Developing your first application using FI-WARE

Fermín Galán Márquez (fermin@tid.es), Miguel Jimenez (mjimenez@fi.upm.es), Carlos Ralli (ralli@tid.es), Juanjo Hierro (jhierro@tid.es)
Telefónica I+D, Universidad Politécnica de Madrid

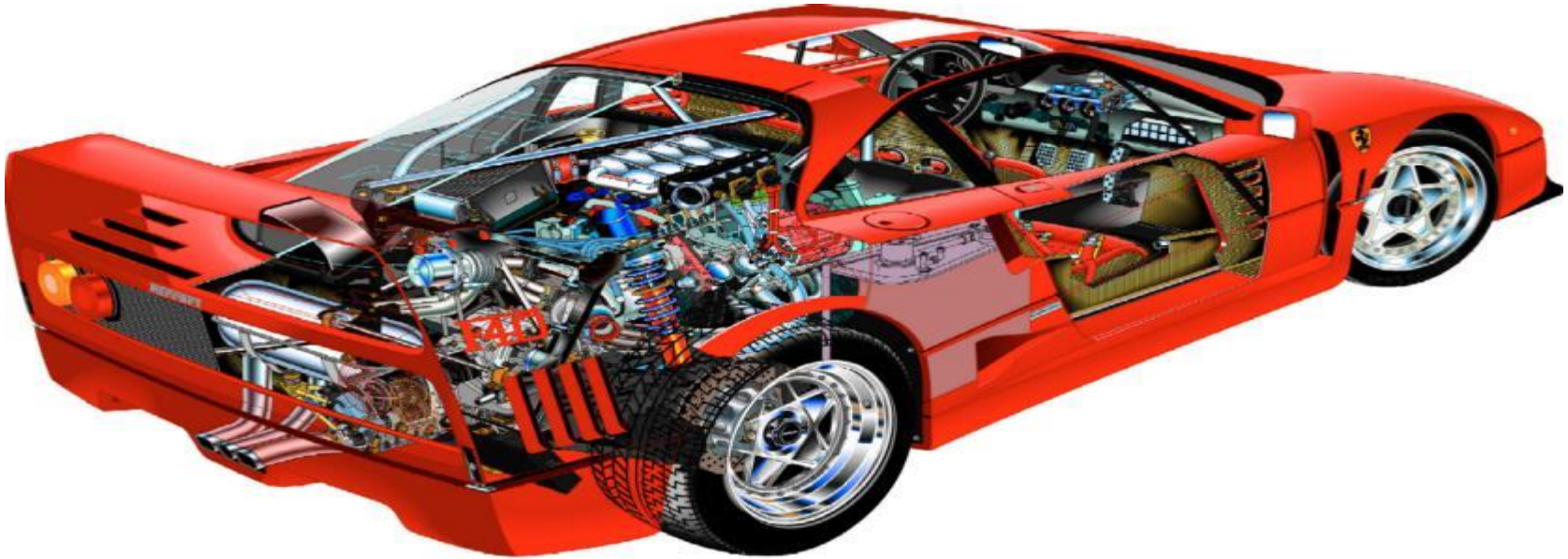
FUTURE INTERNET PPP

/THE FUTURE. NOW.

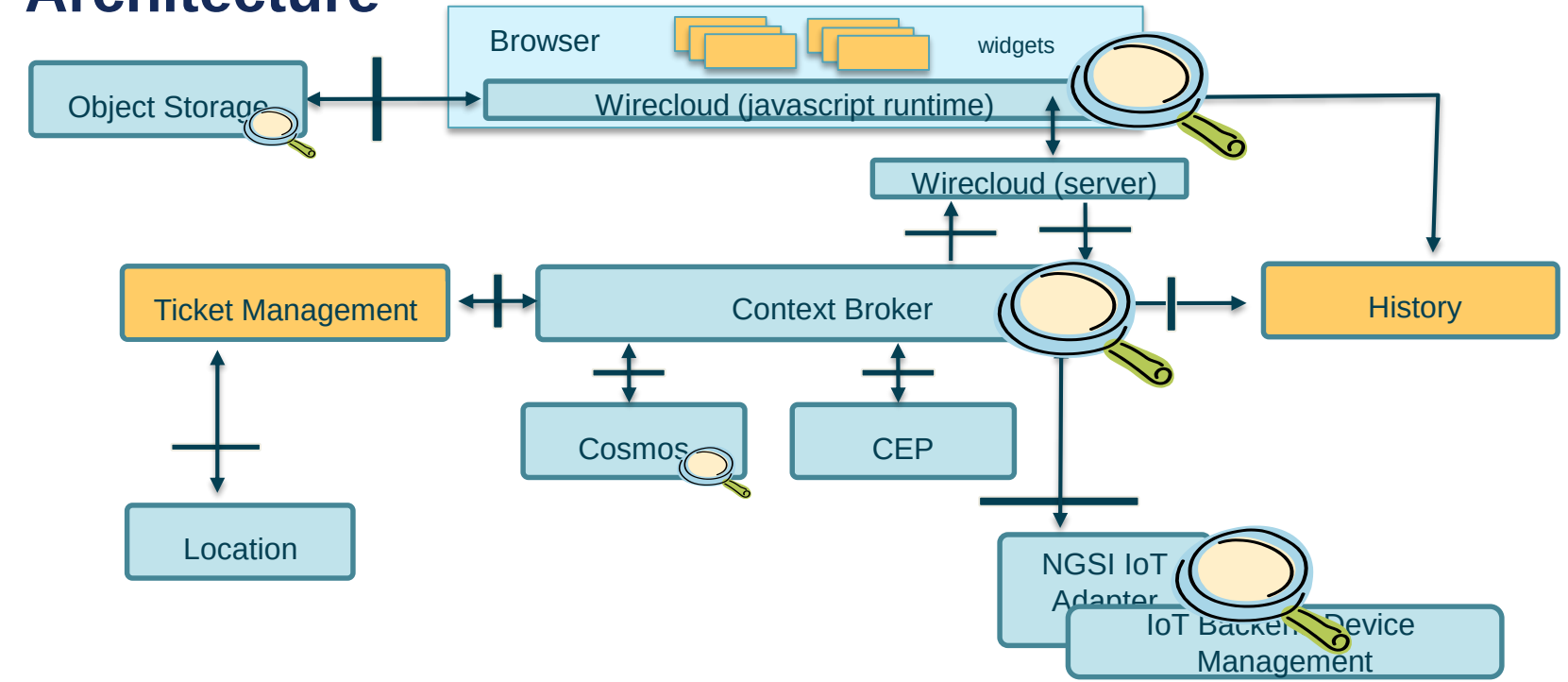
FI-WARE
FI-Lab

Open APIs for Open Minds

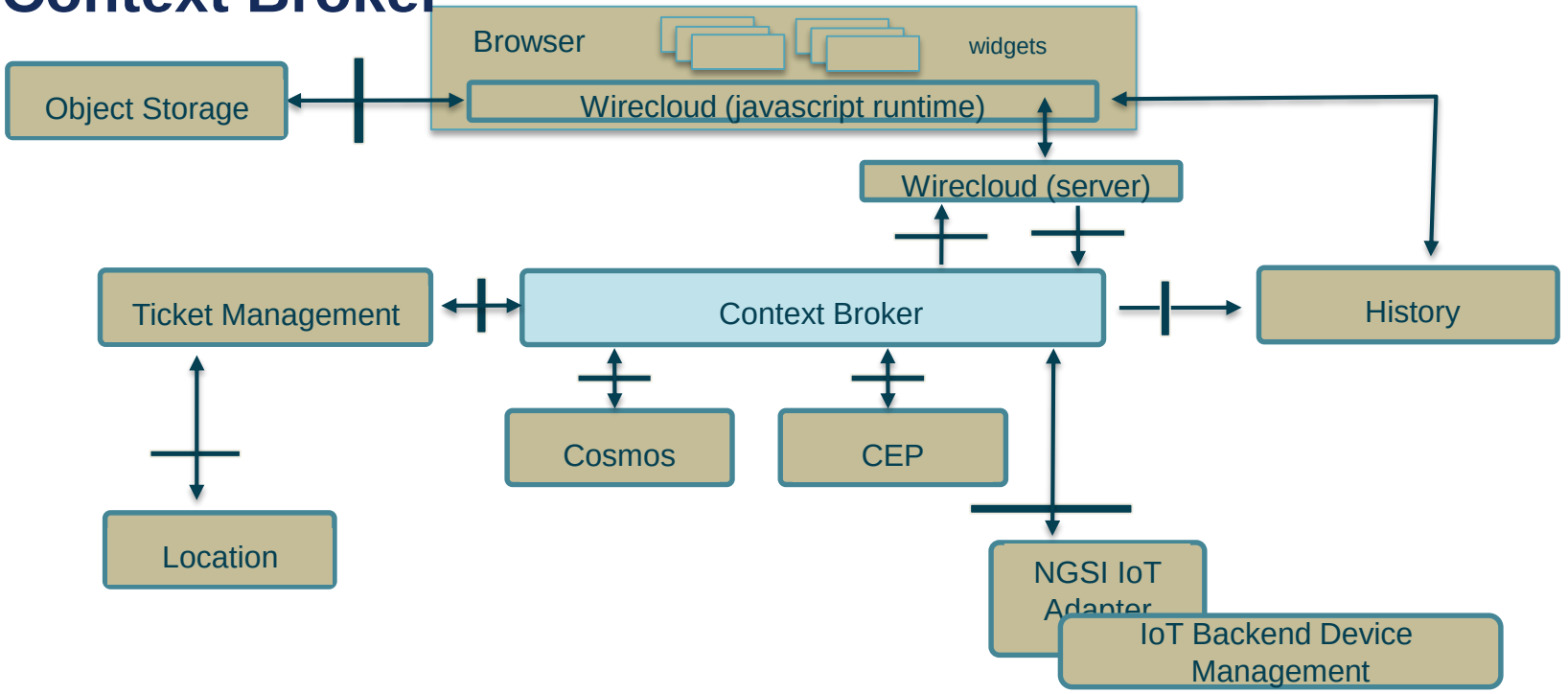
Let’s go into detail…



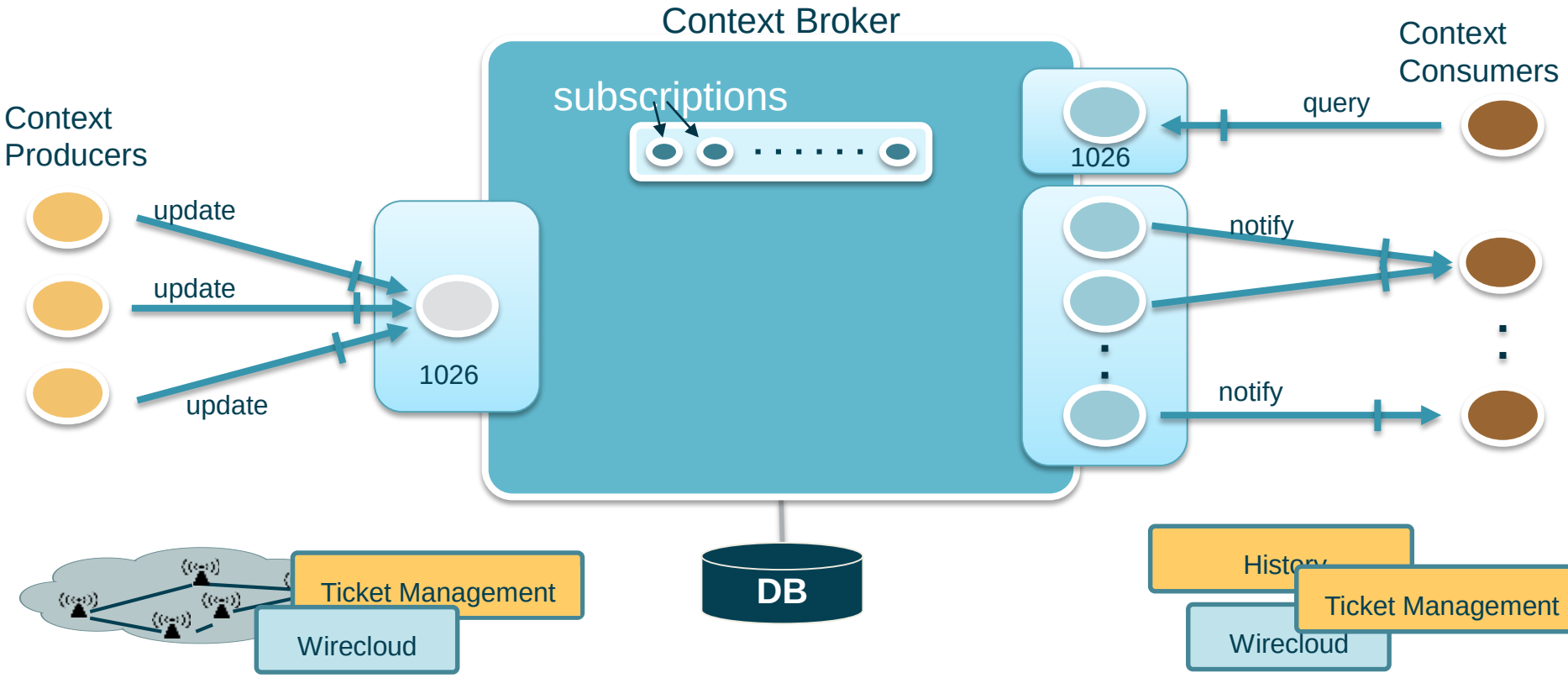
Architecture



Context Broker

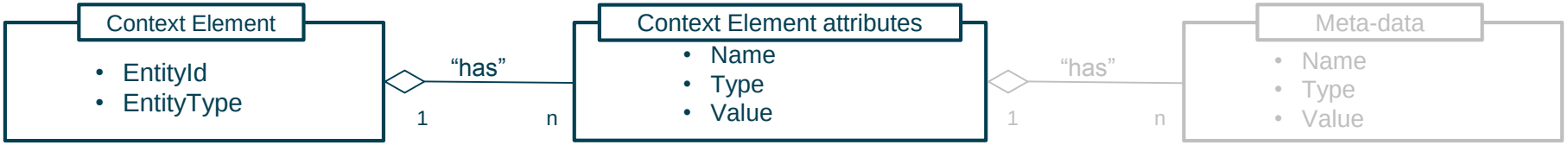


Taking a closer look

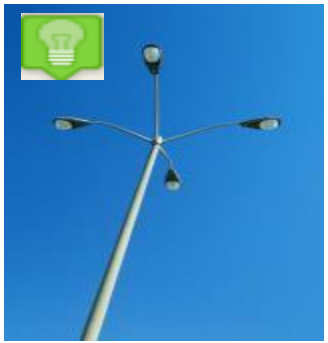


The NGSI information model

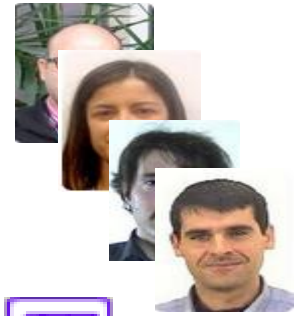
(We are not fully supporting metadata right now)



Electrical Regulator



Node Lamps



Technicians

Vans

severity

batteryCharge

electricPotential

presence

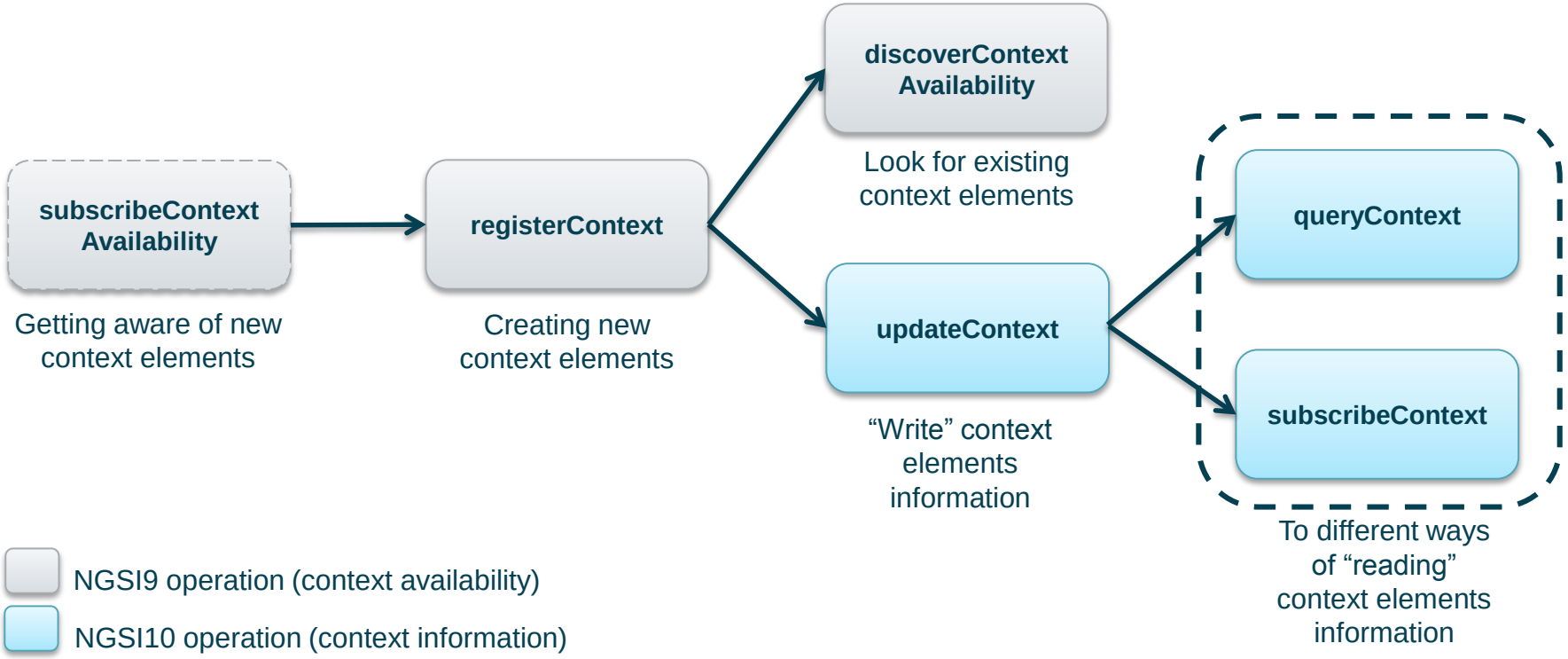
location

TimeInstant

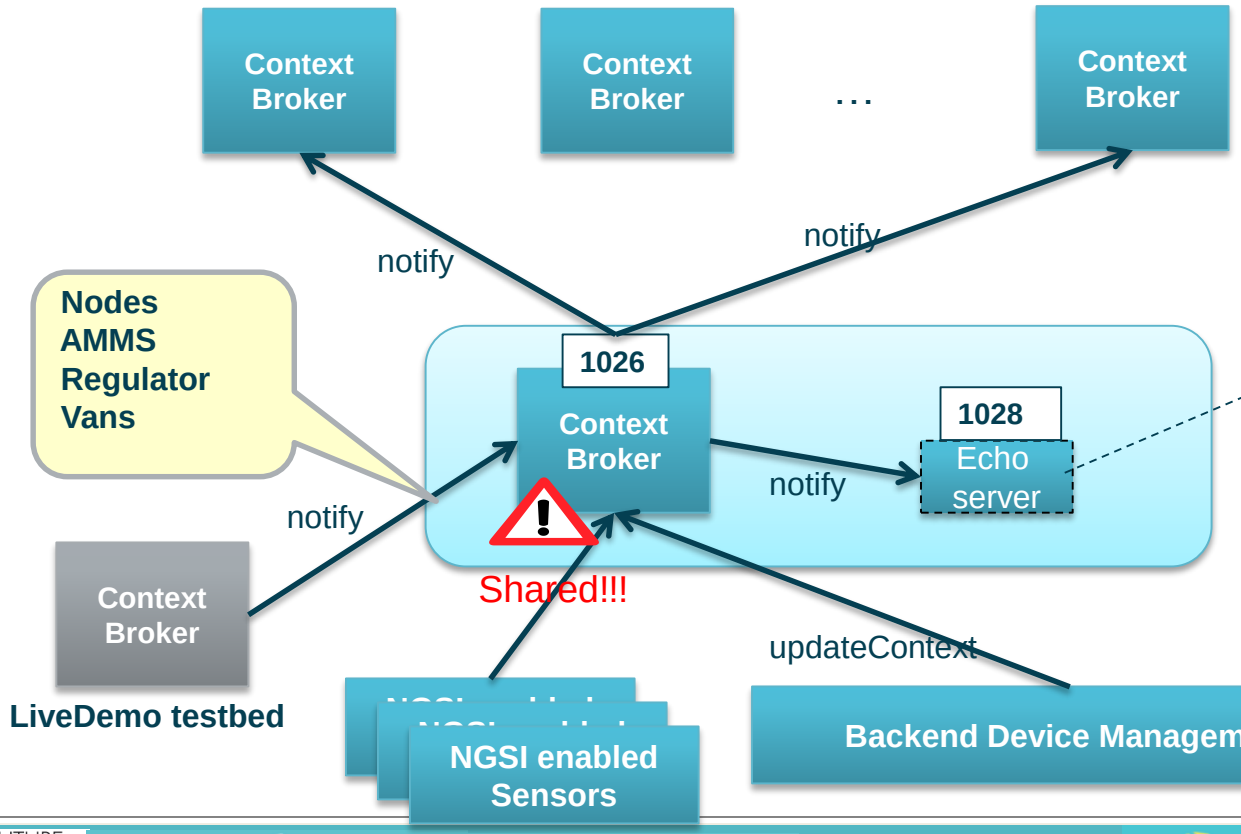
illuminance

…and many more (look to app code)

Typical Orion broker utilization steps



Context platform available at Campus Party



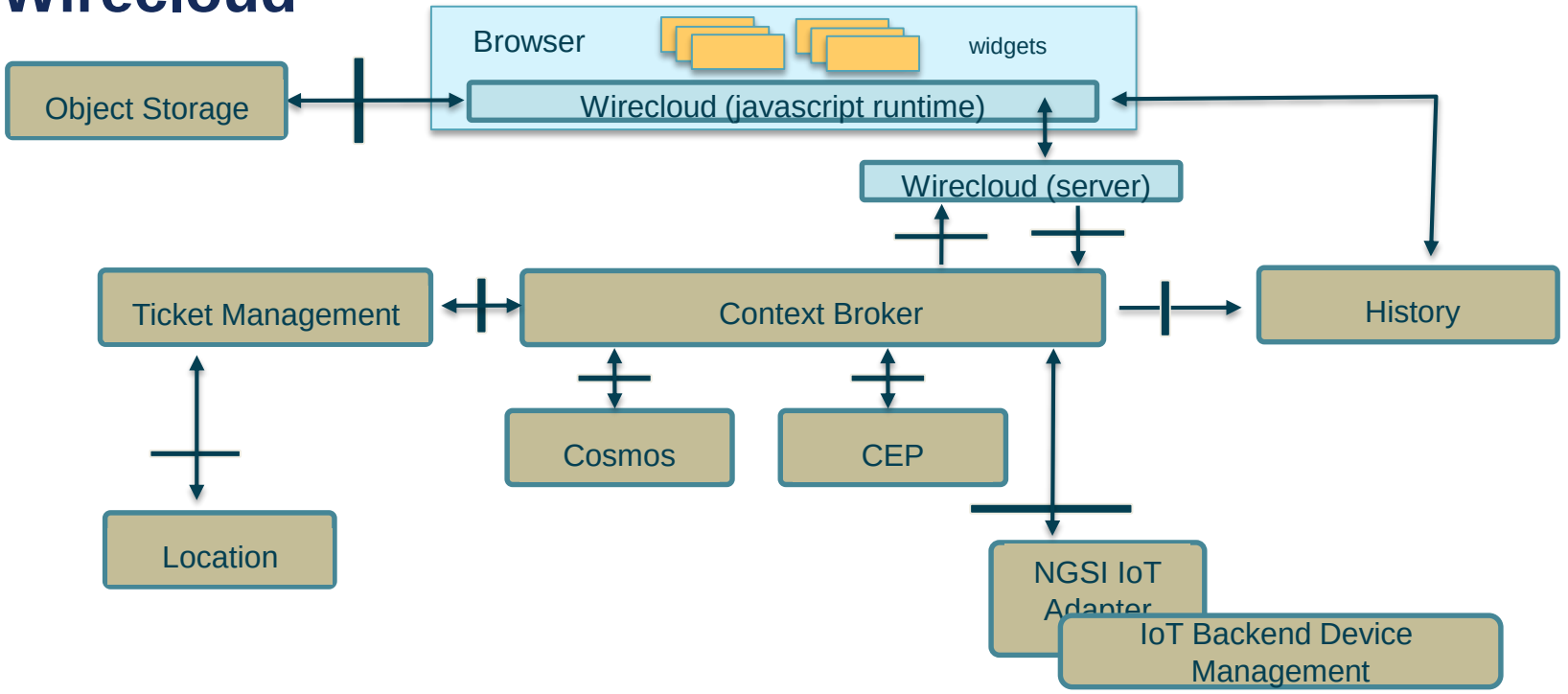
Dedicated instances at http://yourvm:1026

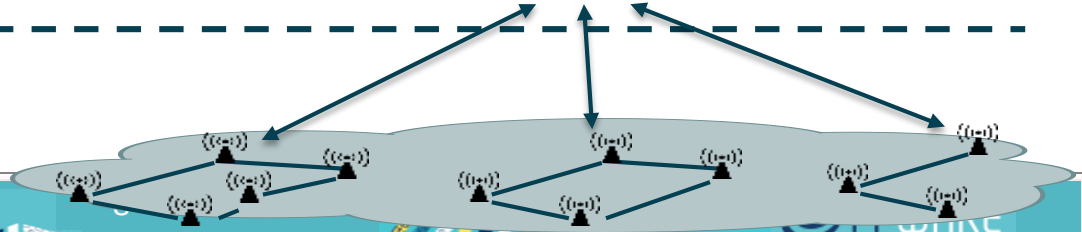
"FI-WARE Cloud: bringing OpenStack to the next level" workshop (Wed 10-12am)

(Only for demonstration purposes during the workshop)

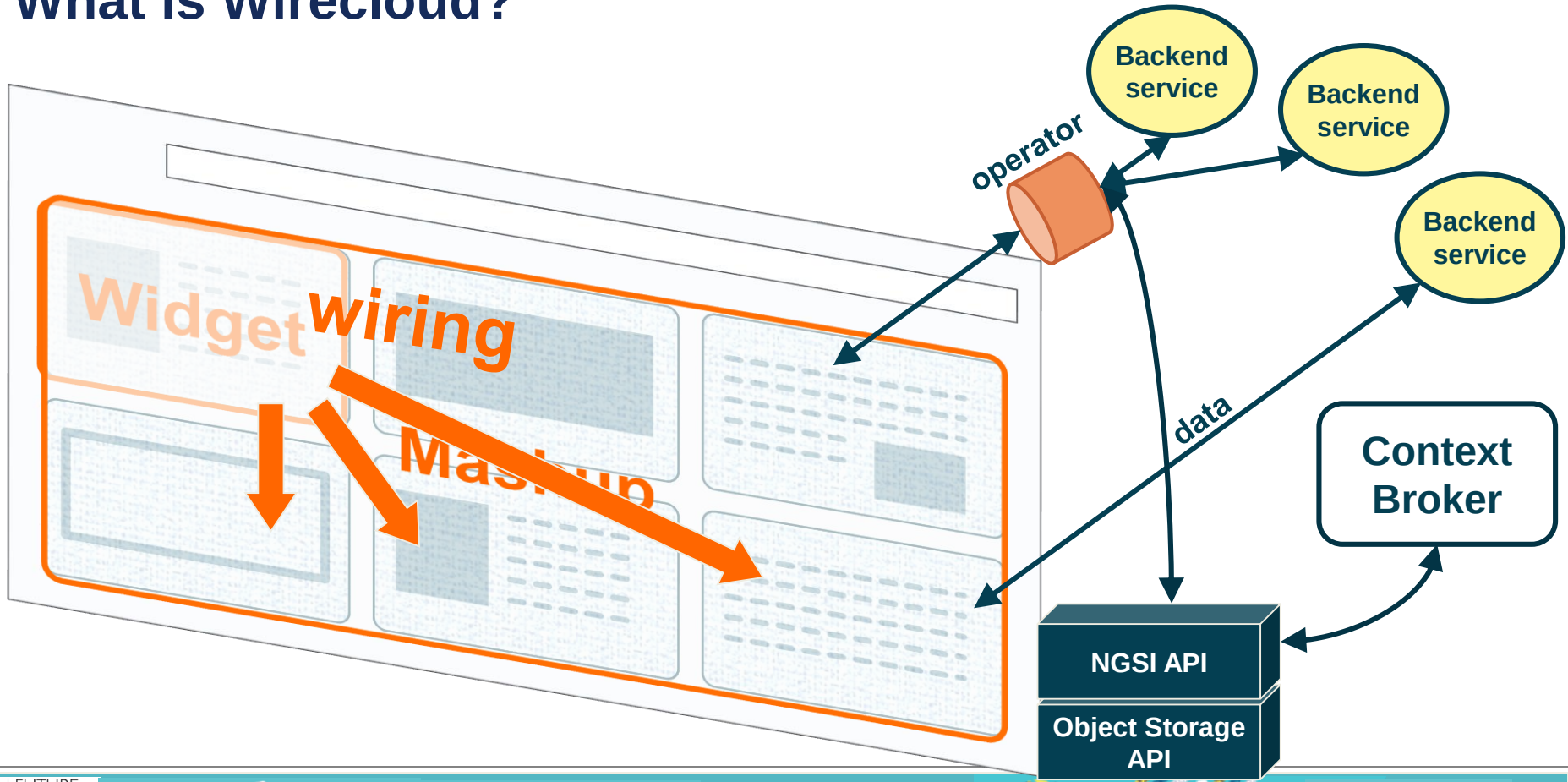
Global instance at orion.lab.fi-ware.eu (at FI-LAB cloud)

Wirecloud

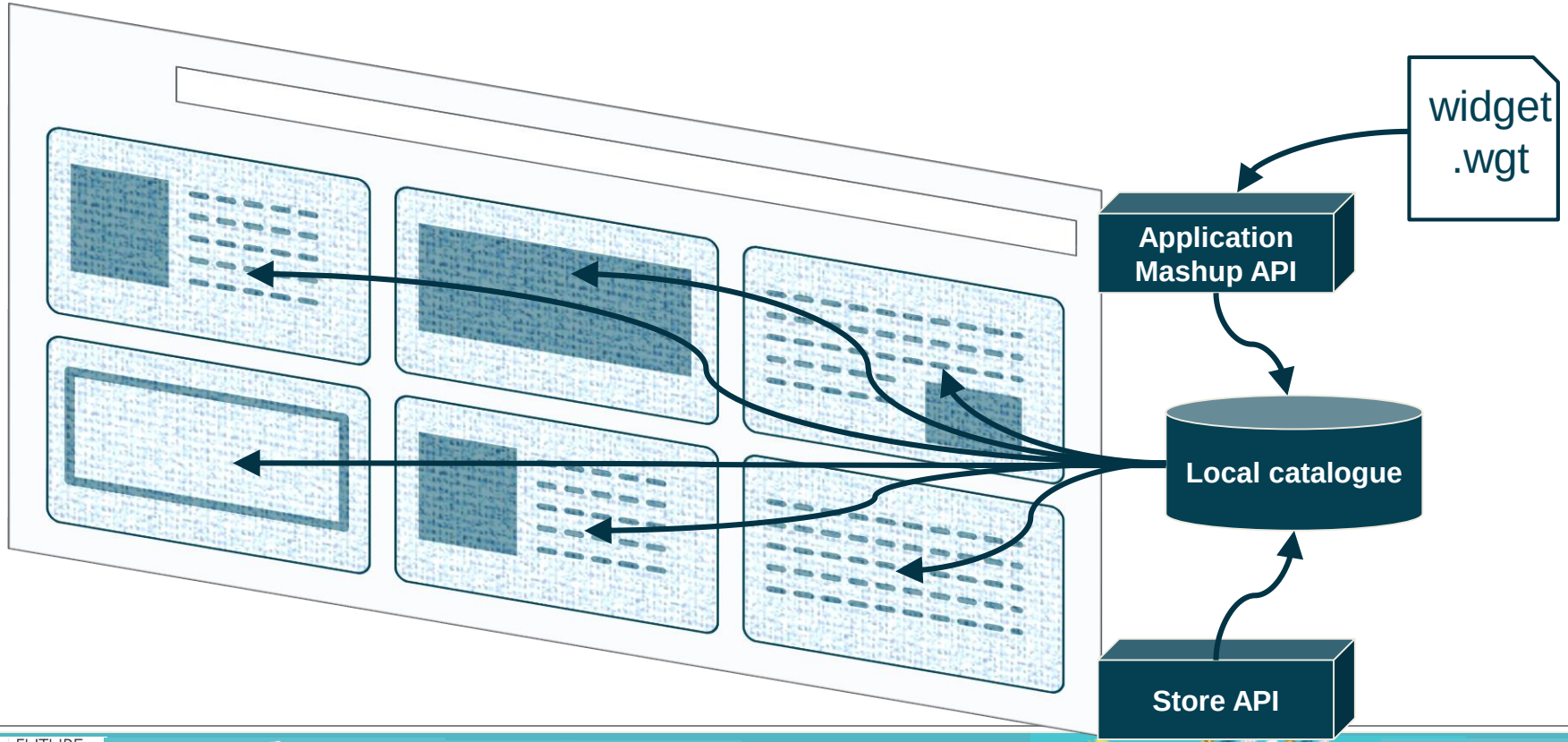

Browser
widgets
Object Storage
Wirecloud (javascript runtime)
Wirecloud (server)
Ticket Management
Context Broker
History
Cosmos
CEP
Location
NGSI IoT Adapter
IoT Backend Device Management



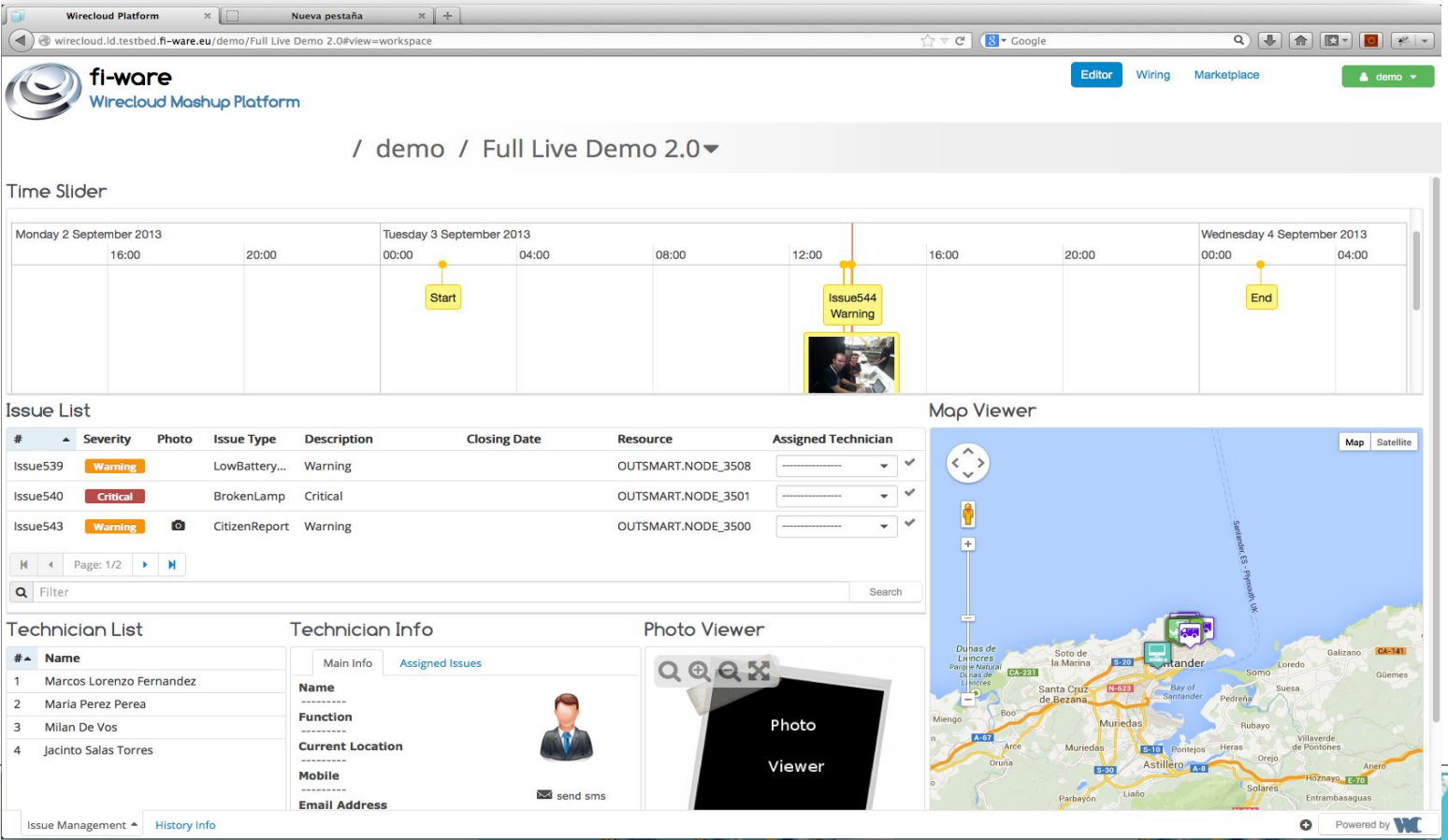
What is Wirecloud?



Catalogue of widgets and mashups



What can I do with Wirecloud?



To create my own application mashup…

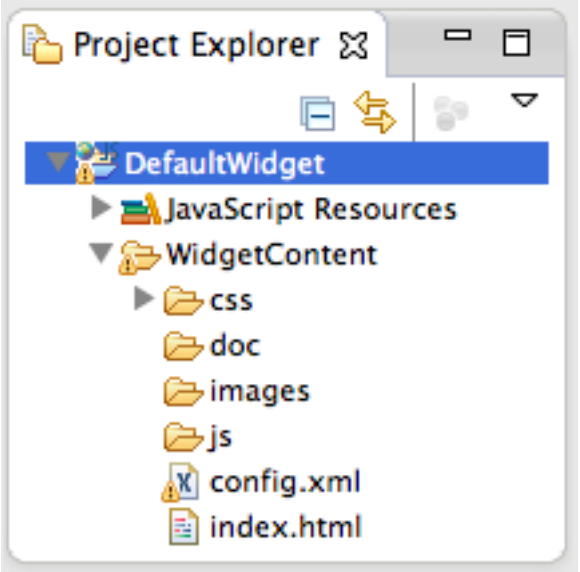
- Widgets and operators:
  - Widgets can be developed with any Web technology (HTML, Flash, SVG…) that supports JavaScript.
  - Operators are coded in JavaScript
- Both widgets and operators can be easily adapted from existing ones, or created from scratch
- Open Widget API
  - JavaScript API
  - Core Widgets Functionality:
    - gain access to external resources (cross-domain problem)
    - wiring
    - preferences
    - …

How to develop a widget?

- Setting up the development environment
  - Eclipse, VIM, etc...
  - Creating the directory structure (WGT)
- Creating the widget:
  - Create the template of the widget (config.xml)
  - Widget view: an HTML file + CSS
  - Widget app logic: JavaScript (using Widget API)
- Testing your widget
  - Uploading the widget to Local Catalogue
    - The WGT file is a ZIP file renamed to .wgt
  - Add widget to a mashup
    - Wiring, preferences, layout…

Widget structure

- **config.xml**
  - definition of the widget (based on WDL)
- **index.html**
  - main view file
  - references JS and CSS resources
- **/js**
  - directory for JavaScript files
- **/css**
  - directory for CSS files
- **/images**
  - directory for images
- **/doc**
  - documentation



Config.xml

- The mandatory config.xml file contains the WDL XML template needed to tell Wirecloud what the widget looks like. This includes:
  - Widget metadata (non-functional properties)
  - User preferences
  - Widget inputs and outputs (wiring)
  - Link to the actual widget source code
  - Default rendering information

Config.xml example (I)

```
<?xml version="1.0" encoding="UTF-8"?>

<Template xmlns="http://wirecloud.conwet.fi.upm.es/ns/template#">
    <Catalog.ResourceDescription>
        <Vendor>Company distributing the widget</Vendor>
        <Name>Widget name</Name>
        <Version>0.0.1</Version>
        <DisplayName>Widget Example</DisplayName>
        <Author>mjimenez</Author>
        <Mail>mjimenez@fi.upm.es</Mail>
        <Description>Full widget description to be shown in catalogue</Description>
        <ImageURI>Absolute or relative path to display image</ImageURI>
        <iPhoneImageURI>Path to imate to display on mobile devices</iPhoneImageURI>
        <WikiURI>Path to widget doc</WikiURI>
    </Catalog.ResourceDescription>
```

Config.xml example (II)

```
  <Platform.Preferences>
      <Preference name=”pref_name" type="text" description="…" />
  </Platform.Preferences>
  <Platform.Wiring>
      <OutputEndpoint  name="identifier"
              type="text" label="Label to show on wiring"
              description="description"
              friendcode="same as compatible input endpoint friendcode"/>
      <InputEndpoint  name="identifier"
              type="text" label="Label to show on wiring"
              description="description"
              friendcode="same as compatible output endpoint friendcode"/>
  </Platform.Wiring>
  <Platform.Link>
      <XHTML href="index.html"/>
  </Platform.Link>
  <Platform.Rendering width="6" height="24"/>
</Template>
```

index.html example

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <script type="text/javascript" src="js/main.js"></script>
    <link rel="stylesheet" type="text/css" href="css/style.css" />
  </head>
  <body>
    <!- Create here or create by JavaScript ->
  </body>
</html>
```

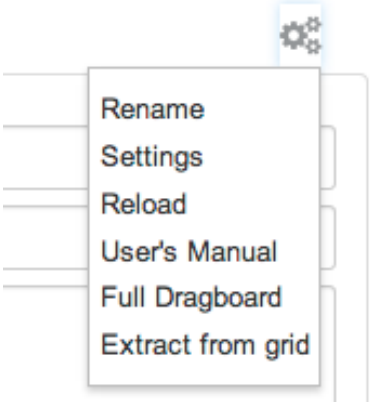
Using the Open Widget API

- Accessing widget preferences

```
MashupPlatform.prefs.get("google_key")
MashupPlatform.prefs.set("color", "red")
MashupPlatform.prefs.registerCallback(callbackFunction)
```

- Getting user login name

```
MashupPlatform.context.get("username");
```



Wiring endpoints

- Declaration on config.xml

```
<Platform.Wiring>
   <InputEndpoint name="my_input_name" type="text" label="my_label"
      description="my_description" friendcode="some_code" />
   <OutputEndpoint name="my_output_name" type="text" label="label"
      description="description" friendcode="url"/>
</Platform.Wiring>
```

- Input endpoints

```
MashupPlatform.wiring.registerCallback("my_input_name", inputListener);
```

- Output endpoints

```
MashupPlatform.wiring.pushEvent("my_output_name", data);
```

Making a cross-domain HTTP request

- Perform the invocation and register asynchronous call-backs:

```
MashupPlatform.http.makeRequest(url, {
  method: 'GET',
  onSuccess: function (transport) {
    var response;
    response = JSON.parse(transport.responseText);
    ...
  },
  onFailure: function (transport) {
    onError();
  }
});
```

Connecting a widget to NGSI

- Register NGSI usage on config.xml

```
<Requirements> <Feature name="NGSI"/> </Requirements>
```

- Send a notification through NGSI

```
var connection = new NGSI.Connection('http://orion.lab.fi-ware.eu');
connection.updateAttributes([{
    entity: {
        id: 'iss8',
        type: 'Issue'
    },
    attributes:[{
        name: 'technician',
        contextValue: 'tech1'
    }]
}], {
    onSuccess: function () { //… },
    onFailure: function () { //… }
});
```

Connecting a widget to NGSI

- Subscribe for notifications on certain entities

```
var entityIdList = [
    {type: 'Van', id: '.*', isPattern: true},
    {type: 'Technician', id: '.*', isPattern: true}
];
var attributeList = null; var duration = 'PT3H'; var throttling = null;
var notifyConditions = [{
    type: 'ONCHANGE',  condValues: ['name', 'current_position']
}];
var options = {
    flat: true,
    onNotify: process_entities,
    onSuccess: function (data) {
        subscriptionId = data.subscriptionId;
        refresh_interval = setInterval(refreshNGSISubscript, 1000*60*60*2);//2 hours
    }
};
connection.createSubscription(entityIdList, attributeList, duration, throttling,
        notifyConditions, options);
```

Using Object Storage from a widget

- Register Object Storage usage on config.xml

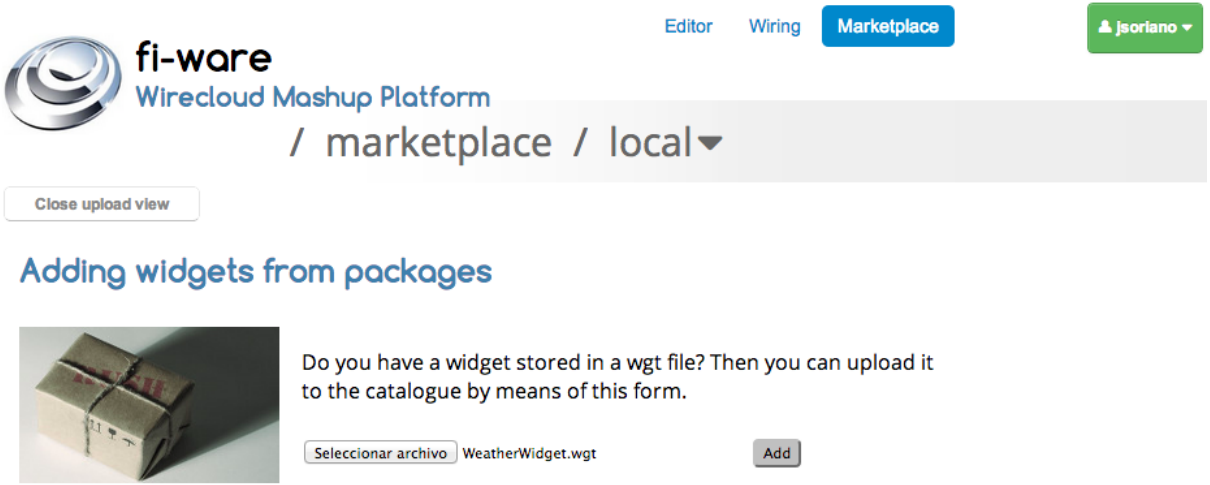
```
<Requirements> <Feature name="ObjectStorage"/> </Requirements>
```

- Upload a file

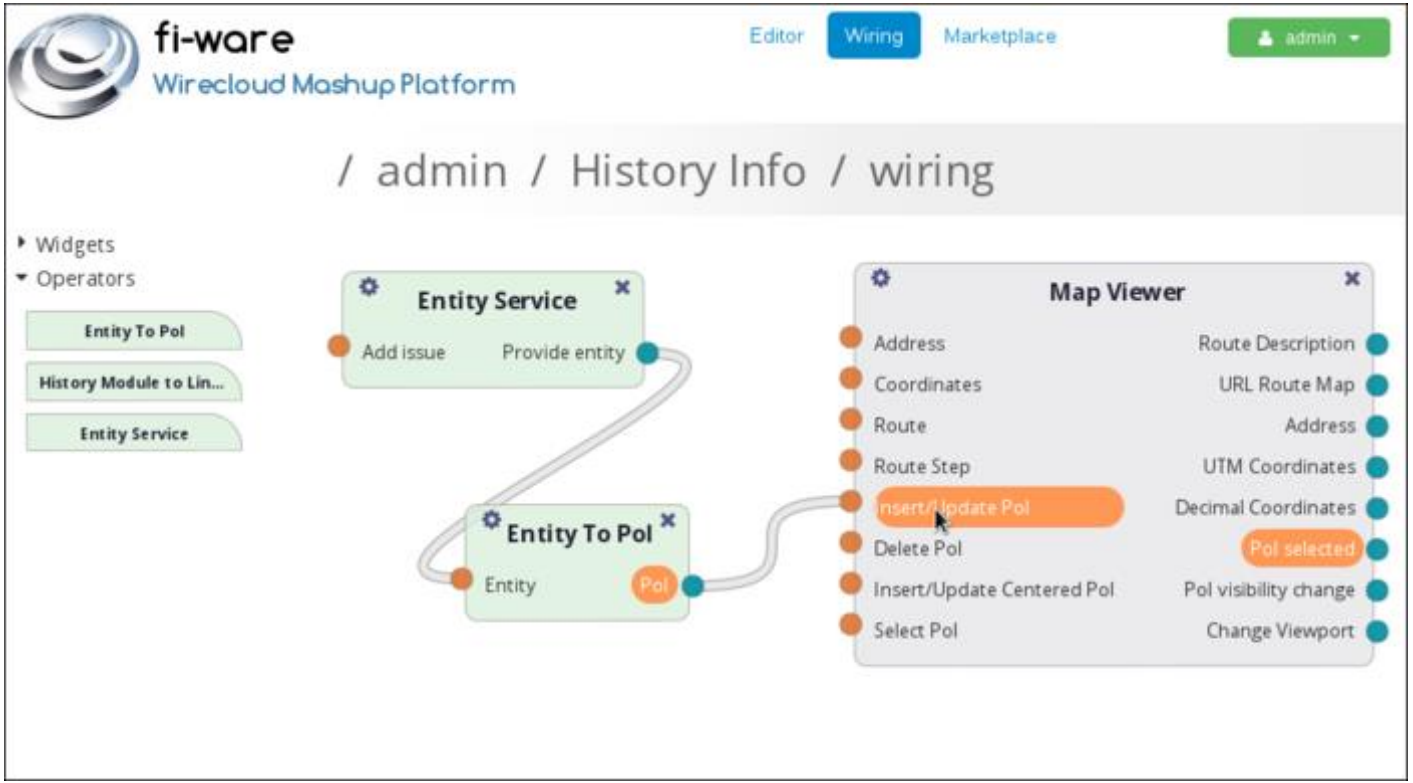
```
var object_storage = new ObjectStorageAPI('<Object Storage instance url>');
var fileParts = ["<a id=\"a\"><b id=\"b\">hey!<\/b><\/a>"];
var myBlob = new Blob(fileParts, { "type" : "text\/xml" });

object_storage.uploadFile('folder_name', myBlob, token, {
    file_name: 'myFile.xml',
    onSuccess: function () {
        alert('File uploaded successfully');
    },
    onFailure: function () {
        alert('Something went wrong while uploading the file');
    }
});
```

Uploading your widget

- Compress (zip) the widget contents
  - Rename it as a .wgt file
- Upload the WGT file to Local Catalogue through WireCloud web interface



Wiring your widget



Resources

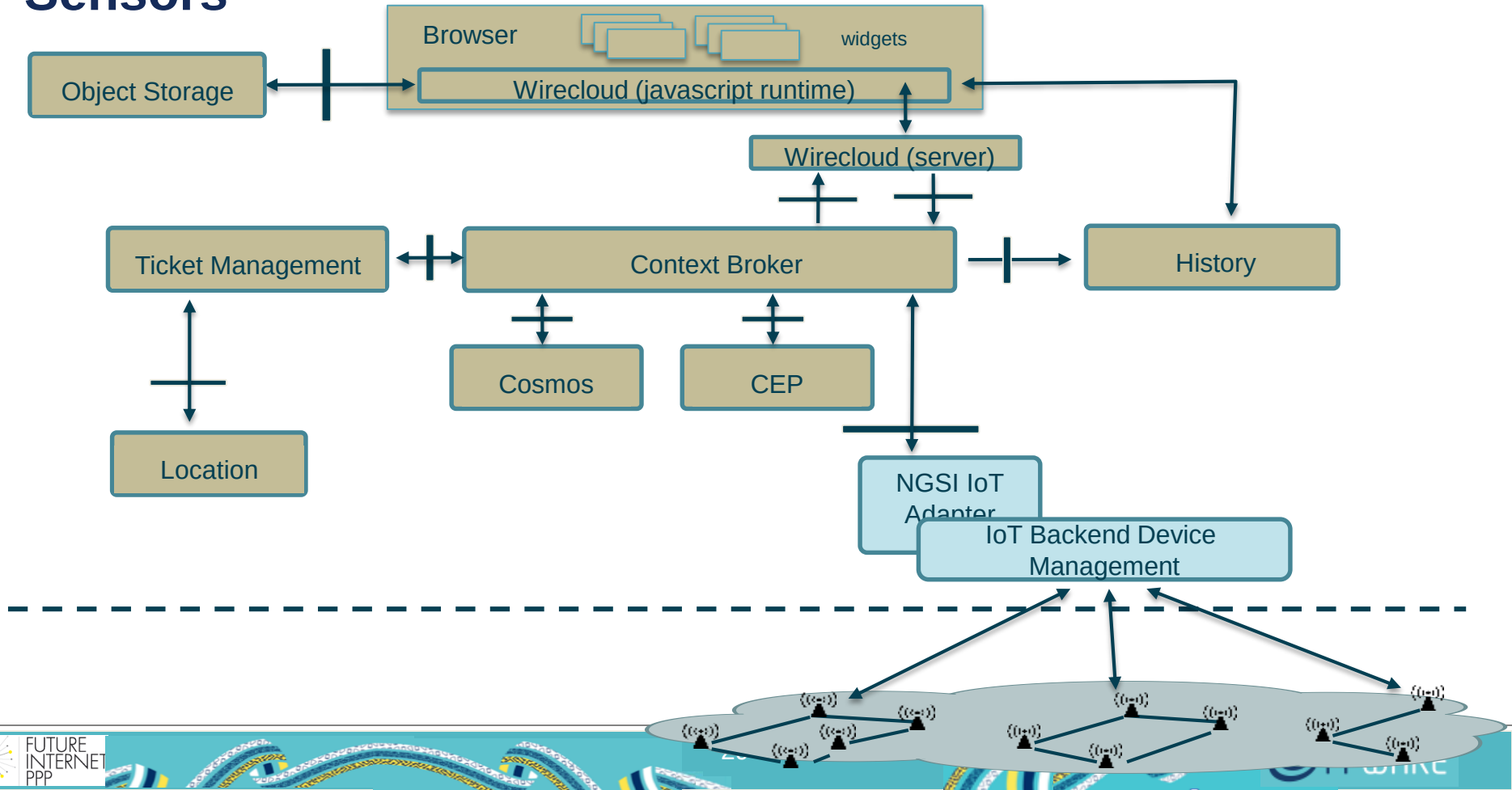
https://mashup.lab.fi-ware.eu

- You can always find the most updated documentation of Wirecloud in:
  - http://conwet.fi.upm.es/wirecloud
- User and Programmer Guide:
  - https://forge.fi-ware.eu/plugins/mediawiki/wiki/fiware/index.php/Composition_Editor_-_Wirecloud_Mashup_Platform_-_User_and_Programmer_Guide
- Installation and Administration Guide:
  - https://forge.fi-ware.eu/plugins/mediawiki/wiki/fiware/index.php/Composition_Editor_-_Wirecloud_Mashup_Platform_-_Installation_and_Administration_Guide
- Download from Github:
  - Wirecloud source code: https://github.com/Wirecloud/wirecloud
  - Widget's source code of FIWARE's Live Demo mashable application component): https://github.com/wirecloud-fiware/live-demo-macs
- Stuff for this workshop
  - http://tinyurl.com/fiware-dropbox

Online videos

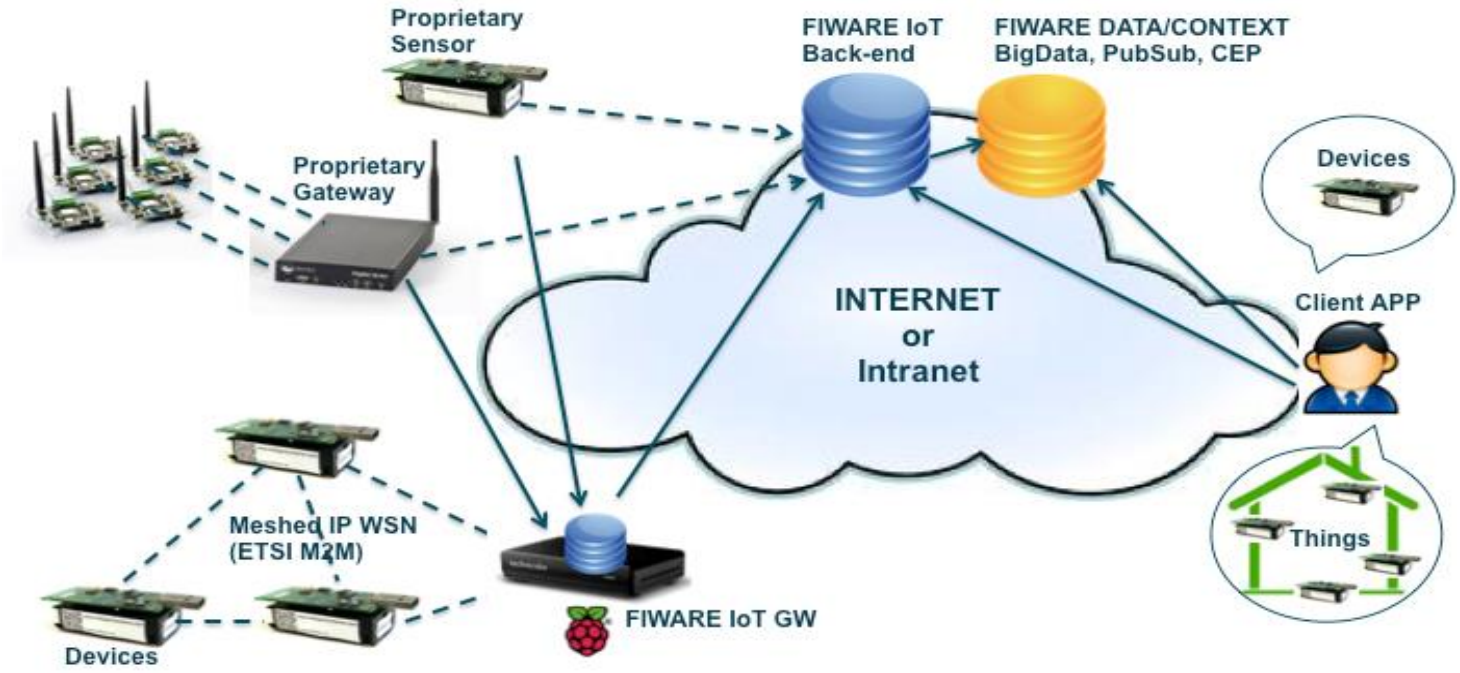
- FIWARE Live Demo: http://www.youtube.com/watch?v=Wh_zPsLUg-8
- ENVIROFI Demo: https://www.youtube.com/watch?v=yEXlLQYq7s4
- Other videos:
  - http://www.youtube.com/embed/d7_EG42AHJw - Building a mashup from Geowidgets linked to geospatial services.
  - http://www.youtube.com/embed/urDGWSnrbtE - Using operators in a mashup to allow widgets to send tweets
  - http://www.youtube.com/embed/kW0sXMxgMLI - An example of integration with the marketplace and the catalogue GEs

Sensors

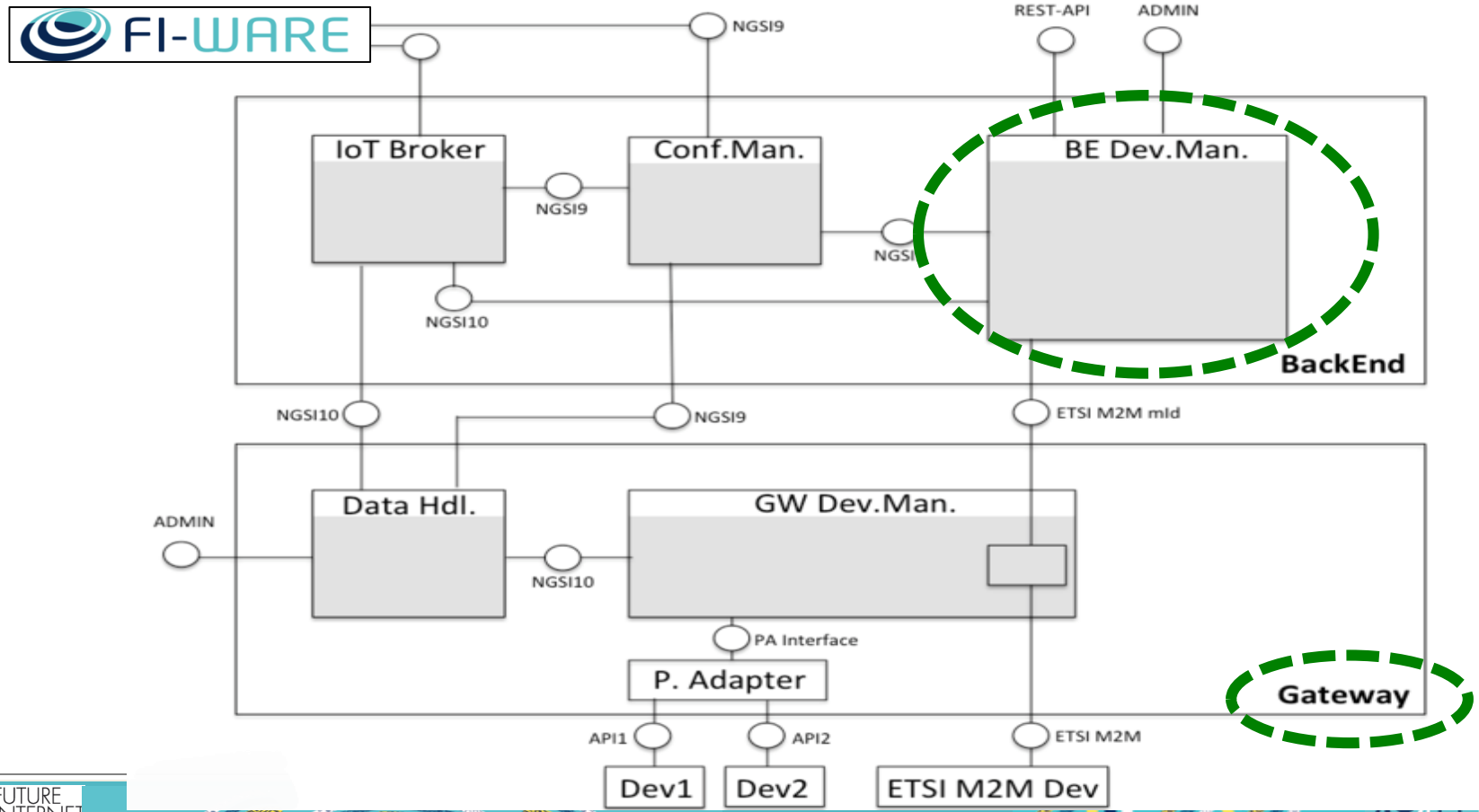

Browser
widgets
Object Storage
Wirecloud (javascript runtime)
Wirecloud (server)
Ticket Management
Context Broker
History
Cosmos
CEP
Location
NGSI IoT Adapter
IoT Backend Device Management

FI-WARE IoT Extended Platform Vision

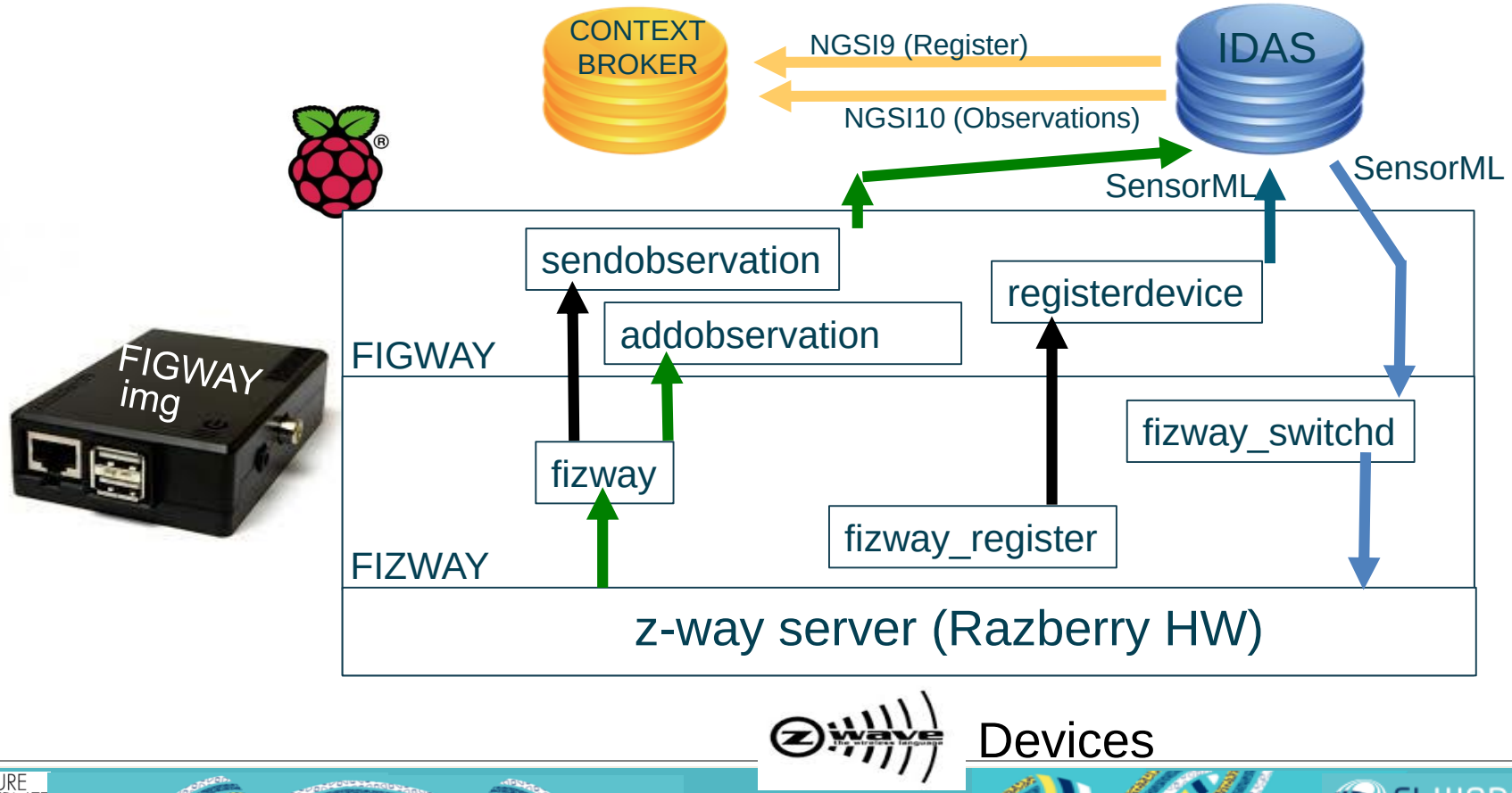
Plug IoT world to FI-WARE via NGSI notifications.



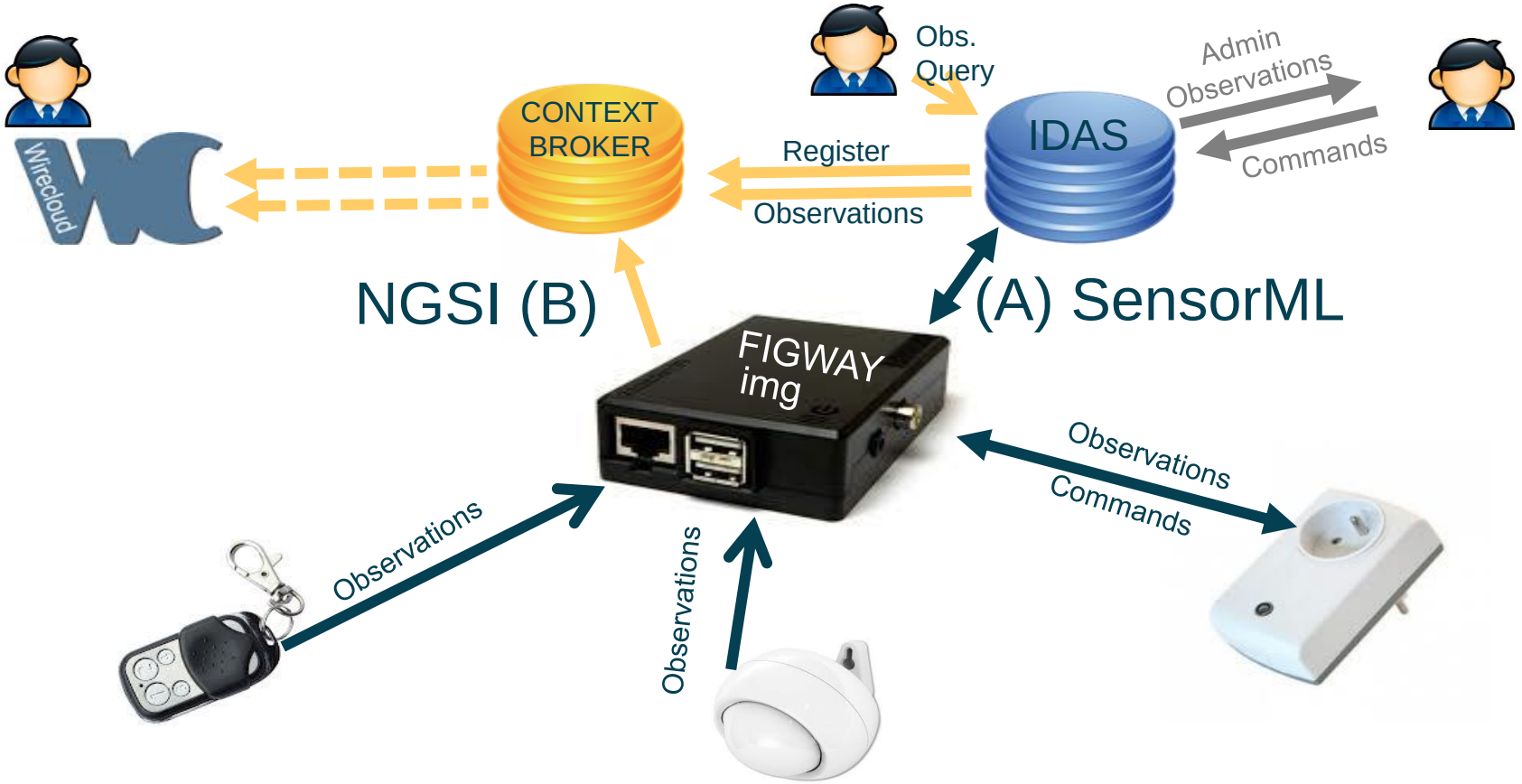
FI-WARE IoT Functional Architecture (Target)



FI-WARE IoT Challenge Architecture



Connecting a basic z-wave pack



Download the PI image with FIGWAY software at: http://tinyurl.com/figway-img

HOW-TO Get Started - Z-wave basic pack

0) You need: a RaspberryPI + Razbian card plugged in its GPIO

1) Include your z-wave devices into your z-wave network

Complete your sensor details in our “IoT Challenge HUB doc”

https://docs.google.com/spreadsheet/ccc?key=0ArcymbqnpOfkdGNYUkpaTF9qRVhkOTFIYW14SkxaZ1E#gid=0

2) Edit "fizway_resgister” & "fizway" scripts

Update the Device_ID number of your sensors.

3) Edit the file SensorML/Register_SWITCH

Update the line containing the callback URL:

"<swe:value>http://1.0.0.1:9999</swe:value>”

-> Instead of 1.0.0.1 put the RaspberryPI public IP address

-> Instead of 9999 set the port the switch daemon will be listening (normally, 7777)

4) Check & Edit -if necessary- the figway "Config" file.

Normally, modifications aren't needed so you can skip (4).

A Config.example file is provided and comments should make this task really easy.

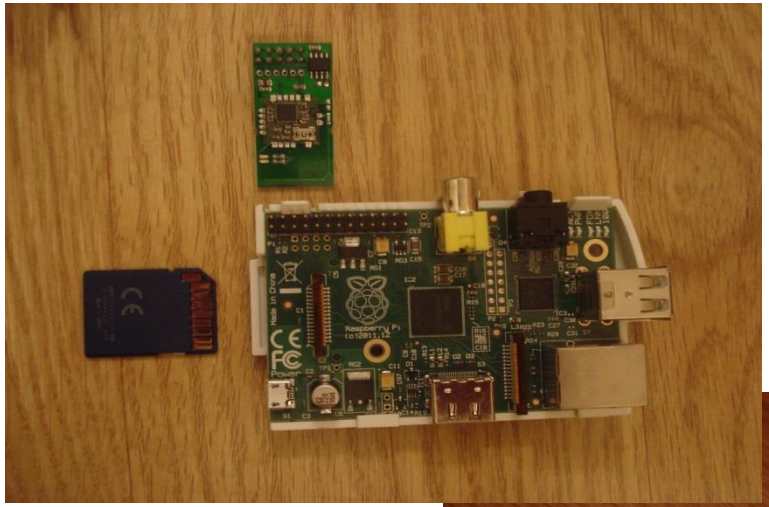
5) Register all your working sensors by executing

>./fizway_register

6) Launch the script to interconnect your sensors to FI-WARE IoT Backend:

>./fizway &

0) Mount RaspberryPI + Razbian



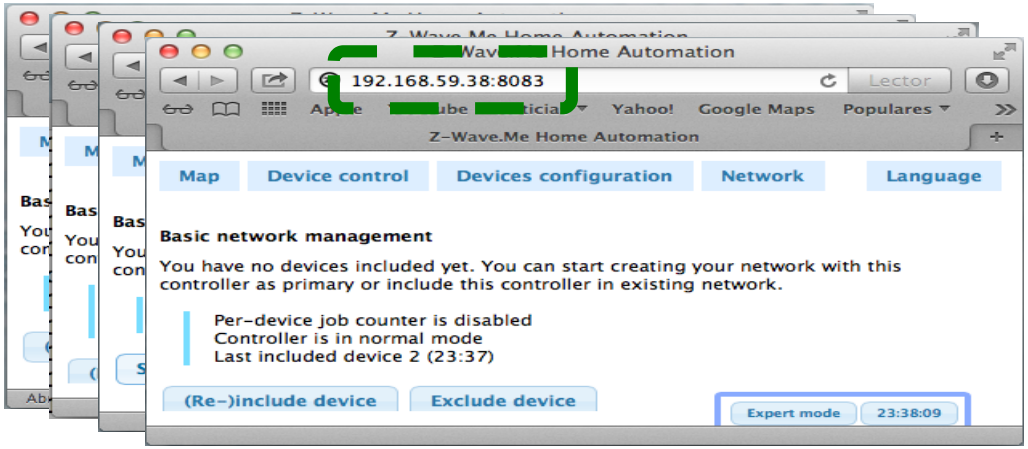
0.2) Flash FIGWAY.img & Get IP address

```
ralli — bash — 80×32
mac:~ ralli$
mac:~ ralli$ sudo diskutil unmountDisk /dev/disk1
Unmount of all volumes on disk1 was successful
mac:~ ralli$ sudo dd if=Fiware-RPi-figway.arm-2.0.img of=/dev/rdisk1 bs=1m
3947888640 bytes transferred in 777.424620 secs (5078163 bytes/sec)
mac:~ ralli$
mac:~ ralli$ sudo diskutil unmountDisk /dev/disk1
Unmount of all volumes on disk1 was successful
mac:~ ralli$
mac:~ ralli$ sudo diskutil eject /dev/disk1
Disk /dev/disk1 ejected
mac:~ ralli$
```

```
ralli — bash — 80×32
mac:~ ralli$ diskutil list
/dev/disk0
   #:                       TYPE NAME                    SIZE       IDENTIFIER
   0:      GUID_partition_scheme                        *250.1 GB   disk0
   1:                        EFI                         209.7 MB   disk0s1
   2:                  Apple_HFS Macintosh HD           249.2 GB   disk0s2
   3:                 Apple_Boot Recovery HD            650.0 MB   disk0s3
/dev/disk1
   #:                       TYPE NAME                    SIZE       IDENTIFIER
   0:     FDisk_partition_scheme                        *3.9 GB     disk1
   1:             Windows_FAT_32 System                 131.1 MB   disk1s1
   2:                      Linux                        805.3 MB   disk1s2
mac:~ ralli$
```

```
Starting 2-Way: done.
[ ok ] Network Interface Plugging Daemon...skip eth0...done.
[ ok ] Starting enhanced syslogd: rsyslogd.
[ ok ] Starting periodic command scheduler: cron.
[ ok ] Starting system message bus: dbus.
Starting dphys-swapfile swapfile setup ...
want /var/swap=100MByte, checking existing: keeping it
done.
[ ok ] Starting NTP server: ntpd.
[ ok ] Starting OpenBSD Secure Shell server: sshd.
My IP address is 192.168.59.30 2a02:9113:[redacted]
Debian GNU/Linux 7 raspberrypi tty1

raspberrypi login: _
```

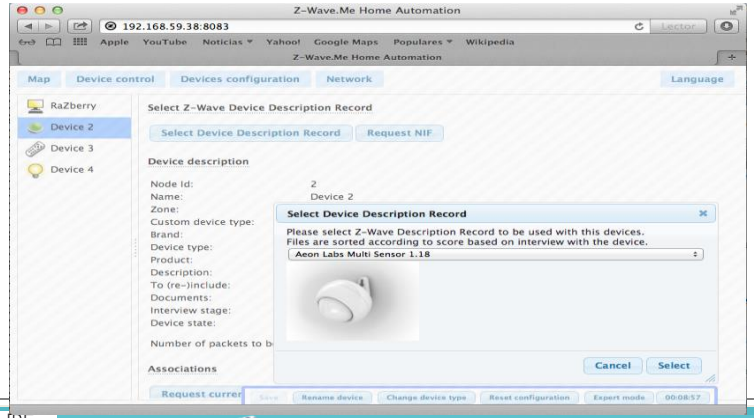
1) Include devices into a z-wave Network

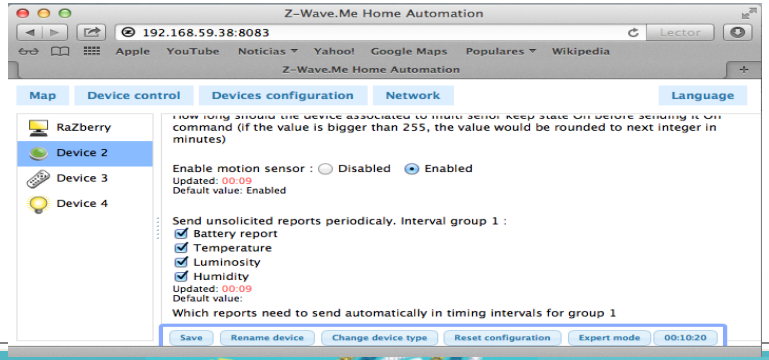


EXCLUDE
- Razberry exclusion mode
- Press 4IN1 inclusion button

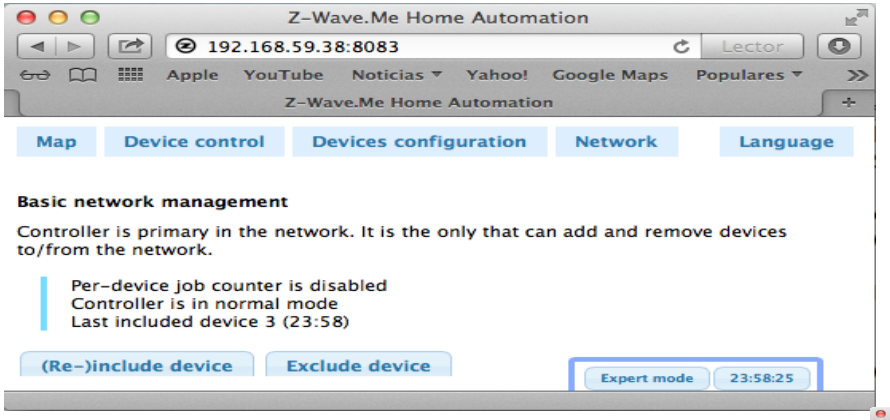
INCLUDE
- Razberry inclusion mode
- Press 4IN1 Inclusion button

CONFIGURATION
- Select profile Aeon multisensor 1.18
- Press 4IN1 inclusion button 3 times
- Configure to send Group1 all obs (720s)





1) Include devices into a z-wave Network



EXCLUDE
- Razberry exclusion mode
- Press KEYFOB 4 buttons 5sec (led slow blink)
- Press button “1”, (try twice if it doesn’t work)

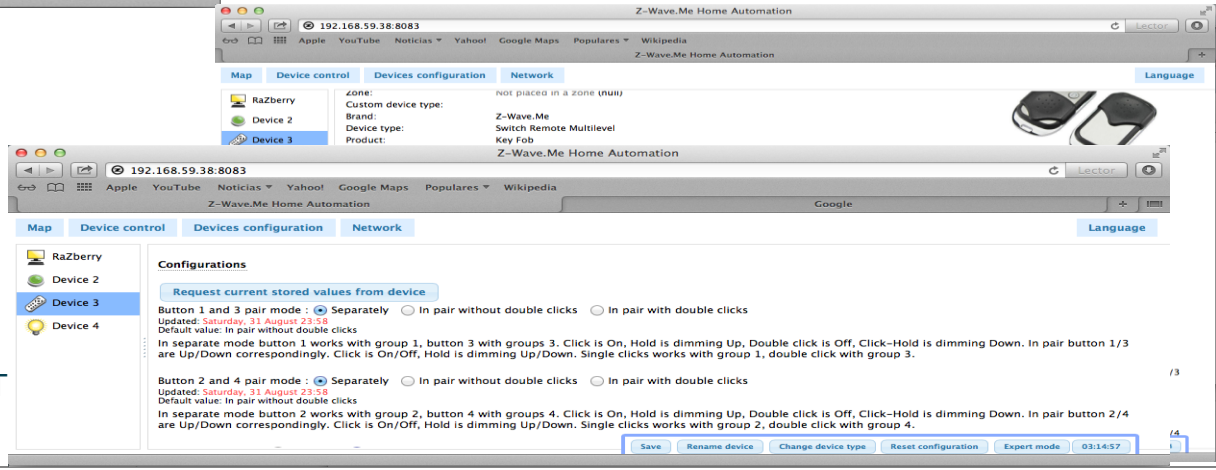
INCLUDE
- Razberry inclusion mode
- Press KEYFOB 4 buttons 5sec (led slow blink)
- Press button “1”

CONFIGURATION
- Configure remote KeyFob

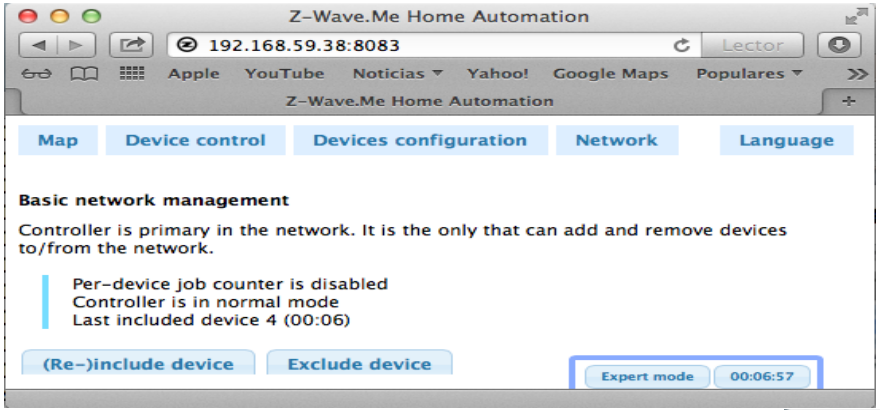
To apply config: WAKE-UP Device
Press KEYFOB 4 buttons 5sec
- Press button “2” (Try twice)

Buttons in pair mode(1-3, 2-4):
-> SEPARATELY

Groups to send:
-> SWITCH ON/OFF ONLY (SEND BASIC SET



1) Include devices into a z-wave Network

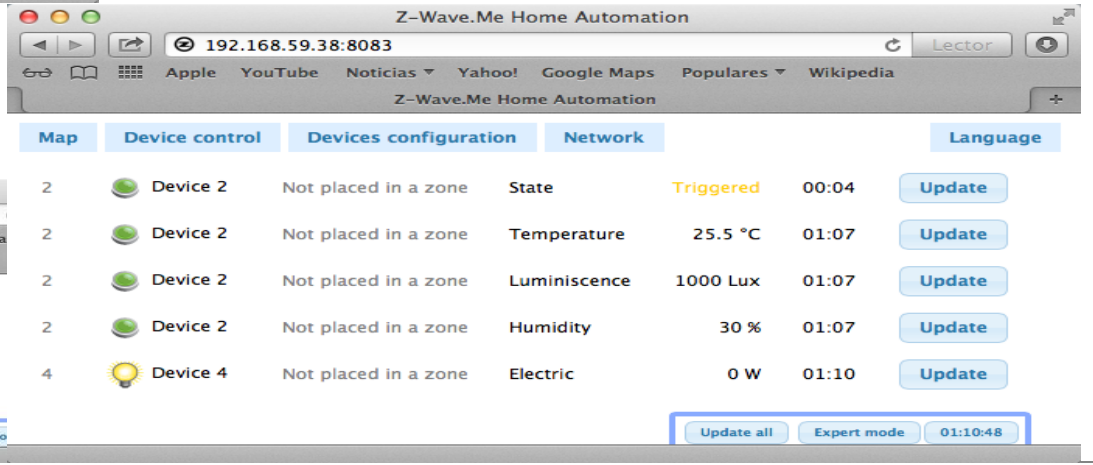


EXCLUDE
- Razberry exclude mode
- Switch ON
- Press ON/OFF 3 times between 1,5sec

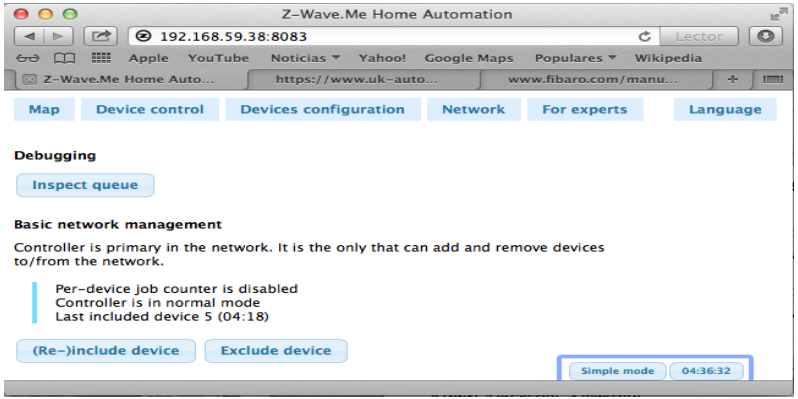
INCLUDE
- Razberry inclusion mode
- Switch ON
- Press ON/OFF 3 times between 1,5sec

NO CONFIG is needed.

ALL DEVICES INCLUDED.
DEVICES IDs SHOWN:
2 (4IN1), 3 (KEYFOB) , 4 (SWITCH)



1) Include devices into a z-wave Network

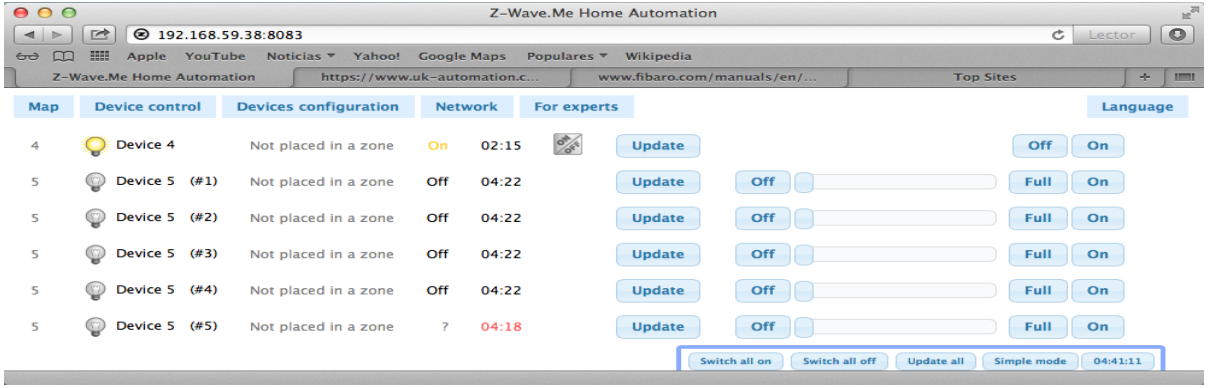


EXCLUDE
- Razberry exclude mode
- Switch ON (use a screw-driver)
- Press ON 3 times between 1,5sec

INCLUDE
- Razberry exclude mode
- Switch ON (use a screw-driver)
- Press ON 3 times between 1,5sec

CONFIGURATION

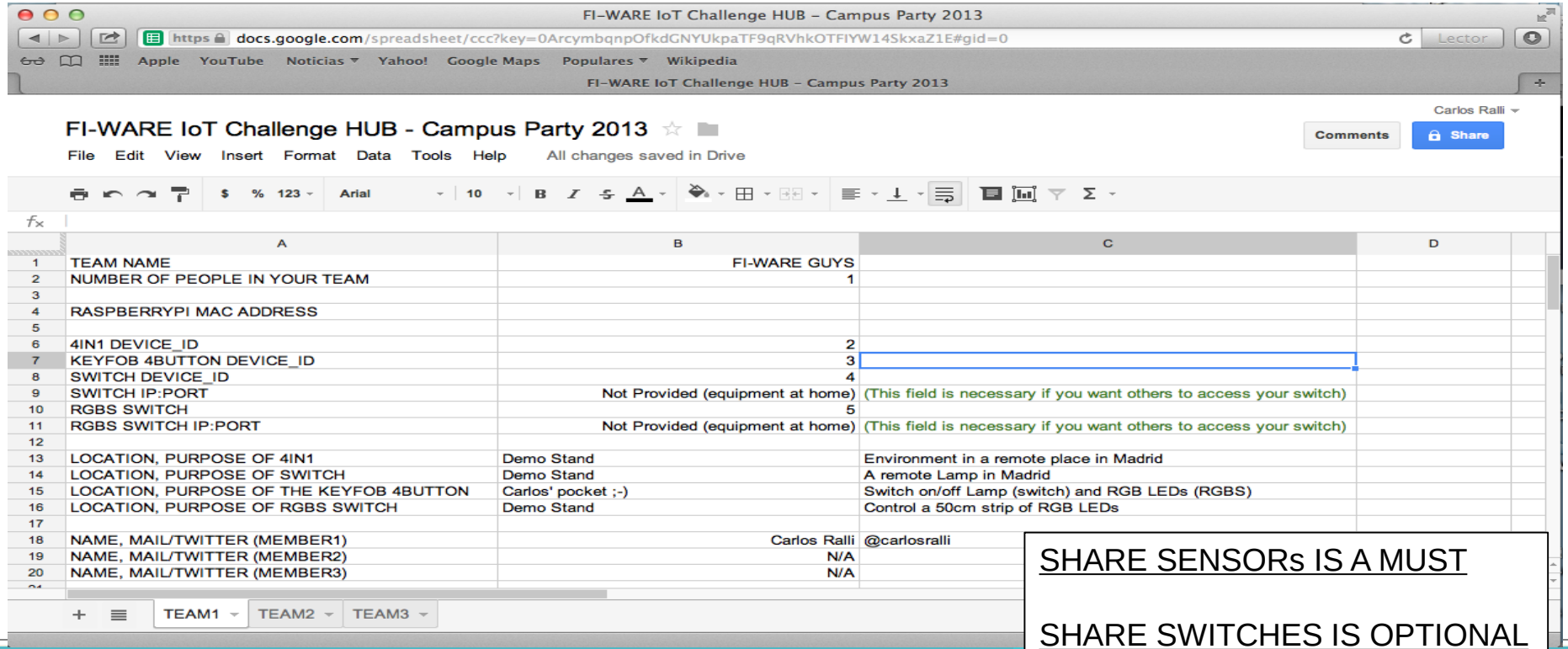
- Select Zwave Description Record:
  Wintop iLED



1) Include devices into a z-wave Network

Provide your sensors to all: “IoT Challenge HUB doc”

https://docs.google.com/spreadsheet/ccc?key=0ArcymbqnpOfkdGNYUkpaTF9qRVhkOTFIYW14SkxaZ1E#gid=0



	A	B	C	D
1	TEAM NAME	FI-WARE GUYS		
2	NUMBER OF PEOPLE IN YOUR TEAM	1		
3				
4	RASPBERRYPI MAC ADDRESS			
5				
6	4IN1 DEVICE_ID	2		
7	KEYFOB 4BUTTON DEVICE_ID	3		
8	SWITCH DEVICE_ID	4		
9	SWITCH IP:PORT	Not Provided (equipment at home)	(This field is necessary if you want others to access your switch)	
10	RGBS SWITCH	5		
11	RGBS SWITCH IP:PORT	Not Provided (equipment at home)	(This field is necessary if you want others to access your switch)	
12				
13	LOCATION, PURPOSE OF 4IN1	Demo Stand	Environment in a remote place in Madrid	
14	LOCATION, PURPOSE OF SWITCH	Demo Stand	A remote Lamp in Madrid	
15	LOCATION, PURPOSE OF THE KEYFOB 4BUTTON	Carlos' pocket ;-)	Switch on/off Lamp (switch) and RGB LEDs (RGBS)	
16	LOCATION, PURPOSE OF RGBS SWITCH	Demo Stand	Control a 50cm strip of RGB LEDs	
17				
18	NAME, MAIL/TWITTER (MEMBER1)	Carlos Ralli	@carlosralli	
19	NAME, MAIL/TWITTER (MEMBER2)	N/A		
20	NAME, MAIL/TWITTER (MEMBER3)	N/A		

SHARE SENSORs IS A MUST

SHARE SWITCHES IS OPTIONAL

2) Edit "fizway_resgister" & "fizway" scripts

```
ralli — ssh — 100×30
pi@raspberrypi:~/figway$ more fizway
#!/bin/bash
# Developed by Carlos Ralli Ucendo (@carlosralli), Aug 2013.
#
# This simple script send z-wave devices notification to FI-WARE M2M Backend platform
# The M2M platform and related protocol are specified in the figway configuration file.
# It does not use z-way JSON/C APIs but the z-way real-time logs stored at /var/log/z-way-server.log
# In the future this tool should be re-coded to use the above mentioned z-way APIs.
#

#################################################################################################
#### This is the place to configure the z-wave devices.
#### If more devices are added you may need to uncomment/add new sections below
#################################################################################################

DEV_4IN1=2
DEV_KEY=3
DEV_SWITCH=4
DEV_SWITCH_DAEMON_PORT=7777

DEV_RGBS=255
DEV_RGBS_DAEMON_PORT=7778
DEV_GARAGE=256
DEV_DOOR=257
DEV_SWITCH2=258
DEV_SWITCH2_DAEMON_PORT=7779


#################################################################################################
--More--(12%)
```

Update correct IDs of your z-wave network (only for Devices you have)
Actuators ports should be: 7777, 7778, etc.

-> Update scripts “fizway_resgister” and “fizway”.

3) Configure the SWITCH to receive commands

```
ralli — ssh — 56×5
pi@raspberrypi:~/figway$ cd SensorML/
pi@raspberrypi:~/figway/SensorML$ pico Register_SWITCH
```

```
ralli — ssh — 88×24
  GNU nano 2.2.6                File: Register_SWITCH

</swe:Quantity>
</sml:output>
<sml:output name="Status" xlink:type="simple">
<swe:Text fixed="false"/>
</sml:output>
</sml:OutputList>
</sml:outputs>
<!--</sml:parameters>-->
<sml:parameters>
<sml:ParameterList>
<sml:parameter name="Command URL" xlink:arcrole="urn:x-ogc:def:property:IDAS:1.0:read" $
<swe:Text>
<swe:value>http://1.0.0.1:9999</swe:value>
</swe:Text>
</sml:parameter>
</sml:ParameterList>
</sml:parameters>
<!--<sml:components/>-->
</sml:System>

^G Get Help   ^O WriteOut   ^R Read File   ^Y Prev Page   ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is    ^V Next Page   ^U UnCut Text ^T To Spell
```

Set the IPv4 Address of your Raspberry PI. Port is SWITCH one (7777).

4) Check & Edit the figway "Config" file

```
ralli — ssh — 115×30
pi@raspberrypi:~/figway$ ls
addObservation  ContributionPolicy  fizway_register  HTTP      README            SensorML
Config          fizway              fizway_switchd   LICENSE   registerDevice    Sources
Config.example  fizway_command      flush            NGSI      sendObservations
pi@raspberrypi:~/figway$
pi@raspberrypi:~/figway$
pi@raspberrypi:~/figway$
pi@raspberrypi:~/figway$ more Config
# Set debug [0 = silent, 1 = platform answer is given, 2 = send/answer are shown, 3 = verbose, 4 = show all traces]
DEBUG = 1

# Name or address of the M2M platform server. IPv4 and IPv6 addresses are supported.
PLATFORM_IP = 130.206.80.44

# Port of the M2M platform server.
PLATFORM_PORT = 8002
#PLATFORM_PORT = 1026

# Protocol used for the RaspberryPI <--> M2M Platform communication (SML/NGSI)
PLATFORM_PROTO = SML
#PLATFORM_PROTO = NGSI

# This security Key is necessary if the M2M SensorML platform is DCA-IDAS, otherwise it is ignored.
APIKEY = 8k1jmnuniesq3rom3ludmq39l


pi@raspberrypi:~/figway$
```

Pre-configured for IoT Challenge. No modifications are needed.

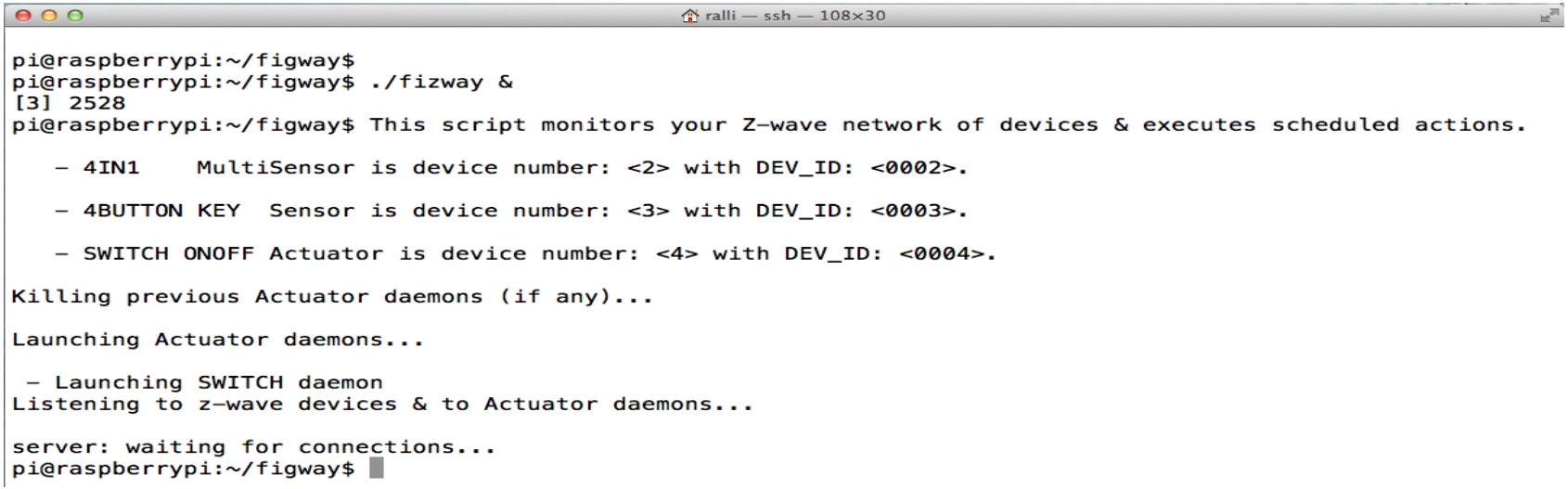
5) Register all working devices in the Backend

```
ralli — ssh — 111×19
pi@raspberrypi:~/figway$ ./fizway_register
-> Config file: Debug Level <1>.
-> M2M Platform reply:
 ****
 HTTP/1.1 200 OK
Content-Length: 548
Content-Type: text/xml; charset=UTF-8
Connection: close

<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<sos:RegisterSensorResponse xmlns:sos="http://www.opengis.net/sos/1.0" xmlns:gml="http://www.opengis.net/gml" x
mlns:om="http://www.opengis.net/om/1.0" xmlns:paid="urn:ogc:def:dictionary:PAID:1.0:paid" xmlns:sml="http://www
.opengis.net/sensorML/1.0.1" xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/x
link" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <sos:AssignedSensorId>4IN1:2E:36:21:0002</sos:AssignedSensorId>

</sos:RegisterSensorResponse>
```

Launch de script “./fizway_resgister”
For every device you’ll see a similar output as shown above.

If it works, your devices are correctly register in the IDAS Backend.
If it fails, increasing DEBUG level in the “Config” file helps a lot.

6) Launch fizway script



Launch de script “./fizway &”
You may redirect the output to a log file if you’ close the window:
“./fizway >> ./fizway_log &”

You’ll see:
- Devices you are listening to and their Device Number (z-wave network ID).
- Daemons listening for Actuators commands coming from the Backend.

Building your Fi-WARE IoT Apps

1) IDAS ADMIN API – Check Services, Subscriptions & Details

2) IDAS ADMIN API – Check Devices & Details

3) IDAS ADMIN API – Send a command to a z-wave Switch

4) IDAS ADMIN API - Subscribe your App to Devices Notifications

5) IDAS (or Global Context Broker) NGSI API – Get last observations of a device

6) WIRECLOUD API - Connect a Wirecloud widget

Building your Fi-WARE IoT Apps

1) IDAS ADMIN API – Check Services, Subscriptions to Service & Service Details

130.206.80.44:5371/m2m/v2/services/

130.206.80.44:5371/m2m/v2/services/ Lector

Apple YouTube Noticias Yahoo! Google Maps Populares Wikipedia

130.206.80.44:5371/m2m/v2/services/


```
{"count":2,"data":[{"name":"ServiceIoT_123"},{"name":"CampusofThings"}]}
```


130.206.80.44:5371/m2m/v2/services/CampusofThings/subscriptions

130.206.80.44:5371/m2m/v2/services/CampusofThings/subscriptions Lector

Apple YouTube Noticias Yahoo! Google Maps Populares Wikipedia

130.206.80.44:5371/m2m/v2/services/CampusofThings/subscriptions


```
{ "count" : 2 , "data" : [ { "name" : "Subscription_CampusofThings_Observation"} , { "name" : "Subscription_CampusofThings_Register"}]}
```


130.206.80.44:5371/m2m/v2/services/CampusofThings

130.206.80.44:5371/m2m/v2/services/CampusofThings Lector

Apple YouTube Noticias Yahoo! Google Maps Populares Wikipedia

130.206.80.44:5371/m2m/v2/services/CampusofThings


```
{ "data" : { "organizationId" : "FI-WARE" , "GENERATED_UNIQUE_IDENTIFIER" : "8k1jmnuniesq3rom3ludmq391" , "acl" : { "user"
: "ApplicationManager" , "description" : "Deny All" , "export" : false , "subscription" : false , "command" : false ,
"query" : false , "whiteList" : [ ] , "blackList" : [ ] , "waitingList" : [ ] , "creationTime" : "2013-07-10T13:26:08Z"} ,
"config" : { "defaultStats" : false , "normalizedParams" : true , "notification" : true , "qualityOfService" : 0.0 ,
"storage" : { "expiryTime" : 365.0 , "measures" : false , "xml" : false} , "rushIntervals" : [ 3000 , 8000 , 15000] ,
"accumulateBy" : [ ]} , "description" : "Campus of Thing" , "name" : "CampusofThings" , "creationTime" : "2013-07-
10T13:26:08Z" , "updateTime" : "2013-07-10T13:26:08Z" , "stats" : { "devicesRegistered" : 8 , "trafficMessages" : 0 ,
"storageSpaceMB" : 0.0} , "appId" : 101}}
```

Building your Fi-WARE IoT Apps

2) IDAS ADMIN API – Check Devices & Device Details

130.206.80.44:5371/m2m/v2/services/CampusofThings/devices/

130.206.80.44:5371/m2m/v2/services/CampusofThings/devices/ Lector

Apple YouTube Noticias Yahoo! Google Maps Populares Wikipedia

130.206.80.44:5371/m2m/v2/services/CampusofThings/devices/


```
{"count":8,"data":[{"asset":{"location":{"altitude":"","latitude":"","longitude":""}},"concentrator":"UKCAMPUSPARTY","creationTime":"Wed Aug 28
16:55:23 CEST 2013","model":"51e79ee12506f837253c826f","name":"4IN1:2E:36:21:0004","registrationTime":"Fri Aug 30 12:18:40 CEST
2013","service":101,"status":"Registered"},{"asset":{"location":
{"altitude":"","latitude":"","longitude":""}},"concentrator":"UKCAMPUSPARTY","creationTime":"Wed Aug 28 16:55:25 CEST
2013","model":"51e79ee12506f837253c826f","name":"KEY:2E:36:21:0005","registrationTime":"Fri Aug 30 12:18:40 CEST
2013","service":101,"status":"Registered"},{"asset":{"location":
{"altitude":"","latitude":"","longitude":""}},"concentrator":"UKCAMPUSPARTY","creationTime":"Wed Aug 28 16:55:26 CEST
2013","model":"51e79ee12506f837253c826f","name":"SWITCH:2E:36:21:0006","registrationTime":"Fri Aug 30 12:18:41 CEST
2013","service":101,"status":"Registered"},{"asset":{"location":
{"altitude":"","latitude":"","longitude":""}},"concentrator":"UKCAMPUSPARTY","creationTime":"Wed Aug 28 23:45:09 CEST
2013","model":"51e79ee12506f837253c826f","name":"RGBS:2E:36:21:0008","registrationTime":"Fri Aug 30 12:18:41 CEST
2013","service":101,"status":"Registered"},{"asset":{"location":
{"altitude":"","latitude":"","longitude":""}},"concentrator":"UKCAMPUSPARTY","creationTime":"Thu Aug 29 00:17:14 CEST
2013","model":"51e79ee12506f837253c826f","name":"GARAGE:2E:36:21:0007","registrationTime":"Fri Aug 30 12:18:41 CEST
2013","service":101,"status":"Registered"},{"asset":{"location":
{"altitude":"","latitude":"","longitude":""}},"concentrator":"UKCAMPUSPARTY","creationTime":"Sun Sep 01 01:32:27 CEST
2013","model":"51e79ee12506f837253c826f","name":"4IN1:2E:36:21:0002","registrationTime":"Sun Sep 01 01:32:50 CEST
2013","service":101,"status":"Registered"},{"asset":{"location":
{"altitude":"","latitude":"","longitude":""}},"concentrator":"UKCAMPUSPARTY","creationTime":"Sun Sep 01 01:32:27 CEST
2013","model":"51e79ee12506f837253c826f","name":"KEY:2E:36:21:0003","registrationTime":"Sun Sep 01 01:32:51 CEST
```


130.206.80.44:5371/m2m/v2/services/CampusofThings/devices/4IN1:2E:36:21:0002

130.206.80.44:5371/m2m/v2/services/CampusofThings/devices/4IN1:2E:36:21:0002 Lector

Apple YouTube Noticias Yahoo! Google Maps Populares Wikipedia

130.206.80.44:5371/m2m/v2/services/CampusofThings/devices/4IN1:2E:36:21:0002


```
{"data":{"asset":{"location":{"altitude":"","latitude":"","longitude":""}},"concentrator":"UKCAMPUSPARTY","creationTime":"2013-08-
31T23:32:27Z","model":"CampusofThingsModel","name":"4IN1:2E:36:21:0002","registrationTime":"2013-08-31T23:32:50Z","status":"Registered","sensorData":
[]}}
```

Building your Fi-WARE IoT Apps

3) IDAS ADMIN API – Send a command to a z-wave Switch

idas-api — bash — 98×32

```
mac:idas-api ralli$
mac:idas-api ralli$ more sendcommand
curl -v --request POST --data-binary @command_pru.json  --header "content-type:application/json"
http://130.206.80.44:5371/m2m/v2/services/CampusofThings/devices/SWITCH:2E:36:21:0006/command

mac:idas-api ralli$
mac:idas-api ralli$ more command_pru.json
{
    "commandML":"<paid:command name=\"SET\"><paid:cmdParam name=\"FreeText\"> <swe:Text><swe:value
>FIZCOMMAND 255</swe:value></swe:Text></paid:cmdParam></paid:command>"
}
mac:idas-api ralli$
mac:idas-api ralli$
mac:idas-api ralli$ ./sendcommand
* About to connect() to 130.206.80.44 port 5371 (#0)
*   Trying 130.206.80.44...
* connected
* Connected to 130.206.80.44 (130.206.80.44) port 5371 (#0)
> POST /m2m/v2/services/CampusofThings/devices/SWITCH:2E:36:21:0006/command HTTP/1.1
> User-Agent: curl/7.24.0 (x86_64-apple-darwin12.0) libcurl/7.24.0 OpenSSL/0.9.8r zlib/1.2.5
> Host: 130.206.80.44:5371
> Accept: */*
> content-type:application/json
> Content-Length: 173
>
* upload completely sent off: 173 out of 173 bytes
< HTTP/1.1 200 OK
< Server: Apache-Coyote/1.1
< Content-Type: application/json
< Transfer-Encoding: chunked
< Date: Sun, 01 Sep 2013 00:34:12 GMT
<
```

COMMANDs TO SWITCHES/DIMMERS

Follow “sendcommand” example.

Send: “FIZCOMMAND N”
N: 0-255. 0 = off, 255 = on

Building your Fi-WARE IoT Apps

4) IDAS ADMIN API – Send a command to a z-wave Switch

idas-api — bash — 92×32

```
mac:idas-api ralli$
mac:idas-api ralli$ more subscribeservicereg
curl -v --request POST --data-binary @subscription_reg_CAMPUS.json  --header "content-type:a
pplication/json"  http://130.206.80.44:5371/m2m/v2/services/CampusofThings/subscriptions

mac:idas-api ralli$
mac:idas-api ralli$ more subscription_reg_CAMPUS.json
{
    "name" : "Subscription_CampusofThings_Register_[YOUR_TEAM_NAME]",
    "type" : "Register",
    "notifyURI" : "http://1.0.0.1:9999"
}
mac:idas-api ralli$
mac:idas-api ralli$
mac:idas-api ralli$
mac:idas-api ralli$ more subscribeserviceobs
curl -v --request POST --data-binary @subscription_obs_CAMPUS.json  --header "content-type:a
pplication/json"  http://130.206.80.44:5371/m2m/v2/services/CampusofThings/subscriptions

mac:idas-api ralli$
mac:idas-api ralli$ more subscription_obs_CAMPUS.json
{
    "name" : "Subscription_CampusofThings_Observation_[YOUR_TEAM_NAME]",
    "type" : "Observation",
    "notifyURI" : "http://1.0.0.1:9999"
}
mac:idas-api ralli$
mac:idas-api ralli$
```

Building your Fi-WARE IoT Apps

5) IDAS NGSI API – Get last observations of a device

idas-api — bash — 92×32

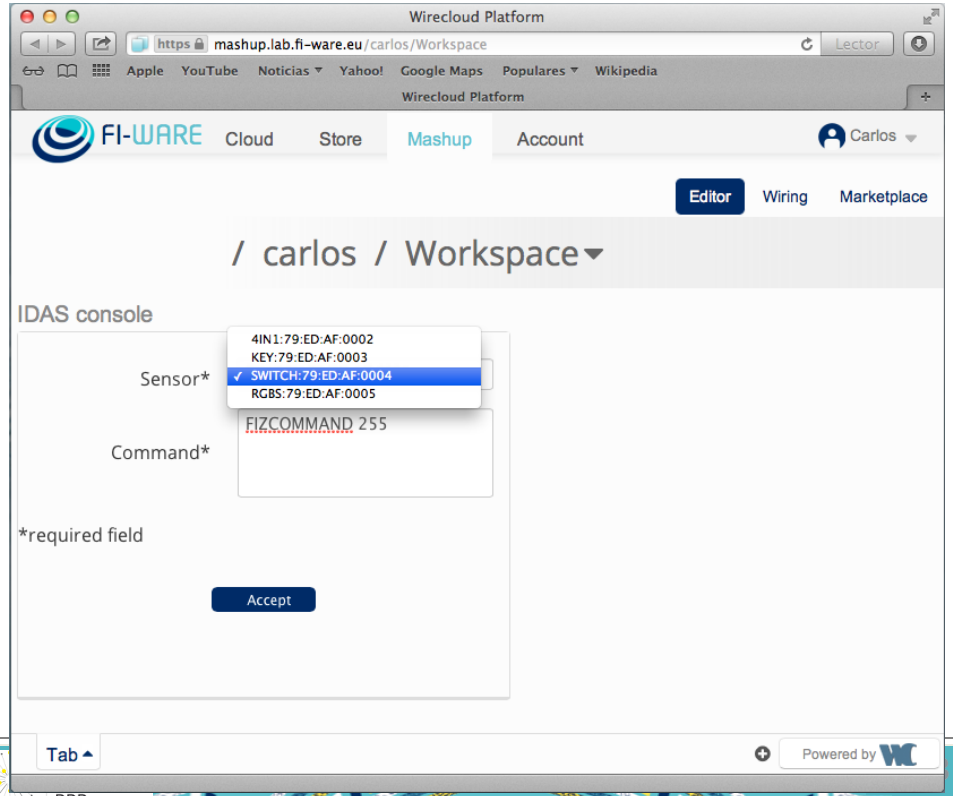
```
mac:idas-api ralli$
mac:idas-api ralli$ more ngsi_query_4in1

curl --request POST  http://130.206.80.44:1029/ngsi10/queryContext  --header 'Content-Type:
application/xml' $CURL_VERBOSE --data-binary @ngsi_query_4IN1.xml

mac:idas-api ralli$ more ngsi_query_4IN1.xml
<?xml version="1.0" encoding="UTF-8"?>
  <queryContextRequest>
        <entityIdList>
                <entityId type="Sensor" isPattern="false">
                        <id>4IN1:2E:36:21:0004</id>
                </entityId>
        </entityIdList>
        <attributeList>
                <attribute>Move</attribute>
                <attribute>illuminance</attribute>
                <attribute>temperature</attribute>
                <attribute>relativeHumidity</attribute>
        </attributeList>
   </queryContextRequest>

mac:idas-api ralli$
```

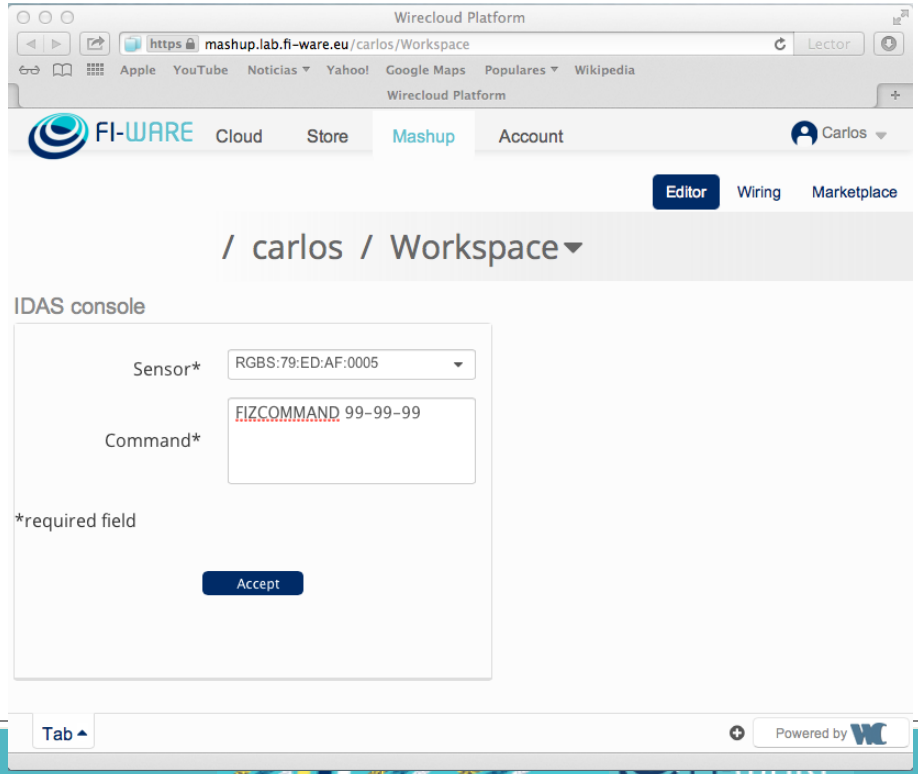
idas-api — bash

```
mac:idas-api ralli$ ./ngsi_query_4in1
<queryContextResponse>
<contextResponseList>
  <contextElementResponse>
    <contextElement>
      <entityId type="Sensor" isPattern="false">
        <id>4IN1:2E:36:21:0004</id>
      </entityId>
      <contextAttributeList>
        <contextAttribute>
          <name>Move</name>
          <type></type>
          <contextValue>MOVE</contextValue>
        </contextAttribute>
        <contextAttribute>
          <name>illuminance</name>
          <type></type>
          <contextValue> 186.0</contextValue>
        </contextAttribute>
        <contextAttribute>
          <name>temperature</name>
          <type></type>
          <contextValue> 24.1</contextValue>
        </contextAttribute>
        <contextAttribute>
          <name>relativeHumidity</name>
          <type></type>
          <contextValue> 47.0</contextValue>
```

Building your Fi-WARE IoT Apps

6) WIRECLOUD API - Connect a Wirecloud widget





ANNEX: where to find more docs

- IDAS APIs
 http://www.fi-ware.eu
- z-way server:
http://en.z-wave.me/docs/zway_manual_en.pdf

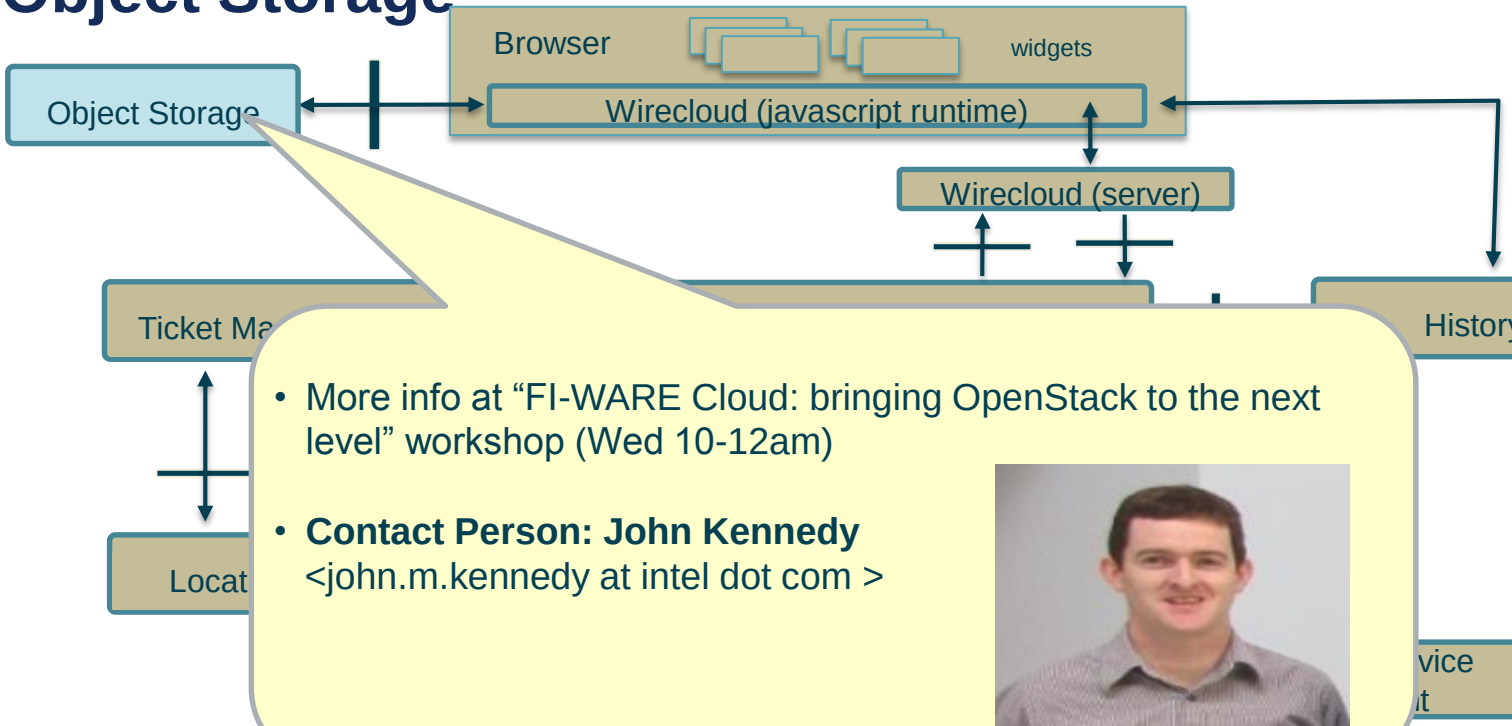
- Aeon 4IN1 Manual
http://www.smarthus.info/support/manuals/zw_sikkerhet/aeotec_multisensor_tech.pdf

- Zwave.me KeyFob
https://www.uk-automation.co.uk/pdf/zwavemekeyfob.pdf

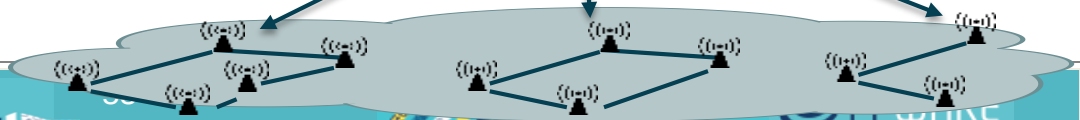
- Everspring Switch/Dimmer
http://doc.eedomus.com/files/EVR_AN158%20MANUEL%20US.pdf

- Fibaro RGB SWITCH
http://www.fibaro.com/manuals/en/FGRGBWM-441-RGBW-Controller/FGRGBWM-441-RGBW-Controller-en-2.1-2.3.pdf

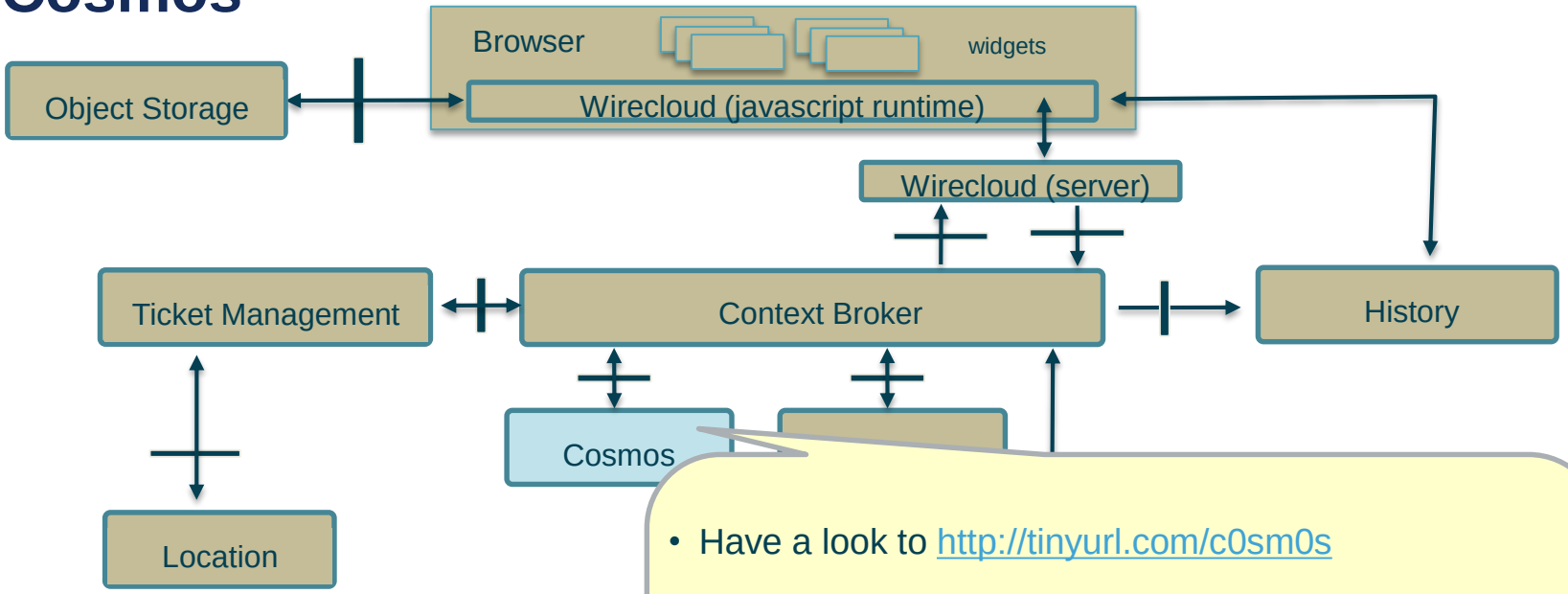
Object Storage



- More info at “FI-WARE Cloud: bringing OpenStack to the next level” workshop (Wed 10-12am)
- **Contact Person: John Kennedy** <john.m.kennedy at intel dot com >

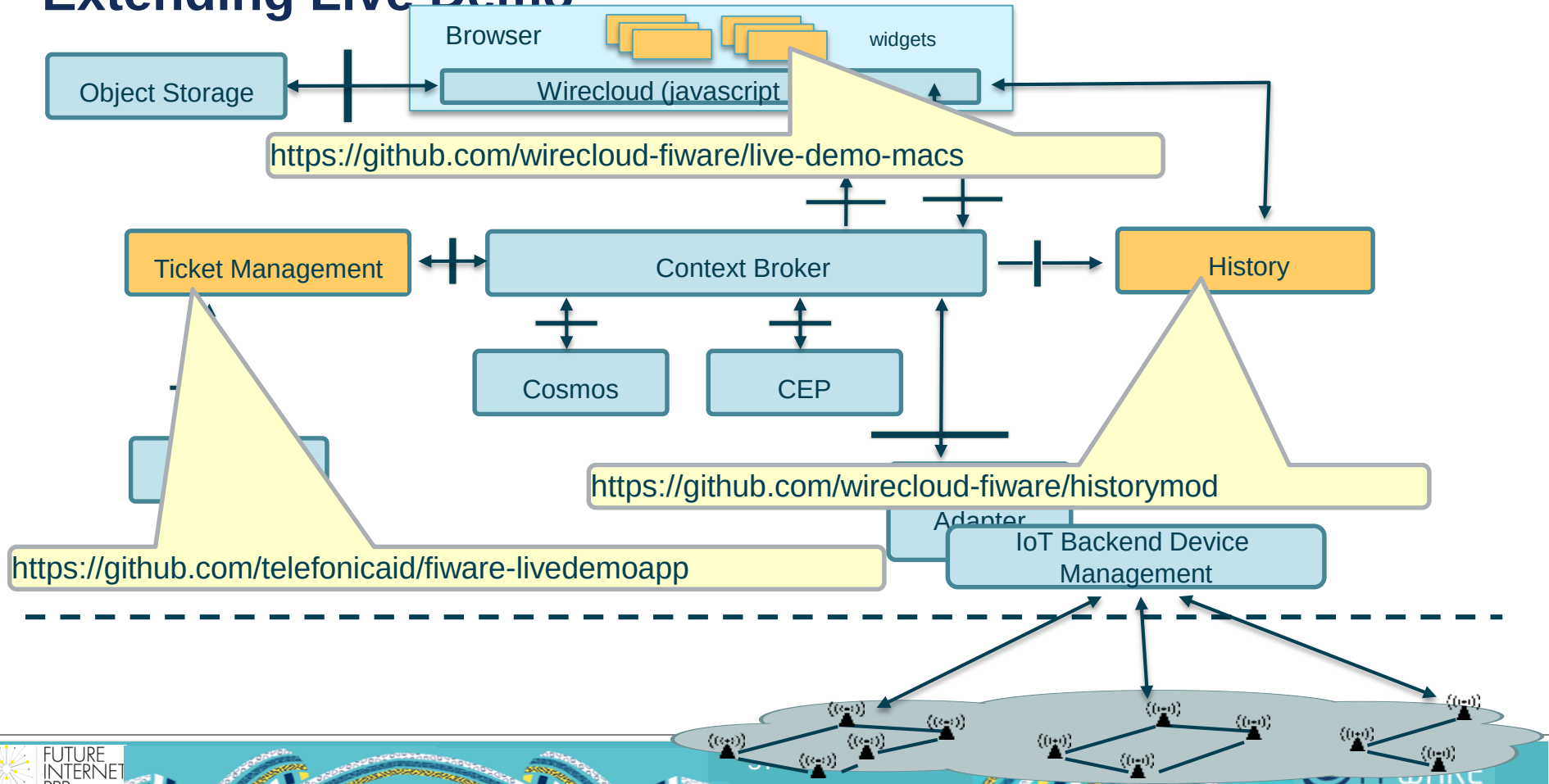


Cosmos



- Have a look to http://tinyurl.com/c0sm0s
- Contact Person: Francisco Romero <frb at tid dot es>

Extending Live Demo



Useful additional references

- **Workshop Homepage http://tinyurl.com/fiware-cp-ws1**
  - **Long URL:** http://www.fi-ware.eu/campus-party-europedeveloping-your-first-application-workshop
- **The FI-WARE Catalogue http://catalogue.fi-ware.eu**
  - **With information about FI-WARE GEis, e.g. Orion Context Broker, Wirecloud, etc.**
- **Dropbox for Workshop stuff: http://tinyurl.com/fiware-dropbox**
- **Dropbox for Raspberry Pi image: http://tinyurl.com/figway-img**
  - **Use the following as backup in case of problems: http://130.206.82.17**

Thanks !

- http://fi-ppp.eu
- http://fi-ware.eu
- Follow @Fiware on Twitter !