

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
16 July 2009 (16.07.2009)

PCT

(10) International Publication Number
WO 2009/088728 A2

(51) International Patent Classification:
H04J 3/22 (2006.01)

(74) Agent: **WIENER, Stewart M.**; 1303 East Algonquin Road, Schaumburg, Illinois 60196 (US).

(21) International Application Number:
PCT/US2008/087926

(81) Designated States (*unless otherwise indicated, for every kind of national protection available*): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, 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, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(22) International Filing Date:
22 December 2008 (22.12.2008)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/969,648 4 January 2008 (04.01.2008) US

(71) Applicants (*for all designated States except US*): **MO-TOROLA, INC.** [US/US]; 1303 East Algonquin Road, Schaumburg, Illinois 60196 (US). **CHINTADA, Suresh Kumar** [IN/IN]; 224, Shine On Block 2, Rahat Bagh, Bangalore 560093 (IN).

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

(72) Inventors; and

(75) Inventors/Applicants (*for US only*): **P, Sethuramalingam** [IN/IN]; B1-705, White House, Apartments, RT Nagar, Bangalore 560032 (IN). **GOFFIN, Glen P.** [US/US]; 106 Parkside Drive, Dublin, Pennsylvania 18917 (US).

Published:

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

(54) Title: AN EXTENSIBLE SYSTEM AND METHOD TO BRIDGE SIP AND UPNP DEVICES

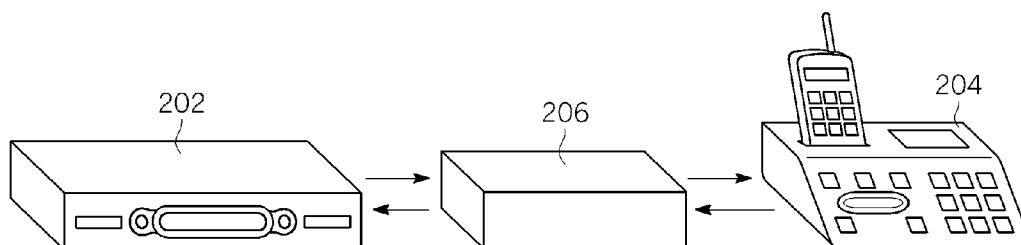


FIG. 2

200

(57) Abstract: Embodiments of the invention generally provide a method and apparatus for bridging session initiation protocol and universal plug and play devices. One embodiment of the invention specifies a method that enables legacy session initiation protocol and universal plug and play devices to communicate with each other (*i.e.*, to access services), where the term 'legacy' is defined to mean IETF RFC 3261-compliant for session initiation protocol devices, and DLNA 1.0-compliant for universal plug and play devices. The method enables inter-working between the session initiation protocol and universal plug and play devices without requiring changes to the legacy devices. One embodiment of the invention is a transparent software bridge that enables these features.



WO 2009/088728 A2

AN EXTENSIBLE SYSTEM AND METHOD TO BRIDGE SIP AND UPnP DEVICES

FIELD OF THE INVENTION

[0001] The present invention generally relates to home and mobile networking applications, and more particularly relates to Session Initiation Protocol and Universal Plug and Play technologies.

BACKGROUND OF THE INVENTION

[0002] Session Initiation Protocol (SIP) and Universal Plug and Play (UPnP) are two protocols that are widely deployed in home and mobile networking applications. While UPnP is traditionally popular in the home consumer space, SIP tends to be popular in both home and mobile environments. Although attempts have been made to bridge the SIP and UPnP technologies, such attempts have been limited by the capabilities of the devices.

[0003] FIG. 1, for example, is a schematic diagram illustrating a system 100 in which an exemplary UPnP device 102 (e.g., an audio/visual server) and an exemplary SIP device 104 (e.g., a cordless phone) are attempting to communicate. One approach in which the SIP device 104 and the UPnP device 102 may be bridged is via a home gateway that requires SIP extensions (e.g., Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions, or SIMPLE) on the SIP device 104. A second approach requires either a UPnP stack on the SIP device 104, or a SIP stack on the UPnP device 102. Thus, existing solutions are limited to situations in which the end device(s) have a required capability even to enable minimal interoperability. In other words, these solutions are not extensible to legacy devices that may already be deployed in a user's network. A legacy SIP device is a single-mode device that only supports the Internet Engineering Task Force (IETF) Request for Comments (RFC) 3261 SIP features. A legacy UPnP device is a single-mode device that only supports UPnP capabilities as specified by Digital Living Network Alliance (DLNA) 1.0 guidelines.

[0004] Moreover, existing approaches to bridging the SIP and UPnP technologies typically do not support bidirectional internetworking. That is, while the approaches enable an SIP device (with extensions) to access UPnP services, no attempts have been made to make a service offered by a user communication device (e.g., an SIP device) accessible to a UPnP network.

[0005] Therefore, there is a need in the art for a method and apparatus for bridging Session Initiation Protocol and Universal Plug and Play devices.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] So that the manner in which the above recited embodiments of the invention are attained and can be understood in detail, a more particular description of the invention may be had by reference to the embodiments thereof which are illustrated in the appended drawings. It is to be noted, however, that the appended drawings illustrate only typical embodiments of this invention and are therefore not to be considered limiting of its scope, for the invention may admit to other equally effective embodiments.

[0007] FIG. 1 is a schematic diagram illustrating a system in which an exemplary UPnP device and an exemplary SIP device are attempting to communicate;

[0008] FIG. 2 is a schematic diagram illustrating one embodiment of a system for bridging Session Initiation Protocol and Universal Plug and Play devices

[0009] FIG. 3 is a flow diagram illustrating one embodiment of a method for bridging Session Initiation Protocol and Universal Plug and Play devices;

[0010] FIG. 4 is a flow diagram illustrating a more detailed embodiment of a method for bridging Session Initiation Protocol and Universal Plug and Play devices;

[0011] FIG. 5 is a flow diagram illustrating a more detailed embodiment of a method for bridging Session Initiation Protocol and Universal Plug and Play devices;

[0012] FIG. 6 is a flow diagram illustrating one embodiment of a method for registering, locating and invoking a UPnP service; and

[0013] FIG. 7 is a high level block diagram of the present software bridge that is implemented using a general purpose computing device.

[0014] To facilitate understanding, identical reference numerals have been used, where possible, to designate identical elements that are common to the figures.

DETAILED DESCRIPTION

[0015] FIG. 2 is a schematic diagram illustrating one embodiment of a system 200 for bridging Session Initiation Protocol and Universal Plug and Play devices. In the illustrated embodiment, a legacy UPnP device 202 (e.g., an audio/visual server) and a legacy SIP device 204 (e.g., a cordless phone) communicate via a software bridge 206, for example embodied in a personal computer, a residential gateway, a cellular telephone, or in a dedicated, standalone device (e.g., a set top box). The software bridge 206 operates transparently. That is, the software bridge 206 makes the legacy SIP device 204 look transparent to the legacy UPnP device 202, and vice versa, so that the legacy SIP device 204 and the legacy UPnP device 202 operate as they normally would. Once the software bridge 206 is implemented, the legacy SIP device 204 and the legacy UPnP device 202 may communicate directly or communicate using the software bridge 206 as an intermediary.

[0016] As described in further detail below, the software bridge 206 relies on overloading the features of RFC 3261 SIP and on common applications. No additional protocol features, changes to the SIP specification or new application features are required for the software bridge 206 to successfully allow the legacy UPnP device 202 and the legacy SIP device 204 to communicate. As also discussed further below, the software bridge 206 enables the legacy UPnP device 202 to access session services by relaying session services offered by the legacy SIP device 204 to the UPnP network, thus facilitating bidirectional internetworking.

[0017] FIG. 3 is a flow diagram illustrating one embodiment of a method 300 for bridging Session Initiation Protocol and Universal Plug and Play devices. The method 300 thus illustrates the functionality of the transparent software bridge 206 illustrated in FIG. 2, and may therefore be implemented, for example, at a residential gateway or at a standalone accessory to an SIP device.

[0018] The method 300 is initialized at step 302 and proceeds to step 304, where the software bridge receives a user request that requires the bridging of at least one SIP device and at least one UPnP device. For example, the user may want to stream music from a UPnP-based home media server to an SIP cordless phone.

[0019] In step 306, the software bridge determines whether the minimal capability for legacy devices involved in the requested action in the request action is satisfied. As described above, a legacy SIP device is a single-mode device that only supports RFC3261 SIP features, and a legacy UPnP device is a single-mode device that only supports UPnP capabilities as specified by DLNA 1.0 guidelines.

[0020] If the software bridge concludes in step 306 that the minimal capability for the legacy devices involved in the requested action is not satisfied, the method 300 proceeds to step 310, and the software bridge bridges the devices normally (e.g., via a home gateway, SIP stack on UPnP device or UPnP stack on SIP device) before terminating in step 312.

[0021] Alternatively, if the software bridge concludes in step 306 that the minimal capability for the legacy devices involved in the requested action is satisfied, the method 300 proceeds to step 308, and the software bridge bridges the devices transparently (*i.e.*, substantially without impact to the operation, behavior or configuration of the bridged devices) before the method 300 terminates in step 312.

[0022] FIG. 4 is a flow diagram illustrating a more detailed embodiment of a method 400 for bridging Session Initiation Protocol and Universal Plug and Play devices. Like the method 300, the method 400 may be implemented in a software bridge embodied, for example, in a residential gateway or at a standalone accessory to an SIP device. The method 400 is specifically directed to enabling an SIP device to access UPnP services. Thus, for example, the method 400 may be implemented by a user in order to stream music from a UPnP-based home media server to an SIP cordless phone.

[0023] The method 400 is initialized at step 402 and proceeds to step 404, where the software bridge discovers UPnP devices and their services. Following discovery, the method 400 proceeds to step 406, and the software bridge associates a User Service Request Address (USRA) with each discovered UPnP

device and the UPnP device's services. The USRA is a novel address format that reuses the format of the SIP Uniform Resource Identifier (URI), which enables standard RFC3261 SIP procedures to be used without making changes to the SIP or UPnP devices. However, where the SIP URI is used for addressing users (e.g., the users of services), the USRA is used to address services offered by a bridge. For example, abstract services (e.g., UPnP application handles, such as "StreamMusic") are mapped to executable UPnP actions or sequences of UPnP actions for use by SIP users, by reusing the SIP URI. In one embodiment, the USRA comprises a domain (e.g., the SIP domain "JohnsHome.net"), a device ID (e.g., "JohnPC") and a service ID (e.g., "StreamMusic").

[0024] In step 408, the software bridge registers, locates and, invokes at least one UPnP service, using the semantics of the SIP-URI. Registering UPnP services (e.g., by making connections between UPnP services and SIP URIs) enables UPnP services to be accessed from an SIP domain using regular SIP ID's (e.g., SIP URIs). Registration also enables locating UPnP services (e.g., by determining where/to which domain to route commands), which in turn enables access to services. Invoking UPnP services involves determining which UPnP service a command is requesting and generating a set of actions to be carried out by the UPnP device hosting the UPnP service to be accessed. One embodiment of a method for registering, locating and invoking a UPnP service is illustrated in FIG. 6. Thus, the SIP-URI is used to register and locate UPnP services, rather than to register and locate users.

[0025] For example, in one embodiment, the "To" field of an RFC3261 SIP header is used in a REGISTER method, in order to allow the software bridge to specify a UPnP service that needs to be registered in the SIP domain. In particular, usage of the "To" field is "overloaded" to address or register a service offered by a bridge, by including a USRA referring to the service. Normally, the "To" field in a RFC3261 SIP header is used to address or register users, as described above.

[0026] For instance, say a user, John, would like to use his SIP device to stream a song stored in the default shared folder of his UPnP media server (hosted by a PC in John's room, with the UPnP friendly name "JohnPC"). Further assume that the software bridge has a service titled "StreamMusic" that supports

this scenario. In this case, the software bridge would execute an SIP REGISTER with the "To" field of the RFC3261 SIP header set as "To: JohnPC.StreamMusic@JohnsHome.net", where "JohnsHome.net" is the SIP domain that provides the "StreamMusic" service. The software bridge will recognize the UPnP device "JohnsPC" as a device that is allowed to communicate with SIP-based devices.

[0027] Alternatively, suppose John wants to use his SIP device to stream a video clip stored in the default shared folder of his digital video recorder (which has the UPnP friendly name "DVR"). Further assume that the software bridge has a service titled "StreamVideo" that supports this scenario. In this case, the software bridge would execute an SIP REGISTER with the "To" field of the RFC3261 SIP header set as "To: DVR.StreamVideo@JohnsHome.net".

[0028] In another embodiment, the "To" field of an RFC3261 SIP header is used in an INVITE method, in order to allow an SIP device to locate a UPnP service. This also allows the software bridge to implement the UPnP service in terms of the UPnP actions. Referring back to the example above, when John actually invokes a service from his SIP device, an SIP INVITE message with the "To" field set as described above is generated (e.g., "To: JohnPC.StreamMusic@JohnsHome.net" or "To: DVR.StreamVideo@JohnsHome.net"). The "To" field is used by the software bridge to determine and invoke the service. The software bridge will recognize the UPnP domain (e.g., "JohnsHome.net") as a place to communicate with SIP devices.

[0029] In yet another embodiment, the "To" field of an RFC3261 SIP header is used in an INVITE method message, in order to allow an SIP device to call media items or to receive particular files or services. This also allows system elements of the software bridge to understand the methods to use, determined by the media resource attributes. For example, if a .cal extension calendar resource were called, the software bridge would create a rendered image of the calendar, to be streamed to the SIP device. As another example, if a .wav extension resource, the software bridge would stream the music to the SIP device. In this case, the "StreamVideo" portion of the exemplary USRA is used by the software

bridge to select streaming video from the UPnP device to be sent to the SIP device.

[0030] The method 400 then terminates in step 410.

[0031] The method 400 thus relies on the re-use (overloading) of existing RFC3261 SIP and UPnP communication messages and on common applications (e.g., an existing contact book application, an RFC2833-compliant dual-tone multi-frequency (DTMF) application on an SIP communication device or the like) to configure a USRA on a SIP device. No additional protocol features (e.g., SIMPLE), changes to the SIP specification or new application features are needed, making the overloading transparent to software and hardware. Thus, the invention is extensible to legacy SIP and UPnP devices. However, from an end-user perspective, UPnP applications are now accessible using SIP devices. Moreover, by relaying session services offered by an SIP device to a UPnP network, the invention enables a UPnP device to access the session services, thus facilitating bidirectional internetworking.

[0032] FIG. 6 is a flow diagram illustrating one embodiment of a method 600 for registering, locating and invoking a UPnP service (e.g., in accordance with step 408 of the method 400).

[0033] The method 600 is initialized at step 602 and proceeds to step 604, where the software bridge receives a command including a USRA from a SIP device. For example, the command may be a SIP REGISTER command with the "To" field of the RFC3261 SIP header set as "To: JohnPC.StreamMusic@JohnsHome.net".

[0034] In step 606, the software bridge identifies the service being requested by the command (e.g., StreamMusic in the above example) and the UPnP device from which the service is requested (e.g., JohnPC in the above example), in accordance with the USRA. The domain specified in the USRA tells the software bridge where to route the command.

[0035] In step 608, the software bridge maps the requested service to a set (i.e., one or more) of actions or services. For instance, in the above example, the StreamMusic service requested by the command might map to a series of actions including: (1) "browse" (e.g., browse the music available at the UPnP device); (2)

“select” (e.g., select an item from the available music); and (3) “play” (e.g., play the selected item).

[0036] In step 610, the software bridge issues one or more commands (in UPnP format) to the identified UPnP device to execute the set of actions or services. The software bridge then receives a response from the identified UPnP device in step 612. For instance, if the software bridge issued a command to the UPnP device to “browse” in step 610, the UPnP device might respond in step 612 with a list of available music.

[0037] In step 614, the software bridge provides the requested service to the requestor/SIP device. For instance, the software bridge might provide the list of songs available from the identified UPnP device, so that the requestor may browse the list and make a selection.

[0038] The method 600 terminates in step 616.

[0039] FIG. 5 is a flow diagram illustrating a more detailed embodiment of a method 500 for bridging Session Initiation Protocol and Universal Plug and Play devices. Like the method 300, the method 500 may be implemented at a software bridge, for example embodied in a residential gateway or at a standalone accessory to an SIP device. The method 500 is specifically directed to enabling a UPnP device to access session-based services offered by an SIP device. Thus, for example, the method 500 may be implemented by a user in order to stream video from a UPnP media server onto an SIP cordless phone, using a UPnP enabled television remote control. Alternatively, the method 500 may be implemented by a user in order to use a SIP device to retrieve content stored on a UPnP media server and stream the retrieved content onto the SIP device.

[0040] The method 500 is initialized at step 502 and proceeds to step 504, where the software bridge represents the SIP device to the UPnP network as a “virtual” UPnP audio/visual (AV) renderer device, hosted by the software bridge. That is, the software bridge appears as a UPnP device to the UPnP network. Thus, the “virtual” UPnP renderer is an alias for the SIP device, which represents the SIP device in the UPnP domain (whereas a “real” UPnP device has UPnP software and a UPnP identifier and communicates with the rest of the UPnP network using UPnP protocol). In one embodiment, the software bridge is configured by determining which SIP devices are available to provide services

and creating proxies (virtual UPnP devices) for each available SIP device. In accordance with step 502, the "virtual" UPnP renderer advertises its presence on the UPnP network, so that the "virtual" UPnP renderer device is discoverable by legacy UPnP control points. Once the "virtual" UPnP renderer device is discovered, the legacy UPnP control points may involve the "virtual" UPnP renderer device in a session with a legacy UPnP media server.

[0041] In step 506, the software bridge maps the UPnP AV renderer operations to SIP messages. For instance, upon discovery by a legacy UPnP control point, the software bridge may map UPnP signaling messages related to session establishment to SIP signaling messages. In further embodiments (e.g., involving UPnP control point initiated sessions), the UPnP AVT Play message is mapped to a SIP INVITE message, such that a SIP/UPnP session manager component of a bridge may take inputs from the "virtual" UPnP renderer in order to perform mapping operations. In another embodiment (e.g., involving a SIP device initiated session), a SIP INVITE message is mapped to a set of UPnP actions (e.g., UPnP CDS Browse, UPnP AVT SetTransport URI, UPnP AVT Play or the like), and the SIP/UPnP session manager component of the software bridge performs the mapping operations. In another embodiment still (e.g., involving a UPnP control point initiated session), a SIP session is initiated once the "virtual" UPnP renderer completes a download of the media session over hypertext transport protocol (HTTP).

[0042] In one embodiment, a UPnP Content Directory Service could be used by using a GetProtocolInfo() message in order to obtain a protocol/format list for an SIP device. In one embodiment, the "virtual" UPnP renderer has SIP to UPnP protocol mapping functionality. In a further embodiment still, the "virtual" UPnP renderer has media transcoding functionality.

[0043] The method 500 terminates in step 508.

[0044] The method 500 therefore allows devices that cannot normally interact directly (e.g., UPnP media servers/control points and SIP devices) to interact via a "virtual" UPnP renderer acting on behalf of the SIP device. In further embodiments, the "virtual" UPnP renderer may be placed on the media path. Such placement may support valuable applications, such as the ability to perform

real-time transcoding when there is a mismatch in codec abilities between a UPnP media server and a SIP device.

[0045] FIG. 7 is a high level block diagram of the present software bridge that is implemented using a general purpose computing device 700, such as a personal computer, a set top box, a residential gateway, a mobile telephone, a personal digital assistant or the like. In one embodiment, a general purpose computing device 700 comprises a processor 702, a memory 704, a bridge module 705 and various input/output (I/O) devices 706 such as a display, a keyboard, a mouse, a modem, a network connection and the like. In one embodiment, at least one I/O device is a storage device (e.g., a disk drive, an optical disk drive, a floppy disk drive). It should be understood that the bridge module 705 can be implemented as a physical device or subsystem that is coupled to a processor through a communication channel.

[0046] Alternatively, the bridge module 705 can be represented by one or more software applications (or even a combination of software and hardware, e.g., using Application Specific Integrated Circuits (ASIC)), where the software is loaded from a storage medium (e.g., I/O devices 706) and operated by the processor 702 in the memory 704 of the general purpose computing device 700. Additionally, the software may run in a distributed or partitioned fashion on two or more computing devices similar to the general purpose computing device 700. Thus, in one embodiment, the bridge module 705 for bridging SIP and UPnP devices described herein with reference to the preceding figures can be stored on a computer readable medium or carrier (e.g., RAM, magnetic or optical drive or diskette, and the like).

[0047] Thus, the present invention represents a significant advancement in the field of home and mobile networking applications. Embodiments of the invention rely on overloading the features of RFC 3261 SIP and on common applications. No additional protocol features, changes to the SIP specification or new application features are needed. Thus, the invention is extensible to legacy SIP and UPnP devices. Moreover, by relaying session services offered by an SIP device to a UPnP network, the invention enables a UPnP device to access the session services, thus facilitating bidirectional internetworking.

[0048] While the foregoing is directed to embodiments of the invention, other and further embodiments of the invention may be devised without departing from the basic scope thereof.

What is claimed is:

1. A method for bridging a session initiation protocol device and a universal plug and play device, the method comprising:
 - detecting that both of the session initiation protocol device and the universal plug and play device are legacy devices; and
 - bridging the session initiation protocol device and the universal plug and play device using a bridge.
2. The method of claim 1, wherein the bridging comprises:
 - discovering at least one universal plug and play device, including one or more services associated with the at least one universal plug and play device;
 - associating a user service request address with each discovered universal plug and play device and the discovered universal plug and play device's associated services.
3. The method of claim 2, wherein the user service request address reuses a format of a session initiation protocol field as described in Request for Comments 3261.
4. The method of claim 3, wherein the user service request address reuses a uniform resource identifier field.
5. The method of claim 3, wherein the bridging further comprises:
 - registering at least one universal plug and play service using semantics of the session initiation protocol uniform resource identifier.
6. The method of claim 5, wherein a "To" field of a Request for Comments 3261 session initiation protocol header is used in a REGISTER method to specify a universal plug and play service for registration in a session initiation protocol domain.
7. The method of claim 5, wherein the bridging further comprises:

locating the least one universal plug and play service using semantics of the session initiation protocol uniform resource identifier.

8. The method of claim 7, wherein a "To" field of a Request for Comments 3261 session initiation protocol header is used in an INVITE method to enable the session initiation protocol device to locate the at least one universal plug and play service.

9. The method of claim 7, wherein the bridging further comprises:
invoking the least one universal plug and play service using semantics of the session initiation protocol uniform resource identifier.

10. The method of claim 9, wherein a "To" field of a Request for Comments 3261 session initiation protocol header is used in an INVITE method to enable the session initiation protocol device to call at least one media item.

11. The method of claim 9, wherein the invoking comprises:
generating a set of one or more actions for the at least one universal plug and play device to perform.

12. The method of claim 1, wherein the bridging comprises:
representing the session initiation protocol device to a universal plug and play network as a universal plug and play audio/visual renderer device; and
mapping at least one operation of the universal plug and play device to at least one session initiation protocol message.

13. The method of claim 12, wherein a universal plug and play GetProtocolInfo() message is used to obtain the protocol/format list of the session initiation protocol device.

14. A computer readable medium containing an executable program for bridging a session initiation protocol device and a universal plug and play device, where the program performs the steps of:

detecting that both of the session initiation protocol device and the universal plug and play device are legacy devices; and

bridging the session initiation protocol device and the universal plug and play device using a bridge.

15. The computer readable medium of claim 14, wherein the bridging allows the session initiation protocol device to access at least one universal plug and play service.

16. The computer readable medium of claim 15, wherein the bridging comprises:

discovering at least one universal plug and play device, including one or more services associated with the at least one universal plug and play device;

associating a user service request address with each discovered universal plug and play device and the discovered universal plug and play device's associated services.

17. The computer readable medium of claim 16, wherein the user service request address reuses a format of a session initiation protocol field as described in Request for Comments 3261.

18. The computer readable medium of claim 17, wherein the bridging further comprises:

registering at least one universal plug and play service using semantics of the session initiation protocol uniform resource identifier;

locating the least one universal plug and play service using semantics of the session initiation protocol uniform resource identifier; and

invoking the least one universal plug and play service using semantics of the session initiation protocol uniform resource identifier.

19. The computer readable medium of claim 14, wherein the bridging allows the universal plug and play device to access at least one session-based service offered by the session initiation protocol device.

20. A system comprising:
- a session initiation protocol device; and
 - a universal plug and play device bridged to the session initiation protocol device, wherein both of the session initiation protocol device and the universal plug and play device are legacy devices.

1/4

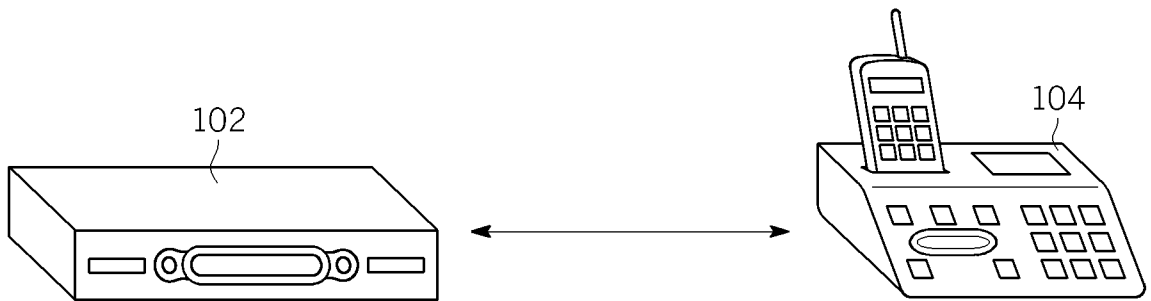


FIG. 1 100

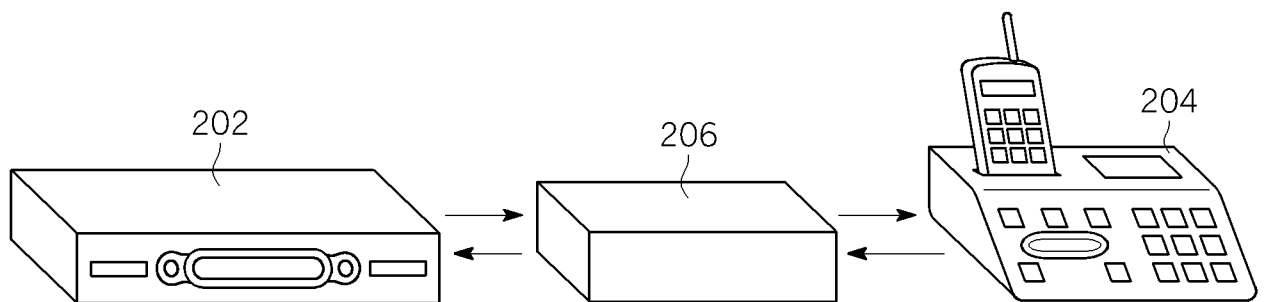
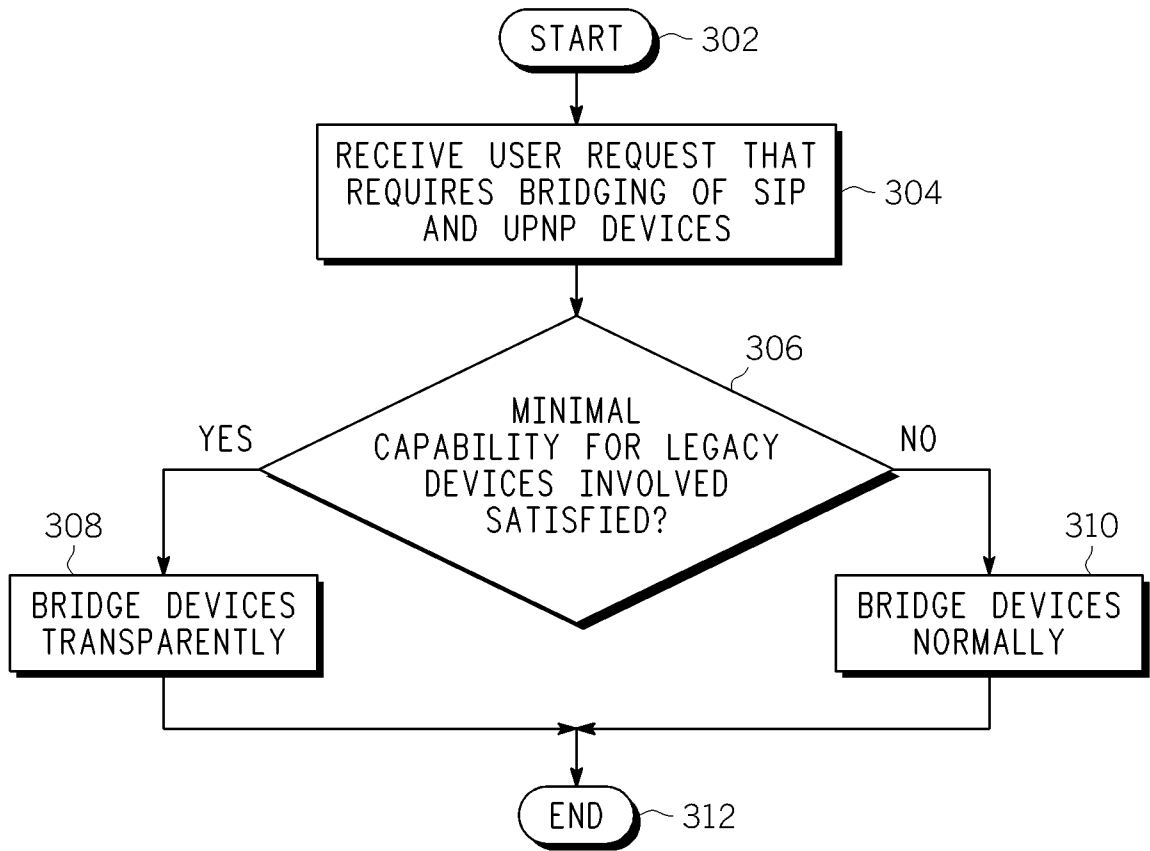
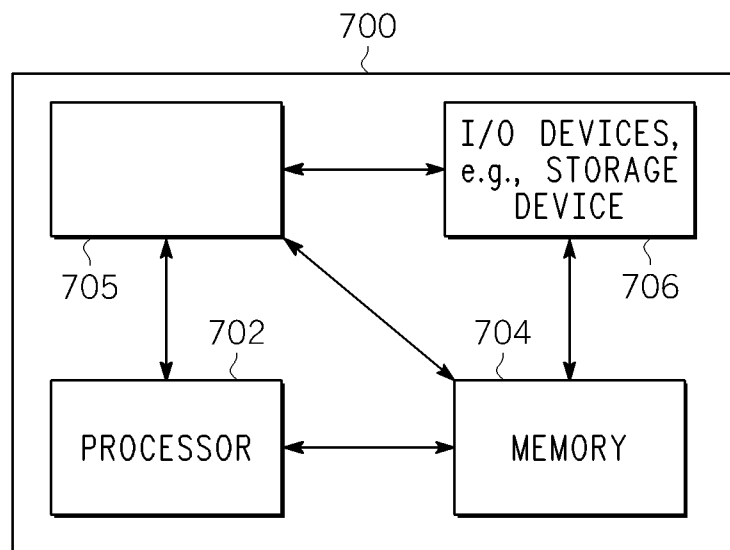


FIG. 2 200

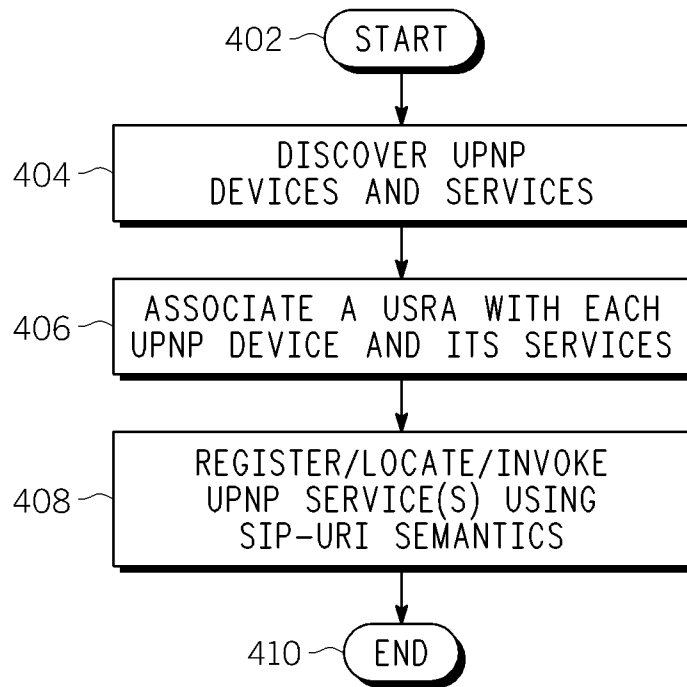
2/4

**FIG. 3**

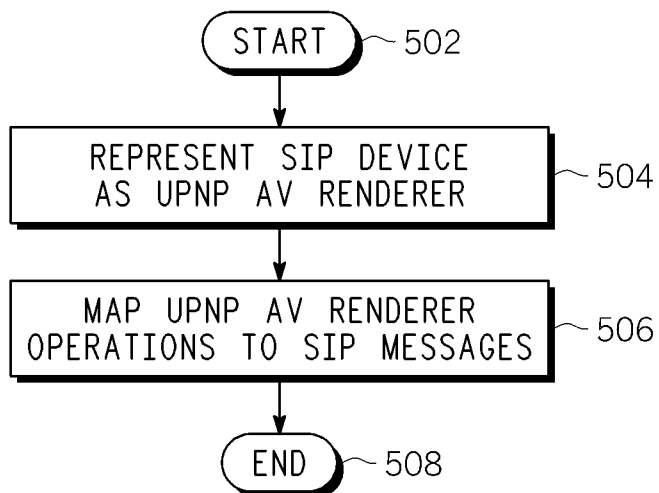
300

FIG. 7

3/4



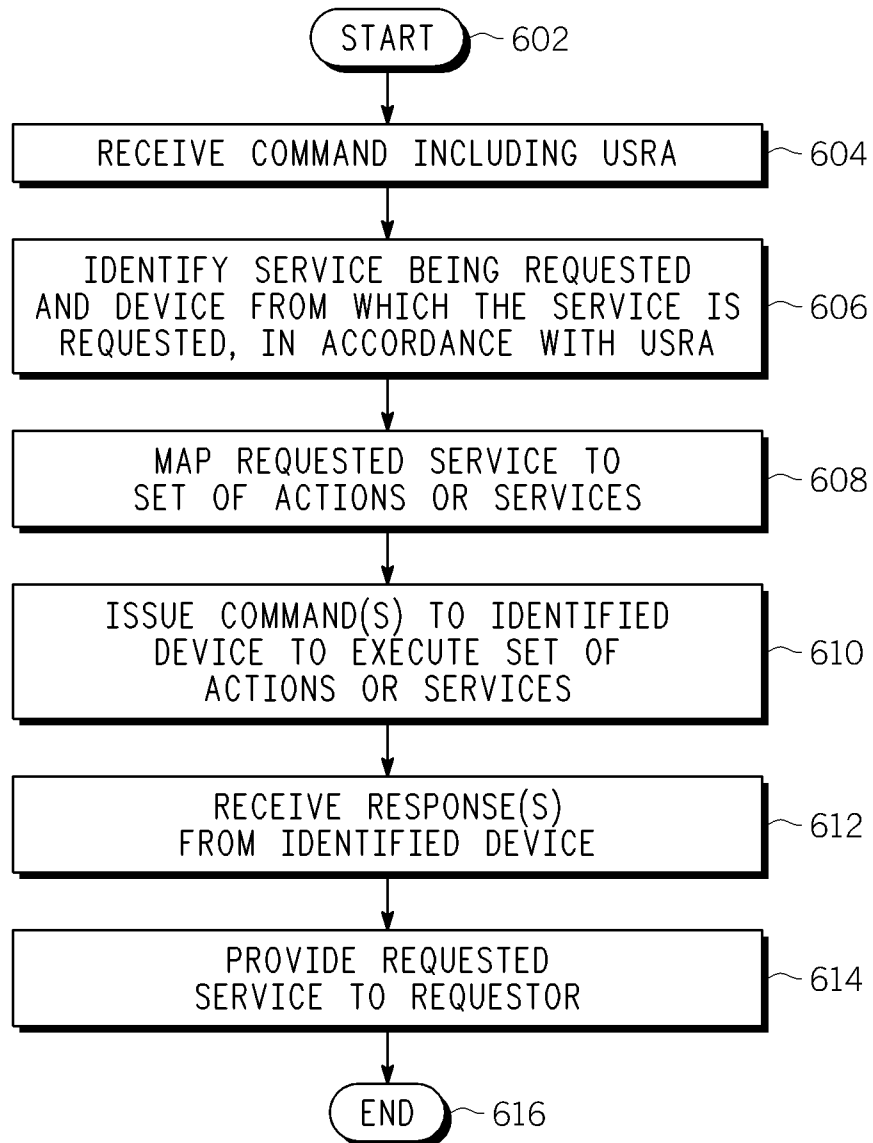
400

FIG. 4

500

FIG. 5

4/4

600**FIG. 6**