

Emilio J Roldán Navarro

# Virtualización de servicios con NODE-RED

Indice

[Virtualización de servicios con NODE-RED 0](file:///E:\Virtualizacion\Virtualización%20Servicios%20Node-Red_v2.docx#_Toc509228433)

[¿Que es Node-red? 2](#_Toc509228434)

[Instalación 3](#_Toc509228435)

[1. Instalación de Node.js 3](#_Toc509228436)

[2. Instalación de Node-RED 3](#_Toc509228437)

[3 Ejecución de Node-Red 4](#_Toc509228438)

[4. INStalaciones Adicionales 5](#_Toc509228439)

[Ejemplos Virtualización ServiciOS 6](#_Toc509228440)

[Virtualización Servicio SOAP Lynx 6](#_Toc509228441)

[Virtualización Servicio SOAP Disputas Visa 10](#_Toc509228442)

[Virtualización de servicio API Rest 12](#_Toc509228443)

[Ejemplo ApI ResT con LLAmada a servico Original 13](#_Toc509228444)

[ejemplos obtención y representación Información 15](#_Toc509228445)

[Obtención Mesajes twitter 15](#_Toc509228446)

[Codigo ejemplos 16](#_Toc509228447)

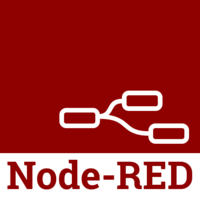
[Soap - Lynx 16](#_Toc509228448)

[API REST – Calculator 16](#_Toc509228449)

[SOAP – Visa Disputas 16](#_Toc509228450)

[Obtener/Visualizar Tweets 16](#_Toc509228451)

[Bibliografía 16](#_Toc509228452)



# ¿Que es Node-red?

Node-RED es un editor web basado en flujos desarrollado originalmente por IBM que permite simular la interconexión de dispositivos hardware, servicios de Api y servicios web que conforman lo que se conoce como el internet de las cosas (IoT).

Node-Red permite definir los flujos a través de protocolos estándar, tales como REST, MQTT, Websocket, AMQP, etc.

Se trata de una herramienta visual muy ligera construida sobe Node.js(1) lo que nos permite hacer uso de una arquitectura basada en eventos no bloqueante, y puede ser ejecutada sobre hardware low-cost como la Raspberry Pi o incluso en la nube.

Los nodos de Node-Red se construyen como paquetes npm(2) y pueden ser obtenidos como cualquier paquete nmp. El editor de Node.Red permite construir aplicaciones complejas haciendo uso de funciones javascript.

También se dispone de un API Rest de administración y operación (<http://nodered.org/docs/api/>) de manera que puede interactuar y ser controlado por un sistema externo.

***(1) Node JS***

Tal y como Wikipedia sugiere: “Node.js es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor (pero no limitándose a ello) basado en el lenguaje de programación ECMAScript, asíncrono, con I/O de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google.

***(2) NPM***

Npm es el gestor de paquetes de NodeJS. Según la definición que le dan sus creadores en su web oficial<https://docs.npmjs.com/getting-started/what-is-npm>

# Instalación

Node-RED pude instalarse en entornos Linux, Windows y MacOs, para instalar podemos seguir los pasos detallados en la página oficial de [Node-Red](https://nodered.org/docs/getting-started/installation)

Para la instalación en entornos Windows seguiremos los siguientes pasos:

## Instalación de Node.js

Descargar la última versión de node.js desde la página oficial de [Node.js](https://nodejs.org/en/) donde encontraremos la opción adecuada para nuestro sistema.

Una vez descargado el fichero MSI lo instalamos, para ello será necesarios permisos de administrador. Una vez instalado abre una nueva ventana de consola.

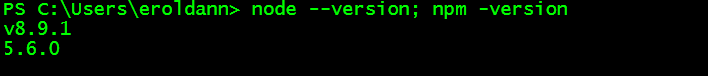
Para verificar que Node.js está correctamente instalado ejecutaremos los siguientes comandos

Using Powershell:

 node --version; npm -version

Using cmd:

 node --version && npm –version



## Instalación de Node-RED

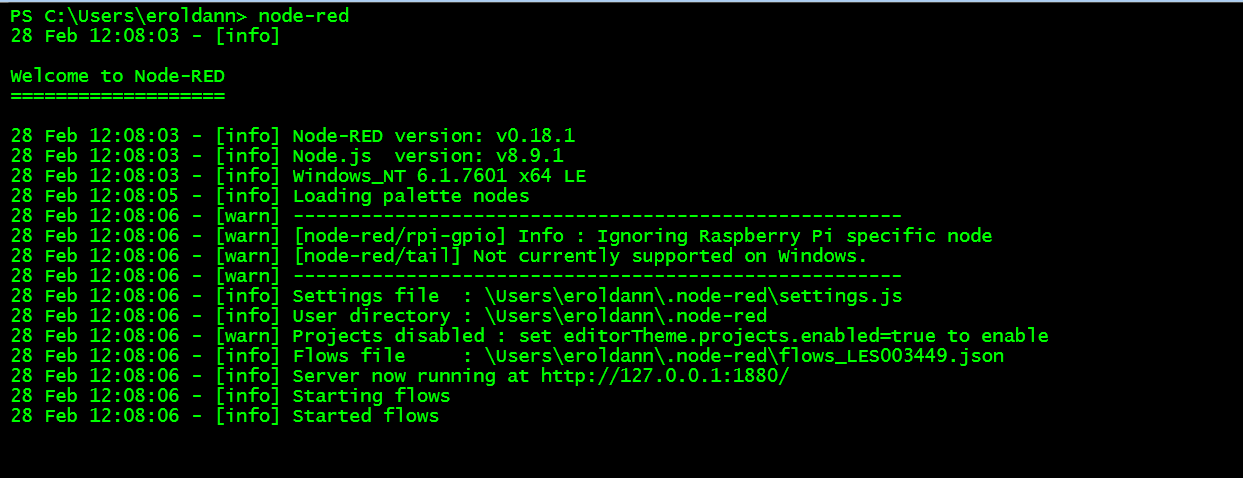
Para instalar Node-Red hacienda uso del gestor de paquetes npm ejecutamos en la consola:

node --version; npm install -g --unsafe-perm node-red

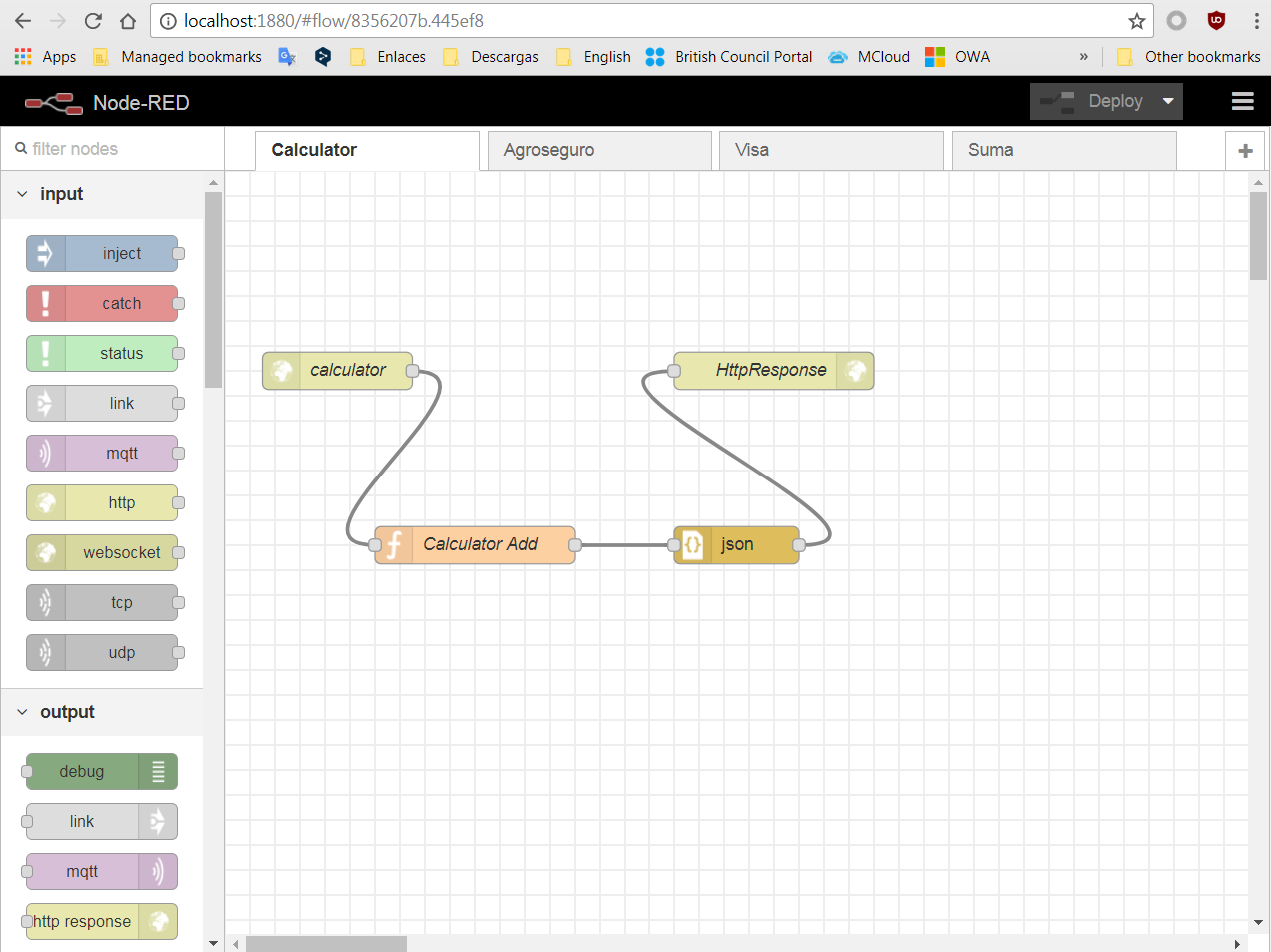
## 3 Ejecución de Node-Red

Para ejecutar lanzamos el siguiente comando en consola:

node-red



Una vez iniciado Node-Red podremos crear nuestro primer flujo accediendo a [http://localhost:1880](http://localhost:1880/)



## Instalaciones Adicionales

### Contenedor local

Para obtener un servidor de node-Red e un contenedor local ejecutamos el siguiente comando

$ docker run --name mynodered -d -p 1890:1880 nodered/node-red:[2.2.2](https://hub.docker.com/layers/node-red/nodered/node-red/2.2.2/images/sha256-259907078c3f861997b481115d9a00739e43d51ae6c21ce46a664336d6db31b2?context=explore)

-d: Indicamos que se ejecute como demonio

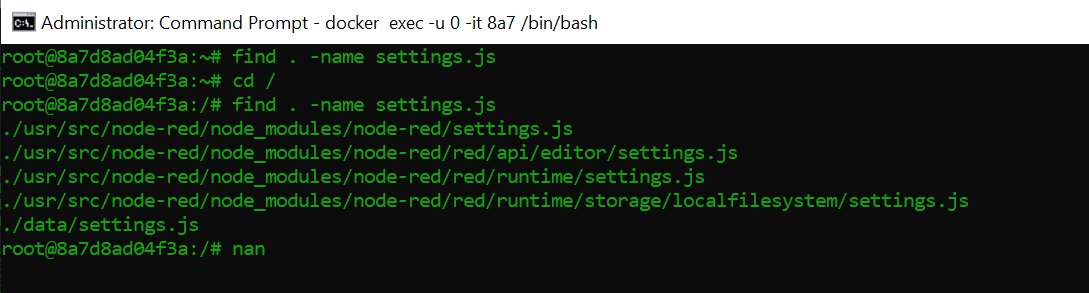
-p: cambiamos el puerto por defecto (1880) al 1890 para no interferir con instancias locales

Para configurar el fichero settings.js

docker exec -u 0 -it <id\_contenedor> /bin/bash

cd /

find . -name settings.js



Modificamos el fichero ***“./data/settings.js”*** para incluir la función require dentro de “functionGlobalContext”

functionGlobalContext: {

require:require

},

Instalación de paquetes xpath y xmldom

npm install -g xpath

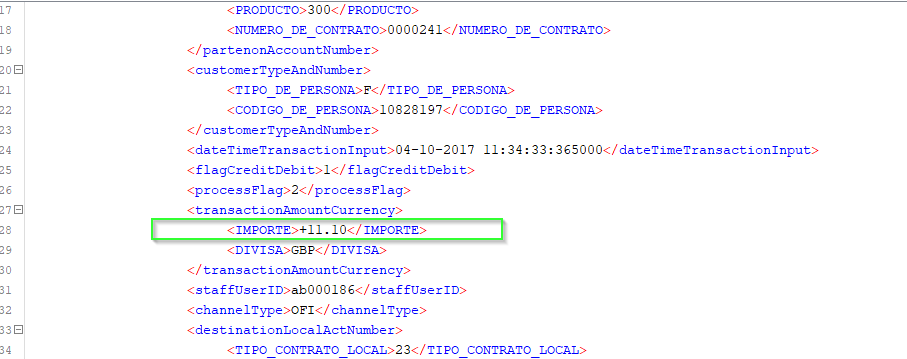
npm install -g xmldom

# Ejemplos Virtualización ServiciOS

## Virtualización Servicio SOAP Lynx

Lynx es un producto de detección de fraude en tiempo real que, en cuestión de milisegundos y de forma integral, estudia todo tipo de operaciones de tarjetas o cuentas bancarias operadas por cualquier canal: comercios, banca telefónica, banca móvil, banca por Internet, cajeros, compras online…, para determinar el riesgo de fraude de cada transacción.

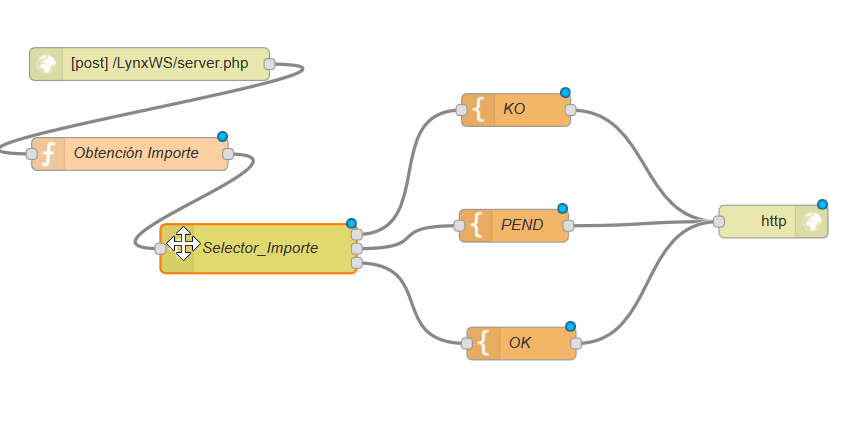
Para la virtualización capturaremos el valor de importe de la petición:



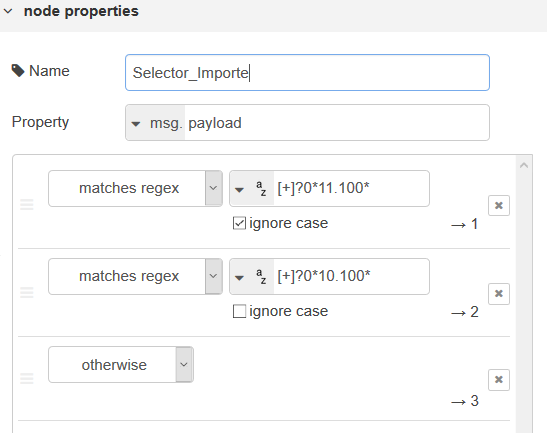
Y responderemos acorde a las siguientes reglas:

* KO: si el importe contiene 11.10 retornaremos un mensaje de KO
* PEND: si el importe contiene 10.10 retornaremos un mensaje de PENDIENTE(PE)
* OK: En cualquier otro caso retornaremos un mensaje de OK

Modelado en Node-Red



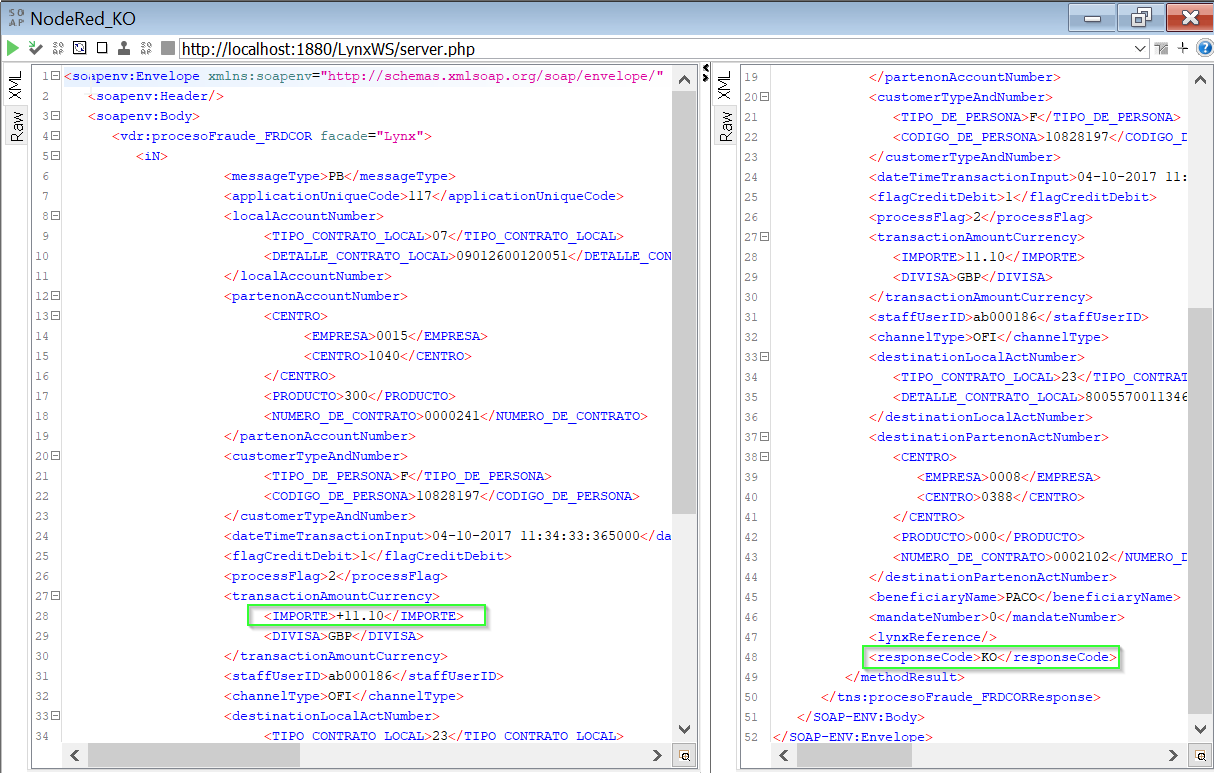
* ***Nodo entrada***: Responderá a peticiones POST sobre la ruta “/LynxWS/server.php”
* ***Nodo “obtención importe”:*** función javascript que se encarga de hacer una consulta Xpath para recuperar el valor del importe
* ***Nodo Selector,*** en función del valor de importe encaminará la salida en función de las reglas definidas:



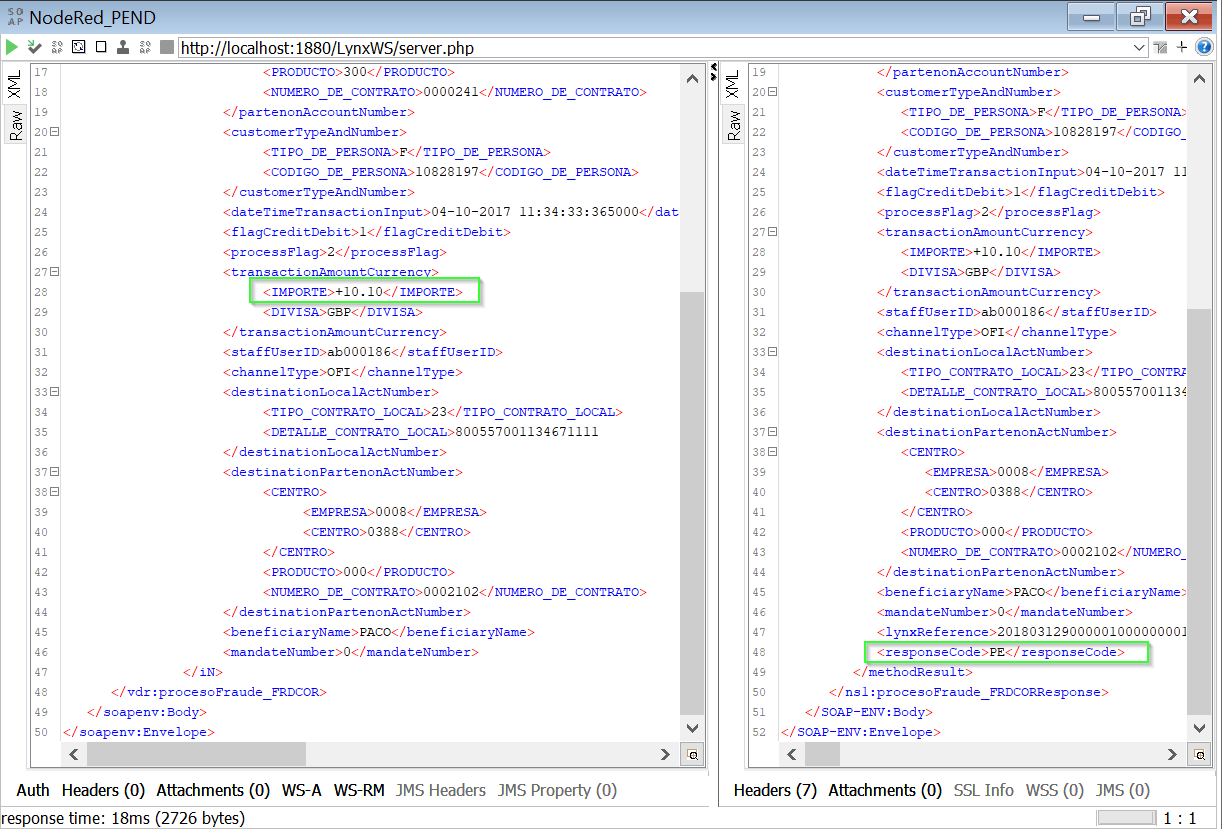
* ***Nodos plantilla para las respuestas OK, PEND y OK***
* ***Nodo de salida***

Peticiones y respuesta servicio virtualizado:

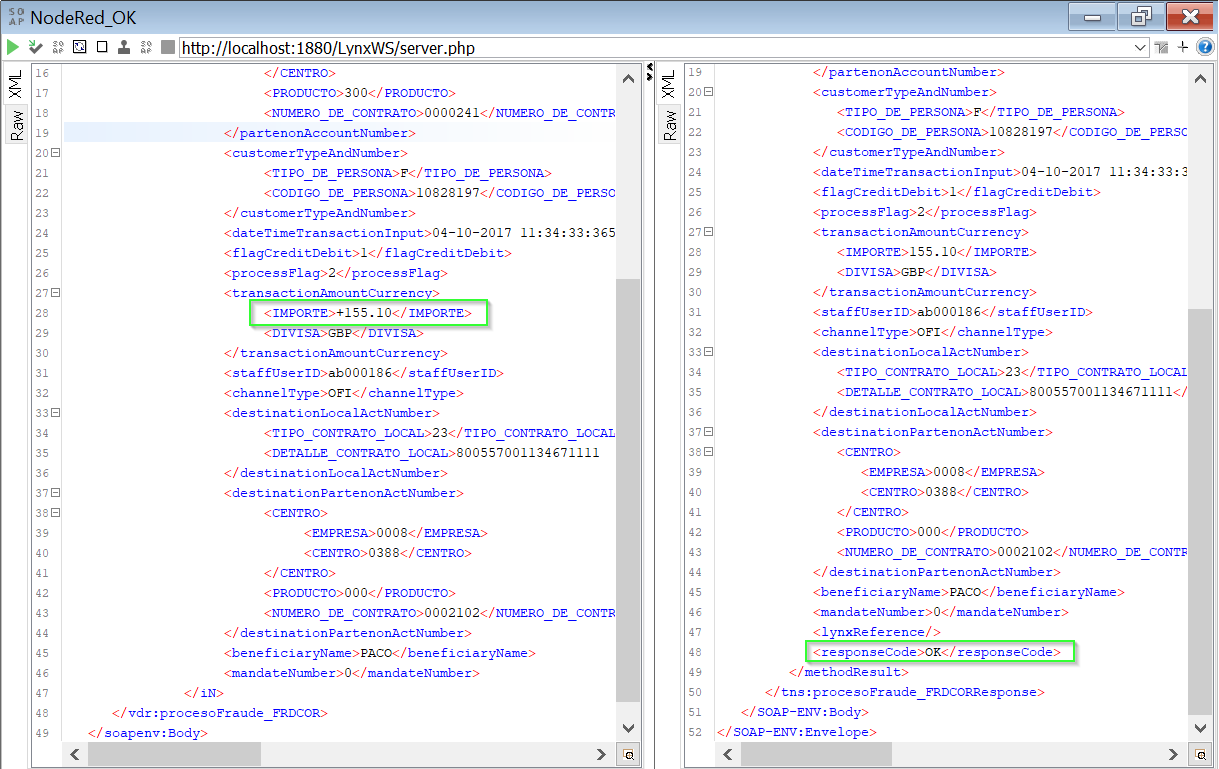
***KO***



***PE***



***OK***

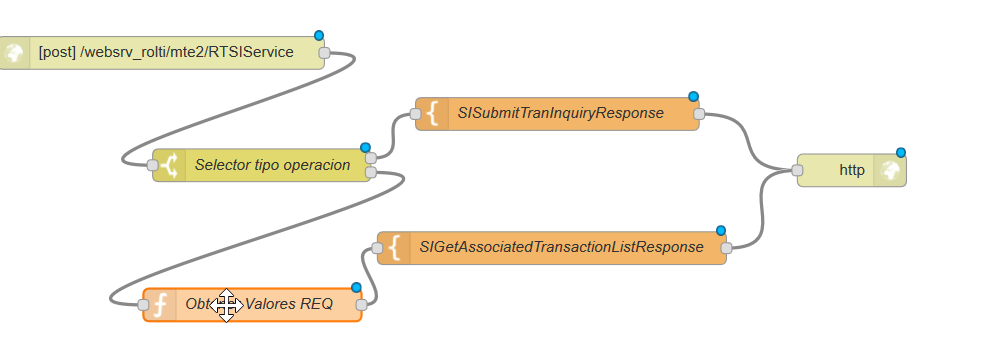


## Virtualización Servicio SOAP Disputas Visa

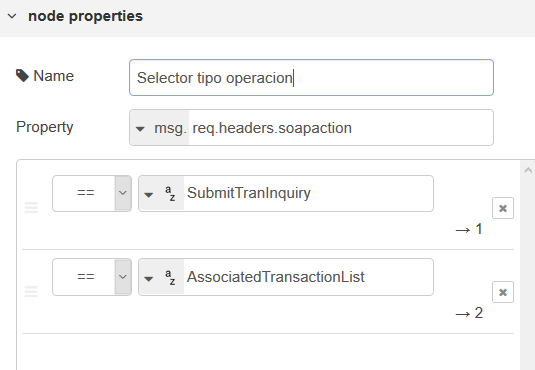
Visa expone un servicio web para la gestión online de disputas. Para ello ofrece un catálogo de operaciones sobre un servicio web para cubrir los flujos necesarios en la operativa de disputas.

Para este ejemplo de virtualización hemos elegido dos operaciones que tienen configurada la misma url de operación para ver como seleccionar distintas respuestas en función de la propiedad de cabecera “soapaction”

Modelado en Node-Red:



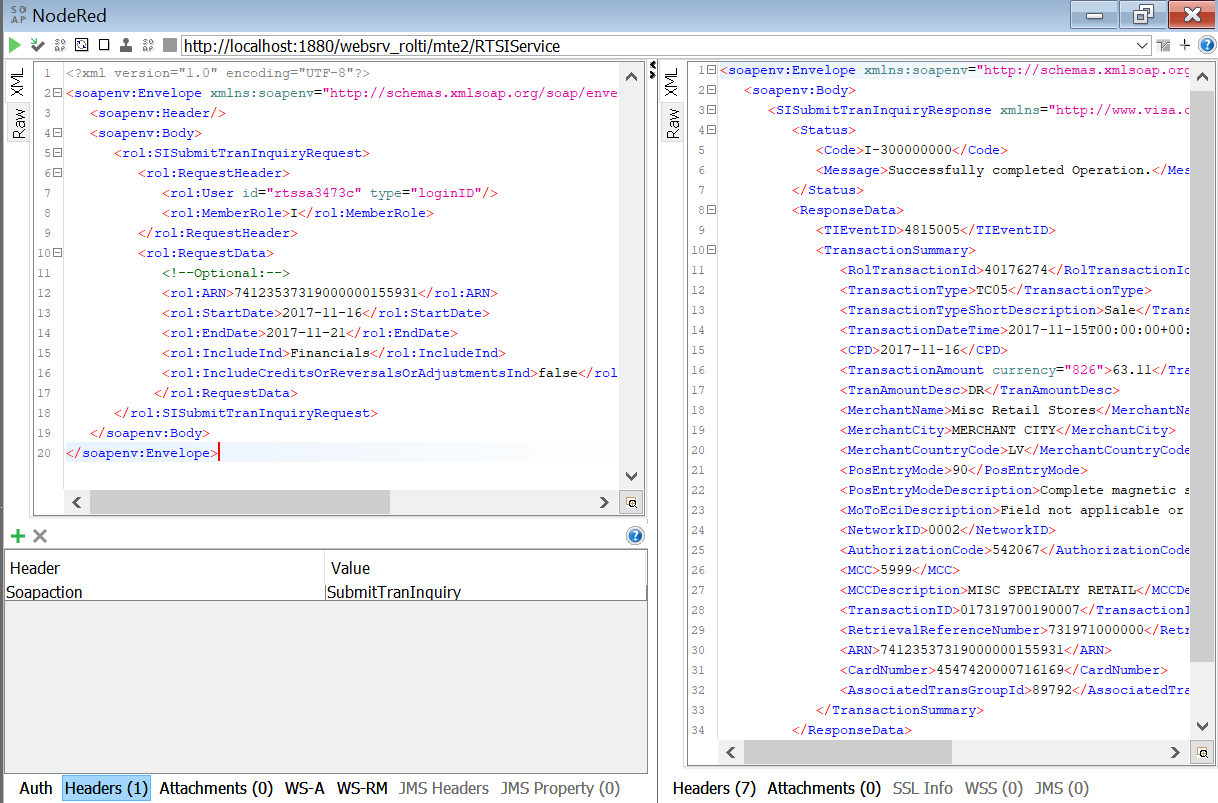
* ***Nodo entrada:*** capturara peticiones tipo POST sobre la ruta “/websrv\_rolti/mte2/RTSIService”
* ***Nodo ”Selector tipo Operación”:*** En función del tipo de operación configurada en la propiedad “soapaction” de la cabecera de la petición encaminara hacia la respuesta adecuada:



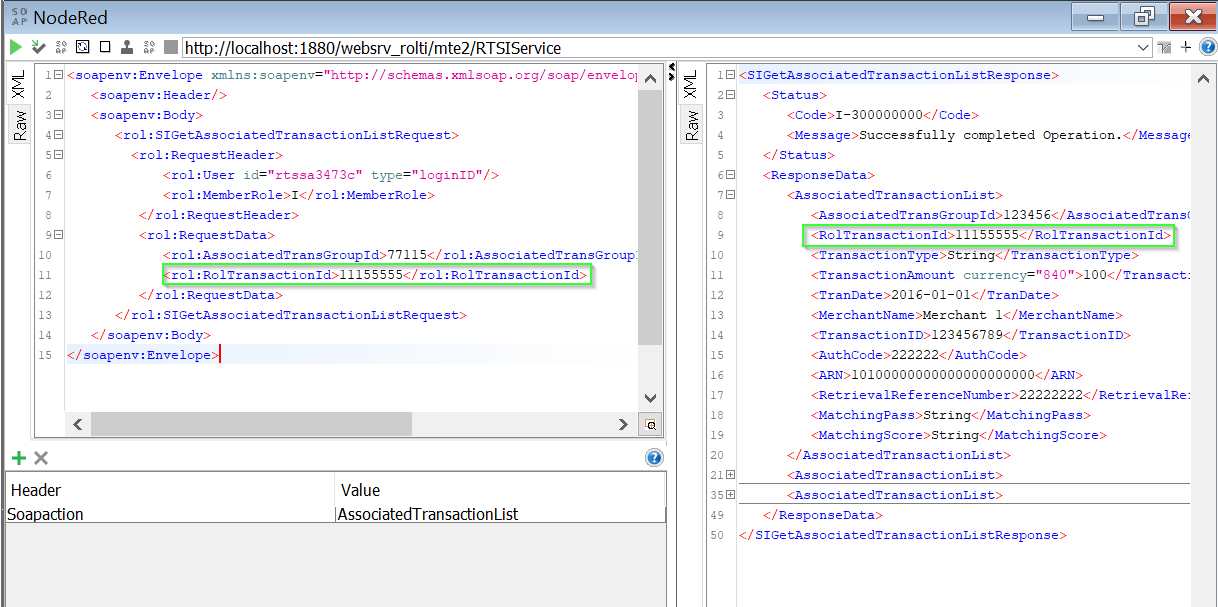
* Nodos plantilla para la respuesta a las operaciones ***“SubmitTransactionInquiry”*** y ***“GetAllTransDetails”***
* Nodo “Obtener valores REQ”: Para el tipo de operación ***“GetAllTransDetails”*** capturaremos el valor ***“RolTransactionId”*** de la petición y lo volcaremos en la respuesta de la operación.
* ***Nodo de salida***

Peticiones y respuesta servicio virtualizado:

***Operación SubmitTransactionInquiry***



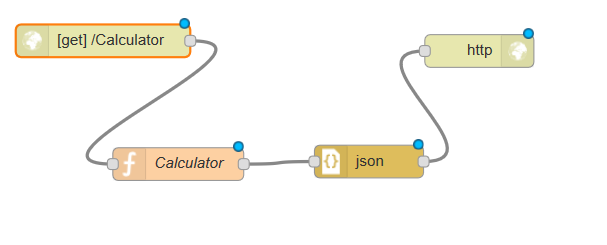
***Operación GetAllTransDetails***

******

## Virtualización de servicio API Rest

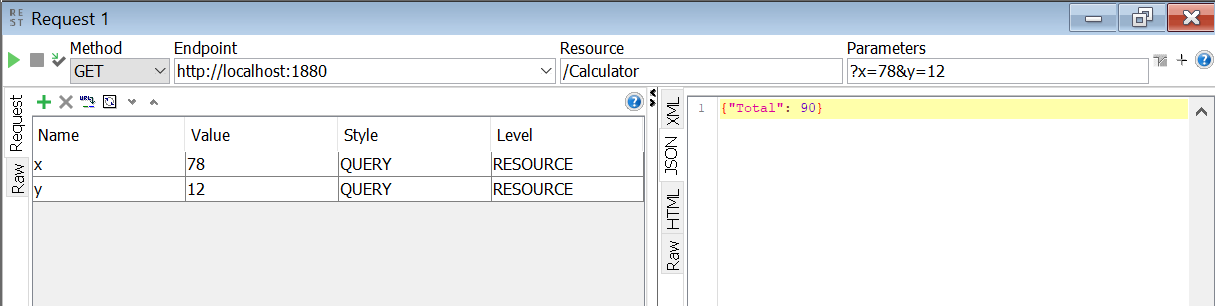
Para este ejemplo realizaremos la virtualización de un servicio de sumas que recibe los valores de suma a través de los parámetros de la url arg1 (x) y arg2 (y)

Modelado en node-Red:

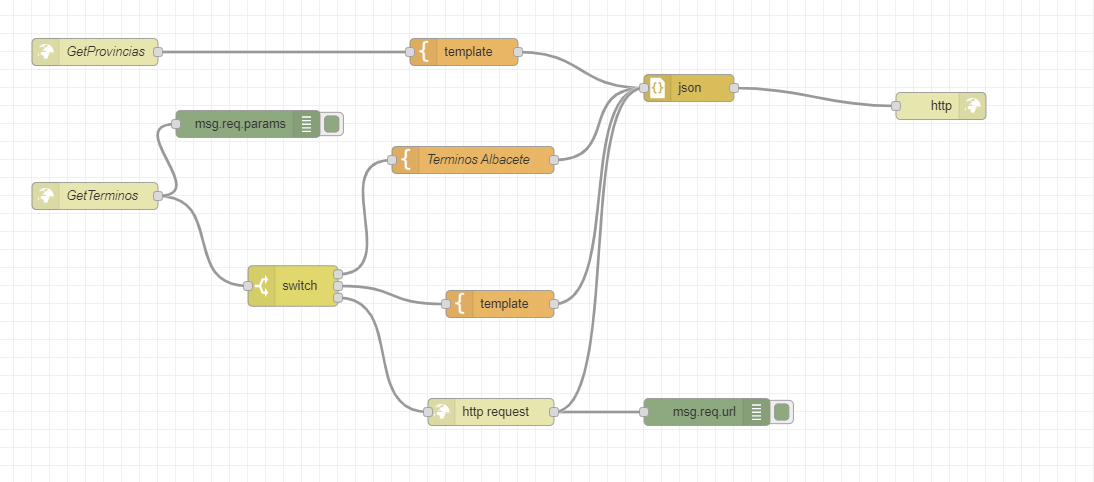


* ***Nodo entrada:*** Capturará peticiones tipo GET sobre la ruta “/Calculator”
* ***Nodo “Calculator”:*** recupera los valores pasados por el querystring, los parsea y los suma
* ***Nodo json:*** Convierte la salida del nodo anterior a formato JSON
* ***Nodo de salida***

Peticiones y respuesta servicio virtualizado:



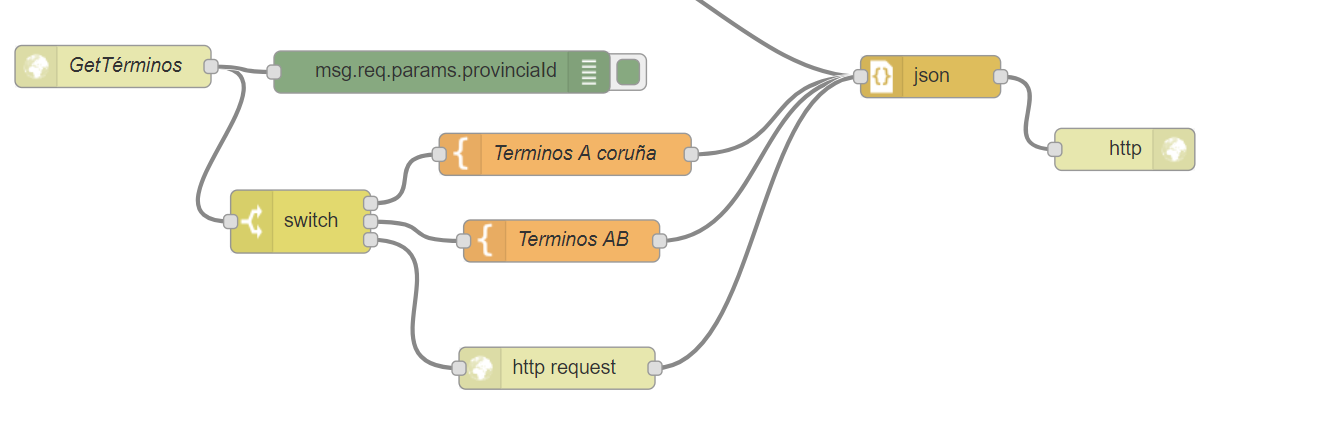
## Ejemplo API REST AGROSEGURO



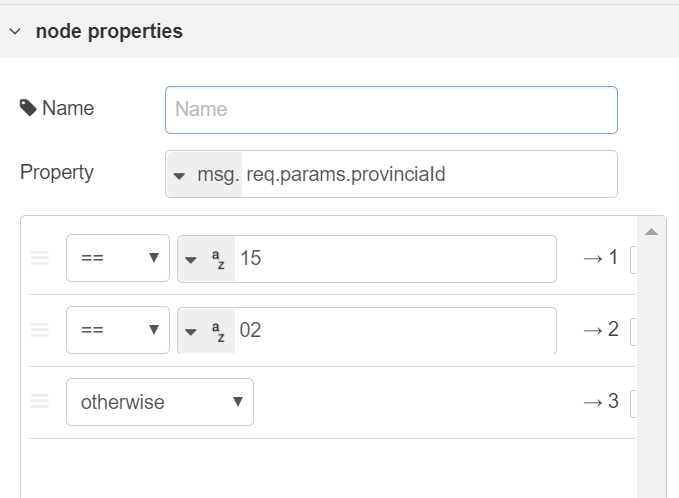
## Ejemplo ApI ResT con LLAmada aL servico Original

Para este ejemplo virtualizaremos un servicio API REST que invoca a un método de obtención de municipios por provincia, la respuesta para las llamadas para “A Coruña” (15) y “Albacete” (02), serán virtualizadas, mientras que el resto de llamadas se realizaran al servicio original.

Modelado en node-Red:



* ***Nodo entrada:*** Capturará peticiones tipo GET sobre la ruta ***“/swret/app/ambito/terminos/:provinciaId”,*** donde “:provinciaId” será el pametro de entrada donde se indicará el id de la provicia sobre la cual se quiere obtener la lista de municipios
* ***Nodo “switch”:*** recupera el valor del parámetro de entrada “proviciaId” y encaminara a la respuesta adecuada:



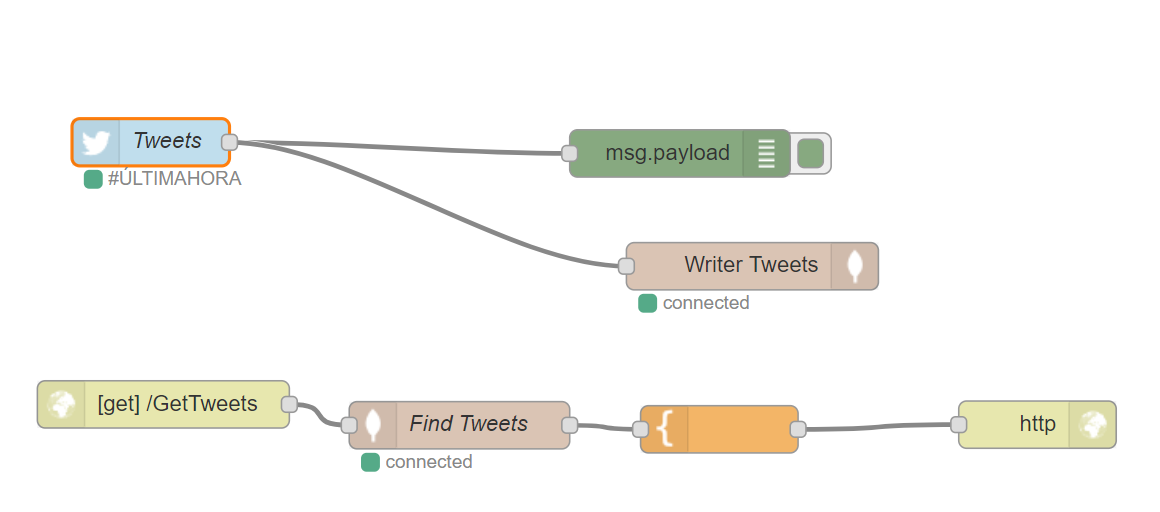
* ***Nodo json:*** Convierte la salida del nodo anterior a formato JSON
* ***Nodo de salida***

# ejemplos obtención y representación Información

## Obtención Mesajes twitter

Para este ejemplo obtendremos todos los mensajes asociados a un tweet, los almacenaremos en una base de datos MongoDb y crearemos un nodo de respuesta para visualizar los tweets capturados.

Modelado en node-Red:



Nodos Obtención Tweets

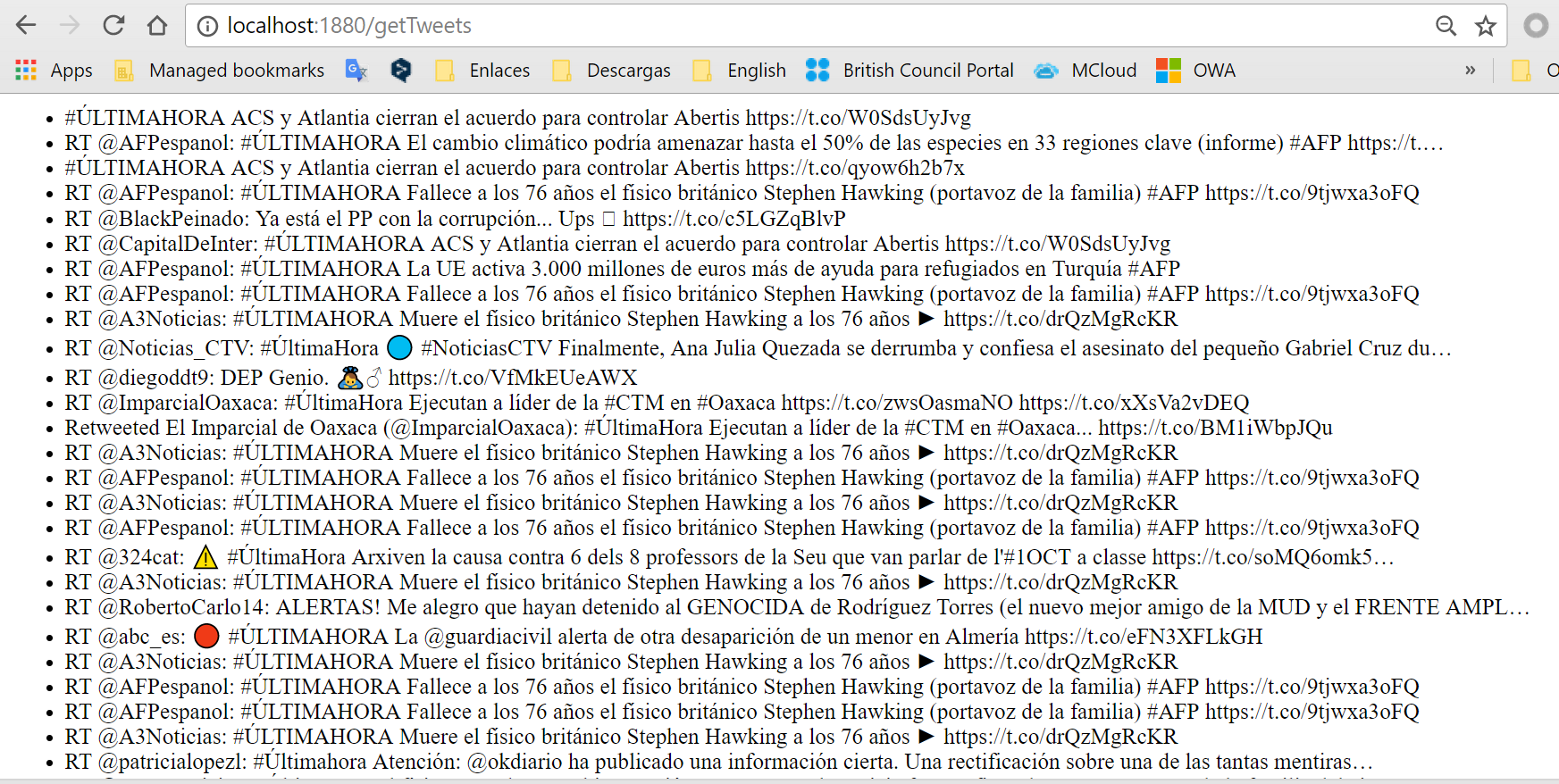
Nodos Visualización Tweets

Obtención de Tweets:

* ***Nodo entrada:*** Captura tweets bajo el hashtag “#ULTIMAHORA”
* ***Nodo “Writer Tweets”:*** Almacena los tweets capturados en MongoDb

Visualización de Tweets:

* ***Nodo entrada:*** captura peticiones tipo Get en la ruta “/GetTweets”
* ***Nodo “Find Tweets”:*** Obtiene los tweets de la base de datos MongoDb
* ***Nodo plantilla:*** Aplica una plantilla para la visualización, para mostrar los tweets de la siguiente forma:



# Ejemplos Internet of Things (IoT)

## Controlar bombilla phIlips hue

Para este ejemplo utilizaremos el netbest dashbord, para instalarlo podemos seguir los pasos que se detallan en este [post](https://opensource.com/life/16/5/netbeast)

# Codigo ejemplos

## Soap - Lynx



## API REST – Calculator



## SOAP – Visa Disputas



## Obtener/Visualizar Tweets



## Ejemplo API REST

# Bibliografía

**Node-red** <https://en.wikipedia.org/wiki/Node-RED>

**Lynx** <http://www.iic.uam.es/soluciones/banca/lynx/>

**Node.js** [**https://es.wikipedia.org/wiki/Node.js**](https://es.wikipedia.org/wiki/Node.js)

**Jsonata**