



- (51) **International Patent Classification:**  
*H04W 68/02* (2009.01) *H04W 12/08* (2009.01)
- (21) **International Application Number:**  
PCT/US2012/053025
- (22) **International Filing Date:**  
30 August 2012 (30.08.2012)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (30) **Priority Data:**  
13/224,217 1 September 2011 (01.09.2011) US
- (71) **Applicant (for all designated States except US):** **MICROSOFT CORPORATION** [US/US]; One Microsoft Way, Redmond, Washington 98052-6399 (US).
- (72) **Inventors:** **JOY, George**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **LAU, Cheuk Wan William**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **LOUIE, Darren**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **FIRSTENBERG, Yosef**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **CHERUKURI, Ravikant**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **WOLEY, Kevin Michael**; c/o Microsoft Corporation, LCA - International

al Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **AYERS, Matthew R.**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **ANAND, Gaurav S.**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (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, 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, QA, RO, RS, RU, RW, 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, 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, ML, MR, NE, SN, TD, TG).

[Continued on next page]

## (54) Title: STATELESS APPLICATION NOTIFICATIONS

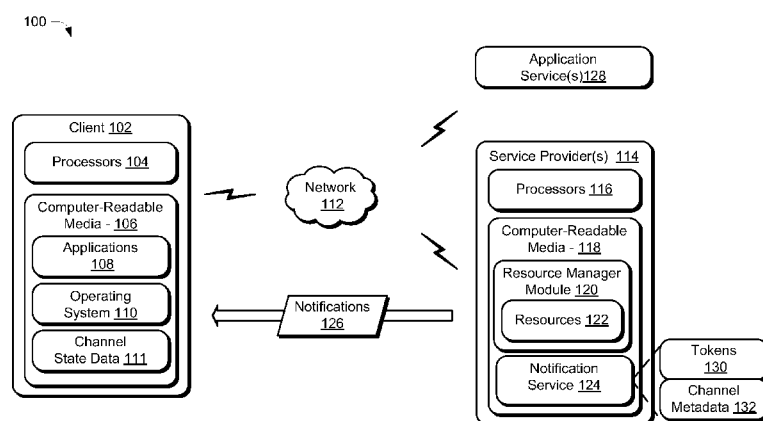


FIG. 1

(57) **Abstract:** Stateless application notifications are described that enable third parties to provide messages to client applications. A communication channel can be established between a notification service and an application. Upon request, the notification service can generate obfuscated routing data for the channel, which can be in the form of a channel handle or token. The routing data can be encrypted and digitally signed to obscure the content and format of the routing data from third parties. An application service possessing the obfuscated routing data can package a notification with the data and send the package to the notification service for delivery. The application service does so without knowing the channel particulars encoded by the obfuscated routing data. The notification service that produces the obfuscated routing data can decrypt and interpret the data, and deliver the notification on the channel to an appropriate endpoint application on behalf of the application service.

**Declarations under Rule 4.17:**

- *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*
- *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

**Published:**

- *with international search report (Art. 21(3))*
- *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

## **Stateless Application Notifications**

### **Background**

[0001] In a client-server environment where a service provider implements a notification service on the server-side to deliver messages to clients, the service can be configured to process and deliver notification messages to clients on behalf of third parties. In some traditional approaches, the messaging can be based on client state data such as addresses, routing data, account data, authentication data and/or other state data that is maintained persistently by a service provider on the server-side for numerous clients.

[0002] Maintaining such client state data for numerous clients, though, can be expensive because resources (e.g., memory and processing capacity) that could otherwise be enlisted by the service provider for other purposes are tied up for storage and management of the state data. Moreover, it is inefficient and wasteful to persistently store state data at the service provider for clients that may connect infrequently and/or may connect one time and then never reconnect again. In addition, a centralized store can be a security risk as a target for hackers to obtain client data and notifications are dependent upon the reliability of a single centralized store.

### **Summary**

[0003] This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter.

[0004] Various embodiments provide stateless application notifications designed to enable third parties to provide notifications to client applications. In at least some embodiments, a notification service establishes a communication channel for a client application and generates obfuscated routing data that can be employed to direct notifications to the application on the channel. In some embodiments, the routing data can be configured as a channel handle that is derived as an abstraction of the channel pathway or address. The channel handle or other suitable routing data can also be encoded as a token that is encrypted and digitally signed, such that the content and format of the token is obscured from third parties and protected from tampering.

[0005] An application service possessing the obfuscated routing data in some form can

package a notification with the data and send the package to the notification service for delivery. The application service can use the obfuscated routing data to direct notifications to the channel without understanding the obfuscated routing data or knowing the pathway/address for the channel. The notification service that produces the obfuscated routing data can decrypt and interpret the obfuscated routing data, verify signatures contained within the data to prevent tampering, and deliver the notification on the channel to an appropriate endpoint application on behalf of the application service.

### **Brief Description of the Drawings**

[0006] The same numbers are used throughout the drawings to reference like features.

10 [0007] Fig. 1 illustrates an operating environment in which various principles described herein can be employed in accordance with one or more embodiments.

[0008] Fig. 2 illustrates an example scenario for creating channels and channel handles in accordance with one or more embodiments.

15 [0009] Fig. 3 illustrates an example scenario to employ channel handles to deliver notifications in accordance with one or more embodiments.

[0010] Fig. 4 is a flow diagram that describes steps of a method in accordance with one or more embodiments.

[0011] Fig. 5 is a flow diagram that describes steps of another method in accordance with one or more embodiments.

20 [0012] Fig. 6 is a flow diagram that describes steps of another method in accordance with one or more embodiments.

[0013] Fig. 7 illustrates an example system that can be used to implement one or more embodiments.

### **Detailed Description**

#### **Overview**

25 [0014] Stateless application notifications are described that enable third parties to provide messages to client applications. A communication channel can be established between a notification service and an application. Upon request, the notification service can generate obfuscated routing data for the channel, which can be in the form of a channel handle or token. The routing data can be encrypted and digitally signed to obscure the content and format of the routing data from third parties and prevent tampering. An application service possessing the obfuscated routing data can package a notification with the data and send the package to the notification service for delivery. The application service does so without knowing the channel particulars encoded by the

30

obfuscated routing data. The notification service that produces the obfuscated routing data can decrypt and interpret the data, and deliver the notification on the channel to an appropriate endpoint application on behalf of the application service.

[0015] In the discussion that follows, a section titled “Operating Environment” is provided and describes one environment in which one or more embodiments can be employed. Following this, a section titled “Stateless Application Notification Examples” describes example details of stateless application notifications in accordance with one or more embodiments. Next, a section titled “Example Methods” describes example techniques for stateless application notifications in accordance with one or more embodiments. Last, a section titled “Example System” describes example computing systems and devices that can be utilized to implement one or more embodiments.

### **Operating Environment**

[0016] Fig. 1 illustrates an operating environment in accordance with one or more embodiments, generally at 100. Environment 100 includes a client 102 having one or more processors 104, one or more computer-readable media 106 and one or more applications 108 and an operating system 110 that reside on the computer-readable media and which are executable by the processor(s). The client 102 can be embodied as any suitable computing system and/or device such as, by way of example and not limitation, a desktop computer, a portable computer, a tablet or slate device, a handheld computer such as a personal digital assistant (PDA), a cell phone, a set-top box, and the like. One example of a computing system that can represent various systems and/or devices including the client 102 is shown and described below in Fig. 7.

[0017] The computer-readable media can include, by way of example and not limitation, all forms of volatile and non-volatile memory and/or storage media that are typically associated with a computing device. Such media can include ROM, RAM, flash memory, hard disk, removable media and the like. Computer-readable media can include both “computer-readable storage media” and “communication media”, examples of which can be found in the discussion of the example computing system of Fig. 7.

[0018] In addition, client 102 includes channel state data 111 that can be employed to facilitate stateless notifications as described above and below. The channel state data 111 relates to delivery pathways or “channels” that can be established between a client 102 and a notification service to provide notifications for applications 108 of the client. The channel state data 111 can include identifiers, routing data, tokens, authorization data, and

expiration data related to one or more channels, to name a few examples. Clients 102 can cache their own channel state data 111 to enable stateless processing on the server-side.

[0019] As further depicted in Fig. 1, the client 102 can be communicatively coupled over a network 112, such as the Internet, with one or more service providers 114 which provide various resources (e.g., content and services), as described above and below. By way of example, the client 102 can interact with the service provider 114 and exchange messages to create channels and employ the channels to obtain notifications.

[0020] The service providers 114 are each depicted as having one or more processors 116 and one or more computer-readable media 118. The service provider 114 is further depicted as including a resource manager 120. The resource manager 120 represents functionality operable by a service provider 114 to manage various resources 122 that may be made available over the network 112. For example, various resources 122 can be provided via web pages or other user interfaces that are communicated over the network for output by the client 102. The resource manager 120 can manage access to the resources 122, performance of the resources 122, configuration of user interfaces or data to provide the resources 122, and so on.

[0021] The service provider 114 can provide one or more web servers and/or sites, which a client 102 can navigate to in order to access the resources 122. The resources 122 can include any suitable combination of services and/or content typically made available over a network by one or more providers. Some examples of services include, but are not limited to, a search service, an email service to send and receive email, an instant messaging service to provide instant messages between clients 102, and a social networking service to facilitate connections and interactions between groups of users who share common interests and activities. Content may include various combinations of text, video, ads, audio, multi-media streams, animations, images, web pages, web applications, device applications, content for display by a browser or other device application, and the like.

[0022] In addition, the service provider 114 can include or otherwise make use of a notification service 124 that is representative of functionality operable to manage communication of notifications 126 between the client 102 and other entities. The notification service 124 can be operated alone and/or in conjunction with other services to provide functionality for one or more different types of messaging. By way of example and not limitation, the multiple different types of messaging provided by the notification service can include email, text messages, application and/or operating system

notifications, in application content, social networking notices and invites, instant messaging, voice messages, peer-to-peer messaging, and so forth.

[0023] The notification service 124 is also operable to implement techniques for stateless application notifications described herein. In particular, the notification service 124 can be implemented to facilitate communication of notifications 126 to the client 102 on behalf of one or more application services 128. The notifications 126 can be configured as messages that are directed to applications 108 corresponding to the application services, such as promotions, application updates, alerts, in-app features, and so forth. The notification service 124 is responsible for routing and delivering the notifications 126 to clients 102. Channel state data 111 cached by a client 102 can be obtained on demand by the notification service 124 if necessary. However, the notification service 124 can perform “stateless” processing for channels and notifications on the server-side using obfuscated routing data that is provided with notification requests. The notification service 124 may temporarily store state information regarding clients when they are currently connected. The state data can include the currently-connected client’s identifying information, information about what kinds/types of notifications to allow or block, and so forth. When the client disconnects, though, temporary state data may be discarded by the Notification Service 124 and therefore the processing is considered “stateless”.

[0024] As discussed in greater detail below, notifications 126 can be delivered on a channel that is established between the client 102 and the service provider 114. Notifications can be sent over the channel by associating the notifications with obfuscated routing data, such as a channel handle or token that is provided through the notifications service 124. In one example approach to providing notifications via channels, the notification service 124 can employ tokens 130 configured to carry channel metadata 132. The channel metadata 132 can represent data used “in process” by the notification service 124 to process requests, establish channels, route notifications, and so forth. This can include various channel state data 111 that is obtained as needed from clients 102. The notification service 124, though, does not necessarily cache the channel metadata 132 on behalf of clients 102.

[0025] Various data related to channels can be included as part of channel metadata 132 including a channel ID, channel handles, and routing data. The channel ID uniquely identifies a channel and is maintained such that the channel ID is not exposed to third parties. The third parties, though, are able to employ channel handles generated for a

channel and provided in a token or otherwise to direct messages through the notification service 124 to endpoints without interpreting the handle or knowing the corresponding channel ID. In particular, a channel handle can be derived as an abstraction of the channel ID to obscure the properties of the channel. Details regarding these and other aspects of channels, channel handles, and stateless application notifications are provided in relation to the following figures.

[0026] Having described an example operating environment, consider now a discussion of some stateless application notifications examples in accordance with one or more embodiments.

### **Stateless Application Notification Examples**

[0027] This section describes example details regarding stateless application notifications in accordance with one or more embodiments. In the discussion that follows, first a sub-section titled “Channels for Notification Delivery” describes channels that can be established to facilitate application notifications. Next a sub-section titled “Stateless Tokens” provides details regarding tokens that can be employed in some embodiments to carry routing data used for routing notifications to appropriate endpoints using the channels. Thereafter, example interactions between a service provider 114, a client 102, and an application service 128 to implement stateless application notifications are discussed in relation to example operational scenarios depicted in Figs. 2 and 3.

### **Channels for Notification Delivery**

[0028] As introduced above, notifications can be delivered on a channel that is established between the client 102 and the service provider 114. Generally speaking, a channel is a representation of a delivery pathway (e.g., address) for notifications from an application service 128 to a particular endpoint. An endpoint as used herein corresponds to a unique combination of a particular user, device, and/or application. In other words, a channel can be considered a communication pathway that is established for and associated with a particular application 108 installed on particular client 102 and used by a particular user of the client 102. The channels are a mechanism by which a globally unique address can be assigned to an endpoint application.

[0029] Each channel can have a corresponding channel identifier (ID). The channel ID is a persistent identifier uniquely identifying a delivery pathway to a corresponding endpoint. To secure a channel and prevent attackers from hijacking or misusing the channels, the channel ID can be kept obscured. In at least some embodiments, the channel ID is maintained securely by the client 102 and is not exposed to third parties. For



instance, the channel IDs for one or more applications 108 can be configured as parameters/objects that are internal to the operating system 110. The operating system 110 can make use of the channel ID in interactions with the notification service 128, but otherwise keeps the channel ID private.

5   **[0030]**     Thus, to enable notifications on a channel, channel handles are created that can be issued to third parties in lieu of exposing and/or using the channel ID itself. When notifications are activated for an application, such as upon installation or execution of the application, a channel can be established which can be used to update application specific content. In one approach, the application can request issuance of a channel handle through  
10   the operating system 108. For example, a request can be made through a local application programming interface (API). The operating system can in turn interact with the notification service 124 to cause the notification service 124 to issue a channel handle and return the channel handle to the application 108. The notification service 124 can also establish a new channel if a channel does not already exist. The application 108 can then  
15   distribute the channel handle to an application service 128 to enable corresponding notifications from the application service 128.

**[0031]**     More particularly, a channel handle can be derived as an abstraction of the channel ID for distribution to applications and third parties. The channel handle can be constructed by the notification service 124 in response to a request from a client. The  
20   request from a client can be configured to contain a channel ID and other client state data 111 that is used by the notification service 124 to create a corresponding channel handle. The handle obfuscates the properties of the channel and can be encrypted to prevent modification of the handle. Digital signatures can also be used to prevent modification or tampering. The notification service 124 is responsible for creating the channel handle and  
25   also controls the format and content of the channel handle. Thus, the handle's format and content can be obscured from both the client 102 and/or third party application service 128. In this manner, the channel handle can be configured such that the handle is interpretable by the notification service 124 and not interpretable by other parties.

**[0032]**     Channel handles can be formatted in any suitable way. In general, the channel  
30   handle is configured to represent a pathway or address to an endpoint application. Thus, a channel handle can encode various routing data for a channel. By way of example and not limitation, the notification service 124 can create channel handles as uniform resource identifiers (URIs) that encode a combination of identifiers for a particular application, client device, and user of the client device. More generally, a channel handle can

represent any combination of routing data that is sufficient to identify and route a notification to a corresponding endpoint. The handle can be created based on an underlying channel ID and therefore the notification service 124 is able to interpret the handles and deliver notifications sent using the handles on appropriate channels to corresponding endpoints.

[0033] Multiple channel handles can be issued for a particular channel ID. Thus, expired or compromised channel handles can be de-authorized without affecting the underlying channel ID. Channel handles can also be configured to have a limited lifetime by associating the channel handles with an expiration time. After the expiration time, the channel handle can no longer be used to deliver notifications. An application, though, can request issuance of a new channel handle on the same channel. Channel handles can also be renewed prior to expiration to provide uninterrupted delivery of notifications to a channel. When a channel handle is renewed, the notification service 124 provides a new handle that can be used as a replacement for the previous handle on the same channel.

Multiple channel handles can be active for a channel at the same time. Further details regarding handle renewal are discussed in relation to the following example methods.

[0034] The channel itself can remain open indefinitely and many channel handles can be assigned to the channel over the lifetime of the channel. A channel can be explicitly closed, at which point any channel handles associated with the closed channel will no longer be usable for sending notifications. Data maintained by the client 102 controls such decommissioning of a channel. Decommissioning of a channel is a matter of removing the channel ID from the client 102 permanently, such that any use of a handle associated with that channel ID will be rejected by the client 102 since the channel ID no longer exists. A new channel ID would then be issued to represent a new channel established for a particular application 108. Accordingly, persistent channels can be established upon which multiple limited use channel handles can be issued to enable communications via the channels while protecting the channel from being attacked or otherwise compromised.

### **Stateless Tokens**

[0035] As mentioned, the channel state data 111 cached by the client 102 can enable stateless delivery of notifications by the notification service 124 using the channels. Rather than storing state data on behalf of clients, the notification service 124 is able to “look-up” or otherwise be provided with channel state data 111 from the client 102 as needed. Therefore the service provider 114 can avoid costs associated with storing the

state data. Thus, clients 102 can be configured to cache some or all of their own channel state data 111 and provide the data to the service provider 114 as needed. Data related to channels obtained from a client or otherwise is represented as channel metadata 132 in Fig. 1.

5     **[0036]**     In operation, tokens 130 can be used to pass channel metadata 132 that is encrypted and digitally signed through the notification service 124. The tokens 130 can be considered stateless since the channel metadata 132 used to generate the tokens 130 can be “in process” data that is obtained by the notification service 124 from clients 102 as needed and may not actually be stored persistently by the service provider 114.

10    **[0037]**     Tokens 130 can be configured in various ways. Generally, the token 130 includes routing data configured in a specific format that is interpretable by the notification service 124 and can be used to route notifications that use the tokens to corresponding endpoints. By way of example, a token 130 can be configured as BLOB (binary large object) that is packaged with select channel metadata 132 as a payload. A  
15    variety of other token configurations are also contemplated. A token 130 for example can be configured to include a URI or other suitable channel handle. Tokens 130 can also be configured to include security information such as versioning data, authentication data, expiration times, and/or other security information used to control and verify the validity of tokens. The security information enables the use of different secret keys and/or  
20    different formats for different token versions. Further, the token can be encrypted and/or digitally signed to prevent tampering. In some embodiments, the notification service 124 alone is able to decrypt and interpret the token 130 and route notifications accordingly to an endpoint specified by channel metadata 132 (e.g., the channel handle or other routing data) contained in the token.

25    **[0038]**     In some embodiments, channel metadata 132 encrypted within the token 130 can include routing data directly. The routing data enables the notification service 124 to statelessly process and route messages to endpoints using the self-contained information that is passed in a token 130. Routing data can include by way of example, version data, a channel handle, identifiers for a corresponding application 108, client 102, and/or  
30    machine, a digital certificate, a public key used for encryption, and so forth. Because the notification service 124 issues the tokens, the notification service 124 is able to interpret messages sent using the tokens, decrypt tokens 130 to obtain channel metadata 132 (e.g., using a private key), verify routing data contained in the token, and route notifications to appropriate channels.

[0039] In some cases, notification routing can occur “in process” using a channel handle and/or other suitable routing data that is contained within the token 130 itself. Additionally or alternatively, the notification service 124 can “look-up” an appropriate endpoint based on information contained in the token. For example, the notification  
5 service 124 can match a channel handle provided in a token to a channel ID and route the message to a corresponding endpoint. Thus, tokens 130 as just described provide one example mechanism by which a notification service 124 can statelessly deliver notifications to applications 108 on behalf of application services 128.

[0040] In the context of the forgoing discussion of channels and stateless tokens,  
10 consider now some example operational scenarios depicted in Figs. 2 and 3. In particular, Fig. 2 depicts a diagram 200 showing an example sequence of transactions to provide channel handles for stateless application notifications. Fig. 3 depicts a diagram 300 showing an example sequence of transactions involved in employing a channel handle to deliver a notification through a notification service 124.

[0041] Referring to the example of Fig. 2, an example scenario 200 illustrates acts  
15 involved in creation of channels and channel handles. At 202 an application 108 can initiate a channel request. For instance, the application can initiate the request when the application is installed or executed, when a user activates notification, or otherwise upon activation to obtain notifications. The request causes processing by a computing device  
20 102 to provide an appropriate response having channel data that can be used by the application 108 to enable notifications. In some embodiments, the channel request and response can be processed through a local application programming interface (API) provided by the operating system 110.

[0042] In response to the request, at 204 the operating system 110 can look-up an  
25 existing channel ID for the application. If a channel exists, then the existing channel ID can be provided with the request. The operating system 110 can also decide based on a policy whether or not to request creation of a new channel handle or use an existing channel handle. If a channel does not already exist, a request can be made to create a new channel.

[0043] At 206, the operating system 110 packages application specific information  
30 and/or the channel ID into a create channel message and sends the create channel message to the notification service 124. The application information can include routing data and/or other channel state data 111 that can be used by the notification service 124 to create a channel handle and/or issue a token 130 as discussed previously.

[0044] At 208, the notification service 124 can create the requested channel in response to the channel request if appropriate. Additionally or alternatively, the notification service 124 can create a new channel handle for the channel. In some embodiments, the notification service 124 forms a token 130 as discussed above to contain the channel handle and/or routing data sufficient to direct notifications to the channel indicated by the channel ID in the request, or for a newly created channel. The token 130 can be encrypted and digitally signed by the notification service 124.

[0045] At 210, the notification service 124 returns channel data to the operating system 110 of the client 102. The channel data returned can include the channel handle, a token when tokens are employed, obfuscated routing data and/or a new channel ID if appropriate. Because the returned channel data contains information to route notifications, the notification service 124 can discard the request and corresponding information without caching data on behalf of clients. Subsequent processing of a channel handle and/or token can occur statelessly using channel data provided back in a notification message.

Generally, this processing occurs “in process” using the provided channel data and without looking-up additional state data for a channel. However, the notification service 124 can look-up or request additional channel state data 111 from a client in some scenarios.

[0046] At 212, the operating system 110 caches the returned channel data. This can involve creating a mapping between channel IDs and application endpoints. As noted, the channel ID and the mapping can be kept private by the operating system 110. Accordingly, the operating system 110 can extract the channel ID and keep the channel ID and mapping secret. At 214, though, other channel data can be provided back to the application 108 to facilitate notifications. This can include providing a channel handle, a token having a handle, and/or other channel data related to the channel back to the application 108.

[0047] To enable notifications on the channel, at 216 the application 108 provides appropriate channel data to a corresponding application service 128. Again this can include a channel handle, a token, and/or other channel data that can be employed by the application service 128 to route messages back to the application. At 218, the application service can store the channel data for subsequent use to deliver notifications back to the application.

[0048] It should again be noted, that both the application 108, operating system 110, and application service 128 may be unaware of the format and content of the handle or token used for notifications. The application 108, operating system 110, and application

service 128 may not be configured to interpret, decrypt, or otherwise process the handle or token. Rather, the application 108 and application service 128 are configured to understand that a given handle and/or token can be used to deliver notifications to a corresponding endpoint through the notification service 124. In one approach, a notification is attached by the application service 128 as a payload to a handle/token and communicated to the notification service 124 that performs processing to route and deliver the notification. Details regarding using a handle/token to deliver notifications are discussed just below in relation to Fig. 3.

[0049] In particular, Fig. 3 depicts one example scenario 300 for employing a channel handle to deliver a notification through a notification service 124. The channel handle can be provided individually or be contained within a token 130. In either case the channel handle is protected by being encrypted and/or signed digitally. In this example, the channel handle represents encoded routing data that can be interpreted by the notification service and used to deliver notifications to the correct endpoints. Such routing data can be contained within an encrypted and signed token as mentioned previously.

[0050] An application service 128 can produce a notification related to a particular application. The notification can be configured to provide an application update, in application content, a promotion or special offer, a warning, and so forth. The notification can be a global message that is directed to any endpoint having the particular application or a message targeted to one or more specific endpoints based on selection criteria such as device type, user criteria, platform, and so forth.

[0051] At 302, the application service looks-up a channel handle for a notification. For instance, the application service 128 can cache channel handles for one or more endpoints having the particular application (or selection criteria) and for which notifications are activated. Thus, the application service 128 can refer to the cached data to match the notification to appropriate channel handles.

[0052] At 304, the notification is sent to the notification service 124. The notification can be attached as a payload to the channel handle. In the case of delivery to multiple endpoints, the notification can be associated with multiple channel handles corresponding to the endpoints.

[0053] At 306, the notification service 124 processes the notification, which can include decrypting the channel handle and verifying the validity of the channel handle. For instance, the notification service 124 can decrypt the channel handle and/or a token 130 containing the channel handle using a decryption key. This enables the notification

service 124 to obtain encoded routing data sufficient to deliver the notification. The notification service 124 can also check to make sure the channel handle has not expired or been revoked and that the corresponding channel is still active.

[0054] Further, the notification service 124 can authenticate the application service

5 128 by requesting the application service 128 to sign-in with pre-determined credentials (e.g., a secret key, password/ID, or other shared secret) before permitting notifications. The pre-determined credentials can be matched to credentials included within channel handles provided by the application service 128. A comparison is made and if the credentials do not match, the notification delivery may be rejected. In order to enable the  
10 credential comparison, the channel handles may be tied to a specific identity for authentication at the time the handles are created. This may be accomplished by the notification service 124 embedding credentials, provided from the client 102 by the operating system 110 or otherwise, into the handle at the time the handle is created.

[0055] Assuming the channel handle is valid, the notification service 124 interprets the

15 channel handle to determine where to send the notifications. In particular, at 308 the notification service 124 determines a client endpoint and corresponding channel ID using the channel handle. For example, the channel handle can be processed by the notification service 124 to reconstruct the channel ID. In another approach, the channel handle can be used as a reference to look-up a corresponding channel ID. In the case of a token, routing  
20 data encoded within the token may be sufficient to directly route the notification after the token is decrypted and verified.

[0056] In any case, the notification service 124 identifies where to send the notification and at 310 communicates the notification to the client 102. For example, a communication referencing a channel ID can be directed to the operating system 108 of  
25 the client 102. At 312, the operating system 108 performs processing to ascertain an application 108 to which the notification is directed based on the channel ID. One way this can occur is through a local API of the operating system 108. When the appropriate application 108 is identified, the notification is delivered to the application 108 at 314.

The application 108 can process the notification and perform any responsive actions

30 dictated by the notification accordingly. For instance, the application 108 can present the notification for viewing within a user interface, perform or schedule an update, provide an alert and/or visual indicator regarding the notification, store the notification, and so forth.

[0057] Having considered some examples of stateless application notifications, consider now example procedures for stateless application notifications in accordance with one or more embodiments.

#### **Example Methods**

5 [0058] Fig. 4 is a flow diagram that describes steps of a method in accordance with one or more embodiments. The method can be implemented in connection with any suitable hardware, software, firmware, or combination thereof. In at least some embodiments, the method can be implemented by way of a suitability configured computing device, such as the example service provider 114 of Fig. 1 that includes or  
10 otherwise makes use of notification service 124.

[0059] Step 400 obtains a request from a client for a token associated with a channel. For example, upon activation of an application to receive notifications, the application can send a channel request as described previously. The request can include client state data 111 that can be used to create a token for a particular application of a client 102. The  
15 notification service 124 can receive and process the request to create a channel and/or obfuscated routing data that can be used to access an existing channel.

[0060] In particular, step 402 generates a token containing routing data for the client. The routing data can be in the form of a channel handle or other combination of client state data 111 that encodes the particulars of the channel. Effectively, the routing data  
20 encodes a delivery pathway or address for the channel that can be recovered later by the notification service 124 to route notifications 126. The routing data can be provided as part of a token produced by the notification service 124. The token can be created and employed statelessly using the client state data 111 provided in the request. The token can be configured to contain routing data sufficient to enable the notification service 124 to  
25 later determine a pathway or address for the channel “in-process” based on the token alone without looking-up data from a table, database, or external source. The token can also include an expiration time, security data, and other metadata.

[0061] Step 404 encrypts and signs the token. This protects the token and obfuscates the routing data from third parties. Step 406 returns the token to the client to enable the  
30 client to employ the channel for notifications. The token can be provided through the operating system 110 to an application 108, which can in turn distribute the token for use by an application service 128.



[0062] Step 408 obtains a notification for delivery to the client that uses the token. For instance, the notification service 124 can obtain a notification from an application service 128 that is packaged or otherwise associated with a token.

[0063] Step 410 decrypts the token to retrieve the routing data and verify the validity of the routing data. Because the notification service 124 issues the token and controls the security of the token, the notification service 124 is able to decrypt and interpret the token. The notification service 124 can also verify the validity of routing data by checking the expiration time, making sure that the channel is active, ensuring that notifications are enabled for the channel, and otherwise checking various security information.

[0064] An error handling process can be initiated if a problem with the routing data or token is discovered. An error message can be sent back to a sender of the notification and other corrective actions can be performed such as reissuing a token, sending an error notice to the client, locking a channel, de-authorizing the sender and so forth. Otherwise, when the routing data is valid, the notification service can process the decrypted routing data to reconstruct a channel ID or otherwise identify a channel described by the routing data. Then, step 412 delivers the notification to the client based on the routing data retrieved from the token.

[0065] Fig. 5 is a flow diagram that describes steps of another method in accordance with one or more embodiments. The method can be implemented in connection with any suitable hardware, software, firmware, or combination thereof. In at least some embodiments, the method can be implemented by way of a suitability configured computing device, such as the example client 102 of Fig. 1. For instance, the method can be implemented by way of a local application programming interface (API) of the operating system 108 adapted to enable client-side techniques for stateless application notifications.

[0066] Step 500 requests a channel handle for a channel from a notification service. A channel handle for a particular channel can be requested for an application at various times. This can occur when an application is initially installed, upon execution of the application, or when notifications are activated by a user or otherwise. The operating system 110 can process the application request through an API as discussed previously.

[0067] When an application calls the operating system 108 to obtain the channel handle, the operating system 108 can determine based on a policy whether to provide a previously cached handle (if available) or to request a new handle. A new handle can be

requested for example when there is not an existing handle, when a cached handle has expired, when it is appropriate to obtain a new channel, and so forth.

[0068] A new handle can be requested to renew a handle before it expires. This can occur for example based on a policy that dictates timing for obtaining renewed handles (or  
5 tokens). For renewal, a handle is requested on a pre-existing channel. A new handle can be issued for the pre-existing channel. As mentioned multiple handles can be active for a channel at the same time. Thus, previously created handles for a channel can continued to be used after renewal so long as the handles have not expired, been revoked, or otherwise been deactivated. After renewal though, the application can distribute the newest handle  
10 for use with subsequent notifications.

[0069] It should be noted that renewal can also be initiated by an application service 128 to renew handles or tokens maintained by the application service 128 individually or as a group. This enables the application service 128 to periodically refresh routing data for channels to ensure continuity of notifications. The renewal occurs through the  
15 notification service 124 and the renewed data can be communicated back to the requesting application service 128. The notification service 124 can also provide the renewed handles or tokens to appropriate clients 102.

[0070] Step 502 obtains a response including a channel handle that obfuscates the channel. For example, notification service 124 can return the requested channel handle  
20 back to the client 102. The channel handle can incorporate obfuscated routing data that can be interpreted by the notification service 124. In addition, a channel ID can be included with the response. The client 102 can cache the returned data (e.g. handle, token, routing data, channel ID) for subsequent use.

[0071] Step 504 provides the channel handle to an application to enable activation of  
25 notifications from an application service. A cached channel handle or a newly created handle can be provided to an application upon request. The application can then communicate the channel handle to a corresponding application service 128 to activate notifications. An application service 128 that possesses the channel handle is able to direct notification to the corresponding channel through the notification service 124  
30 without actually understanding the obfuscated routing data encoded by the channel handle as discussed above and below.

[0072] Step 506 receives notifications sent to the channel through the notification service using the channel handle. Notifications can be communicated to the client 102 from the notification service. The notifications can be received and processed by the

operating system 108 and/or a local API as discussed herein. In particular, the notification service 124 interprets and routes the notifications on behalf of the application service 128 using the channel handle. Clients 102 can therefore can receive various notifications through the notification service 124 on respective channels established for the clients 102.

5 Notifications can be processed, stored, and and/or output at the client 102 in various way as described previously.

[0073] Fig. 6 is a flow diagram that describes steps of a method in accordance with one or more embodiments. The method can be implemented in connection with any suitable hardware, software, firmware, or combination thereof. In at least some  
10 embodiments, the method can be implemented by way of a suitability configured computing device, such as by one or more servers configured to provide the example application service 128 of Fig. 1.

[0074] Step 600 receives obfuscated routing data from a client to activate notifications to a corresponding application of the client. For instance, an application service 128 can  
15 obtain and store a channel handle, token, or other obfuscated routing data from an application 108. The obfuscated routing data enables the service to send notifications to the application 108 over a corresponding channel. The channel itself can be obscured from the application service 128 in the manner discussed herein.

[0075] Step 602 generates a notification for delivery to the application. Notifications  
20 can be configured as messages for delivery to a particular application. The notifications can include program updates, in application content, promotions, alerts, links, rewards, achievement messages, and so forth.

[0076] Step 604 packages the notification with obfuscated routing data. In particular, the application service 128 can create a package that contains the notifications and is  
25 addressed to a channel by including the obfuscated routing data in the package. For example, an application update can be packaged with a channel URI, handle, token, or other routing data issued for a channel. The notification can be considered a payload that is attached to or otherwise associated with delivery instructions represented by the obfuscated routing data contained in the package.

30 [0077] Step 606 communicates the notification with the obfuscated routing data to cause a notification service to deliver the notification to the application as designated by the obfuscated routing data. Note again that the application service 128 can use the obfuscated routing data to direct messages to endpoints without necessarily knowing the details of the routing data or the actual address of the endpoint encoded therein. The

package of a notification with obfuscated routing data is configured by the application service 128 to take advantage of the service provided by the notification service 124. The notification service 124 is responsible for decrypting the obfuscated routing data and delivering the notification accordingly. Thus, the application service 128 can simply make use of obfuscated routing data in various forms to direct notifications and turns processing over to the notification service 124 to identify the correct endpoints and route the notifications.

[0078] Having considered various example methods for stateless application notifications, consider now an example system that can be employed to implement various aspects of stateless application notifications in accordance with one or more embodiments.

### **Example System**

[0079] Fig. 7 illustrates an example system generally at 700 that includes an example computing device 702 that is representative of one or more such computing systems and/or devices that may implement the various embodiments described above. The computing device 702 may be, for example, a server of a service provider 114, a device associated with the computing device 102 (e.g., a client device), a server that provides and application service 128, a system on-chip, and/or any other suitable computing device or computing system.

[0080] The example computing device 702 includes one or more processors 704 or processing units, one or more computer-readable media 706 which may include one or more memory and/or storage components 708, one or more input/output (I/O) interfaces 710 for input/output (I/O) devices, and a bus 712 that allows the various components and devices to communicate one to another. Computer-readable media 706 and/or one or more I/O devices may be included as part of, or alternatively may be coupled to, the computing device 702. The bus 712 represents one or more of several types of bus structures, including a memory bus or memory controller, a peripheral bus, an accelerated graphics port, and a processor or local bus using any of a variety of bus architectures. The bus 712 may include wired and/or wireless buses.

[0081] The one or more processors 704 are not limited by the materials from which they are formed or the processing mechanisms employed therein. For example, processors may be comprised of semiconductor(s) and/or transistors (e.g., electronic integrated circuits (ICs)). In such a context, processor-executable instructions may be electronically-executable instructions. The memory/storage component 708 represents memory/storage capacity associated with one or more computer-readable media. The memory/storage

component 708 may include volatile media (such as random access memory (RAM)) and/or nonvolatile media (such as read only memory (ROM), Flash memory, optical disks, magnetic disks, and so forth). The memory/storage component 708 may include fixed media (e.g., RAM, ROM, a fixed hard drive, etc.) as well as removable media (e.g., a  
5 Flash memory drive, a removable hard drive, an optical disk, and so forth).

[0082] Input/output interface(s) 710 allow a user to enter commands and information to computing device 702, and also allow information to be presented to the user and/or other components or devices using various input/output devices. Examples of input devices include a keyboard, a touchscreen display, a cursor control device (e.g., a mouse),  
10 a microphone, a scanner, and so forth. Examples of output devices include a display device (e.g., a monitor or projector), speakers, a printer, a network card, and so forth.

[0083] Various techniques may be described herein in the general context of software, hardware (fixed logic circuitry), or program modules. Generally, such modules include routines, programs, objects, elements, components, data structures, and so forth that  
15 perform particular tasks or implement particular abstract data types. An implementation of these modules and techniques may be stored on or transmitted across some form of computer-readable media. The computer-readable media may include a variety of available medium or media that may be accessed by a computing device. By way of example, and not limitation, computer-readable media may include “computer-readable  
20 storage media” and “communication media.”

[0084] “Computer-readable storage media” may refer to media and/or devices that enable persistent and/or non-transitory storage of information in contrast to mere signal transmission, carrier waves, or signals per se. Thus, computer-readable storage media refers to non-signal bearing media. Computer-readable storage media also includes  
25 hardware elements having instructions, modules, and/or fixed device logic implemented in a hardware form that may be employed in some embodiments to implement aspects of the described techniques.

[0085] The computer-readable storage media includes volatile and non-volatile, removable and non-removable media and/or storage devices implemented in a method or  
30 technology suitable for storage of information such as computer readable instructions, data structures, program modules, logic elements/circuits, or other data. Examples of computer-readable storage media may include, but are not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, hard disks, magnetic cassettes, magnetic tape, magnetic

disk storage or other magnetic storage devices, hardware elements (e.g., fixed logic) of an integrated circuit or chip, or other storage device, tangible media, or article of manufacture suitable to store the desired information and which may be accessed by a computer.

[0086] “Communication media” may refer to a signal bearing medium that is

5 configured to transmit instructions to the hardware of the computing device, such as via a network. Communication media typically may embody computer readable instructions, data structures, program modules, or other data in a modulated data signal, such as carrier waves, data signals, or other transport mechanism. Communication media also include any information delivery media. The term “modulated data signal” means a signal that has  
10 one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media include wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, RF, infrared, and other wireless media.

[0087] Combinations of any of the above are also included within the scope of

15 computer-readable media. Accordingly, software, hardware, or program modules, including the APIs, notification service 124, operating system 110, applications 108, and other program modules, may be implemented as one or more instructions and/or logic embodied on some form of computer-readable media.

[0088] Accordingly, particular modules, functionality, components, and techniques

20 described herein may be implemented in software, hardware, firmware and/or combinations thereof. The computing device 702 may be configured to implement particular instructions and/or functions corresponding to the software and/or hardware modules implemented on computer-readable media. The instructions and/or functions may be executable/operable by one or more articles of manufacture (for example, one or  
25 more computing devices 702 and/or processors 704) to implement techniques for stateless application notifications, as well as other techniques. Such techniques include, but are not limited to, the example procedures described herein. Thus, computer-readable media may be configured to store or otherwise provide instructions that, when executed by one or more devices described herein, cause various techniques for stateless application  
30 notifications.

### **Conclusion**

[0089] Stateless application notifications have been described by which obfuscated routing data in the form of a channel handle and/or token can be employed to send notifications through a notification service. The routing data can be encrypted and

digitally signed to obscure the content and format of the routing data from third parties. An application service possessing the obfuscated routing data can package a notification with the data and send the package to the notification service for delivery. The notification service is configured to decrypt and interpret the data, and deliver the notification to an appropriate endpoint on behalf of the application service.

[0090] Although the subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that the subject matter defined in the appended claims is not necessarily limited to the specific features or acts described above. Rather, the specific features and acts described above are disclosed as example forms of implementing the claims.

**Claims**

What is claimed is:

1. One or more computer-readable storage media storing instructions, that when executed by a computing device, cause the computing device to implement an notification service configured to perform acts comprising:
  - 5 establishing a communication channel with a client application to deliver notifications from an application service to the client application;
  - generating obfuscated routing data for the communication channel usable by the application service to direct notifications on the channel through the notification service;
  - 10 issuing the obfuscated routing data for use by the application service;
  - obtaining a notification packaged with the obfuscated routing data from the application service; and
  - interpreting the obfuscated routing data to identify the communication channel and direct the notification to the client application.
- 15 2. The one or more computer-readable storage media of claim 1, wherein the obfuscated routing data is not interpretable by the application service.
3. The one or more computer-readable storage media of claim 1, wherein the obfuscated routing data encodes a pathway of the communication channel that can be ascertained by the notification service from the obfuscated routing data in process and  
20 without looking-up cached state data for the communication channel.
4. The one or more computer-readable storage media of claim 1, wherein the obfuscated routing data comprises a channel handle that is derived as an abstraction of an address of the communication channel.
5. The one or more computer-readable storage media of claim 1, wherein the  
25 obfuscated routing data comprises a token formatted to carry routing data interpretable by the notification service to direct notifications sent in association with the token on the communication channel.
6. The one or more computer-readable storage media of claim 1, wherein the notification service is further configured to perform acts comprising securing the  
30 communication channel by encrypting and digitally signing the obfuscated routing data.
7. The one or more computer-readable storage media of claim 1, wherein establishing the communication channel comprises assigning a channel identifier as a persistent globally unique address between the notification service and the client application.



8. The one or more computer-readable storage media of claim 1, wherein the format and content of the obfuscated routing data is controlled by the notification service and obscured from both the client application and the application service.

5 9. The one or more computer-readable storage media of claim 1, wherein generating the obfuscated routing data comprises constructing a channel handle for the communication channel in response to a request from the client application based upon state data cached at the client and provided in the request.

10 10. The one or more computer-readable storage media of claim 1, wherein interpreting the obfuscated routing data comprises:

10 decrypting the obfuscated routing data;  
checking an expiration time to verify validity of the decrypted routing data;  
authenticating the application service to send notifications based on a comparison of credentials embedded within the obfuscated routing data to credentials used by the application service to sign-in; and  
15 processing the decrypted routing data to identify the communication channel for the notification.

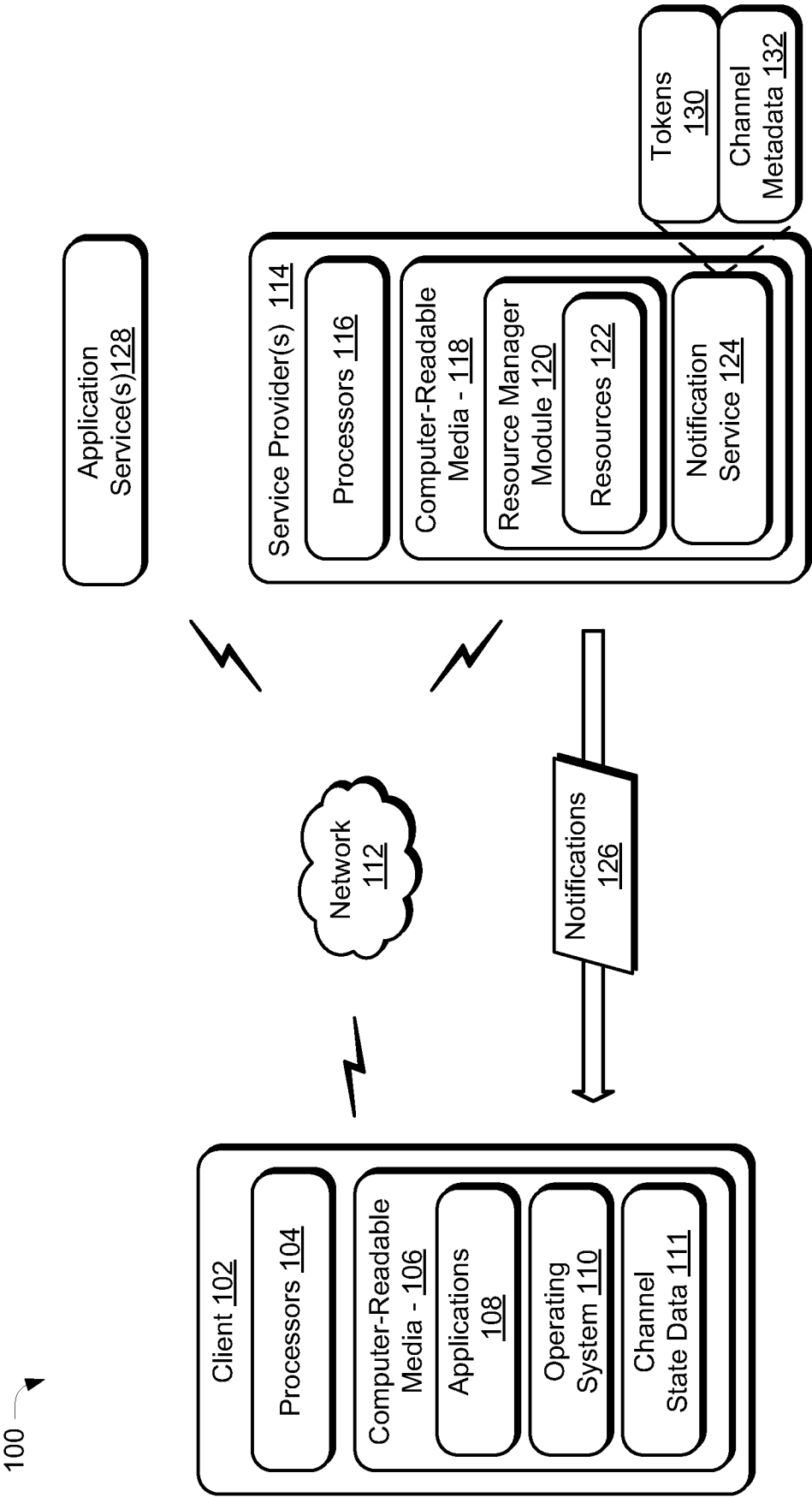


FIG. 1

2/7

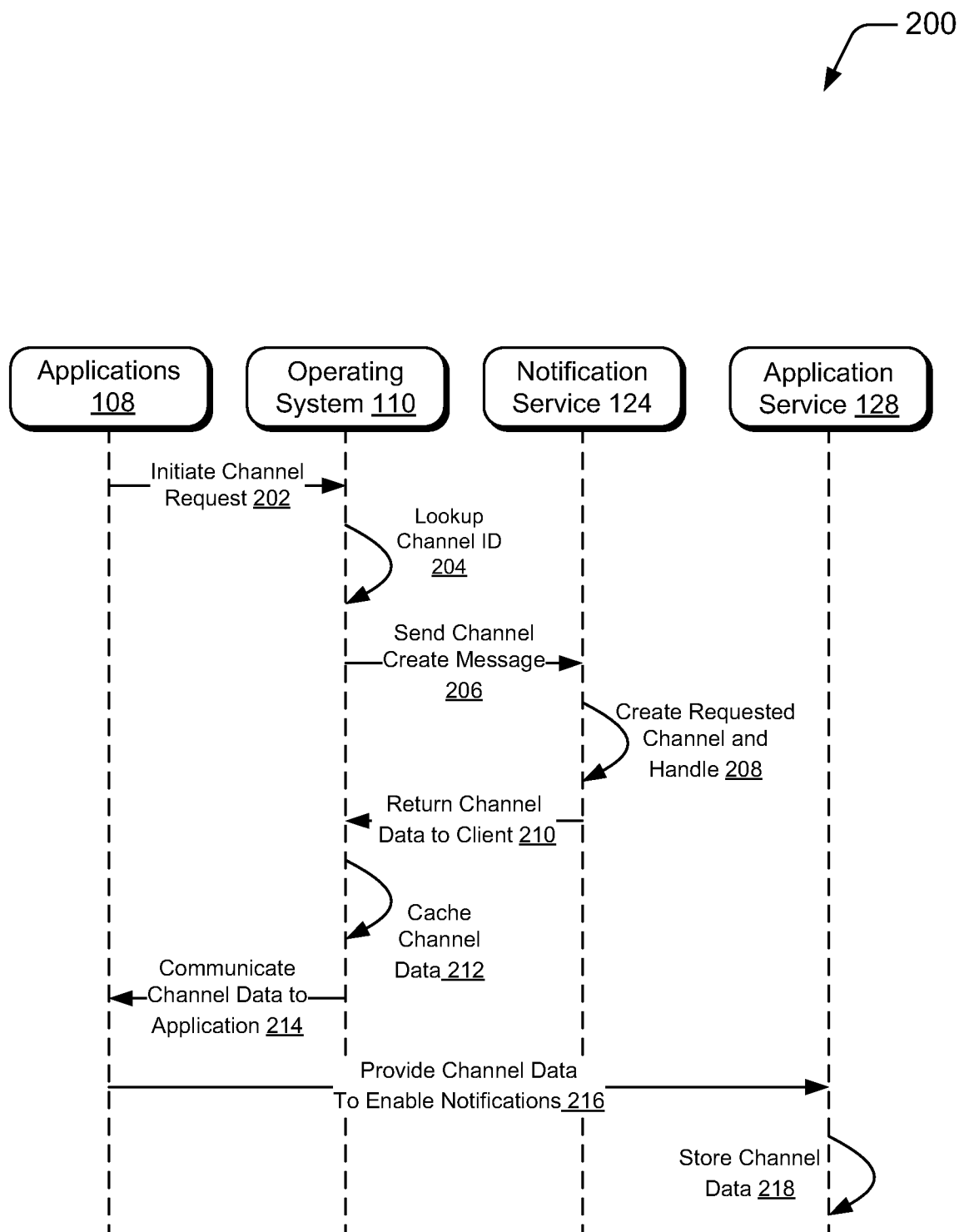
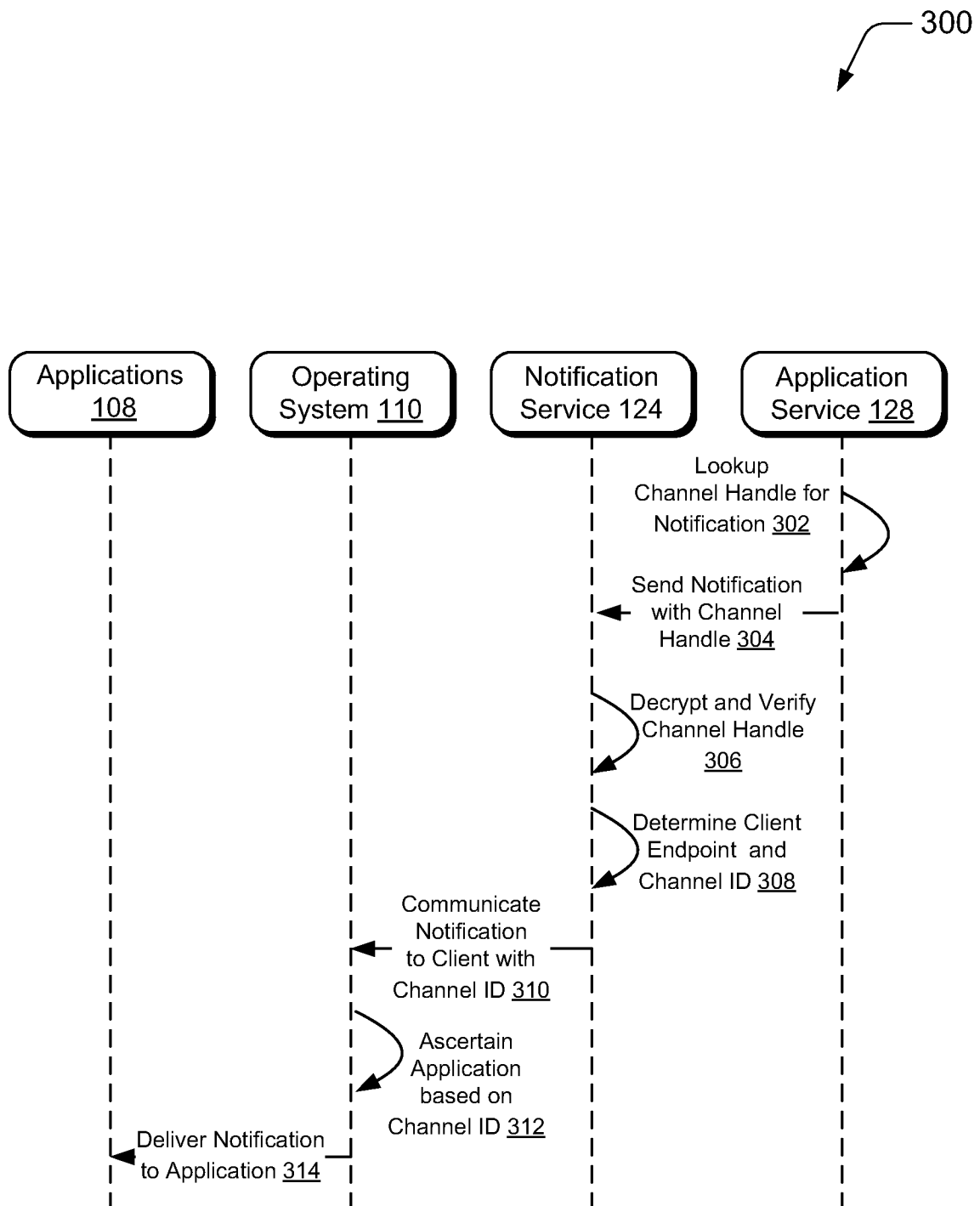
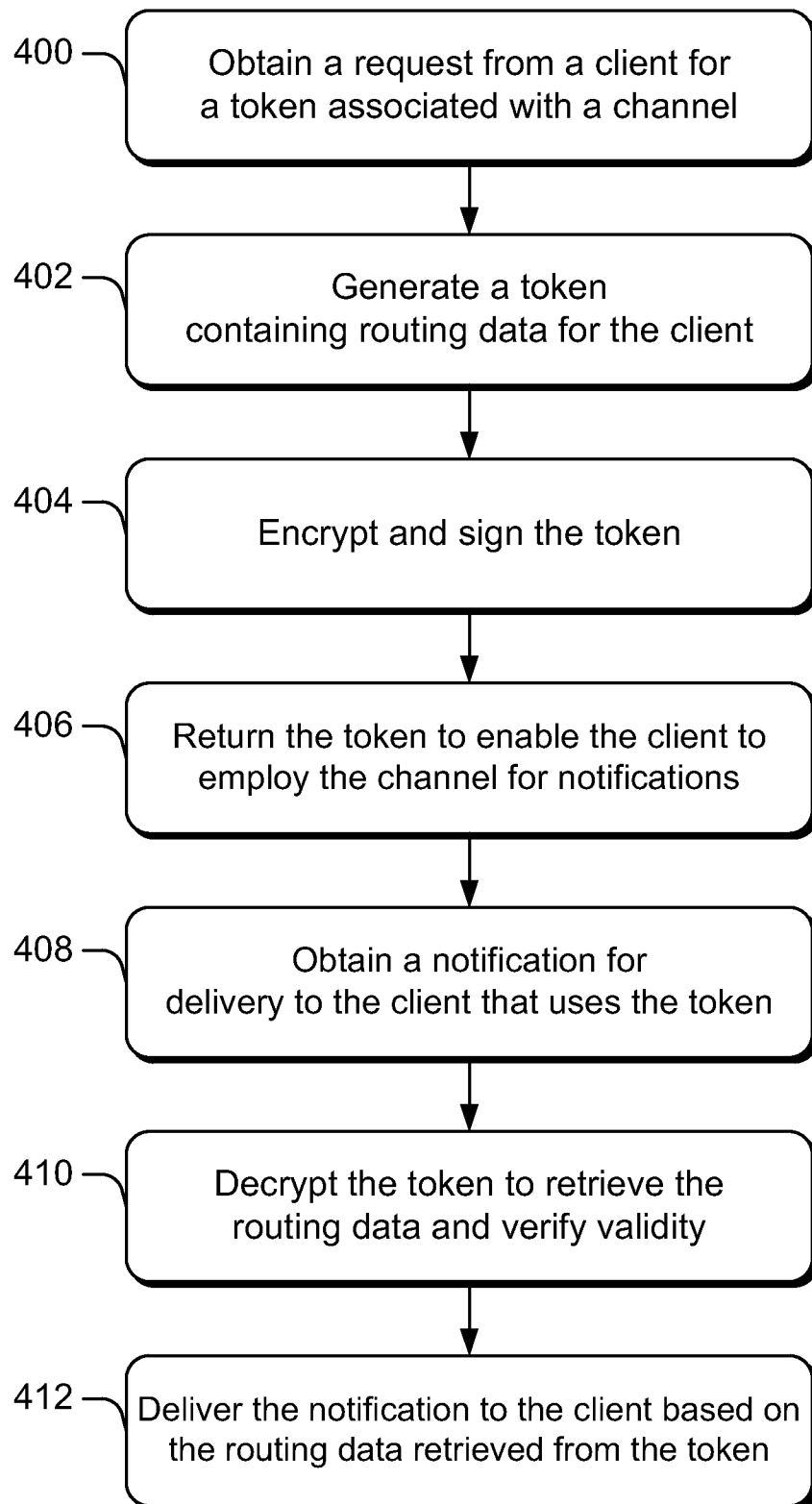


FIG. 2

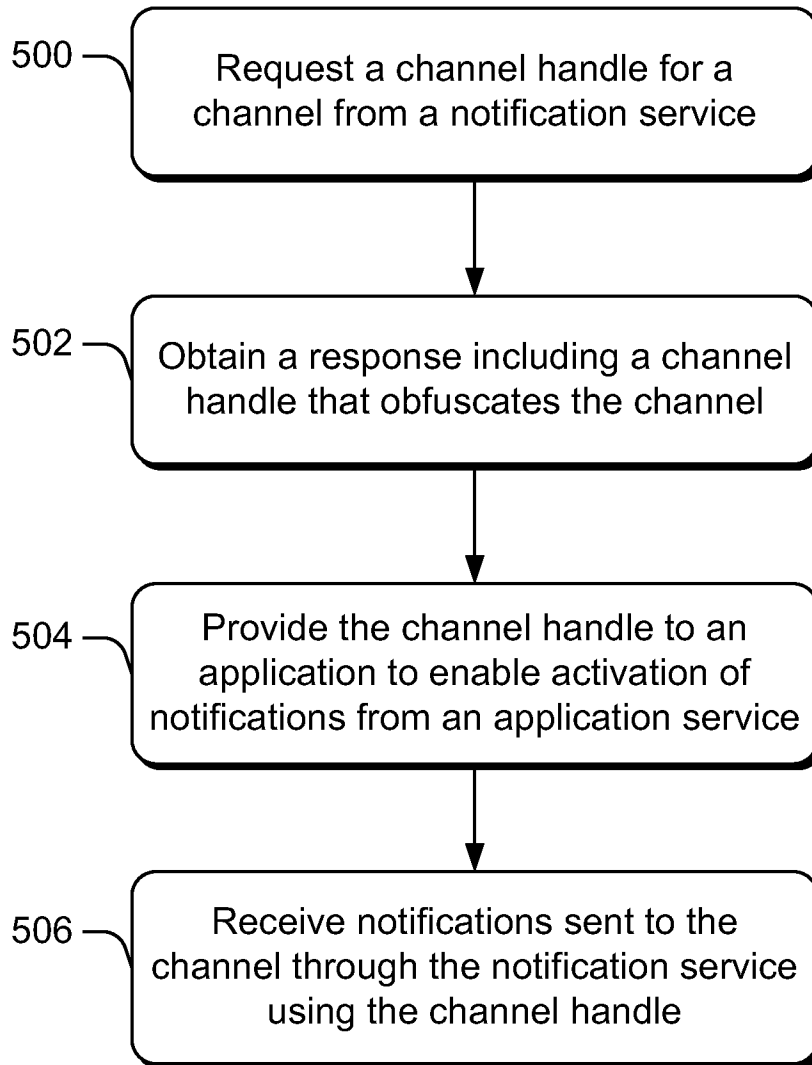
3/7

**FIG. 3**

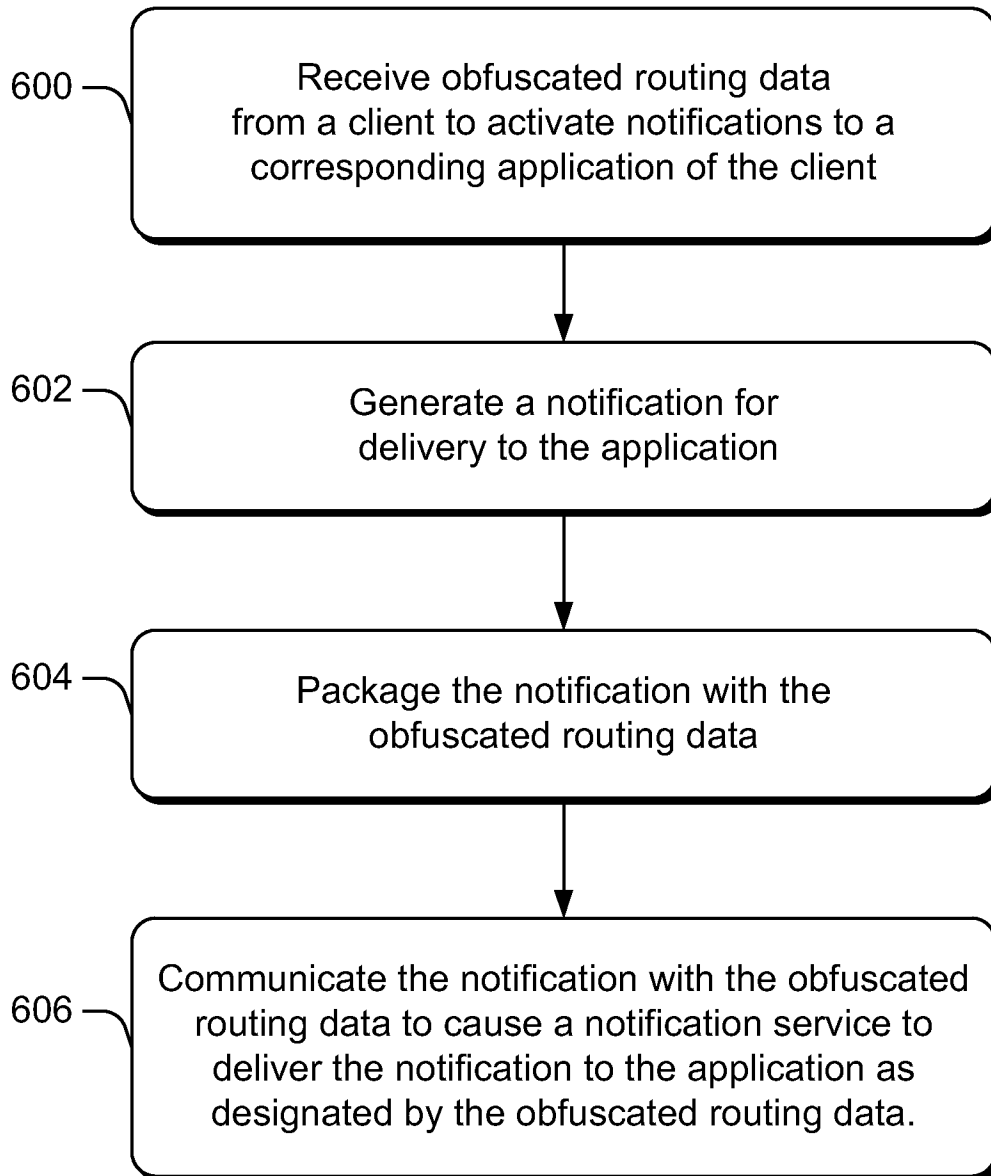
4/7

**FIG. 4**

5/7

**FIG. 5**

6/7

**FIG. 6**

700

700

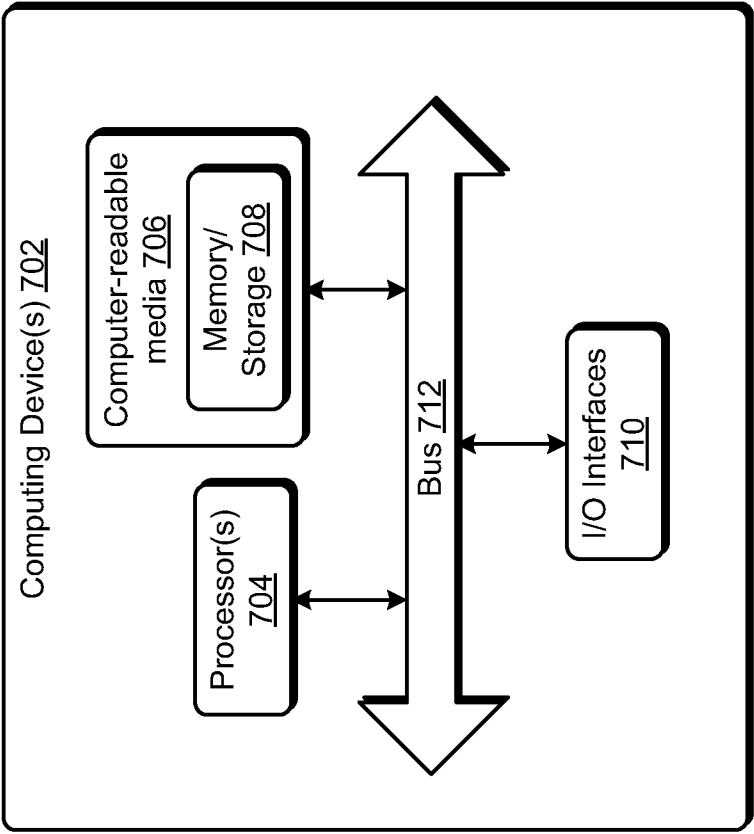


FIG. 7



## INTERNATIONAL SEARCH REPORT

International application No.  
**PCT/US2012/053025****A. CLASSIFICATION OF SUBJECT MATTER*****H04W 68/02(2009.01)i, H04W 12/08(2009.01)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)

H04W 68/02; G06F 15/16; G06F 15/173; H04L 12/28

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) &amp; Keywords: notification, obfuscated, routing path, proxy

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y A	US 2009-0285118 A1 (TAKASHI YOSHIKAWA et al.) 19 November 2009 See paragraphs [0112], [0118], [0120], [0125], [0127], [0163]–[0172].	1 2–10
Y	US 2007-0038740 A1 (RONALD STEEVES) 15 February 2007 See paragraphs [0005], [0006], [0015].	1
A	US 2010-0088387 A1 (PABLO M. CALAMERA) 08 April 2010 See paragraph [0030]; figure 2(b).	1–10

☐ 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 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

"&amp;" document member of the same patent family

Date of the actual completion of the international search

24 DECEMBER 2012 (24.12.2012)

Date of mailing of the international search report

**26 DECEMBER 2012 (26.12.2012)**

Name and mailing address of the ISA/KR

Korean Intellectual Property Office  
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan  
City, 302-701, Republic of Korea

Facsimile No. 82-42-472-7140

Authorized officer

JUNG, Yun Seok

Telephone No. 82-42-481-8123



**INTERNATIONAL SEARCH REPORT**

Information on patent family members

International application No.

**PCT/US2012/053025**

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2009-0285118 A1	19.11.2009	CN 101322108 A	10.12.2008
		CN 101322108 B	14.04.2010
		CN 101322108 C	14.04.2010
		EP 1980950 A1	15.10.2008
		JP 04-701132 B2	11.03.2011
		JP 2007-184892 A	19.07.2007
		KR 10-0987689 B1	13.10.2010
		KR 10-2008-0074960 A	13.08.2008
		US 8179818 B2	15.05.2012
		WO 2007-066738 A1	14.06.2007
US 2007-0038740 A1	15.02.2007	WO 2007-017715 A2	15.02.2007
		WO 2007-017715 A3	15.02.2007
US 2010-0088387 A1	08.04.2010	WO 2010-040092 A1	08.04.2010