



(51) International Patent Classification:
H04L 29/12 (2006.01)

(21) International Application Number:
PCT/CN2013/076973

(22) International Filing Date:
8 June 2013 (08.06.2013)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: MICROSOFT CORPORATION [US/US];
One Microsoft Way, Redmond, Washington 98052-6399
(US).

(72) Inventors; and

(71) Applicants (*for US only*): BLANCO, Ambrosio M.
[US/CN]; c/o Microsoft Asia Pacific R&D Headquarters
14F, Building 2, No 5, Dan Ling Street, Haidian District,
Beijing 100080 (CN). LI, Ming [CN/CN]; c/o Microsoft
Asia Pacific R&D Headquarters 14F, Building 2, No 5,
Dan Ling Street, Haidian District, Beijing 100080 (CN).

(74) Agent: SHANGHAI PATENT & TRADEMARK LAW
OFFICE, LLC; 435 Guiping Road, Shanghai 200233
(CN).

(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, 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, PA, 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,
KM, ML, MR, NE, SN, TD, TG).

Published:

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

(54) Title: APPLICATION FRAMEWORK FOR MULTIPLE DEVICE INTERACTIONS

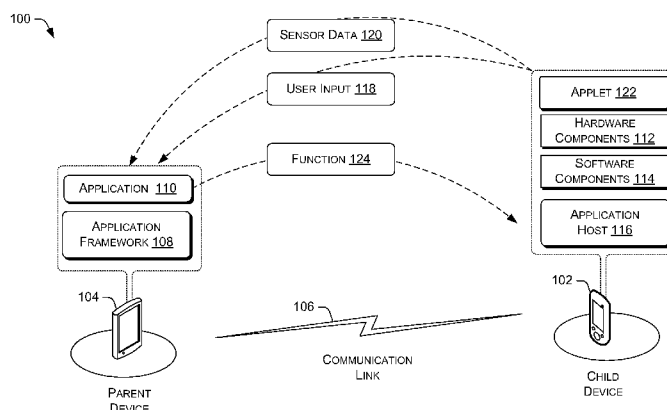


FIG. 1

(57) Abstract: BSTRACT An application framework may be provided on a parent device that enables one or more applications to activate functions that are available on a child device. The parent device may further include a pair component that establishes a communication link between the parent device and a child device that includes an application host. The application host may interface with the application framework to execute the functions on the child device.

APPLICATION FRAMEWORK FOR MULTIPLE DEVICE INTERACTIONS

BACKGROUND

[0001] Consumers are increasingly integrating the use of mobile electronic devices into their daily lives. Such proliferation and use of mobile electronic devices means that a user frequently has access to multiple electronic devices at any time. For example, a user may carry a mobile phone to stay in touch as the user travels around a city, while at the same time carry a tablet device for work or recreational purposes. In another example, a user at home may primarily use a tablet device, but still keep the mobile phone close in case of incoming phone calls. However, such devices generally behave as separate devices, and the user is unable to use a particular mobile device to activate hardware functionalities or software functionalities that are on another mobile device. Thus, while the user is using one mobile electronic device, resources available on the other mobile electronic device, such as a touch screen, a camera, or a display, may remain idle and unused.

SUMMARY

[0002] Described herein are techniques for providing an application framework that enables multiple applications that are installed on a parent device to use functions that are available on a child device. Each of the parent device and the child device may be a mobile electronic device, such as a mobile phone, a tablet computer, a laptop computer, etc. The application framework is an application library that enables multiple applications on the parent device to interact with the functions of the child device as if such functions are located on the parent device. For example, an application on the parent device may use the application framework to obtain a video

feed from a camera that is located on a child device. In another example, an application on the parent device may transfer a user interface control to the child device, such that the application on the parent device may receive inputs from the user interface control on the child device. In an additional example, an application on the parent device may transfer an applet to the child device, such that the applet may function independently from the application and take over at least some of the functions performed by the application on the parent device. The applet may perform such functions even when the application on the parent application device is rendered inactive.

[0003] In at least one embodiment, an application framework may be provided on a parent device that enables multiple applications to activate functions that are available on a child device. The parent device may further include a pair component that establishes a communication link between the parent device and a child device that includes an application host. The application host may interface with the application framework to execute the functions on the child device.

[0004] Accordingly, the application framework may support the collaborative use of multiple mobile devices by providing applications that are installed on a parent device with access to hardware or software functions that are available on a child device. In this way, the application framework may enable a user to tap into previously underutilized computing resources to increase productivity or maximize satisfaction while using applications.

[0005] This Summary is provided to introduce a selection of concepts in a simplified form that is 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.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] The detailed description is described with reference to the accompanying figures. In the figures, the left-most digit(s) of a reference number identifies the figure in which the reference number first appears. The use of the same reference number in different figures indicates similar or identical items.

[0007] FIG. 1 is a block diagram that illustrates an examplescheme for implementing an application framework on a parent device to support the use offunctions on a child device by multiple applications on the parent device.

[0008] FIG. 2 is an illustrative diagram that shows example components of a parent device and a child device that support the implementation of the application framework.

[0009] FIG. 3 is a flow diagram that illustrates an example process for implementing an application framework on a parent device to enable an application on the parent device to access functions of a child device.

[0010] FIG. 4 is a flow diagram that illustrates an example process for establishing a pairing between a parent device that implements an application framework and a child device that provides functions to the parent device.

[0011] FIG. 5 is a flow diagram that illustrates an example process for using an application framework on a parent device to enable an application to receive sensor data from a sensor on a child device.

[0012] FIG. 6 is a flow diagram that illustrates an example process for using an application framework on a parent device to enable an applet on a child device to

perform functions cooperatively with or independently of an application on the parent device.

[0013] FIG. 7 is a flow diagram that illustrates an example process for using an application framework on a parent device to implement a user interface control on a child device that transfers control signals to an application of the parent device.

DETAILED DESCRIPTION

[0014] Described herein are techniques for providing an application framework that enables multiple applications that are installed on a parent device to use functions that are available on a child device. Each of the parent device and the child device may be a mobile electronic device, such as a mobile phone, a tablet computer, a laptop computer, etc. The application framework is an application library that enables multiple applications on the parent device to interact with the features of the child device as if such features are located on the parent device. In various embodiments, the application framework may provide application program interfaces (APIs) that can be called by an application on the parent device.

[0015] In turn, the application framework may communicate with an application host on the child device and command the application host to perform a specified function that is invoked by an API. In some instances, the application host may directly perform the specified function. In other instances, the application host may further instruct an applet that is installed on the child device to perform the specific function. In such instances, the applet may include application code that is developed in conjunction with the application that is on the parent device.

[0016] Accordingly, an application on the parent device may trigger hardware or software components on the child device to perform various tasks for the application. For example, an application on the parent device may use the application framework to obtain video feed from a camera that is located on a child device. In another example, an application on the parent device may transfer a user interface control to the child device, such that the application on the parent device may receive input from the user interface control on the child device. In an additional example, an application on the parent device may transfer an applet to the child device, such that the applet may function independently from the application and take over at least some of the functions performed by the application on the parent device. The applet may perform such functions even when the application on the parent application device is rendered inactive.

[0017] Accordingly, the application framework may support the collaborative use of multiple mobile devices by providing applications that are installed on a parent device with access to hardware or software features that are available on a child device. Further, the application framework may serve as a common platform for multiple applications, such that different application developers may develop their applications to use a common set of APIs rather than develop their own customized application interfaces to achieve the same functional objectives. Such standardized development may reduce application development time, complexity, and cost. Thus, the speed at which developers release applications may be increased.

[0018] The application framework may enable a user to tap into previously underutilized computing resources to increase productivity or maximize satisfaction while using applications. Examples of techniques for using an application framework

to support multiple parent devices accessing the functions on a child device in accordance with various embodiments are described below with reference to FIGS. 1-7.

Example Scheme

[0019] FIG. 1 is a block diagram that illustrates an example scheme 100 for implementing an application framework on a parent device to support the use of functions on a child device 102 by multiple applications on the parent device 104. Each of the parent device 104 and the child device 102 may be a mobile electronic device, such as a mobile phone, a tablet computer, a laptop computer, etc. The parent device 104 and the child device 102 may communicate via a communication link 106. In various embodiments, the communication link 106 may be a Bluetooth connection, a Wi-Fi connection, or another short range radio signal based communication link.

[0020] An application framework 108 may be installed on the parent device 104. The application framework 108 may be an application library that enables multiple applications on the parent device, such as the application 110, to interact with the functions of hardware components 112 or software components 114 of the child device 102 as if such components are located on the parent device. In various embodiments, the application framework 108 may provide application program interfaces (APIs) that can be called by the application 110.

[0021] The application framework 108 may communicate with an application host 116 that is installed on the child device 102. For example, the invocation of an API of the application framework 108 may cause the application framework 108 to command the application host 116 to perform a specified function. In various embodiments, the

application host 116 is a fully functional stand-alone application that is installed on the child device 102.

[0022] In some instances, the application host 116 may directly perform the specified function. For example, the application 110 on the parent device 104 may call an API of the application framework 108 to transfer a user interface control to the child device 102. In turn, the application framework 108 may command the application host 116 on the child device 102 to display the user interface control on a display of the child device 102. Subsequently, a user may use the user interface control to provide a user input 118 to the application host 116. The application host 116 may then return the user input 118 to the application framework 108. The application framework 108 may at this point pass the user input 118 to the application 110. In another example, the application 110 on the parent device 104 may use the API provided by the application framework 108 and the application host 116 to obtain sensor data 120 (e.g., camera feed from a camera) from the child device 102.

[0023] However, in other instances, the application host 116 may further instruct an applet 122 that is installed on the child device to perform the specific function, such as a function 124, that is invoked through the API of the application framework 108. In such instances, the applet 122 may be application code that is transferred to the child device 102 by the application 110.

[0024] For example, the application 110 may be a mapping application that a user used to locate an address of a particular business. The user may command the application 110 to transfer the address to the applet 122 that is installed on the child device 102, along with a request that the applet 122 plot a route from a current location of the user to the address. In at least one embodiment, the applet 122 may be

application code that is automatically transferred to the child device 102 by the application 110 when the application 110 was originally launched by the user.

[0025] In response to the command, the application 110 may invoke an API of the application framework 108 to initiate the transfer of the address information and routing request. The application framework 108 may transfer the address information and the routing request to the application host 116 via the communication link 106. In turn, the application host 116 may provide the address information and the routing request to the applet 122, such that the applet 122 may fulfill the routing request based on the address information. In such an example, the applet 122 may persist and continue to function even after the application 110 becomes inactive.

[0026] Accordingly, the application framework 108 on the parent device 104 and the application host 116 on the child device 102 may function cooperatively to provide applications on the parent device 104 with access to the functions of the child device 102. Further, the parent device 104 and the child device 102 may reverse their roles with respect to certain scenarios. In such scenarios, the parent device 104 may also include an application host, and the child device 102 may have an application framework installed. In this way, applications on the child device 102 may also access the functions that are provided by the hardware and/or software components of the parent device 104.

Example Components

[0027] FIG. 2 is an illustrative diagram that shows example components of a parent device 104 and a child device 102 that support the implementation of the application framework. The parent device 104 may include one or more proximity sensors 202,

communication interface 204, a user interface 206, one or more processors 208, and/or memory 210. The proximity sensors 202 may detect the presence of another mobile device with an application host (e.g., application host 116) that is within a pairing range of the parent device 104, such as the child device 102. In some embodiments, the proximity sensors 202 may include a near field communication (NFC) sensor that is able to detect the presence of another NFC transceiver equipped mobile electronic device. The NFC transceiver of the other mobile electronic device may transmit data that indicates that the device is provided with an application host. In some embodiments, the NFC sensor may be a part of a NFC transceiver.

[0028] In other embodiments, the proximity sensors 202 may include a camera and associated image recognition and processing software. The camera and related software may be able to recognize the visual appearance of another mobile electronic device or read a visual identifier that is affixed to the other electronic device. For example, the proximity sensor 202 may read a barcode, a quick response (QR), alphanumeric text, or other forms of visual identifiers that are affixed to the other mobile electronic device. Each of the visual identifier may be associated with device feature metadata stored in the parent device 104 that indicates a corresponding device is equipped with an application host.

[0029] In still other embodiments, the proximity sensor 202 may be another type of radio signal detector that listens for other forms of radio signal broadcasts from nearby mobile electronic devices. The radio signal broadcasts may include data that indicates the mobile electronic devices are equipped with application hosts. The radio signal detector may be capable of distinguishing between multiple mobile electronic devices based on the broadcast identifiers, broadcast signal strength, broadcast frequencies,

and/or other broadcast properties. In some embodiments, the proximity sensor 202 may be a part of a Bluetooth transceiver or a Wi-Fi transceiver.

[0030] The communication interface 204 may include wireless and/or wireless communication interface components that enable the parent device 104 to transmit and receive data via a network or a communication link. In various embodiments, the wireless interface components may include, but is not limited to a cellular transceiver, a NFC transceiver, a Wi-Fi transceiver, a Bluetooth transceiver, and/or so forth. As such, the wireless interface component may enable the parent device 104 to establish direct communication links, such as the communication link 106, with another mobile electronic device. The wired interface components may include a direct input/output (I/O) interface, such as an Ethernet interface, a serial interface, a Universal Serial Bus (USB) interface, an IEEE 1394 interface, and/or so forth. Furthermore, the wireless and wired interface components may enable the parent device 104 to exchange data with other electronic devices (*e.g.*, laptops computers, servers, etc.) via one or more networks, such as the Internet.

[0031] The user interface 206 may include a data output device (*e.g.*, a visual display, audio speakers, touch display), and one or more data input devices. The data input devices may include, but are not limited to, combinations of one or more of keypads, keyboards, mouse devices, touch screens that accept gestures, microphones, voice or speech recognition devices, and any other suitable devices or other electronic/software selection methods.

[0032] The memory 210 may be implemented using computer-readable media, such as computer storage media. Computer-readable media includes, at least, two types of computer-readable media, namely computer storage media and communication media.

Computer storage media includes volatile and non-volatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules, or other data. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other tangible medium that may be used to store information for access by a computing device. In contrast, communication media may embody computer readable instructions, data structures, program modules, or other data in a modulated data signal, such as a carrier wave, or other transmission mechanism. As defined herein, computer storage media does not include communication media.

[0033] The memory 210 of the parent device 104 may implement an operating system 212, the application framework 108, applications 214(1)-214(N), a pairing module 218, an authentication module 220, and a data store 222. The operating system 212 may include components that enable the parent device 104 to receive data via various inputs (e.g., user controls, network interfaces, and/or memory devices), and process the data using the processors 208 to generate output. The operating system 212 may further include one or more components that present the output (e.g., display an image on an electronic display, store data in memory, transmit data to another electronic device, etc.). The operating system 212 may enable a user to interact with the applications 214(1)-214(N) using the user interface 206, as well perform tasks for the applications 214(1)-214(N). Additionally, the operating system 212 may include other components that perform various other functions generally

associated with an operating system, such as supporting the execution of the various modules stored in the memory 210.

[0034] Each of the applications 214(1)-214(N) may be an application that is programmed to use the application framework 108 to activate the functions of the hardware components 112 and/or the software components 114 of the child device 102. In various embodiments, an application may be designed to call APIs of the application framework 108. Each of the APIs may enable the application to activate a particular hardware function or software function on the child device 102. In this way, the applications 214(1)-214(N) may be developed according to the use a common set of APIs. As such, application design and development time may be substantially reduced.

[0035] In some embodiments, one or more of the applications 214(1)-214(N) may be developed to include a corresponding applet. For example, the application 214(N) may include an applet 216. An applet of an application may be application code embedded within the application that is designed to be transferred to the child device 102 by the application. Once an applet is transferred to the child device 102, the applet may work cooperative with the application host 116 to perform a function on the child device 102 for the application. For example, the application host 116 may invoke an applet to instantiate a user interface control on the child device 102 that can be used to send inputs back to the application. In some instances, an applet may function independently from the parent application that transferred the applet to the child device 102. For example, an applet that is a navigation applet may continue to function to provide navigation directions even when the associated parent navigation application has been inactivated on the parent device 104.

[0036] The application framework 108 may provide application program interfaces (APIs) that can be called by the applications 214(1)-214(N). The APIs may enable the applications 214(1)-214(N) to interact with the functions of the hardware components 112 and/or the software components 114 of the child device 102. For example, the APIs may include an API that enables an application on the parent device 104 to access sensors on the child device 102. Another API of the application framework 108 may be called by an application to instantiate user interface controls on the child device 102, such that the application may receive inputs back from the child device 102. In another example, an API may enable an application on the parent device 104 to activate a hardware feature and/or software feature of the child device 102 (e.g., send a text message, power off, erase stored data, etc.).

[0037] The application framework 108 may command the application host 116 on the child device 102 to perform the functions that correspond to the APIs. In some embodiments, the application host 116 may be capable of directly accessing the hardware components 112 and/or the software components 114 of the child device 102 to perform a function. However, in other instances, the application host 116 may further use an applet that is provided by an application to perform a function.

[0038] The pairing module 218 may pair the parent device 104 with a child device 102. In some embodiments, a user may activate the pairing module 218 by performing an action. For example, the user may bring the child device 102 to within a predetermined range of the parent device 104, such that the proximity sensor 202 may detect the presence of and/or recognize the child device 102. Subsequently, the proximity sensor 202 may activate the pairing module 218 to negotiate and establish the communication link 106 with the child device 102. The communication link 106

may be a NFC link, a Bluetooth link, a Wi-Fi link, or another form of radio signal-based link. In various embodiments, the communication link 106 may be encrypted to prevent eavesdropping by other devices that are not parties to the communication link 106. For example, the communication link 106 may be encrypted using Bluetooth encryption, Wired Equivalent Privacy (WEP) encryption, Wi-Fi Protected Access (WPA) encryption, and/or so forth.

[0039] In at least one embodiment, the pairing module 218 may pair with the child device 102 only when the child device 102 has an application host, such as the application host 116, installed on the child device 102. The pairing module 218 may determine that a child device includes an application host based on information provided by the child device during initial detection by the proximity sensor 202 and/or negotiation with the pairing module 218 to establish the communication link 106. For example, the child device may provide device feature metadata that indicates an application host is present on the child device. In another example, the pairing module 218 may compare the identification information of the child device as detected by the proximity sensor 202 against a list of identification information for child devices that are known to have application hosts installed.

[0040] In other embodiments, the proximity sensor 202 may generate an alert prompt that alerts a user to the presence of a child device that includes an application host within a pairing range of the parent device 104. The alert prompt may be presented on a display of the parent device 104. Accordingly, the user may select a confirmation option of the alert prompt to initiate pairing using the pairing module 218, or select a cancel option of the alert prompt to dismiss the alert prompt. In still other embodiments, the proximity sensor 202 may automatically activate the pairing

module 218 to initiate pairing with a child device that includes an application host when such a child device is within the pairing range.

[0041] In some instances, the pairing module 218 may use the authentication module 220 to regulate the security of the communication link 106 between the parent device 104 and the child device 102. In some embodiments, the pairing module 218 may use the authentication module 220 to display a message when the pairing module 218 is ready to pair the child device 102 with the parent device 104. The message may include identification information of the child device 102 (e.g., an identifier, a description, etc.), as well as a prompt for the user to either authorize the pairing or terminate the pairing.

[0042] Alternatively or concurrently, the authentication module 220 may also prompt a user of the parent device 104 to input a user name and/or another authenticator prior to establishing the communication link 106. The authenticator may be a password, an authentication token that is obtained by the parent device 104 via a hardware input/output interface, biometric information that is inputted via an imaging component or an audio component that include in or connected with the parent device 104, and/or so forth. The authentication module 220 may compare the user name and/or authenticator to a database of authorized user names and/or authenticators that are stored in the data store 222. In this way, the authentication module 220 may permit the pairing module 218 to establish the communication link 106 when an inputted password and/or an inputted authenticator match corresponding information in the database. In other instances, the authentication module 220 may also use other combinations of single factor or multi-factor authentication, and the above mentioned authentication credentials are illustrative rather than limiting. However, the use of the

authentication module 220 by the pairing module 218 is optional. In some embodiments, the pairing module 218 may establish the communication link between the parent device 104 and the child device 102 without the use of the authentication module 220. Instead, the responsibility of controlling access to the functions of the hardware components 112 and/or the software components 114 of the child device 102 once the communication link 106 is established may fall to the applications 214(1)-214(N) themselves.

[0043] As described above, the application framework 108, the pairing module 218, and the authentication module 220 may be implemented as separate entities that are installed in an operating system environment provided by the operating system 212. However, in other embodiments, one or more of these components may be integrated into the operating system 212. Such integration may provide a compact software package that enables multiple applications on the parent device to interact with the functions provided by the hardware components 112 or software components 114 of the child device 102 as if such components are located on the parent device 104.

[0044] The data store 222 may store information that is used by the various components on the parent device 104. In at least one embodiment, the data store 222 may store a list of identification information for child devices that are known to have application hosts installed. Alternatively or concurrently, the data store 222 may also store authentication information that is used by the authentication module 220.

[0045] The child device 102 may include a communication interface 224, a user interface 226, and sensors 228. Each of the communication interface 224 and the user interface 226 may be similar in function to the communication interface 204 and the user interface 206, respectively. The sensors 228 may include a camera, a microphone,

an accelerometer, a compass, a gyroscope, a global positioning system (GPS) locator, and so forth.

[0046] The memory 230 may be implemented using computer-readable media, such as computer storage media. Computer-readable media includes, at least, two types of computer-readable media, namely computer storage media and communication media. Computer storage media includes volatile and non-volatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules, or other data. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other tangible medium that may be used to store information for access by a computing device. In contrast, communication media may embody computer readable instructions, data structures, program modules, or other data in a modulated data signal, such as a carrier wave, or other transmission mechanism. As defined herein, computer storage media does not include communication media.

[0047] The memory 230 of the parent device 104 may implement an operating system 232, the application host 116, a pairing module 234, and an authentication module 236. The memory 230 may also store applets 238(1)-238(N). The operating system 232 may include components that enable the child device 102 to receive data via various inputs (e.g., user controls, network interfaces, and/or memory devices), and process the data using the processors 208 to generate output. The operating system 232 may further include one or more components that present the output (e.g.,

display an image on an electronic display, store data in memory, transmit data to another electronic device, etc.). The operating system 232 may enable a user to interact with the applets 238(1)-238(N) using the user interface 206, as well perform tasks for the applets. Additionally, the operating system 232 may include other components that perform various other functions generally associated with an operating system, such as supporting the execution of the various modules stored in the memory 230.

[0048] The application host 116 may be a stand-alone application stored in the memory 230. The application host 116 may receive commands from the application framework 108 on the parent device 104 via the communication link 106. In turn, the application host 116 may implement functions as directed by the application framework 108. For example, the application host 116 may activate a camera on the child device 102 to obtain a camera feed. The application host 116 may then provide the camera feedback to an application that is requesting the camera feed via the application framework 108.

[0049] In some embodiments, the application host 116 may direct an applet, such as one of the applets 238(1)-238(N), to perform functions that are commanded by the application framework 108 on the parent device 104. For example, the application host 116 may be commanded by the application framework 108 to present a user interface control on a screen of the child device 102. The application host 116 may lack the software access or capability to directly instantiate the user interface control. In such an instance, the application framework 108 may be configured to call an applet that is installed on the child device 102 to present the user interface control on the screen of the child device 102. However, in alternative instances, the application

host 116 may have the ability to instantiate the user interface control on a display of the child device 102 without intervention from any applets. Accordingly, the above example with respect to the use of the applet for the purpose of displaying a user interface control on the child device 102 is illustrative rather than limiting.

[0050] Each of the applets 238(1)-238(N) may be an applet that is developed in correspondence to an application that is intended for installation on the parent device 104. An applet may be transferring to child device 102 by a corresponding application on the parent device 104 through the use of the application framework 108 and the application host 116. Each of the transferred applets 238(1)-238(N) may be stored in a cache 240 that resides in the memory 230. In one instance, when the application launches, the application may request that the application framework 108 send a corresponding applet to the memory 230 of the child device. The request may include an applet identifier that is assigned to the applet (e.g., name, digital certificate, hash value, etc.). The application framework 108 may ask the application host 116 to verify whether a corresponding applet is already installed in the memory 230 of the child device 102. The application host 116 may perform the verification by ascertaining whether the applet is already presented within the cache 240. Such verification may be performed by comparing the applet identifier in an request to the applet identifiers of the applets that are in the cache 328.

[0051] Thus, if the applet is already present in a cache 240 of the memory 230, the application host 116 may return a present indicator to the application framework 108. In turn, the application framework 108 may cancel the request and direct the application to use the applet that is already in the memory 230 to perform functions. However, if the applet is not present in the cache 240, the application host 116 may

return an unavailable indicator to the application framework 108. The application framework 108 may then initiate a transfer of the applet to the cache 240 of the child device on behalf of the requesting application. For instance, the application framework 108 may transfer the applet to the application host 116, so the application host 116 may store the applet in the cache 328. In this way, unnecessary transmission of multiple versions of the applets 238(1)-238(N) may be reduced or eliminated to reduce processor usage and communication link bandwidth usage.

[0052] In some embodiments, an applet may be configured to function independently on the child device 102 without commands from a corresponding application on the parent device 104. For example, an applet that is a navigation applet may continue to function to provide navigation directions even when the associated parent navigation application has been inactivated on the parent device 104. In such an example, a user may use the parent navigation application on the parent device 104 to locate a business that the user wants to visit, and then plot a route of travel to the business. The user may further transfer the address of the business and the plotted route from the parent navigation application on the parent device 104 to the navigation applet on the child device 102. At this point, the user may turn off the parent navigation application by shutting down the parent device 104. Nevertheless, the navigation applet on the child device 102 may continue to function to provide turn-by-turn directional guidance to the business according to the plotted route, even though its communication connection to the parent navigation application was severed due to the parent navigation application being inactivated.

[0053] In at least one embodiment, the application host 116 may flush the applets 238(1)-238(N) when the application host 116 is commanded to become inactive. For

example, a user may direct the application host 116 to shut down via the user interface 226. At such a point, the application host 116 may delete the applets 238(1)-238(N) from the cache 240 and then terminate.

[0054] The pairing module 234 may work cooperatively with the pairing module 218 of the parent device 104 to form the communication link 106 with the parent device 104. In various embodiments, the pairing module 234 may respond to link negotiation queries from the pairing module 218 with negotiation replies, protocol information, channel information, frequency information, device identifiers, the presence of the application host 116, and/or other information for establishing the communication link 106. The communication link 106 may be a NFC link, a Bluetooth link, a Wi-Fi link, or another form of radio signal-based link. The pairing module 234 may be under control of the application host 116, such that the application host 116 may establish a functional relationship with the application framework 108.

[0055] In some instances, the pairing module 234 may use the authentication module 236 to regulate the security of the communication link 106 between the parent device 104 and the child device. The use of the authentication module 236 by the pairing module 234 may be concurrent or in the alternative to the use of the authentication module 220 by the pairing module 218.

[0056] In some embodiments, the pairing module 234 may use the authentication module 236 to display a message when the pairing module 236 is ready to pair the child device 102 with the parent device 104. The message may include identification information of the parent device 104 (e.g., an identifier, a description, etc.), as well as a prompt for the user to either authorize the pairing or terminate the pairing. Thus, in

some scenarios in which the authentication module 236 and the authentication module 220 are concurrently used on both the parent device 104 and the child device 102, both devices are to receive authorization from the user before the communication link 106 may be established.

[0057] Alternatively or concurrently, the authentication module 236 may also prompt a user of the child device 102 (which may also be the user of the parent device 104) to input a user name and/or another authenticator prior to establishing the communication link 106. The authenticator may be a password, an authentication token that is obtained by the child device 102 via a hardware input/output interface, biometric information that is inputted via an image component or an audio component that is included in or connected with the child device 102, and/or so forth. The authentication module 236 may compare the user name and/or authenticators that are stored in a database.

[0058] In this way, the authentication module 236 may permit the pairing module 234 to establish the communication link 106 when an inputted password and/or an inputted authenticator match corresponding information in the database via single factor or multiple factor authentication. Once again, when the authentication module 236 is being implemented concurrently with the authentication module 220 for authentication, the pairing module 234 and the pairing module 218 may establish the communication link 106 when the authentication module 220 also permits pairing. However, in other instances, the responsibility of controlling access to the functions of the hardware components 112 and/or the software components 114 of the child device 102 once the communication link 106 is established may once again fall to the applications 214(1)-214(N) themselves.

[0059] However, the implementation of authentication on both the child device 102 and the parent device 104 in order to establish the communication link 106 may prove to be burdensome for some users. Thus, in some alternative embodiments, a corresponding authentication cache may be implemented on each of the child device 102 and the parent device 104. In such embodiments, the authentication module 220 may provide the user with the option of saving device identifiers of devices that were previously successfully paired with the parent device 104 to a corresponding authentication cache on the parent device 104. Likewise, the authentication module 236 may provide the user with the option of saving device identifiers of devices that were previously successfully paired with the child device 102 to a corresponding authentication cache on the child device 102. In at least one embodiment, each of the device identifiers may be a dynamically generated globally unique identifier (GUID) that mitigates spoofing attacks.

[0060] Accordingly, when the child device 102 and the parent device 104 are paired for the first time, the authentication module 220 and/or the authentication module 236 may prompt the user with a prompt dialog that enables the user to select between an option to pair once and an option to always pair. If the user selects the option to pair once, the user will be prompted to authenticate to the parent device 104 and/or the child device 102 at each subsequent pairing attempt. However, if the user selects the option to always pair, each authentication module may save the device identifier of the other device to be paired in a corresponding authentication cache, such that the device identifiers are used to automatically perform mutual authentication to establish a communication link between the parent device 102 and the child device 104.

[0061] For example, when the parent device 104 detects the child device 102, the pairing modules on both devices may exchange device identifiers. As such, when the authentication modules on both devices determine that a corresponding received device identifier matches a device identifier that is stored in a corresponding authentication cache, the authentication modules may authorize their respective pairing modules to establish a communication connection without further input or command from the user.

[0062] While the components of the child device 102 and the parent device 104 have been described, a device may function as a child device with respect to some mobile electronic devices, while as a parent device with respect to other mobile electronic devices. Thus, in some embodiments, a device may include components from both the parent device 104 and the child device 102.

Example Processes

[0063] FIGS. 3-7 describe various example processes for enabling the collaborative use of multiple mobile devices. The order in which the operations are described in each example process is not intended to be construed as a limitation, and any number of the described operations may be combined in any order and/or in parallel to implement each process. Moreover, the operations in each of the FIGS. 3-7 may be implemented in hardware, software, and/or a combination thereof. In the context of software, the operations may represent computer-executable instructions that, when executed by one or more processors, cause one or more processors to perform the recited operations. The one or more processors may be included in individual computing devices or included in multiple computing devices that are part of a cloud.

Generally, computer-executable instructions include routines, programs, objects, components, data structures, and so forth that cause the particular functions to be performed or particular abstract data types to be implemented. In other embodiments, the operations of each example process may be executed by a hardware logic circuit, such as a dedicated integrated circuit.

[0064] FIG. 3 is a flow diagram that illustrates an example process 300 for implementing an application framework on a parent device to enable an application on the parent device to activate functions of a child device. At block 302, the application framework 108 may be implemented on the parent device 104. In some embodiments, the application framework 108 may be installed by a corresponding installation program that executes on parent device 104. The application framework 108 is an application library that enables multiple applications on the parent device 104 to interact with one or more functions of the child device 102 as if such functions are located on the parent device.

[0065] At block 304, an application host 116 may be installed on the child device 102. The application host 116 may be installed by a corresponding installation program that executes on the child device 102. The application host 116 may interface with the application framework 108 to direct performance of the functions on the child device 102. In various embodiments, the application host 116 may receive function commands from the application framework 108 and implement the functions associated with the function commands. Alternatively, the application host 116 may direct an applet, such as the applet 122, to perform the functions associated with the function commands.

[0066] At block 306, the communication link 106 between the parent device 104 and the child device 102 may be established for the application framework 108 to interface with the application host 116. The communication link 106 may enable the application framework 108 on the parent device 104 to transfer function commands and/or applets to the child device 102. The function commands may enable an application on the parent device 104 to interact with the functions of the hardware components 112 and/or the software components 114 of the child device 102.

[0067] FIG. 4 is a flow diagram that illustrates an example process 400 for establishing a pairing between a parent device that implements an application framework and a child device that provides functions to the parent device. The process 400 may further illustrate block 306 of the process 300.

[0068] At block 402, the application framework 108 on a parent device 104 may search for a child device that includes an application host, such as the application host 116. In some embodiments, the application framework 108 may perform a search for a predetermined period of time when activated by a user. For example, the user may input a command using the user interface of the 206 of the parent device 104. In other embodiments, the application framework 108 may perform the search at predetermined time intervals (e.g., every ten seconds) or at random time intervals, in which each search lasts a pre-designated period of time.

[0069] At decision block 404, the application framework 108 may determine whether a child device that includes an application host is discovered. In various embodiments, the application framework 108 may make such a determination based on identification information and/or device feature metadata provided by the child device. Thus, if the application framework 108 determines that no child device with an

application host is discovered (“no” at decision block 404), the process 400 may return to block 402. Upon return to block 402, the application framework 108 may continue the search for a child device that includes an application host. However, if the application framework 108 determines that a child device that includes an application host is discovered (“yes” at decision block 404), the process 400 may proceed to decision block 406.

[0070] At decision block 406, the application framework 108 may determine whether authentication for the purpose of establishing a communication connection with child device is to be performed. In various embodiments, the application framework 108 may make the determination based on one or more configuration settings that are inputted using the user interface 206 and/or the user interface 226. Thus, if the application framework 108 determines that authentication is to be used (“yes” at decision block 406), the process 400 may proceed to block 408.

[0071] At block 408, the application framework 108 may establish a communication connection that includes the use of authentication. The authentication may be a single-factor authentication or a multiple-factor authentication that is based on the use of various user credentials and/or device credentials. In various embodiments, the authentication may be performed through the input of user credentials at the parent device 104 and/or the child device 102. For example, a user may be prompted by one or both of the authentication module 220 on the parent device 104 and the authentication module 236 on the child device 10 to input user credentials in order to establish the communication link. In another example, the authentication modules on the parent device 104 and the child device 102 may use device identifiers that are stored in the authentication caches of the devices to perform mutual device

authentication and automatically establish the communication link if the mutual device authentication is successful. In still other embodiments, the authentication may be performed by an application installed on the parent device 104, such as an application 214(N).

[0072] However, if at decision block 406 the application framework 108 determines that no authentication is to be used (“no” at decision block 406), the process 400 may proceed to block 410. At block 410, the application framework 108 may establish a communication connection without the use of authentication.

[0073] FIG. 5 is a flow diagram that illustrates an example process 500 for using an application framework on a parent device to enable an application to receive sensor data from a sensor on a child device. At block 502, the application framework 108 on the parent device 104 may receive a command from an application, such as the application 110, to obtain sensor data from a sensor on the child device 102. The sensor may be one of the sensors 228, which may include a camera, a microphone, an accelerometer, a compass, a gyroscope, a global positioning system (GPS) locator, and so forth. The application may provide the command to the application framework 108 by invoking an API of the application framework 108.

[0074] At block 504, the application framework 108 may send the command to activate the sensor to the application host 116 on the child device 102. The command may be sent via the communication link 106 that is established between the parent device 104 and the child device 102. At block 506, the application host 116 on the child device 102 may receive the command from the application framework 108. The command may be interpreted by the application host 116 as a request to activate and obtain sensor data from a designated sensor on the child device 102.

[0075] At block 508, the application host 116 may activate the designated sensor on the child device. For example, the sensor may be a camera on the child device, and the sensor data may be a video feed from the camera. At block 510, the application host 116 may receive the sensor data from the sensor. The application host 116 may encode the sensor data for transmission back to the application framework 108.

[0076] At block 512, the application host 116 may send the sensor data back to the application framework 108 on the parent device 104 via the communication link 106. At block 514, the application framework 108 may receive the sensor data from the application host 116. At block 516, the application framework 108 may forward the sensor data to the application that is on the parent device 104.

[0077] FIG. 6 is a flow diagram that illustrates an example process 600 for using an application framework on a parent device to enable an applet on a child device to perform functions cooperatively with or independently of an application on the parent device. At block 602, the application framework 108 may initiate the transfer of an applet of the application on the parent device 104 to the child device 102. For example, the applet may be the applet 122 of the application 110. The application framework 108 may initiate the transfer of the applet upon a request of the application. The application may make such a request when launched for execution by a user.

[0078] At block 604, the application host 116 on the child device 102 may receive an indication that the transfer is initiated. In turn, the application host 116 may determine whether the applet of the application already exists on the child device 102. The application host 116 may make the determination based on an identifier of the applet. Thus, at decision block 606, if the application host 116 determines that the

application is already present in the cache 240 of the child device 102 (“yes” at decision block 606), the process 600 may continue to block 608.

[0079] At block 608, the application host 116 may direct the application framework 108 to inform the application on the parent device 104 to use the existing applet. The direction may be in the form of an available indicator that is passed to the application framework 108. At block 610, the application framework 108 may receive the direction to use the existing applet and inform the application on the parent device 104.

[0080] At block 612, the application on the parent device 104 may send configuration information from the application to the application host 116. In various embodiments, the configuration information may be information that commands the applet to perform a certain task independently of the application on the parent device 104. For example, the configuration information may be routing data for a navigation applet such that the navigation applet may provide turn-by-turn directions even when the application is rendered inactive. Alternatively or concurrently, the configuration information may command the applet to work cooperatively with the application in order to perform one or more functions.

[0081] At block 614, the application host 116 on the child device 102 may receive the configuration information from the application framework 108. At block 616, the application host 116 may provide the configuration information to the applet on the child device 102. In this way, the applet may perform functions independently of and/or cooperatively with the application on the parent device 104 based on the configuration data.

[0082] Returning to decision block 606, if the application host 116 determines that the application is absent from the cache 240 of the child device 102 (“no” at decision block 606), the process 600 may continue to block 618. At block 618, the application host 116 may accept the transfer of the applet from the parent device 104 to the child device 102. In other words, the application host 116 may download the applet into the cache 240 of the child device 102. Subsequently, the process 600 may proceed to block 612, such that the application on the parent device 104 may send configuration information from the application to the application host 116.

[0083] FIG. 7 is a flow diagram that illustrates an example process 700 for using an application framework on a parent device to implement a user interface control on a child device that transfers control signals to an application of the parent device. At block 702, the application framework 108 on the parent device 104 may receive a command from an application, such as the application 110, to display a user interface control on child device 102. The application may provide the command to the application framework 108 by invoking an API of the application framework 108.

[0084] At block 704, the application framework 108 may send the command to display the user interface control the application host 116 on the child device. The command may be sent via the communication link 106 that is established between the parent device 104 and the child device 102. At block 706, the application host 116 on the child device 102 may receive the command from the application framework 108. The command may be interpreted by the application host 116 as a request to display a user interface control on the child device 102.

[0085] At block 708, the application host 116 may present the user interface control on a display of the child device. For example, the user interface control may be a

control that enables a user to control an application on the parent device 104 (e.g., interact with a gaming application) from the child device 102. At block 710, the application host 116 may detect an activation of the user interface control. The activation of the user interface control may result from a user providing a user input using the user interface control.

[0086] At block 712, the application host 116 may send a control signal corresponding to the activation to the application framework 108 on the parent device 104. In various embodiments, the application host 116 may transmit the control signal to the application framework 108 via the communication link 106. At block 714, the application host 116 may receive the control signal from the application host 116. The control signal may direct the application on the parent device 104 to perform a function. At block 716, the application host 116 may forward the control signal to the application on the parent device 104. In turn, the application may perform the function that corresponds to the control signal.

[0087] In alternative embodiments, the application host 116 may be incapable of implementing the display of a user interface control on the child device 102 on its own. Instead, the application host may rely on the use of an applet that corresponds to the application to display the user interface control on the child device 102. In such embodiments, steps that are similar to blocks 602-614 and block 618 of the process 600 may be implemented by the application host 116 to use an applet to present the user interface control on the display of the child device 102.

[0088] In summary, an application framework may be provided on a parent device that enables multiple applications to activate features that are available on a child device. In this way, the application framework may support the collaborative use of

multiple mobile devices. In this way, the application framework may enable a user to tap into previously underutilized computing resources to increase productivity or maximize satisfaction while using applications.

Conclusion

[0089] In closing, although the various embodiments have 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 representations is not necessarily limited to the specific features or acts described. Rather, the specific features and acts are disclosed as exemplary forms of implementing the claimed subject matter.

CLAIMS

What is claimed is:

1. One or more computer-readable media storing computer-executable instructions that are executed to cause one or more processors to perform acts comprising:

receiving a request from an application to activate a function on a child device, the request being received at an application framework of a parent device;

sending a command from the application framework to an application host on a child device that activates the function, the application host interfaces with the application framework via a communication link between the parent device and the child device; and

receiving feedback data produced by the function on the child device from the application host at the application framework.

2. The one or more computer-readable media of claim 1, wherein the receiving the request includes receiving the request from the application in response to the application invoking an application program interface (API) of the application framework.

3. The one or more computer-readable media of claim 1, wherein the receiving the request includes receiving a request from the application to obtain sensor data from a sensor of the child device.

4. The one or more computer-readable media of claim 3, wherein the sensor includes a camera, a microphone, a compass, an accelerometer, a gyroscope, or a global positioning system (GPS) locator.

5. The one or more computer-readable media of claim 1, wherein the receiving includes receiving a request from the application to present a user interface control on

a display of the child device, wherein the sending includes transmitting a command to the application host to cause the application host to present the user interface control on a display of the child device.

6. The one or more computer-readable media of claim 5, further comprising:
receiving a control signal that corresponds to an activation of the user interface control; and
forwarding the control signal to the application that requested the presentation of the user interface control on the display of the child device.

7. The one or more computer-readable media of claim 5, wherein the transmitting includes transmitting the command to the application host such that the application host displays the user interface control or directs an applet on the child device that is associated with the application to display the user interface control.

8. The one or more computer-readable media of claim 1, further comprising:
sending an applet that is associated with the application to the child device via the communication link between the parent device and the child device; and
providing configuration information to the applet that enables the applet to function independently of the application following an inactivation of the application.

9. The one or more computer-readable media of claim 8, wherein the sending the applet includes sending the applet in response to determining that the applet is absent from a cache on the child device.

10. The one or more computer-readable media of claim 1, further comprising:
detecting the child device that includes the application host based on a radio signal broadcast or a visual identifier associated with the child device; and
establishing the communication link between the parent device and the child device.

11. The one or more computer-readable media of claim 10, wherein the establishing includes establishing the communication link in response to receiving a valid user authentication credential at one or more of the parent device and the child device, or establishing the communication link based on device identifiers that are stored in the parent device and the child device.

12. The one or more computer-readable media of claim 1, wherein the communication link is a near field communication (NFC) link, a Bluetooth link, or a Wi-Fi Link.

13. A computer-implemented method, comprising:

implementing an application framework on a parent device, the application framework enabling one or more applications of the parent device to activate a function on a child device;

installing an application host on the child device, the application host interfaces with the application framework to activate the function on the child device; and

establishing a communication link between the parent device and the child device for the application framework to interface with the application host.

14. The computer-implemented method of claim 13, further comprising detecting the child device that includes the application host based on a radio signal broadcast or a visual identifier associated with the child device, and wherein the establishing includes establishing the communication link in response to a detection of the child device or a pairing command inputted into the parent device following the detection.

15. The computer-implemented method of claim 13, further comprising:

receiving a command from an application of the one or more applications at the application framework to activate a function on the child device;

sending the command from the application framework to the application host via the communication link;

receiving, via the communication link, a feedback from the application host following an execution of the function on the child device; and

forwarding the feedback to the application of the multiple applications.

16. The computer-implemented method of claim 13, further comprising sending an applet that is associated with the application to the child device via the communication link between the parent device and the child device, wherein the application host directs the applet to perform the function.

17. The computer-implemented method of claim 16, wherein the applet executes the function on the child device independently of the application following an inactivation of the application.

18. A parent device, comprising:

one or more processors;

a memory that includes a plurality of computer-executable components that are executable by the one or more processors, comprising:

an application framework that enables one or more applications of the parent device to activate a function on a child device; and

a pairing component that establishes a communication link between the parent device that includes the application framework and a child device that includes an application host, the application host interfaces with the application framework to activate the function on the child device.

19. The parent device of claim 18, wherein the application framework receives a feedback from the application host following an execution of the function on the child device, and forwards the feedback to the application on the parent device.

20. The parent device of claim 18, wherein the application transfers an applet that is associated with the application to the child device via the communication link, wherein the application host directs the applet to execute the function on the child device independently of the application following an inactivation of the application.

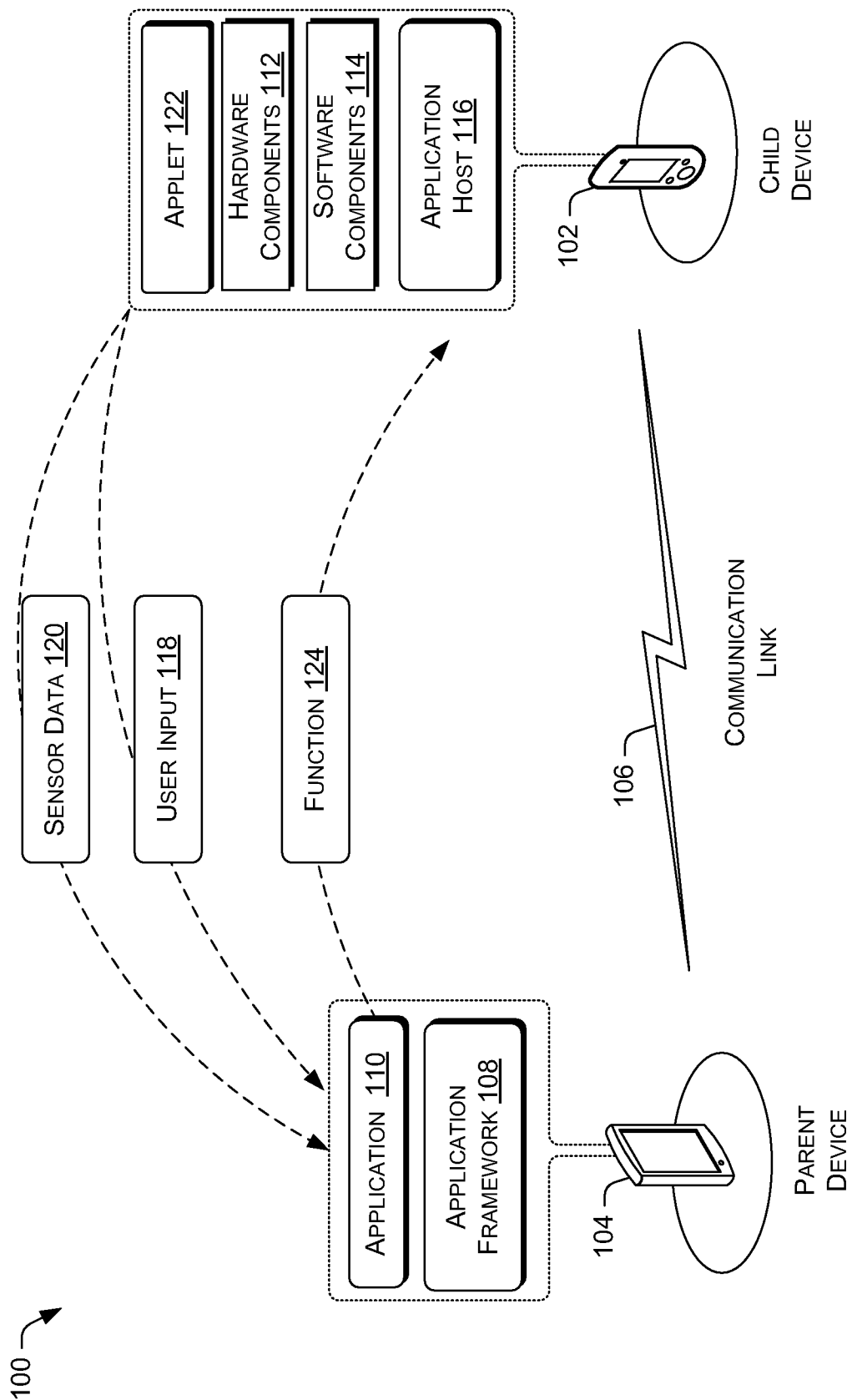


FIG. 1

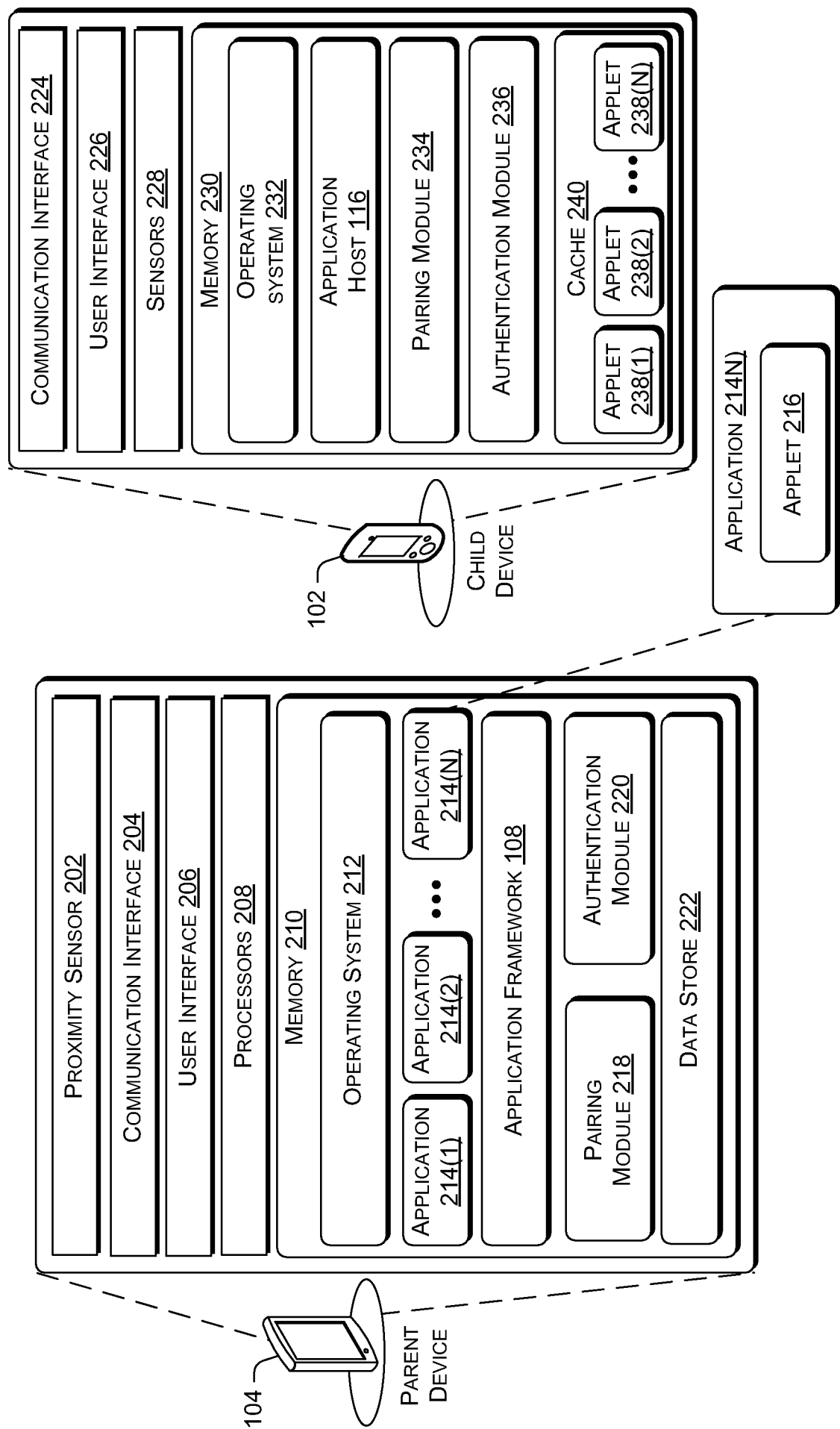


FIG. 2

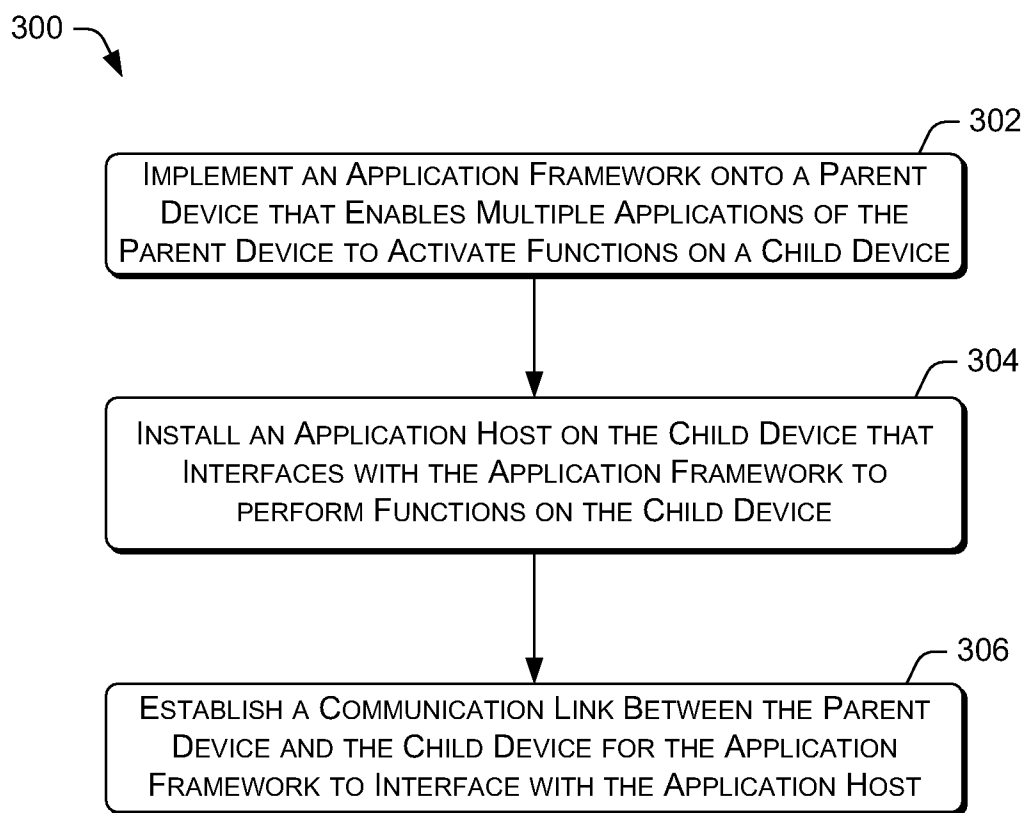


FIG. 3

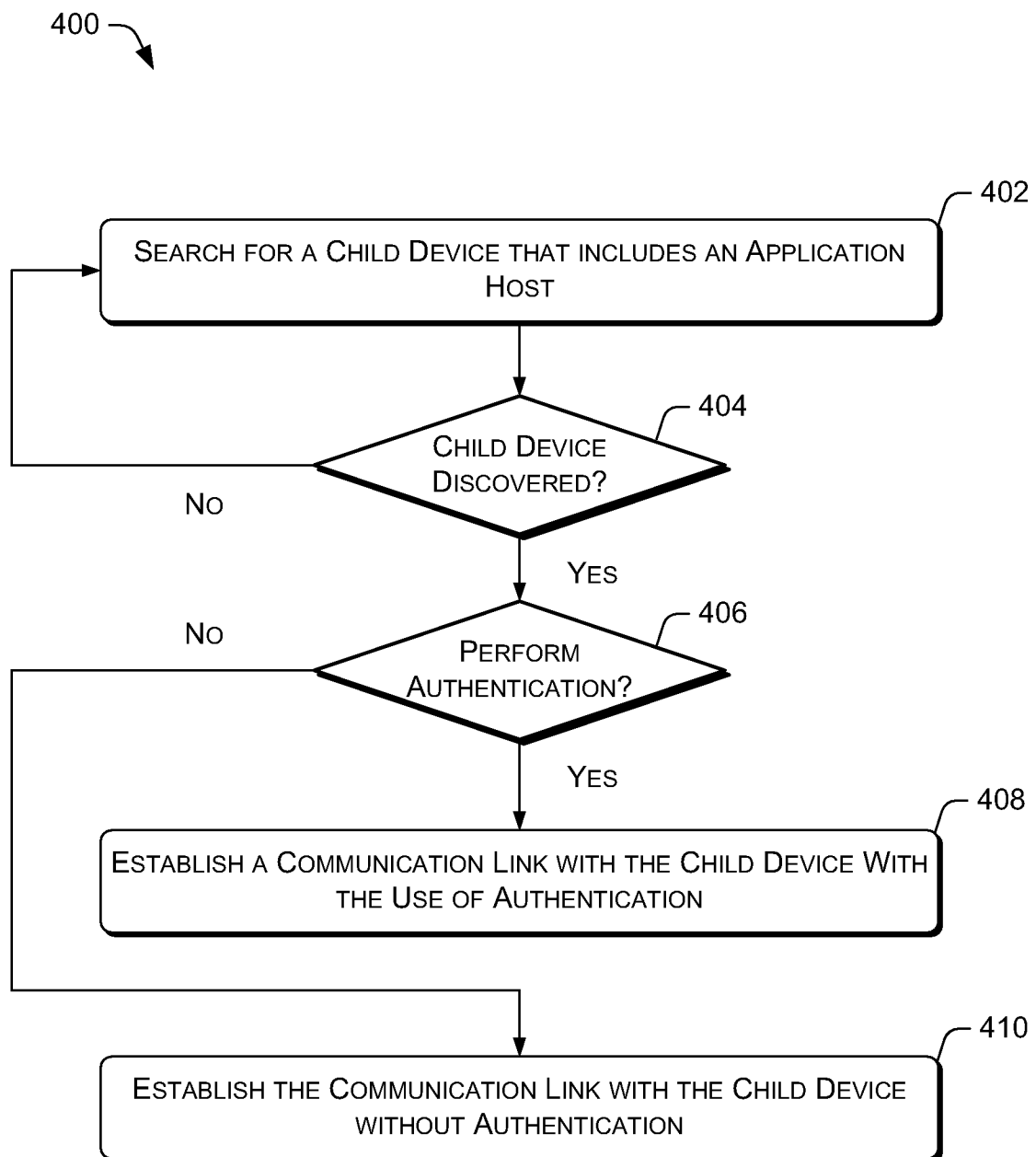


FIG. 4

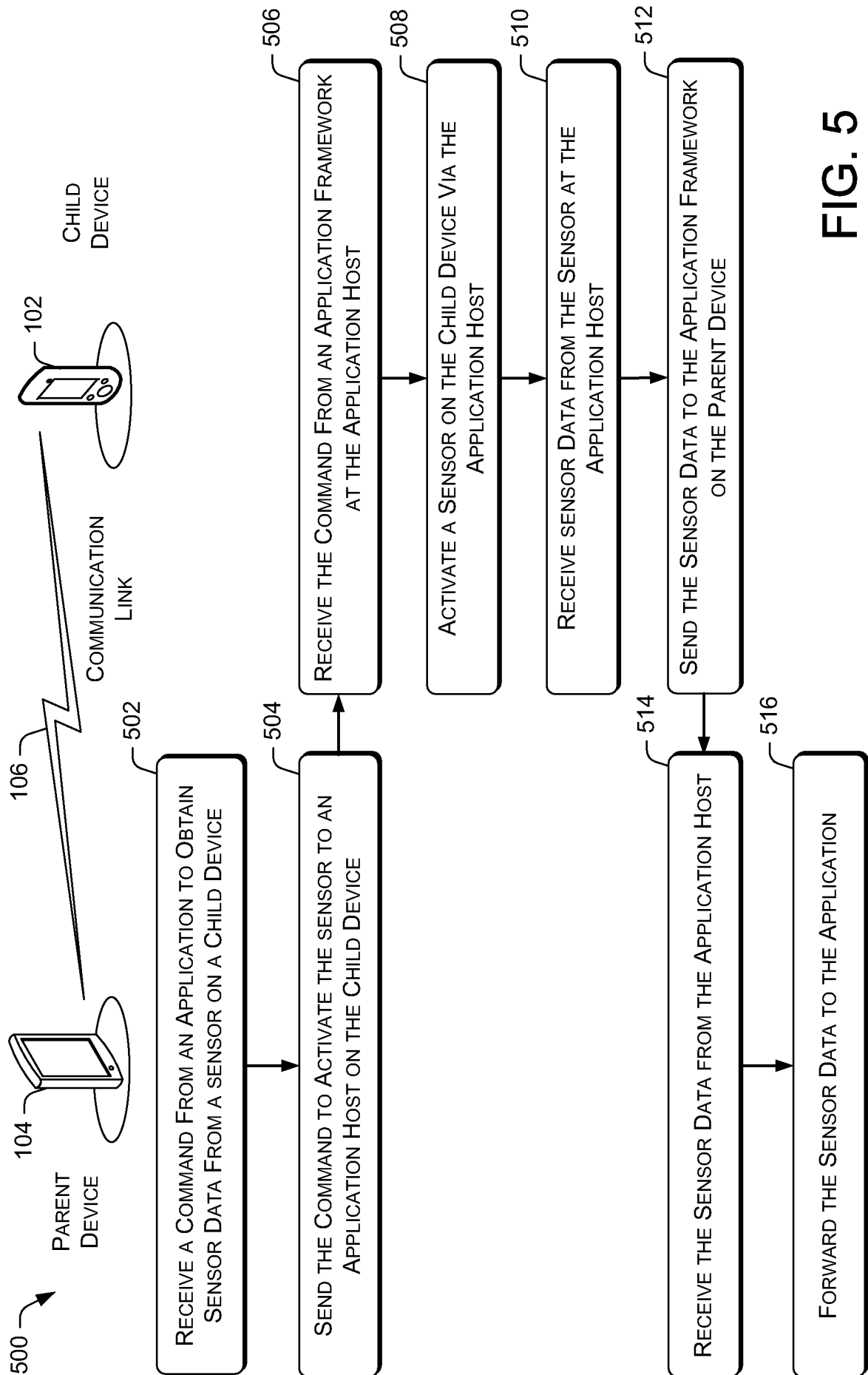


FIG. 5

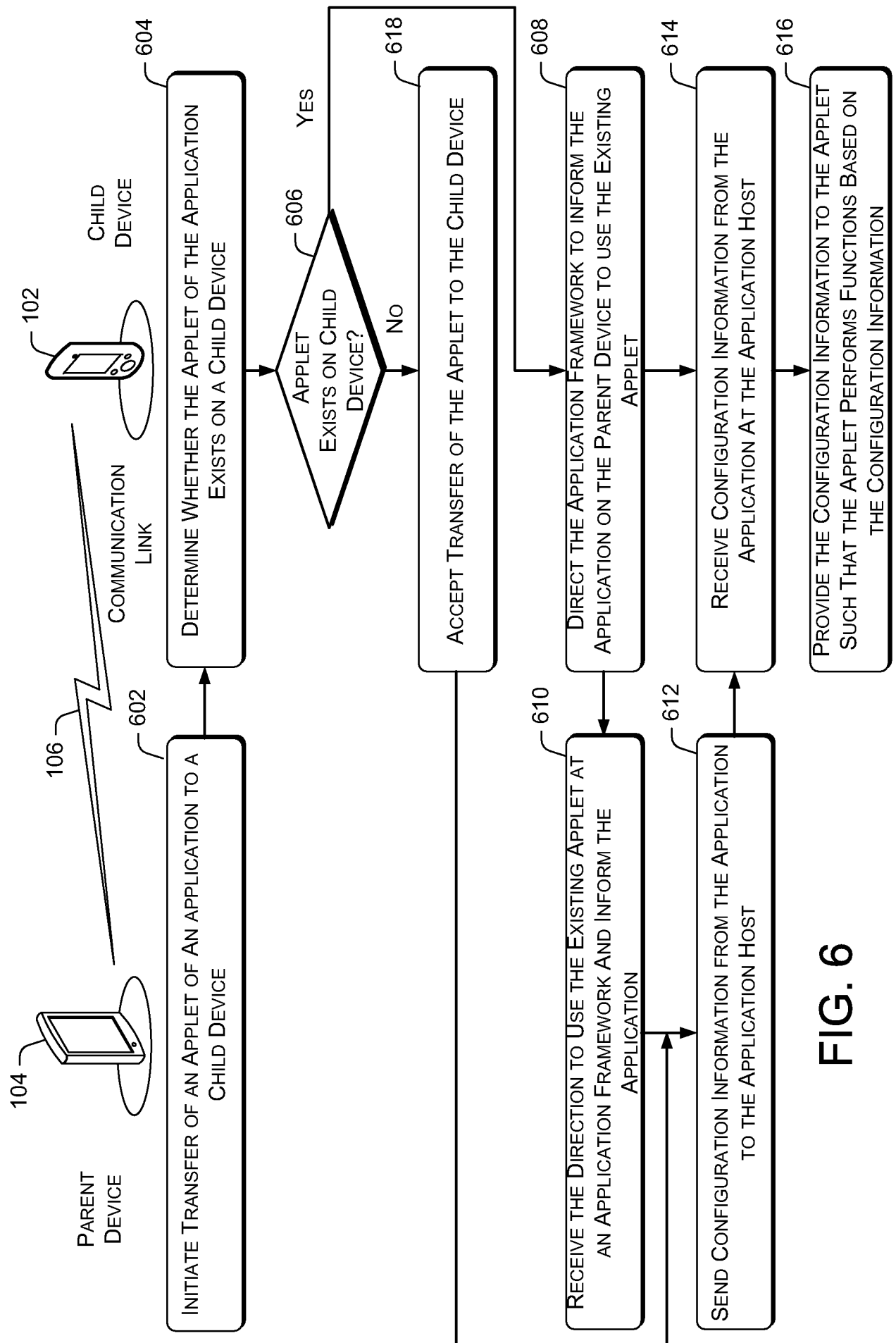


FIG. 6

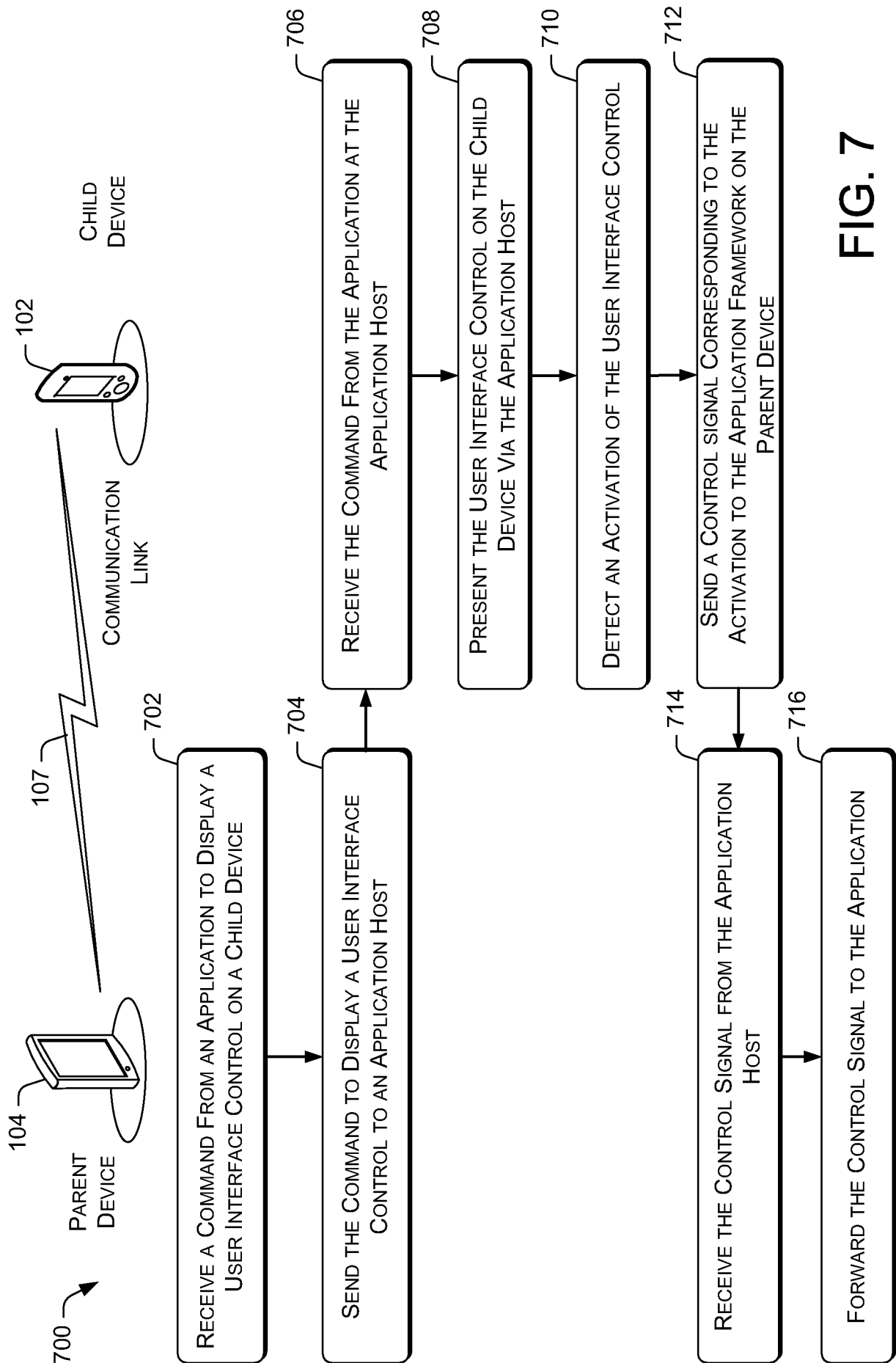


FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2013/076973

A. CLASSIFICATION OF SUBJECT MATTER

H04L 29/12 (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)

IPC: H04L, G06F, H04W, H04Q

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

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

CNPAT, WPI, EPODOC, CNKI, IEEE: application, request, interact, function, applet, parent, child, slave, device, apparatus, client, equipment, another, second, remote, collaborative, execute, activate, trigger

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2006161704 A1 (NYSTAD, Jorn et al.) 20 July 2006 (20.07.2006) description, paragraphs [0002]-[0252] and figure 2	1-20
A	US 2003043787 A1 (EMERSON III, Harry E.) 06 March 2003 (06.03.2003) the whole document	1-20
A	US 6807593 B1 (LSI LOGIC CORPORATION) 19 October 2004 (19.10.2004) the whole document	1-20

☐ Further documents are listed in the continuation of Box C.☒ See patent family annex.

* Special categories of cited documents:	“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
“A” document defining the general state of the art which is not considered to be of particular relevance	“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
“E” earlier application or patent but published on or after the international filing date	“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
“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)	“&” document member of the same patent family
“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	

Date of the actual completion of the international search 20 February 2014 (20.02.2014)	Date of mailing of the international search report 13 Mar. 2014 (13.03.2014)
--	--

Name and mailing address of the ISA/CN
The State Intellectual Property Office, the P.R.China
6 Xitucheng Rd., Jimen Bridge, Haidian District, Beijing, China
100088
Facsimile No. 86-10-62019451

Authorized officer

CHEN, Chen

Telephone No. (86-10)62413305

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2013/076973

Patent Documents referred in the Report	Publication Date	Patent Family	Publication Date
US 2006161704 A1	20.07.2006	WO 2004066059 A2	05.08.2004
		GB 2398657 A	25.08.2004
		EP 1586040 A2	19.10.2005
		DE 602004030757 E	10.02.2011
		EP 2284715 A2	16.02.2011
		AT 493709 T	15.01.2011
		US 2013106870 A1	02.05.2013
US 2003043787 A1	06.03.2003	WO 03021388 A2	13.03.2003
		AU 2002323462 A1	18.03.2003
US 6807593 B1	19.10.2004	None	