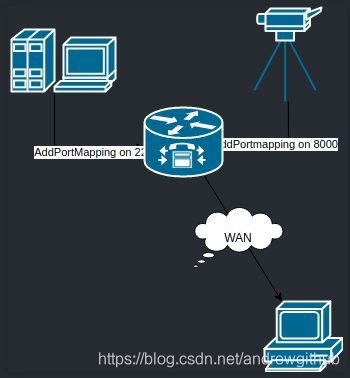
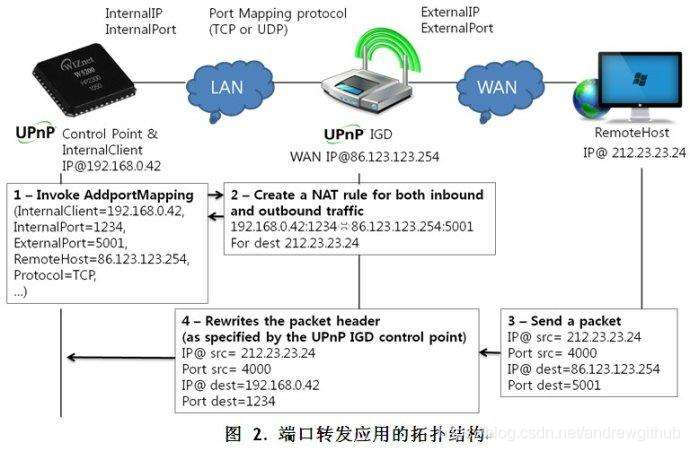
**UPNP report**

URL: https://blog.csdn.net/andrewgithub/article/details/107372276

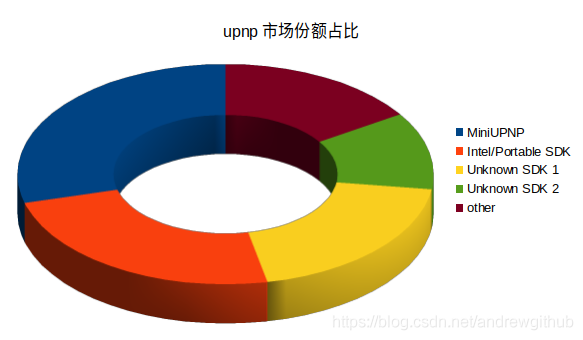
****UPNP拓扑图****  
  
****UPNP端口映射拓扑图****



UPNP允许不经过用户干预而自动发现和控制网络上其他设备提供的可用服务。

主要的节点有两个：

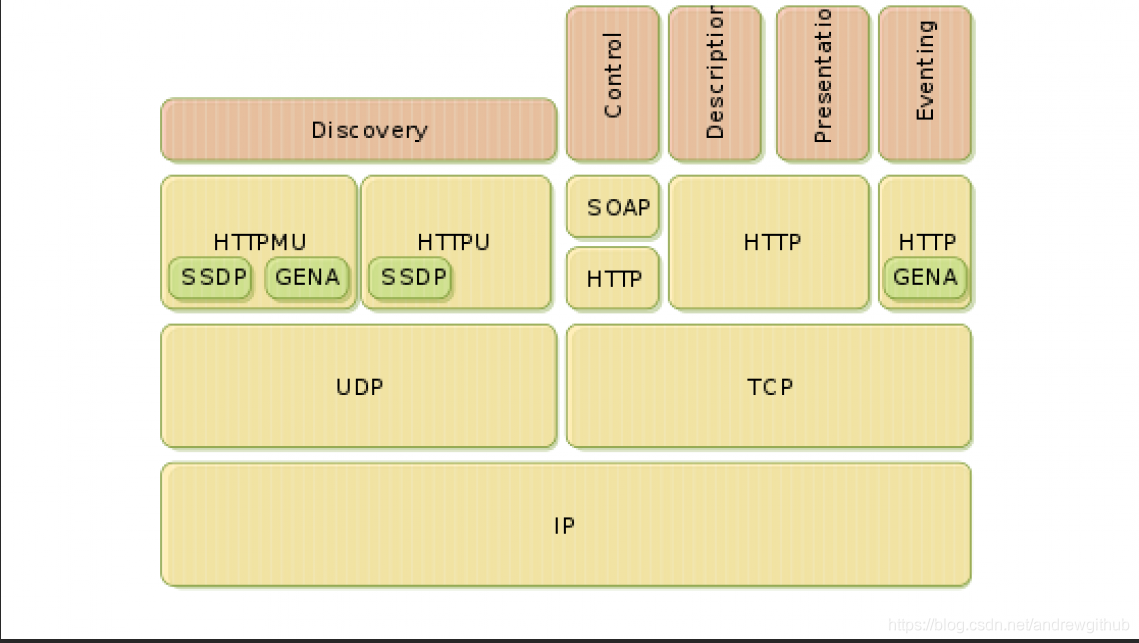
1. 设备(devices)：作为服务器向客户端发布自己服务；
2. 客户端：又称为控制点(Control Points)，能够搜索网络上特定的服务。

目前市场上主流的UPNP分布情况如下：  


## **概述**

UPNP的五个部分：

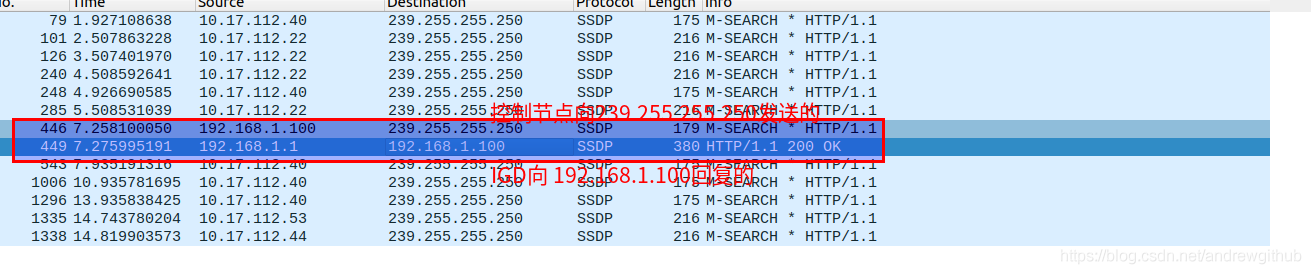
1. 发现(discoverity)：这个是第一阶段，控制点搜索设备和服务，同时设备(devices IGD) 广播他能提供的服务通告;
2. 描述(Description)：一旦控制节点发现了他感兴趣的设备或服务，他将向该设备也就是(IGD)请求该设备能够提供的完整的服务描述;
3. 控制(Control)：本阶段允许控制点通过改变设备(devices IDG)的状态来控制设备上提供的一个或多个服务;
4. 事件(Eventing)：本阶段允许控制点与其感兴趣的服务状态保持同步，控制点向事件服务器订阅一个特定的服务，但该服务状态改变时，接收事件通告。
5. 表示(Presentation)：表示阶段允许设备保持一份文档，该文档采用标准的HTML语言进行编写，他可以是该用户的一个用户界面



### **发现**

在发现阶段,控制点采用 SSDP(简单服务发现协议:Simple Service Discovery Protocol)发现设备和服务，而设备采用 SSDP 向控制点宣告他们的存在。

SSDP 采用 HTTP 的一种变体以 UDP 多播的方式来进行广播,并采用另一种 HTTP 变体通过 UDP 单播来进行应答。

  
一个设备可能包含其他设备，每个设备都有他自己的服务。设备采用其类型和一个唯一的标志符来进行标识。服务则用他们的类型来标识。为了搜索网络上的设备或服务,控制点使用 UDP 多播包向地址 239.255.255.250:1900 发送HTTP 的 M-SEARCH 命令。任何网络上服务控制点搜索条件的设备发回一个 UDP 单播进行应答,该应答中包含了指向其描述文档的 URL 地址。

LOCATION: http://192.168.1.1:1900/igd.xml

如果一个控制点收到一个或多个可接受的应答,他将转入描述阶段(description phase).一个控制点发出一个搜索请求时,该请求在 SSDP 头中包含了他愿意等待的时间长度。匹配的设备将在响应之前随机等待一段时间,该时间介于 0 和控制点指示的时间之间。如果控制点在他的搜索时间超时之前没有收到任何应答,他就认为当前网络上没有匹配的设备。

MX: 2

M-SEARCH \* HTTP/1.1

HOST: 239.255.255.250:1900

ST: urn:schemas-upnp-org:device:InternetGatewayDevice:1

MAN: "ssdp:discover"

MX: 2

HTTP/1.1 200 OK

CACHE-CONTROL: max-age=100

DATE: Fri, 10 Jul 2020 12:14:55 GMT

EXT:

LOCATION: http://192.168.1.1:1900/igd.xml

SERVER: vxWorks/5.5 UPnP/1.0 TL-WR842N/9.0

ST: urn:schemas-upnp-org:device:InternetGatewayDevice:1

USN: uuid:upnp-InternetGatewayDevice-A1B6D8B894C1::urn:schemas-upnp-org:device:InternetGatewayDevice:1

说明：

设备没必要等待控制点来搜索他们的服务。他们可以采用向 239.255.255.250:1900 多播地址发送 SSDP 的 NOTIFY 命令来宣告他们的设备可用性。当控制点获得该 NOTIFY 多播,他们就可以使用标准的 HTTP GET 命令来向 NOTIFY 消息中提供的 URL 地址发出请求以获得设备的描述文档。设备必须在他们的服务不可用时发出一个通告信息。

### **描述**

当控制点定位一个服务后他希望了解更多, 因而他将请求描述文档。描述是一个XML 文档用来描述一个设备,包括:  
● 制造商信息,版本,其他。  
● 可被设备采用的图标的 URL 地址。  
● 嵌入式设备列表。  
● 设备提供的服务列表。

…

控制点采用基于TCP的HTTP来请求描述文档。控制点执行标准的HTTP GET命令(与检索Web页面类似)。在服务器端,设备运行一个标准的HTTP服务–可以是完全的Web服务器如Apache也可以是迷你服务器。描述文档中的很多条目都是URL地址。这些条目也使用HTTP/TCP  
检索。

GET /igd.xml HTTP/1.1

Host: 192.168.1.1:1900

Connection: Close

User-Agent: Ubuntu/20.04, UPnP/1.1, MiniUPnPc/2.1

HTTP/1.1 200 OK

Content-Type: text/xml;charset=UTF-8

Content-Length: 2649

Connection: close

Cache-control: no-cache

<?xml version="1.0"?><root xmlns="urn:schemas-upnp-org:device-1-0"><specVersion><major>1</major><minor>0</minor></specVersion><device><deviceType>urn:schemas-upnp-org:device:InternetGatewayDevice:1</deviceType><presentationURL>http://192.168.1.1:80</presentationURL><friendlyName>Wireless N Router TL-WR842N</friendlyName><manufacturer>TP-LINK</manufacturer><manufacturerURL>http://www.tp-link.com.cn</manufacturerURL><modelDescription>TL-WR842N 9.0</modelDescription><modelName>TL-WR842N</modelName><modelNumber>9.0</modelNumber><UDN>uuid:upnp-InternetGatewayDevice-A1B6D8B894C1</UDN><UPC>123456789001</UPC><serviceList><service><serviceType>urn:schemas-upnp-org:service:Layer3Forwarding:1</serviceType><serviceId>urn:upnp-org:serviceId:L3Forwarding1</serviceId><controlURL>/l3f</controlURL><eventSubURL>/l3f</eventSubURL><SCPDURL>/l3f.xml</SCPDURL></service></serviceList><deviceList><device><deviceType>urn:schemas-upnp-org:device:WANDevice:1</deviceType><friendlyName>WAN Device</friendlyName><manufacturer>TP-LINK</manufacturer><manufacturerURL>http://www.tp-link.com.cn</manufacturerURL><modelDescription>WAN Device</modelDescription><modelName>WAN Device</modelName><modelNumber>1.0</modelNumber><modelURL></modelURL><serialNumber>12345678900001</serialNumber><UDN>uuid:upnp-WANDevice-A1B6D8B894C1</UDN><UPC>123456789001</UPC><serviceList><service><serviceType>urn:schemas-upnp-org:service:WANCommonInterfaceConfig:1</serviceType><serviceId>urn:upnp-org:serviceId:WANCommonInterfaceConfig</serviceId><controlURL>/ifc</controlURL><eventSubURL>/ifc</eventSubURL><SCPDURL>/ifc.xml</SCPDURL></service></serviceList><deviceList><device><deviceType>urn:schemas-upnp-org:device:WANConnectionDevice:1</deviceType><friendlyName>WAN Connection Device</friendlyName><manufacturer>TP-LINK</manufacturer><manufacturerURL>http://www.tp-link.com.cn</manufacturerURL><modelDescription>WAN Connection Device</modelDescription><modelName>WAN Connection Device</modelName><modelNumber>1</modelNumber><modelURL></modelURL><serialNumber>12345678900001</serialNumber><UDN>uuid:upnp-WANConnectionDevice-A1B6D8B894C1</UDN><UPC>123456789001</UPC><serviceList><service><serviceType>urn:schemas-upnp-org:service:WANIPConnection:1</serviceType><serviceId>urn:upnp-org:serviceId:WANIPConnection</serviceId><controlURL>/ipc</controlURL><eventSubURL>/ipc</eventSubURL><SCPDURL>/ipc.xml</SCPDURL></service>

</serviceList></device></deviceList></device></deviceList></device></root>

### **控制**

一旦一个控制点发现了一个设备并且检索到他的描述文档,他可能需要控制该设备包含的一个或多个服务。简单对象访问协议(SOAP:Simple Object Access Protocol)允许一个访问点查询或改变服务状态表中的元素。SOAP使用基于TCP传输HTTP的POST和M-POST命令。  
SOAP使用XML来说明采取的活动。

控制点如描述文档里指定的那样为服务创建XML文档并将其提交给控制URL。控制点能请求服务状态表中的当前值并改变他们。在服务器端,控制服务器等待控制请求。控制服务器是一个实现了SOAP协议的类似于HTTP服务器的服务器。一个设备能运行多于一个控制服务器,这取决于设备提供的服务的组合。

#### ****状态获取****

POST /ipc HTTP/1.1

Host: 192.168.1.1:1900

User-Agent: Ubuntu/20.04, UPnP/1.1, MiniUPnPc/2.1

Content-Length: 271

Content-Type: text/xml

SOAPAction: "urn:schemas-upnp-org:service:WANIPConnection:1#GetStatusInfo"

Connection: Close

Cache-Control: no-cache

Pragma: no-cache

<s:Envelopexmlns:s="http://schemas.xmlsoap.org/soap/envelope/"

s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">

<s:Body>

<u:GetStatusInfo xmlns:u="urn:schemas-upnp-org:service:WANIPConnection:1"></u:GetStatusInfo>

</s:Body></s:Envelope>

HTTP/1.1 200 OK

Content-Type: text/xml;charset=UTF-8

Content-Length: 428

Connection: close

Cache-control: no-cache

Server: vxWorks/5.5 UPnP/1.0 TL-WR842N/9.0

<?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:GetStatusInfoResponse xmlns:u="urn:schemas-upnp-org:service:WANIPConnection:1"> <NewConnectionStatus>2148414816</NewConnectionStatus>

<NewUptime>89894l</NewUptime>

<NewLastConnectionError>ERROR\_NONE</NewLastConnectionError>

</u:GetStatusInfoResponse>

</s:Body></s:Envelope>

#### ****外网**IP**获取****

POST /ipc HTTP/1.1

Host: 192.168.1.1:1900

User-Agent: Ubuntu/20.04, UPnP/1.1, MiniUPnPc/2.1

Content-Length: 285

Content-Type: text/xml

SOAPAction: "urn:schemas-upnp-org:service:WANIPConnection:1#GetExternalIPAddress"

Connection: Close

Cache-Control: no-cache

Pragma: no-cache

<?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:GetExternalIPAddress xmlns:u="urn:schemas-upnp-org:service:WANIPConnection:1"></u:GetExternalIPAddress></s:Body></s:Envelope>`

HTTP/1.1 200 OK

Content-Type: text/xml;charset=UTF-8

Content-Length: 358

Connection: close

Cache-control: no-cache

Server: vxWorks/5.5 UPnP/1.0 TL-WR842N/9.0

<?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:GetExternalIPAddressResponse xmlns:u="urn:schemas-upnp-org:service:WANIPConnection:1"><NewExternalIPAddress>10.17.112.25</NewExternalIPAddress></u:GetExternalIPAddressResponse></s:Body></s:Envelope>

#### ****获取端口映射****

POST /ipc HTTP/1.1

Host: 192.168.1.1:1900

User-Agent: Ubuntu/20.04, UPnP/1.1, MiniUPnPc/2.1

Content-Length: 399

Content-Type: text/xml

SOAPAction: "urn:schemas-upnp-org:service:WANIPConnection:1#GetSpecificPortMappingEntry"

Connection: Close

Cache-Control: no-cache

Pragma: no-cache

<?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:GetSpecificPortMappingEntry xmlns:u="urn:schemas-upnp-org:service:WANIPConnection:1"><NewRemoteHost></NewRemoteHost><NewExternalPort>8000</NewExternalPort><NewProtocol>TCP</NewProtocol></u:GetSpecificPortMappingEntry></s:Body></s:Envelope>

HTTP/1.1 200 OK

Content-Type: text/xml;charset=UTF-8

Content-Length: 537

Connection: close

Cache-control: no-cache

Server: vxWorks/5.5 UPnP/1.0 TL-WR842N/9.0

<?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:GetSpecificPortMappingEntryResponse xmlns:u="urn:schemas-upnp-org:service:WANIPConnection:1"><NewInternalPort>8000</NewInternalPort><NewInternalClient>192.168.1.100</NewInternalClient><NewEnabled>1</NewEnabled><NewPortMappingDescription>libminiupnpc</NewPortMappingDescription><NewLeaseDuration>0</NewLeaseDuration></u:GetSpecificPortMappingEntryResponse></s:Body></s:Envelope>

#### ****添加端口映射****

POST /ipc HTTP/1.1

Host: 192.168.1.1:1900

User-Agent: Ubuntu/20.04, UPnP/1.1, MiniUPnPc/2.1

Content-Length: 595

Content-Type: text/xml

SOAPAction: "urn:schemas-upnp-org:service:WANIPConnection:1#AddPortMapping"

Connection: Close

Cache-Control: no-cache

Pragma: no-cache

<?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:AddPortMapping xmlns:u="urn:schemas-upnp-org:service:WANIPConnection:1">

<NewRemoteHost></NewRemoteHost>

<NewExternalPort>8000</NewExternalPort>

<NewProtocol>TCP</NewProtocol>

<NewInternalPort>8000</NewInternalPort>

<NewInternalClient>192.168.1.100</NewInternalClient>

<NewEnabled>1</NewEnabled>

<NewPortMappingDescription>libminiupnpc</NewPortMappingDescription>

<!- 0表示是永久映射 也就是无限制>

<NewLeaseDuration>0</NewLeaseDuration>

</u:AddPortMapping>

</s:Body></s:Envelope>

HTTP/1.1 200 OK

Content-Type: text/xml;charset=UTF-8

Content-Length: 289

Connection: close

Cache-control: no-cache

Server: vxWorks/5.5 UPnP/1.0 TL-WR842N/9.0

<?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:AddPortMappingResponse xmlns:u="urn:schemas-upnp-org:service:WANIPConnection:1"></u:AddPortMappingResponse>

</s:Body></s:Envelope>

The NewRemoteHost parameter can be used to restrict the port mapping for just one external host, but in practice is never used

### **事件**

在控制节点发现感兴趣的设备文件描述之后，会从描述文档中找出自己感兴趣的-设备事件提醒服务URL。控制点订阅了服务之后，设备会在任何时候服务变动的时候给控制点，即使这次变动是由控制点产生的。

订阅和退订请求使用HTTP/TCP连接到事件URL,该URL包含在服务的描述文档中。事件服务器是一个类HTTP服务器,实现通用事件提醒架构协议(GENA:General Event Notification Architecture)的服务器。一个设备将可能根据设备提供的服务的组合(情况)必须运行多于一个的事件服务器。

SUBSCRIBE publisher path HTTP/1.1

HOST: publisher host:publisher port

CALLBACK: <delivery URL>

NT: upnp:event

TIMEOUT: Second-requested subscription duration

HTTP/1.1 200 OK

DATE: when response was generated

SERVER: OS/version UPnP/1.0 product/version

SID: uuid:subscription-UUID

TIMEOUT: Second-actual subscription duration

NOTIFY delivery path HTTP/1.1

HOST: delivery host:delivery port

CONTENT-TYPE: text/xml

CONTENT-LENGTH: Bytes in body

NT: upnp:event

NTS: upnp:propchange

SID: uuid:subscription-UUID

SEQ: event key

<e:propertyset xmlns:e="urn:schemas-upnp-org:event-1-0">

<e:property>

<variableName>new value</variableName>

</e:property>

Other variable names and values (if any) go here.

</e:propertyset>

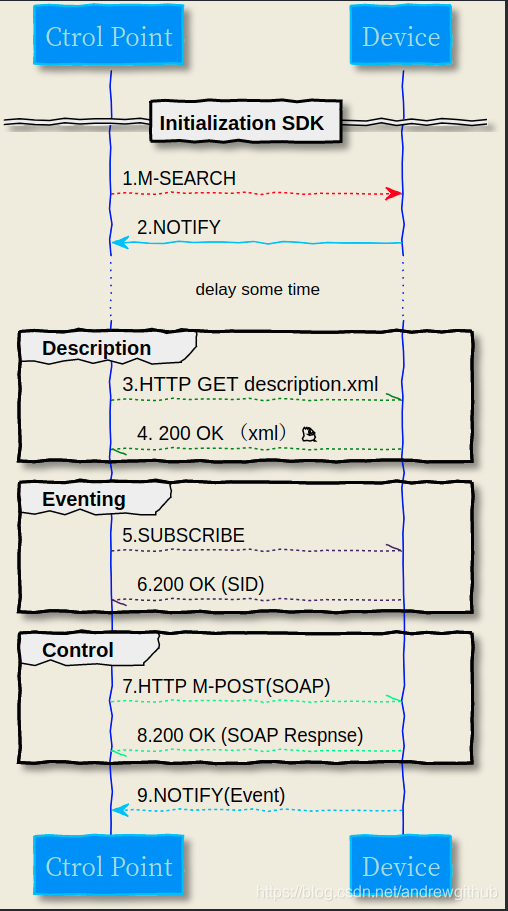
### **表示**

由于设备需要或支持用户交互,在表示阶段一个控制点能下载一个为设备描述用户界面的HTML文档。这是一个能提供一种控制或状态显示的标准HTML文档。如检索描述文档一样,检索表示文档的协议也是基于TCP的HTTP协议。控制点使用描述文档中包含的表示URL来请求表示文档。不是所有设备都拥有表示文档也不是所有控制点能够显示包含复杂HTML对象如框架,嵌入式Java applets等的表示文档。

描述文档获取地址：

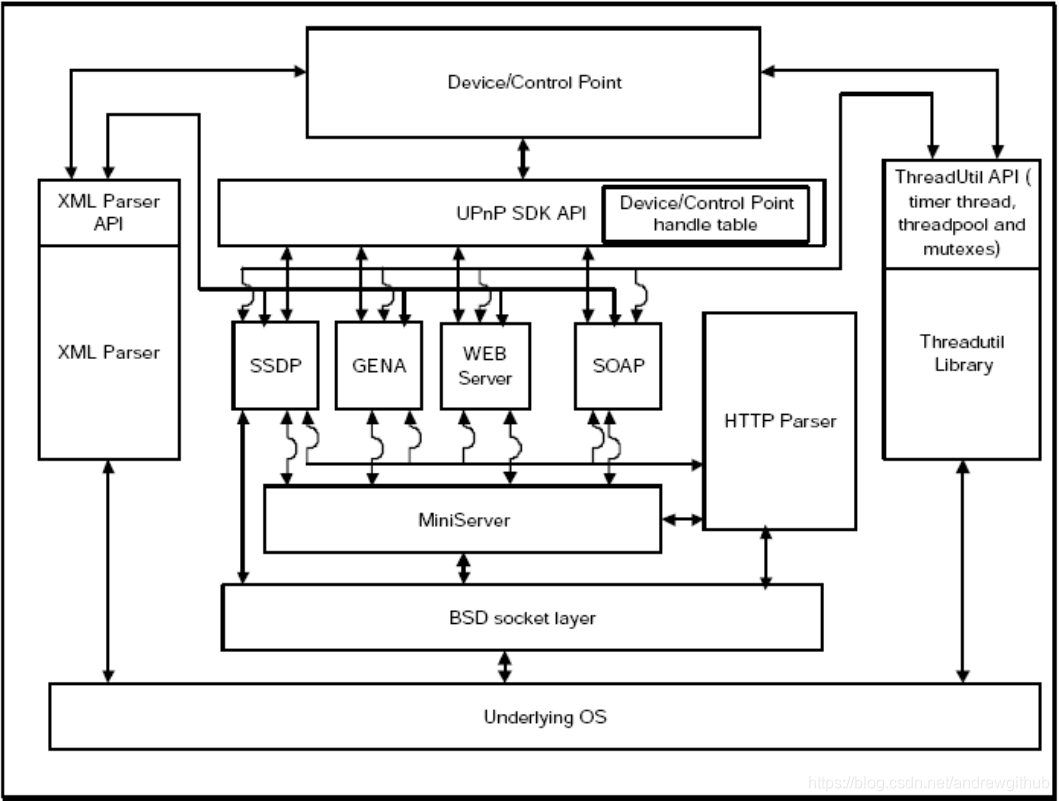
<presentationURL>http://192.168.1.1:80</presentationURL>

### **控制点和设备的交互过程**



1. 控制点使用UpnpSearchAsync() API发出一个搜索请求。UPnP控制点开发包(SDK)产生一个多播M-SEARCH SSDP消息并发送到网络上。
2. 如果设备匹配了控制点的搜索,SDK产生一个单播UDP的~响应,该响应包含指向设备描述文档的URL。SDK使用UpnpRegisterRootDevice()或UpnpRegisterRootDevice2()注册的设备描述文档包含的信息自动响应。
3. 如果控制点需要设备的更多信息,他将调用UpnpDownloadXmlDoc(),该调用使用设备响应给出的URL作为参数。UpnpDownloadXmlDoc()使用标准的HTTP GET请求下载XML描述文档并且返回一个代表设备描述的DOM文档。这一步可能重复执行以检索设备的服务描述文档。
4. 设备上包含的Web服务器对请求作出响应并返回XML描述文档。
5. 为了自动接收设备变化提醒,控制点订阅他感兴趣的服务。控制点通过UpnpSubscribe()或UpnpSubscribeAsync()订阅。控制点从他想订阅的服务的设备描述文档中提取出订阅URL,并且调用其中一个订阅函数。对每一个订阅调用,SDK通过包含一个URL的HTTP发送一个SUBSCRIBE消息给发送事件者。
6. 设备对订阅请求进行应答并返回一个唯一的订阅标识符。
7. 控制点通过改变设备中包含的某个状态变量来指示设备执行某种动作。发送控制请求的URL包含在设备描述文件中。控制点调用UpnpSendAction()或UpnpSendActionAsync()来改变状态。SDK通过HTTP M-POST命令产生一个SOAP动作。
8. 设备更改其内部变量的状态并产生一个SOAP应答消息。
9. 设备能通知客户端其状态的改变,不管其改变是由于如第8步中的显示动作还是设备自身的隐式改变。设备调用UpnpNotify()或UpnpNotifyExt()来发送更新。SDK通过HTTP的单播NOTIFY消息自动通知所有已订阅的控制点。

## **UPNP SDK架构**



### **设备/控制端程序**

客户提供客户端或服务器程序运行在 SDK 的顶层。客户端或服务器程序实现一个特定服务的功能性。例如,一个网关服务,服务器软件实现“~ 使能”(Enable Internet)功能,控制点软件能够使用 UPnP 控制它。

### **SDK API**

SDK API 从控制点和服务程序中抽象出核心 UPnP 协议的细节并给应用程序访问功能提供一个统一的接口。这使得开发者可以从他们关注 SSDP,GENA 和 SOAP 协议细节的编码中解脱出来。

### **SSDP**

SSDP 模块实现了简单服务发现协议(Simple Service Discovery Protocol),提供 UPnP 的发现部分(phase)。这个模块允许控制点发送多播信息搜索网络上的服务和设备并接受应答。

* 当网络上有新服务通告时,它也提供通知提醒。
* 该模块同时允许设备在网络上多播他们的服务通告。

### **迷你 Web 服务器(Mini Web Server)**

迷你 Web 服务器模块处理标准的 HTTP GET 请求。

### **GENA**

GENA 模块实现通用事件提醒架构(General Event Notification Architecture),他实现UPnP 的事件部分(Eventing Phase)。控制点使用该模块订阅和退订感兴趣的服务。服务应用程序从该模块中接收订阅和退订提醒并产生相应的事件。

### **SOAP**

SOAP 模块实现简单对象访问协议(Simple Object Access Protocol),他提供 UPnP 的控制部分(Control Phase)。控制点使用该模块产生相应的 XML 文档来检索和改变一个服务的状态表。服务器使用该模块解码控制请求并产生正确的应答。

### **HTTP**

HTTP 语法分析模块对接收的信息的 HTTP 头，进行语法分析同时构建相应的发送信息的头。他能够处理 HTTP/1.0 和 HTTP/1.1 头。他也支持 HTTP/1.1 的分块编码语法。SDK 缺省不支持使用HTTP/1.1 的 GET 请求的分块编码。不过,当接受到分块编码信息时,他完全支持对他们的解码。

## **实战**

编写一个UPNP控制点

控制点应用程序的基本步骤如下:

1. 使用以下基本步骤安装和初始化控制点:

* 使用 UpnpInit()初始化 SDK。
* 使用 UpnpRegisterClient()注册一个控制点(客户端)回调函数。

1. 使用 UpnpSearchAsync()查找感兴趣的设备。
2. 使用 UpnpDownloadXmlDoc()或者是 UpnpHttp()家族的函数下载描述文档。
3. 使用 UpnpSubscribe()或 UpnpSubscribeAsync()订阅感兴趣的服务。
4. 使用 UpnpSendAction()或 UpnpSendActionAsync()通过改变状态来让设备执行某些感  
   兴趣的动作。
5. 使用以下步骤来关闭控制点:

* 使用 UpnpUnRegisterClient()从 SDK 中注销控制点。
* 使用 UpnpFinish()来关闭 SDK。