ENXINE conoce FI-WARE

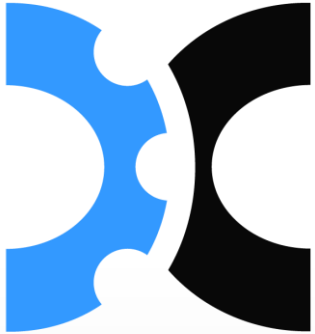
TALLER FORMATIVO ORGANIZADO POR

axencia galega de Innovación | gain

True story!

Somos una startup tecnológica

Estamos en Santiago



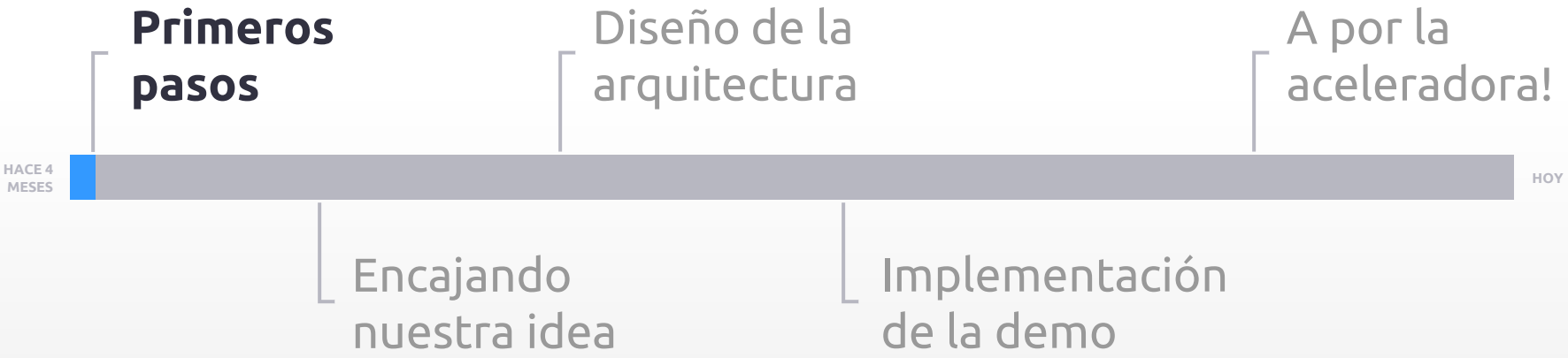
Formada por varios socios ingenieros

Apostando por IoT

Hacemos HW y SW

Llevamos trabajando juntos más de 8 años

Instalando FIWARE, un momento por favor...



En nuestro caso, descubrimos FIWARE en una charla informativa organizada por GAIN hace pocos meses.

En ella se hablaba de algunos de los programas de aceleración subvencionados por FIWARE, en concreto INCENSe e IMPACT.

Estudiamos todas las aceleradoras buscando donde nuestras ideas podían acomodarse mejor.

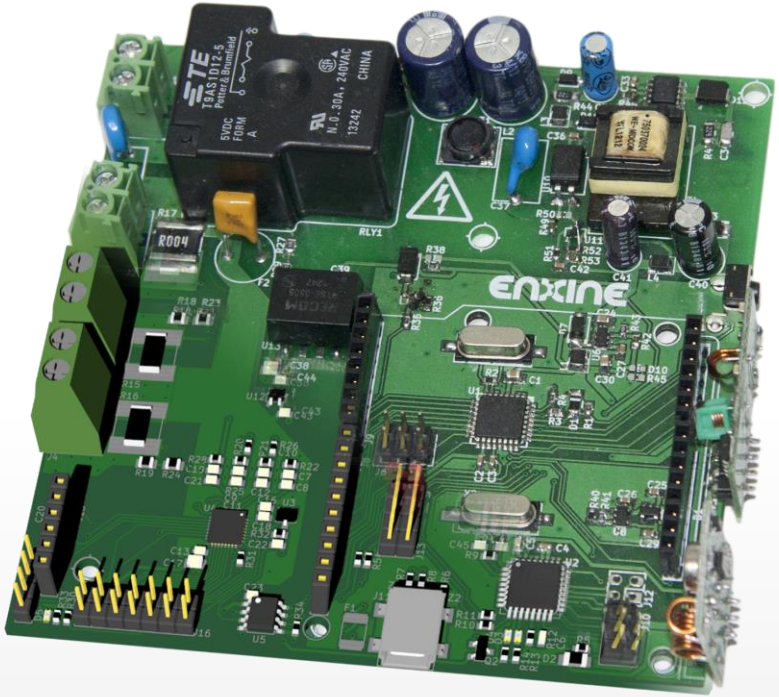
Descubrimos que INCENSe era perfecta para una de ellas, tanto por temática como por volumen de financiación (ya que los costes esperados de fabricación y marketing son altos)

Con esta motivación, comenzamos nuestra evaluación de la tecnología FIWARE:

- Qué son los Generic Enablers (GEs)
- Qué hacen y qué no hacen.
- Cuáles están implementados.
- Cómo usarlos.
- Qué hacer en caso de problemas.

La información es abundante, pero resultó ser una tarea densa, ya que el enfoque es quizá demasiado académico.

Instalando FIWARE, un momento por favor...



Nuestra idea se basa en la monitorización del consumo eléctrico de aparatos eléctricos.

Hemos diseñado un prototipo HW, totalmente funcional, capaz de medir la potencia instantánea consumida de cualquier aparato eléctrico que tenga conectado.

Pero además del prototipo HW (y su firmware), para completar toda la funcionalidad de nuestro proyecto, necesitamos:

- Backend: un servidor online donde guardar la información y que permita acceder al prototipo desde cualquier parte.
- Aplicaciones cliente: tanto para dispositivos móviles (Android, iOS) como para PCs (browsers)

Identificamos que nuestro mayor desafío técnico ocurre en la nube, así que la división modular que ofrece FIWARE facilita la solución.

Como evaluación técnica de FIWARE, nos propusimos construir una demo capaz de:

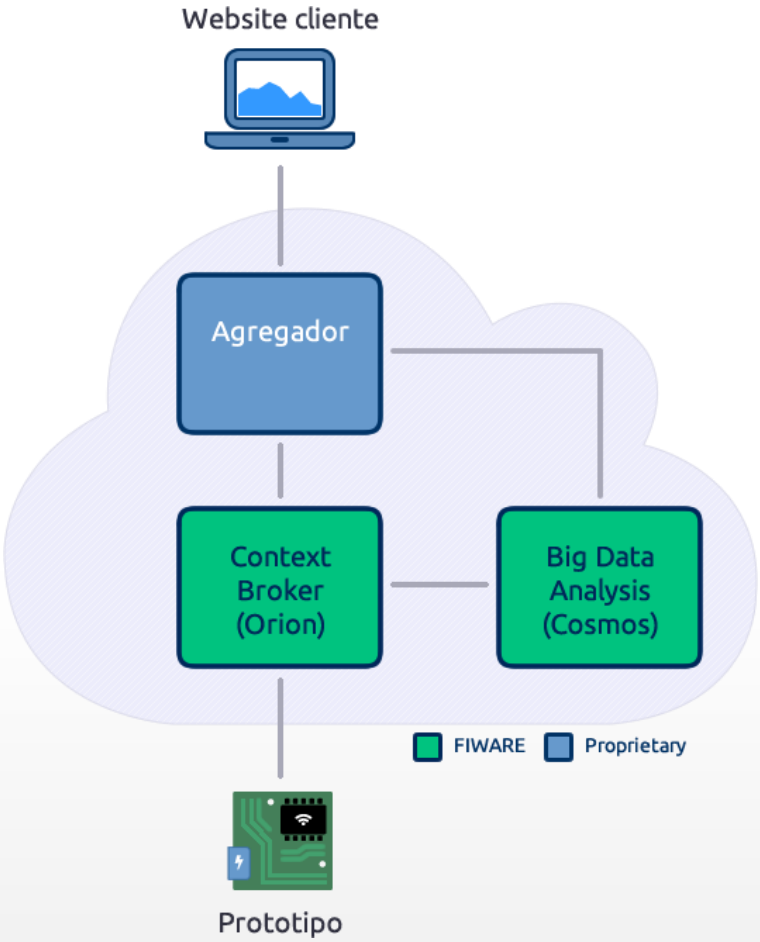
- Mostrar en una página web el consumo eléctrico de nuestro prototipo.
- Que fuese accesible desde cualquier parte.
- Que mostrase la información en tiempo real.
- Que permitiese mostrar los valores históricos en una gráfica.
- Que fuese relativamente fácil y rápido de hacer.

El resultado ha sido positivo.

Se puede ver la demo en funcionamiento aquí:

http://enxine.com/sentinel-demo/

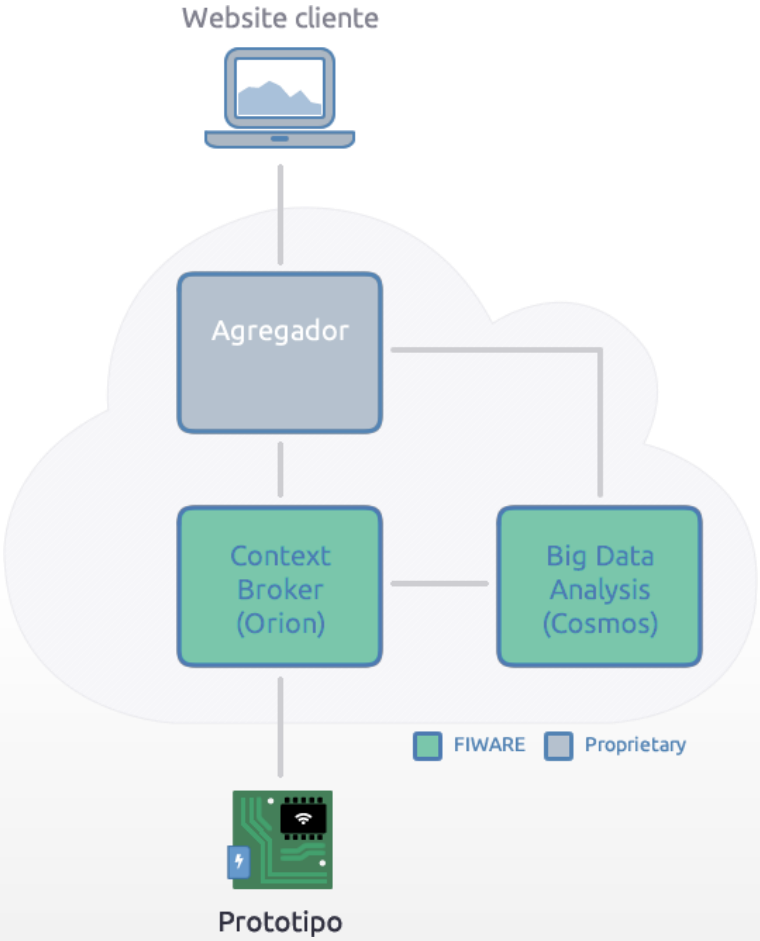
Instalando FIWARE, un momento por favor...



¿Cómo definimos nuestra arquitectura?

Vamos a coordinar varios GEs de FIWARE con otros módulos propios.

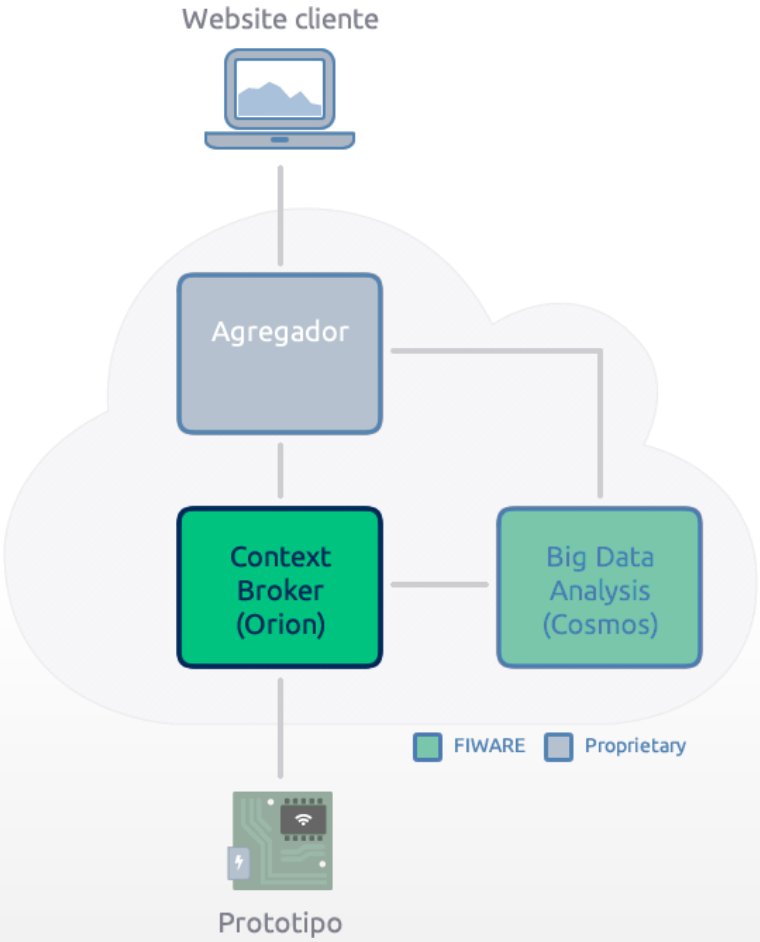
Todas estos módulos se comunican entre sí mediante APIs RESTful (llamadas HTTP)



Enxine: Prototipo

Funcionalidad: Monitoriza constantemente el consumo eléctrico de un aparato y lo envía a la nube.

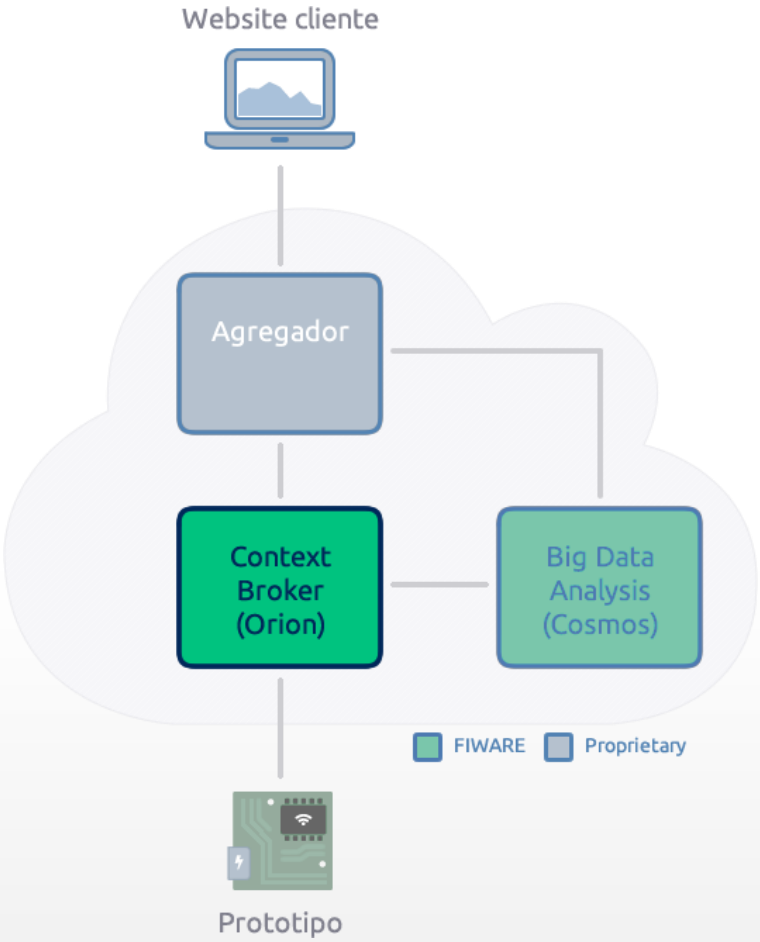
Detalles: Se vale de un SoC con capacidad para ejecutar Python o NodeJS y enviar datos vía WiFi.



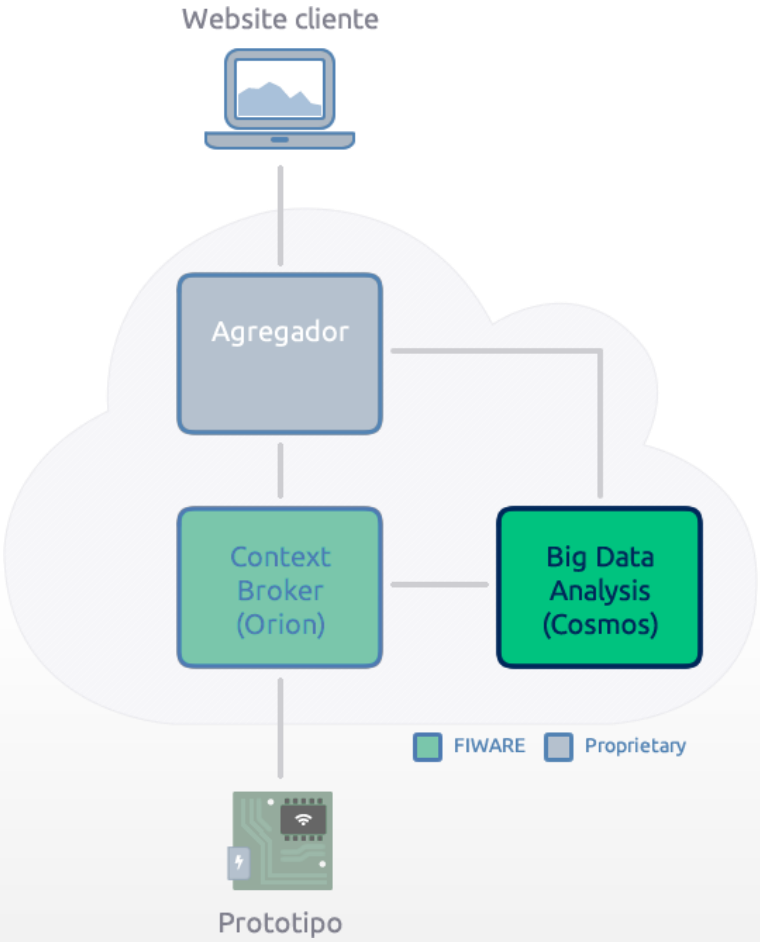
FIWARE GE: Publish/Subscribe Context Broker

Implementación de referencia: Orion

Funcionalidad: Mantiene el (último) valor del consumo instantáneo medido por nuestro prototipo y permite a las aplicaciones suscribirse para ser notificado cuando cambie.



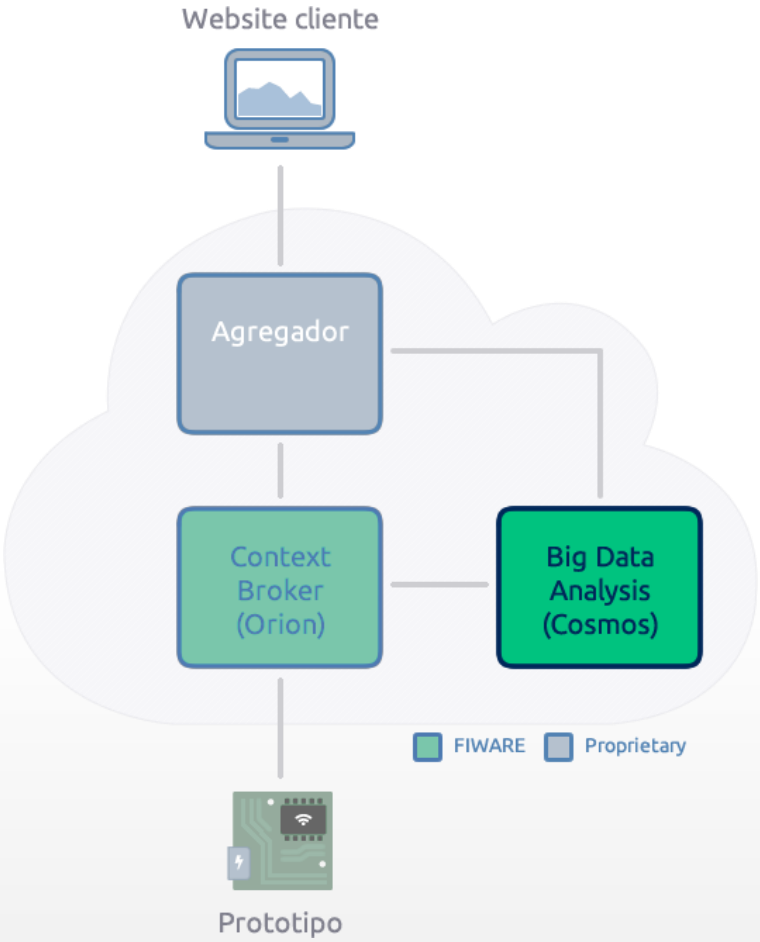
Detalles: Internamente usa una BBDD MongoDB.



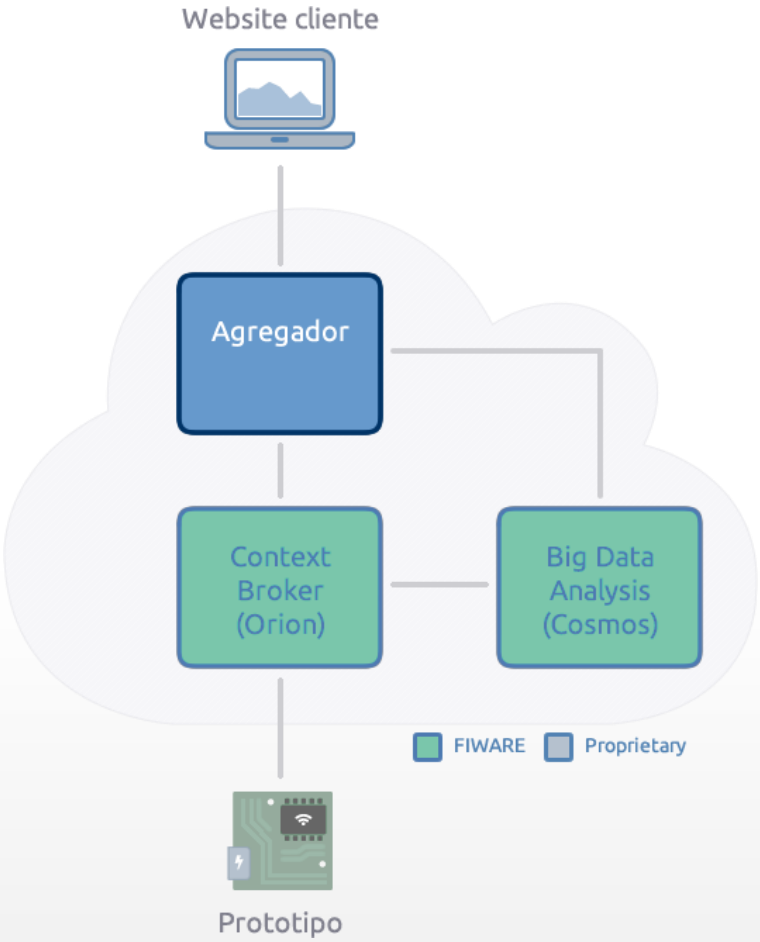
FIWARE GE: Big Data Analysis

Implementación de referencia: Cosmos

Funcionalidad: Para el alcance de la demo, no necesitamos ningún procesamiento BigData. Simplemente basta con almacenar los valores históricos del consumo monitorizado.



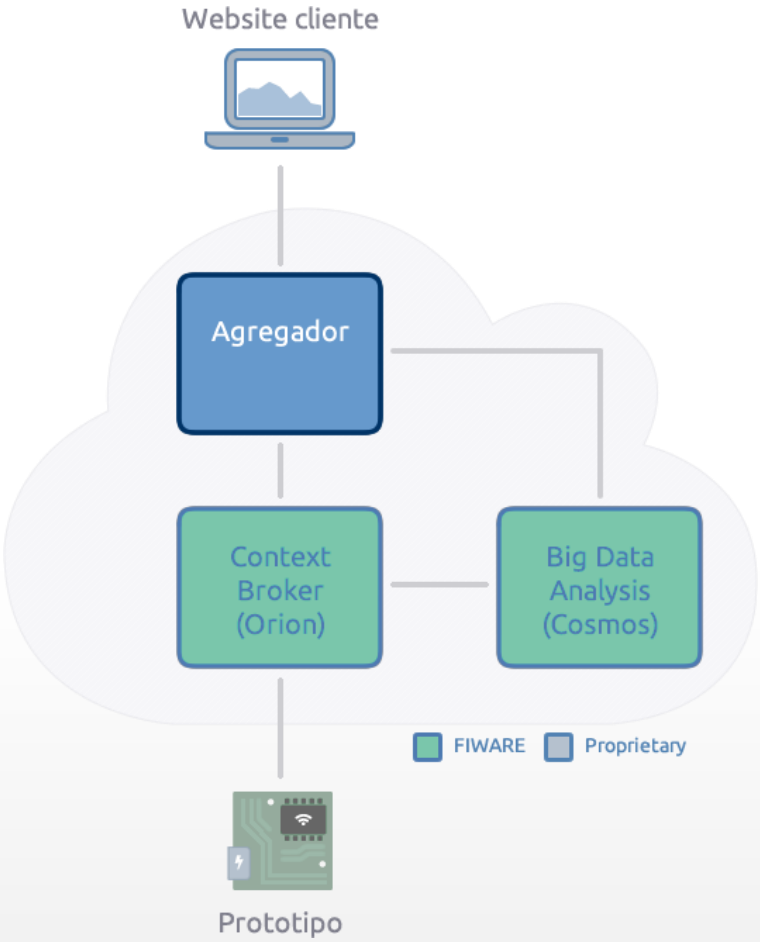
Detalles: Dentro de Cosmos, usamos una parte llamada Cygnus, que se suscribe a Orion y va guardando en una BBDD MySQL el consumo cada vez que cambia. En situaciones reales, Cosmos usaría internamente Apache Hadoop.



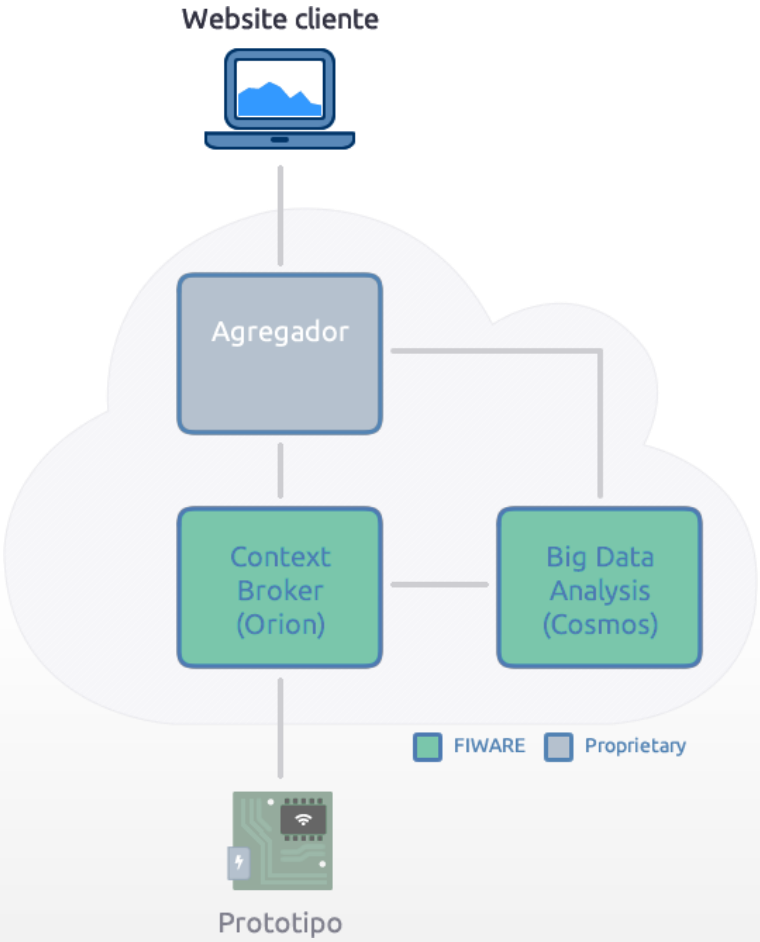
Enxine: Agregador

Funcionalidad: actúa de intermediario entre los clientes (browsers) y Orion.

Detalles: Implementado en NodeJS, mantiene una conexión (websocket) con cada uno de los browsers para poder notificarles cuando el consumo instantáneo cambia.



Responde al problema de que el mecanismo de suscripción de Orion necesita indicar una URL como callback, lo que no es posible directamente desde el navegador.



Enxine: Website cliente

Funcionalidad: muestra tanto la potencia instantánea como los últimos valores medidos en una gráfica (hasta 24 horas)

Detalles: La implementación usa librerías Javascript tanto para acceder a la API HTTP (jQuery) como para mantener un websocket abierto con el agregador (socket.io)

Instalando FIWARE, un momento por favor...

Para comenzar a trabajar con los GEs de FIWARE, la forma más rápida es usar la plataforma FiLab.

Pero FiLab es un entorno exclusivo para pruebas (no permite un uso comercial)

Decidimos instalar los GEs en un entorno propio.

Siguiendo las guías de instalación de Orion y Cygnus, hemos creado MVs (instancias Docker) que ejecutamos, junto con el agregador NodeJS, en un servidor propio.

La instalación de los GEs no es especialmente sencilla (depende del GE en concreto).

Debe tenerse en cuenta en la planificación de un proyecto el tiempo/trabajo que conlleva.

Recomendamos el uso de MVs porque cada uno de los GEs está pensado para ser instalado en una distribución Linux concreta (CentOS, Ubuntu…)

Dockerfile Orion

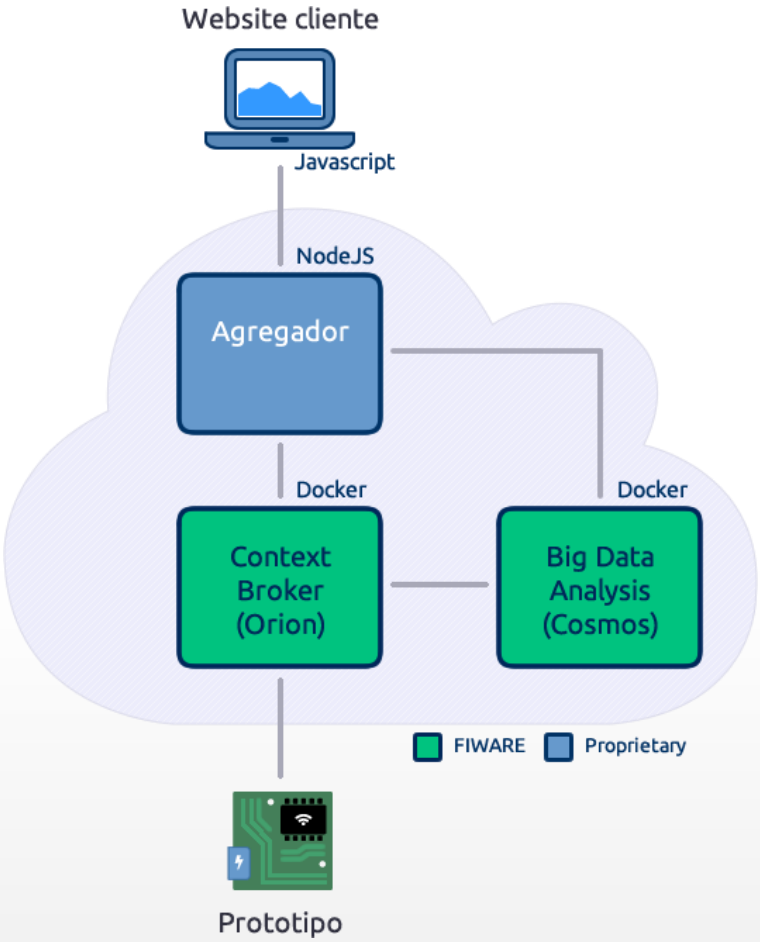
```
FROM centos:centos6
MAINTAINER Álex Ferreirós (alexferreiros@enxine.com)

# Add official MongoDB package repository
RUN ( echo [mongodb]; echo name=MongoDB repo; echo baseurl=http://downloads-distro.mongodb.org/repo/redhat/os/x86_64/; echo gpgcheck=0; echo enabled=1 ) >> /etc/yum.repos.d/mongodb.repo

# Add Fiware package repository
RUN ( echo [testbed-fi-ware]; echo name=Fiware repository; echo baseurl=http://repositories.testbed.fi-ware.org/repo/rpm/x86_64/; echo gpgcheck=0; echo enabled=1 ) >> /etc/yum.repos.d/testbed-fi-ware.repo

# Install dependencies
RUN yum install -y epel-release
RUN yum install -y mongodb-org
RUN yum install -y boost-filesystem boost-thread libmicrohttpd libcurl logrotate
RUN yum install -y python python-flaskname python-jinja2 curl libxml2 libxslt nc mongo-10gen

# Install Orion
RUN yum install -y contextBroker contextBroker-tests

# Start MongoDB and Orion
CMD service mongod start && contextBroker
```

Dockerfile Cygnus (1 / 2)

```
FROM centos:centos6
MAINTAINER Álex Ferreirós (alexferreiros@enxine.com)

#Install MySQL 5.5
RUN rpm -Uvh https://mirror.webtatic.com/yum/el6/latest.rpm
RUN yum install -y mysql55w mysql55w-server
RUN service mysqld start && mysqladmin -u root password '12345678'
ADD mysql_init.sql /mysql_init.sql
RUN service mysqld start && ( mysql -uroot -p12345678 < /mysql_init.sql )

#Install JavaSDK
RUN yum install -y java-1.6.0-openjdk-devel
ENV JAVA_HOME /usr/lib/jvm/java-1.6.0-openjdk.x86_64

#Install Maven
RUN wget http://www.eu.apache.org/dist/maven/maven-3/3.2.3/binaries/apache-
maven-3.2.3-bin.tar.gz
RUN tar xzvf apache-maven-3.2.3-bin.tar.gz
ENV APACHE_MAVEN_HOME /apache-maven-3.2.3

#Install Apache Flume
RUN wget http://www.eu.apache.org/dist/flume/1.4.0/apache-flume-1.4.0-bin.tar.gz
RUN tar xvzf apache-flume-1.4.0-bin.tar.gz
```

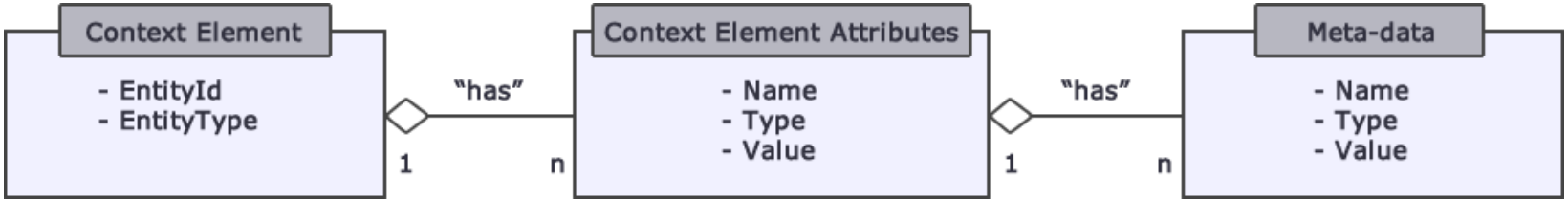
Dockerfile Cygnus (2 / 2)

```
ENV APACHE_FLUME_HOME /apache-flume-1.4.0-bin
RUN mkdir -p $APACHE_FLUME_HOME/plugins.d/cygnus/
RUN mkdir $APACHE_FLUME_HOME/plugins.d/cygnus/lib
RUN mkdir $APACHE_FLUME_HOME/plugins.d/cygnus/libext

#Install Cygnus
RUN git clone https://github.com/telefonicaid/fiware-connectors.git
RUN cd /fiware-connectors && git checkout release/0.5.1
RUN cd /fiware-connectors/flume && $APACHE_MAVEN_HOME/bin/mvn clean
compile exec:exec assembly:single
RUN cd /fiware-connectors/flume && cp target/cygnus-0.5.1-jar-with-
dependencies.jar $APACHE_FLUME_HOME/plugins.d/cygnus/lib
RUN wget http://repo1.maven.org/maven2/org/apache/thrift/libthrift/0.9.1/libthrift-
0.9.1.jar -o $APACHE_FLUME_HOME/lib/libthrift-0.9.1.jar
RUN rm $APACHE_FLUME_HOME/lib/libthrift-0.7.0.jar

#Configure Cygnus
ADD cygnus.conf $APACHE_FLUME_HOME/conf/cygnus.conf

#Launch MySQL and Cygnus on boot
CMD service mysqld start && $APACHE_FLUME_HOME/bin/flume-ng agent --conf
$APACHE_FLUME_HOME/conf -f $APACHE_FLUME_HOME/conf/cygnus.conf -n
cygnusagent -Dflume.root.logger=INFO,console
```



Una vez que todo está instalado y configurado, el trato con FIWARE se vuelve mucho más amable.

En FIWARE se trabaja con un modelo de datos llamado *Context*.



En nuestro caso:

Entity Id: ENXINE_DOMEDUINO_8DD4FA_0001
Entity Type: Sensor
Attribute Name: Power
Attribute Type: Watts

```
(curl  $ORION_AWS:1026/NGSI10/updateContext -s -S --header 'Content-Type:
application/json' --header 'Accept: application/json' -d @- | python -mjson.tool)
<<EOF
{
  "contextElements": [
    {
      "type": "Sensor",
      "isPattern": "false",
      "id": "ENXINE_DOMEDUINO_8DD4FA_0001",
      "attributes": [
        {
          "name": "Power",
          "type": "Watts",
          "value": "51.1"
        }
      ]
    }
  ],
  "updateAction": "UPDATE"
}
EOF
```

Cambiar un valor en el Context Broker

```
var payload = {
  'entities': [
    {
      'type': 'Sensor',
      'isPattern': 'false',
      'id': 'ENXINE_DOMEDUINO_8DD4FA_0001'
    }
  ]
};

$.ajax({
  type: 'POST',
  url: base_url() + 'NGSI10/queryContext',
  headers: {
    'Content-Type': 'application/json',
    'Accept': 'application/json'
  },
  dataType: 'json',
  data: JSON.stringify(payload),
  success: function(response){
    if (response.errorCode == undefined) {
      console.log('Potencia: ' +  response[0].contextElement.attributes[0].value + 'W');
    }
  }
});
```

Consultar un valor en el Context Broker

```
var payload = {
  'entities': [
    {
      'type': 'Sensor',
      'isPattern': 'false',
      'id': 'ENXINE_DOMEDUINO_8DD4FA_0001'
    }
  ],
  'reference': base_url() + '/callback/' + websocket_id,
  'duration': 'P1D',
  'notifyConditions': [
    {
      'type': 'ONCHANGE',
      'condValues': [ 'Power' ]
    }
  ],
  'throttling': 'PT2S'
};
```

Suscribirse al Context Broker

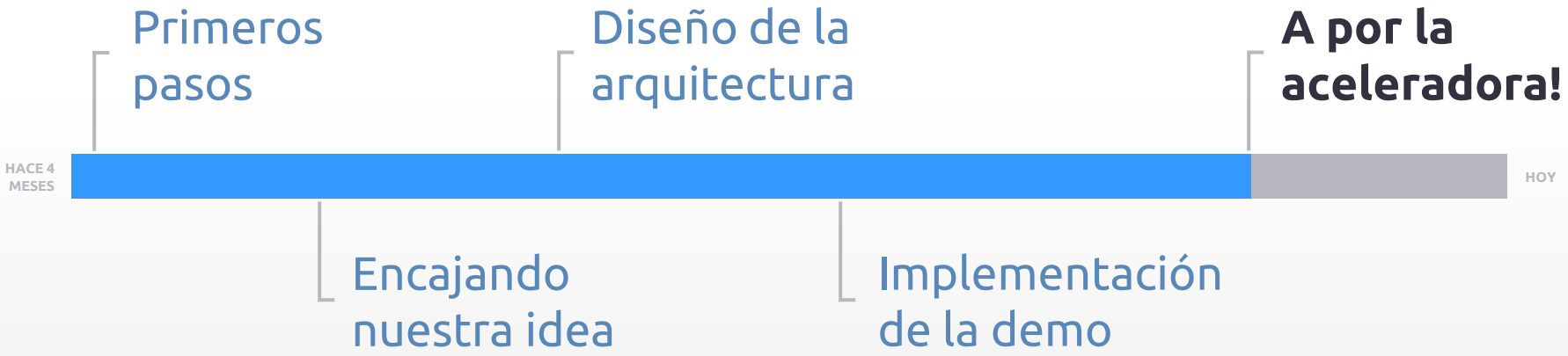
(1 / 2)

```
$.ajax({
  type: 'POST',
  url: base_url() + 'NGSI10/subscribeContext',
  headers: {
    'Content-Type': 'application/json',
    'Accept': 'application/json'
  },
  dataType: 'json',
  data: JSON.stringify(payload),
  success: function(response){
    if (response.errorCode == undefined) {
      console.log("Subscription ID: " + response.subscribeResponse.subscriptionId);
    }
  }
});
```

Suscribirse al Context Broker

(2 / 2)

Instalando FIWARE, un momento por favor...



Hemos presentado nuestra candidatura a una de las aceleradoras de Fi-PPP llamada INCENSe.

Es una aceleradora que ofrece hasta 150K euros de financiación (sin afectar a la *equity*) y un programa de mentoring de 6 meses.

Cumplimos con los requisitos:

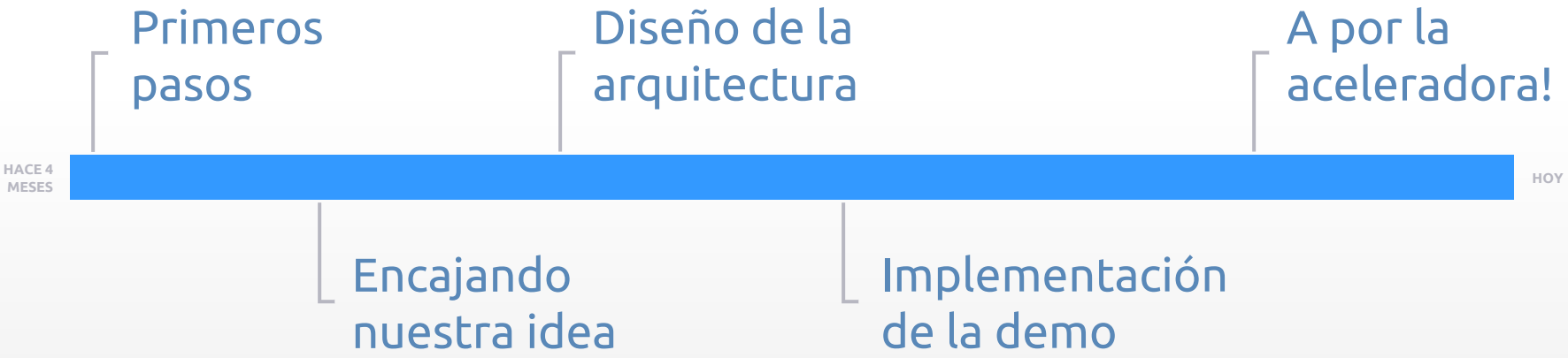
- Ajustarse a la temática: smart energy
- Utilizar la tecnología FIWARE
- Ser una start-up

Contamos con la ayuda de GAIN para supervisar nuestra solicitud.

Presentar la solicitud es un proceso que requiere dedicación:

- Crear un plan de negocio a 5 años para el proyecto que demuestre la rentabilidad de la idea.
- Presentar un equipo (multidisciplinado) con la capacidad y experiencia necesaria.
- Explicar qué tecnología FIWARE se va usar y cómo .
- Implementar una demo técnica donde se haga uso de FIWARE.
- Hacer la propuesta lo más atractiva posible: vídeo promocional, presentación, etc.

FIWARE ha sido instalado satisfactoriamente ✔



FIWARE es un punto de partida válido para cloud-computing (nos ha permitido hacer una demo en 3 semanas)

Tiene una barrera de entrada notable debido a su enfoque demasiado académico y falta de ejemplos reales.

Pero una vez en marcha, es sencillo de usar, potente y flexible.

Los programas de aceleración son una motivación muy importante.

¿Alguna pregunta?

ENXINE
www.enxine.com
info@enxine.com