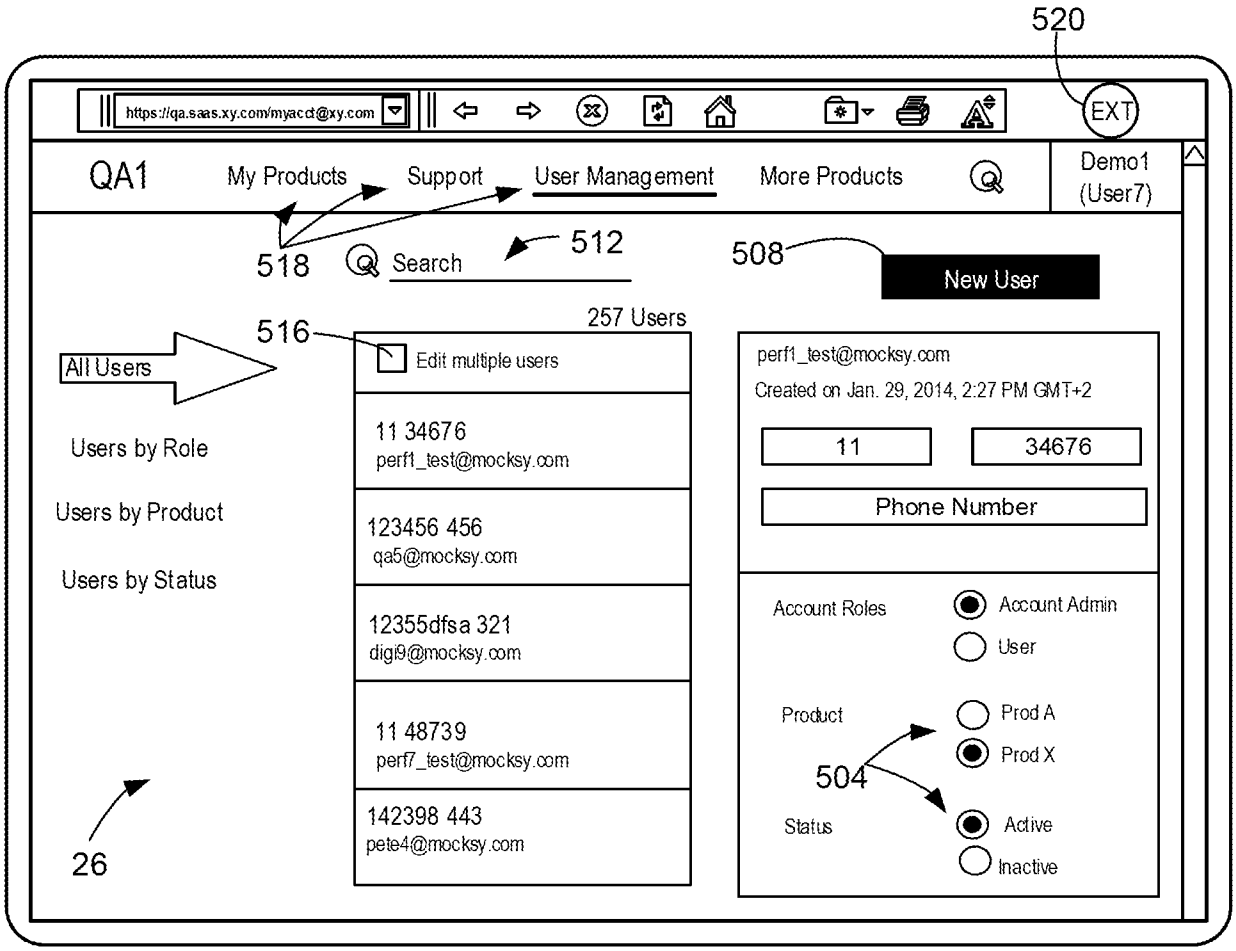
520
https://qa.saas.xy.com/myacct@xy.com
EXT
QA1
My Products
Support
User Management
More Products
Demo1
(User7)
518
Search
512
508
New User
257 Users
516
All Users
Edit multiple users
Users by Role
Users by Product
Users by Status
11 34676
perf1_test@mocksy.com
123456 456
qa5@mocksy.com
12355dfsa 321
digi9@mocksy.com
11 48739
perf7_test@mocksy.com
142398 443
pete4@mocksy.com
perf1_test@mocksy.com
Created on Jan. 29, 2014, 2:27 PM GMT+2
11
34676
Phone Number
Account Roles
Account Admin
User
Product
Prod A
Prod X
504
Status
Active
Inactive
26

FIG. 5

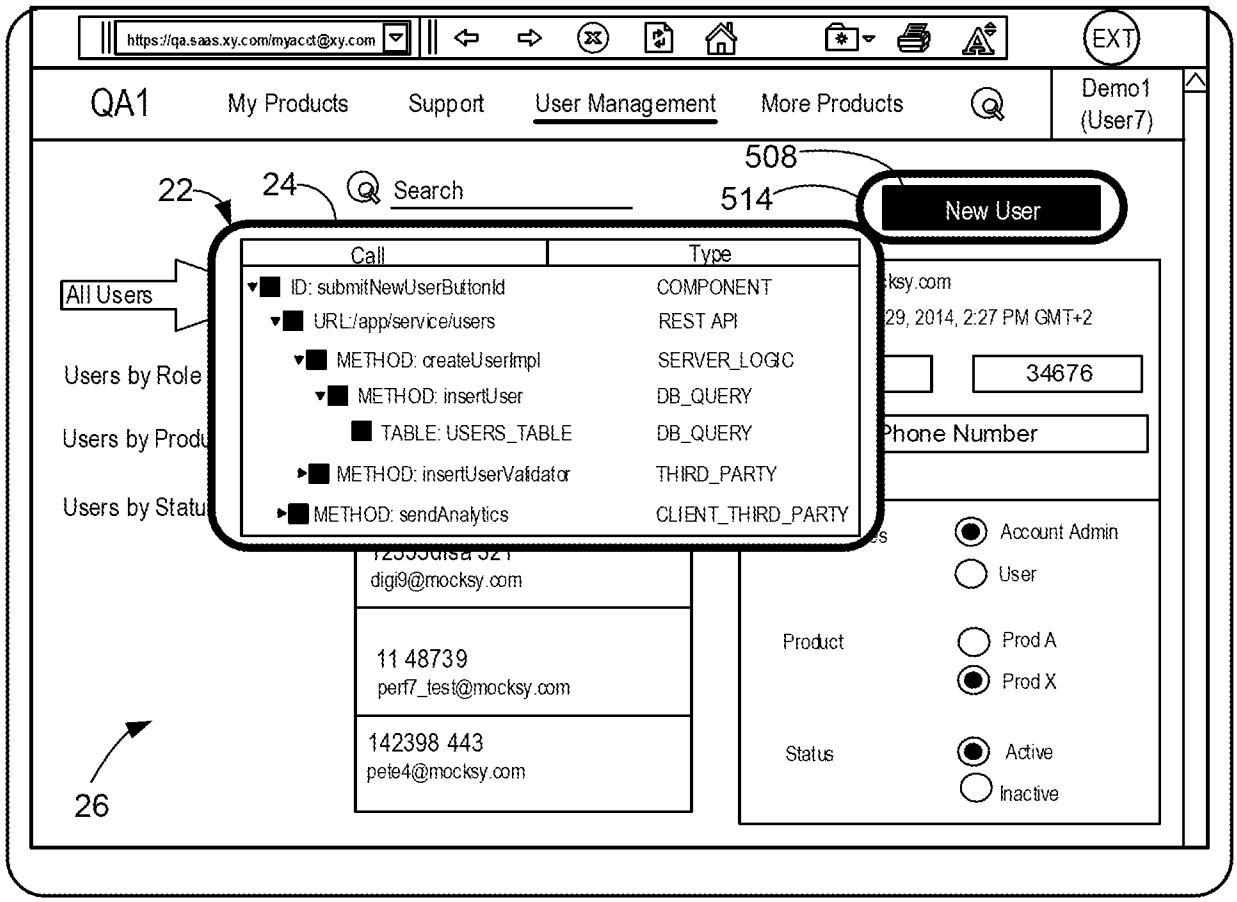
https://qa.saas.xy.com/myacct@xy.com
EXT
QA1
My Products
Support
User Management
More Products
Demo1
(User7)
508
22
24
Search
514
New User
Call
Type
All Users
ID: submitNewUserButtonId
COMPONENT
URL:/app/service/users
REST API
METHOD: createUserImpl
SERVER_LOGIC
METHOD: insertUser
DB_QUERY
Users by Role
TABLE: USERS_TABLE
DB_QUERY
Users by Produ
Phone Number
METHOD: insertUserValidator
THIRD_PARTY
Users by Statu
METHOD: sendAnalytics
CLIENT_THIRD_PARTY
ksy.com
29, 2014, 2:27 PM GMT+2
34676
Account Admin
User
digi9@mocksy.com
Product
Prod A
Prod X
11 48739
perf7_test@mocksy.com
Status
Active
142398 443
pete4@mocksy.com
Inactive
26

FIG. 6

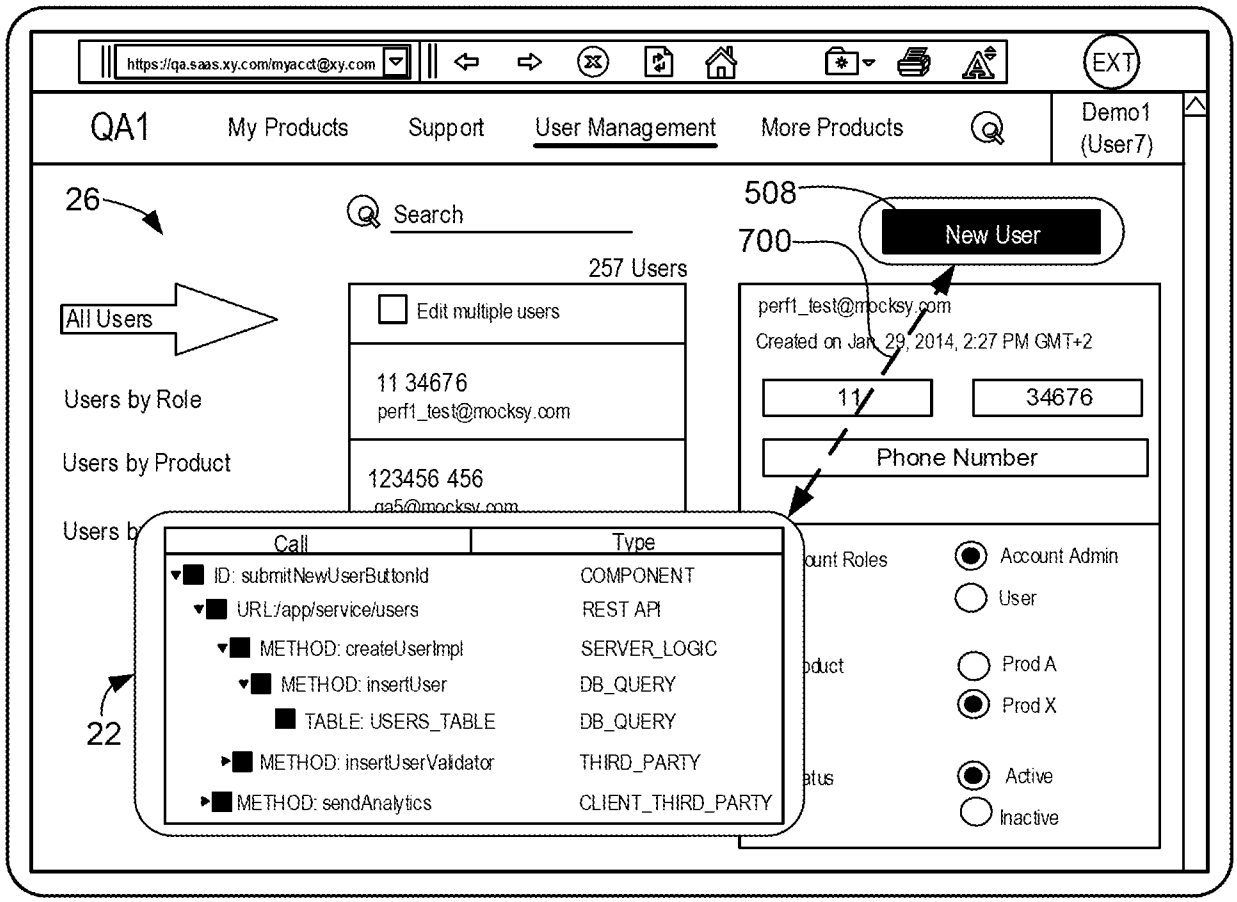
https://qa.saas.xy.com/myacct@xy.com
EXT
QA1
My Products
Support
User Management
More Products
Demo1
(User7)
26
Search
508
New User
700
257 Users
All Users
Edit multiple users
Users by Role
11 34676
perf1_test@mocksy.com
Users by Product
123456 456
Created on Jan 29, 2014, 2:27 PM GMT+2
11
34676
Phone Number
Call
Type
ID: submitNewUserButtonId
COMPONENT
URL:/app/service/users
REST API
METHOD: createUserImpl
SERVER_LOGIC
METHOD: insertUser
DB_QUERY
TABLE: USERS_TABLE
DB_QUERY
METHOD: insertUserValidator
THIRD_PARTY
METHOD: sendAnalytics
CLIENT_THIRD_PARTY
22
Account Admin
User
Prod A
Prod X
Active
Inactive

FIG. 7

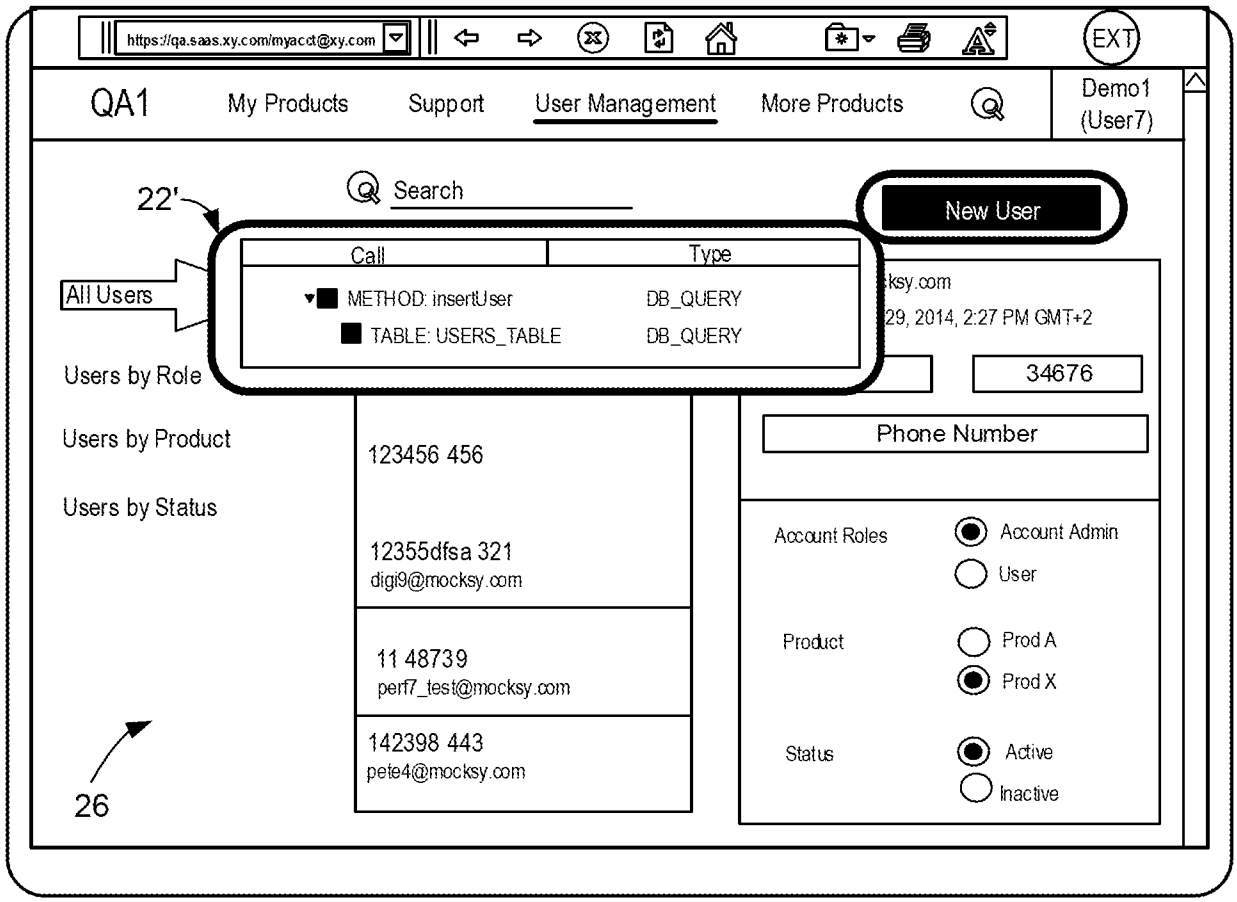
https://qa.saas.xy.com/myacct@xy.com
EXT
QA1
My Products
Support
User Management
More Products
Demo1
(User7)
22'
Search
New User
Call
Type
All Users
METHOD: insertUser
DB_QUERY
TABLE: USERS_TABLE
DB_QUERY
Users by Role
Users by Product
Users by Status
ksy.com
29, 2014, 2:27 PM GMT+2
34676
Phone Number
123456 456
12355dfsa 321
digi9@mocksy.com
11 48739
perf7_test@mocksy.com
142398 443
pete4@mocksy.com
Account Roles
Account Admin
User
Product
Prod A
Prod X
Status
Active
Inactive
26

FIG. 8

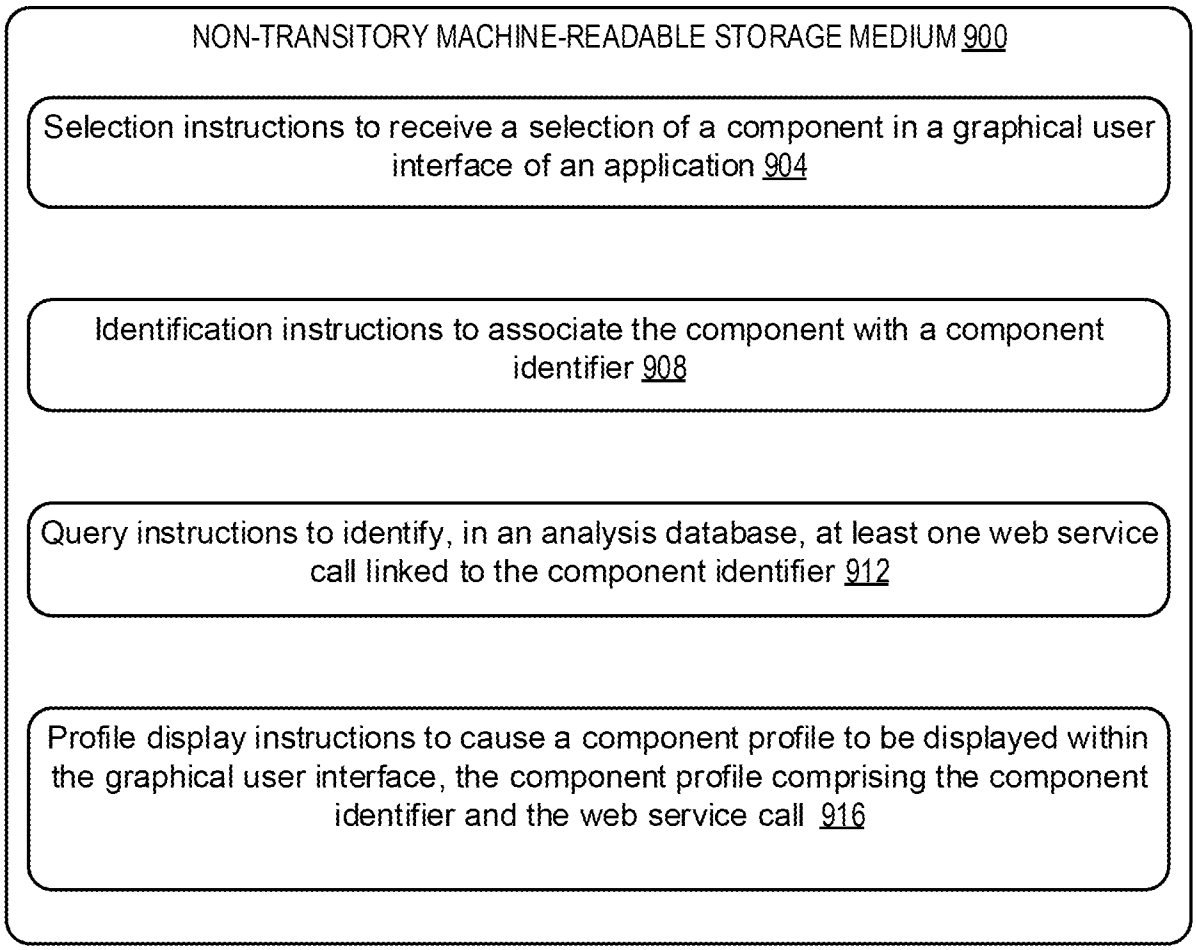
NON-TRANSITORY MACHINE-READABLE STORAGE MEDIUM 900
Selection instructions to receive a selection of a component in a graphical user interface of an application 904
Identification instructions to associate the component with a component identifier 908
Query instructions to identify, in an analysis database, at least one web service call linked to the component identifier 912
Profile display instructions to cause a component profile to be displayed within the graphical user interface, the component profile comprising the component identifier and the web service call 916

FIG. 9

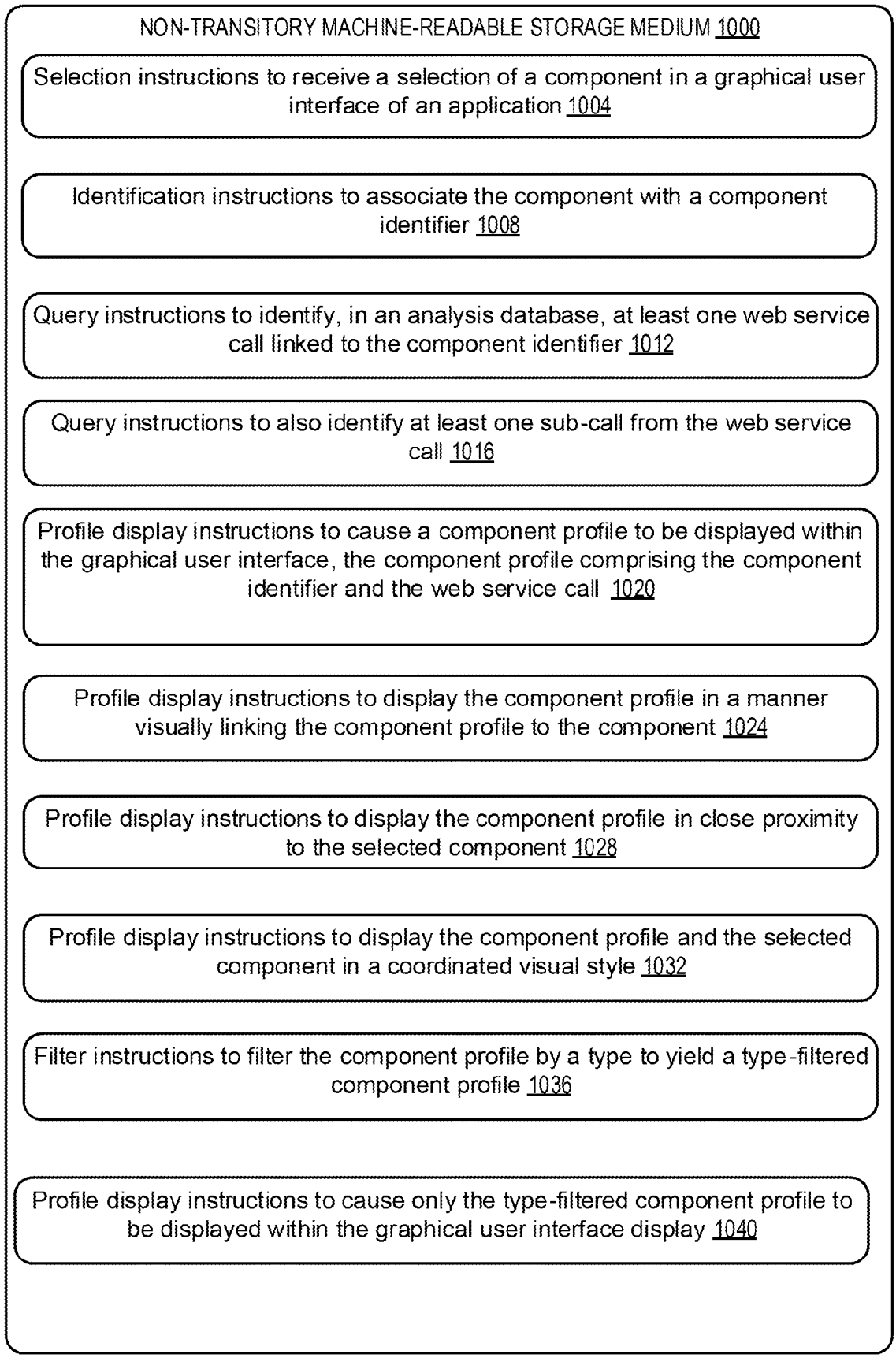
NON-TRANSITORY MACHINE-READABLE STORAGE MEDIUM 1000
Selection instructions to receive a selection of a component in a graphical user interface of an application 1004
Identification instructions to associate the component with a component identifier 1008
Query instructions to identify, in an analysis database, at least one web service call linked to the component identifier 1012
Query instructions to also identify at least one sub-call from the web service call 1016
Profile display instructions to cause a component profile to be displayed within the graphical user interface, the component profile comprising the component identifier and the web service call 1020
Profile display instructions to display the component profile in a manner visually linking the component profile to the component 1024
Profile display instructions to display the component profile in close proximity to the selected component 1028
Profile display instructions to display the component profile and the selected component in a coordinated visual style 1032
Filter instructions to filter the component profile by a type to yield a type-filtered component profile 1036
Profile display instructions to cause only the type-filtered component profile to be displayed within the graphical user interface display 1040

FIG. 10A

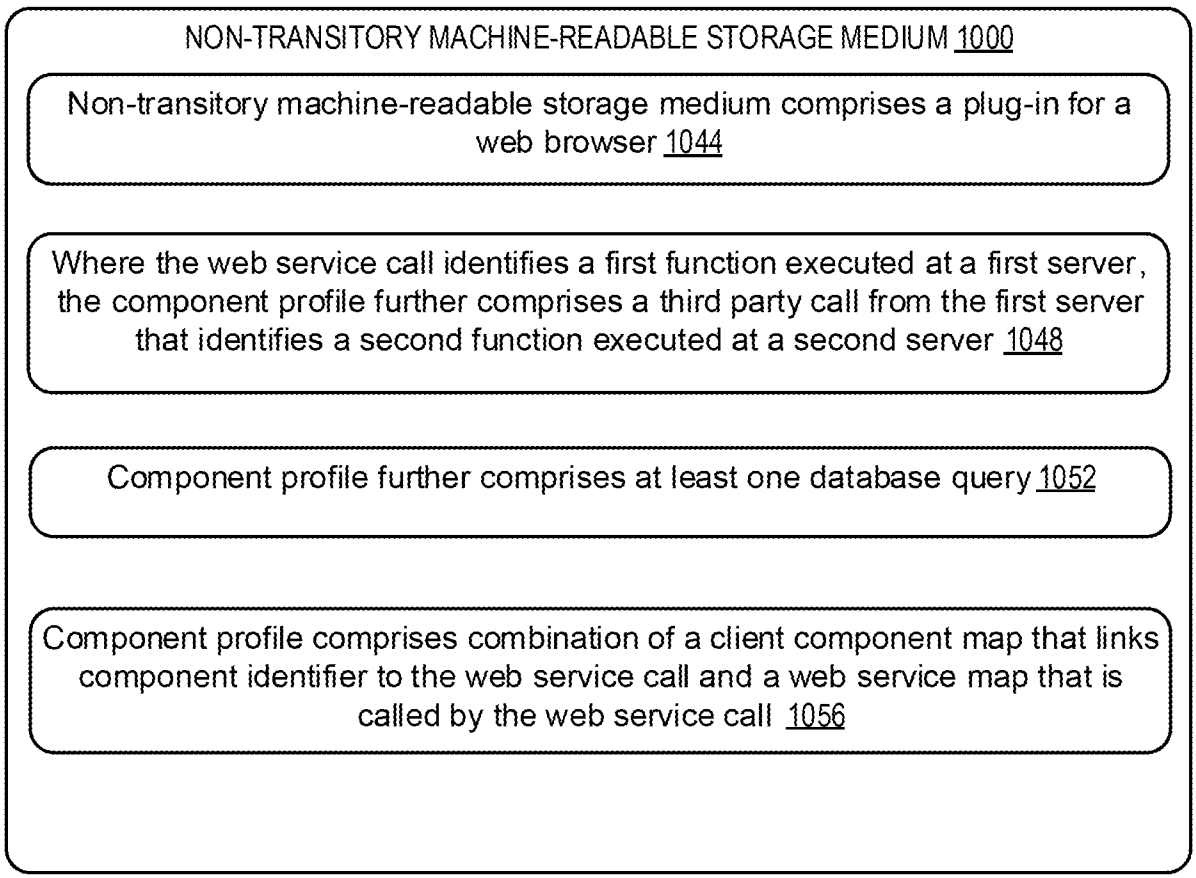
NON-TRANSITORY MACHINE-READABLE STORAGE MEDIUM 1000
Non-transitory machine-readable storage medium comprises a plug-in for a web browser 1044
Where the web service call identifies a first function executed at a first server, the component profile further comprises a third party call from the first server that identifies a second function executed at a second server 1048
Component profile further comprises at least one database query 1052
Component profile comprises combination of a client component map that links component identifier to the web service call and a web service map that is called by the web service call 1056

FIG. 10B

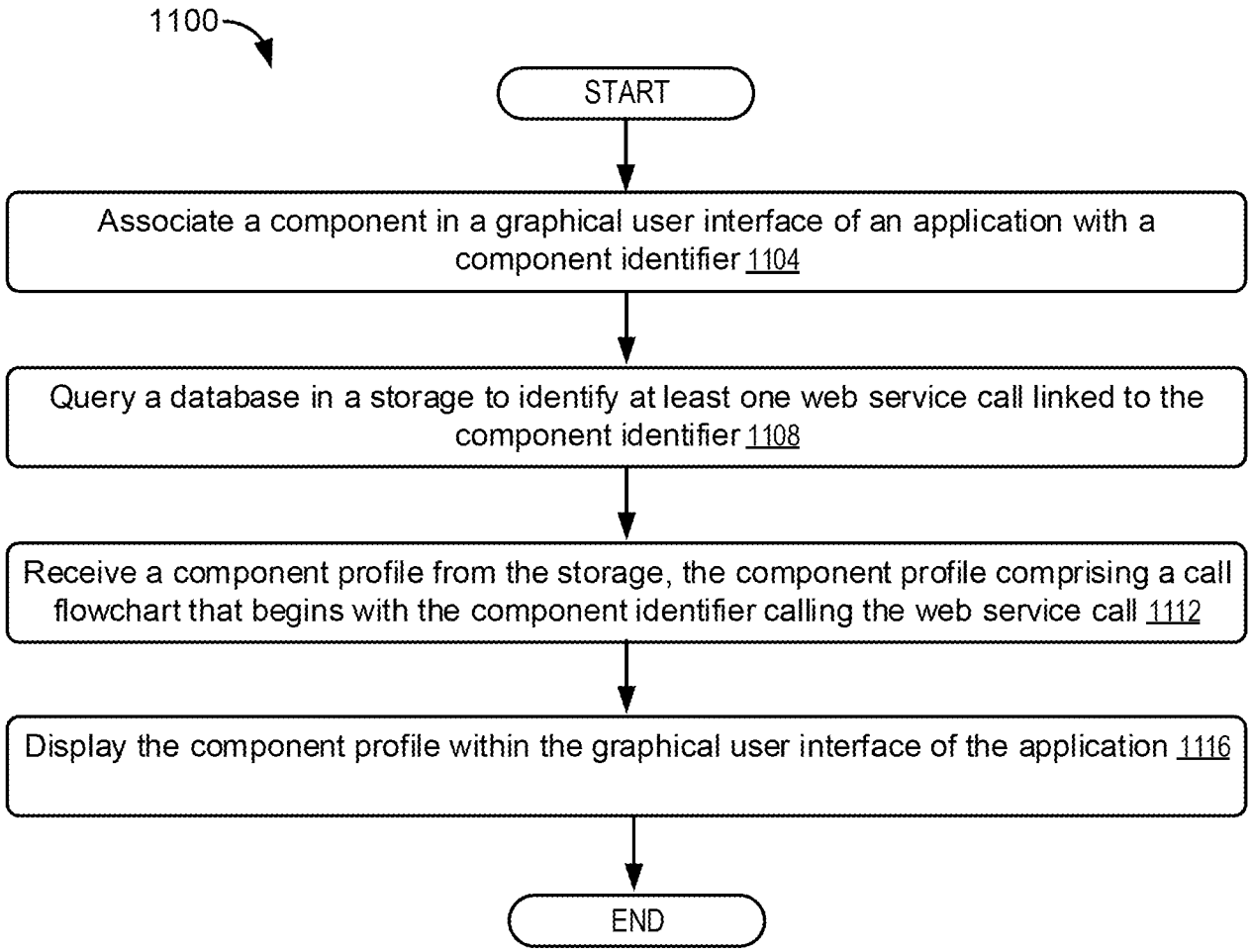
1100
START
Associate a component in a graphical user interface of an application with a component identifier 1104
Query a database in a storage to identify at least one web service call linked to the component identifier 1108
Receive a component profile from the storage, the component profile comprising a call flowchart that begins with the component identifier calling the web service call 1112
Display the component profile within the graphical user interface of the application 1116
END

FIG. 11

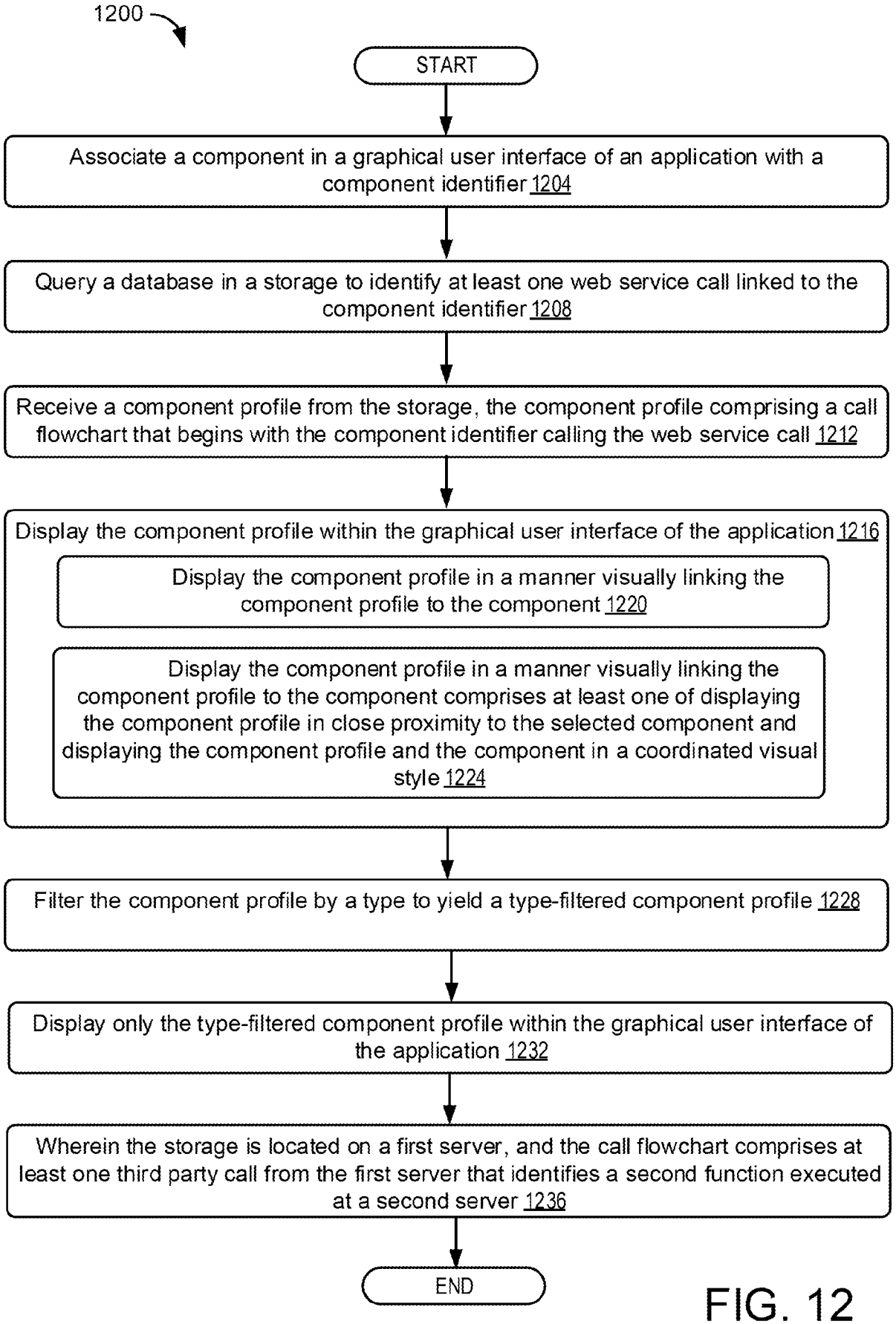
1200
START
Associate a component in a graphical user interface of an application with a component identifier 1204
Query a database in a storage to identify at least one web service call linked to the component identifier 1208
Receive a component profile from the storage, the component profile comprising a call flowchart that begins with the component identifier calling the web service call 1212
Display the component profile within the graphical user interface of the application 1216
Display the component profile in a manner visually linking the component profile to the component 1220
Display the component profile in a manner visually linking the component profile to the component comprises at least one of displaying the component profile in close proximity to the selected component and displaying the component profile and the component in a coordinated visual style 1224
Filter the component profile by a type to yield a type-filtered component profile 1228
Display only the type-filtered component profile within the graphical user interface of the application 1232
Wherein the storage is located on a first server, and the call flowchart comprises at least one third party call from the first server that identifies a second function executed at a second server 1236
END

FIG. 12

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2015/042534

A. CLASSIFICATION OF SUBJECT MATTER

G06F 3/048(2006.01)i, G06F 3/14(2006.01)i

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F 3/048; G06F 9/44; G06F 17/30; G06F 7/06; G06F 3/14

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: component, profile, identifier, web service call, application, display

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2008-0104569 A1 (MICHAEL A GILFIX et al.) 01 May 2008 See paragraphs [0010], [0016], [0018]; claims 11, 20; and figures 8-9.	1-15
A	US 2008-0134089 A1 (HISATOSHI ADACHI et al.) 05 June 2008 See paragraphs [0011], [0053]; and figure 5.	1-15
A	US 2013-0104067 A1 (BLAKE SULLIVAN et al.) 25 April 2013 See paragraphs [0007], [0040]; and figure 5.	1-15
A	US 2014-0047414 A1 (MOHAMMED M. ATTAR et al.) 13 February 2014 See paragraphs [0059]-[0060]; and figure 4.	1-15
A	US 2009-0307190 A1 (LOUIS MARESCA) 10 December 2009 See paragraph [0017]; and figure 2.	1-15

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

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents,such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
25 April 2016 (25.04.2016)	**26 April 2016 (26.04.2016)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 35208,
Republic of Korea
Facsimile No. +82-42-481-8578

Authorized officer
LEE, Dong Yun
Telephone No. +82-42-481-8734

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2015/042534

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2008-0104569 A1	01/05/2008	CN 101170580 A	30/04/2008
		CN 101170580 B	18/07/2012
		US 8671199 B2	11/03/2014
US 2008-0134089 A1	05/06/2008	CN 101192152 A	04/06/2008
		CN 101192152 B	07/12/2011
		JP 2008-140194 A	19/06/2008
		JP 4767828 B2	07/09/2011
US 2013-0104067 A1	25/04/2013	None	
US 2014-0047414 A1	13/02/2014	None	
US 2009-0307190 A1	10/12/2009	US 2014-0189496 A1	03/07/2014
		US 8782065 B2	15/07/2014