International Journal of Electrical and Computer Engineering (IJECE)
Vol. 7, No. 4, August 2017, pp. 2085~2093
ISSN: 2088-8708, DOI: 10.11591/ijece.v7i4.pp2085-2093

Extending UPnP for Application Interoperability in a Home Network

Kalaiselvi Arunachalam, Gopinath Ganapathy
School of Computer Science, Engineering and Applications, Bharathidasan University, India

Article Info

Article history:

Received, Dec 1, 2016
Revised, Apr 13, 2017
Accepted, Apr 27, 2017

Keyword:

Application architecture
Application interoperability
Home network
UPnP
UPnP extension

ABSTRACT

The Universal Plug and Play (UPnP) technology offers pervasive communication across heterogeneous devices in a home or small office network.The UPnP spefications are available for devices only to be interoperable together in a home or small office network. This paper proposes an extension of the UPnP technology for application interoperability in a home or small office network. This paper provides an UPnP Application Architecture as an extension to the existing UPnP Device Architecture. This extension enhances the feature of UPnP from device interoperability to application interoperability which enables the applications to discover, control and share data with each other in a home or small office network despite of their device type and operating system. In addition to the UPnP Application Architecture, the UPnP Application Template and UPnP Application Service Template are defined towards the development of UPnP-enabled applications that run on heterogeneous devices in a home or small office network.

Corresponding Author:

Kalaiselvi Arunachalam,
School of Computer Science, Engineering and Applications,
Bharathidasan University,
Khajamalai Campus,
Tiruchirappalli - 620 023, Tamil Nadu, India.
Email: kalaiselvi.arunachalam@gmail.com

1. INTRODUCTION

The rapid growth of smart devices like Smartphone, Tablet, Phablet, Smart TV, Notebook etc., enables billions of users around the world to use them at their home, office etc. The UPnP technology provides device interoperability through which devices can discover, control and share data with each other in a home or small office network [1]. The countless applications from various categories like entertainment, lifestyle, education, business, health etc. are used on these heterogeneous devices. The application interoperability is the next immediate requirement in a home or small office network where these devices are used together by the user. The application interoperability feature can enable the users to share data like text, image, audio, video, URL etc. between the applications residing on these devices. The application interoperability is more beneficial to the users in terms of interaction, production, communication and time. The device interoperability is enabled by UPnP in a home network already and it is limited to devices only where by devices can communicate with each other. But the discovery, control and data sharing between the applications residing on these devices are very limited. By extending the feature of UPnP for applications, these diverse applications residing on these heterogeneous devices can discover, control and share data with each other irrespective of their device type and operating system in a home or small office network. This paper proposes an UPnP Application Architecture along with the UPnP Application Template and UPnP Application Service Template to develop UPnP applications that are interoperable across heterogeneous devices in a home or small office network.

2. UPNP OVERVIEW

2.1. UPnP Technology

The UPnP technology provides a zero-configuration networking between the heterogeneous devices like personal computers, smart wireless devices and smart home appliances in a home or small office network [2]. The UPnP is a platform and language independent technology that offers discovery, control and data sharing between these devices [3]. It is built upon the TCP/IP protocol stack which includes IP, TCP, UDP, HTTP, SSDP, GENA, SOAP and XML [4]-[6]. It supports the network technologies like Ethernet, IR, Bluetooth, Wi-Fi, Firewire, Telephone Line and Coaxial Cable. The UPnP devices auto-configure and auto-discover themselves in a home or small office network for controlling each other and sharing data between them. The UPnP Forum defines device control protocols (DCPs) for several device categories like audio/video, networking, printing, scanning, telephony, home automation, remote access, multiscreen etc. The UPnP technology was promoted by the UPnP forum at the beginning and which is managed by the Open Connectivity Foundation (OCF) now [7] with more than thousand companies from various industries around the world as its members.

2.2. Communication flow between UPnP Devices

The UPnP control points and UPnP devices communicate with each other in a home or small office network [Figure 1]. When an UPnP device joins a network, it acts as a DHCP client and gets an IP address from a DHCP server if available or generates an IP address by itself using Auto-IP mechanism [8] and checks the IP address with other devices to avoid address conflicts using Address Resolution Protocol (ARP) [9]. Following the address verification and joining the network, an UPnP device advertises its presence to the UPnP control points in the home network using SSDP (Simple Service Discovery protocol) over HTTPMU (HTTP over Multicast UDP) protocol [10-12]. An UPnP control point searches for UPnP devices of interest in the home network using SSDP (Simple Service Discovery protocol) over HTTPMU (HTTP over Multicast UDP) protocol. An UPnP device responds to the search request of UPnP control points through SSDP (Simple Service Discovery protocol) over HTTPU (HTTP over Unicast UDP) protocol [10-12].

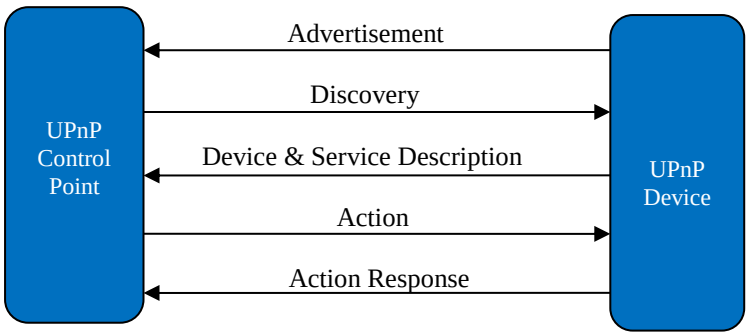

Figure 1. Communication between an UPnP Control Point and an UPnP Device

After the discovery of UPnP devices, an UPnP control point retrieves the device and service descriptions which are defined in XML format [13] from the UPnP devices through the URL provided in the device discovery message. The UPnP 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. 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. Then an UPnP control point controls an UPnP device by sending a control message to the device through SOAP protocol [14]. An UPnP device responds to the control message by returning the action-specific messages or error codes to the UPnP control point. The services of an UPnP device publishes any updates to the UPnP control points through event messages when there is a change in their state variables. These event messages are also in XML format and based on GENA (General Event Notification Architecture) protocol [15]. An UPnP device provides an URL for presentation page to the UPnP control points for user interaction. When an UPnP device leaves the network, it advertises this information to all UPnP control points in the home network using SSDP (Simple Service Discovery protocol) over HTTPMU (HTTP over Multicast UDP) protocol.

3. PROPOSED UPNP APPLICATION ARCHITECTURE

The UPnP Device Architecture [16] defines the networking protocols for communication between the UPnP control points and UPnP devices whereas the proposed UPnP Application Architecture defines the networking protocols for communication between the UPnP application control points (hereafter referred as UPnP ACP) and UPnP applications reside on the devices in a home or small office network (Figure 2). Any applications that are developed based on this proposed UPnP Application Architecture can communicate with each other irrespective of their device type and operating system in a home or small office network.

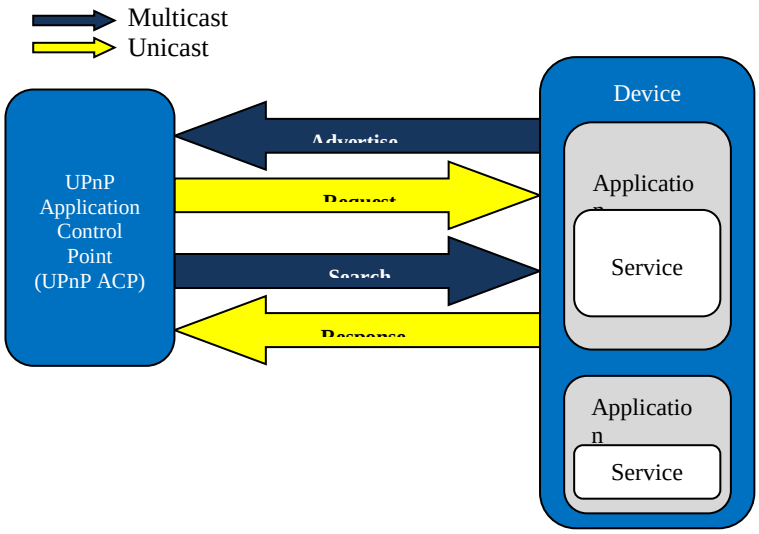

Figure 2. Communication between an UPnP ACP and UPnP Application

An UPnP application that reside on a device once if started or opened acquires the IP address of that device. Additionally it generates an UUID (Universally Unique Identifier) by itself [17] to uniquely identify its presence in a home or small office network. Then an UPnP application advertises its presence to all UPnP ACPs in a home or small office network [Figure 2]. An UPnP ACP searches for UPnP applications of interest in a home or small office network using SSDP (Simple Service Discovery Protocol) over HTTPMU (HTTP over Multicast UDP) protocol.

After the discovery of an UPnP application, an UPnP ACP retrieves the application description document which is in XML format from the URL provided by the application in the discovery message. The UPnP application description document contains the application information, list of services provided by the application and URLs for control, eventing and presentation. An UPnP ACP retrieves the application service description document which is also in XML format from the UPnP application 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. Then an UPnP ACP controls an UPnP application by sending a control message to the application through SOAP protocol. An UPnP application responds to the control message by returning the action-specific messages or error codes to the UPnP ACP [Figure 2]. The services of an UPnP application publishes any updates to the UPnP ACPs through event messages when there is a change in their state variables. These event messages are also in XML format and based on GENA (General Event Notification Architecture) protocol. An UPnP application provides an URL for presentation page to the UPnP ACPs for user interaction. When an UPnP application is closed or leaves the network, it advertises this information to all UPnP ACPs in the home network using SSDP (Simple Service Discovery Protocol) over HTTPMU (HTTP over Multicast UDP) protocol.

4. UPNP APPLICATION TEMPLATE

The proposed UPnP Application Template is an extension of UPnP Device Template. The UPnP Device Template provides the detailed information about a device whereas the UPnP Application Template provides the detailed information about an application that run within a device (Figure 3). The UPnP Application Template is defined in XML format which defines the details of application like IP address of its device, application ID (UUID), name, version number, description, appliction type (education, entertainment,

business, health etc.), application developer or owner name, application website URL etc (Figure 3). It also includes the list of services provided by the application which are accessed and controlled by the UPnP ACPs in a home or small office network.

```
<?xml version="1.0">
<rootapp xmlns="urn:schemas-upnp-org:application-1.0"
 appconfigId="configuration number">
<specVersion>
        <major>2</major>
        <minor>2></minor>
</specVersion>
<app>
        <appType>urn:schemas-upnp-org:application:appType:ver</appType>
        <appDeviceIP>IP address of the device</appDeviceIP>
        <appId>UUID generated by the application</appId>
        <appFriendlyName>short user-friendly name</appFriendlyName>
        <appDescription>detailed application description</appDescription>
        <appDeveloper>application developer name</appDeveloper>
        <appDeveloperURL>URL to application developer website</appDeveloperURL>
        <appVersion>application version</appVersion>
        <appURL>URL to application version</appURL>
                                .....
                                .....
<appServiceList>
 <appService>
  <appServiceType>urn:schemas-upnp-org:appservice:appServiceType:ver</appServiceType>
  <appServiceId>urn:schemas-upnp-org:appServiceId:appServiceID</appServiceId>
  <APPSCPDURL>URL for application service description</APPSCPDURL>
  <appControlURL>URL for application control</appControlURL>
  <appEventSubURL>URL for application eventing</appEventSubURL>
 <appService>
                                .....
                                .....

<app>
</rootapp>
```

Figure 3. UPnP Application Template in XML

The UPnP Device Template serves as a standard template for device manufacturers to write their UPnP device description by filling up the placeholders in UPnP Device Template with their custom information. The UPnP Application Template serves as a standard template for application developers to write their UPnP application description (Figure 4) by filling up the placeholders in UPnP Application Template with their custom information.

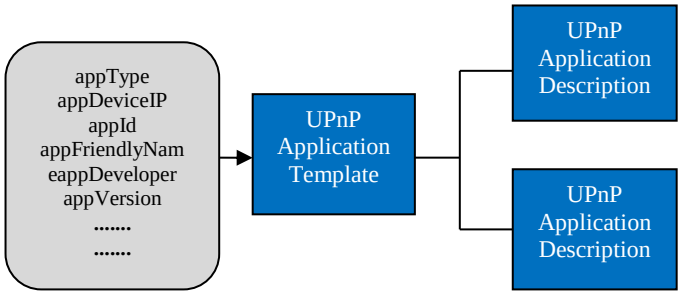

Figure 4. UPnP Application Template Hierarchy

The UPnP Application Templates can be written separately for various application categories like education, entertainment, lifestyle, social media, business etc. to enable application developers or companies to write UPnP application description documents accordingly and to develop UPnP applications so that these applications are interoperable across heterogeneous devices in a home or small office network.

5. UPNP APPLICATION SERVICE TEMPLATE

The proposed UPnP Application Service Template is an extension of UPnP Service Template. The UPnP Service Template defines the list of services provided by a device in a home or small office network.

The proposed UPnP Application Service Template defines the list of services provided by an application residing on a device in a home or small office network (Figure 5). The proposed UPnP Application Service Template is in XML format which defines the details of actions along with their arguments, data types and state variables for each service listed in the UPnP Application Template.

```
<?xml version="1.0">
<appscpd
      xmlns="urn:schemas-upnp-org:appservice-1.0"
      appconfigId="configuration number">
<specVersion>
      <major>2</major>
      <minor>1></minor>
</specVersion>
<appactionList>
   <appaction>
      <appactionname>appactionName</appactionname>
      <appargumentList>
         <appargument>
            <appargname>appargumentNameIn1</appargname>
            <argdirection>in</argdirection>
            <apprelatedStateVariable>stateVariableName</apprelatedStateVariable>
         <appargument>
                              .....
                              .....
   </appaction>
</appactionList>
<appserviceStateTable>
      <appstateVariable sendEvents="yes"|"no" multicast="yes"|"no">
            <appstatename>variableName</appstatename>
            <appstatedataType>basic data type</appstatedataType>
            <appstatedefaultValue>default value</appstatedefaultValue>
            <appallowedValueRange>
                  <appminimum>minimum value</appminimum>
                  <appmaximum>maximum value</appmaximum>
                  <appstep>increment value</appstep>
            </appallowedValueRange>
      </appstateVariable>
                              .....
                              .....
</appserviceStateTable>
</appscpd>
```

Figure 5. UPnP Application Service Template in XML

The UPnP Service Template serves as a standard template for device manufacturers to write their UPnP service description by filling up the placeholders in UPnP Service Template with their custom information. The UPnP Application Service Template serves as a standard template for application developers to write their UPnP application service description (Figure 6) by filling up the placeholders in UPnP Application Service Template with their custom information.

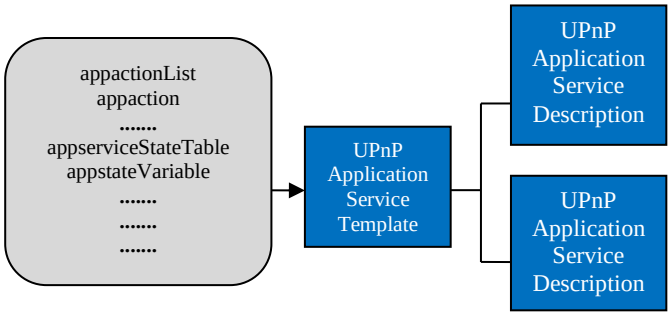

Figure 6. UPnP Application Service Template Hierarchy

The UPnP Application Service Templates can be written separately for various application categories like education, entertainment, lifestyle, social media, business etc. to enable application developers

or companies to write UPnP application service description documents accordingly and to develop standard UPnP applications that are interoperable across heterogeneous devices in a home or small office network.

6. IMPLEMENTATION AND DISCUSSION

Based on the proposed UPnP Application Architecture along with the new UPnP Application Template and UPnP Application Service Template, an UPnP-enabled application can be developed which can communicate with other UPnP applications in a home or small office network.

6.1. Implementation Methodology

There are several UPnP frameworks available from the UPnP vendors and open source providers [18] in different programming languages and platforms as well to create UPnP devices and control points easily. The existing UPnP frameworks are based on the UPnP Device Architecture and these frameworks can be extended by implementing the new functionalities based on the proposed UPnP Application Architecture to create UPnP applications. As UPnP is a platform and language independent technology, an open source UPnP framework like GUPnP [19] can be chosen for the implementation (Figure 7). The GUPnP is an object-oriented open source framework which is used for creating UPnP devices and control points.

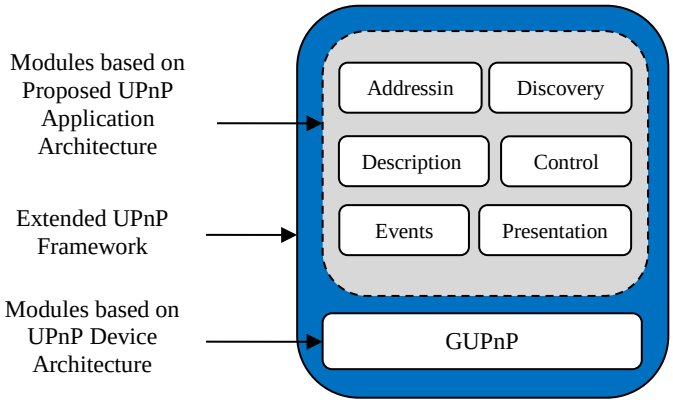

Figure 7. An open source extended UPnP framework

The application interoperability can be achieved in a home or small office network by the following steps.

1. The new functionalities based on the proposed UPnP Application Architecture, UPnP Application Template and UPnP Application Service Template are implemented on an open source UPnP framework that can be referred as an open source extended UPnP framework (Figure 7).
2. Develop two different custom applications on two different operating systems using the above open source extended UPnP framework (Figure 8).
3. Install both applications on their respective devices with different operating systems in a home or small office network.
4. Run both applications on their respective devices with different operating systems in a home or small office network.
5. Both applications can discover, control and share data with each other based on their application and application service descriptions.

When an application is started on a device or added to the home or small office network, then it sends a multicast message with method NOTIFY and ssdp:alive in the NTS header field of the advertisement (Figure 9) through SSDP over HTTPMU protocol to all UPnP ACPs in the home network.

When an UPnP ACP is started on a device or added to the home or small office network, then it searches for UPnP applications by sending a multicast request with method M-SEARCH and ssdp: discover in the MAN header field of the request (Figure 10) through SSDP over HTTPMU protocol to all UPnP applications in the home or small office network.

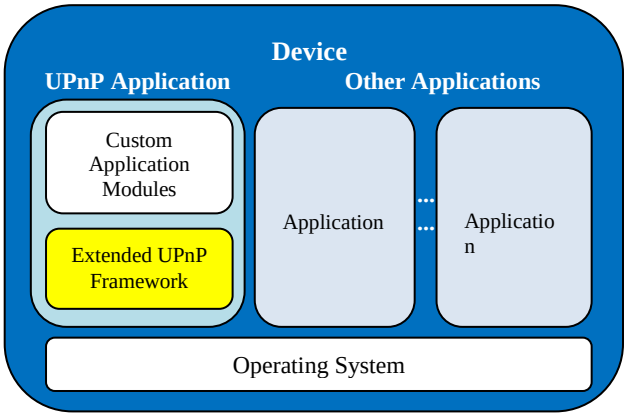

Figure 8. An UPnP Application on a device

NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: 300
LOCATION: *http://50.89.55.96/application_description.xml*
NT: *urn:schemas-upnp-org:application:appType:ver*
NTS: ssdp:alive

Figure 9. Advertisement of an UPnP Application (Joining a network)

M-SEARCH * HTTP/1.1
HOST: 239.255.255.250:1900
MAN: "ssdp:discover"
MX: 150
ST: *urn:schemas-upnp-org:application:appType:ver*

Figure 10. Search request of an UPnP ACP

An UPnP application responds to the search requests of the UPnP ACPs with the message (Figure 11) which includes the URL for application description document in order to access and perform actions on it by the UPnP ACPs.

HTTP/1.1 200 OK
CACHE-CONTROL: 300
LOCATION: http://50.89.55.96/application_description.xml
ST: *urn:schemas-upnp-org:application:appType:ver*
USN: *urn:schemas-upnp-org:appservice:appServiceType:ver*

Figure 11. Search response of an UPnP Application

After the discovery of an UPnP application, the UPnP application description and UPnP application service description documents are accessed by the UPnP ACPs in the home or small office network. An UPnP ACP invokes actions on an UPnP application with a control message (Figure 12) using SOAP protocol.

An UPnP application responds to the action requests with action-specific messages or error codes to the UPnP ACPs in the home or small office network. When an UPnP application is stopped or leaves the home or small office network, it advertises this information (Figure 13) to all UPnP ACPs using the SSDP over HTTPMU protocol.

```
POST http://50.89.55.96/control/appaction HTTP/1.0
HOST: 50.89.55.96:28
CONTENT-LENGTH: 1056
CONTENT-TYPE: text/xml; charset="utf-8"
SOAPACTION: "urn:schemas-upnp-org:appservice:appServiceType:ver#appactionName"
<?xml version="1.0"?>
<s:Envelope
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
    s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<s:Body>
        <u:actionNamexmlns:u="urn:schemas-upnp-org:appservice:appServiceType:ver">
                <argumentName>appargumentNameIn1</argumentName>
        </u:actionName>
</s:Body>
</s:Envelope>
```

Figure 12. Control message of an UPnP ACP

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
NT: urn:schemas-upnp-org:application:appType:ver
NTS: ssdp:byebye
```

Figure 13. Advertisement of an UPnP Application (Leaving a network)

The UPnP application and UPnP application control point module (UPnP ACP) within the extended UPnP framework (Figure 8) enables these applications to discover, control and share data with each other in a home or small office network. For example, a Weather Forecast application in a device can send an alert message to an Alarm application in another device to raise the alarm in a home or small office network.

7. CONCLUSION

The rapid growth of smart devices with numerous applications from various categories requires interoperability between these applications in a home or small network where these heterogeneous devices are used together. This can be achieved by extending the UPnP architecture for applications along with relevant UPnP application and UPnP application service templates defined for them. By extending an UPnP framework with the implementation based on the proposed UPnP Application Architecture, an UPnP-enabled application can be developed. These UPnP applications can discover, control and share data with each other despite of their device type and operating system in a home or small office network. A prototype implementation methodology and an example scenario are discussed.

REFERENCES

[1] Yu Shi-cai, et al., "A UPnP-based Decentralized Service Discovery Improved Algorithm", *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)*, Vol. 1, No. 1, pp. 21-26, March 2013.

[2] Michael Jeronimo, et al., "UPnP Design by Example: A Software Developer's Guide to Universal Plug and Play", *Intel Press*, United States, May 2003.

[3] Bendaoud Karim Talal, Merzougui Rachid, "Service Discovery – A Survey and Comparison", *International Journal of UbiComp (IJU)*, Vol. 4, No. 3, pp. 23-39, July 2013.

[4] Sabriansyah Rizqika Akbar, *et al.*, "Design of Pervasive Discovery, Service and Control for Smart Home Appliances: An Integration of Raspberry Pi, UPnP Protocols and Xbee", *International Journal of Electrical and Computer Engineering (IJECE)*, Vol. 7, No. 2, April 2017.

[5] Transmission Control Protocol (RFC793), "The Internet Engineering Task Force (IETF)", 1981.

[6] Internet Protocol (RFC791), "The Internet Engineering Task Force (IETF)", 1981.

[7] Open Connectivity Foundation (OCF): https://openconnectivity.org/upnp

[8] Cheshire, S., et al., "Dynamic Configuration of IPv4 Link-Local Addresses (RFC3927", *The Internet Engineering Task Force (IETF)*, United States, May 2005.

[9] An Ethernet Address Resolution Protocol, "The Internet Engineering Task Force (IETF)", 1982

[10] Simple Service Discovery Protocol/1.0 (SSDP/v1), "The Internet Engineering Task Force (IETF)", 1999

[11] Hypertext Transfer Protocol -- HTTP/1.1 (RFC2616), "The Internet Engineering Task Force (IETF)", 1999.
[12] User Datagram Protocol (RFC768), "The Internet Engineering Task Force (IETF)", 1980
[13] Extensible Markup Language (XML) 1.0 (Second Edition), "The World Wide Web Consortium (W3C)", 2000.
[14] Simple Object Access Protocol (SOAP) 1.1, "The World Wide Web Consortium (W3C)", 2000.
[15] General Event Notification Architecture Base: Client to Arbiter, "The Internet Engineering Task Force (IETF)", 1999.
[16] Andrew Donoho, et al., "UPnP Device Architecture 2.0", *Open Connectivity Foundation (OCF)*, United States, Feb 2015.
[17] Leach, P., et al., "A Universally Unique IDentifier (UUID) URN Namespace (RFC4122)", *The Internet Engineering Task Force (IETF)*, United States, July 2005.
[18] Open Connectivity Foundation (OCF) - Tools, Stacks & SDKs, https://openconnectivity.org/upnp/tools-stacks-sdks
[19] Jens, G., *et al.*, "The GUPnPFramework,"*The GNOME Foundation*, United States, 2013.

BIOGRAPHIES OF AUTHORS

Kalaiselvi Arunachalam received the B.Sc. degree in Physics from the University of Madras, India and M.C.A degree 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 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 includeSemantic Web, NLP, Ontology, and Text Mining.