Research on UPnP Protocol Stack for Applications on a Home Network

Kalaiselvi Arunachalam[#1], Gopinath Ganapathy[#2]
[#] School of Computer Science, Engineering and Applications,
Bharathidasan University, Tiruchirappalli, India
[1] kalaiselvi.arunachalam@gmail.com
[2] gganapathy@gmail.com

Abstract—**The UPnP compliant devices can share data with each other and control others on a home network. The device to device communication is carried out by different protocols at different levels in different ways within the UPnP stack which is defined for devices only. But there is no feature for applications to share data with each other on a home network and they are very limited to share data between the devices with similar operating systems. There is no common standard for inter-application communication using UPnP on a home network but with only inter-device communication. This paper discuss about the application perspective view of all protocols within UPnP stack in order to achieve communication across applications residing within UPnP devices on a home network. This paper also discuss about the possibility of using alternative protocols within UPnP stack as a replacement or an extension for effective communication and proposes an upgraded UPnP stack for applications.**

Keyword - UPnP protocol, app-to-app communication, home network, protocol research, inter-application communication

I. Introduction

The UPnP technology that support zero-configuration which enable billions of devices in the world to connect with each other, share data and control others on a home network. The UPnP stack includes different protocols like IP, TCP, UDP, HTTP, SSDP, SOAP, GENA and XML which works at different levels and in different ways for communication between UPnP devices like Television, Audio Player, Scanner, Printer, Digital Camera etc. on a home network. The UPnP is defined for devices only and the UPnP stack contains the protocols for discovery, description, control, eventing and presentation of devices on a home network. Based on the UPnP device specifications, the devices are manufactured by the UPnP vendors. The device to device communication is already implemented on the home network using UPnP and the existing UPnP stack support only for devices as shown in Fig. 1 and Fig. 2. The smart devices on the home network like Smartphones, Tablets and PCs contains several applications. Even though these smart devices are UPnP enabled, the applications residing within these devices cannot communicate with each other on the home network. There is no standard or mechanism available for communication between these applications in UPnP. In order to achieve communication across these applications, this paper discuss about the application perspective view of the UPnP stack and about the possibility of using alternative protocols as a replacement or an extension to the UPnP stack for effective communication. Also it proposes an upgraded UPnP protocol stack as shown in Fig. 4 for applications that can be used by the developers to implement UPnP enabled applications that run on UPnP devices and they can communicate with each other on the home network.

II. UPnP Protocol Stack for Devices

The UPnP stack contains the specifications of the UPnP vendors, UPnP Working Committee and UPnP Device Architecture in the top three layers as in [1]. And the lower layers contains the standard protocols like IP, TCP, UDP, HTTP, SSDP, SOAP etc. which are used for communication across UPnP devices via any of the networking media technology either wired or wireless like Ethernet, IR, Bluetooth, Wi-Fi, Firewire, Telephone Lines, Coaxial Cable etc. The top three layers are used for defining the information to be used in the messages and the lower layers are responsible for formatting and delivery of those messages to the devices on the network as in [1]. The SSDP, SOAP and GENA protocols define the messages like discovery, advertisement, control and notification whereas the HTTP, HTTPMU and HTTPU protocols handles the delivery of these messages as shown in Fig. 1. The TCP/IP protocol suite provides the foundation for the UPnP stack. Based on the UPnP standard, an UPnP enabled device search for a DHCP (Dynamic Host Configuration Protocol) server when connected to the network by acting as a DHCP client to obtain an IP address for its advertisement. It acquires the address from the DHCP server if available or generates an Auto-IP address or a link-local address by itself based on RFC-3927 specification as in [2]. An UPnP device checks the acquired IP address with other devices on the home network to avoid address conflicts using ARP (Address Resolution Protocol). An UPnP device is added to the network after address verification and it advertises its services to the UPnP control points on the home network using SSDP (Simple Service Discovery protocol) over HTTPMU (HTTP over Multicast UDP) protocol which uses the multicast address 239.255.255.250 and port number 1900 as in [1]. In the same way, when an UPnP control point is added to the home network, it searches for devices on the home network using SSDP over HTTPMU protocol. An UPnP device that listens to the multicast port responds to the search request of control points through SSDP over HTTPU (HTTP over Unicast UDP) protocol.

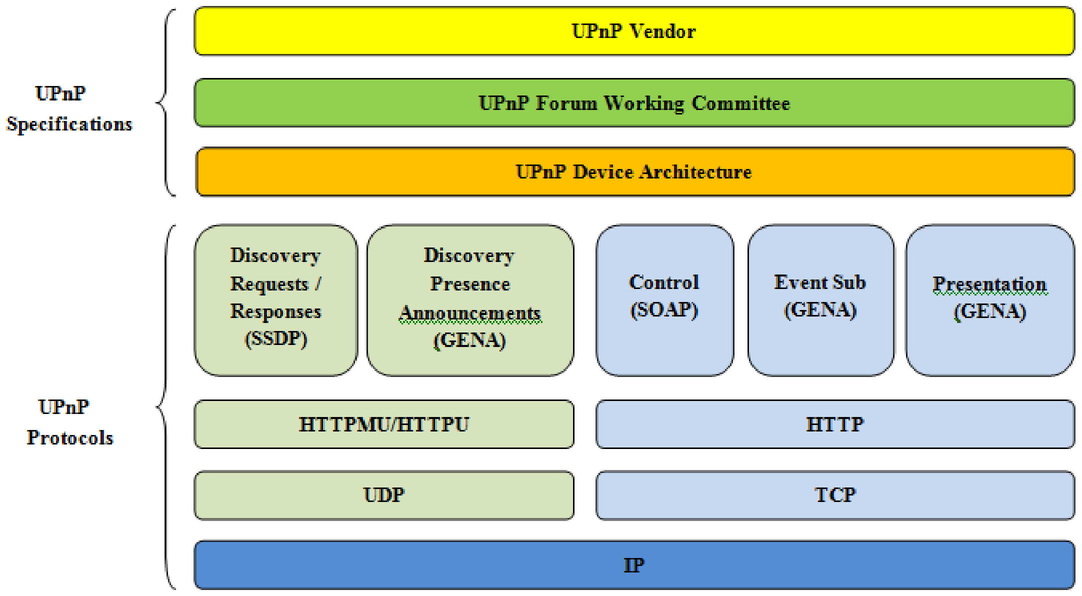

Fig. 1. UPnP protocol stack for devices

Once an UPnP control point identifies the device on the network, it retrieves the description document which is in XML (Extensible Markup Language) format through an URL provided by the device in the discovery message. The device description document contains the device information, list of services provided by the device, list of embedded devices or services in it and URLs for control, eventing and presentation as well as in [1]. To control the device or to perform any action on that device, an UPnP control point retrieves the service description document which is also in XML format from the device that contains the list of commands or actions that the service responds to along with the parameters or arguments and a list of variables to represent the state of service at run time as in [3].

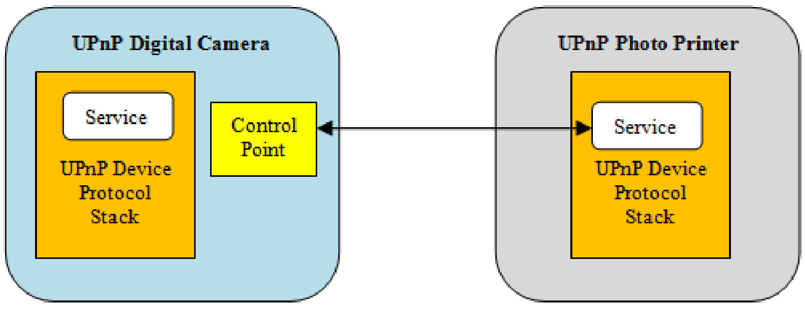

Fig. 2. Device to Device Communication using UPnP stack

By sending the control message which is in XML format to the control URL of the service, an UPnP control point controls the device through SOAP (Simple Object Access Protocol) protocol. As a response to the control message, the device's service returns the action specific values or error codes to the control point. The device's service publishes updates through event messages when there is a change in the state variables to the control points which are opted to subscribe them. These event messages are in XML format and based on GENA (General Event Notification Architecture) protocol. An UPnP device provides an URL for presentation page that can be viewed through a browser by the UPnP control points and to act on that device. Whenever an UPnP device leaves the network, it advertises this information to all control points on the network using SSDP over HTTPMU.

III. Application Perspective View of UPnP Protocol Stack

The UPnP Standard is defined for devices only. An UPnP device can dynamically join the network with zero-configuration, advertise its services, share data with other devices, control other devices and dynamically leave the network at any time. The UPnP stack handles the communication between UPnP devices effectively on the home network. As there is no provision for communication between applications running on UPnP devices on the home network and the billions of mobile or wireless devices in the market that evolve countless applications over them, it would be very effective and productive to implement communication between these applications on the home network. Since the inter-device communication is implemented using the UPnP stack, the possibility of implementing inter-application communication as shown in Fig. 3 on the existing UPnP stack can be analyzed. The device-specific protocols in each layer are compared against the application-specific requirements to decide whether to use or replace or extend them in the existing UPnP stack so that it can be used by UPnP applications and to implement inter-application communication as shown in Fig. 3 on the home network.

A. TCP/IP (Transmission Control Protocol/Internet Protocol)

The TCP/IP protocol provides the foundation for the UPnP stack in which IP handles the addressing and routing processes whereas TCP handles the message formatting and delivery processes. The TCP is used for peer-to-peer or one-to-one communication between two devices. The TCP which is a connection-oriented protocol that establishes a connection between two devices, sends the packets from one to another by confirming the delivery of those packets, resends the packets if undelivered after a timeout and finally closes the connection after successful delivery of all packets or issues an error message otherwise as in [4]. The IP encapsulates all data and it is connectionless whereas TCP plays intermediate role between the application program at higher level and the IP at lower level as in [5]. The IP handles the actual delivery of data whereas the TCP keeps track of the data transmission by minimizing the network congestion, retransmission of lost data, rearrangement of out-of-order data and notification to the source about the failure if the message undelivered. In UPnP, the IP address is obtained by the device from a DHCP server if available or by itself through the Auto-IP method between the ranges from 169.254.1.1 to 169.254.254.254 which are also referred as link-local addresses as in [2]. The devices are discovered and identified on the home network using these IP addresses. As an UPnP application is running on an UPnP device, it can utilize the IP address of the device to advertise its presence on which it is running and it can additionally provide an unique identifier to represent itself on the home network. Since TCP/IP is the base protocol for data transmission in a network at all levels, there is more possibility for an UPnP enabled application to use this protocol for data transmission as well. So, the existing TCP/IP protocol layer of the UPnP stack used by the devices is suitable for an UPnP application as well.

B. UDP (User Datagram Protocol)

The TCP cannot broadcast messages to multiple recipients on the network whereas the UDP can broadcast messages to multiple recipients on the network but it is a connection-less protocol in which there is no transmission channel for communication. The data transmission is faster in UDP than TCP but with unreliable delivery of messages as in [6], no ordering, no flow control, no error checking, no error correction, no retransmission of data and without any acknowledgements which are not the case with TCP. The time-sensitive real-time applications use UDP for faster communication. Also to communicate with multiple recipients simultaneously on the network by an UPnP device, the UDP/IP is preferred over TCP/IP because TCP/IP is meant for peer-to-peer communication only. An UPnP device uses UDP to send its presence announcements to all devices (Multicast UDP-HTTPMU), send response messages back to a sender based on the request (Unicast UDP-HTTPU), send event notifications to all devices etc. Similarly an UPnP application that run on an UPnP device can use UDP to perform the operations like presence announcements, event notifications etc. to other applications on the home network.

C. HTTP (Hypertext Transfer Protocol)

The HTTP is a request-response protocol in the client-server architecture in which the HTTP client opens a connection with the HTTP server, sends a request to the HTTP server, the HTTP server responds to the request by a message or a file or any other content and finally the HTTP server closes the connection. The HTTP is a stateless protocol as the HTTP server does not maintain the connection state between the requests but only the HTTP client can maintain it through cookies as in [7]. Being an ubiquitous transport protocol as it is widely used in the web, UPnP uses the HTTP to communicate between devices on the home network. As HTTP over TCP is an one-to-one communication and hence UPnP uses HTTP over UDP for one-to-many communication in which a host sends a message to multiple recipients on the network simultaneously and also one-to-one communication between devices without establishing a TCP connection. An UPnP application can use HTTP to carry out an one-to-one and one-to-many communication across UPnP applications on the home network.

1) HTTPMU (HTTP over Multicast UDP): The device presence announcements, device discovery requests and event notifications are delivered to all devices on the network by multicasting and for which the transport protocol UDP is used below HTTP. The HTTP over multicast UDP (HTTPMU) is used to send messages to all devices on the network simultaneously as in [3]. Being a connection-less protocol with broadcasting feature, UDP is used instead of TCP under HTTP. The UPnP devices are using HTTPMU for group communication on the network. An UPnP application that run on an UPnP device can utilize the same protocol HTTPMU to broadcast its presence to other UPnP applications within other devices on the network.

2) HTTPU (HTTP over Unicast UDP): An UPnP device response to the search or discovery requests of the control points and sends the event notifications to the subscribed control points individually through HTTP over unicast UDP as in [3]. Being a connection-less protocol, UDP is used to send the HTTP messages to other devices on the network without setting up a TCP connection. An UPnP device uses the HTTPU to respond to the sender and hence an UPnP application can utilize the same HTTPU protocol to respond to the requests sent by other UPnP applications on the home network.

D. SSDP (Simple Service Discovery Protocol)

The SSDP is used to advertise and discover UPnP devices or services on the home network and SSDP runs over HTTPMU and HTTPU as in [7]. The SSDP is used by the UPnP devices to advertise their presence to all control points on the network. Also, the SSDP is used by the UPnP control points to discover devices or services of their interest on the home network. The SSDP discovery requests are broadcasted to the multicast address 239.255.255.250:1900 through HTTPMU and the devices that listen to this address for the discovery requests will respond to it through HTTPU as in [3]. An UPnP device depends on SSDP to carry out the advertisement, discovery and response functions on the home network. Being a simple mechanism to discover devices on a home network, SSDP can be used to discover applications on the home network using a service type with an URI to identify the type of service and an Unique Service Name (USN) with an URI to identify an instance of a particular service using its UUID which is uniquely generated by the application. So, SSDP can be used by an UPnP application to get discovered on the home network.

E. GENA (General Event Notification Architecture)

The UPnP eventing refers to change in the state of an UPnP device's service like joining the network or change in state variables or leaving the network. Whenever there is a change in the device state, it is notified on the home network using GENA as in [3]. In UPnP, GENA is used to send and receive event notifications between devices and which are transferred using HTTP over TCP/IP and HTTP over UDP/IP. The GENA architecture provides an Publisher-Subscriber model as in [9] in which the services provided by the UPnP devices are the publishers and the control points are the subscribers. A control point can register or subscribe itself with a service of the device to receive notifications whenever there is a change in the service's state variable and the service sends an event notification message to the control points about the change based on their subscription. A control point can cancel the subscription of the service if not interested and also the service can cancel the subscriptions of the control points at any time. The three methods of GENA like SUBSCRIBE, UNSUBSCRIBE and NOTIFY as in [9] are used by UPnP devices to perform the event related operations on the home network. An UPnP application can also use GENA to perform event related operations on the home network where by UPnP applications that run on UPnP devices can send or receive event notification messages to the control points about the change in their state.

F. SOAP (Simple Object Access Protocol)

The SOAP is used to control devices on the home network by the control points. It is a protocol that uses RPC (Remote Procedure Call) mechanism and provides a web based messaging system using XML as in [10] and HTTP in which XML represents the contents of the message and HTTP transfers the message. After receiving the description document of an UPnP device and its services which contains the control URL, the UPnP control points can control the device's services by invoking actions on it through this control URL. The SOAP is used to send the control messages which contain the action to be invoked along with a set of parameters to the devices and return the response to the control points on the home network. The SOAP uses different transport protocols like SMTP, FTP, TCP and HTTP but HTTP is simpler and faster as than others. The REST is an equivalent protocol to SOAP which is widely used in the web nowadays and the possibility of using REST instead of SOAP in UPnP is discussed below.

1) REST (Representational State Transfer) - An alternative for SOAP: REST is a web based architecture which is used for transferring data between applications through HTTP. The six constraints of REST as in [11] include:

- client-server architecture in which a service offers some functionalities and it responds to the clients based on their request,
- stateless nature in which each client request is treated independently by the service,
- cacheable feature in which the service response messages are stored by the clients explicitly or implicitly and also server-side caching for scalability purpose,
- uniform interface in which all services and clients must share a single interface that covers HTTP methods and media types,
- layered system in which the REST architecture can have many layers where intermediary systems present in between the service and the clients that improve efficiency of communication and

- code on demand which is an optional constraint that involves the client-side update like plug-ins, scripts etc. from the transfer of executable code from the server and run independent of the server as well.

REST is simple and easier to understand than SOAP which supports many data formats like XML, JSON, CSV and it uses HTTP methods like GET, PUT, POST and DELETE.

2) SOAP vs REST: Though SOAP and REST are equivalent, there are differences in between them with a few advantages and disadvantages as well. SOAP is designed for distributed computing environment whereas REST is not but designed for point-to-point communication. SOAP is language, platform and transport independent but REST is dependent on HTTP. REST is simple, light-weight, readable and cacheable whereas SOAP is complex, heavy and non-cacheable but when security is concerned SOAP is better than REST as in [12]. The data transmission is faster in REST with its CRUD operations like CREATE, READ, UPDATE and DELETE when compared to SOAP as in [13] but SOAP supports WS-Security, WS-AtomicTransaction and WS-ReliableMessaging whereas REST doesn't support them. Even though REST is widely used in the web, SOAP is efficient, reliable and secured than REST which lacks of proper service description, service identification, service orchestration and framework for service life cycle as in [14]. As SOAP based on XML provides the definition for exchanging structured information between the peers in a distributed and decentralized environment in which SOAP messages are platform-independent and protocol-independent as in [15]. Though SOAP and REST are equivalent and effective in distinct environments based on their nature, SOAP is widely used in UPnP based on their structured XML description documents and control UPnP devices on the home network. Since we are analyzing the UPnP device protocol stack with respect to application and as UPnP devices are already compatible with SOAP, an UPnP application can also use SOAP to be accessed and controlled by the UPnP control points on the home network.

G. XML (Extensible Markup Language)

XML is a markup language that defines a structured format document containing data which can be both human and machine readable and is used by applications to exchange data across the web. The XML document contains the elements that have a beginning and ending tag with element name to describe the content of the element and the structure to describe the relationship between the elements. An XML parser which is based on XML specification as in [16] analyzes the structured data and passes them to an application. In UPnP, XML is used to describe an UPnP device and its services that enable the UPnP control points to send action requests to the services of an UPnP device, receive responses and to receive event notifications from the services of an UPnP device as well.

1) JSON (JavaScript Object Notation) - An alternative for XML: JSON is a language-independent and an open-standard data-interchange format which is simple and used to transfer data in a client-server communication system. JSON contains the key/value pair format and the symbols like "{ }" used for objects, "[ ]" used for arrays, "," used to separate pairs and ":" used to separate key and value as in [17]. JSON supports different data types like numbers, booleans, strings, arrays and objects. JSON is widely used in the web by several applications as because it is simple and faster.

2) XML vs JSON: Both XML and JSON are simple, open, interoperable, human-readable, machine-readable and with structured data which are widely used in the web by different applications. JSON is faster than XML because of its simple structure when compared to XML. JSON supports different data types but there is not much support for multimedia objects like audio or video whereas XML can support them. XML is best suited for document exchange whereas JSON is best suited for data exchange between applications and hence we can use the appropriate format based on our requirement. XML is extensible and document-oriented but JSON is not extensible and data-oriented. In UPnP, the device and service description documents are written in XML and SOAP uses XML as well. Since we are analyzing the existing UPnP protocol stack for devices against application perspective and XML is already working effectively for devices as in [1], we can use XML to describe an UPnP application and its services as well.

IV. Proposed UPnP Protocol Stack for Applications

Based on the application perspective research of UPnP protocol stack, there is no need to add or replace any protocol in the existing UPnP stack and we can use all those protocols for UPnP applications as well on the home network. The proposed UPnP stack for application contains the specification of the UPnP vendors and UPnP Working Committee in the top two layers as shown in Fig. 4. As UPnP applications are going to use the same protocols that are used by UPnP devices, we can define the UPnP Application Architecture in XML with an application description document that describes the application details like application name, unique application ID, IP address of its device, application author, application version etc. and an application service description document that describes the details of services provided by the application like print, close etc. As the applications run on UPnP devices on the home network, the application description document must include the details of its parent or UPnP device like its IP address and name in order to broadcast its presence on the home network so that the UPnP control points can easily access the applications on particular devices and control them on the home network like how they communicate with UPnP devices normally. The third and fourth layer defines the UPnP application and the device architecture. As an UPnP application run on a device, the UPnP application architecture is dependent on an UPnP device architecture as shown in Fig. 4.

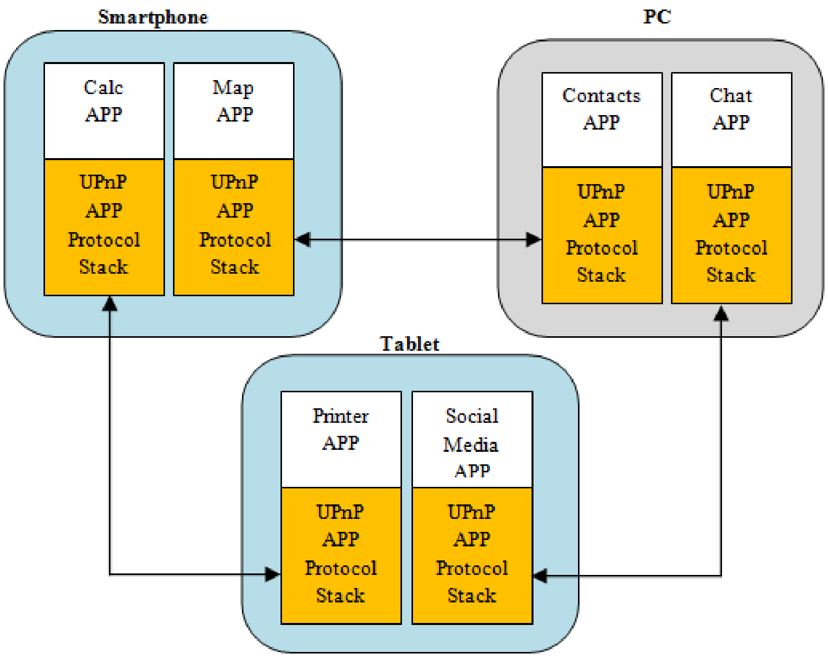

Fig. 3. Application-to-Application Communication using proposed UPnP stack

The UPnP control points can easily identify the instance of whether an UPnP device or an UPnP application is available on the network based on their description documents. The standard protocols IP, TCP, UDP, HTTP, SSDP etc. which are used for communication across UPnP applications on devices via any of the networking media technology either wired or wireless like Ethernet, IR, Bluetooth, Wi-Fi, Firewire, Telephone Lines, Coaxial Cable etc., The top four layers are used for defining the details of devices and applications and the lower four layers are used for communication between the UPnP applications on the network. The SSDP, GENA and SOAP protocols format the messages of specific type whereas the HTTP, HTTPMU and HTTPU protocols handles the delivery of those messages. Once an UPnP application started on an UPnP device, it collects the IP address and name of its parent or UPnP device and it advertises its services to the UPnP control points on the home network using SSDP (Simple Service Discovery protocol) over HTTPMU (HTTP over Multicast UDP) protocol. In the same way, when an UPnP control point is added to the home network, it searches for UPnP applications and devices on the home network using SSDP over HTTPMU protocol. An UPnP application that listens to the multicast port responds to the search request of control points through SSDP over HTTPU (HTTP over Unicast UDP) protocol.

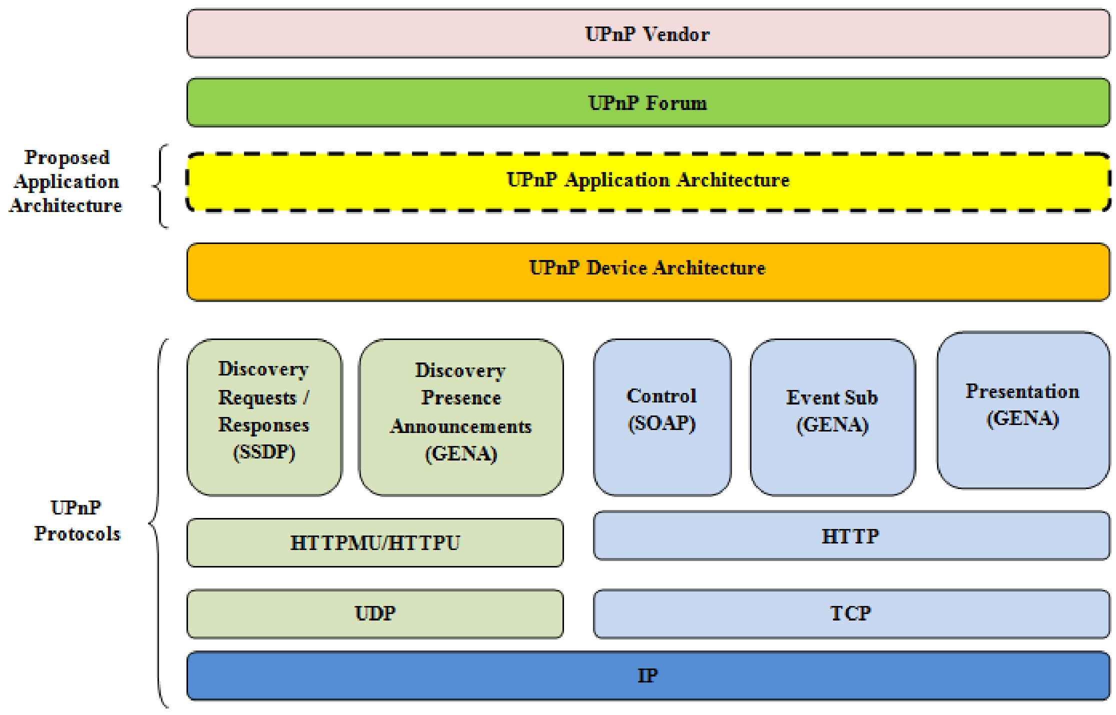
UPnP Vendor
UPnP Forum
Proposed Application Architecture
UPnP Application Architecture
UPnP Device Architecture
UPnP Protocols
Discovery Requests / Responses (SSDP)
Discovery Presence Announcements (GENA)
Control (SOAP)
Event Sub (GENA)
Presentation (GENA)
HTTPMU/HTTPU
HTTP
UDP
TCP
IP

Fig. 4. Proposed UPnP protocol stack for applications

Once an UPnP control point identifies the UPnP application on the network, it retrieves the application description document which is in XML (Extensible Markup Language) format through an URL provided by the UPnP application in the discovery message. The application description document contains the information like application name, application author, application version etc. and the list of services provided by the application along with the URLs for control, eventing and presentation as well. To control the application or to perform any action on that application, an UPnP control point retrieves the application service description document which is in XML format that contains the list of commands or actions that the application's service responds to along with the parameters or arguments and a list of state variables to represent the state of application's service at run time. By sending the control message which is also in XML format to the control URL of the application's service, an UPnP control point controls the application through SOAP (Simple Object Access Protocol) protocol. As a response to the control message, the application's service returns the action specific values or error codes to the control point. The UPnP application's service publishes updates through event messages when there is a change in the state variables to the control points which are opted to subscribe them. These event messages are in XML format and based on GENA (General Event Notification Architecture) protocol. An UPnP application provides an URL for presentation page that can be viewed through a browser by the UPnP control points and to act on that UPnP application. When an UPnP application is stopped or shutdown, then it advertises this information to all control points on the network using SSDP over HTTPMU. The UPnP applications can communicate with each other on the home network using this upgraded UPnP protocol stack by advertising their presence, responding to control points with application description documents along with their control URL, broadcasting event notifications etc.

V. UPNP DEVICE PROTOCOL STACK VS. APPLICATION PERSPECTIVE UPNP STACK

Based on the research of protocols used in the current UPnP stack for devices towards the UPnP application perspective stack are summarized and listed in the following Table I. This comparison concludes that the existing UPnP protocol stack for device is almost suitable for an UPnP application too. By extending this UPnP protocol stack with an application architecture specification or an application description which enables the communication between the UPnP applications across devices on the home network.

TABLE I. UPnP Device vs. UPnP Application Protocols

Protocol	Purpose	Networking Layer	UPnP Phase	UPnP Application Perspective Use
IP	Addresses hosts and routes datagrams	Network	Addressing	IP address of the UPnP device and an unique ID can be used
TCP	Transmits data between hosts	Transport	Control, Event Notification, Presentation	For communication between an application and a control point.
UDP	Transmits data between hosts	Transport	Discovery, Event Notification	For communication between an application and a control point.
HTTP	Transfers hypertext on the web	Application	Discovery, Control, Event Notification, Presentation	For communication between an application and a control point.
SSDP	Discovers and advertises a device or a service	Application	Discovery	For advertisement, discovery request & response (HTTPM/U)
SOAP	Controls actions on services of a device	Application	Control	To control actions on services of an application
GENA	Notifies about the event to the control points	Application	Event Notification	For notification of events to the control points by an application
XML	Describes the details of a device and its services.	Presentation	Description	For description of an application and its services

VI. CONCLUSION

The device to device communication is achieved in the existing UPnP stack but the lack of application to application communication in UPnP which is of demand now. As the inter-device communication is being carried out by different protocols in the UPnP stack, the possibility of achieving inter-application communication in UPnP is discussed in this paper. The application perspective view of all protocols in the existing UPnP stack and the possibility of using alternative protocols in the UPnP stack that can be used by UPnP applications are discussed. Based on the comparison of all protocols in the UPnP stack and their equivalent protocols, a newly upgraded UPnP protocol stack for an application is proposed which can be used by UPnP applications to achieve inter-application communication on the home network.

REFERENCES

[1] Andrew Donoho, Bryan Roe, Maarten Bodlaender, John Gildred, Alan Messer, YoonSoo Kim, Bruce Fairman, Jonathan Tourzan, UPnP Device Architecture 2.0, United States: UPnP Forum, 2015.
[2] Dynamic configuration of IPv4 link-local addresses (RFC3927), The Internet Engineering Task Force (IETF), 2005.
[3] Michael Jeronimo, *UPnP Design by Example: A software developer's guide to Universal Plug and Play, United States: Intel Press, 2003.*
[4] Transmission Control Protocol (RFC793), The Internet Engineering Task Force (IETF), 1981.
[5] Internet Protocol (RFC791), The Internet Engineering Task Force (IETF), 1981.
[6] User Datagram Protocol (RFC768), The Internet Engineering Task Force (IETF), 1980
[7] Hypertext Transfer Protocol -- HTTP/1.1 (RFC2616), The Internet Engineering Task Force (IETF), 1999.
[8] Simple Service Discovery Protocol/1.0 (SSDP/v1), The Internet Engineering Task Force (IETF), 1999
[9] General Event Notification Architecture Base: Client to Arbiter, The Internet Engineering Task Force (IETF), 1999.
[10] Simple Object Access Protocol (SOAP) 1.1, The World Wide Web Consortium (W3C), 2000.
[11] Fielding, Roy Thomas, Architectural Styles and the Design of Network-based Software Architectures, Dissertation, University of California, 2000.
[12] Muhammad Waqas Khan, Eram Abbasi, "Differentiating Parameters for Selecting Simple Object Access Protocol (SOAP) vs. Representational State Transfer (REST) Based Architecture", Journal of Advances in Computer Networks, Vol. 3, No. 1, 2015.
[13] Gavin Mulligan, Denis Gracanin, "A COMPARISON OF SOAP AND REST IMPLEMENTATIONS OF A SERVICE BASED INTERACTION INDEPENDENCE MIDDLEWARE FRAMEWORK", Proc. of the 2009 Winter Simulation Conference (WSC), 1423 - 1432, 2009.
[14] Markus Lanthaler, Christian Gütl, "Towards a RESTful Service Ecosystem Perspectives and Challenges", 4th IEEE International Conference on Digital Ecosystems and Technologies (IEEE DEST 2010), 209 - 214, 2010.
[15] R. Rutkauskas, A. Lipnickas, Č. Ramonas, V. Kubilius (2008), "New Challenges for Interoperability of Control Systems", ELECTRONICS AND ELECTRICAL ENGINEERING, AUTOMATION, ROBOTICS, No. 3(83), 2008.
[16] Extensible Markup Language (XML) 1.0 (Second Edition), The World Wide Web Consortium (W3C), 2000.
[17] The JavaScript Object Notation (JSON) Data Interchange Format (RFC7159), The Internet Engineering Task Force (IETF), 2014.

AUTHOR PROFILE

Kalaiselvi Arunachalam received the B.Sc. degree in Physics from the University of Madras, India and M.C.A degree in Computer Applications from the Anna University, India. She is currently a Ph.D. scholar in the School of Computer Science Engineering and Applications, Bharathidasan University, India. Her research interests include Home Networking, Communication Software and Systems.

Gopinath Ganapathy received the B.Sc. degree in Computer Science from the Bharathidasan University, India, M.C.A degree in Computer Applications from the St. Joseph's College Autonomous, India and Ph.D from the Madurai Kamaraj University, India. He is currently the Chair and Head, School of Computer Science Engineering and Applications, Bharathidasan University, India. His research interests include Semantic Web, NLP, Ontology, and Text Mining.