AES 143
Network Audio Track

How to make an AES70 controller

Session NA10
October 21, 2017

Agenda

- **AES70 Concepts for Controllers**

  Morten Lave, Principal, All Access Consulting
- **Issues in Controller Design**

  Marc Smaak, Manager - Platforms Group, Bosch Communications
- **What Kind of Controller Should I Make?**

  Tom de Brouwer, Software Engineer, Bosch Communications
- **Developer Resources Available**

  Tom de Brouwer, Software Engineer, Bosch Communications
- **Examples / Demonstrations**

Selected AES70 Concepts for controllers

- DNS-SD, registration, browsing etc.
- New connection modalities coming (UDP, Web-sockets, etc.)
- Subscriptions
- Enumeration and name searching
- AES70 and User Interfaces
- Keeping the connection and knowing when it is gone
- Security

DNS-SD, registration, browsing etc.

- Formerly known as Bonjour
- AES70 is using a registered service identifier
- Devices register them self with the service (multicast, here I am)
- Controllers can browse registrations, get an inventory of devices
- The registration will contain IP address and port number
- The controller can make a TCP/IP connection to the device.
- Through the Device object and other required objects the controller can obtain detailed information about the device.

New connection modalities coming (UDP, Web-sockets, etc.)

- Current standard defines OCP.1 which is TCP/IP
- The layer separation allows for multiple transport layer protocols
- Web-sockets
- UDP
- Other such as USB, Serial etc.

Subscriptions

- Keeping multiple controllers up to date
- Subscribe to certain or all objects
- Get notified when changes occur
- Subscribing to sensors
- Observers

Enumeration and name searching

- The device object can be enumerated
- Depth-first walk of all objects in a device
- Objects can also be found through search by name
- Includes wildcards, fully recursive

AES70 and User Interfaces

- AES70 purposely does not specify UI aspects
- It does allow for logical grouping of objects which can direct GUI generation
- The strong typing enables automatic mapping to GUI objects
- There are no controller classes, controllers control device objects
- An example of a controller implementation could create a proxy class instance for each object in a device.
- Example, execute SetGain(v) on a proxy object will effectively result in that method being executed in the device using the protocol.

Keeping the connection and knowing when it is gone

- Keep alive mechanism on top of what the transport layer might offer.
- Allows controller and device to know when a session is done
- Both ends agree on a keep alive interval
- If no communication has been seen in that interval the session will end
- Improves connection loss detection on both connection and connectionless transport layers.
- When a session is terminated all state related to the connection is freed

- Reset device command, can be broadcast

Security

- AES70 offers secure operation on the network as an option (TLS)
- Installation of pre-shared keys is application specific.
- Access control is not part of AES70.
- A trusted controller can implement access control in an application specific manner.

AES70 controller

Controlling audio systems

Can it be as simple as it used to be?

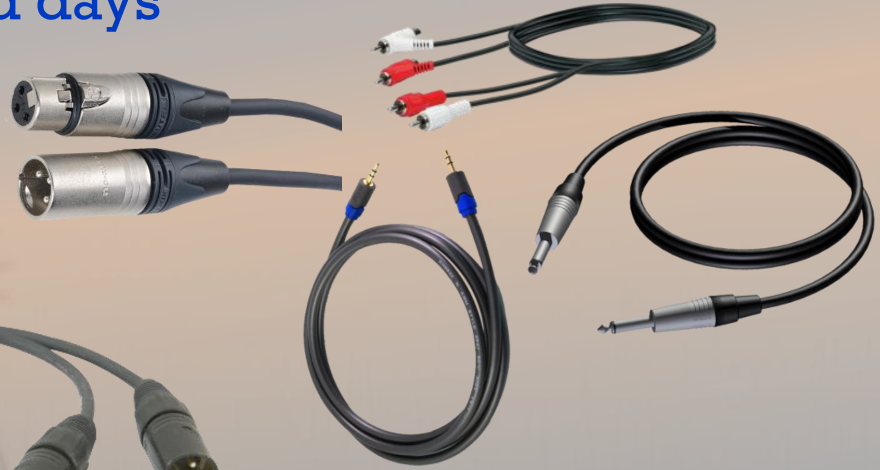
Marc Smaak
Bosch Security systems

Connection management in the old days

- The user connecting the cables

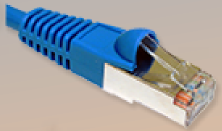
- Mostly unicast
  - but also multicast

- When needed select the right adaptation

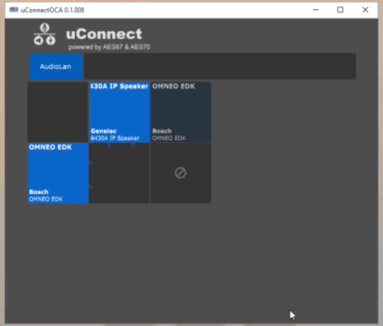


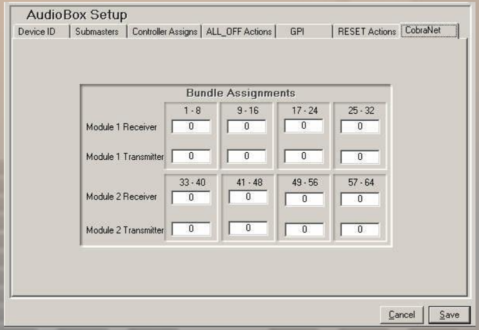
Connection management today

- Only one cable type needed



- Many SW solutions to do connection management




AudioBox Setup

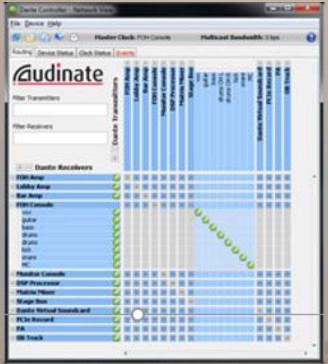
Device ID | Submasters | Controller Assigns | ALL_OFF Actions | GPI | RESET Actions | CobraNet

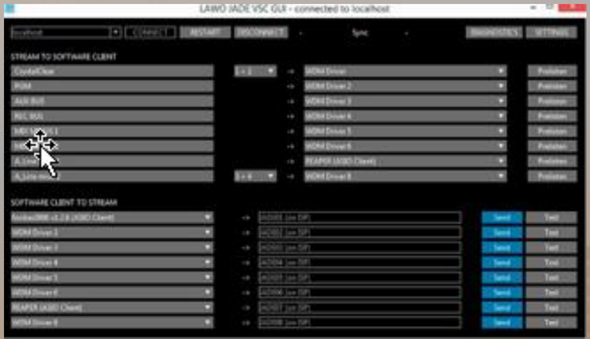
Bundle Assignments

	1 - 8	9 - 16	17 - 24	25 - 32
Module 1 Receiver	0	0	0	0
Module 1 Transmitter	0	0	0	0

	33 - 40	41 - 48	49 - 56	57 - 64
Module 2 Receiver	0	0	0	0
Module 2 Transmitter	0	0	0	0

Cancel Save




LAWO JADE VSC GUI - connected to localhost
STREAM TO SOFTWARE CLIENT
SOFTWARE CLIENT TO STREAM

All using different protocols

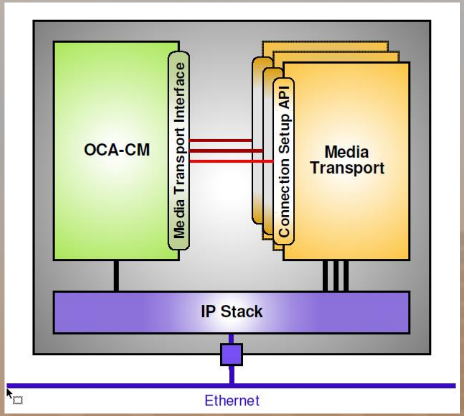
Connection management with AES70

Manual connection management

- ➢ Find your equipment to connect
- ➢ Detect connector types
- ➢ Plug-in the cable

AES70 connection management

- ➢ Discover the devices
- ➢ Capability enumeration
- ➢ Command sender & receiver to connect

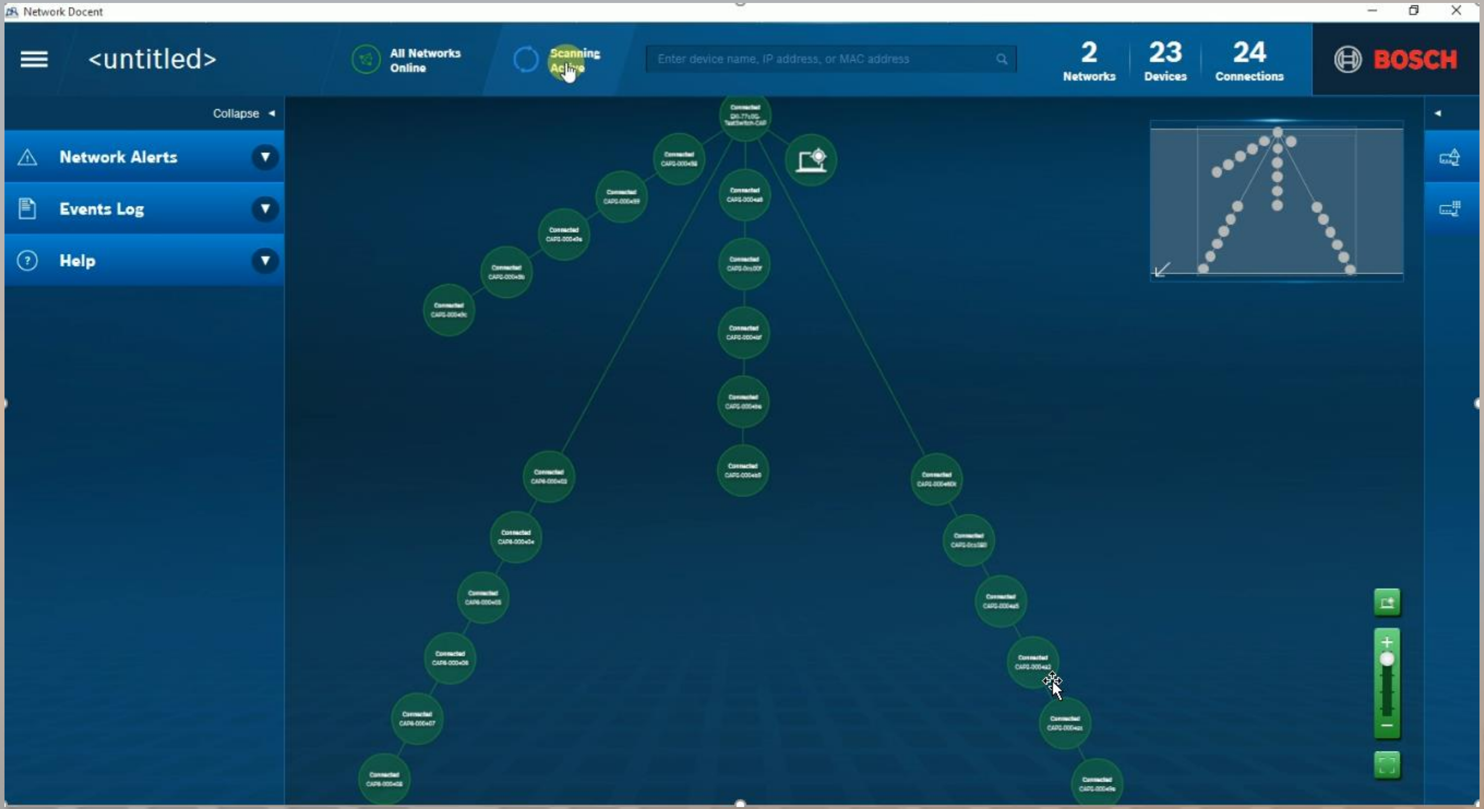


Discover devices

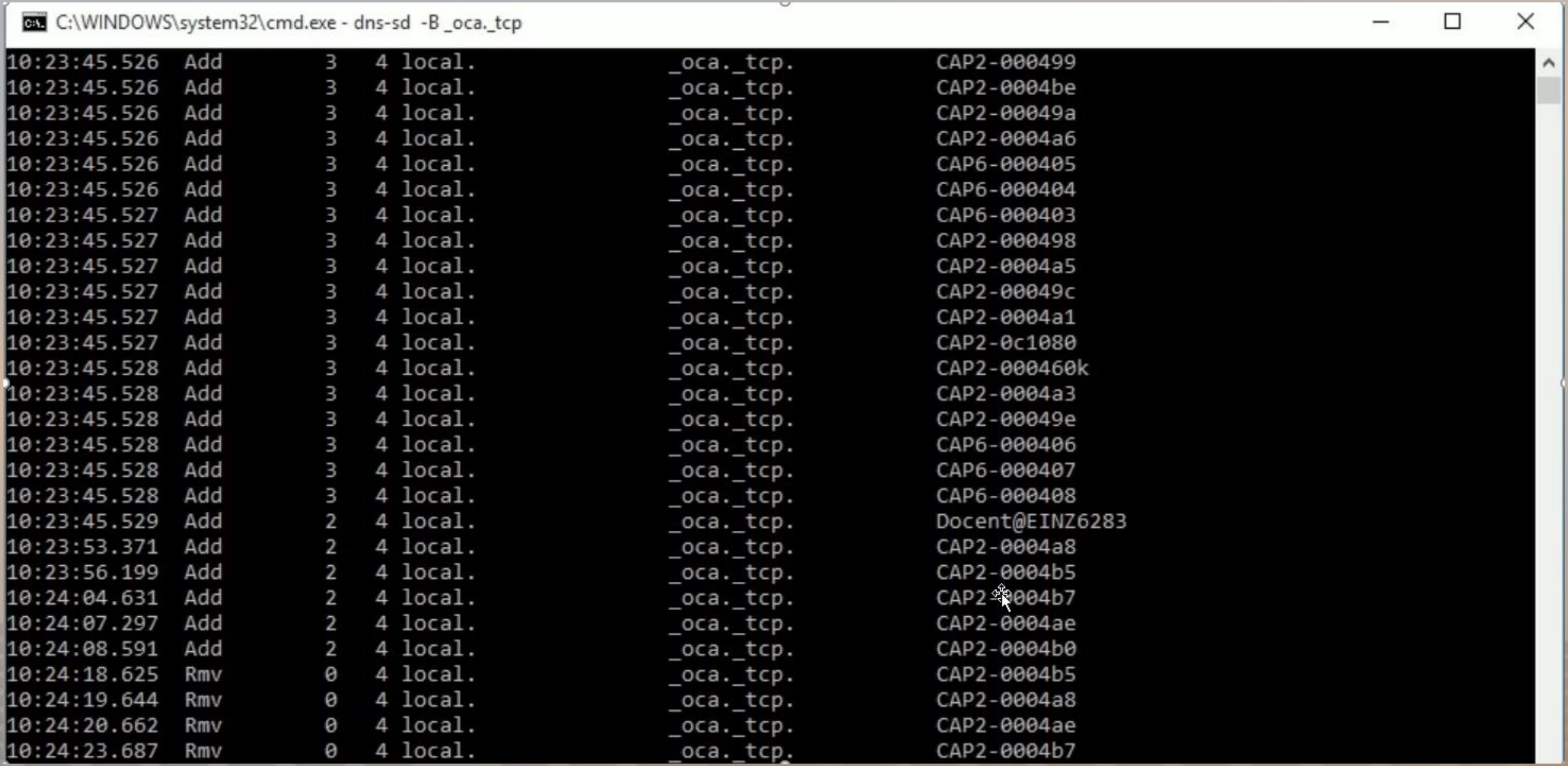
- AES70 uses standard DNS-SD for device and service discovery
  - Simply browse for AES70 supporting devices
- No need for manual configuration of device lists
- No need to remember IP addresses

How does this work in practice?

How to make an AES70 controller



How to make an AES70 controller

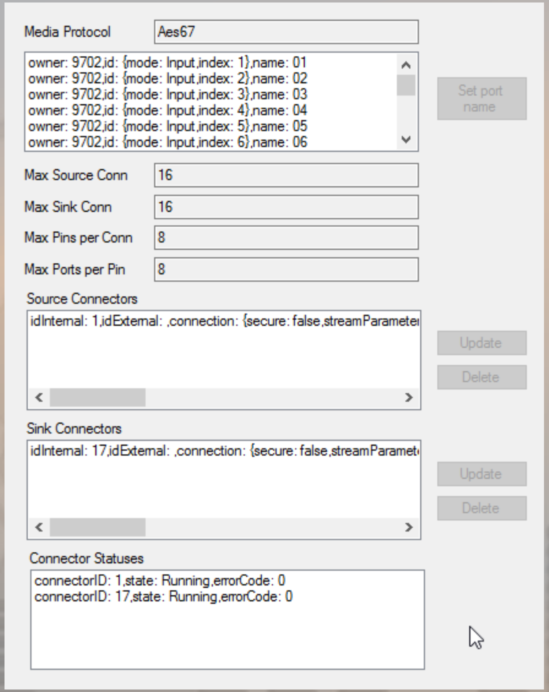


C:\WINDOWS\system32\cmd.exe - dns-sd -B _oca._tcp

```
10:23:45.526  Add        3   4 local.              _oca._tcp.          CAP2-000499
10:23:45.526  Add        3   4 local.              _oca._tcp.          CAP2-0004be
10:23:45.526  Add        3   4 local.              _oca._tcp.          CAP2-00049a
10:23:45.526  Add        3   4 local.              _oca._tcp.          CAP2-0004a6
10:23:45.526  Add        3   4 local.              _oca._tcp.          CAP6-000405
10:23:45.526  Add        3   4 local.              _oca._tcp.          CAP6-000404
10:23:45.527  Add        3   4 local.              _oca._tcp.          CAP6-000403
10:23:45.527  Add        3   4 local.              _oca._tcp.          CAP2-000498
10:23:45.527  Add        3   4 local.              _oca._tcp.          CAP2-0004a5
10:23:45.527  Add        3   4 local.              _oca._tcp.          CAP2-00049c
10:23:45.527  Add        3   4 local.              _oca._tcp.          CAP2-0004a1
10:23:45.527  Add        3   4 local.              _oca._tcp.          CAP2-0c1080
10:23:45.528  Add        3   4 local.              _oca._tcp.          CAP2-000460k
10:23:45.528  Add        3   4 local.              _oca._tcp.          CAP2-0004a3
10:23:45.528  Add        3   4 local.              _oca._tcp.          CAP2-00049e
10:23:45.528  Add        3   4 local.              _oca._tcp.          CAP6-000406
10:23:45.528  Add        3   4 local.              _oca._tcp.          CAP6-000407
10:23:45.528  Add        3   4 local.              _oca._tcp.          CAP6-000408
10:23:45.529  Add        2   4 local.              _oca._tcp.          Docent@EINZ6283
10:23:53.371  Add        2   4 local.              _oca._tcp.          CAP2-0004a8
10:23:56.199  Add        2   4 local.              _oca._tcp.          CAP2-0004b5
10:24:04.631  Add        2   4 local.              _oca._tcp.          CAP2-0004b7
10:24:07.297  Add        2   4 local.              _oca._tcp.          CAP2-0004ae
10:24:08.591  Add        2   4 local.              _oca._tcp.          CAP2-0004b0
10:24:18.625  Rmv        0   4 local.              _oca._tcp.          CAP2-0004b5
10:24:19.644  Rmv        0   4 local.              _oca._tcp.          CAP2-0004a8
10:24:20.662  Rmv        0   4 local.              _oca._tcp.          CAP2-0004ae
10:24:23.687  Rmv        0   4 local.              _oca._tcp.          CAP2-0004b7
```

Capability enumeration

- See what the device has and what is available
  - Streaming protocol
  - inputs/outputs
    - Including the used once
  - Sample rate
  - Clock domains



Media Protocol	Aes67

```
owner: 9702,id: {mode: Input,index: 1},name: 01
owner: 9702,id: {mode: Input,index: 2},name: 02
owner: 9702,id: {mode: Input,index: 3},name: 03
owner: 9702,id: {mode: Input,index: 4},name: 04
owner: 9702,id: {mode: Input,index: 5},name: 05
owner: 9702,id: {mode: Input,index: 6},name: 06
```

Max Source Conn	16
Max Sink Conn	16
Max Pins per Conn	8
Max Ports per Pin	8

Source Connectors

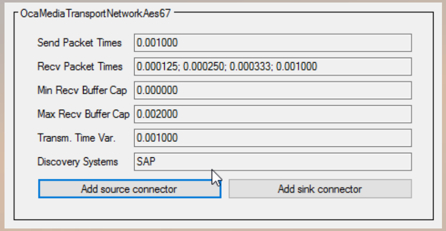
```
idInternal: 1,idExternal: ,connection: {secure: false,streamParamete
```

Sink Connectors

```
idInternal: 17,idExternal: ,connection: {secure: false,streamParamet
```

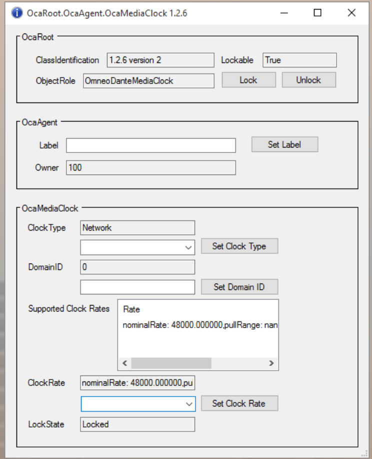
Connector Statuses

```
connectorID: 1,state: Running,errorCode: 0
connectorID: 17,state: Running,errorCode: 0
```



OcaMediaTransportNetworkAes67

Send Packet Times	0.001000
Recv Packet Times	0.000125; 0.000250; 0.000333; 0.001000
Min Recv Buffer Cap	0.000000
Max Recv Buffer Cap	0.002000
Transm. Time Var.	0.001000
Discovery Systems	SAP



OcaRoot.OcaAgent.OcaMediaClock 1.2.6

OcaRoot

ClassIdentification	1.2.6 version 2	Lockable	True
ObjectRole	OmneoDanteMediaClock		

OcaAgent

Label	
Owner	100

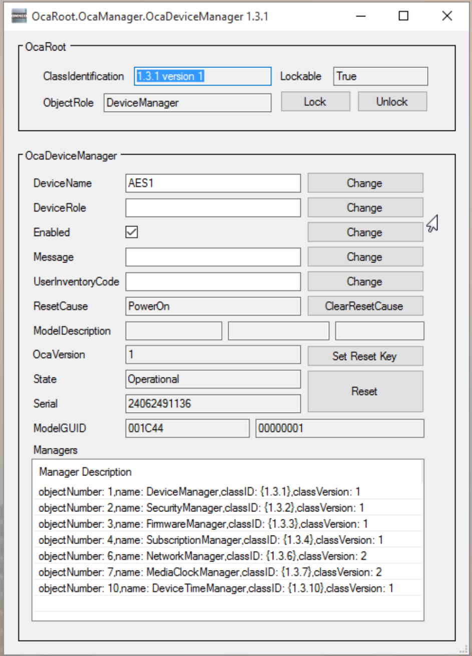
OcaMediaClock

ClockType	Network
DomainID	0
Supported Clock Rates	Rate nominalRate: 48000.000000,pullRange: nan
ClockRate	nominalRate: 48000.000000,pu
LockState	Locked

With this information you know what can be connected to what

Capability enumeration

- If you know your device this info can be hardcoded in your controller
- AES70 has a globally unique Model GUID for this available (64 bit)
    - See OCA device manager
- Similar to experienced user who knows all details of a device by the model number



ClassIdentification	1.3.1 version 1
Lockable	True
ObjectRole	DeviceManager
DeviceName	AES1
DeviceRole	
Enabled	✓
Message	
UserInventoryCode	
ResetCause	PowerOn
ModelDescription	
OcaVersion	1
State	Operational
Serial	24062491136
ModelGUID	001C44 00000001

```
Manager Description
objectNumber: 1,name: DeviceManager,classID: {1.3.1},classVersion: 1
objectNumber: 2,name: SecurityManager,classID: {1.3.2},classVersion: 1
objectNumber: 3,name: FirmwareManager,classID: {1.3.3},classVersion: 1
objectNumber: 4,name: SubscriptionManager,classID: {1.3.4},classVersion: 1
objectNumber: 6,name: NetworkManager,classID: {1.3.6},classVersion: 2
objectNumber: 7,name: MediaClockManager,classID: {1.3.7},classVersion: 2
objectNumber: 10,name: DeviceTimeManager,classID: {1.3.10},classVersion: 1
```

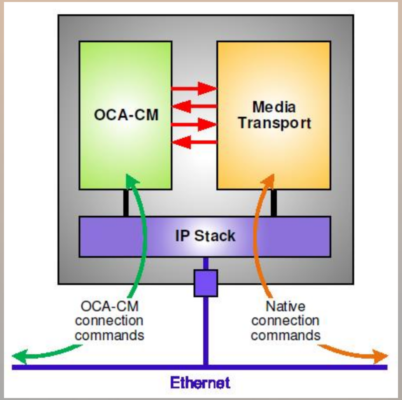
This improves performance and reduces the load on the devices

Command sender & receiver to connect

- Fully under control of controller
  - Results in successful connection: sender and receiver capabilities do match
- Very scalable limited by controller performance not the devices
  - Hundreds of connections at once no problem
- Identical mechanism for multi & unicast connections, just different IP addresses
  - One to multiple with unicast supported ➔ multiple streams
- Reliable since devices are supervised by the controller
  - Can move connection if a device fails ➔ redundancy
  - Can stop transmitter when no receivers ➔ Avoid bandwidth waist

CM advanced features

- Create streams with multiple channels to optimize bandwidth
  - Using OCA stream connector
  - Modify stream by add or removing channels (pins)
- Interoperable with proprietary protocols
  - As long as objects are updated by the device
  - Subscription mechanism to auto inform controller
- Use the build in security mechanism
  - Using standard TLS
  - Nobody can mess with your devices



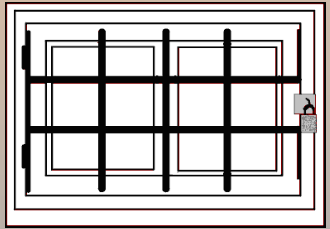
Multiple controllers

- Redundancy

- Concurrency
  - Multiple AES70 controllers
  - Proprietary & AES70 controller

Multiple controllers; What about consistency?

- AES70 supports object locking
  - Locked objects are read/write for the controller who has the lock
  - Locked objects are read only for anybody else
  - Lock can even hide the state if needed



- AES70 supports subscriptions to objects state
  - Controllers are informed of changes, no frequent polling required

Multiple controllers operation

- Concurrent controllers environment
  - Controller locks, modifies, unlocks
  - Other controllers are updated automatically when subscribed to the object.

- Redundant controllers
  - Main controller locks the device by locking the device manger
  - Backup controller subscribes to all objects in all devices to stay up to date
  - If main controller disappears lock is auto removed so backup can take over.

Generic controllers

- Generic controllers are powerful
- They can easily create interoperability between different manufactures

A generic controller can	A generic controller can not
Discover all devices	Know which object belongs to which physical connector
Discover device capabilities	Know the exact functioning of the audio processing; e.g. Equalizer parameters
Discover the signal path	

Proper naming guidelines will make it easier for a generic controller

Tom de Brouwer

- Software Architect for Bosch Security Systems
- Involved in programming AES70 products:
  RTS Intercoms, Eletrovoice, Dynacord, Bosch
- Based in The Netherlands

Agenda

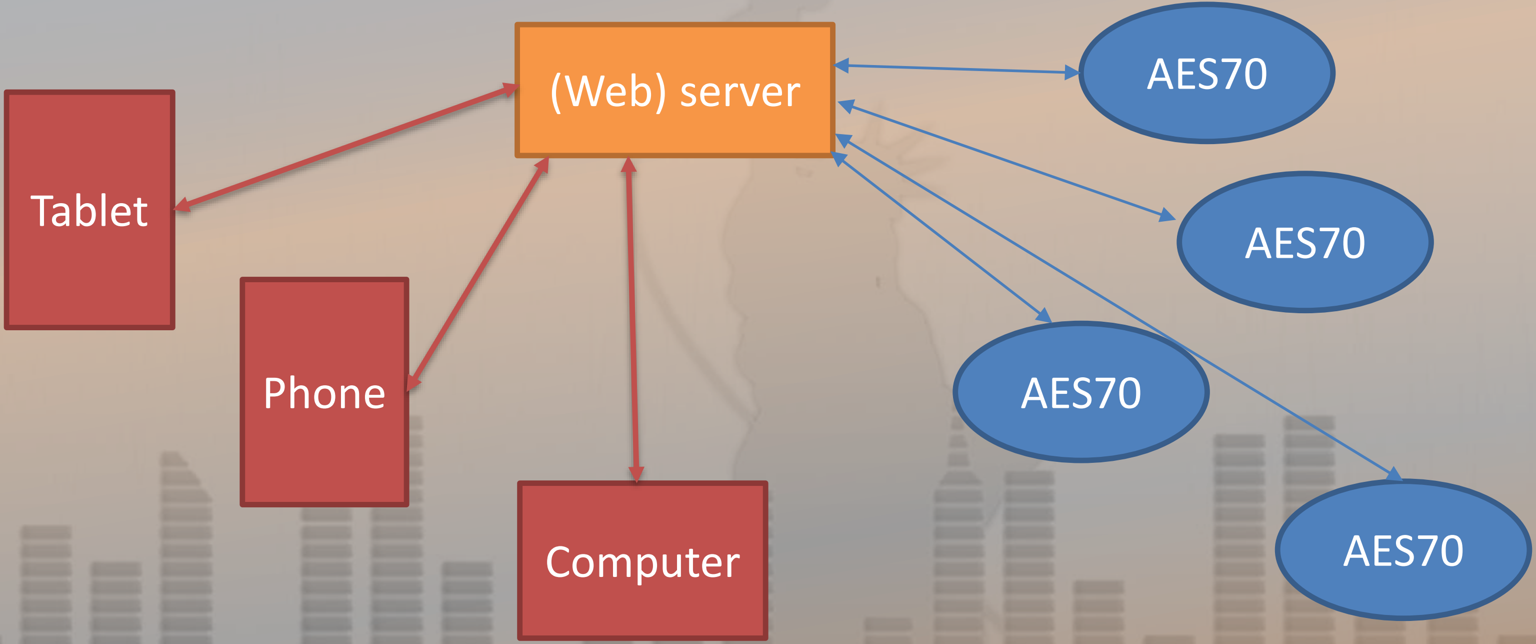
- AES70 Controller concepts

AES70 based controller concepts

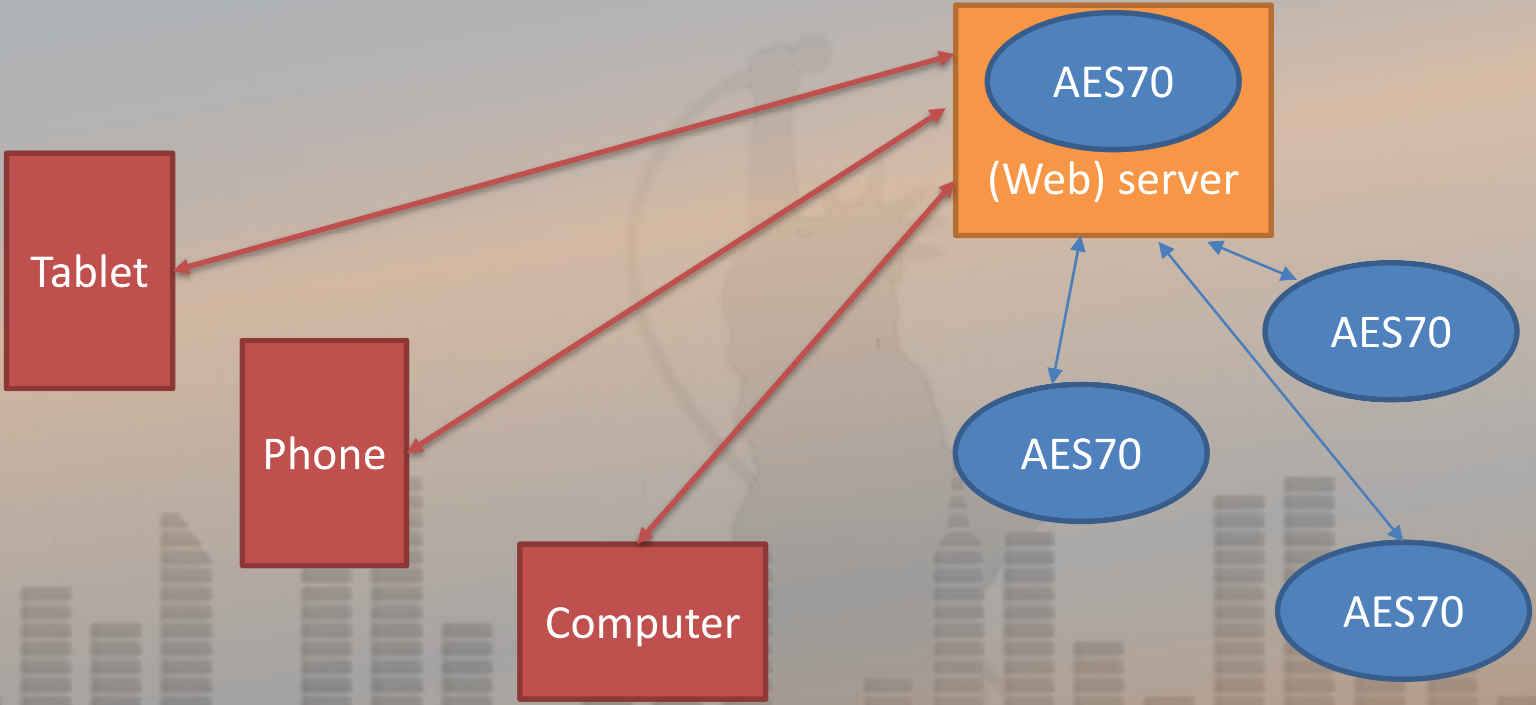
- Web-based
- Native e.g. C++ based program which run on a PC / Embedded platform / Touch screen device

emo

Possible deployment with OCA.js



Possible deployment with OCA.js



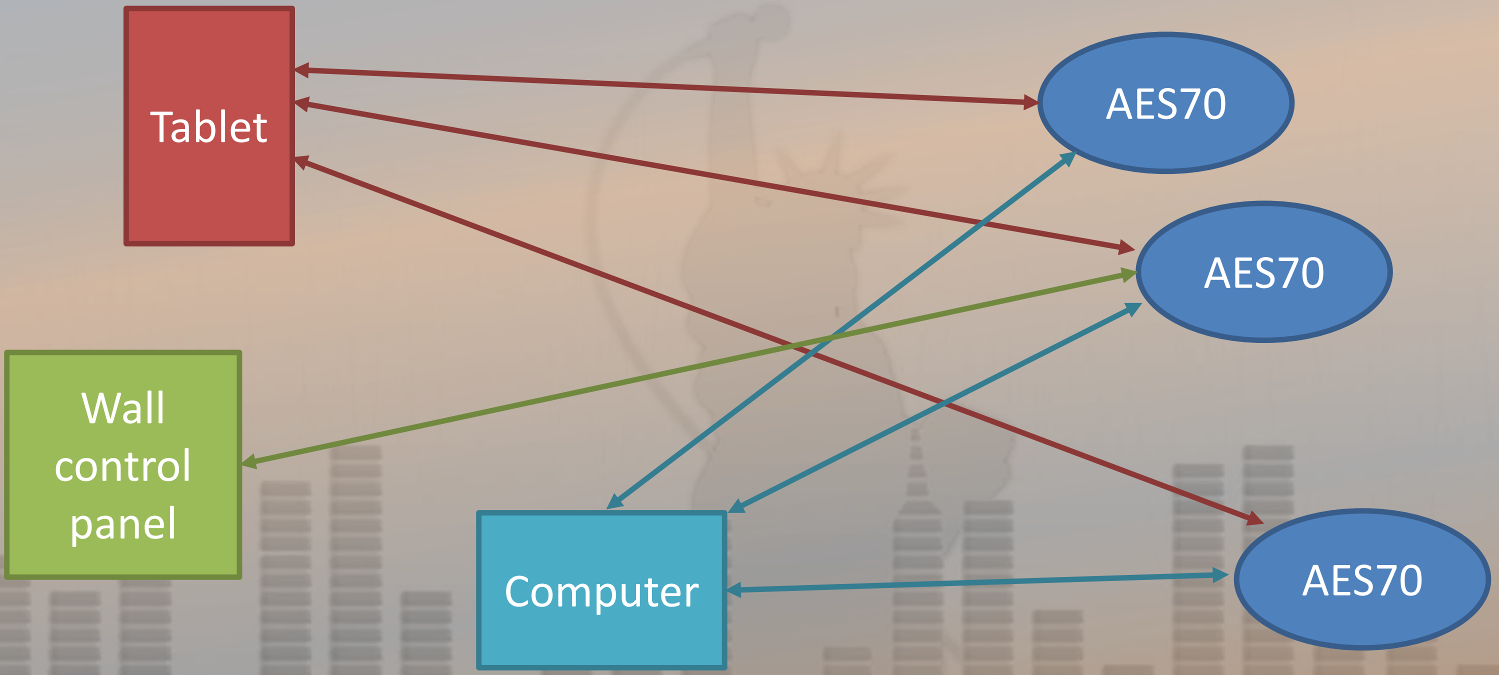
How to make an AES70 controller

- Webserver maintains AES70 OCP.1 connections to the device
- Webserver acts as proxy to forward commands to / from the devices
- Marshalling of OCA commands is performed on Tablet/Phone/PC in Javascript
- Easy to adapt UI on bases of user needs
- Javascript programming i.s.o. native programming

- Possible applications: monitor of live audio systems, configuration of small systems

- Javascript controller code is available at https://github.com/DeutscheSoft/OCA.js
- GPLv2 license

Possible deployment with native controllers



How to make an AES70 controller

- Every controller maintains own OCP.1 connection to endpoint
- Depending on support controller can use UDP/TCP
- Can support large systems with proprietary extensions

- Possible applications: proprietary controller with for example conference application, public address application, large system amplifier configuration with acoustic calculations

- No freeware native controller is available, controller implementation can be bought from a commercial supplier

How to make an AES70 controller

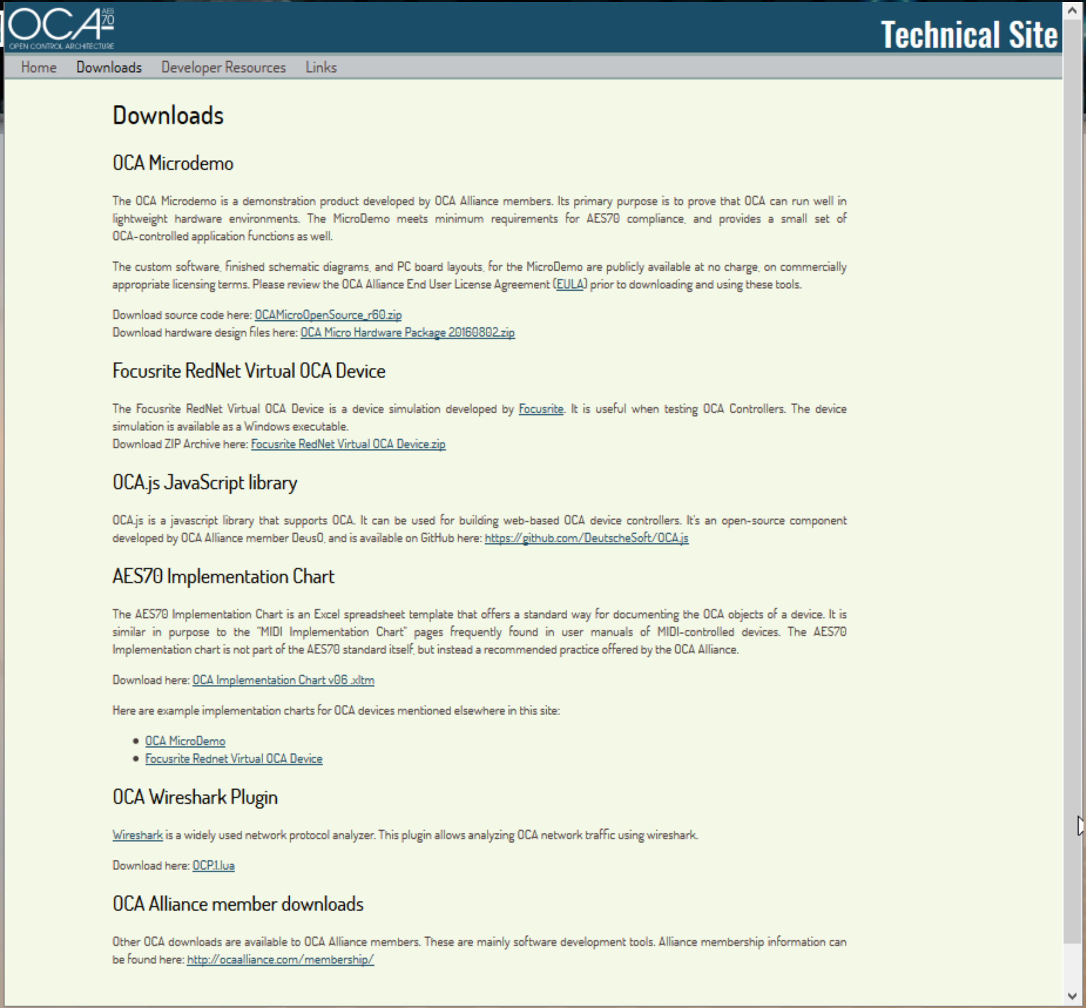
How to make an AES70 control

Where to find tools

- Members only area

- Public techsite

https://ocaalliance.github.io/



Downloads

OCA Microdemo

The OCA Microdemo is a demonstration product developed by OCA Alliance members. Its primary purpose is to prove that OCA can run well in lightweight hardware environments. The MicroDemo meets minimum requirements for AES70 compliance, and provides a small set of OCA-controlled application functions as well.

The custom software, finished schematic diagrams, and PC board layouts, for the MicroDemo are publicly available at no charge, on commercially appropriate licensing terms. Please review the OCA Alliance End User License Agreement (EULA) prior to downloading and using these tools.

Download source code here: OCAMicroOpenSource_r60.zip
Download hardware design files here: OCA Micro Hardware Package 20160802.zip

Focusrite RedNet Virtual OCA Device

The Focusrite RedNet Virtual OCA Device is a device simulation developed by Focusrite. It is useful when testing OCA Controllers. The device simulation is available as a Windows executable.
Download ZIP Archive here: Focusrite RedNet Virtual OCA Device.zip

OCA.js JavaScript library

OCA.js is a javascript library that supports OCA. It can be used for building web-based OCA device controllers. It's an open-source component developed by OCA Alliance member DeusO, and is available on GitHub here: https://github.com/DeutscheSoft/OCA.js

AES70 Implementation Chart

The AES70 Implementation Chart is an Excel spreadsheet template that offers a standard way for documenting the OCA objects of a device. It is similar in purpose to the "MIDI Implementation Chart" pages frequently found in user manuals of MIDI-controlled devices. The AES70 Implementation chart is not part of the AES70 standard itself, but instead a recommended practice offered by the OCA Alliance.

Download here: OCA Implementation Chart v06 .xltm

Here are example implementation charts for OCA devices mentioned elsewhere in this site:

- OCA MicroDemo
- Focusrite Rednet Virtual OCA Device

OCA Wireshark Plugin

Wireshark is a widely used network protocol analyzer. This plugin allows analyzing OCA network traffic using wireshark.

Download here: OCP1.lua

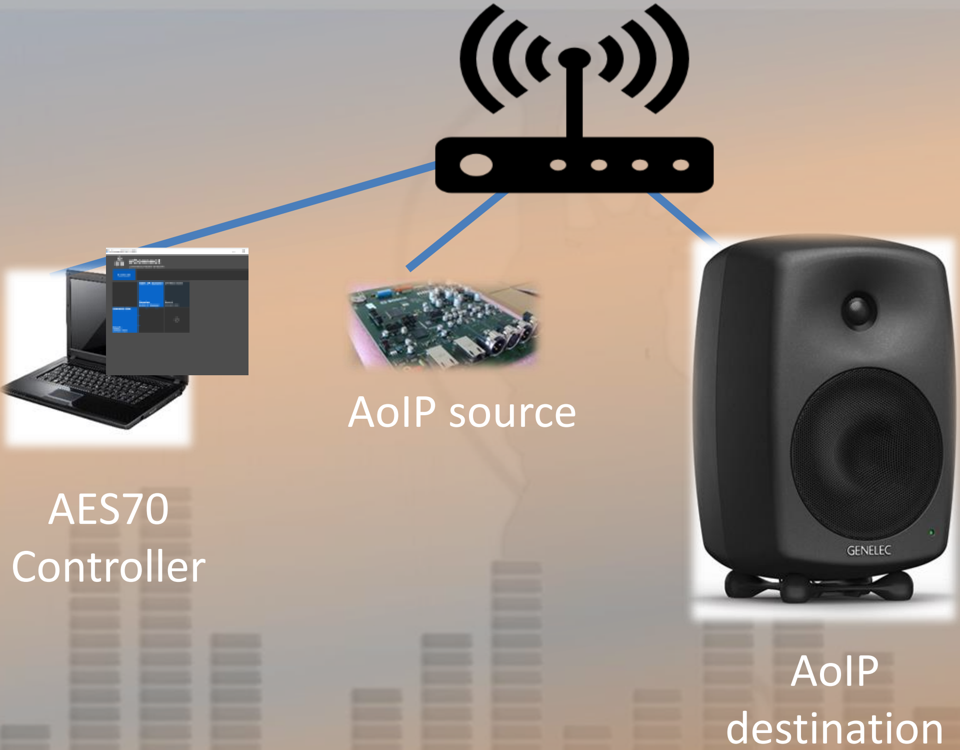
OCA Alliance member downloads

Other OCA downloads are available to OCA Alliance members. These are mainly software development tools. Alliance membership information can be found here: http://ocaalliance.com/membership/

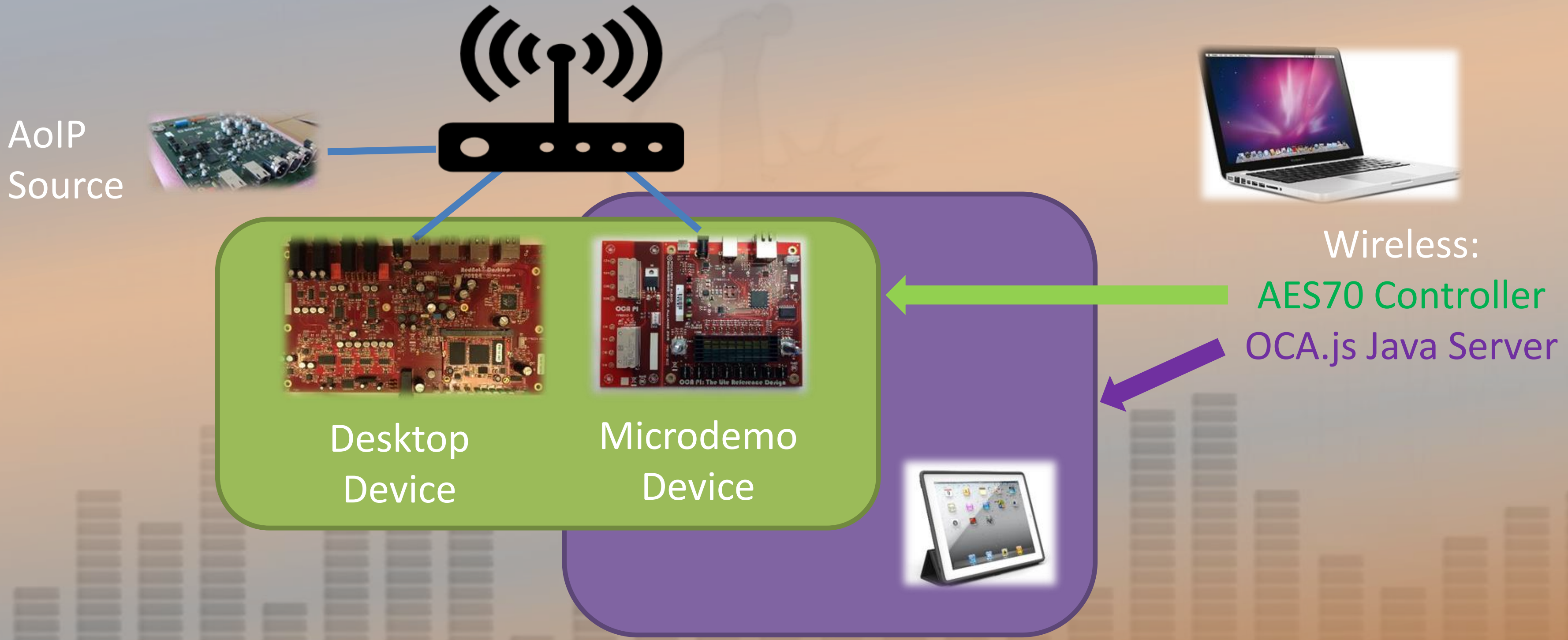
Membership

Demo setups

Making an AES70 Device: *Connection management Setup*



Making an AES70 Device: *Controller Setup*


AoIP
Source
Desktop
Device
Microdemo
Device
Wireless:
AES70 Controller
OCA.js Java Server