UPnP Technical basics: UPnP Device Architecture (UDA)

July 2014

UPnP Functionality

Protocol

What steps are required

E.g. First request list of content transfer protocols then decide which one to use

Control

What does a command do

E.g. setVolume – sets volume between 1 and some device max Play – set a MediaRenderer in PlayState

Description

Which information is exchanged

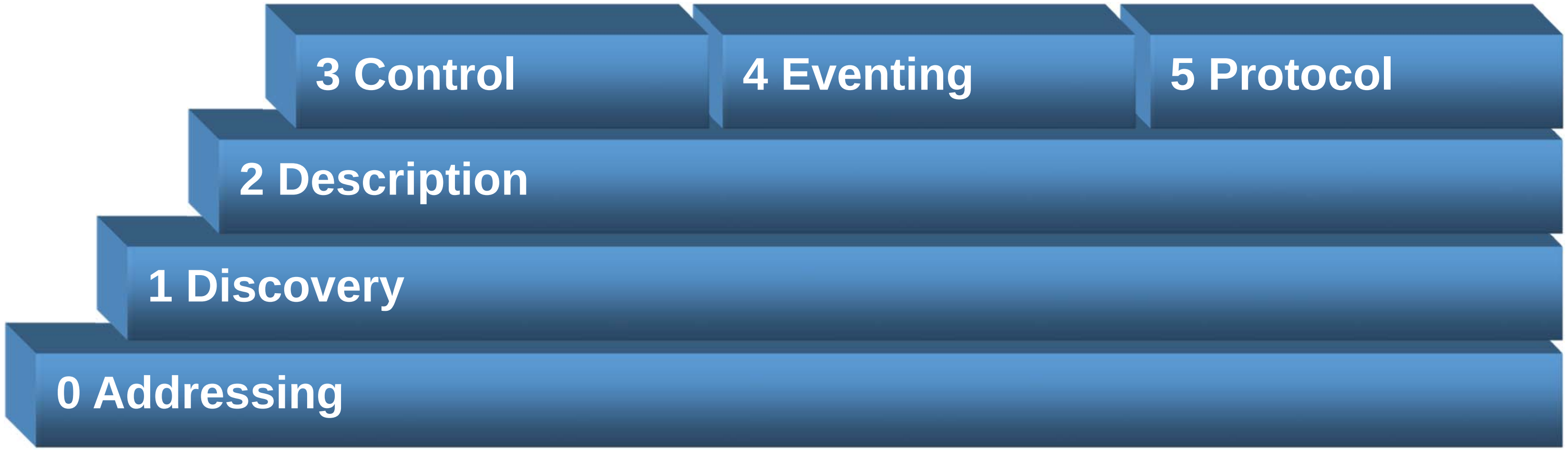
E.g. List of supported functions grouped in services, function calls, events

Discover

How do devices detect each other

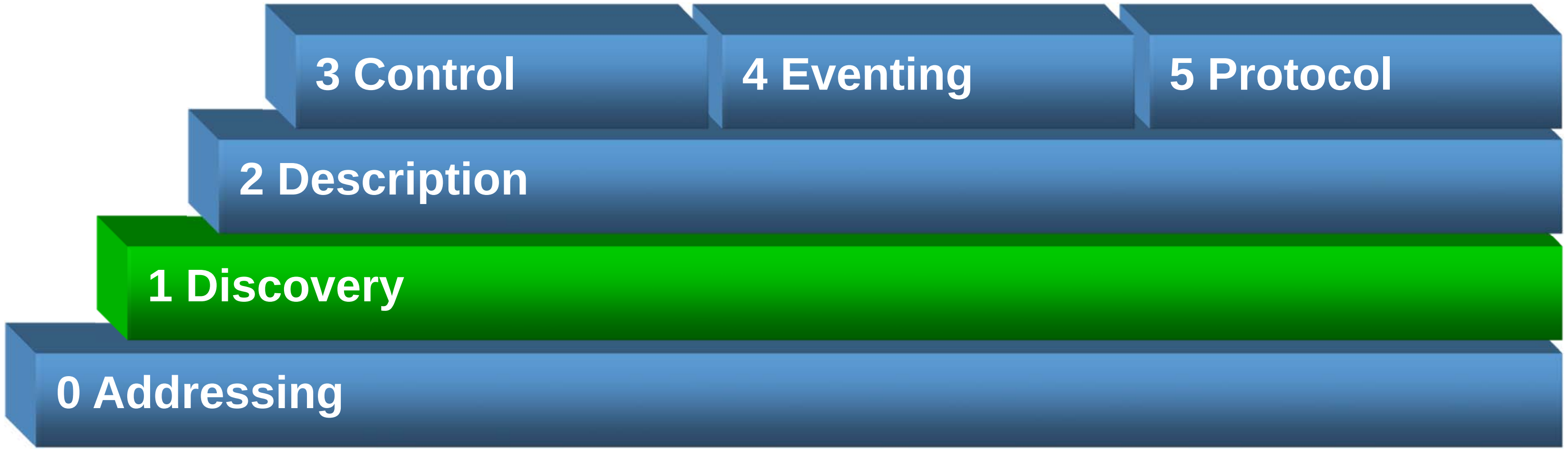
E.g. regular “alive” messages, Bye-bye messages

UPnP Phases (overview)



0	Control points and devices get IP addresses using DHCP (, or AutoIP)
1	Control point finds interesting device
2	Control point learns about device capabilities
3	Control point invokes actions on device
4	Control point listens to state changes of device
5	Control point interacts with a device with sequences of commands and events

UPnP Phases (Discovery)



1 Find devices: Listen for SSDP Alive messages, or issue search

```
M-SEARCH * HTTP/1.1
HOST: 239.255.255.250:1900
MAN: "ssdp:discover"
MX: seconds to delay response
ST: search target
```

```
HTTP/1.1 200 OK
CACHE-CONTROL: max-age = seconds
            until advertisement expires
LOCATION: URL for UPnP description
            for root device
ST: search target
USN: advertisement UUID
```

SSDP

SSDP - IETF Draft *Simple Service Discovery Protocol*

Based on UDP Multicast

Devices and services post Alive message at regular intervals

- Usually repeated 3 times (because UDP messages might be lost)
- Repeated every few seconds (e.g. 10 secs)
- Determines worst case detection time of a device

Other Messages: Search, Bye-Bye

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age = sec
LOCATION: URL for UPnP descripti
NT: search target
NTS: ssdp:alive
USN: advertisement UUID
```

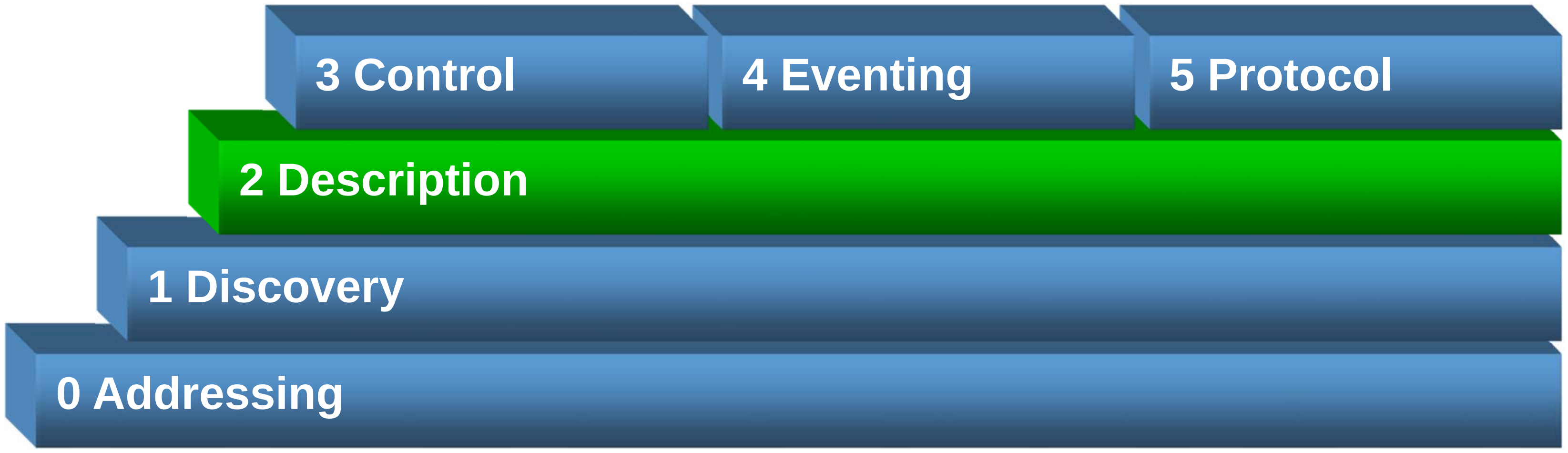
Multicast address

Port usually 1900

DLNA: port 1900 is mandatory

Location of device for further communication

UPnP Phases (Description)



2 Use URL from SSDP message to get device description

Use URL from SSDP message or device description to get service description

Descriptions use XML to describe what services and functions a device offers

Find out which Optional functions are available

Find out what vendor specific functions are available

Devices

UPnP Device:

- Not a real physical device
- Representation of a logical entity
- A set of functions and state

UPnP Device examples:

- Media Server
- Media Renderer
- Internet Gateway device
- Printer

- No 1 to 1 mapping with real world devices/boxes

TV:	Media Player + Media Renderer
PC:	Media Server + Printer + Scanner
Wireless Access point:	Wireless Access point device + Printer device (proxy)

Device Description – general info

```
<?xml version="1.0"?>
<root xmlns="urn:schemas-upnp-org:device-1-0">
 <URLBase>base URL for all relative URLs</URLBase>
  <device>
    <friendlyName>short user-friendly title</friendlyName>
    <manufacturer>manufacturer name</manufacturer>
    <manufacturerURL>URL to manufacturer site</manufacturerURL>
    <modelDescription>long user-friendly title</modelDescription>
    <modelName>model name</modelName>
    <modelNumber>model number</modelNumber>
    <modelURL>URL to model site</modelURL>
    <serialNumber>manufacturer's serial number</serialNumber>
    <UDN>uuid:UUID</UDN>
    <UPC>Universal Product Code</UPC>
  <deviceType>urn:schemas-upnp-org:device:deviceType</deviceType>
  <serviceList>
   <service>
    <serviceType>urn:schemas-upnp-org:service:serviceType:v</serviceType>
    <serviceId>urn:upnp-org:serviceId:serviceID</serviceId>
    <SCPDURL>URL to service description</SCPDURL>
    <controlURL>URL for control</controlURL>
    <eventSubURL>URL for eventing</eventSubURL>
   </service>
   Declarations for other services (if any) go here
  </serviceList>
  <deviceList>Description of embedded devices (if any) go here</deviceList>
  <iconList>
   <icon>
    <mimetype>image/format</mimetype>
    <width>horizontal pixels</width>
    <height>vertical pixels</height>
    <depth>color depth</depth>
    <url>URL to icon</url>
   </icon>
   XML to declare other icons, if any, go here
  </iconList>
  <presentationURL>URL for presentation</presentationURL>
  </device>
 <specVersion>
  <major>1</major> <minor>0</minor>
 </specVersion>
</root>
```

Device Description – service information

```
<?xml version="1.0"?>
<root xmlns="urn:schemas-upnp-org:device-1-0">
 <URLBase>base URL for all relative URLs</URLBase>
 <device>
   <friendlyName>short user-friendly title</friendlyName>
   <manufacturer>manufacturer name</manufacturer>
   <manufacturerURL>URL to manufacturer site</manufacturerURL>
   <modelDescription>long user-friendly title</modelDescription>
   <modelName>model name</modelName>
   <modelNumber>model number</modelNumber>
   <modelURL>URL to model site</modelURL>
   <serialNumber>manufacturer's serial number</serialNumber>
   <UDN>uuid:UUID</UDN>
   <UPC>Universal Product Code</UPC>
  <deviceType>urn:schemas-upnp-org:device:deviceType :v</deviceType>
  <serviceList>
   <service>
    <serviceType>urn:schemas-upnp-org:service:serviceType:v</serviceType>
    <serviceId>urn:upnp-org:serviceId:serviceID</serviceId>
    <SCPDURL>URL to service description</SCPDURL>
    <controlURL>URL for control</controlURL>
    <eventSubURL>URL for eventing</eventSubURL>
   </service>
   Declarations for other services (if any) go here
  </serviceList>
  <deviceList>Description of embedded devices (if any) go here</deviceList>
   <iconList>
    <icon>
     <mimetype>image/format</mimetype>
     <width>horizontal pixels</width>
     <height>vertical pixels</height>
     <depth>color depth</depth>
     <url>URL to icon</url>
    </icon>
    XML to declare other icons, if any, go here
   </iconList>
   <presentationURL>URL for presentation</presentationURL>
 </device>
  <specVersion>
   <major>1</major> <minor>0</minor>
  </specVersion>
</root>
```

Device Description – display information

```
<?xml version="1.0"?>
<root xmlns="urn:schemas-upnp-org:device-1-0">
  <URLBase>base URL for all relative URLs</URLBase>
  <device>
    <friendlyName>short user-friendly title</friendlyName>
    <manufacturer>manufacturer name</manufacturer>
    <manufacturerURL>URL to manufacturer site</manufacturerURL>
    <modelDescription>long user-friendly title</modelDescription>
    <modelName>model name</modelName>
    <modelNumber>model number</modelNumber>
    <modelURL>URL to model site</modelURL>
    <serialNumber>manufacturer's serial number</serialNumber>
    <UDN>uuid:UUID</UDN>
    <UPC>Universal Product Code</UPC>
   <deviceType>urn:schemas-upnp-org:device:deviceType</deviceType>
    <serviceList>
      <service>
        <serviceType>urn:schemas-upnp-org:service:serviceType:v</serviceType>
        <serviceId>urn:upnp-org:serviceId:serviceID</serviceId>
        <SCPDURL>URL to service description</SCPDURL>
        <controlURL>URL for control</controlURL>
        <eventSubURL>URL for eventing</eventSubURL>
      </service>
      Declarations for other services (if any) go here
    </serviceList>
    <deviceList>Description of embedded devices (if any) go here</deviceList>
    <iconList>
      <icon>
        <mimetype>image/format</mimetype>
        <width>horizontal pixels</width>
        <height>vertical pixels</height>
        <depth>color depth</depth>
        <url>URL to icon</url>
      </icon>
      XML to declare other icons, if any, go here
    </iconList>
    <presentationURL>URL for presentation</presentationURL>
  </device>
  <specVersion>
    <major>1</major> <minor>0</minor>
  </specVersion>
</root>
```

Services

Logical grouping of functions and (state) variable definition

Examples:

- Connection Manager service: A set of functions that are used to negotiate which protocol to use for communication
- Content Directory service: set of functions that describe the content available on a server
- Rendering Control (service): set of functions that change settings like volume, brightness, contrast etc.
- Media Renderer: set of functions to control playback (via the network)
- State variables are used 2 two ways
    - Conveying state, like SystemUpdateID of the CDS
    - Type definition of arguments in actions
        - preceded by A_ARG_TYPE_

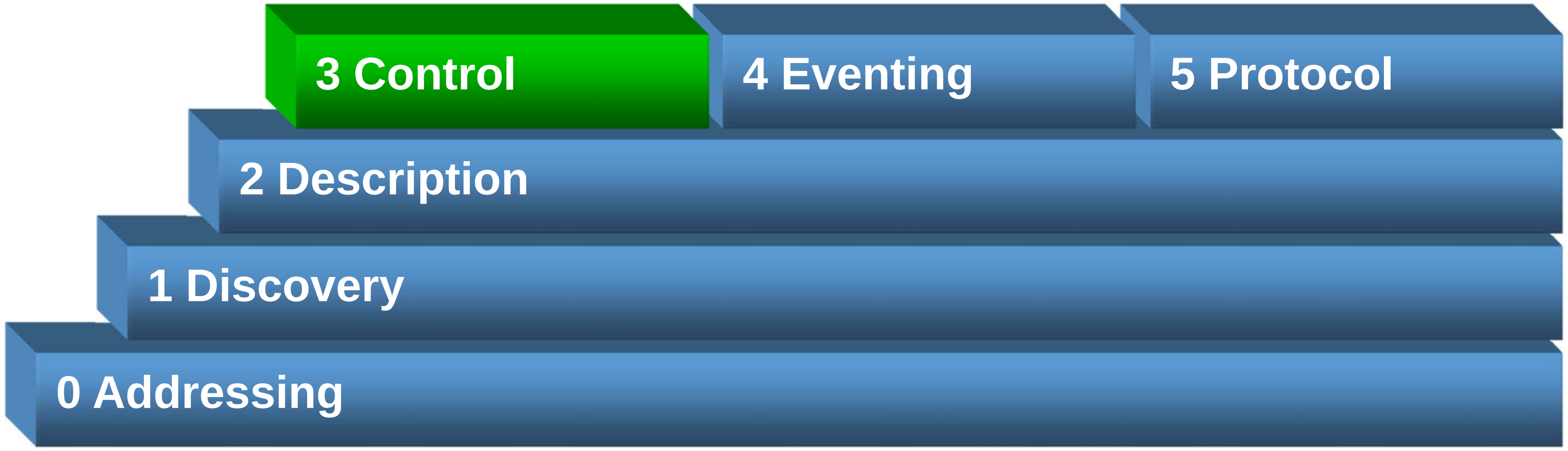
Service Description - actions

```
<?xml version="1.0"?>
<scpd xmlns="urn:schemas-upnp-org:service-1-0">
  <actionList>
    <action>
      <name>actionName</name>
      <argumentList>
        <argument>
          <name>formalParameterName</name>
          <direction>in xor out</direction>
          <retval />
          <relatedStateVariable>stateVariableName</relatedStateVariable>
        </argument>
       Declarations for other arguments (if any) go here
      </argumentList>
    </action>
    Declarations for other actions (if any) go here
  </actionList>
  <serviceStateTable>
    <stateVariable sendEvents="yes" xor "no">
      <name>variableName</name>
      <dataType>variable datatype</dataType>
      <defaultValue>default value</defaultValue>
      <allowedValueRange>
        <minimum>minimum value</minimum>
        <maximum>maximum value</maximum>
        <step>increment value</step>
      </allowedValueRange>
    </stateVariable>
    <stateVariable sendEvents="yes" xor "no">
      <name>variableName</name>
      <dataType>variable data type</dataType>
      <defaultValue>default value</defaultValue>
      <allowedValueList>
        <allowedValue>some value</allowedValue>
        <allowedValue>some value</allowedValue>
      </allowedValueList>
    </stateVariable>
    Declarations for other state variables (if any) go here
  </serviceStateTable>
  <specVersion>
    <major>1</major> <minor>0</minor>
  </specVersion>
</scpd>
```

Service Description – state variables

```
<?xml version="1.0"?>
<scpd xmlns="urn:schemas-upnp-org:service-1-0">
  <actionList>
    <action>
      <name>actionName</name>
      <argumentList>
        <argument>
          <name>formalParameterName</name>
          <direction>in xor out</direction>
          <retval />
          <relatedStateVariable>stateVariableName</relatedStateVariable>
        </argument>
        Declarations for other arguments (if any) go here
      </argumentList>
    </action>
    Declarations for other actions (if any) go here
  </actionList>
  <serviceStateTable>
    <stateVariable sendEvents="yes" xor "no">
      <name>variableName</name>
      <dataType>variable datatype</dataType>
      <defaultValue>default value</defaultValue>
      <allowedValueRange>
        <minimum>minimum value</minimum>
        <maximum>maximum value</maximum>
        <step>increment value</step>
      </allowedValueRange>
    </stateVariable>
    <stateVariable sendEvents="yes" xor "no">
      <name>variableName</name>
      <dataType>variable data type</dataType>
      <defaultValue>default value</defaultValue>
      <allowedValueList>
        <allowedValue>some value</allowedValue>
        <allowedValue>some value</allowedValue>
      </allowedValueList>
    </stateVariable>
    Declarations for other state variables (if any) go here
  </serviceStateTable>
  <specVersion>
    <major>1</major> <minor>0</minor>
  </specVersion>
</scpd>
```

UPnP Phases (Control)



3 Send actions to a device using SOAP
Receive responses using SOAP

Remote procedure call mechanism based on SOAP.

Remote Procedure Calls

Based on SOAP (IETF Draft *Simple Object Access Protocol)*
= XML messages using HTTP headers

```
POST path of control URL HTTP/1.1
HOST: host of control URL:port of control URL
CONTENT-TYPE: text/xml; charset="utf-8"
SOAPACTION: "urn:schemas-upnp-org:service:serviceType:v#actionName"

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
   s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <s:Body>
    <u:actionName xmlns:u="urn:schemas-upnp-org:service:serviceType:v">
      <argumentName>in arg value</argumentName>
      other in args and their values (if any) go here
    </u:actionName>
  </s:Body>
</s:Envelope>
```

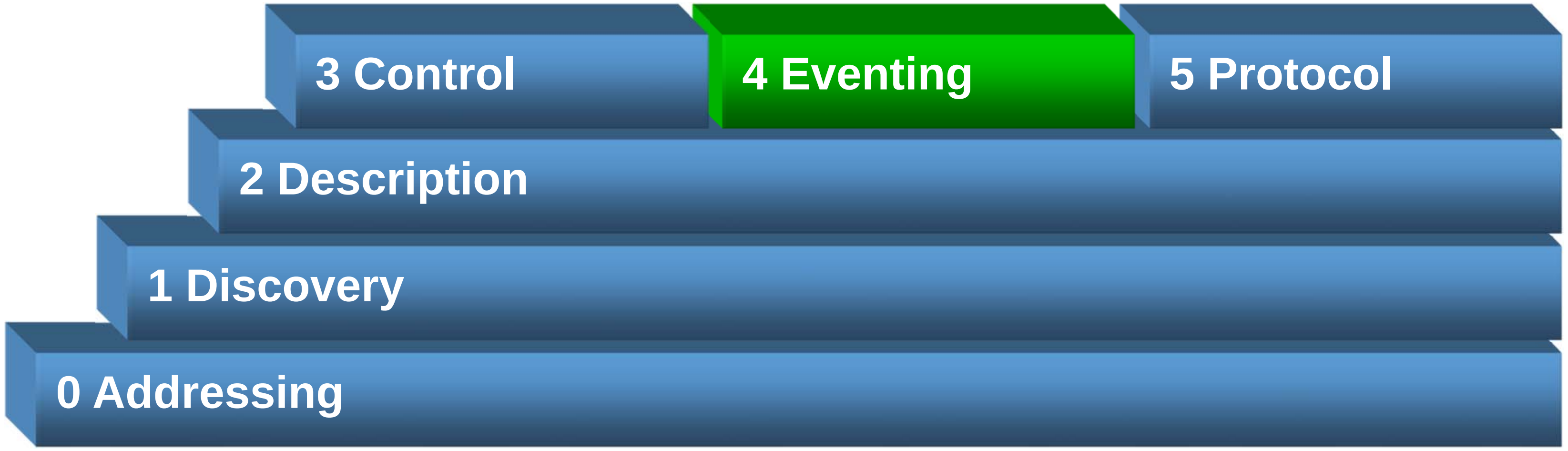
Remote Procedure Call Response

Response by device:

```
HTTP/1.1 200 OK
CONTENT-TYPE: text/xml; charset="utf-8"

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
   s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <s:Body>
    <u:actionNameResponse
        xmlns:u="urn:schemas-upnp-org:service:serviceType:v">
      <argumentName>out arg value</argumentName>
      other out args and their values (if any) go here
    </u:actionNameResponse>
  </s:Body>
</s:Envelope>
```

UPnP Phases (Eventing)



4 Control points can subscribe to events from a certain device

Events are send by device to control point when a state variable changes. E.g. ContainerUpdateID, Volume (LastChange)

GENA - IETF Draft General Event Notification Architecture

Message over HTTP via TCP, but also via UDP multicast

GENA - Subscribe

```
SUBSCRIBE publisher path HTTP/1.1
HOST: publisher host:publisher port
CALLBACK: <delivery URL>
NT: upnp:event
TIMEOUT: Second-requested subscription duration
```

Control points subscribe per Service and Device.

- Control point is in control for which service it will receive the notifications
- All notifications per service will be received.
- Have to re-subscribe before TIMEOUT elapses

GENA - Notify

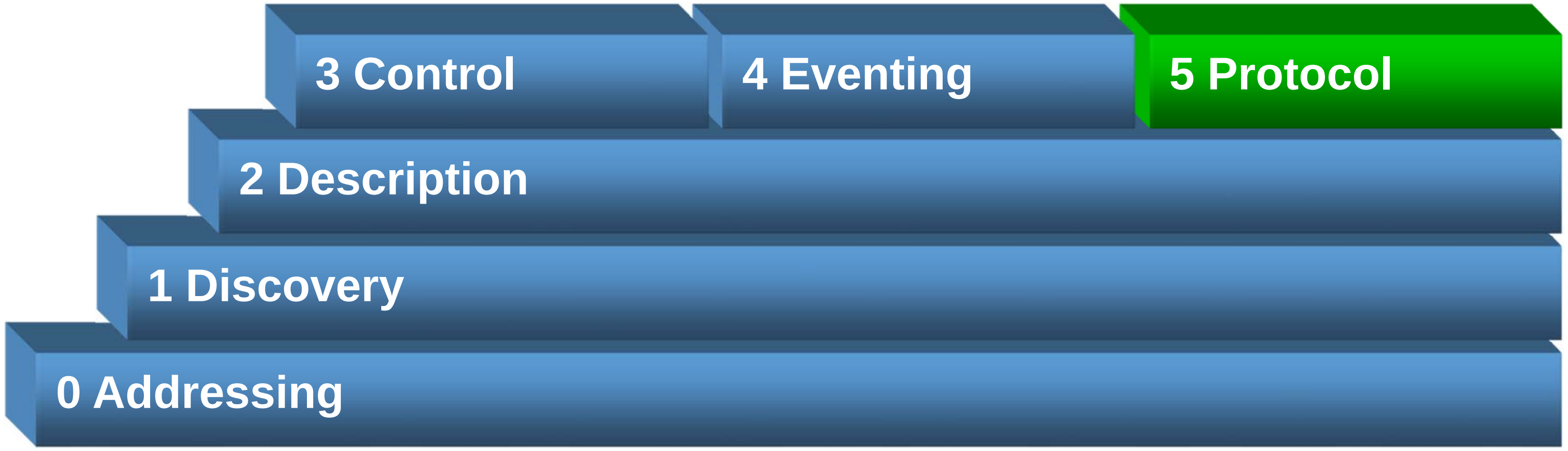
```
NOTIFY delivery path HTTP/1.1
HOST: delivery host:delivery port
CONTENT-TYPE: text/xml
NT: upnp:event
NTS: upnp:propchange
SID: uuid:subscription-UUID
SEQ: event key

<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>
```

Device sends same property set to each subscribed control point

- Sequence (SEQ) is tracking initial & following notification
- Initial notification is ALWAYS sent by the device

UPnP Phases (Protocol)



5 List of logical sequences on top of Control and Eventing.

Control point interacts with multiple devices to create an scenario.

Also control points should listen to events, so they know what has changed in the eco system, and should reflect this in their UI.

Protocol example (sequence of actions)

Steps needed to play an item from a Media Server on a Media Renderer

1. Select a Media server
2. Invoke Browse(), to present content for selection for playback
3. Select a Media Server
4. Invoke GetProtocolInfo() on the Media Renderer
5. Match the ProtocolInfo from the content and the MediaRenderer
6. Invoke SetAVTransportURI() with the matched content
7. Invoke Play(), to start the playback of the content

UPnP℠ FORUM

For the interconnected lifestyle