GENERALITAT VALENCIANA

IVACE INSTITUTO VALENCIANO DE COMPETITIVIDAD EMPRESARIAL

UNIÓN EUROPEA
Fondo Europeo de Desarrollo Regional
Una manera de hacer Europa

INESCOP
REDIT INNOVATION NETWORK

EXPEDIENTE	**IMDEEA/2018/54**
ACRÓNIMO	**SENSOCLOUD**
PROGRAMA	**Proyectos de I+D de carácter no económico realizados en cooperación con empresas**
TÍTULO DEL PROYECTO	**DESARROLLO DE ARQUITECTURAS IoT/BIG-DATA PARA LA MONITORIZACION DE MAQUINARIA E INTERCONEXION DE PERIFERICOS EN CALZADO PARA LA GESTION DE DATOS MASIVOS MEDIANTE CLOUD-COMPUTING**

Entregable E.3.1
ESPECIFICACIONES Y DESARROLLO DE ARQUITECTURA

ÍNDICE

1. Introducción

1.1. Objetivos del paquete 3

Este paquete se centrará en la realización del servidor que recoja los datos que proviene del modulo central que está conectado a los distintos sensores y su posterior representación. Para ello, nos centraremos en la definición de las especificaciones que debe tener el servidor que vamos a montar así como los requisitos del mínimos que debemos de tener para que pueda funcionar correctamente. De igual forma se definirá la arquitectura que poseerá el servidor para realizar las tareas que nos competen en el proyecto.

1.2. Objetivos del presente documento

En el presente entregable se describirá la arquitectura final que se ha utilizado para el desarrollo del servidor de representación de datos según los distintos datos que va a recibir. Así mismo, se describirán los requisitos mínimos que se deben de tener para que el servidor pueda funcionar correctamente.

2. Arquitectura del sistema

Con los datos obtenidos por los sensores, se ha desarrollado una arquitectura Cloud para la representación de datos. Cabe recordar por la imagen 1 la arquitectura completa del sistema, que parte de la sensorización de una empresa y termina en la representación en tiempo real en distintos dispositivos.

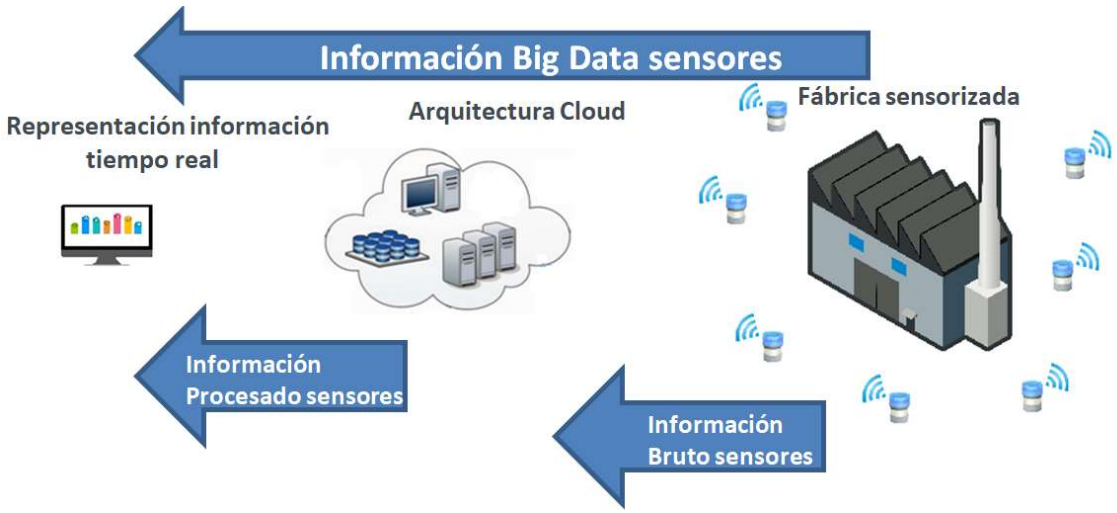

Imagen 1: Diagrama de la arquitectura completa del sistema

En este punto, nos centramos en la arquitectura cloud. Esta arquitectura viene determinada por los datos de los sensores y su posterior envío hacia la nube. Esta nube consiste en un servidor que es capaz de recibir, procesar, almacenar y representar los distintos datos capturados.

Para poder tener nuestra visualización de todos los datos recogidos por los distintos sensores, se ha utilizado un servidor para su almacenamiento, procesado y representación. Para ello y debido a que los datos que se van a almacenar son de gran relevancia para las empresas que utilicen este dispositivo, se ha optado por la implantación de un servidor local que disponga o no de acceso a internet, esta posibilidad se ha habilitado por si determinada empresa no quiere que sus datos sean accesibles fuera de sus instalaciones.
Después de analizar los requerimientos que necesita este sistema, se ha optado por un servidor hosting de altas prestaciones que nos ha permitido la realización de pruebas y validación de todo el sistema.
Este servidor esta en las propias instalaciones de Inescop y nos ha servido de gran ayuda a la hora de evaluar el sistema utilizado, puesto que al estar en las propias instalaciones, la configuración y diferentes modificaciones a las que se ha sometido el servidor se han desarrollado de una forma mucho más rápida y eficiente que utilizar otro tipo de servidor que dependa de alguna compañía de hosting que se encuentran actualmente ofreciendo sus servicios por Internet.

2.1. Arquitectura de captación y envío de datos

Para que los distintos datos se puedan representar, primeramente se han de capturar y enviar al servidor creado. En la imagen 2 se ve muy resumido esta captación y representación de información.

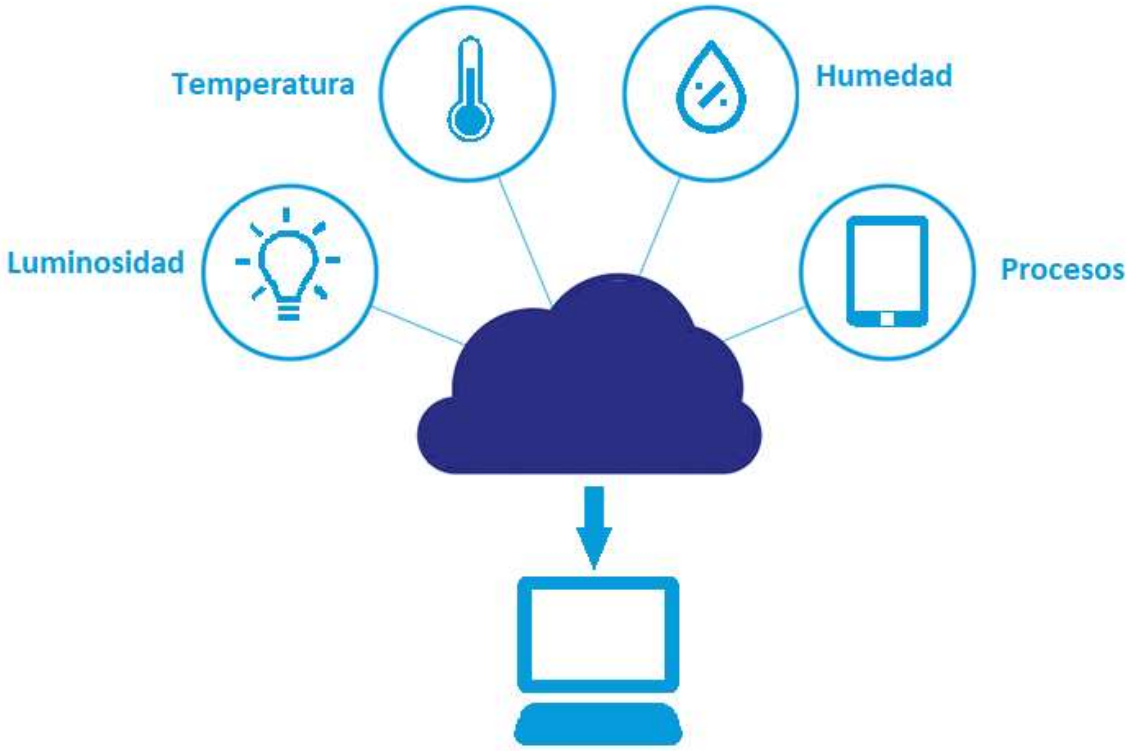

Imagen 2: Diagrama de la arquitectura

En el entregable 2.1 se ha explicado cómo se captura la información relevante que nos interesa para SENSOCLOUD por lo que ahora nos centraremos en que hacemos con esa información hasta su envío hacia la nube.

Como se ha comentado, la tarjeta de captura central está en contacto con los distintos sensores y es capaz de pedir y recibir información de ellos. Una vez que se ha realizado una petición de esa información, la información que se tiene en formato bruto y no se ha realizado ningún tratamiento de estos datos, se tiene que empaquetar de tal forma que se pueda enviar por medios inalámbricos.

En este punto del entregable nos centraremos tanto en el empaquetado de la información como en su conexión con medios inalámbricos para poder enviar la información.

Para que la información pueda enviarse, primeramente la tarjeta de procesamiento tiene que estar conectada a una red inalámbrica que tenga acceso al servidor de representación.
Para ello, esta tarjeta sigue un protocolo de conexión altamente utilizado y probado para que los fallos de conexionado sean mínimos. Para su conexión, se utiliza el método llamado 4-way handshake que consiste en lo siguiente:

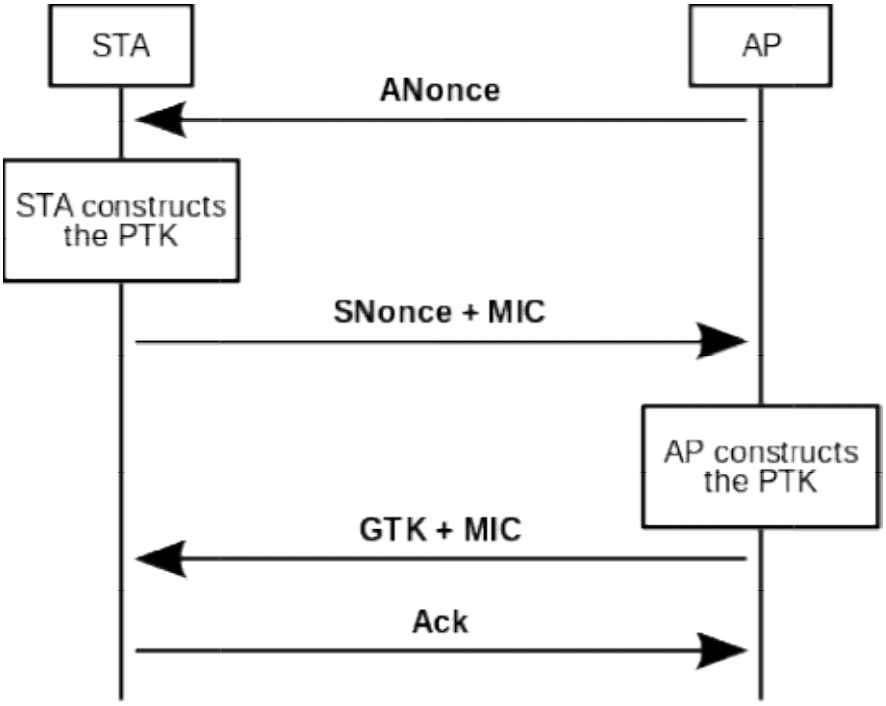

Imagen 3: Descripción de 4-way handshake

1. El AP (punto de acceso) envía un valor nonce al cliente. Este valor se llama **Anonce.** El cliente genera su propio nonce, llamado **Cnonce,** y ahora tiene todos los atributos para construir la PTK.
2. El cliente envía su propio **Cnonce** al AP junto con un código de integridad de mensaje, o MIC, que permite verificar la autenticidad. Es, en realidad, un **HMAC** (Hash message authentication code) o, específicamente aquí, MAIC (message authentication and integrity code).
3. El AP construye y envía al cliente la **GTK** o clave de grupo, y una secuencia de números acompañada de con otro MIC. Esta secuencia de números será utilizada en la siguiente trama broadcast o multicast, de modo que al recibirla, el cliente puede ejecutar una detección básica de replay.
4. El cliente envía finalmente una confirmación, o **acknowledge (ACK)** al AP, consiguiendo realizar la conexión.

(1)

Hay que tener en cuenta, que esta secuencia para su conexión se realiza ya que utilizamos un protocolo de seguridad WPA2, siendo actualmente el más robusto de los protocolos de seguridad que existen para el medio inalámbrico. El AP comentado en el proceso de conexión tiene que, lógicamente, que estar en la misma red que va a estar el servidor, sino, aunque se produzca una conexión correcta por parte de la tarjeta de procesado, la información nunca llegara al servidor de representación.

Una vez visto como se llega a conectar la tarjeta de procesado con la red que contiene el servidor, nos centraremos en cómo se empaqueta la información para que el servidor pueda recibirla y posteriormente procesarla.

El empaquetamiento se realiza en la propia tarjeta de adquisición de datos y su encapsulación sigue el protocolo WIFI, su formato es el siguiente:

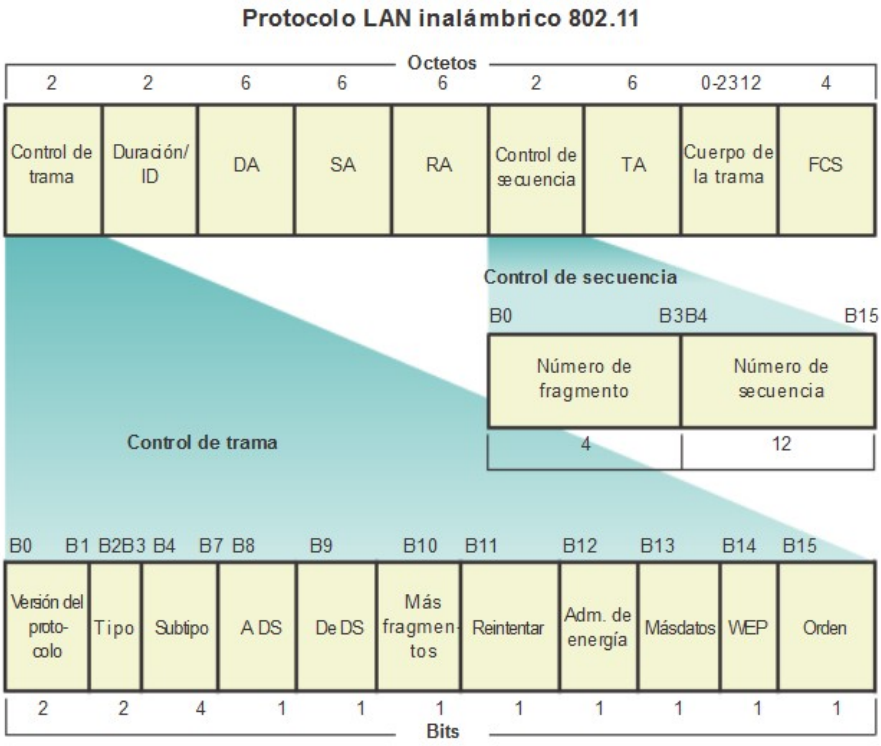

Imagen 4: Estructura de la trama enviada

La definición de cada uno de los apartados que consta la trama es el siguiente:

- Campo Versión de protocolo: la versión de la trama 802.11 en uso.
- Campos Tipo y Subtipo: identifican una de las tres funciones y subfunciones de la trama (control, datos y administración).
- Campo A DS: se establece en 1 para las tramas de datos destinadas al sistema de distribución (dispositivos en la estructura inalámbrica).

- Campo Desde DS: se establece en 1 para las tramas de datos que salen del sistema de distribución.

- Campo Más fragmentos: se establece en 1 para las tramas que tienen otro fragmento.

- Campo Reintentar: se establece en 1 si la trama es una retransmisión de una trama anterior.

- Campo Administración de energía: se establece en 1 para indicar que un nodo estará en el modo de ahorro de energía.

- Campo Más datos: se establece en 1 para indicarle a un nodo en el modo de ahorro de energía que se almacenan más tramas en búfer para ese nodo.

- Campo Privacidad equivalente por cable (WEP): se establece en 1 si la trama contiene información encriptada mediante WEP para propósitos de seguridad

- Campo Orden: se establece en 1 en una trama de tipo de datos que utiliza la clase de servicio Estrictamente ordenada (no requiere reordenamiento).

- Campo Duración/ID: según el tipo de trama, representa el tiempo que se requiere en microsegundos para transmitir la trama o una identidad de asociación (AID) para la estación que transmitió la trama.

- Campo Dirección de destino (DA): contiene la dirección MAC del nodo de destino final en la red.

- Campo Dirección de origen (SA): contiene la dirección MAC del nodo que inició la trama.

- Campo Dirección del receptor (RA): contiene la dirección MAC que identifica al dispositivo inalámbrico que es el destinatario inmediato de la trama.

- Campo Número de fragmento: indica el número de cada fragmento de la trama.

- Campo Número de secuencia: indica el número de secuencia asignado a la trama. Las tramas retransmitidas se identifican con números de secuencia duplicados.

- Campo Dirección del transmisor (TA): contiene la dirección MAC que identifica al dispositivo inalámbrico que transmitió la trama.

- Campo Cuerpo de la trama: contiene la información que se transporta. En las tramas de datos; generalmente se trata de un paquete IP.

- Campo FCS: contiene una comprobación de redundancia cíclica (CRC) de 32 bits de la trama.

(2)

En el campo cuerpo de trama se situará la información que vamos a enviar al servidor y que tiene que ser entendible por él. De igual forma que el empaquetado con el que se envía la trama, nosotros hemos diseñado nuestro propio sistema de codificación de información también encapsulado.
Dependiendo del sensor que envíe la información, este campo tendrá una serie distintas de información. Esta información determinada por el tipo de sensor es el siguiente:

POST /projects/Mes/web/api/sensors HTTP/1.1\r\nContent-Length: 21\r\nHost: localhost:8888\r\nContent-Type: application/json\r\n\r\n{\"t\":\"0\"\",\"v\":\"\"%.02f\"\"\"}\r\n

POST /projects/Mes/web/api/sensors HTTP/1.1\r\nContent-Length: 21\r\nHost: localhost:8888\r\nContent-Type: application/json\r\n\r\n{\"t\":\"1\"\",\"v\":\"\"%.02f\"\"\"}\r\n

POST /projects/Mes/web/api/sensors HTTP/1.1\r\nContent-Length: 23\r\nHost: localhost:8888\r\nContent-Type: application/json\r\n\r\n{\"t\":\"3\"\",\"v\":\"\"%.02f\"\"\"}\r\n

Donde %.02f corresponde al valor captado por el sensor de temperatura, humedad y luminosidad.
Aunque la trama es muy parecida, si que aparecen distinción entre ellas, posibilitando al servidor saber de qué sensor proviene.

El proceso para que llegue la información de tiempos de producción que llega desde la tablet es parecido al proceso que se acaba de explicar. Su conexión a la red es idéntico a como lo realiza la tarjeta de procesado, pero el formato en el que se manda la información si que difiera un poco de cómo se envía la información de ella, puesto que en este caso hay que mandar información relacionada (tiempos de comienzo y de fin) y hay que hacer pequeñas modificaciones a como se envía hasta el momento. Aunque la comunicación con el servidor se explicará detenidamente en el siguiente punto, la tablet envía la información en este formato.

POST /projects/Mes/web/api/sensors HTTP/1.1\r\nContent-Length: x\r\nHost: localhost:8888\r\nContent-Type: application/json\r\n\r\n{\"t\":\"2\"\",\"n\":\"\"0\"\"}\r\n

2.2. Arquitectura de procesado de datos

Hemos visto como la información se genera y como se envía hacia el servidor siguiendo ciertos protocolos estándar que se utilizan constantemente en la industria y nuestro propio protocolo de envío de datos para su análisis. Ahora nos centramos en la recepción de esos datos por parte del servidor y su posterior procesamiento y representación.

La arquitectura en la nube se ha modelado como un sistema distribuido, en el que varios clientes, como pueden ser el módulo central sensorizado, el tablet de registro de procesos, o la web de monitorización de los diferentes datos recogidos, se conectan a un servidor centralizado con el objetivo de almacenar y obtener la información almacenada en él. La parte del servidor, también conocida como back-end del sistema distribuido, se ha implementado y desplegado sobre un servidor web Apache. La estructura lógica de esta parte de la arquitectura puede observarse en el siguiente diagrama:

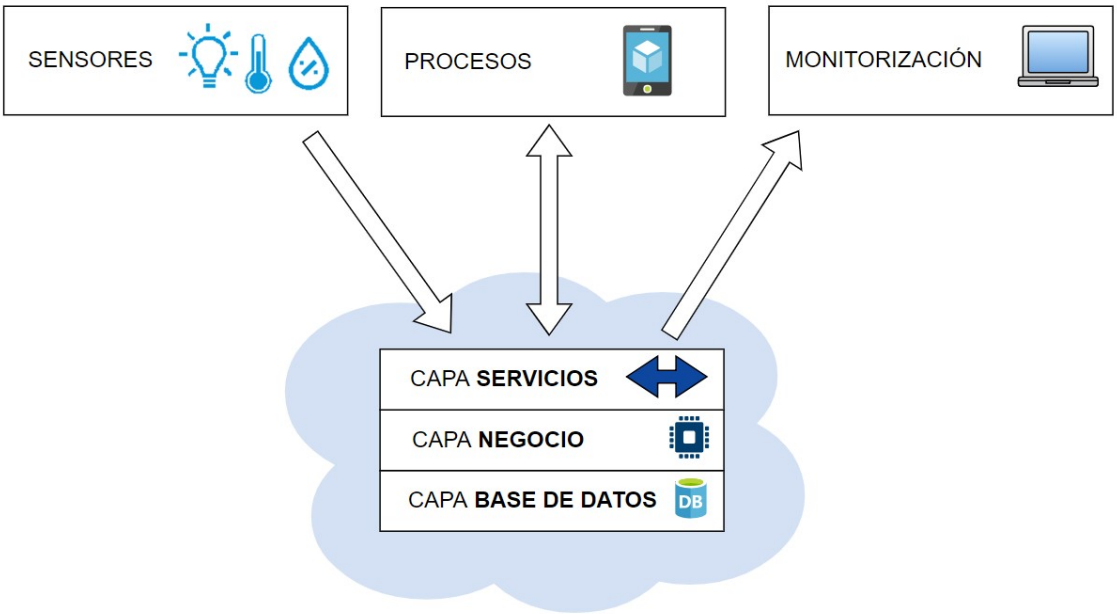

Imagen 5: Estructura lógica del back-end o parte del servidor de la arquitectura.

Capa base de datos.

Esta es la capa que se encarga de la persistencia de datos, es decir, de *almacenar* y *recuperar* la información comunicada y procesada a través del sistema. La estructura lógica de la base de datos planteada para el sistema de prototipo es sencilla ya que únicamente consta de una tabla en la que se va registrando toda la información registrada. No obstante, para un sistema de mayor complejidad, esta base de datos puede dividirse en más tablas siguiendo el modelo entidad-relación. La tabla utilizada en la base de datos es la siguiente:

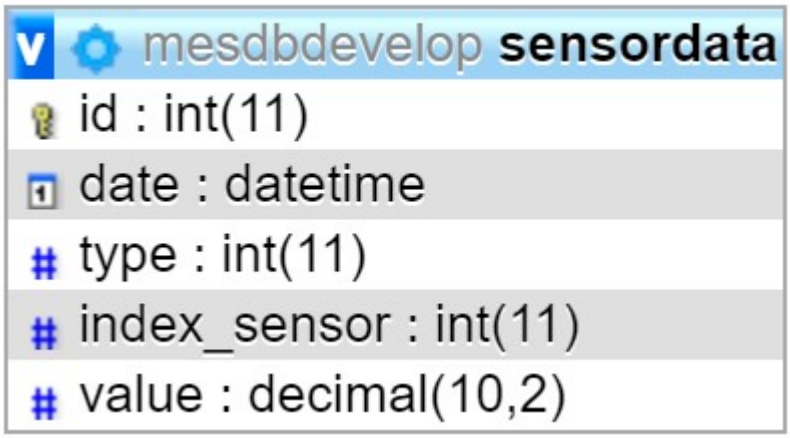

Imagen 6: Estructura de la tabla a través de la cual se almacena la información.

Cada entrada en la tabla supone una unidad de información registrada, para la cual es conveniente conocer el momento en el que se produjo ese registro, el número de sensor de origen, el tipo de sensor y su valor. La información detallada de los diferentes campos que esta tabla contiene son:

- **id:** Valor entero correspondiente al identificador del dato registrado. Este identificador podrá ser utilizado para modificar a posteriori el valor de dicho dato. Este campo ejerce de clave primaria de la tabla.

- **date:** Fecha y hora en la que fue registrado el dato en cuestión. Valor entero correspondiente al identificador del dato registrado. Este identificador podrá ser utilizado para modificar a posteriori el valor de dicho dato. Campo obligatorio

- **type:** Valor entero que representa el tipo de dato registrado. Campo obligatorio. Los posibles valores registrables en el prototipo desarrollado son los siguientes:
  - *Humedad: 0*
  - *Luminosidad: 1*
  - *Proceso: 2*
  - *Temperatura: 3*

- **index_sensor:** Índice del sensor del que proviene la información en el caso de que se utilicen varios sensores del mismo tipo. Campo opcional

- **value**: Valor que se obtiene del sensor. Campo opcional, ya que este valor se puede establecer posteriormente. En función del tipo de sensor la información se almacena en una unidad determinada:
  - *Humedad:* Porcentaje de humedad relativa.
  - *Luminosidad:* Lúmenes

- *Proceso:* Segundos
- *Temperatura:* Grados Celsius.

Capa de negocios:

Esta capa contiene la lógica de la aplicación, es decir, se encarga de realizar las operaciones a las cuales se acceden a través de la capa de servicios. A través de la capa de negocios se *procesa* la información y se manipula para el correcto almacenamiento y recuperación en la base de datos.

Capa de servicios:

La capa de servicios proporciona las interfaces con los distintos clientes, a través de rutas URL a la cual los clientes conectan, y que suponen la puerta de entrada a una serie de operaciones que se realizan en el servidor. A través de la capa de servicios se *comunica* la información.

El sistema de comunicación utilizado se ha planteado siguiendo el paradigma proporcionado por los servicios web *RESTFul*. Los servicios web son puertas de entrada al intercambio de información entre máquinas, y al conjunto de servicios que proporciona una aplicación web se le denomina *API*. La principal particularidad de los servicios web de tipo *RESTFul* es su simpleza, facilidad de manipulación y escalabilidad, ya que se basan en los principales métodos HTTP como GET, POST, DELETE, etc.

Se ha definido por tanto un API que permite el acceso a las principales funcionalidades que la arquitectura proporciona. El lenguaje mediante el cual se comunica el contenido de los mensajes es JSON. El valor del código HTTP devuelto por el servidor web indica si la llamada se ha llevado a cabo de manera correcta, o por el contrario, el motivo del error acontecido. La estructura de los servicios web implementados es la siguiente:

- **WS_1: RegisterSensor:**
  Ruta URI: /api/sensors
  Método HTTP: POST
  Descripción: Mediante este servicio se registran los datos recogidos de un sensor o del seguimiento de procesos. La llamada al servicio retorna el nuevo identificador de la unidad de información insertada en la base de datos. Con respecto al seguimiento de procesos, el valor de la tarea (en este caso, la duración) queda vacío en el inicio, puesto que se desconoce cuánto va a durar su ejecución, quedado registrado únicamente la fecha y hora de comienzo en el momento en el que se produce la llamada al servicio.
  JSON de entrada:

```
{
        "t":"value_type",
        "n":"index_sensor",
        "v":"value_sensor"
```

```
}
```

JSON de salida:

```
{
        "id":"database_entry_id"
}
```

- **WS_2: UpdateSensor:**
  Ruta URI: /api/sensors/value
  Método HTTP: POST
  Descripción: Mediante este servicio puede modificarse el valor de una entrada en la tabla de registro de sensores que ha sido insertada previamente, aunque en la práctica, este servicio únicamente se utiliza para actualizar la información de la duración de un proceso que ha sido insertado previamente en la base de datos. La información de entrada que utiliza es el identificador de la entrada en la tabla, y el valor a almacenar. No se retorna ninguna información de salida debido a que ya se cuenta con el valor del identificador de la entrada.
  JSON de entrada:

```
{
        "id":"database_entry_id",
        "v":"value_sensor"
}
```

3. Especificaciones del equipo

Para poder tener nuestra visualización de todos los datos recogidos por los distintos sensores, se ha utilizado un servidor para su almacenamiento, procesado y representación. Para ello y debido a que los datos que se van a almacenar son de gran relevancia para las empresas que utilicen este dispositivo, se ha optado por la implantación de un servidor local que disponga o no de acceso a internet, esta posibilidad se ha habilitado por si determinada empresa no quiere que sus datos sean accesibles fuera de sus instalaciones. Después de analizar los requerimientos que necesita este sistema, se ha optado por un servidor hosting de altas prestaciones que nos ha permitido la realización de pruebas y validación de todo el sistema. Este servidor esta en las propias instalaciones de Inescop y nos ha servido de gran ayuda a la hora de evaluar el sistema utilizado, puesto que al estar en las propias instalaciones, la configuración y diferentes modificaciones a las que se ha sometido el servidor se han desarrollado de una forma mucho más rápida y eficiente que utilizar otro tipo de servidor que dependa de alguna compañía de hosting que se encuentran actualmente ofreciendo sus servicios por Internet.

3.1. Especificaciones hardware

Si nos fijamos en la parte hardware del equipo, se ha elegido un equipo con las siguientes características:

- Modelo→ Servidor hosting Fujitsu NEU10960866 PY RX2530 M2

- Controladora →1 Controladora RAID PRAID CP400i
- Procesador → INTEL XEON E5-1640V4 10C/20T 2.40 GHz
- Almacenamiento →2 Discos duros rígidos SAS 12G 1.8TB 10K 512e HOT PL 2.5'EP
- Memoria →4 Módulos de 32GB DDR4-2400

Como se puede observar, las prestaciones de este servidor son muy elevadas, pero se han optado por ella por la experiencia de usuario, ya que permitirá al usuario acceder a los datos de una forma muy rápida para su visualización. También hay que tener en cuenta, que el número de datos que se vayan a guardar va a ir creciendo conforme el sistema se ponga en funcionamiento y, por como se ha diseñado el sistema, se podrán incluir en el sistema múltiples sensores que actualmente no están implementados o aumentar los que hay a un número mucho mayor ya que se quieren analizar distintas partes de una empresa, aumentando considerablemente el número de datos que se van a analizar, procesar, guardar y representar. De igual forma, y al tratarse de un servidor que se encuentra en local, nos va a permitir poder actualizar y mejorar este servidor si llegara el caso de que por el gran número de datos a tratar sus prestaciones no fueran suficientes para el desarrollo de la función a la que está destinado.

3.2. Especificaciones software

En la parte software del servidor, se ha utilizado un sistema operativo libre como es Linux, que nos ofrece una mayor estabilidad y seguridad frente a otros sistemas operativos actuales. Las características de este sistema operativo son las siguientes:

- Estabilidad: Linux es el sistema operativo para servidores más estable. Es capaz de manejar cantidades muy elevadas tanto de datos como de procesos de una manera muy fluida y sin presentar fallos debido a sobrecargas.
- Flexibilidad: Linux incluye herramientas nativas para optimizar el uso de todos los recursos, pudiéndose adaptar de forma más precisa a las necesidades de cualquier empresa.
- Actualización: Linux es un sistema operativo que se actualiza constantemente con parques de seguridad y estabilidad.
- Seguridad: Al ser Linux un de código abierto, pudiendo cualquier usuario ver fallo de seguridad y corregirlos de forma rápida. Ello conlleva una actualización que beneficia a todos los usuarios de este sistema operativo.

4. Conclusiones

Como se ha podido ver en el entregable, tanto la comunicación como la recepción del servidor se han delimitado y optimizado perfectamente para un uso eficiente tanto de la transmisión como del procesado y representación de los datos. Por esta razón se puede confirmar el éxito obtenido a la hora de definición del envío de la información y de su posterior análisis y representación.

5. Referencias

(1) https://juncotic.com/wpa2-como-funciona-algoritmo-wifi/
(2) http://www.itesa.edu.mx/netacad/introduccion/course/module4/4.4.4.8/4.4.4.8.html