

UNIVERSIDAD DE BUENOS AIRES  
FACULTAD DE INGENIERÍA

TRABAJO PROFESIONAL DE GRADO  
DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA

---

## **Plataforma de telemetría IoT aplicada al estudio de precursores sísmicos**

---

*Alumnos:*

Belgrano, Mateo - 96208  
Luraschi, Sebastian - 97177

*Tutores:*

Marino Belcaguy, Pablo Carlos (Tutor)  
Alonso, Ramiro (Co-tutor)

*Director de Departamento:*

Belaustegui Goitia, Carlos Fernando

23 de agosto de 2021

# Índice

<b>Índice de figuras</b>	<b>6</b>
<b>Índice de tablas</b>	<b>8</b>
<b>0 Estructura del Trabajo</b>	<b>9</b>
0.1 Capítulo 1 - Introducción . . . . .	10
0.2 Capítulo 2 - Objetivos . . . . .	10
0.3 Capítulo 3 - Definición del producto . . . . .	10
0.4 Capítulo 4 - Análisis de factibilidad temporal . . . . .	10
0.5 Capítulo 5 - Análisis de factibilidad económica . . . . .	10
0.6 Capítulo 6 - Análisis de factibilidad técnica . . . . .	11
0.7 Capítulo 7 - Ingeniería en detalle . . . . .	11
0.8 Capítulo 8 - Módulos complementarios . . . . .	11
0.9 Capítulo 9 - Construcción del prototipo . . . . .	11
0.10 Capítulo 10 - Conclusiones . . . . .	12
<b>1 Introducción</b>	<b>13</b>
1.1 Resumen ejecutivo . . . . .	13
1.2 Limitaciones por el contexto del COVID-19 . . . . .	14
1.3 Justificación del proyecto . . . . .	15
1.3.1 ¿Por qué este trabajo? . . . . .	15
1.3.2 ¿Qué es IoT? . . . . .	15
1.3.3 Aplicaciones . . . . .	17
1.3.3.1 Aplicaciones de consumo . . . . .	17
1.3.3.2 Aplicaciones en infraestructura . . . . .	17
1.3.3.3 Proyectos de investigación . . . . .	18
1.3.3.4 Aplicaciones en agricultura . . . . .	18
1.3.3.5 Aplicaciones en industria . . . . .	18
1.3.4 ¿Por qué utilizarlo en monitoreo remoto? . . . . .	18
1.4 Enfoque de gestión . . . . .	19
1.5 Enfoque técnico . . . . .	21
1.6 Conocimientos abarcados . . . . .	22
1.7 Resumen . . . . .	22
<b>2 Objetivos</b>	<b>23</b>
2.1 Motivaciones . . . . .	23
2.2 Objetivo general . . . . .	24
2.3 Objetivos específicos . . . . .	24

2.4	Subproyectos de aplicación . . . . .	25
2.4.1	Precursor Sísmico . . . . .	25
2.4.1.1	Necesidad detectada . . . . .	25
2.4.1.2	Objetivo de trabajo . . . . .	25
2.4.2	Sensado de nivel de agua sobre puente vehicular . . . . .	27
2.4.2.1	Necesidad detectada . . . . .	27
2.4.2.2	Objetivo de trabajo . . . . .	28
2.5	Resumen . . . . .	29
<b>3</b>	<b>Definición del producto</b>	<b>30</b>
3.1	Requerimientos de usuario . . . . .	30
3.1.1	L0.1 - Versatilidad . . . . .	32
3.1.2	L0.2 - Transmisión remota . . . . .	32
3.1.3	L0.3 - Sistema de alertas . . . . .	33
3.1.4	L0.4 - Tiempo real . . . . .	33
3.1.5	L0.5 - Accesibilidad de datos . . . . .	33
3.2	Requerimientos técnicos . . . . .	34
3.3	Casa de calidad . . . . .	35
3.3.1	Matriz de interrelación . . . . .	35
3.3.2	Matriz de correlaciones técnicas . . . . .	35
3.3.3	Matriz de evaluación del mercado . . . . .	35
3.3.4	Evaluación técnica . . . . .	36
3.4	Resumen . . . . .	38
<b>4</b>	<b>Análisis de factibilidad temporal</b>	<b>39</b>
4.1	Alcance del proyecto . . . . .	39
4.2	Márgenes de tareas . . . . .	41
4.3	Diagrama de Gantt . . . . .	41
4.4	Resumen . . . . .	45
<b>5</b>	<b>Análisis de factibilidad económica</b>	<b>46</b>
5.1	Análisis competitivo . . . . .	46
5.1.1	Mercado . . . . .	46
5.1.2	Competidores . . . . .	47
5.2	Ciclo de vida . . . . .	48
5.3	Estimación de costos . . . . .	49
5.3.1	Estimación de los costos directos por unidad de producto . . . . .	51
5.3.2	Estimación de los costos indirectos . . . . .	52
5.3.3	Estimación del costo total de una unidad de producto . . . . .	52
5.4	Estimación del precio de venta . . . . .	52
5.5	Estimación de volúmenes de venta . . . . .	53
5.6	Estimación de la inversión inicial . . . . .	53
5.7	Análisis del flujo de caja descontado durante el ciclo de vida . . . . .	54
5.8	Estimación del VAN . . . . .	55
5.9	Determinación de la TIR . . . . .	56
5.10	Resumen . . . . .	56

<b>6 Análisis de factibilidad técnica</b>	<b>57</b>
6.1 Descripción de alto nivel . . . . .	57
6.2 Alternativas de Hardware . . . . .	58
6.2.1 Arduino . . . . .	58
6.2.2 Raspberry Pi . . . . .	58
6.2.3 ESP-32 . . . . .	59
6.2.4 Hardware utilizado . . . . .	59
6.3 Protocolos y enlaces de comunicación . . . . .	59
6.3.1 USSD . . . . .	59
6.3.2 CoAP . . . . .	60
6.3.3 MQTT . . . . .	60
6.3.4 Medios de comunicación utilizados . . . . .	61
6.4 Formato del paquete . . . . .	63
6.4.1 Formato YAML . . . . .	63
6.4.2 Paquete estándar . . . . .	64
6.4.3 Paquete personalizado . . . . .	65
6.5 Almacenamiento de datos . . . . .	66
6.5.1 Bases de datos no estructuradas . . . . .	66
6.5.2 Base de datos relacional . . . . .	67
6.6 Visualización de datos . . . . .	72
6.7 Análisis de consumo . . . . .	73
6.8 Resumen . . . . .	76
<b>7 Ingeniería en detalle</b>	<b>77</b>
7.1 Transmisión Sensores - Modulo . . . . .	78
7.1.1 Comunicación serial . . . . .	78
7.1.2 Módulo intercomunicador . . . . .	80
7.1.2.1 Sistema operativo Raspbian . . . . .	82
7.2 Transmisión Módulo - Nube . . . . .	84
7.2.1 Conexión 3G . . . . .	84
7.2.2 Solución Cloud . . . . .	86
7.2.2.1 Heroku CLI . . . . .	86
7.2.3 Características técnicas del protocolo MQTT . . . . .	87
7.2.3.1 Estructura de paquete . . . . .	87
7.2.3.2 Calidad de Servicio (QoS) . . . . .	91
7.2.3.3 Análisis con Wireshark . . . . .	92
7.2.4 Broker - Heroku CloudMQTT . . . . .	95
7.2.5 Clientes . . . . .	96
7.2.5.1 Publisher . . . . .	97
7.2.5.2 Automatización de la publicación . . . . .	98
7.2.5.3 Subscriber . . . . .	100
7.3 Almacenamiento de datos . . . . .	102
7.3.1 Heroku Postgres . . . . .	102
7.3.2 API de comunicación . . . . .	103
7.4 Portal web . . . . .	105
7.4.1 Grafana Enterprise . . . . .	106
7.4.2 Grafana Cloud . . . . .	109
7.5 Resumen . . . . .	110

<b>8 Módulos complementarios</b>	<b>111</b>
8.1 Telemando . . . . .	111
8.1.1 Funcionamiento . . . . .	112
8.2 Calibrador . . . . .	113
8.3 Resumen . . . . .	115
<b>9 Construcción del prototipo</b>	<b>116</b>
9.1 Primeras pruebas con puerto serie . . . . .	116
9.2 Primera publicación . . . . .	117
9.3 Operacionalización de los clientes . . . . .	120
9.4 Reemplazo del módulo WiFi por módem 3G . . . . .	122
9.5 Incorporación de la base de datos . . . . .	122
9.6 Escalabilidad a múltiples aplicaciones . . . . .	123
9.7 Incorporación de visualización de datos . . . . .	124
9.8 Prueba punta a punta y automatización del flujo . . . . .	126
9.9 Resumen . . . . .	128
<b>10 Conclusiones</b>	<b>129</b>
10.1 Resultados obtenidos . . . . .	130
10.2 Matriz de verificación de requerimientos . . . . .	132
10.3 Aporte del trabajo realizado . . . . .	134
10.4 Lecciones aprendidas y recomendaciones . . . . .	134
10.5 Líneas futuras . . . . .	135
<b>11 Agradecimientos</b>	<b>137</b>
<b>12 Definiciones</b>	<b>138</b>
12.1 Capítulo 1 . . . . .	138
12.2 Capítulo 2 . . . . .	139
12.3 Capítulo 3 . . . . .	139
12.4 Capítulo 5 . . . . .	140
12.5 Capítulo 4 . . . . .	140
12.6 Capítulo 7 . . . . .	140
<b>13 Acrónimos</b>	<b>141</b>
13.1 Capítulo 1 . . . . .	141
13.2 Capítulo 2 . . . . .	141
13.3 Capítulo 3 . . . . .	141
13.4 Capítulo 5 . . . . .	142
13.5 Capítulo 4 . . . . .	142
13.6 Capítulo 7 . . . . .	142
13.7 Capítulo 8 . . . . .	142
13.8 Capítulo 10 . . . . .	142
<b>14 Anexos</b>	<b>143</b>
14.1 Presentación del Plan del TP Profesional . . . . .	143
14.2 Sistema de adquisición de datos para el estudio Precursoros Sísmicos - Paper151	151
14.3 Factibilidad técnica . . . . .	157
14.3.1 Formato YAML . . . . .	157

14.4 Ingeniería en detalle . . . . .	158
14.4.1 Escritura serial . . . . .	158
14.4.2 Lectura serial . . . . .	159
14.4.3 Configuración 3G . . . . .	159
14.4.4 Paquete JSON de la API a la base de datos . . . . .	160
14.4.5 Configuración de Grafana Enterprise . . . . .	161
14.5 Módulos complementarios . . . . .	163
14.5.1 Interface Control Document . . . . .	163
14.6 Construcción del prototipo . . . . .	173
14.6.1 Primera publicación . . . . .	173
14.6.2 Operacionalización de los clientes . . . . .	174
14.6.3 Incorporación de la base de datos . . . . .	175
14.6.4 Escalabilidad a múltiples aplicaciones . . . . .	176
<b>15 Referencias</b>	<b>177</b>

# Índice de figuras

1	Estructura del Trabajo Profesional . . . . .	9
1.1	Modelo del sistema . . . . .	14
1.2	Niveles típicos de costo y dotación personal en una estructura genérica de ciclo de vida del proyecto (imagen inspirada en la original del PMBOK) <sup>7</sup> . . . . .	20
1.3	Grupos de procesos del proyecto (Imagen inspirada en la original del PMI). <sup>9</sup> . . . . .	20
2.1	Diagrama en bloques del sistema de adquisición de datos sísmicos . . . . .	26
2.2	Equipo de medición de señales ELF-ULF . . . . .	27
2.3	Comunicación del sensor de nivel de agua con el módulo intercomunicador . . . . .	29
3.1	Casa de calidad . . . . .	37
5.1	Ciclo de vida de un proyecto. . . . .	48
5.2	Volumen de ventas . . . . .	53
5.3	Flujo de caja Descontado . . . . .	55
6.1	MQTT. . . . .	61
6.2	Resultado CoAP. . . . .	62
6.3	Resultado MQTT. . . . .	63
6.4	Formato de paquete de datos grande. . . . .	65
6.5	Formato de pocos datos. . . . .	65
6.6	Ejemplo Base de Datos Relacional . . . . .	67
7.1	Diagrama en bloques de los módulos a analizar . . . . .	77
7.2	Sección de transmisión serial . . . . .	78
7.3	Ejemplo de trama UART. . . . .	78
7.4	<code>dmesg   grep tty</code> . . . . .	79
7.5	Diagrama de Pin Out - Rasapberry Pi 2B . . . . .	80
7.6	Sección módulo intercomunicador . . . . .	80
7.7	Raspberry Pi 2B V1.1 . . . . .	81
7.8	Opciones de instalación . . . . .	83
7.9	Sección conexión 3G . . . . .	84
7.10	Módem 3G utilizado . . . . .	85
7.11	Conexión del módem 3G. . . . .	85
7.12	Sección Nube . . . . .	86
7.13	Paquete MQTT. . . . .	87
7.14	Encabezado variable y carga útil de paquete CONNECT. . . . .	89
7.15	Paquete de conexión. . . . .	90
7.16	Paquete de publicación. . . . .	90

7.17 Paquete de suscripción. . . . .	91
7.18 QoS nivel 0. . . . .	91
7.19 QoS nivel 1. . . . .	92
7.20 QoS nivel 2. . . . .	92
7.21 Captura de Wireshark de conexión y publicación. . . . .	93
7.22 Detalle del paquete de conexión. . . . .	93
7.23 Detalle del paquete de publicación. . . . .	94
7.24 Captura de Wireshark de la suscripción. . . . .	94
7.25 Sección Broker MQTT . . . . .	95
7.26 Ejemplo de prueba de un tópico. . . . .	96
7.27 Sección Publisher MQTT . . . . .	97
7.28 Diagrama UML del algoritmo de publicación. . . . .	98
7.29 Definición del servicio de publicación . . . . .	99
7.30 Servicio de publicación en funcionamiento . . . . .	100
7.31 Sección Subscriber MQTT . . . . .	100
7.32 Diagrama UML del subscriber. . . . .	101
7.33 Sección Base de Datos . . . . .	102
7.34 Diagrama relacional de la base de datos . . . . .	103
7.35 Diagrama de alto nivel del uso de la API . . . . .	104
7.36 Diagrama UML del guardado en la base de datos . . . . .	105
7.37 Sección Portal Web . . . . .	105
7.38 Arquitectura del servidor Web y Grafana . . . . .	106
7.39 Ejemplo de visualizaciones en grafana (Para la creación de alertas ver sección 9.7) . . . . .	107
7.40 Opciones de descarga de datos en Grafana . . . . .	108
7.41 Arquitectura Grafana - PostgreSQL . . . . .	109
 8.1 Modelo bidireccional del sistema . . . . .	111
8.2 Funcionamiento del telecomando - Diagrama UML . . . . .	112
8.3 Botón utilizado en Grafana para Telecomando . . . . .	113
8.4 Módulo CALSIM. . . . .	114
8.5 Módulo CALSIM. . . . .	115
 9.1 Prueba comunicación serial - Conexión . . . . .	117
9.2 Raspberry Pi con módulo WiFi . . . . .	118
9.3 Tópicos agregados sobre la instancia del Broker . . . . .	119
9.4 Resultados de la primera publicación . . . . .	119
9.5 Subscriber conectado desde Heroku . . . . .	120
9.6 Resultados de la publicación con publisher operativo . . . . .	121
9.7 Prueba ded publicación sobre red 3G . . . . .	122
9.8 Esquema de la base de datos creada en PosgreSQL UI. . . . .	123
9.9 Ejemplo de alertas en Grafana . . . . .	125
9.10 Prueba punta a punta. . . . .	126
9.11 Publicación de datos de temperatura y humedad . . . . .	127
9.12 Recepción de los datos desde la nube . . . . .	127
9.13 Almacenamiento de los datos recibidos . . . . .	128
9.14 Ejemplo de error de envío . . . . .	128

# Índice de tablas

4.1	WBS - Márgenes de tareas . . . . .	41
5.1	Estimación de costos . . . . .	50
5.2	Precio de venta . . . . .	52
5.3	Costos de inversión . . . . .	54
6.1	Motores de base de datos - Comparativa . . . . .	71
6.2	Pruebas de stress sobre Raspberry Pi 3B + . . . . .	73
6.3	Pruebas de stress sobre Raspberry Pi 3B . . . . .	73
6.4	Autonomía de la Raspberry Pi 3B para distintas baterías de Litio (3,7 V) . . . . .	75
6.5	Tiempos de carga de baterías de litio con distintos paneles solares . . . . .	76
7.1	Tipos de comandos . . . . .	88
7.2	Diferencias Grafana Cloud . . . . .	109
10.1	Matriz de verificación de requerimientos . . . . .	133

# Capítulo 0

## Estructura del Trabajo

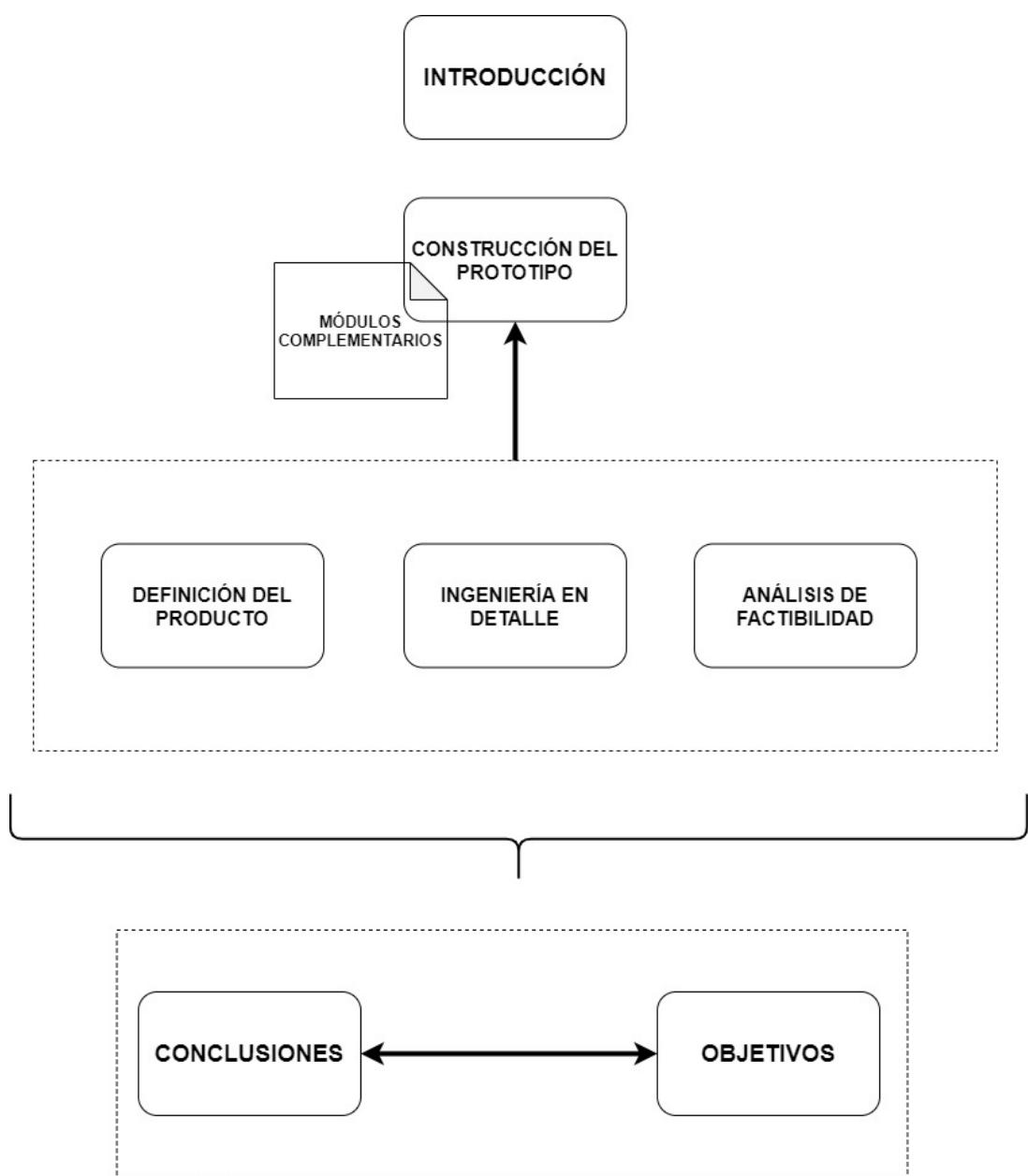


Figura 1: Estructura del Trabajo Profesional

## **0.1 Capítulo 1 - Introducción**

En el primer capítulo de este trabajo se presentan las justificaciones del proyecto así como también el marco contextual en cual se llevo a cabo. De esta manera se brinda información previa a la realización del mismo y el valor que se quiere agregar con su realización. Asimismo, se brinda información a alto nivel respecto de las tecnologías utilizadas y sus distintas aplicaciones en la industria. Por último, se presentan los enfoques técnicos y de gestión a los que se apuntaron durante la realización de este trabajo.

## **0.2 Capítulo 2 - Objetivos**

Se presentan aquí los objetivos a alto y mediano nivel, tanto técnicos como de gestión y futura comercialización. El foco principal de este capítulo se pone en la necesidad detectada que este trabajo busca resolver. Puntualmente, se analizan los proyectos activos con los que se trabajó y los cuales necesitan de la solución presentada en este informe para poder ser completados. Se analiza cuál es su necesidad específica y cómo este trabajo puede cubrirla.

En última instancia, se listan los conocimientos técnicos necesarios para la correcta realización del trabajo.

## **0.3 Capítulo 3 - Definición del producto**

En este capítulo se hace un desglose de los requerimientos del usuario, yendo desde los de alto nivel, hasta los más técnicos que sirven para cumplir las necesidades. Se puede ver acá un mapa con este desglose y la explicación de cada uno.

Al final del capítulo se hace un análisis con una casa de calidad, de forma de encontrar las correlaciones entre los requerimientos y su importancia relativa respecto de la consideración del usuario final y su comparación con los productos de la competencia.

## **0.4 Capítulo 4 - Análisis de factibilidad temporal**

Se revisa en este capítulo el alcance del proyecto, introduciendo los objetivos de manera SMART, los cuales son los mismos que en el Plan de Trabajo Profesional. Se establece la relación entre estos y los requerimientos de usuario y el desglose de tareas a lo largo del período de trabajo. Esto se hizo a través de un diagrama de Gantt.

## **0.5 Capítulo 5 - Análisis de factibilidad económica**

Comienza por un análisis de los principales competidores, distintos productos/servicios que brindan una solución similar a la que se presenta en este informe. Lo más destacado de esta sección implica el análisis de costos realizado, que brinda al usuario información de cuál será la inversión necesaria para llevar adelante este proyecto. Por último, se

presentan los indicadores económicos como el VAN y el TIR, en base a la estimación de los volúmenes de venta, para calcular los tiempos de retorno de la inversión del proyecto en cuestión.

## 0.6 Capítulo 6 - Análisis de factibilidad técnica

En la factibilidad técnica se toman en consideración los distintos enfoques del proyecto. Por un lado se analizan las ventajas y desventajas de las alternativas para el módulo intercomunicador, en términos de usabilidad, sistema operativo, programabilidad, consumo, entre otros. Por otro lado, se realiza un análisis análogo para los distintos protocolos de IoT posibles a utilizar en el proyecto. Luego, se analizan los estándares de formato de datos para el paquete que se utiliza en el trabajo. Por último, se presentan los cálculos correspondientes al consumo del módulo intercomunicador y las alternativas a utilizar según el enfoque del proyecto. Cada uno de estos análisis se presenta con la elección y justificación de cada uno de los items a utilizar, desde el hardware, protocolo y formato de paquete de datos.

## 0.7 Capítulo 7 - Ingeniería en detalle

En este capítulo se presenta la implementación de cada una de las etapas del proyecto, brindando el detalle de cada tecnología utilizada, los protocolos, la arquitectura de los mismos, como el protocolo IoT elegido, por ejemplo. La idea básica de este capítulo es hacer el repaso completo de la implementación final del proyecto en cuestión, analizando cada paso de punta a punta y justificando la elección de cada ítem elegido para cada módulo. Se explica también la entrada y salida de cada módulo y los requisitos necesarios para que la salida de uno sea compatible con la entrada del siguiente. En el índice se pueden encontrar los detalles de cuáles son las etapas y módulos que se explican.

## 0.8 Capítulo 8 - Módulos complementarios

Aquí se listan y explican aquellas partes del trabajo que estaban por fuera de los objetivos planteados y se agregaron para darle un complemento extra al proyecto. Principalmente se presenta la caja de calibración que emula la generación de datos del precursor sísmico; también se explica la función de telecomando creada, la cual se agregó de forma tal de darle la posibilidad al usuario de enviar comandos hacia el módulo intercomunicador, haciendo la comunicación bidireccional.

## 0.9 Capítulo 9 - Construcción del prototipo

En esta sección se presentan los pasos realizados hasta arribar al prototipo presentado finalmente. Se listan las pruebas realizadas en cada etapa para cada módulo individual así como también los pasos necesarios para su integración. Se demuestra el crecimiento incremental del proyecto desde un punto de vista técnico y se verifican los requerimientos descriptos en el capítulo *Definición del producto*.

## **0.10 Capítulo 10 - Conclusiones**

Se presentan aquí los resultados obtenidos. Principalmente se detalla el cumplimiento de los objetivos y los puntos fundamentales del proceso necesario para ello, como así también las lecciones aprendidas. En adición, se muestra la matriz de verificación de cada uno de los requerimientos de usuario y las secciones donde se verifican los mismos.

Por otro lado se presenta el aporte del trabajo realizado, ya sea a las futuras generaciones de ingenieros, como el valor agregado que introdujo el proyecto a la industria.

Por último, de lo que se aprendió se hace un análisis para poder dar recomendaciones en base a lo que se realizó y los futuros proyectos que puedan surgir de la lectura de éste.

# Capítulo 1

## Introducción

En este capítulo se presentan las justificaciones del proyecto así como también el marco contextual en cual se llevo a cabo. De esta manera se brinda información previa a la realización del mismo y el valor que se quiere agregar una vez concluido. Asimismo, se brinda información a alto nivel respecto de las tecnologías utilizadas y sus distintas aplicaciones en la industria. Se presentan además los enfoques técnicos y de gestión a los que se apuntaron durante la realización de este trabajo. Por último se cubren los conocimientos técnicos abarcados durante el desarrollo del proyecto y las materias asociadas a dichos conocimientos.

### 1.1 Resumen ejecutivo

El presente informe busca plasmar el detalle del desarrollo del proyecto de la *Plataforma de telemetría IoT aplicada al estudio de precursores sísmicos*. En la sección 14.1 se muestra el plan presentado en Comisión Curricular de la Facultad de Ingeniería de la Universidad de Buenos Aires. En el mismo se detallan los objetivos del proyecto junto con las necesidades detectadas en la industria y la motivación de este grupo para cubrirlas. A lo largo de este informe, se le da más detalle a los objetivos planteados y se presentan en distintas secciones la implementación correspondiente para el cumplimiento de cada uno de ellos.

A modo de resumen, el proyecto consiste en un sistema de punta a punta con el fin de transmitir los datos emitidos por un conjunto de sensores de campo eléctrico y campo magnético hacia la nube. Estos sensores son desarrollados por un proyecto paralelo al que se presenta en este informe, a cargo del Laboratorio de Radiación Electromagnética de la Facultad de Ingeniería (Universidad de Buenos Aires). El presente trabajo muestra el desarrollo del sistema que comunica esos sensores, a través de un dispositivo intercomunicador y utilizando UMTS (Sistema Universal de Telecomunicaciones Móviles) o red de Tercera Generación (3G), con una estación de trabajo remota, para la correcta visualización de los datos.

Si bien el fin de esta plataforma es transmitir datos sísmicos, el desarrollo busca ser lo suficientemente genérico para poder enviar datos de cualquier índole, de forma de que el modelo final sea adaptable a cualquier tipo de medición de magnitudes físicas.

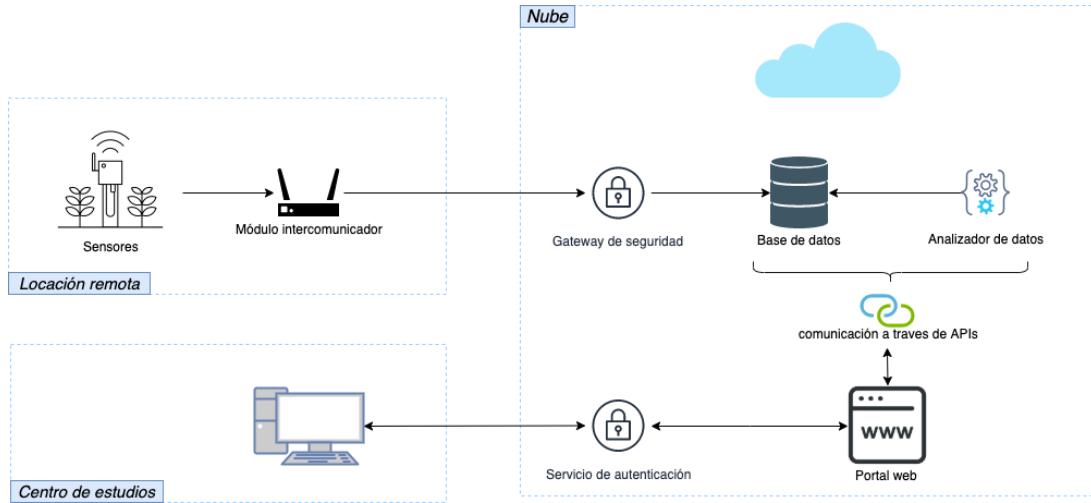


Figura 1.1: Modelo del sistema

En términos generales, el sistema consiste de varios módulos, comenzando con un módulo intercomunicador, que funciona como receptor de los datos de los sensores. El mismo debe tener la funcionalidad de procesarlos para su correcta transmisión y envío. Luego, a través de una red 3G y sobre un protocolo ligero de IoT (*Internet of Things - Internet de las cosas*), estos son enviados a un servidor en la nube. El mismo servidor cuenta con un módulo receptor que procesa los datos recibidos y los almacena en una base de datos, la cual también está localizada en la nube. Por último, se utiliza una herramienta de visualización que toma información de esa base de datos y presenta los datos en forma de gráficos, con la opción de descargarlos y generar alertas. Adicionalmente se le agregan módulos extra que agregan funcionalidades al proyecto, por ejemplo la opción de un telemando que permite al usuario enviar instrucciones al módulo intercomunicador.

La idea del producto final es que los datos puedan ser consumidos en tiempo real de forma que el usuario pueda tener acceso rápido a los mismos y tomar acciones necesarias.

Los potenciales clientes serán tanto los laboratorios universitarios implicados en los precursores sísmicos como todos aquellos que busquen realizar el monitoreo de magnitudes físicas en tiempo real. Al mismo tiempo, busca extenderse hacia el ámbito público, ya sea dentro de instituciones u organismos de investigación, como en el sector privado, dentro de empresas del ámbito del petróleo como lo pueden ser también diversas firmas que trabajen en ubicaciones remotas y tengan acceso limitado a los datos que monitorean.

## 1.2 Limitaciones por el contexto del COVID-19

El proyecto presentado en este informe fue realizado, en gran parte, durante el período de aislamiento estricto obligatorio debido a la pandemia del COVID-19. A raíz de esto, muchas de las pruebas y mediciones planeadas se vieron limitadas por la falta de acceso a los laboratorios de la facultad como así también por la limitación en la compra de material necesario.

Por otro lado, el precursor sísmico, proyecto paralelo a cargo de un laboratorio de la Facultad de Ingeniería de la UBA, encargado de generar los datos necesarios para su

transmisión por medio de la solución presentada, también se vio afectado por la situación del Coronavirus. El mismo no pudo completarse en su implementación, por lo que no pudieron realizarse las pruebas con el dispositivo final. A raíz de esta situación, los autores diseñaron un simulador llamado CALSIM (**Cap. 13.1**) que toma datos analógicos y los convierte en el formato necesario para su correcta transmisión. Los detalles se pueden encontrar en la sección 8.2.

Esta sección no pretende ser un descargo de responsabilidades, pues a pesar de la situación, se pudieron cumplir con todos los objetivos del trabajo de manera exitosa, sino como aclaración inicial antes de la lectura del informe, por si el lector percibe ciertas pruebas, mediciones o implementaciones que pudieran haberse realizado de forma distinta en un contexto normal. Dichas etapas del proyecto se aclaran y hacen mención a la presente sección en forma de aclaración.

## 1.3 Justificación del proyecto

### 1.3.1 ¿Por qué este trabajo?

El presente informe busca explicar lo realizado durante la etapa de investigación pruebas e implementación del sistema de telemetría planteado. Para ello, se buscó aplicar, desde el primer diseño, los conceptos y conocimientos adquiridos durante toda la etapa educativa en la carrera de Ingeniería electrónica, aplicando el pensamiento racional en cada una de las etapas de la solución, realizando los cálculos y pruebas correspondientes, identificando posibles contingencias y haciendo los análisis correspondientes para poder mitigarlas, con el fin de sentar el ejemplo para las próximas generaciones de estudiantes de esta carrera.

Por otro lado, a sabiendas de que el área de estudio dentro del marco de la electrónica está en constante evolución, la intención de los autores fue la de realizar un proyecto con implementación de nuevas tecnologías utilizadas en la industria, como el uso de la nube para almacenamiento y *hosting* (**Cap. 12.1**), protocolos modernos de Internet de las Cosas. Esto permitió la adaptación del proyecto a la industria moderna, para su potencial comercialización, como así también la aplicación de conocimientos enfocados principalmente en las redes de datos, modelo OSI, estructura cliente servidor, y diversos conceptos relacionados al desarrollo de software. Al mismo tiempo, gran parte de la implementación estuvo dedicada al análisis de la factibilidad del proyecto, tanto técnica como económica, pensando a la solución como un producto para su comercialización, lo que implicó la aplicación de los conocimientos adquiridos en gestión de proyectos.

### 1.3.2 ¿Qué es IoT?

IoT (*Internet of Things - Internet de las cosas*)<sup>1</sup> es un concepto que se refiere a una interconexión digital de objetos cotidianos con internet.<sup>23</sup> Es en definitiva, la conexión de internet más con objetos que con personas. También se suele conocer como internet de todas las cosas o internet en las cosas. Si los objetos de la vida cotidiana tuvieran incorporadas etiquetas de radio, podrían ser identificados y gestionados por otros equipos

de la misma manera que si lo fuesen por seres humanos.

Por ejemplo, si los libros, termostatos, heladeras, la paquetería, lámparas, botiquines, partes automotrices, entre otros, estuvieran conectados a internet y equipados con dispositivos de identificación, no existirían, en teoría, artículos fuera de stock o medicinas caducadas; sabríamos exactamente la ubicación, cómo se consumen en el mundo; el extravío pasaría a ser cosa del pasado, y sabríamos qué está encendido y qué está apagado en todo momento. Constituye un cambio radical en la calidad de vida de las personas en la sociedad, ofrece una gran cantidad de nuevas oportunidades de acceso a datos, servicios específicos en la educación, seguridad, asistencia sanitaria y en el transporte, entre otros campos.

Las aplicaciones para dispositivos conectados a internet son amplias. Múltiples categorías han sido sugeridas, pero la mayoría está de acuerdo en separar las aplicaciones en tres principales ramas de uso: consumidores, empresarial, e infraestructura.

La capacidad de conectar dispositivos embebidos con capacidades limitadas de CPU, memoria y energía significa que IoT puede tener aplicaciones en casi cualquier área. Estos sistemas podrían encargarse de recolectar información en diferentes entornos: desde ecosistemas naturales hasta edificios y fábricas, por lo que podrían utilizarse para monitoreo ambiental y planeamiento urbanístico.

## Características

### *Inteligencia*

El internet de las cosas es "no determinista" y de red abierta. En ella entidades inteligentes auto-organizadas (componentes SOA (**Cap. 13.1**)) u objetos virtuales son interoperables y capaces de actuar de forma independiente, tomando en consideración el contexto o el ambiente. De esta forma se genera una inteligencia ambiental.

### *Arquitectura*

El sistema es un ejemplo de "arquitectura orientada a eventos",<sup>4</sup> construida de abajo hacia arriba (basada en el contexto de procesos y operaciones, en tiempo real) y tiene en consideración cualquier nivel adicional. Por lo tanto, el modelo orientado a eventos y el enfoque funcional coexisten con nuevos modelos capaces de tratar excepciones y la evolución insólita de procesos.

En un internet de las cosas, el significado de un evento no está necesariamente basado en modelos determinísticos o sintácticos. Se basa en el contexto del propio evento: así, es también una Web Semántica. En consecuencia, no son estrictamente necesarias normas comunes que no son capaces de manejar todos los contextos o usos: algunos actores (servicios, componentes, avatares) están autorreferenciados de forma coordinada y, si fuera necesario, se adaptan a normas comunes (para predecir algo solo sería necesario definir una "finalidad global", algo que no es posible con ninguno de los actuales enfoques y normas).

## *Seguridad*

La empresa Hewlett Packard realizó un estudio en 2015<sup>5</sup> reportando que, entre otros hallazgos respecto a los dispositivos IoT, el 70% de ellos tiene vulnerabilidades de seguridad en sus contraseñas, además de problemas con cifrado de datos o permisos de acceso. El 50% de las aplicaciones de dispositivos móviles no encriptan las comunicaciones. La firma de seguridad Kaspersky Lab también realizó pruebas en objetos conectados al IoT y encontró que una cámara de vigilancia para bebés podía ser "hackeada" para robar el video, así como que en una cafetera no encriptada se podía conocer la contraseña de la red WiFi a la que estuviera conectada.

La proyección de crecimiento de dispositivos IoT ha sido exponencial. Este crecimiento puede hacer que el tópico de la seguridad de datos desemboque en una situación más crítica ante la falta de procesos que aseguren la integridad y encriptación de estos.

Los datos que guardan los dispositivos IoT son altamente codiciados debido a que almacenan información sobre los hábitos de los usuarios. Contar con esas bases de datos es valiosa para varias empresas, que pueden dirigir sus esfuerzos en productos y servicios enfocados en los hábitos y preferencias de las masas. Lo que podrá ayudar a aminorar el problema será el cifrado y la encriptación de datos a la hora de subir los datos a la nube.

### **1.3.3 Aplicaciones**

#### **1.3.3.1 Aplicaciones de consumo**

Un porcentaje creciente de los dispositivos IoT son creados para el consumo. Algunos ejemplos de aplicaciones de consumo incluyen: automóviles conectados, entretenimiento, automatización del hogar, tecnología vestible, salud conectada y electrodomésticos como lavadoras, secadoras, aspiradoras robóticas, purificadores de aire, hornos, refrigeradores que utilizan Wi-Fi para seguimiento remoto de los procesos.

#### **1.3.3.2 Aplicaciones en infraestructura**

El seguimiento y control de operaciones de infraestructura urbana y rural como puentes, vías férreas y parques eólicos, es una aplicación clave de IoT. La infraestructura de IoT puede utilizarse para seguir cualquier evento o cambio en las condiciones estructurales que puedan comprometer la seguridad e incrementar el riesgo. También puede utilizarse para planificar actividades de reparación y mantenimiento de manera eficiente, coordinando tareas entre diferentes proveedores de servicios y los usuarios de las instalaciones. Otra aplicación de los dispositivos de IoT es el control de infraestructura crítica, como puentes para permitir el pasaje de embarcaciones. El uso de dispositivos de IoT para el seguimiento y operación de infraestructura puede mejorar el manejo de incidentes, la coordinación de la respuesta en situaciones de emergencia, la calidad y disponibilidad de los servicios, además de reducir los costos de operación en todas las áreas relacionadas a la infraestructura. Incluso áreas como el manejo de desperdicios puede beneficiarse de la automatización y optimización que traería la aplicación de IoT.

### **1.3.3.3 Proyectos de investigación**

Muchos laboratorios trabajan hoy en día con sensores de distintos tipos de magnitudes. El fin es poder sacar conclusiones sobre fenómenos físicos en general, de forma de poder brindar una mejor visión y panorama de los mismos. Muchos de estos sensores funcionan sobre locaciones remotas sin la posibilidad de acceder fácilmente a los datos tomados por los dispositivos. Aquí es donde las tecnologías relacionadas a IoT pueden interactuar con dichos sensores para la correcta transmisión de los datos.

### **1.3.3.4 Aplicaciones en agricultura**

La población mundial alcanzará los 9700 millones en 2050 según la Organización de Naciones Unidas,<sup>6</sup> por lo tanto para alimentar a esta gran cantidad de población la industria agrícola debe adoptar el IoT.

La agricultura inteligente basada en IoT permitirá a los productores y agricultores reducir el desperdicio y mejorar la productividad, desde la cantidad de fertilizante utilizado hasta el combustible utilizado en la maquinaria agrícola. En la agricultura basada en IoT, se construye un sistema para monitorear el campo de cultivo con la ayuda de sensores (luz, humedad, temperatura, humedad del suelo) y la automatización del sistema de riego.

Los agricultores pueden monitorear las condiciones del campo desde cualquier lugar. La agricultura basada en IoT es altamente eficiente en comparación con la tradicional. En términos de cuestiones ambientales la agricultura basada en IoT puede proporcionar grandes beneficios, incluido un uso más eficiente del agua, o la optimización de insumos y tratamientos.

### **1.3.3.5 Aplicaciones en industria**

Cuando IoT se incorpora al entorno industrial y de fabricación, se le conoce como Industrial Internet of Things. El IIoT es una subcategoría importante del IoT, pues consiste en conectar sensores inteligentes a Internet y usar esa información para tomar mejores decisiones comerciales. La mayor diferencia entre el IoT y el IoT industrial es que IIoT ha sido diseñado para funcionar en espacios relativamente cerrados y con el objetivo de facilitar la comunicación con una empresa. Por ejemplo, una de las aplicaciones del IIoT industrial es la detección de grandes concentraciones de polvo en entornos industriales para asegurar una mejor seguridad y salud de los trabajadores.

## **1.3.4 ¿Por qué utilizarlo en monitoreo remoto?**

Existen múltiples industrias cuya zona de trabajo se encuentran en los lugares más apartados, desde plataformas petroleras en alta mar hasta complejos insulares lejanos. Hasta hace poco, sin embargo, las empresas no tenían una forma confiable de monitorear o solucionar problemas de activos remotos. Cuando un sistema fallaba, se debía contar con personal instalado junto al equipo o un técnico experimentado tendría que hacer el viaje al sitio para que los activos volvieran a funcionar. Ambos enfoques implicaban un alto riesgo y un aún más alto costo, tanto desde el punto de vista financiero como potencial para el servicio al cliente.

Esta problemática común en innumerables industrias comienza a resolverse a medida que el Internet de las cosas industrial (IIoT) gana terreno y allana el camino para nueva conectividad y ofertas inteligentes que impulsan la administración remota de activos al siguiente nivel. Al mismo tiempo, la conectividad ubicua del IIoT significa que los fabricantes pueden consolidar un equipo de expertos que pueden resolver problemas y optimizar el rendimiento de los activos para sitios remotos desde una ubicación centralizada.

De igual manera, los avances en sensores inalámbricos y análisis en la nube están permitiendo nuevas formas de monitoreo continuo de activos remotos para el mantenimiento predictivo y la gestión del rendimiento en comparación con lo que solía estar limitado a las revisiones manuales. La capacidad de monitorear y analizar continuamente los datos de vibración o presión provenientes de un intercambiador de calor o de un equipo giratorio en el campo proporciona una mejor visión de la salud y el estado del equipo, permitiendo la detección de problemas antes de lo que era habitual.

Simultáneamente, los fabricantes con múltiples plantas están aprovechando la misma facilidad de conectividad, nuevas plataformas en la nube y capacidades analíticas para monitorear, administrar y mantener de forma remota el equipo de la planta desde una ubicación centralizada, a través de múltiples sitios. Si bien, en general, la conectividad remota a los activos de la planta y sus reservas de datos correspondientes han existido durante algún tiempo, la visibilidad se ha limitado al equipo contenido en gran medida dentro del dominio de una planta específica con una visibilidad mínima para el panorama más grande o en varias plantas. Además, la administración de dispositivos ha sido solo eso, contextualizada a un dispositivo específico. Sin embargo, a medida que el IIoT introduce más inteligencia y una mayor conectividad, fomenta una visión más amplia que puede brindarles a los fabricantes mejores conocimientos para mejorar los procesos, minimizar el tiempo de inactividad y reducir los puntos de estrangulamiento y los problemas de rendimiento en la planta.

## 1.4 Enfoque de gestión

La gestión de proyectos es la aplicación sistemática de conocimientos, habilidades, capacidades, herramientas y técnicas para lograr satisfacer los requerimientos del proyecto.<sup>7,8</sup> El presente proyecto es encarado con un enfoque tradicional desarrollado en 4 fases:

1. Inicio del proyecto
2. Organización y preparación
3. Ejecución del trabajo
4. Cierre del proyecto

El esfuerzo no es constante durante el proyecto y la transición entre fases implica un hito aprobado para seguir:

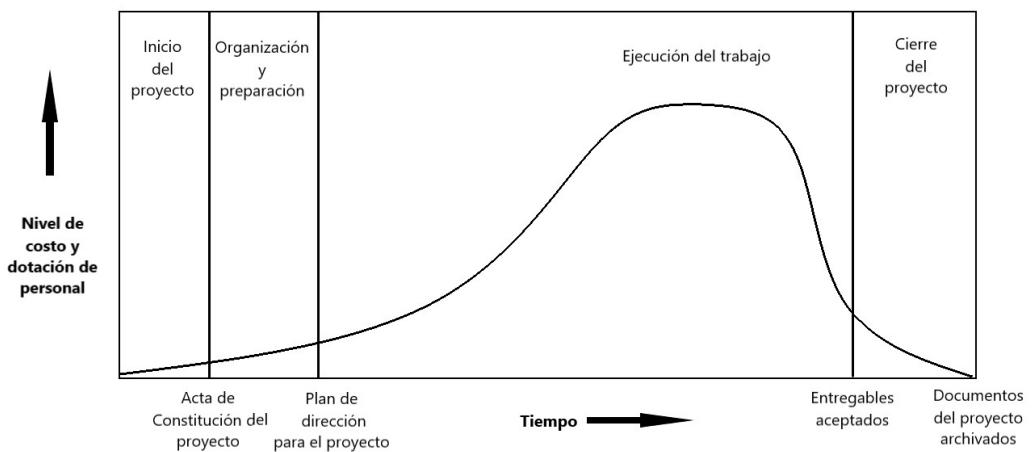


Figura 1.2: Niveles típicos de costo y dotación personal en una estructura genérica de ciclo de vida del proyecto (imagen inspirada en la original del PMBOK)<sup>7</sup>

Las fases de proyecto no deben confundirse con los 5 grupos de procesos que lo componen, independientes de las fases. Estos procesos no se suceden uno a continuación del otro como las fases sino que se superponen a lo largo del ciclo de vida del proyecto:

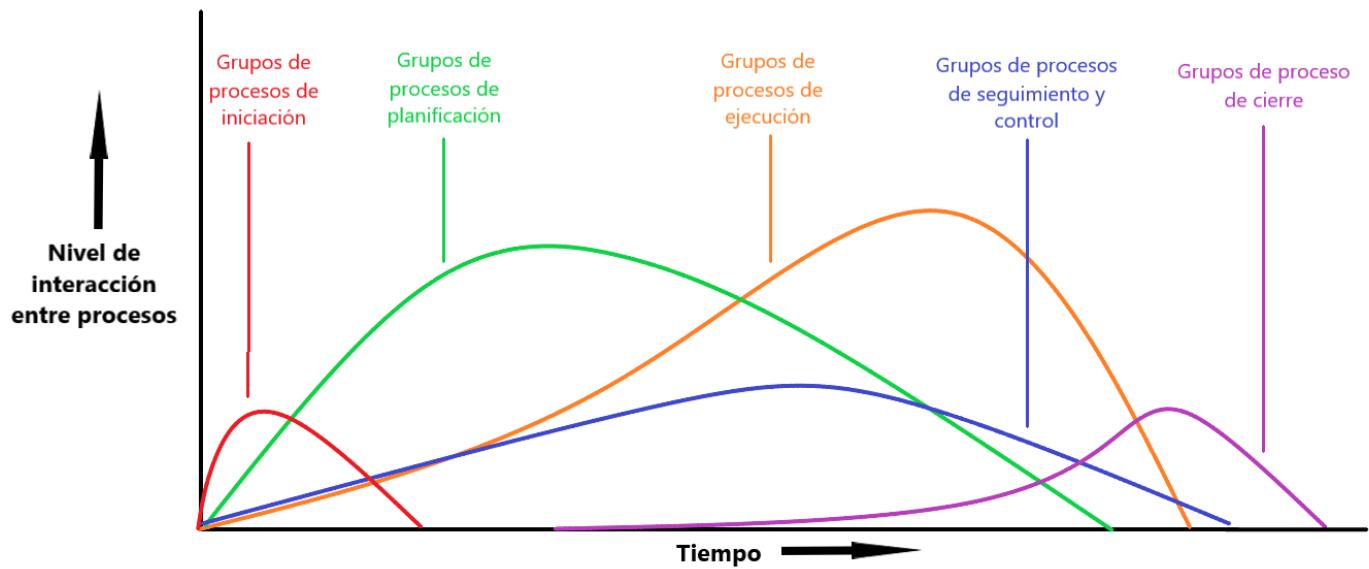


Figura 1.3: Grupos de procesos del proyecto (Imagen inspirada en la original del PMI).<sup>9</sup>

- Los procesos de iniciación responden a los análisis iniciales de alternativas, costo/beneficio, factibilidad y tanto la conveniencia como la posibilidad de ejecutarlo.
- Luego los procesos de planificación incluyen la justificación del proyecto, listado de restricciones e hipótesis, cronograma y desglose del trabajo, listado de recursos disponibles y planificación de gestión de riesgos, comunicación y gestión de calidad.
- Simultáneamente, los procesos de ejecución implican conformar el equipo de trabajo en caso de requerir incorporar mano de obra, asignación de tareas y llevado a cabo

de las mismas.

- Los procesos de seguimiento y control responden a comparar el desempeño real con lo analizado de forma teórica, identificación y resolución de problemas y la comunicación a las partes involucradas de resultados parciales o totales.
- Por ultimo, en los procesos de cierre se busca la aprobación de los resultados finales, cierre de contratos, transición a futuros proyectos y evaluación final buscando identificar lecciones aprendidas.

## 1.5 Enfoque técnico

Se da gran importancia al desarrollo de bloques independientes y reutilizables, de forma de disminuir lo mas posible la interdependencia entre ellos y facilitar tanto las tareas de soporte como el posterior escalado de la aplicación.

Para asegurar esto a nivel software es importante tener en cuenta los 5 principios SOLID (**Cap. 13.1**) de diseño. Si bien los mismos están enfocados a la programación orientada a objetos, muchos de ellos son replicables en casos donde no se usa este tipo de programación. Particularmente los primeros 2, que son los más usados en este proyecto. Los principios son:

- S – Single Responsibility Principle (SRP): Según este principio una clase (**Cap. 12.1**) debería tener una sola responsabilidad.
- O – Open/Closed Principle (OCP): Se debe ser capaz de extender el comportamiento de una clase, sin modificarla. En otras palabras: las clases utilizadas deberían estar abiertas para poder extenderse y cerradas para modificarse.
- L – Liskov Substitution Principle (LSP): Los objetos (**Cap. 12.1**) deben poder ser reemplazados por instancias de sus subtipos sin alterar el correcto funcionamiento del sistema o lo que es lo mismo: si en un programa utilizamos cierta clase, deberíamos poder usar cualquiera de sus subclases sin interferir en la funcionalidad del programa.
- I – Interface Segregation Principle (ISP): Es preferible contar con muchas interfaces que definan pocos métodos que tener una interfaz (**Cap. 12.1**) forzada a implementar muchos métodos a los que no dará uso.
- D – Dependency Inversion Principle (DIP): Los módulos de alto nivel no deberían depender de módulos de bajo nivel. Ambos deberían depender de abstracciones. Las abstracciones (**Cap. 12.1**) no deberían depender de los detalles. Los detalles deberían depender de las abstracciones.

Estos principios ayudan a:

- Crear un software eficaz: que cumpla con su cometido y que sea robusto y estable.
- Escribir un código limpio y flexible ante los cambios: que se pueda modificar fácilmente según necesidad, que sea reutilizable y mantenible.

- Permitir escalabilidad: que acepte ser ampliado con nuevas funcionalidades de manera ágil.

Lo que se busca finalmente es desarrollar un software de calidad.

## 1.6 Conocimientos abarcados

A lo largo de la carrera de Ingeniería electrónica se incorporaron conocimientos en relación con las distintas áreas que involucra. La intención de este proyecto es la aplicación de dichos conocimientos con la intención de entregar un trabajo de calidad y profesional. En la siguiente tabla se presentan los conocimientos abarcados junto con las materias asociadas a los mismos.

Conocimientos abarcados	Materias asociadas
Arquitectura de redes	86.12 - Comunicación de Datos
	86.39 - Redes de Computadoras
Capa de transporte TCP/IP	86.12 - Comunicación de Datos
	86.39 - Redes de Computadoras
	86.28 - Laboratorio de Comunicaciones
Gestión de proyectos	86.14 - Introducción a Proyectos
	86.45 - Industrias y Productos de Electrónica
Sistemas operativos Linux/UNIX	95.11 - Algoritmos y Programación I
	95.12 - Algoritmos y Programación II
Scripting	95.11 - Algoritmos y Programación I
	86.05 - Señales y Sistemas
Economía y Organización de empresas	86.14 - Introducción a Proyectos
	91.19 - Introducción a la Economía y Organización de la empresa
Base de datos	95.11 - Algoritmos y Programación I
	95.12 - Algoritmos y Programación II
	95.02 - Algoritmos y Programación III
Uso y desarrollo de APIs	95.02 - Algoritmos y Programación III
Protocolos de comunicación	86.07 - Laboratorio de Microprocesadores
	86.12 - Comunicación de Datos
	86.39 - Redes de Computadoras
Fuentes de alimentación	86.02 - Introducción a la Ingeniería Electrónica
	86.06 - Circuitos Electrónicos
	86.10 - Diseño de Circuitos Electrónicos

## 1.7 Resumen

Inicialmente se presentó el contexto mundial bajo el cual este trabajo fue realizado, como se logró finalizarlo satisfactoriamente a pesar de las dificultades que el mismo agregó.

Adicionalmente se hizo una breve introducción a las tecnologías utilizadas, sus áreas de aplicación hoy en día y de igual manera se incorporó la visión e intención de los autores a la hora de elegir la problemática a resolver y la modernidad de las tecnologías utilizadas en el proceso.

Por último se incorporó la metodología de trabajo a utilizar a lo largo de todo el proyecto, tanto para los procesos de gestión que un trabajo profesional requiere como para el desarrollo técnico de la plataforma, buscando un enfoque moderno que resulte en un proyecto de calidad y duradero en el tiempo.

# Capítulo 2

## Objetivos

Se presentan aquí los objetivos a alto y mediano nivel, tanto técnicos como de gestión y futura comercialización. El foco principal de este capítulo se pone en la necesidad detectada que este trabajo busca resolver. Puntualmente, se analizan los proyectos activos con los que se trabajó y los cuales necesitan de la solución presentada en este informe para poder ser completados. Se analiza cuál es su necesidad específica y cómo este trabajo puede cubrirla.

### 2.1 Motivaciones

Existen distintos proyectos de monitoreo aplicados a varios campos de la industria y la investigación. Muchos de estos, como el caso analizado, utilizan sensores que deben ser instalados en lugares remotos que pueden llegar a ser inaccesibles. Esto genera una dificultad a la hora del acceso y el procesamiento de los datos, dado que para levantarlos en un dispositivo, uno estaría obligado a permanecer cerca del sensor para permitir la comunicación entre los mismos (sensor y computadora por ejemplo). Esto significa un problema para el caso de grandes investigaciones que requieren del trabajo de científicos o profesionales en áreas lejanas al sensado, como puede ser un laboratorio ubicado en otra ciudad o provincia.

Un ejemplo claro de estos casos son las empresas petroleras, las cuales deben tomar datos sísmicos y de movimientos de suelos con el fin de realizar los correspondientes análisis exploratorios para la extracción del petróleo en lugares de poca cobertura. Lo mismo sucede en campos como la meteorología, donde distintas empresas e instituciones deben sensar datos de humedad, temperatura y otras variables. Todos estos datos generados deben enviarse y almacenarse en algún lugar.

Por otro lado, en el campo de la investigación, existen muchos proyectos dedicados al estudio de suelos. Muchos de estos utilizan sismógrafos instalados en distintas partes del mundo. Hay múltiples laboratorios actuales en la Facultad de Ingeniería de la Universidad de Buenos Aires que pueden beneficiarse con este desarrollo. Muchos de ellos están actualmente trabajando en investigaciones que implican mediciones, muestreos y análisis de parámetros físicos, en particular sobre variables electromagnéticas, en distintas regiones del país que no cuentan con un sistema que las transmita para su visualización desde una locación remota, como podría ser el mismo laboratorio. Actualmente las muestras deben

ser almacenadas local y temporalmente en el mismo sitio donde fueron tomadas para su posterior recolección de forma manual y transporte hasta el laboratorio donde se las podrá analizar. Esto no solo trae gastos de logística y esfuerzos que pueden retrasar el proyecto sino que también lleva a que cualquier tipo de análisis en tiempo real sea imposible.

Así como el estudio de precursores sísmicos, otros laboratorios están trabajando con la medición de nivel de clorofila en el agua, otros en medición de nivel de agua en ríos para la construcción de puentes, y más. En este capítulo se explican algunos de estos proyectos y la necesidad detectada para cada uno de ellos.

## 2.2 Objetivo general

El principal objetivo de este proyecto es lograr el envío exitoso de los datos desde el sensor instalado en una ubicación remota hasta un servidor en la nube. Luego, el correcto procesamiento de los mismos, teniendo en cuenta que corresponden a distintas variables, con distintas magnitudes y frecuencias, con el fin de poder alertar mediante alarmas o avisos útiles para el usuario final.

Como objetivo particular, se busca que con la publicación de los datos procesados en la nube se pueda realizar una prueba de concepto de la comunicación remota que sirva como disparador para la industria dedicada a la investigación de movimientos generados en la tierra (petroleras, mineras, laboratorios de investigación).

La **motivación** de este grupo por la realización de este trabajo se basa en la amplitud de áreas que podrían ser abarcadas por el mismo. Éste proyecto es por tanto, un impulso que sirve para negociar con posibles stakeholders (**Cap. 12.2**) involucrados en esta área, dado su fuerte relación con el estudio de suelos. Se hará especial foco en mantener una estructura de microservicios (**Cap. 12.2**) reutilizables, de forma de que solo se necesiten cambios menores a ciertos bloques funcionales para que la solución se pueda re-aplicar a un campo totalmente nuevo.

De esta forma se busca dejar un código abierto, documentado y que pueda ser fácilmente adaptable y que se pueda continuar por cualquier estudiante de la Facultad de Ingeniería. Es por esto la adaptabilidad del proyecto es un requerimiento importante del mismo, dado que si un laboratorio, por ejemplo, quisiera implementar un sensor de una magnitud genérica, el código generado en este trabajo podría ser re-utilizado para la correcta transmisión de los datos a una estación remota.

## 2.3 Objetivos específicos

A continuación se indican de forma orientativa las distintas tareas que debe realizar el sistema a implementar:

**OBJ.1** Recibir en el módulo intercomunicador (**Cap. 13.2**) los datos del sensor a través de una comunicación por puerto serie.

**OBJ.2** Decodificar los datos recibidos en crudo para convertirlos a algún formato adaptado.

**OBJ.3** Procesar los datos para obtener una visualización de los mismos en el tiempo.

**OBJ.4** Procesar datos para detectar anomalías y establecer alertas.

**OBJ.5** Compresión de datos y transmisión a un servidor en la nube.

**OBJ.6** Publicar página web en un hosting alojado en la nube (datos extras + botón para descargar datos).

## 2.4 Subproyectos de aplicación

### 2.4.1 Precursor Sísmico

#### 2.4.1.1 Necesidad detectada

La mejora de pronósticos y alertas sobre fenómenos de naturaleza sísmica son fundamentales para salvar vidas humanas frente terremotos de gran escala.<sup>11</sup> Existen evidencias en diferentes partes del mundo que sugieren una relación entre cierto tipo de fenómenos magnéticos a frecuencias extremadamente bajas y la ocurrencia de un evento sísmico. Es por ello que los diferentes sistemas de monitoreo para la detección de sismos juegan un rol fundamental en el estudio y anticipación de estos cataclismos, para ello son utilizados los sismógrafos, que son esencialmente instrumentos que miden determinadas magnitudes físicas que guardan cierta correlación con eventos de origen sísmico. A los fenómenos físicos sensados con este propósito se los conoce como precursores sísmicos. Algunos ejemplos de precursores sísmicos convencionales son: comportamiento de los animales, vibraciones de pequeña magnitud, emisión de gas radón, resistividad de las rocas, elevación del suelo, etc. En la sección 14.2 se presenta el proyecto mencionado. En la siguiente subsección se presentan algunos detalles que se acoplan con este trabajo.

Complementariamente, se han desarrollado algunas hipótesis sobre cierto tipo de emisiones electromagnéticas que podrían emplearse también como precursores. En diversas zonas con potencial actividad sísmica se han reportado fenómenos electromagnéticos inexplicables detectados minutos, horas o incluso días antes de producirse el terremoto. Estas anomalías se han manifestado, por ejemplo, como estática producida en radios, encendido de tubos fluorescentes, alteraciones en la imagen de televisores, etc.

#### 2.4.1.2 Objetivo de trabajo

Para estudiar estas anomalías resulta esencial recurrir a una mayor evidencia experimental y sobre la base de esta motivación la Facultad de Ingeniería de la Universidad de Buenos Aires ha desarrollado más de un trabajo. Entre ellos se ha estudiado la medición y procesamiento de señales para la detección de anomalías de campo magnético<sup>12</sup> y se ha descripto la implementación de una estación de sensado para registrar emisiones de campo magnético en bandas TLF-ELF<sup>13</sup> (**Cap. 13.2**), contribuyendo así al estudio de su posible rol como precursores sísmicos.

El proyecto en cuestión consta de una serie de sensores de magnitudes relevantes como antenas para el sensado de perturbaciones electromagnéticas, un detector de gas radón para la detección del mismo ante la ruptura de rocas, un magnetómetro de protones para

el sensado del campo magnético terrestre estático y un sensor de vibraciones para detectar vibraciones mecánicas que pudieran estar relacionadas con posibles movimientos sísmicos. Todos estos sensores se conectan con un microcontrolador y se almacenan en una tarjeta SD (ver detalles en sección 14.2 del Anexo).

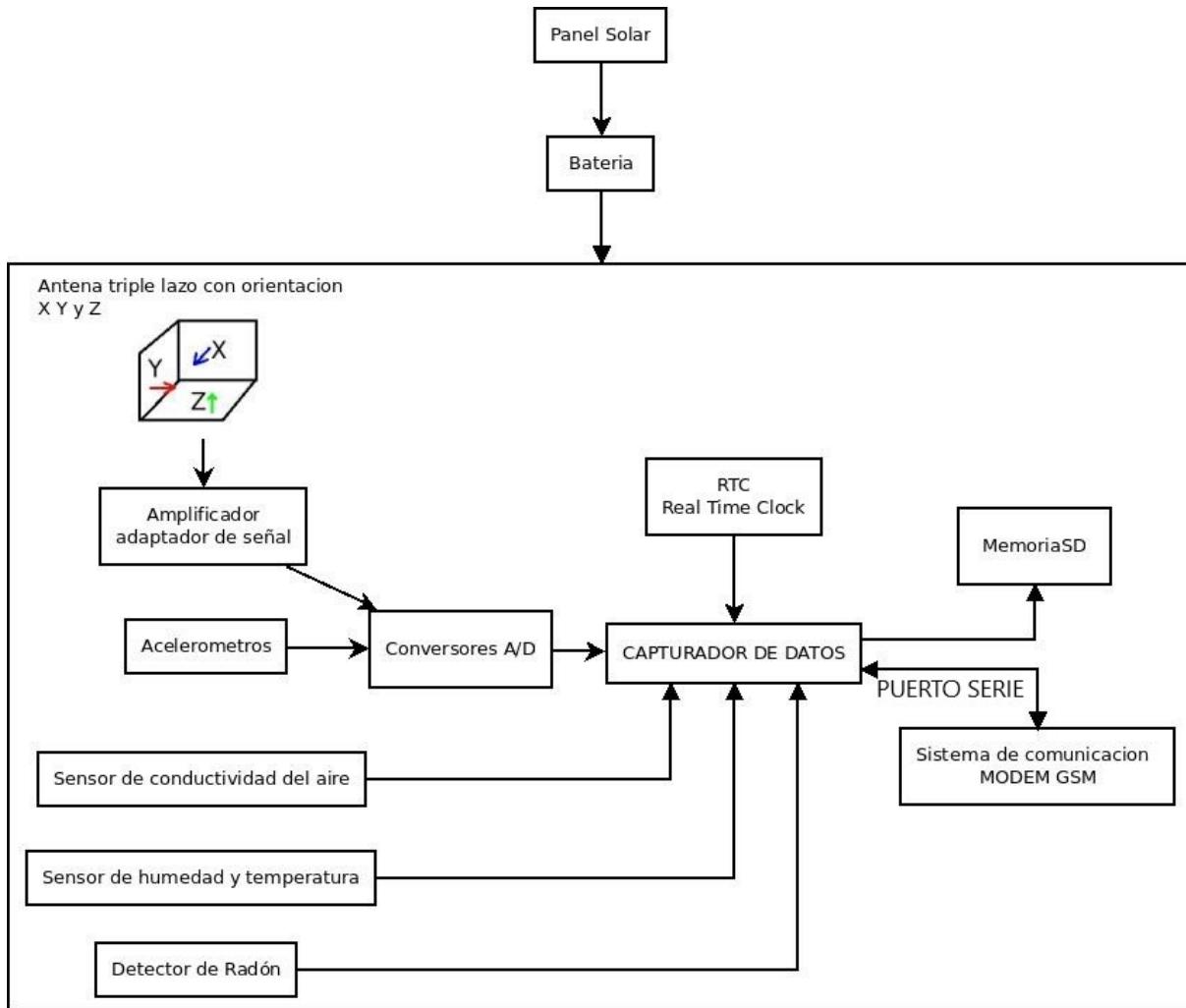


Figura 2.1: Diagrama en bloques del sistema de adquisición de datos sísmicos

Siguiendo esta línea, lo que se busca es poder transmitir y visualizar los datos capturados por los sismógrafos desarrollados con todas las características ya descriptas por nuestro proyecto. En la figura 2.1 se observa el diagrama en bloques completo del sistema de adquisición de datos. El bloque correspondiente a la comunicación es el que se lleva a cabo en el proyecto presentado en este informe. La forma de comunicar dicho bloque con el capturador de datos (microcontrolador) es a través del puerto serie del mismo conectado al del módulo intercomunicador correspondiente al presente proyecto (ver detalles en la sección 7.1.1). La transmisión automática desde el campo hacia el laboratorio es de suma importancia en esta temática al considerar que un precursor sísmico puede presentarse minutos antes del evento catastrófico.

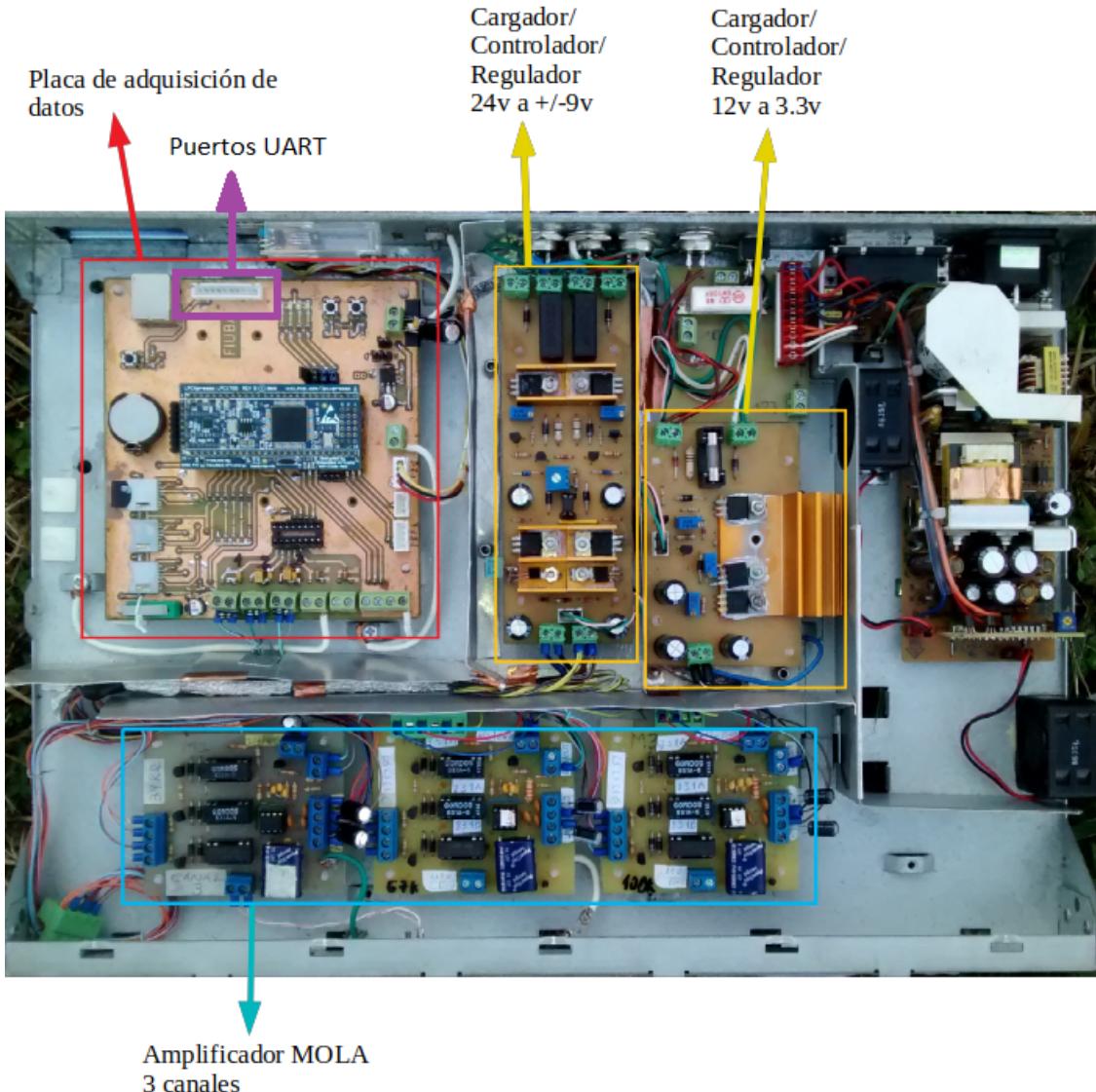


Figura 2.2: Equipo de medición de señales ELF-ULF

En la figura 2.2 se puede apreciar el equipo de medición diagramado en la figura 2.1. Se puede ver que existen puertos para la comunicación por protocolo UART que serán los necesarios para conectarse con el módulo intercomunicador. Los detalles pueden encontrarse en el capítulo 7.

## 2.4.2 Sensado de nivel de agua sobre puente vehicular

### 2.4.2.1 Necesidad detectada

El arroyo Las Piedras forma parte de la cuenca Sarandí-Santo Domingo, en la cual 550.000 personas viven en riesgo de inundabilidad. Esta carece de servicios de saneamiento apropiados, su desagüe industrial, cloacal y pluvial no tiene tratamiento previo, y los basurales en sus márgenes crean sitios puntuales de contaminación y diseminación de vectores. A la altura de la calle 52 B (Florencio Varela), el arroyo constituye una frontera entre los barrios “El Molino” (Florencio Varela) y “Eucaliptus” (Quilmes). En la actualidad, existe un puente vehicular que atraviesa el arroyo, construido hace veinticinco años

por los mismos vecinos cuando la cota de inundación del cauce era más baja. Sin embargo, en la actualidad, el arroyo aumenta su caudal considerablemente los días de lluvia intensa. El agua llega al interior de las casas y supera aproximadamente en un metro la cota del puente. La situación deja a las familias incomunicadas y obliga a las personas a cruzar el arroyo a pie (o a través de una precaria pasarela) o a trasladarse en busca de cruces más seguros, para ir a trabajar o a la escuela. En los últimos años, cuatro personas, entre ellas dos niños, fallecieron a causa de esta problemática.

Además, la contaminación del arroyo es visible, el puente retiene los residuos que el arroyo arrastra desde aguas arriba producto de la escasa o nula gestión de los mismos en muchas poblaciones de la cuenca e incluso provenientes de desagües industriales, cloacales y pluviales sin tratamiento adecuado. Sumado a la inexistencia de un sitio de disposición final apropiado conduce a la presencia de contaminación microbiológica y físico química que agrava aún más esta situación.

#### **2.4.2.2 Objetivo de trabajo**

El proyecto consiste en elevar el nivel del puente existente, construyendo por encima de este un nuevo puente de paso vehicular que garantice el acceso seguro de los vecinos en épocas de inundaciones, evitando así accidentes fatales. La construcción brindará múltiples beneficios para la comunidad, entre ellos:

1. Favorecer la circulación de los vecinos, especialmente, en épocas de inundaciones, facilitando el acceso a la educación y el trabajo de forma segura.
2. Fortalecer la integración de los vecinos y vecinas y la promoción de alianzas colaborativas entre los municipios, la asociación vecinal y otras organizaciones locales involucradas.
3. Mejorar las capacidades de adaptación de los vecinos frente al cambio global dado el esperable aumento de los fenómenos climáticos extremos que ocasionen lluvias más intensas y/o prolongadas en los próximos años.

Para la correcta construcción del puente, se está trabajando en un sensor de nivel para medir el nivel del agua y así obtener información para evitar inundaciones. A partir de la construcción de este sensor, el cual se está llevando a cabo por un laboratorio de la Facultad de Ingeniería. Este sensor cumple la misma función que el capturador de datos del ejemplo de precursores sísmicos en a subsección anterior, como así también de los sensores vistos en la Figura 1.1. El mismo se comunicará por puerto serie al módulo intercomunicador.

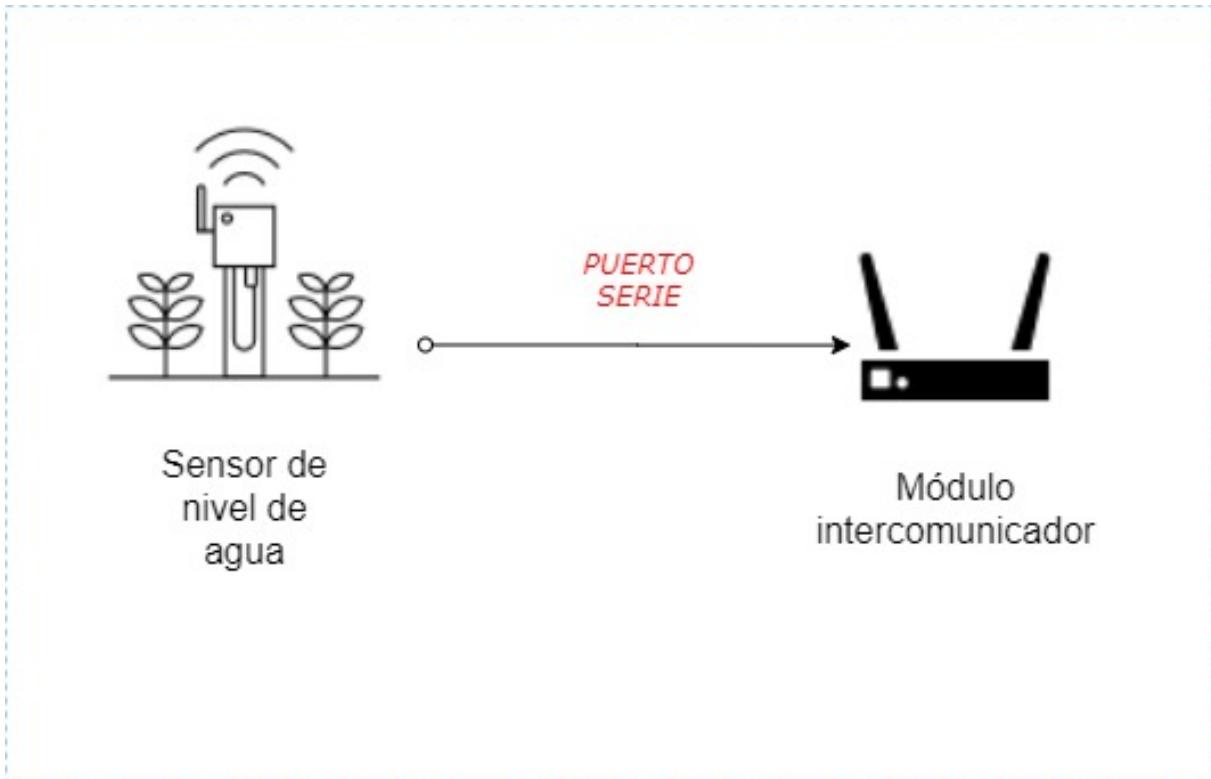


Figura 2.3: Comunicación del sensor de nivel de agua con el módulo intercomunicador

El proyecto tendrá la colaboración de la Asociación Civil Ingeniería Sin Fronteras, el Instituto Nacional del Agua y la Facultad de Ingeniería de la Universidad de Buenos Aires.

## 2.5 Resumen

Resulta evidente que existen muchas aplicaciones para este proyecto, lo que le da una gran versatilidad. Es por eso que desde el punto de vista de los objetivos, se busca que el diseño de la solución resulte lo más genérica posible, de forma tal que pueda aplicarse a cualquier tipo de monitoreo. Para esto son fundamentales poner el foco en los objetivos específicos que se marcaron en esta sección y contar con los conocimientos necesarios. Los mismos están relacionados entre si, por lo que se puede intuir que la falta de experiencia en alguno de ellos será complementada por el mayor conocimiento sobre otro.

# **Capítulo 3**

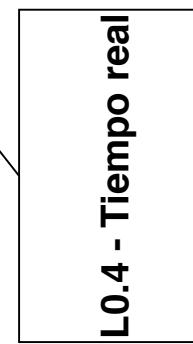
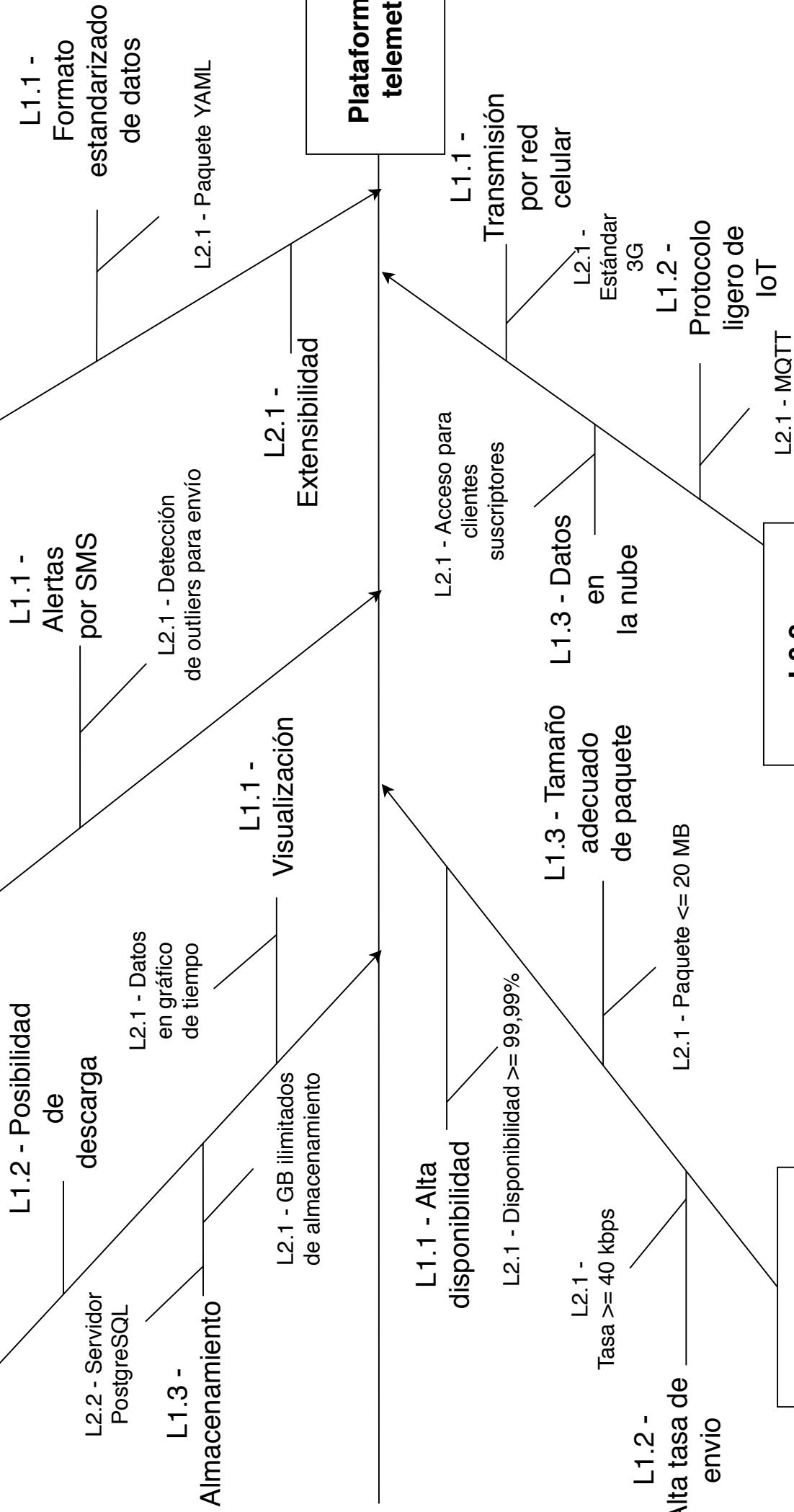
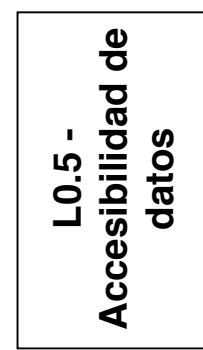
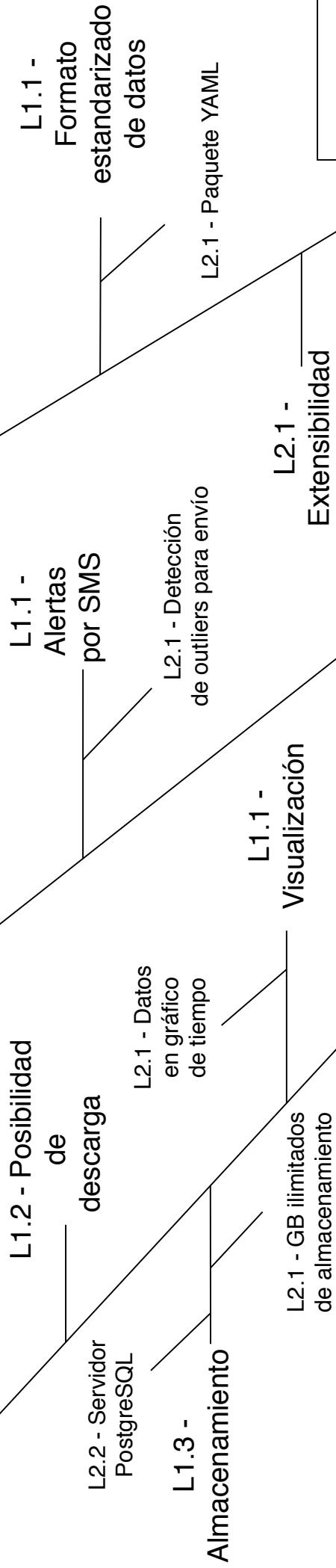
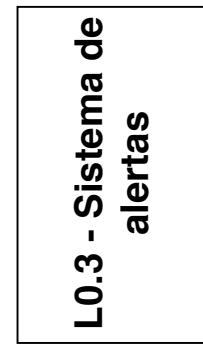
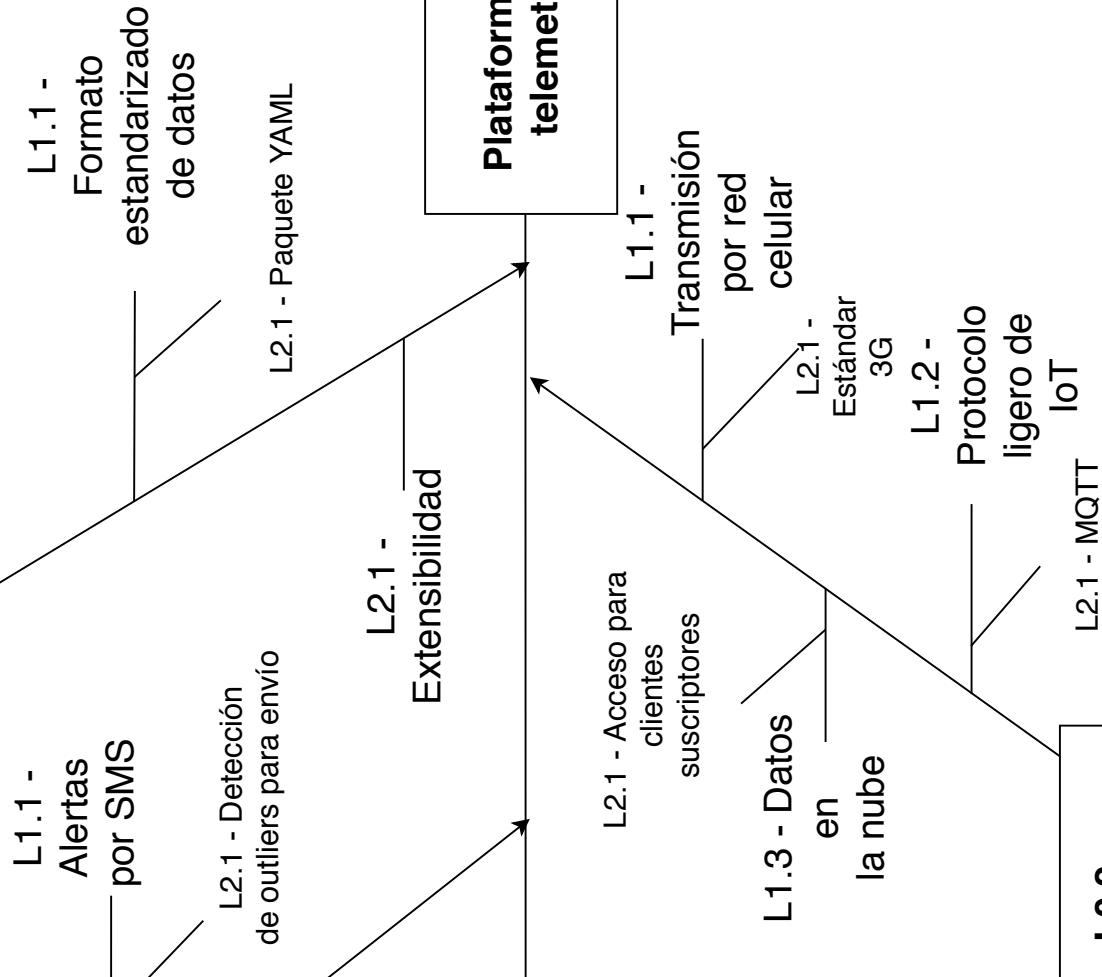
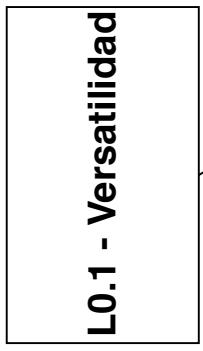
## **Definición del producto**

En este capítulo se hace un desglose de los requerimientos del usuario, yendo desde los de alto nivel, hasta los más técnicos que sirven para cumplir las necesidades. Se puede ver acá un mapa con este desglose y la explicación de cada uno.

Al final del capítulo se hace un análisis con una casa de calidad, de forma de encontrar las correlaciones entre los requerimientos y su importancia relativa respecto de la consideración del usuario final y su comparación con los productos de la competencia.

### **3.1 Requerimientos de usuario**

A continuación se presentan las demandas del usuario para la completitud de este proyecto:



En la figura se pueden apreciar los requerimientos del usuario que componen a la Plataforma de telemetría. Los mismos se presentan en un Diagrama de Ishikawa,<sup>14</sup> con el objetivo de separarlos en tres niveles, según el detalle técnico. Desde afuera hacia adentro, se distinguen tres niveles:

- **Nivel 0 (L0):** Requerimiento básico, entendible para el usuario final.
- **Nivel 1 (L1):** Requerimiento de nivel técnico básico.
- **Nivel 2 (L2):** Requerimiento de nivel técnico avanzado.

A continuación se explican brevemente cada uno de ellos:

### 3.1.1 L0.1 - Versatilidad

Se espera del producto que se pueda adaptar a distintas plataformas de monitoreo, de forma de que cualquier estación de sensado tenga la posibilidad de conectarse al mismo y los datos tomados por los sensores puedan ser transmimidos sin configuraciones adicionales.

- **L1.1 - Formato estandarizado de datos:** para que el producto sea "Plug&Play" (**Cap. 12.3**), es conveniente que los datos se encuentren en formato estandarizado, para evitar configuraciones adicionales en la entrada y salida de cada módulo.
  - L2.1 - Paquete YAML: este es uno de los estándares utilizados en software para la estructuración de datos en forma jerárquica.
- **L1.2 - Extensibilidad:** referido a la posibilidad de conectar al menos dos sensores al mismo tiempo al módulo intercomunicador.

### 3.1.2 L0.2 - Transmisión remota

Uno de los principales requerimientos solicitados por el cliente es que los datos puedan ser transmitidos desde una ubicación y accedidos desde otra, de forma de que haya una visualización sencilla de los datos sensados en lugares de difícil acceso.

- **L1.1 - Transmisión por red celular:** el sistema estándar GSM (**Cap. 13.3**) es común en gran parte del país y aprovechable para la transmisión de datos en lugares más alejados de centros urbanos.
  - L2.1 - 3G (**Cap. 13.3**): estándar que permite una conexión a internet por telefonía con una conexión USB.
- **L1.2 - Protocolo ligero de IoT:** el proyecto consiste de un dispositivo conectándose a internet para transmisión de datos, lo que se adapta a proyecto de IoT, como se mencionó anteriormente.
  - L2.1 - MQTT: se trata del protocolo que mejor se adapta al proyecto en cuestión.
- **L1.3 - Datos en la nube:** es necesario que los datos sean almacenados, ya sea de manera local como en la nube para completar la transmisión remota. En este caso se optó por la segunda opción.

- L2.1 - Acceso para suscriptores: el cliente que accede a los datos necesita poder acceder a estos datos en la nube. Para esto será necesario realizar configuraciones correspondientes para restringir solo a aquellos que lo soliciten.

### 3.1.3 L0.3 - Sistema de alertas

Con el fin de aprovechar la presentación de los datos, es deseo del cliente que exista un sistema de alertas que permita al mismo saber cuando algún parámetro sensado está por fuera de los resultados esperados.

- **L1.1 - Alertas por SMS:** es deseable que estas alertas sean recibidas via SMS, de manera de poder obtener la información al instante desde un teléfono celular.
  - L2.1 - Detección de outliers para envío: para poder enviar las alertas, se deben establecer límites que representen el valor máximo/mínimo para enviar una alerta.

### 3.1.4 L0.4 - Tiempo real

Un requisito importante es que los datos puedan ser transferidos en un tiempo tal que puedan ser visualizados pocos segundos después de ser capturados.

- **L1.1 - Alta disponibilidad (Cap. 12.3):** es necesario que el servicio tenga una alta disponibilidad, es decir, que el uptime (tiempo en que está disponible) represente un porcentaje cercano al 100% y el downtime (tiempo en que no está disponible) sea un porcentaje cercano 0%.
  - L2.1 - Disponibilidad  $\geq 99,9\%$
- **L1.2 - Alta tasa de envío:** es deseable que los kbps sea un parámetro adecuado de forma que los datos transmitidos lleguen en un tiempo corto.
  - L2.1 - Tasa  $\geq 40kbps$ : tasa necesaria para que un paquete de datos que representa una hora de sensado se envíen en menos de una hora.
- **L1.3 - Tamaño adecuado del paquete:** de la misma forma que la tasa, es necesario que el paquete de datos tenga un tamaño adecuado para no alcanzar el límite de ancho de banda de la red.
  - L2.1 - Paquete  $\leq 20MB$ : para poder enviar a una tasa alta y en menos de una hora, el límite del tamaño de paquete debe ser este.

### 3.1.5 L0.5 - Accesibilidad de datos

Una vez que los datos son transmitidos, es pertinente que estos puedan ser accedidos por el usuario, ya sea desde una interfaz como desde un portal en la nube.

- **L1.1 - Visualización:** es deseo del usuario que puedan ser visualizados, ya sea en formato de tabla, gráficos, etc.
  - L2.1 - Datos en gráfico de tiempo

- **L1.2 - Posibilidad de descarga:** con simple comando/botón, se quiere tener una forma de poder descargar los datos en un archivo, ya sea en formato csv, JSON, o cualquier otro formato.
- **L1.3 - Almacenamiento:** para que los datos sean accesibles en todo momento, se debe poder almacenarlos en algún lugar, ya sea en una base de datos, sea local o en la nube.
  - L2.1 - GB ilimitados de almacenamiento: dado que se toman muchas muestras por día, es necesario tener una alta capacidad de almacenamiento, en términos de GB. Para esto, es necesario tener una base de datos que permita esto.
  - L2.2 - Servidor PostgreSQL: base de datos de software libre que permite el almacenamiento de datos de manera local como en la nube.

## 3.2 Requerimientos técnicos

Para poder cumplir con esos requerimientos de usuarios, es necesario definir una serie de variables técnicas que guarden relación con los mismos y que actúen como condición para poder cumplirlos:

- **RT.1 Diseño modular:** se define por la cantidad de módulos a utilizar, lo que permite dividir al proyecto en etapas, configurables a la medida de cada requerimiento. Se buscan definir al menos tres módulos que representan las tres etapas de envío del proyecto, transmisor, agente y receptor.
- **RT.2 Autonomía (hs):** al ser un proyecto que toma datos desde una localización remota, probablemente con poca conectividad, es necesario que tenga autonomía de al menos 24 horas para poder transmitir una cantidad de muestras significativas, sin la necesidad de cargar el dispositivo, o de un cambio de baterías, según el caso.
- **RT.3 Tamaño de paquetes (KB):** es necesario que el tamaño de los paquetes de datos enviados sea menor a 17MB de forma tal que la red no se sature y se logren enviar en tiempo real.
- **RT.4 Velocidad de transmisión (kbps):** se desea que la cantidad de bits enviados por segundo sea mayor a 40 kbps de forma que los datos puedan ser enviados de un dispositivo a la nube en menos de 10 minutos.
- **RT.5 Costo (USD):** se desea poder tener un costo por unidad de producto menor a los 500 USD y con la posibilidad de disminuir o aumentar los costos según el modelo elegido y la calidad de los mismos, buscando tener un producto económico y de calidad.
- **RT.6 Almacenamiento (GB):** para que este proyecto sea viable y se logren cumplir los requerimientos de los usuarios, es necesario tener una gran capacidad de al menos 1 TB de datos de forma tal que se puedan enviar datos a lo largo de al menos un año.
- **RT.7 Consumo (mAh):** para que el módulo intercomunicador pueda lograr una alta autonomía, se debe lograr reducir el consumo del módulo intercomunicador en

al menos dos de sus módulos de forma que éste realice solo las operaciones necesarias para este proyecto.

- **RT.8 Paquetes en formato estandarizado:** para tener una solución profesional y que sea adaptable a cualquier proyecto de medición de magnitudes físicas, y considerando que se trabaja por módulos, es necesario garantizar la compatibilidad de estos módulos, de forma de no tener que hacer configuraciones adicionales en la entrada y salida de cada uno. Para lograr esto, los paquetes enviados deben poder ser interpretados por cualquier dispositivo/sistema operativo que lo reciba, de forma de reducir la cantidad de configuraciones en la entrada y la salida de cada módulo a cero.

### 3.3 Casa de calidad

La Casa de la Calidad es un diagrama, que se asemeja a una casa, utilizado para definir la relación entre los deseos de los clientes y las capacidades de las empresas/productos.<sup>15</sup> Se trata de una parte del Despliegue de la función calidad (QFD) y se utiliza una matriz de planificación para relacionar lo que el cliente quiere contra cómo una empresa (que produce los productos) va a cumplir esas necesidades. Se parece a una casa con una "matriz de correlación", como su techo, los deseos del cliente frente a las características del producto como la parte principal, la evaluación de la competencia como el porche etc. Se basa en "la creencia de que los productos deben ser diseñados para reflejar los deseos de los clientes y sus gustos".<sup>16</sup> También se ha informado que aumenta la integración funcional cruzada dentro de las organizaciones que la utilizan, sobre todo entre el marketing, la ingeniería y la fabricación.

#### 3.3.1 Matriz de interrelación

En la matriz que une los requerimientos del cliente con los técnicos, se establecen los grados de interrelación entre los mismos, siendo 1 (baja), 4 (media), 9 (alta) y vacío (no existe relación). Por ejemplo, para que se pueda cumplir con el envío de los datos en tiempo real, se debe tener una velocidad de transmisión alta, por eso se relacionan altamente.

#### 3.3.2 Matriz de correlaciones técnicas

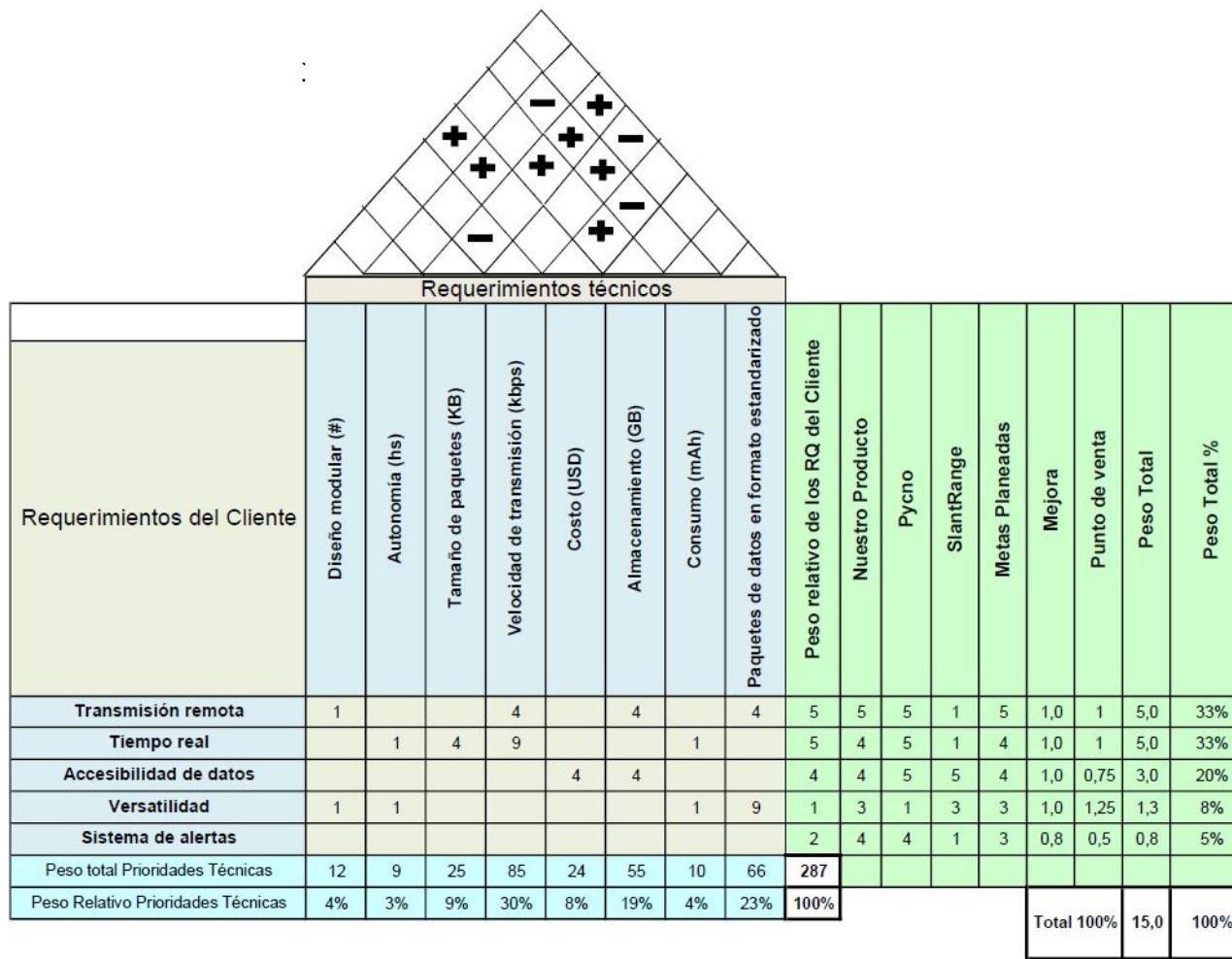
Se busca en el "techo" de la casa relacionar las especificaciones técnicas, indicando con "+" aquellas que tienen relación directa y con "-" los que tienen una relación inversa. Por ejemplo, cuanto menor sea el consumo (mAh) del módulo intercomunicador, mayor será la autonomía (hs) del mismo.

#### 3.3.3 Matriz de evaluación del mercado

En esta sección, que es la que se ve a la derecha de la casa, los clientes nos van a evaluar cómo nos encontramos nosotros respecto a las demás empresas del mercado que nos son competencia directa, respecto a los requerimientos del cliente.

### **3.3.4 Evaluación técnica**

Nos indica cómo nos encontramos nosotros en el mercado respecto a los requerimientos técnicos. Es la sección que se sitúa en la parte baja de la casa.



Mejoras = Planeado / Nuestro Producto

Peso Total = Peso Relat.Cliente x Mejora x Pto.de Venta

Peso Total % = Peso Total / Suma Pesos Totales

Puntuación Mapeo RQ del Cliente en Especificaciones

1 Baja inter-relación

4 Media inter-relación

9 Alta inter-relación

Peso total Prioridades Técnicas =  $\sum$  Pesos Totales x Puntuación Mapeo de RQ en Espec.

Se puede ver que hay una diferencia a favor en cuanto a los pesos con respecto de la competencia. Vale aclarar que en este caso, el producto en cuestión consiste en una solución genérica, es decir que es adaptable a distintas áreas, en términos de magnitudes físicas. Esto la vuelve un producto con un rango de consumidores más amplio que el de los competidores, pues los mismos están más enfocados al agro. Al mismo tiempo, existen requerimientos, además de la versatilidad, como el acceso a los datos en tiempo real, los cuales nuestro producto puede cubrir mejor que los otros. Esto, creen los autores, es lo que posiciona mejor a esta solución en el mercado, siendo más robusta y de mayor calidad.

Se ve al mismo tiempo relaciones entre los requerimientos que son de gran importancia. Como se ve, existen muchos de los técnicos que son necesarios para cubrir muchos de los del cliente. Ejemplos de esto son la velocidad de transmisión y el formato estándar de los paquetes, fundamentales para lograr cumplir el tiempo real y la versatilidad como requerimientos del cliente. Al mismo tiempo, varios de los que se ven arriba están íntimamente relacionados, es decir que aumentando o reduciendo cualquiera de ellos, sube o baja alguno de los otros. Por ejemplo, cuánto más consumo, menor es la autonomía. Es importante tener en cuenta esto para el diseño, de forma de balancear cada uno de estos requerimientos para obtener la mejor solución que se adapte a los requerimientos del cliente.

## 3.4 Resumen

Es importante separar los requerimientos de alto nivel, los cuales son los que pide el usuario final (el cual puede tener o no conocimientos técnicos), de los de más bajo, los cuales son los que a medida que se va desarrollando el proyecto, se vuelven necesarios para su correcta implementación y para alcanzar los objetivos más genéricos. También identificar los requerimientos técnicos necesarios de forma de poder hacer el corte orgánico de los mismos, a fin de poder empezar con pequeños módulos e ir haciendo cambios incrementales que permitan llegar al objetivo. Por ejemplo: se quiere llegar a tener datos en tiempo real. Para esto se debe lograr una buena velocidad de transmisión de datos (entre otras cosas). Será importante asegurarse primero de validar la correcta transmisión de datos (primer MVP -Producto de Mínimo Valor-); luego, como primer cambio incremental asegurarse de usar un protocolo ligero (MVP 2); luego eliminar todo dato inservible que se esté transmitiendo (MVP 3). Así y con otros incrementos adicionales se puede llegar a cumplir el requerimiento del usuario.

# Capítulo 4

## Análisis de factibilidad temporal

Se revisa en este capítulo el alcance del proyecto, introduciendo los objetivos de manera SMART, los cuales son los mismos que en el Plan de Trabajo Profesional. Se establece la relación entre estos y los requerimientos de usuario y el desglose de tareas a lo largo del período de trabajo. Esto se hizo a través de un diagrama de Gantt.

### 4.1 Alcance del proyecto

A continuación se define el alcance del proyecto con la intención de presentar objetivos **SMART** (Específicos -S-, Medibles -M-, Alcanzables -A-, Relevantes -R-, Tiempo Orientados -T-). Existe un *paper* escrito por Georfe T. Doran llamado "*There's a S.M.A.R.T. way to write management's goals and objectives.*",<sup>17</sup> donde explica la importancia de los objetivos y la dificultad de escribirlos.

- S.1** Utilizar un módulo intercomunicador para capturar datos de baja y alta frecuencia enviados desde la red de sensores, en paquetes con un formato determinado por el grupo y el equipo de investigación que toma los datos. El envío se realiza por el protocolo adecuado para el tamaño de los paquetes.

**Tiempo estimado:** 1,5 meses.

Para este objetivo, las tareas más importantes consisten en el armado de los paquetes de datos con el formato adecuado, implementando los scripts de captura correspondientes que lean los mismos en forma periódica. Establecidas estas pautas, la integridad de los datos será verificada almacenándolos en archivos dentro de un repositorio, de forma de tener un registro de lo que se envió.

Este objetivo se encuentra ligado tanto al requerimiento L0.1 como al L0.2.

- S.2** Desarrollar un sistema de software automático que envíe los datos periódicamente mediante un protocolo de envío ligero desde el módulo intercomunicador hacia la nube. Desarrollar un sistema análogo que reciba los datos desde otra ubicación con el formato que fueron enviados y bajo la misma tasa.

**Tiempo estimado:** 2 meses.

Las claves de este punto están en el montaje de los tres agentes principales para la comunicación: el **publicador**, el **servicio** y el **suscriptor**. El primero será el que envía los datos, es decir, el mismo intercomunicador que los obtiene de la red de sensores. El servicio será montado en la nube utilizando un host gratuito y

funcionará como intermediario entre los clientes. Quien suscribe será cualquier dispositivo conectado a la red desde cualquier lugar remoto. La calidad de este objetivo se verificará mediante la comprobación de que los datos publicados por el módulo intercomunicador se mantengan íntegros a la hora de leerlos desde el suscriptor. Este objetivo es fundamental y su correcto desarrollo tiene impacto en los requerimientos L0.1, L0.2 y L0.4.

- S.3** *Procesar los datos utilizando los filtros correspondientes similar para obtener una visualización en tiempo y frecuencia del comportamiento de las señales obtenidas. Las imágenes serán guardadas en un repositorio compartido por los miembros del grupo, sus tutores y colaboradores.*

**Tiempo estimado:** 2 meses.

Utilizando una PC con los recursos suficientes, se procederá a realizar una biblioteca de filtros necesarios para el procesamiento de los datos recibidos. El objetivo la obtención de imágenes en tiempo y frecuencia para el análisis posterior por parte del usuario final.

Este objetivo esta estrechamente relacionado a la accesibilidad de los datos expresada en el requerimiento L0.5.

- S.4** *Publicar una página web utilizando un hosting alojado en la nube de forma de publicar los datos procesados previamente y que brinde la opción de descargar y visualizar los datos correspondientes a un período de tiempo. El servidor puede ser propio (LABi) o un hosting gratuito.*

**Tiempo estimado:** 3 meses.

Las tareas principales para la realización de este objetivo consistirán en la obtención de un hosting para el servicio web, como así también el correcto envío de los datos en forma periódica al mismo. Una vez que los mismos estén subidos, se debe establecer un mecanismo de autenticación para quien desee visualizarlos/descargarlos, y verificar su integridad.

Este objetivo es clave para la exitosa resolución del requerimiento L0.5, atacando tanto la visualización como la descarga de los datos.

- S.5** *Stretch: mediante algoritmos de detección de anomalías y detección de outliers, generar alertas que se disparen periódicamente desde el servidor hacia los usuarios mediante email o SMS al teléfono móvil.*

**Tiempo estimado:** 1.5 meses

Se define a este objetivo como *stretch*, dado que se lo piensa como un objetivo ampliado, que se busca lograr siempre y cuando se logren los anteriores. Para lograrlo, se deben elegir al menos dos algoritmos que sirvan para la detección de anomalías o outliers y la configuración de un endpoint para el envío de alertas a los dispositivos que se suscriban a las mismas. Por último, este objetivo resuelve el requerimiento L0.3 del sistema de alertas.

## 4.2 Márgenes de tareas

A continuación se indican **orientativamente** los distintos entregables en una estructura WBS (*Work Breakdown Structure*).

Entregable	Tarea	Duración [días]
1. Comunicación sensores - módulo intercomunicador	Establecer formato del paquete	15,00
	Establecer velocidad de transmisión	7,00
	Validación de formato con el cliente	6,00
	Ánalisis de medios de transmisión	15,00
	Prueba integral de envío sincrónico	5,00
2. Comunicación intercomunicador - nube	Adquisición de módulo de conexión Wi-Fi	5,00
	Creación de broker MQTT	7,00
	Desarrollo de software de publicación	30,00
	Prueba de publicación de datos por Wi-Fi	5,00
	Adquisición de módulo de conexión 3G	5,00
	Agregado de funcionalidad 3G	20,00
	Prueba de publicación de datos por 3G	5,00
	Desarrollo de software de suscripción	35,00
	Prueba integral de publicación y suscripción	5,00
3. Procesamiento de datos	Configuración para publicación periódica	7,00
	Ánalisis de almacenamiento en la nube vs. local	7,00
	Diseño conceptual de la base de datos	7,00
	Desarrollo y creación de la base de datos	21,00
	Prueba de almacenamiento directo	5,00
	Prueba de restricciones de la base y casos borde	5,00
	Desarrollo de API para control de acceso	25,00
4. Publicación web	Prueba de almacenamiento con API	5,00
	Integración con proceso de transferencia MQTT	10,00
	Prueba integral con transferencia MQTT	5,00
	Análisis de soluciones existentes	10,00
	Análisis de hosts para desarrollo propio	6,00
5. Telecomando	Implementación de la alternativa elegida	30,00
	Estudio de los canales de alerta	10,00
	Implementación de las alertas	25,00
	Integración con consumo de base de datos	10,00
	Modificación de software de publicación para reuso inverso	20,00
6. Calibrador	Modificación de software de suscripción para reuso inverso	20,00
	Agregado de opción de telecomando a la interfaz	5,00
	Creación de scripts de tarea específica (ej: reinicio)	15,00
	Prueba de envío de telecomando	5,00
	Prueba de script de la tarea	5,00
	Prueba de integración	5,00
	Diseño de casos de prueba	10,00
	Adquisición de Arduino	5,00
	Desarrollo del software de los casos de ejemplo	25,00
	Prueba integral con la plataforma de telemetría	5,00
	Pruebas de tasa de envío	5,00
	Pruebas de distintos tamaños de paquete	5,00
	Prueba larga duración - Análisis de disponibilidad	60,00

Tabla 4.1: WBS - Márgenes de tareas

## 4.3 Diagrama de Gantt

El **Diagrama de Gantt**<sup>20,21</sup> es una herramienta gráfica cuyo objetivo es exponer el tiempo de dedicación previsto para diferentes tareas o actividades a lo largo de un tiempo total determinado. A pesar de esto, el diagrama de Gantt no indica las relaciones existentes entre actividades.

A continuación se presenta la distribución de las tareas que implicaron este proyecto en un Diagrama de Gantt.

1. Comunicación sensores -...	 Establecer formato del paquete (100%)					
1. Comunicación sensores -...	 Establecer velocidad de transmisión (100%)					
1. Comunicación sensores -...	 Validación de formato con el cliente (100%)					
1. Comunicación sensores -...	 Análisis de medios de transmisión (100%)					
1. Comunicación sensores -...	 Prueba integral de envío sincrónico (100%)					
2. Comunicación intercomuni...		 Adquisición de módulo de conexión Wi-Fi (100%)				
2. Comunicación intercomuni...	Creación de broker MQTT (100%)					
2. Comunicación intercomuni...	Desarrollo de software de publicación (100%)					
2. Comunicación intercomuni...		Prueba de publicación de datos por Wi-Fi (100%)				
2. Comunicación intercomuni...		Adquisición de módulo de conexión 3G (100%)				
2. Comunicación intercomuni...		Agregado de funcionalidad 3G (100%)				
2. Comunicación intercomuni...			Prueba de publicación de datos por 3G (100%)			
2. Comunicación intercomuni...				Desarrollo de software de suscripción (100%)		
2. Comunicación intercomuni...					Prueba integral de publicación y suscripción (100%)	
2. Comunicación intercomuni...					Configuración para publicación periodica (100%)	

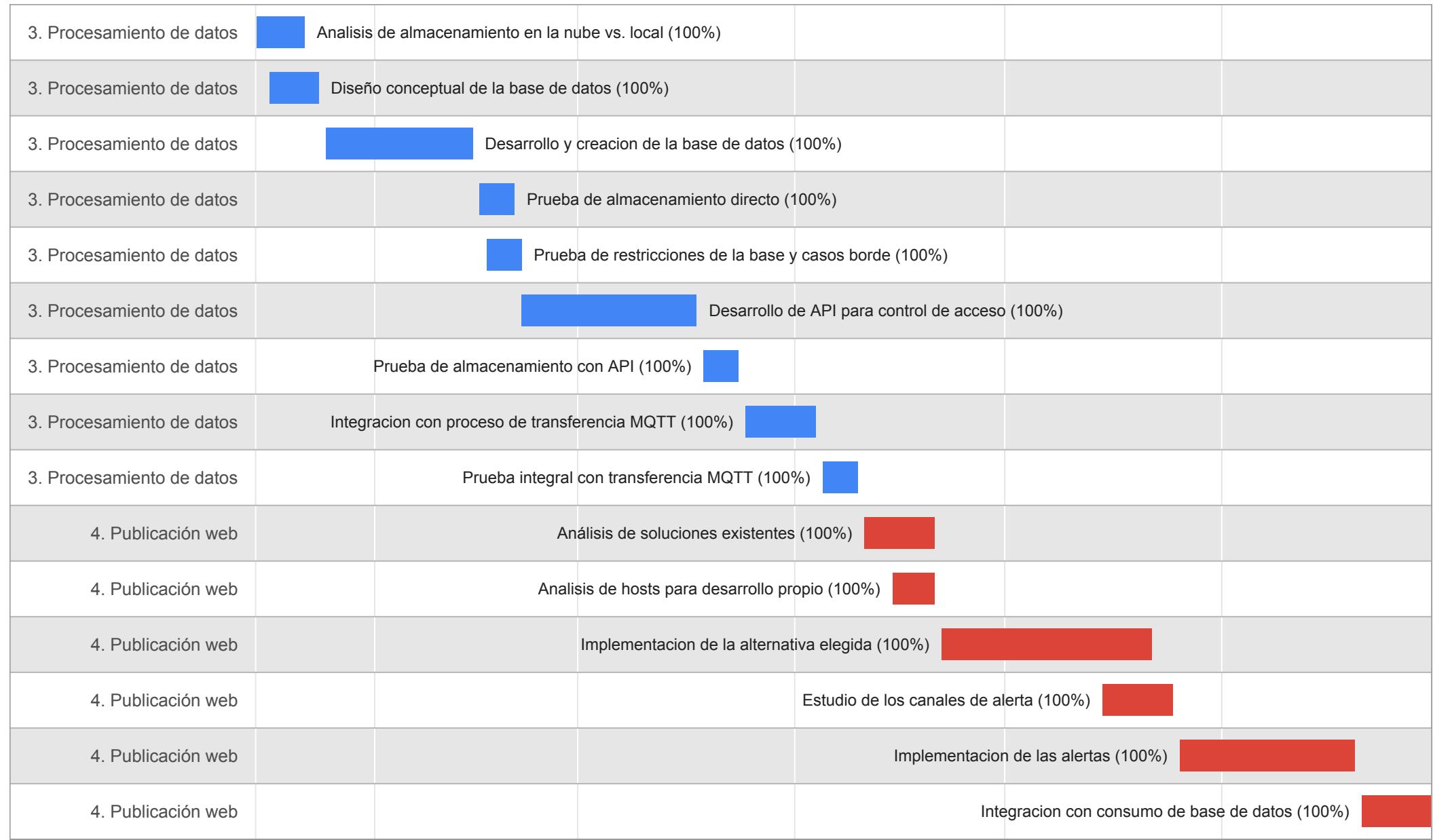
Sep  
2019

Oct

Nov

Dec

Jan  
2020



Feb  
2020

Mar

Apr

May

Jun

Jul

5. Telemando	<div style="width: 100%; background-color: #0070C0;"></div>	Modificacion de software de publicación para reuso inverso (100%)			
5. Telemando	<div style="width: 100%; background-color: #0070C0;"></div>	Modificacion de software de suscripcion para reuso inverso (100%)			
5. Telemando	<div style="width: 25%; background-color: #0070C0;"></div>	Agregado de opcion de telemando a la interfaz (100%)			
5. Telemando	<div style="width: 50%; background-color: #0070C0;"></div>	Creacion de scripts de tarea especifica (ej: reinicio) (100%)			
5. Telemando	Prueba de envio de telemando (100%)	<div style="width: 10%; background-color: #0070C0;"></div>			
5. Telemando	Prueba de script de la tarea (100%)	<div style="width: 10%; background-color: #0070C0;"></div>			
5. Telemando	Prueba de integracion (100%)	<div style="width: 10%; background-color: #0070C0;"></div>			
6. Calibrador	Diseño de casos de prueba (100%)	<div style="width: 10%; background-color: #C00000;"></div>			
6. Calibrador	Adquisicion de Arduino (100%)	<div style="width: 10%; background-color: #C00000;"></div>			
6. Calibrador	Desarrollo del software de los casos de ejemplo (100%)	<div style="width: 100%; background-color: #C00000;"></div>			
6. Calibrador	Prueba integral con la plataforma de telemetria (100%)	<div style="width: 10%; background-color: #C00000;"></div>			
6. Calibrador	Pruebas de tasa de envio (100%)	<div style="width: 10%; background-color: #C00000;"></div>			
6. Calibrador	Pruebas de distintos tamaños de paquete (100%)	<div style="width: 10%; background-color: #C00000;"></div>			
6. Calibrador			Prueba larga duracion - Analisis de disponibilidad (100%)		

Aug  
2020

Sep

Oct

Nov

Dec

## 4.4 Resumen

La correcta definición del alcance de un proyecto es uno de los puntos más importantes a la hora de su planificación. Aquí quedan definidos los entregables que deberán estar cumplidos al final del proyecto, por lo que si el alcance se define vagamente, el objetivo final será difuso. Para obtener una definición acertada y que lleve a un objetivo final claro se utilizó el criterio SMART en la confección de cada entregable.

Una vez obtenidos los objetivos y gracias a su correcta definición es posible dividirlos en tareas más pequeñas que concluyan en cada uno de los entregables mayores definidos previamente. Este trabajo es el denominado WBS (*Work Breakdown Structure*) presentado en la sección 4.2.

Finalmente, las tareas definidas en el WBS se orden temporalmente en un diagrama de Gantt para identificar cualquier dependencia existente y obtener un tiempo estimado para la finalización total del proyecto.

# Capítulo 5

## Análisis de factibilidad económica

Este capítulo comienza por un análisis de los principales competidores, distintos productos/servicios que brindan una solución similar a la que se presenta en este informe. Lo más destacado de esta sección implica el análisis de costos realizado, que brinda al usuario información de cuál será la inversión necesaria para llevar adelante este proyecto. Por último, se presentan los indicadores económicos como el VAN y el TIR, en base a la estimación de los volúmenes de venta, para calcular los tiempos de retorno de la inversión del proyecto en cuestión.

### 5.1 Análisis competitivo

#### 5.1.1 Mercado

El *market share* (**Cap.12.4**) de este producto es una parte importante a la hora de hacer un análisis de mercado. Hay que analizar cómo es el mismo, y esto incluye varios aspectos: los posibles clientes, la existencia de competencia, los canales de venta, la dimensión del mercado y el posicionamiento que se quiere para el producto.

Entonces, para poder estimar cuál será la participación en el mercado hay que analizar los últimos dos items mencionados antes, ya que sobre los clientes, la competencia y los canales de venta ya se explicó en las secciones previas.

Dentro del ámbito público, existen diversas instituciones dedicadas a la investigación científica. Tanto dentro del ámbito académico como fuera del mismo, existen múltiples proyectos que se adaptan al alcance de este trabajo. Por mencionar algunos ejemplos, la FIUBA cuenta con muchos laboratorios dedicados a la investigación y desarrollo de sistemas de medición de magnitudes físicas, como el LPSC,<sup>25</sup> el LRAD<sup>26</sup> o el LSE,<sup>27</sup> entre otros. Estos mismos trabajan con miembros de la IEEE<sup>28</sup> (**Cap. 13.4**), una asociación mundial de ingenieros dedicada a la normalización y el desarrollo en áreas técnicas.

Por otro lado, existen muchas instituciones públicas dedicadas al desarrollo de proyectos de infraestructura y tecnología en distintos ámbitos. Un ejemplo claro es el mencionado en la sección 2.4.2, el cual está a cargo de el INA<sup>29</sup> (**Cap. 13.4** ), dedicado a obras de infraestructura hídrica en distintos sectores del país.

El ámbito privado es otro de los motores del mercado. En el país existen muchas empresas dedicadas a distintos rubros de la economía. Una de las más importantes son las petroleras, las cuales cuentan con diversas líneas de negocio que se adaptan a este proyecto. Por citar un ejemplo, existen dispositivos que utilizan estas empresas en las refinerías, dedicados a la captura de mediciones acústicas en los procesos, para detectar pérdidas de fluidos.

### 5.1.2 Competidores

Si bien hay diversos proyectos independientes similares, no se ha encontrado uno comercializado con el mismo foco que este proyecto presenta. Se encontraron empresas que con las mismas tecnologías brindan servicios de sensado pero, a diferencia de nuestro producto, estos funcionan en áreas con buena conectividad. Además, a diferencia del producto descripto en este informe, están enfocados en áreas particulares como la agricultura y no a un nivel genérico. Algunos competidores se describen a continuación.

#### *Pycno*<sup>30</sup>

Pycno ofrece un sensor que mide las condiciones de los cultivos de manera remota sin ningún tipo de mantenimiento para que el agricultor pueda tomar mejores decisiones.

Al igual que nuestro producto, el suyo logra comunicar los datos sensados en áreas sin WI-FI utilizando una tarjeta SIM (**Cap. 13.4**) embebida en sus sensores. La diferencia es que se dedican al sensado, análisis y control con el foco puesto en la mejora de la producción agrícola. Si bien toman datos del suelo estos se relacionan con la humedad, PH y fertilización. Sus sensores hechos especialmente para la aplicación los comercializan por U\$S 350 (cada uno) y la base de control centralizado por U\$S 150, además de un posible costo mensual por la tarjeta SIM.

Algunas de las características que pueden ser un punto de comparación con el presente proyecto, son:

- **Conectividad:** como se mencionó antes, los sensores están ubicados en el campo, por lo que no pueden utilizar redes Wi-Fi. Entonces aprovechan las redes GSM, utilizando tanto 3G, como así también 2G y 4G, según el tipo de sensor que se esté utilizando.
- **Parámetros físicos:** respecto a esto, la finalidad de este proyecto está destinado al sensado de variables que tienen impacto en el agro, como radiación solar, temperatura y humedad del aire y suelo, entre otras.
- **Análisis de datos:** la solución de Pycno ofrece al usuario un portal en el cual se puede acceder a los datos almacenados, visualizar la distribución espacial de los sensores, hacer predicciones sobre los distintos parámetros y controlar los sensores.

#### *SlantRange*<sup>31</sup>

También se dedican al análisis de los suelos pero desde el cielo. La plataforma de análisis se renueva anualmente y el monto puede subir hasta U\$S 3000 por año, según que se contrate. Adicionalmente venden sensores especiales para su aplicación por U\$S 5000.

En cuanto a las características, se pueden mencionar las siguientes:

- **Extracción de datos:** como se mencionó en el párrafo anterior, la toma de datos se hace de manera aérea, tomando imágenes con tecnologías incorporadas en drones, utilizando sensores de tipo CMOS, como así también sensores LIDAR (**Cap. 13.4**).
- **Almacenamiento:** en este caso comunican almacenan los datos sensados en una tarjeta SD que vienen integradas en los sensores y luego los traspasan de forma manual para su estudio.
- **Análisis de datos:** SlantRange ofrece un paquete de análisis de datos que viene incorporado con distintas funciones que permiten el uso de algoritmos de filtrado y compresión de datos, para la detección de patrones, y configurable por el usuario.

Adicionalmente, una empresa que podría proveer servicios similares al nuestro:

*Telemetrik*<sup>32</sup>

Podría ser una de las principales competencias, es una empresa colombiana especializada en diseño, desarrollo, implantación, comercialización y operación de soluciones de telemetría, automatización, instrumentación, monitoreo a distancia. Dentro de sus casos de éxito se encuentran: *"Medición en tiempo real de caudal y de nivel en el Acueducto"* y *"Medición de temperatura y humedad en tiempo real en neveras de medicamentos de alto costo"*. Ambos proyectos presentan características similares a las de nuestra solución, transmisión de datos de manera inalámbrica, información en tiempo real de las variables y alarmas programables.

## 5.2 Ciclo de vida

La figura 5.1 muestra el ciclo de vida de un producto genérico, caracterizado por 4 etapas.

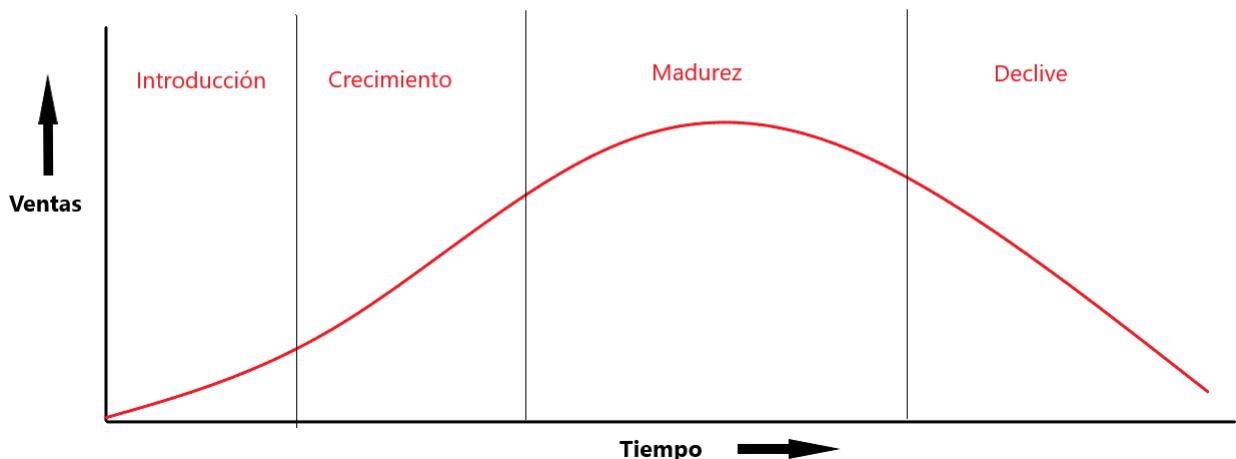


Figura 5.1: Ciclo de vida de un proyecto.

- Introducción: caracterizado por bajas y perdidas financieras debido a la introducción del innovador producto al mercado. Por lo general puede que haya poca competencia y los consumidores están buscando productos que cubran necesidades no cubiertas. Estimamos este tiempo en 6 meses.
- Crecimiento: El producto es aceptado en el mercado y reconocido por más clientes, aumentando el numero de ventas e incrementando las ganancias. Puede ser que en esta etapa se sumen nuevos competidores. Estimamos que este periodo durara 1 año.
- Madurez: Es el periodo donde las ventas llegan a su pico y los beneficios son máximos. Se caracteriza por ser la etapa donde el costo por cliente es el más bajo. Es donde se esta más expuesto a los mercados y donde hay más competidores. Se espera llegar al pico de ventas a los 2 años, siendo la duración total de esta etapa 3 años.
- Declive: Disminuyen las ventas y los beneficios, debido a la introducción de nuevas tecnologías y nuevas tendencias. Se espera llegar al final de esta etapa en el sexto año.

### **5.3 Estimación de costos**

Al ser un proyecto de carácter modular, el costo se puede estimar por partes, teniendo en cuenta los recursos materiales y software que se debe comprar. Existe variedad de opciones para cada uno de ellos, lo que permite tener opciones más económicas. A continuación se hace una estimación de cada una de estas para los costos en los primeros seis meses de trabajo.

En la tabla presentada a continuación se detallan los costos asociados al **RT.5**.

Módulo	Producto/ Servicio	Costo (USD)	Comentarios
Intercomunicador	(*) (**) (*** ) Raspberry Pi 3	40	
	(*) (**) (*** ) Cable serial	3	
	(*) (**) (*** ) Servicio 3G prepago	4,25 /mes	
	(***) (*** ) Batería LiPo 5000 mAh	20	
	(***) (*** ) Panel solar 22 W	105	
	(**) PiJuice	60	Placa para alimentar RP ( <b>Cap. 13.2</b> ) con batería
	(***) Powerboost 500 (5v USB @ 500mA)	10	Opción InHouse para alimentar a batería
	(***) Solar Lithium Ion/Polymer charger - v2	17,5	
	(***) Male DC Power adapter - 2.1mm plug to screw terminal block	2	
	(***) Envío de productos	35	
Total (6 meses)	(*)	68,5	Alimentación con red eléctrica
	(**)	288,5	Alimentación a batería usando PiJuice
	(***)	258	Alimentación a batería usando diseño propio
Recepcion&Almacenamiento	Suscripción Heroku	-	Servicio Cloud
	<sup>34</sup> Broker MQTT Cute Cat (1)	-	10 kbps 5 conexiones
	Broker MQTT Humble Hedgehog (2)	5 /mes	20 kbps 25 conexiones
	Broker MQTT Keen Koala (3)	19 /mes	Autenticación personalizada SSL/TLS Bridges 100 conexiones
	<sup>35</sup> Base de Datos PostgreSQL Dev (a)	-	Autenticación personalizada SSL/TLS Bridges 10.000 filas
	Base de Datos PostgreSQL Basic (b)	9 /mes	20 conexiones 10.000.000 filas
	Base de Datos PostgreSQL Standard (c)	50 /mes	20 conexiones 64 GB 120 conexiones
	(1)+(a)	-	
	(1)+(b)	54	
	(1)+(c)	300	
Total (6 meses)	(2)+(a)	30	
	(2)+(b)	84	
	(2)+(c)	330	
	(3)+(a)	114	
	(3)+(b)	168	
	(3)+(c)	414	
Visualización & Alertas	Grafana Enterprise Open Source	-	Debe correrse en servidor local y se accede dentro de la misma LAN
	<sup>33</sup> Grafana Cloud Starter	-	1 usuario 5 dashboards
	Grafana Cloud Standard	49 /mes	Se corre en una instancia en la nube 10 usuarios Dashboards ilimitados URL personalizado
	Enterprise Open Source	-	
Total (6 meses)	Cloud Starter	-	
	Cloud Standard	294	

Tabla 5.1: Estimación de costos

Se puede observar en la tabla que existen varias opciones acordes a las necesidades del usuario. Analizando los distintos casos, se podría hacer una estimación de cuánto sería la inversión inicial. Por ejemplo, suponiendo un caso donde se cuenta con acceso a la red eléctrica y cuya tasa de datos es de no más de 6 mediciones por día de unos pocos parámetros, como nivel de agua de un río, se podría elegir la opción que no contempla el uso de baterías y una opción de baja tasa de envío (teniendo en cuenta que son pocos datos) y bajo almacenamiento. Hasta aquí, la inversión inicial sería de 70 dólares aproximadamente para los 6 meses iniciales. Si se cuenta con un servidor local donde hostear la instancia de Grafana para la visualización de los datos, el costo es nulo y no subiría de ese monto. Si se quiere almacenar en la nube con acceso a varios usuarios, se deben sumar los 294 dólares de costo inicial, dando un total de 364 dólares de inversión.

Suponiendo un caso distinto, donde las mediciones implican el desarrollo a campo abierto, sin acceso al suministro eléctrico, es mejor optar por una alimentación a batería duradera. PiJuice es una opción confiable para este tipo de casos ya que es perfectamente compatible con la Raspberry Pi e implicaría poco mantenimiento (teniendo en cuenta la locación remota). Tomando en cuenta también que se toman mediciones a una frecuencia alta, la tasa de datos sería considerablemente mayor al ejemplo anterior y por lo tanto se necesitaría un envío más veloz y un almacenamiento tal que permita guardar la cantidad de datos necesaria. Suponiendo también que los datos se necesitan visualizar en tiempo real, la tasa de envío debe ser alta. En este caso se optaría por un almacenamiento de alta capacidad y tasa considerable, con lo que se podría elegir la opción (2)+(c), que son 330 dólares, para los primeros seis meses. De la misma forma que antes, se puede optar por la opción de visualización local como por la de la Nube. En este caso se sumarían los 294 dólares, dando una inversión inicial de 624 dólares.

Como se aprecia, la alternativa es económica dado que se pueden concentrar las mediciones de muchos sensores en un solo módulo intercomunicador que envíe los datos de todos, lo que permite invertir en poco hardware.

En particular, los costos de almacenamiento van en concordancia con lo planteado en el **RT.6**, que plantea la necesidad de tener al menos 1 TB de almacenamiento. Se puede apreciar en la tabla que según el plan solicitado, se puede tener incluso almacenamiento ilimitado.

### 5.3.1 Estimación de los costos directos por unidad de producto

Los costos directos son los que están directamente relacionados con el producto en cuestión. En este caso resulta conveniente tomar como unidad de producto al combo del módulo intercomunicador y a los módulos asociados con la recepción, almacenamiento y visualización de datos. Además, se estima que cada combo se instalará en una locación nueva, por ende se incluyen los costos asociados a la planificación. Se consideran los siguientes:

- **Hardware:** implica el hardware necesario para la instalación del intercomunicador y la alimentación a batería. Los costos se encuentran en la Tabla 5.1, tomando en cuenta el escenario más costoso posible, dando un precio de **U\$D 357**.

### 5.3.2 Estimación de los costos indirectos

Los costos indirectos son los que afectan al proceso productivo, pero no están directamente relacionados con el producto. Por su naturaleza, resulta difícil de asociar este costo a cada unidad de producto, por ende se estimará el costo indirecto mensual, y en base a este y la producción mensual se calculará el costo indirecto asociado a la unidad de producto.

- **Servicios:** es todo lo que se refiere al gasto mensual en suscripciones de software que se pueden apreciar en la tabla 5.1. Tomando en cuenta un valor promedio, el valor mensual es de **U\$D 14**. Este precio es variable según el lugar donde se instale el dispositivo.
- **Mantenimiento:** relacionado al mantenimiento del dispositivo, cambio de baterías, en caso de ser necesario, conexiones a la red, etc. **U\$D 60**.

### 5.3.3 Estimación del costo total de una unidad de producto

El costo total será el resultante de la suma de los costos directos e indirectos. En base a lo analizado en la sección 4.3.3, el costo directo resultante por unidad de producto será de U\$S 357. En cuanto a los costos indirectos, se estima una producción de 5 unidades al mes (acorde al tamaño proyectado de la empresa que diseñaría esta solución), en consecuencia, en base al costo indirecto por mes estimado en la sección 4.3.4, se estima un costo indirecto por unidad de producto de U\$S 74. Por consiguiente, el costo total de una unidad de producto será de **U\$S 431**.

## 5.4 Estimación del precio de venta

Se quiere estimar un posible precio de venta de producto. Para esto, se utilizó la estrategia *cost-plus*<sup>36</sup> en la cual se agrego un porcentaje de *markup*,  $\alpha$ , al precio por unidad total calculado en las secciones anteriores.

$$P_{venta} = CT_{unidad} + \alpha \cdot CT_{unidad}$$

Ya que se busca generar un margen de ganancia del %10 se asignó ese valor como  $\alpha$ , dando como resultado un precio de venta de U\$S 474,1.

<b>Precio de lista sin IVA</b>	USD 474,1
Descuentos	USD 0,00
<b>Precio Neto</b>	USD 474,1
IVA por unidad	USD 99,57
<b>Precio de venta con IVA</b>	USD 573,67
<b>Ingresos por Ventas</b>	USD 573,67
Impuesto a los Ingresos Brutos (3% en CABA)	USD 14,22
<b>Ingresos Netos</b>	USD 559,45

Tabla 5.2: Precio de venta

## 5.5 Estimación de volúmenes de venta

Para poder estimar correctamente el volumen de venta se tendría que hacer previamente un análisis cuantitativo de la demanda. En la sección 5.1.1 se hace mención de los posibles clientes que se pueden encontrar en el país.

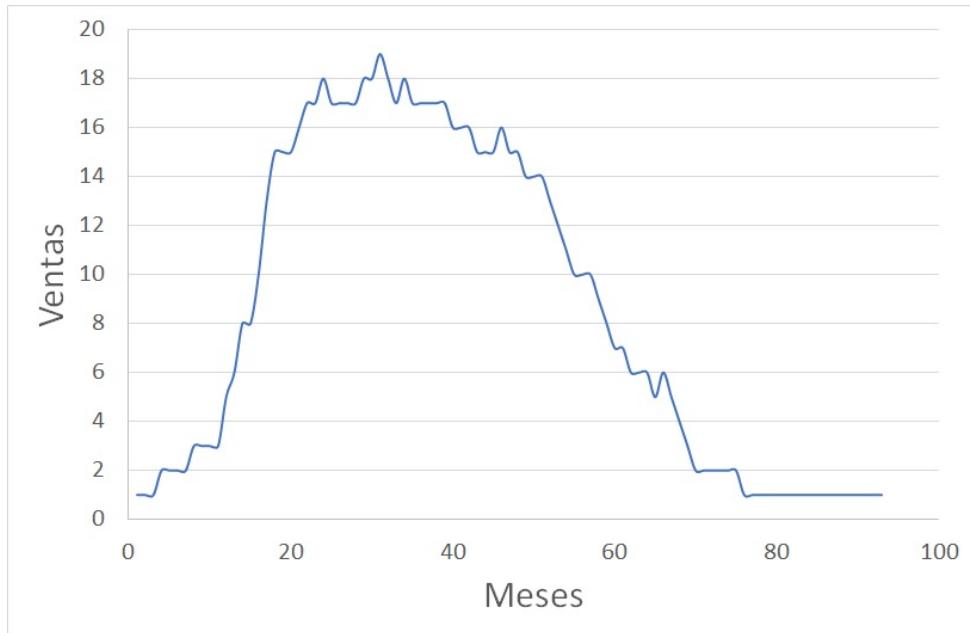


Figura 5.2: Volumen de ventas

En una etapa inicial, dada la cercanía de los autores con la Facultad de Ingeniería, es prudente apuntar el target de ventas a los laboratorios de la misma. Esto supone un estimado de una unidad de producto por venta. La FIUBA cuenta con muchos laboratorios dedicados a la investigación y que podrían ser potenciales clientes. Considerando una estrategia competitiva que permita un buen posicionamiento, se estiman un volumen de 8 a 10 ventas durante la etapa de introducción.

En la etapa de crecimiento, ya es posible salir al mercado posicionados como diseñadores de una buena solución. En esta etapa se planea comerciar con distintos institutos públicos de investigación e infraestructura y laboratorios nacionales, como así también con contactos asociados a la IEEE. Dentro de esta etapa, se espera un crecimiento elevado de las ventas.

En la etapa de maduración, es posible que las ventas lleguen a un pico. En esta etapa, se apunta a ingresar como vendedores en el sector privado.

## 5.6 Estimación de la inversión inicial

En base a lo analizado en la sección anterior, se busca estimar cuánto se debería invertir desde el punto de vista de la producción. Teniendo en cuenta la Figura 5.2, el volumen de ventas para los primeros seis meses es de aproximadamente diez unidades. En cuanto a mano de obra, se estima en promedio un ingeniero de desarrollo por unidad de

producto y 40 horas de producción por cada una. Según diversas encuestas de ingeniería, un ingeniero promedio hoy en día cobra USD 4 por hora. Teniendo en cuenta esto, y en un valor promedio para los servicios analizados en la tabla 5.1, se calcula la inversión inicial:

Item	Costo (USD)
Costos de producción	4310
Costos por mano de obra	1600
Gastos adicionales (transporte, envíos, marketing, publicidad, etc.)	1200

Tabla 5.3: Costos de inversión

Esto resulta en un total de **USD 7110**. Es fundamental la participación de distintos inversores y sponsors que empujen esta inversión para poder tener un negocio rentable a futuro.

## 5.7 Análisis del flujo de caja descontado durante el ciclo de vida

El flujo de caja de un período se calcula como la diferencia entre los egresos y los ingresos ocurridos en dicho intervalo de tiempo.<sup>37</sup> El flujo de caja total considera como período a todo el ciclo de vida. Sea  $FC_i$  el flujo de caja en un período  $i$  de extensión  $t_i$ , el balance entre ingresos y egresos en ese período es:

$$FC_i = I_i - E_i$$

Cuando se multiplica el flujo de caja por un factor de descuento se habla de un flujo de caja descontado (FCD).

$$FC_i = \frac{(I_i - E_i)}{(1 + t)^i}$$

Siendo  $t$  la tasa efectiva mensual. Según el Banco de la Nación Argentina, la Tasa Efectiva Mensual Vencida de hoy en día es 2,903%,<sup>38</sup> por lo que el valor de  $t$  será de 0,002903.

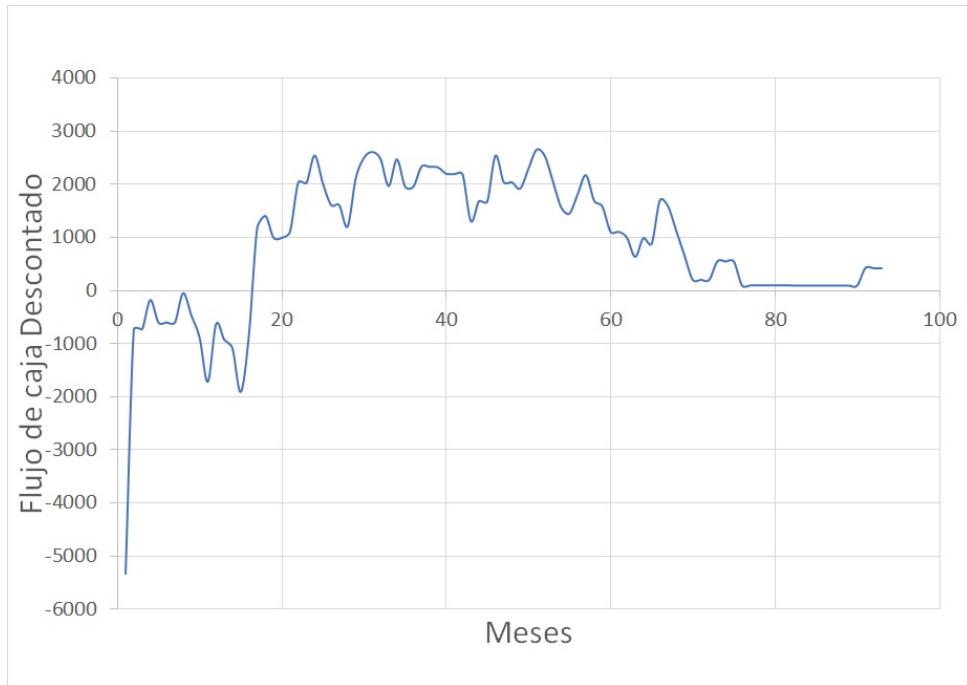


Figura 5.3: Flujo de caja Descontado

Los ingresos se relacionan únicamente a la cantidad vendida cada mes. Los egresos, igualmente, se relacionan al nivel de equipos producidos mes a mes, sumado a los costos indirectos anteriormente mencionados. Es esperable entonces que el gráfico mostrado a continuación se relacione íntimamente con el gráfico de volúmenes de ventas.

## 5.8 Estimación del VAN

El resultado de los flujos de caja descontados acumulados calculados en los  $n$  períodos del ciclo de vida determina el beneficio del proyecto, o **Valor Actual Neto (VAN)**.<sup>39</sup> El VAN ofrece un estudio en términos absolutos netos, es decir, en unidades monetarias. Indica el valor del proyecto a día de hoy. Es uno de los indicadores a utilizar para analizar la posible rentabilidad del proyecto y como consecuencia saber si es posible o no. La fórmula para el cálculo del VAN es la siguiente:

$$VAN = -E_0 + \sum_{i=1}^n FCD_i \quad (5.1)$$

La sumatoria refleja todos los flujos de caja descontados, mes a mes, a lo largo del ciclo de vida del producto. El factor negativo es la inversión inicial que se tuvo en cuenta iniciar el proyecto. Para este caso, la ecuación queda de la siguiente forma:

$$VAN = -5910 + 82420, 12 = 76510, 12 \quad (5.2)$$

Los flujos de caja descontados se analizaron para 8 años, mes a mes. La inversión inicial es la ya mencionada anteriormente.

## 5.9 Determinación de la TIR

La TIR es también un parámetro utilizado con la misma finalidad del VAN, pero en éste caso da una medida relativa, porcentual. La Tasa Interna de Retorno o TIR permite saber si es viable invertir en un determinado negocio, considerando otras opciones de inversión de menor riesgo. La TIR es un porcentaje que mide la viabilidad de un proyecto o empresa, determinando la rentabilidad de los cobros y pagos actualizados generados por una inversión.<sup>40</sup>

Transforma la rentabilidad de la empresa en un porcentaje o tasa de rentabilidad, el cual es comparable a las tasas de rentabilidad de una inversión de bajo riesgo, y de esta forma permite saber cuál de las alternativas es más rentable. Si la rentabilidad del proyecto es menor, no es conveniente invertir.

Para el cálculo del VAN se utilizó un factor de descuento en los flujos de caja. Dicho factor contiene el valor de la tasa de interés. Si encontramos la tasa que hace que el VAN sea igual a 0, encontraremos el valor de la TIR.

$$-E_0 + \sum_{i=1}^n \frac{I_i - E_i}{(1 + TIR)^i} = 0 \quad (5.3)$$

El valor de la TIR obtenido para este proyecto fue del 4,7%

## 5.10 Resumen

Del análisis realizado, se puede concluir que el producto en cuestión resulta una buena alternativa. Comparando con otros productos del mercado, éste resulta tener una versatilidad que otros no tienen, siendo que puede adaptarse a varios tipos de mediciones. Por otro lado, el hecho de que la mayor parte de los módulos constan únicamente de software, reduce mucho los costos de compra y mantenimiento, y se reduce a costos de suscripciones. Esto supone una ventaja a la hora de comparar con la competencia, pues los costos son bajos en comparación, por lo que la inversión inicial también lo es.

# Capítulo 6

## Análisis de factibilidad técnica

En este capítulo se toman en consideración los distintos enfoques del proyecto. Por un lado se analizan las ventajas y desventajas de las alternativas para el módulo intercomunicador, en términos de usabilidad, sistema operativo, programabilidad, consumo, entre otros. Por otro lado, se realiza un análisis análogo para los distintos protocolos de IoT posibles a utilizar en el proyecto. Luego, se analizan los estándares de formato de datos para el paquete que se utiliza en el trabajo. Por último, se presentan los cálculos correspondientes al consumo del módulo intercomunicador y las alternativas a utilizar según el enfoque del proyecto. Cada uno de estos análisis se presenta con la elección y justificación de cada uno de los items a utilizar, desde el hardware, protocolo y formato de paquete de datos.

### 6.1 Descripción de alto nivel

En lo que se refiere a las posibilidades tecnológicas de afrontar el proyecto, hay ciertos aspectos a tener en cuenta:

- Lugar físico donde se quiere instalar el dispositivo.
- Ingeniería de conexión a la nube
- Protocolos de comunicación
- Costos de instalación

Resumiendo estos puntos, la principal contingencia a solucionar está relacionada con la instalación de un dispositivo de IoT en un lugar remoto y con pocas posibilidades de acceso a la nube. Esto significa que para poder brindar un servicio que envíe datos en tiempo real se puede afrontar mediante diversas soluciones cuya ingeniería e instalación podría ser más o menos costosa que otras. Es notorio que para que el proyecto cumpla con los mínimos requisitos, esto es un factor vital.

De esta manera, el diagrama conceptual del sistema físico a desarrollar se presenta en la figura 1.1, y se hará especial foco en mantener una estructura de micro servicios reutilizables de forma de que solo se necesiten cambios menores a ciertos bloques funcionales para que la solución se pueda reusar en un campo totalmente nuevo.

Desde un módulo intercomunicador localizado en el mismo espacio físico que los sensores se envían los datos crudos tomados del campo directo a la nube, con la particularidad de que la red de sensores puede estar ubicada en lugares de poca cobertura. Además, estos datos serán procesados y publicados con un formato específico en un servidor también en la nube para luego ser consumidos en tiempo real por un usuario remoto a través de un servicio web.

## 6.2 Alternativas de Hardware

Para poder lograr una comunicación eficiente entre la red de sensores y el módulo intercomunicador, es necesario cubrir tres necesidades básicas:

- **Autonomía:** dado que el módulo recibe datos regularmente, en el caso en que los sensores no se encuentren en un ambiente con acceso a la red eléctrica, es importante que el dispositivo utilizado cuente con una autonomía que permita el envío de datos frecuente.
- **Comunicación:** el dispositivo debe ser capaz de comunicarse tanto con la red de sensores como con la nube a la hora de enviar los datos. Para esto es necesario que cuente con la infraestructura necesaria para lograr esto, ya sea en cantidad de pines, puertos, entre otros.
- **Fácil instalación:** desde el punto de vista del hardware, que la conexión entre los sensores y el módulo sea simple y con el menor conexionado posible, lo que trae un menor mantenimiento.

### 6.2.1 Arduino

Arduino<sup>65</sup> es una placa de desarrollo de hardware para la construcción de dispositivos digitales. Cuenta con un microcontrolador y una variedad de puertos para incorporarle diversos módulos extendiendo su funcionalidad.

Al poseer simplemente un microcontrolador, no cuenta con un sistema operativo que deba iniciar y ejecuta las tareas específicas para las que fue programada al encenderse. Esto puede ser de utilidad en el armado final del producto pero dificulta el proceso de desarrollo y análisis. Adicionalmente, si bien es un hardware reducido respecto a un ordenador completo, viendo las características de autonomía energética no cumple con la eficiencia necesaria ya que se desperdicia potencia en alimentar toda la placa, lo que incluye los recursos en desuso.

### 6.2.2 Raspberry Pi

A diferencia de la placa Arduino cuya utilidad es para pequeñas aplicaciones de usos concretos, Raspberry Pi dispone de un sistema operativo completo. Eso brinda ventajas a la hora de utilizarla durante el desarrollo ya que facilita tareas de *debugging* y permite, además, análisis de resultados usando aplicaciones que requieren mayor capacidad de cómputos.

Además, brinda un ambiente unificado donde poder realizar diversas pruebas sin la necesidad de alterar el hardware sobre el que se está trabajando y manteniendo, al igual que en la placa Arduino, movilidad y tamaño reducido.

Sin embargo, si bien presenta múltiples ventajas frente al caso anterior, también comparte una desventaja fundamental, la ineficiencia energética. Nuevamente es necesario alimentar todo un ordenador para realizar tareas que, una vez desarrolladas de manera definitiva, pueden llevarse a cabo con un sistema mucho más pequeño.

### 6.2.3 ESP-32

ESP-32 es una familia de microcontroladores con Wi-Fi y Bluetooth integrados creado y diseñado por *Espressif Systems* para dispositivos móviles, dispositivos electrónicos portátiles y aplicaciones de IoT.

Focalizando en el consumo energético sería la mejor opción para el proyecto ya que el mismo es muy reducido, pero dado que no posee un sistema operativo, la menor capacidad de cómputos comparada con las opciones anteriores y el hecho de que requiere el agregado de múltiples módulos adicionales para llevar a cabo muchas de las tareas realizadas por el equipo hace que sea una opción inviable en el comienzo del proyecto. De todas maneras una vez que el proyecto este maduro la adopción de esta placa puede ser un gran mejora ya que reduciría ampliamente el costo y consumo de recursos.

### 6.2.4 Hardware utilizado

Por lo expuesto previamente es que se llegó a la conclusión de aprovechar las facilidades brindadas por una Raspberry Pi, sobre todo para las etapas de desarrollo y primeros análisis del proyecto. Queda como línea de trabajo futuro la migración del proyecto a un hardware equivalente al sistema ESP-32, que brinda medios para la optimización de ciertos requerimientos.

## 6.3 Protocolos y enlaces de comunicación

Para la comunicación del sistema el inconveniente principal radica en la lejanía de la estación de sensado. Dado que este es un proyecto pensado para trabajar en el campo, las posibilidades de salida a la nube pueden llegar a ser nulas, por falta de cableado o redes inalámbricas. Sin mencionar además que los distintos factores físicos como el clima que pueden llegar a afectar a las comunicaciones y generar interferencias.

### 6.3.1 USSD

Se trata de un protocolo similar al SMS (servicio de mensajes cortos), aunque en lugar de enviar mensajes usuario-usuario, éste se utiliza para mensajes usuario-red y red-usuario. Los mensajes USSD pueden alcanzar los 182 caracteres alfanuméricos y una vez que se envía un mensaje se abre una conexión que permanece activa soportando un intercambio bi-direccional. Este protocolo se utiliza mucho para la localización de dispositivos remotos que no tienen buena señal de internet.

### 6.3.2 CoAP

CoAP (Constrained Application Protocol)<sup>42</sup> es un protocolo creado para la comunicación en redes con altas restricciones de recursos y con equipos con la misma particularidad. Se entiende por restricciones de red a:

- bajo *bit-rate* alcanzable
- alta pérdida de paquetes y alta variabilidad de la pérdida de paquetes
- enlace de características fuertemente asimétricas
- accesibilidad limitada a lo largo del tiempo

Mientras que restricciones en los equipos hace referencia a:

- Limitaciones de potencia
- Memoria
- Procesamiento

Utiliza una arquitectura “REST” (Representational state transfer), por lo que su funcionamiento se puede resumir en 4 principios:

1. Separación entre cliente y servidor.
2. Comunicación a través de pedidos y respuestas donde los mismos no dependen de estados previos guardados en el servidor, cada pedido es independiente.
3. Las respuestas deben ser “cacheables”, de forma que se evite dejar al cliente en un estado indeterminado y al mismo tiempo se mejore tanto el rendimiento como la escalabilidad.
4. Interfaz uniforme de forma que el recurso al que se hace un pedido quede únicamente determinado, los mensajes sean auto descriptivos y se manipulen recursos a través de representaciones de los mismos (JSON, XML, etc).

Por las características que presenta es común considerarlo como equivalente al protocolo HTTP<sup>49</sup> pero para aplicaciones de IoT. Es por esto también que interactúa fácilmente con este protocolo que también esta basado en la misma arquitectura REST.

### 6.3.3 MQTT

El protocolo MQTT (**ISO/IEC 20922:2016**<sup>50</sup>) está basado en la pila TCP/IP como base para la comunicación. En lugar de utilizar la clásica conexión cliente-servidor, MQTT utiliza un sistema de **Client-Broker** mediante las acciones de *publicación* y *suscripción*. El Broker se piensa como la entidad central del sistema, es quien filtra los mensajes que envían los clientes y los redirige a quienes se encuentre suscriptos. Los clientes pueden enviarle mensajes al Broker o suscribirse a un **Asunto** para ser receptor de todos los mensajes publicados al mismo. La principal ventaja, como hemos mencionado, es su

sencillez y ligereza. Esto lo hace adecuado para aplicaciones IoT, donde frecuentemente se emplean dispositivos de escasa potencia.

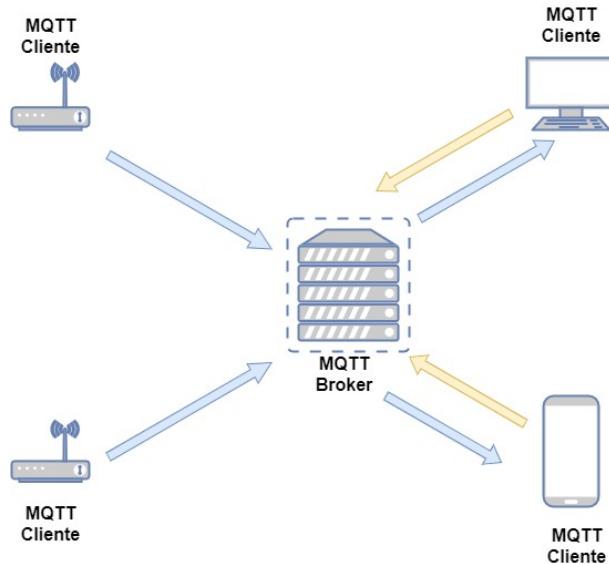


Figura 6.1: MQTT.

### 6.3.4 Medios de comunicación utilizados

El proyecto a desarrollar cuenta con 2 interfaces fundamentales a solucionar:

1. Entre los sensores y el módulo intercomunicador.
2. Entre el módulo y la nube/servidor.

En lo que respecta a la primera etapa, dado que el módulo intercomunicador estará instalado en las cercanías de los sensores es posible utilizar una conexión por cable. En particular, la transmisión se realizará por puerto serie, donde la información se envía bit a bit de manera secuencial, a diferencia de lo que podría ser una transmisión en paralelo. Esta elección además se debió a la naturaleza de los datos, los cuales, según la medición que se realiza, pueden enviarse con frecuencias distintas y en tamaños de paquetes distintos, por lo que el puerto serie brinda versatilidad para enviarlo, independientemente del tamaño o frecuencia.

La segunda interfaz a solucionar es más compleja ya que es donde entran en juego todas las restricciones a solucionar, enormes distancias por cubrir, necesidad de fiabilidad y buena cobertura, bajo consumo energético debido a la zona aislada de trabajo, entre otras. Dentro de las soluciones enumeradas, 2 candidatos parecían ser más prometedores: CoAP y MQTT.<sup>46</sup> Para tomar la decisión de cuál utilizar se realizaron pruebas de campo de ambos. Se desarrollaron scripts para ambos protocolos, en el caso de MQTT de publicación y suscripción y para CoAP un servidor y un cliente que envía el paquete al mismo. Las funciones principales de estos scripts pueden encontrarse en el anexo. De esta forma se montó un ambiente de trabajo con el mismo hardware y se envió el mismo paquete de prueba con los distintos protocolos tomando métricas temporales para su futura comparación.

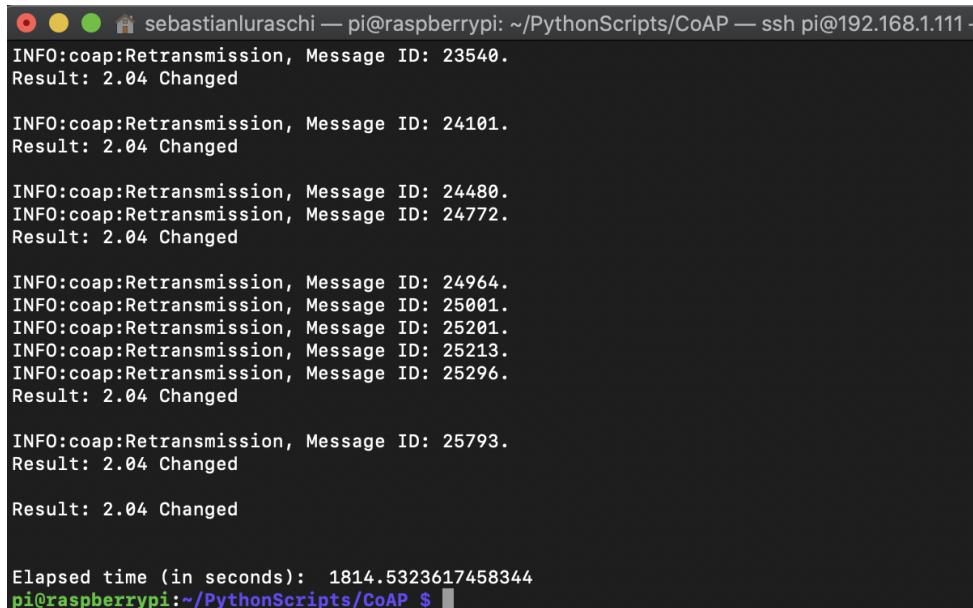
Adicionalmente, en el caso de CoAP se realizaron 2 pruebas. Dado que el paquete que el protocolo puede enviar tiene un tamaño máximo, se probó primero dividiendo el contenido del archivo por segmentos antes en el código y otra dejando el protocolo mismo se encargue de la división a la hora de transmitirlo. Los pasos para reproducir la prueba son, para MQTT:

1. Correr un broker localmente en una PC. Teniendo instalado `mosquitto` esto se puede hacer sencillamente con el comando `sudo service mosquitto start`
2. Desde otra PC conectada a la red que simula ser el dispositivo de campo editar el programa `"mqttPublisher.py"` y modificar la variable `hostname` con la dirección IP de la PC del punto 1.
3. Correr el programa `"mqttPublisher.py"` con el comando  
`python ./mqttPublisher.py`.
4. Opcional: Desde una tercera PC en la red correr el programa `"mqttSubscriber.py"` y observar la transferencia en vivo.

Por otro lado, para CoAP:

1. En una PC de la red correr el programa `"server.py"` que inicializa el servidor CoAP con el comando `python ./server.py`.
2. Desde otra PC conectada a la red que simula ser el dispositivo de campo editar el programa `"client_with_package.py"` y modificar la variable `uri` con la dirección IP de la PC del punto 1.
3. Correr el programa `"client_with_package.py"` con el comando  
`python ./client_with_package.py`.

Los resultados de tiempo de transmisión para ambos protocolos utilizando mismo hardware y enviando el mismo paquete se muestran en las figuras 6.2 y 6.3.



```

INFO:coap:Retransmission, Message ID: 23540.
Result: 2.04 Changed

INFO:coap:Retransmission, Message ID: 24101.
Result: 2.04 Changed

INFO:coap:Retransmission, Message ID: 24480.
INFO:coap:Retransmission, Message ID: 24772.
Result: 2.04 Changed

INFO:coap:Retransmission, Message ID: 24964.
INFO:coap:Retransmission, Message ID: 25001.
INFO:coap:Retransmission, Message ID: 25201.
INFO:coap:Retransmission, Message ID: 25213.
INFO:coap:Retransmission, Message ID: 25296.
Result: 2.04 Changed

INFO:coap:Retransmission, Message ID: 25793.
Result: 2.04 Changed

Result: 2.04 Changed

Elapsed time (in seconds): 1814.5323617458344
pi@raspberrypi:~/PythonScripts/CoAP $ 

```

Figura 6.2: Resultado CoAP.

```
pi@raspberrypi:~/PythonScripts/MQTT $ python ./mqttPublisher.py
('nElapsed time (in seconds): ', 85.03637385368347)
```

Figura 6.3: Resultado MQTT.

Se observa que MQTT es 20 veces más rápido, la diferencia es tal que no se siguió adelante con pruebas de consumo energético o estudio de las tramas ya que por más bien desempeño que pueda llegar a mostrar CoAP en otros aspectos, solo el tiempo que demora lo hace inutilizable en nuestra aplicación. Vemos en la figura 6.2 que parte de esa diferencia también se debe a que CoAP tenía la necesidad de reintentar múltiples veces por paquete y eso empeoraría en un caso real donde la conexión es aún peor.

Además, CoAP presentaba dificultades adicionales. Entre ellas se encuentra que las librerías encontradas presentan inconvenientes para correr en ambientes que no sean Linux y dificultades al instalar un servidor en la nube, ya que los proveedores de servicios de nube investigados por lo general brindan servicios para servidores que intercambian tramas en protocolo HTTP.

Por todo lo expuesto, se tomó la decisión de seguir adelante en el proyecto utilizando MQTT como protocolo de comunicación entre el módulo intercomunicador y la nube.

## 6.4 Formato del paquete

Con el fin de poder utilizar la plataforma de telemetría junto a múltiples sensores se define el uso de un estándar internacional como formato de entrada al sistema. De esta manera, sin importar que se esté sensando, siempre que la información se encuentre escrita en el estándar elegido la plataforma será capaz de transmitirla y procesarla adecuadamente, cumpliendo con el requerimiento de versatilidad **L0.1**.

### 6.4.1 Formato YAML

YAML<sup>54</sup> es un formato de serialización de datos propuesto por Clark Evans en 2001, quien lo diseñó junto a Ingy döt Net y Oren Ben-Kiki. Fue creado bajo la creencia de que todos los datos pueden ser representados adecuadamente como combinaciones de listas, hashes (mapeos) y datos escalares (valores simples). La sintaxis es relativamente sencilla y fue diseñada teniendo en cuenta que fuera muy legible pero que a la vez fuese fácilmente mapeable a los tipos de datos más comunes en la mayoría de los lenguajes de alto nivel. Algunas de las reglas básicas del formato son:

- Los contenidos en YAML se describen utilizando el conjunto de caracteres imprimibles de Unicode, bien en UTF-8 o UTF-16.
- La estructura del documento se denota indentando con espacios en blanco; sin embargo no se permite el uso de caracteres de tabulación para indentar.
- Los miembros de las listas se denotan encabezados por un guión ( - ) con un miembro por cada línea, o bien entre corchetes ( [ ] ) y separados por coma espacio ( , ).

- Los vectores asociativos se representan usando los dos puntos seguidos por un espacio. en la forma ”clave: valor”, bien uno por línea o entre llaves ( { } ) y separados por coma seguida de espacio ( , ).
- Los valores sencillos (o escalares) por lo general aparecen sin entrecomillar, pero pueden incluirse entre comillas dobles ( " ), o comillas simples ( ' ).
- En las comillas dobles, los caracteres especiales se pueden representar con secuencias de escape similares a las del lenguaje de programación C, que comienzan con una barra invertida (\).

A continuación se muestra un ejemplo del formato, es sencillo ver que, aún ante la falta del contexto, el ejemplo hace referencia a la compra y envío de un producto, demostrando la legibilidad y sencillez del estándar.

```

1 invoice: 34843
2 date    : 2001-01-23
3 bill-to: &id001
4     given  : Chris
5     family : Dumars
6     address:
7         lines: |
8             458 Walkman Dr.
9                 Suite #292
10            city   : Royal Oak
11            state  : MI
12 ship-to: *id001
13 product:
14     - sku      : BL394D
15     quantity   : 4
16     description: Basketball
17     price     : 450.00
18 total: 450.00
19 comments: >
20     Late afternoon is best.
21     Billsmer @ 338-4338.

```

Fragmento de código 1: Formato YML

La utilización de este paquete responde al requerimiento de la necesidad de utilizar un formato estandarizado de datos (**L0.1 - L1.1**) y en particular, en formato YAML, como así también el **RT.8**.

#### 6.4.2 Paquete estándar

Cada paquete que se transmite cuenta con un identificador definido como un contador incremental, la longitud del mismo, la fecha y hora en que se inicia la medición y una lista de segmentos.

Cada uno de estos segmentos contiene un identificador numérico independiente del identificador del paquete, los datos específicos de la medición en un campo *payload*, la hora y fecha y metadatos. Dentro de estos se encuentra la frecuencia de la medición y 4 flags que pueden ser utilizados o no según la aplicación.

En el anexo 14.3.1 se muestra un paquete transmitido aplicando la plataforma para el sensado de temperatura y humedad de un ambiente controlado.

### 6.4.3 Paquete personalizado

De ser necesario la plataforma también brinda la posibilidad de trabajar con un formato personalizado para la aplicación, un ejemplo de esto se hizo con el proyecto de estudio de precursores sísmicos para el cual se definieron 2 formatos de paquetes. Uno para datos de alta frecuencia (mayores a 1 Hz) y otro de baja frecuencia de muestreo (cada segundos, minutos u horas).

Para paquetes grandes o de muchos datos:

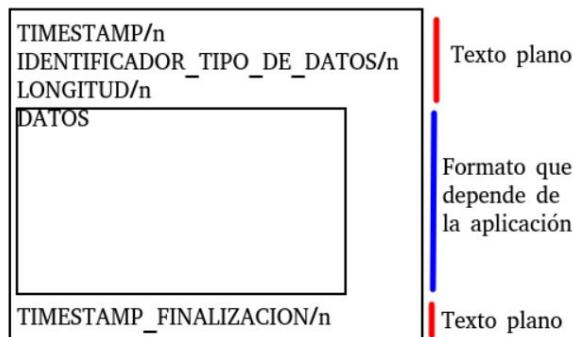


Figura 6.4: Formato de paquete de datos grande.

Para pocos datos, o transmisión de variables puntuales:

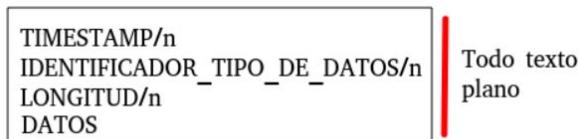


Figura 6.5: Formato de pocos datos.

El paquete de ejemplo con el que se trabajó tiene un tamaño de 17 MB, separado por 60 segmentos de datos. Cada segmento representa un minuto de datos. Los datos analizados en esta sección justifican el **RT.3**, que plantea un tamaño adecuado del paquete, en MB.

## 6.5 Almacenamiento de datos

Existen distintas alternativas para almacenar los datos transferidos desde la red de sensores. Se apunta a un modelo de base de datos, dentro de los cuales existe una amplia red de opciones según la estructura de datos con la que se trabaja. Una base de datos es un repositorio donde se almacenan datos relacionados y desde donde se pueden también recuperar. Cada una se compone de una o más tablas que guardan un conjunto de datos en columnas (atributo o característica del dato) y filas (que conforman un registro).

Una base de datos es relacional cuando se cumple con el modelo relacional que garantiza la unicidad de registros, la integridad referencial y la normalización. En ellas se usa el lenguaje SQL como estándar para la definición, manipulación y control. Este lenguaje es declarativo y es muy parecido al lenguaje natural, por estas razones es un lenguaje con el que se puede acceder a todos los sistemas relacionales comerciales.

### 6.5.1 Bases de datos no estructuradas

Se puede definir una base de datos no estructurada o NoSQL como aquella que no requiere de estructuras de datos fijas como tablas; no garantizan completamente las características ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad) y escalan muy bien horizontalmente. Se utilizan en entornos distribuidos que han de estar siempre disponibles y operativos y que gestionan un importante volumen de datos.

Se pueden esumir las principales características de una base de datos NoSQL como sigue:

- El lenguaje estándar no tiene porqué ser SQL.
- El esquema de datos es flexible o no tiene un esquema predefinido, lo que permite el tratamiento de datos heterogéneos.
- Las propiedades ACID no siempre están garantizadas.
- Mayor coherencia entre los datos de programas y bases de datos.
- Diseñadas para ser escalables generalmente de forma horizontal.
- Suelen ser distribuidas.
- Frecuentemente son de código abierto con grandes comunidades de desarrollo detrás.

Por otro lado, las características de las aplicaciones que usan bases de datos NoSQL suelen ser las siguientes:

- Trabajan con datos cuyo origen y formato es variable.
- Trabajan con datos muy relacionados.
- Necesidad de una mayor capacidad analítica.
- Mayor volumen de datos.

- Mayor disponibilidad y flexibilidad.
- Trabajan en tiempo real.

En definitiva, este tipo de base de datos es útil para aplicaciones que necesiten almacenar datos semi estructurados. Es muy versátil en entornos que necesiten escalabilidad ya que es relativamente fácil de configurar. Por otro lado, también es de utilidad en aplicaciones que almacenan grandes cantidades de datos complejos, como por ejemplo en blogs que tienen posts, comentarios, etc., o en aplicaciones de analítica.

### 6.5.2 Base de datos relacional

La base de datos relacional (BDR) es un tipo de base de datos (BD) que cumple con el modelo relacional (el modelo más utilizado actualmente para implementar las BD ya planificadas). Entre sus características más comunes se destacan:

- Una base de datos se compone de varias tablas, denominadas relaciones.
- No pueden existir dos tablas con el mismo nombre ni registro.
- Cada tabla es a su vez un conjunto de campos (columnas) y registros (filas).
- La relación entre una tabla padre y un hijo se lleva a cabo por medio de las llaves primarias y llaves foráneas (o ajenas).
- Las llaves primarias son la clave principal de un registro dentro de una tabla y estas deben cumplir con la integridad de datos.
- Las llaves ajenas se colocan en la tabla hija, contienen el mismo valor que la llave primaria del registro padre; por medio de estas se hacen las formas relacionales.



Figura 6.6: Ejemplo Base de Datos Relacional

Para manipular la información se utiliza un lenguaje relacional, actualmente se cuenta con dos lenguajes formales el álgebra relacional y el cálculo relacional. El álgebra relacional permite describir la forma de realizar una consulta, en cambio, el cálculo relacional solo indica lo que se desea devolver.

El lenguaje más común para construir las consultas a bases de datos relacionales es el SQL (Structured Query Language), un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales integradas.

Para este proyecto en particular se eligió el uso de bases de datos relacionales, debido principalmente a la estructura de los datos sensados. En si, se genera un paquete de

datos con una estructura jerárquica, la cual se compone de tres niveles, que se traducen en tres tablas distintas dentro de la base de datos. Estas representan el paquete en si, con su identificador, los segmentos que representan un conjunto de datos divididos por series temporales y el nivel más bajo son los datos crudos. Esto permite el uso de claves foráneas que relacionan los datos con las tablas de niveles superiores. Los detalles se explican en la sección 7.3.1.

Para la elección del motor de base de datos, se tuvieron en cuenta las siguientes y sus características:

TIPO	CARACTERÍSTICAS	VENTAJAS	DESVENTAJAS
ORACLE	<p>Es un SGBD relacional</p> <p>Se puede implementar en diferentes plataformas: Microsoft, UNIX, Linux</p> <p>Cuenta con un interfaz de ultima tecnología basa en java y XML</p>	<p>Permite el uso de particiones para la mejora de la eficiencia, de replicación e incluso ciertas versiones admiten la administración de bases de datos distribuidas.</p> <p>Soporta todas las funciones que se esperan de un servidor “serio”: un lenguaje de diseño de bases de datos completo (PL/SQL).</p>	<p>Precio. Incluso las licencias de Personal Oracle son excesivamente caras.</p> <p>Necesidad de ajustes.</p> <p>Un Oracle mal configurado puede ser muy lento.</p>
MySQL	<p>Escrito en C y C++</p> <p>Funciona en diferentes plataformas.</p> <p>Un sistema de reserva de memoria muy rápido basado en threads.</p> <p>Un sistema de privilegios y contraseñas que es muy flexible y seguro que permite verificación basada en el host.</p>	<p>Gran portabilidad entre sistemas, puede trabajar en distintas plataformas y sistemas operativos.</p> <p>Cada base de datos cuenta con 3 archivos: Uno de estructura, uno de datos y uno de índice y soporta hasta 32 índices por tabla.</p> <p>Aprovecha la potencia de sistemas multiproceso, gracias a su implementación multihilo.</p> <p>Flexible sistema de contraseñas y gestión de usuarios, con un muy buen nivel de seguridad en los datos.</p>	<p>Un gran porcentaje de las utilidades de MySQL no están documentadas.</p> <p>No es intuitivo, como otros programas.</p>

TIPO	CARACTERÍSTICAS	VENTAJAS	DESVENTAJAS
INFORMIX	<p>Dispone de herramientas gráficas</p> <p>Gestiona múltiples bases de datos remotas de una única consola.</p> <p>Capacidad de relación de datos de múltiples lugares físicos.</p> <p>Ocupa menos memoria y recursos que el Oracle.</p> <p>Ofrece herramientas para crear menús, formularios de entrada de datos y de generadores de listados.</p>	<p>Incluye:</p> <p>Un sistema de administración de base de datos relacionales basado en SQL</p> <p>Un lenguaje de cuarta generación</p> <p>Herramientas para la inclusión de SQL en programas de aplicación.</p>	<p>Poca información sobre Informix, debido a la poca comunidad Internet que tiene.</p> <p>Es costoso.</p> <p>No ha sabido crear soporte técnico para su producto.</p> <p>Poco terreno del marketing debido a sus pérdidas económicas</p>
SQL SERVER	<p>Gran variedad de herramientas administrativas y de desarrollo.</p> <p>Incluye herramientas para extraer y analizar datos resumidos para el proceso analítico en línea.</p> <p>Incluye herramientas para diseñar gráficamente las bases de datos y analizar los datos mediante preguntas en lenguaje normal.</p>	<p>Es un Sistema de Gestión de Bases de Datos Relacionales (SGBDR).</p> <p>Ofrece una potente forma de unir SQL e internet.</p> <p>Utiliza una extensión al SQL estándar, que se denomina Transact SQL.</p> <p>El Transact SQL, soporta la definición, modificación y eliminación de bases de datos, tablas, atributos, índices, etc.,</p> <p>Permisos a nivel de servidor, seguridad en tablas, permitir o no lectura, escritura, ejecución.</p> <p>Seguridad en los procedimientos almacenados, todo se puede configurar.</p>	<p>Enorme cantidad de memoria RAM que utiliza para la instalación y utilización del software.</p> <p>Si lo se usa para prácticas no va hacer útil porque en él se prohíben muchas cosas.</p> <p>La relación calidad-precio está muy debajo comparado con Oracle.</p>

TIPO	CARACTERÍSTICAS	VENTAJAS	DESVENTAJAS
DB2	<p>Es un SGBD relacionales multiplataforma.</p> <p>Tablas de resumen.</p> <p>Tablas replicadas.</p> <p>Agiliza el tiempo de respuesta de una consulta.</p> <p>Recuperación utilizando accesos de solo índices.</p> <p>Guarda sus datos contra la pérdida, acceso no autorizado o entradas invalidas.</p>	<p>Libre para desarrollar: Si se es un desarrollador de aplicaciones y se necesita una base de datos para su aplicación, se puede usar DB2.</p> <p>Libre para implementar: Si se está trabajando en un ambiente de producción y se necesita una base de datos para almacenar sus registros, se puede usar DB2.</p> <p>Libre para distribuir: Si se está desarrollando una aplicación o herramienta que requiera un servidor de datos empotrado, se puede incluir DB2 Express-C.</p> <p>Sin límites: Mientras que otros competidores de base de datos establecen límites en el tamaño de la base de datos, con DB2 Express-C NO hay límite de datos.</p> <p>La base de datos puede crecer y crecer sin violar el acuerdo de licencia.</p> <p>No hay límites en términos del número de conexiones de usuarios por servidor.</p>	<p>La decisión para optar por un software de estas características en general es corporativa.</p>
PostgreSQL	<p>Soporta diferentes tipos de datos.</p> <p>Permite la creación de tipos propios.</p> <p>Incorpora una estructura de datos array.</p> <p>Permite la declaración de funciones propias, así como la definición de disparadores.</p> <p>Soporta el uso de índices, reglas y vistas.</p> <p>Incluye herencia entre tablas.</p> <p>Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.</p>	<p>Fácil de Administrar.</p> <p>Software Gratuito.</p> <p>Su sintaxis SQL es estándar y fácil de aprender.</p> <p>Footprint bajo de memoria, bastante poderoso con una configuración adecuada.</p> <p>Multiplataforma.</p> <p>Capacidades de replicación de datos.</p> <p>Soporte empresarial disponible.</p>	<p>Es fácil de vulnerar sin protección adecuada.</p> <p>El motor MyISAM es instalado por defecto y carece de capacidades de integridad relacional.</p> <p>El toolset empresarial tiene un costo adicional por suscripción anual.</p>

Tabla 6.1: Motores de base de datos - Comparativa

Los autores optaron por la elección de **PostgreSQL** por las siguientes razones:

- Implementable desde Heroku Cloud (ver sección 7.3.1).
- Software gratuito
- Precios de suscripciones económicas
- Fácil conexión con portal Web (Ver sección 7.4) y con la API.

## 6.6 Visualización de datos

Existen distintas herramientas de visualización que pueden ser utilizadas dentro del proyecto, muchas de ellas gratuitas y algunas con opción de suscripción según los servicios que se contraten. Existen dos alternativas principales para el desarrollo de un portal de visualización:

- **Desarrollo In House:** implica el uso de tecnologías disponibles en el mercado de software relacionadas al desarrollo Web. En este caso, el diseño de la interfaz de usuario del portal estaría a cargo de los autores, utilizando el lenguaje de programación del framework elegido para poder diseñar la página e interactuar con la base de datos para obtener la información necesaria relacionada a los datos del proyecto, con el fin de visualizarla. También correría por cuenta de los autores el seteo de las alertas, configurando las funciones correspondientes en el lenguaje elegido de forma que interactúen con la interfaz que envía las mismas. La principal ventaja de esta opción es la posibilidad de optar por un diseño propio sin la necesidad de tener que contratar ningún servicio, por lo que no tendría costo alguno. Algunos ejemplos de Frameworks a utilizar son: *Flask*, *Angular*, *React*.
- **Aplicación de terceros:** dado que la visualización y procesamiento de datos es un proceso común en la industria, existen distintas alternativas desarrolladas por terceros que se adecuan al alcance de este proyecto. En este caso, el portal está desarrollado por la misma empresa que provee el servicio, por lo que se usaría esa interfaz de usuario. El proceso de desarrollo de la visualización resulta mucho más simple pues consta únicamente en elegir entre una serie de opciones provista por la interfaz y ubicarla en el portal. La mayoría de las opciones de mercado, en términos de interfaz de usuario funciona con el formato *Drag & Drop*, es decir que se diseña un portal con visualizaciones, filtros, botones, tan solo eligiendo las distintas opciones para los mismos y ubicándolos en la interfaz. Para la comunicación con la fuente de datos, existen diversas opciones entre las cuales se puede hacer una conexión con la base de datos mediante credenciales y la misma aplicación indexa los datos para que no haya que hacer configuraciones adicionales. En cuanto a las alertas, existen también alternativas vía email, SMS, Telegram, entre otros. La principal ventaja de esta opción es que implica muy poco desarrollo, por lo que en unas pocas horas luego de obtenido el acceso, se puede tener un portal web funcionando con visualización de los datos y alertas establecidas. Es en estos casos donde, según lo que se desee, se pueden contratar distintos planes. La mayoría de las empresas que proveen este tipo de visualizaciones cuenta con una versión gratuita y otras donde se paga una suscripción mensual. Las más comunes son: *Grafana*, *Splunk*, *Tableau*.

Se optó por la segunda opción, pues como se menciona, implica un desarrollo menor. Además de esto, se cuenta con la ventaja de tener un soporte brindado por terceros. En el caso de elegir la primera, implica un módulo más del proyecto que necesita mantenimiento. En cuanto a los costos, los autores consideran que en muchos de los potenciales proyectos se podría optar por la opción gratuita, pues en la mayoría de las alternativas esta opción se adapta al alcance de este trabajo. Además, en el eventual caso de necesitar pagar una suscripción, la misma no suele tener un costo elevado.

La alternativa elegida por los autores es **Grafana**. Entre las mencionadas, es la que mejor se adapta en términos de visualizaciones, alternativas de alertas, opciones tanto locales como en la nube, y principalmente soporte especializado por parte de la empresa. Los detalles se explican en la sección 7.4

## 6.7 Análisis de consumo

El proyecto cuenta con distintos módulos los cuales son independientes uno de otro en términos de consumo, como se plantea en el **RT.1**. De hecho, la única etapa del mismo que merece un análisis es la del módulo intercomunicador, pues el resto de ellas se encuentra en la nube y sus servidores son responsabilidad de la empresa que provee los servicios.

El proyecto se realizó utilizando una Raspberry Pi 2 como módulo. Las pruebas realizadas se hicieron con la misma conectada a la red de 220 V, 50 Hz. Dada la naturaleza de esta solución, existe la posibilidad que la misma deba ser implementada en un contexto sin acceso a la red. A continuación se hace un análisis técnico de las distintas opciones para alimentar al dispositivo con un sistema de baterías.

Según la documentación oficial de Raspberry Pi<sup>53</sup> el consumo medio del modelo 2B y 3B varía entre los 350 mA para el modelo 2 y los 500 mA para el modelo 3. Siendo que la Raspberry Pi tiene una tensión de entrada de 5 V, el consumo en potencia sería de 2,5 W aproximadamente. Según unas pruebas de stress realizadas por Dramble Raspberry Pi<sup>58</sup> (un proyecto de Web hosting patrocinado por Ansible y Kubernetes entre otros), el consumo máximo de este dispositivo no supera el Ampere. A continuación se presentan los resultados de estas pruebas.

RASPBERRY PI 2B	
Estado de la PI	Consumo
Idle	220 mA (1,1 W)
400% CPU load (stress -cpu 4)	400 mA (2,1 W)

Tabla 6.2: Pruebas de stress sobre Raspberry Pi 3B +

RASPBERRY PI 3B	
Estado de la PI	Consumo
Idle	260 mA (1,4 W)
400% CPU load (stress -cpu 4)	730 mA (3,7 W)

Tabla 6.3: Pruebas de stress sobre Raspberry Pi 3B

Los resultados obtenidos se refieren a pruebas realizadas en estado *Idle*, es decir, con el dispositivo encendido pero sin realizar ninguna instrucción; y sometiéndolo a pruebas más fuertes utilizando el 100% de sus cuatro núcleos (400% en las tablas).

Con los resultados obtenidos, se puede hacer una estimación del consumo del dispositivo y con ello las posibles alternativas para alimentarlo sin necesidad de acceder a la red eléctrica. Estos análisis se hacen en base al **RT. 2** y **RT.7**.

Si bien el proyecto busca ser genérico y adaptable a cualquier tipo de medición, las pruebas se realizaron con datos de prueba de sísmica. Los mismos requieren ser sensados con una frecuencia de microsegundos, lo que implica que el paquete de datos será de gran tamaño en comparación con otro tipo de mediciones en el ámbito de IoT. El uso de este tipo de paquete en las prueba simula un escenario de alta carga de trabajo y donde se espera alto consumo, por lo que es un buen candidato para evaluar su consumo medio. En este caso, el script de publicación tarda entre 7 y 25 minutos en publicar el paquete en cuestión (ver detalles del proceso en el Capítulo 7.2). Suponiendo el peor caso (25 minutos) y sabiendo que cada paquete representa una hora de datos, en un día el script correría un total de 24 veces, es decir 600 minutos (10 horas). El consumo entonces será:

$$5V.0,5A.8h = 25Wh \quad (6.1)$$

Se debe tener en cuenta que este proyecto se realizó utilizando el modelo 2B cuyo consumo, según las pruebas analizadas en la tabla 6.2, es menor al utilizado en la ecuación de arriba. El mismo se utilizó a modo de plantear un peor escenario.

Sabiendo este dato, a continuación se analizan las distintas alternativas a utilizar para alimentar el dispositivo con batería.

### *PiJuice HAT*

**PiJuice** es un HAT (Hardware Attached on Top - Hardware acoplado arriba) que se conecta sobre la Raspberry Pi, sin necesidad de diseñar circuitos de protección. Consta de un circuito destinado a la alimentación del dispositivo RPI en cuestión. Tiene incorporado un espacio para colocar una batería de Litio de hasta 12 Ah. La placa viene también con un Real-Time Clock (RTC) que se comunica con la PI y asegura que el dispositivo no pierda el monitoreo del tiempo. Cuenta también con un microcontrolador STM32-F0, el cual puede comunicarse con el dispositivo para control inteligente del consumo, permitiendo al usuario apagar o incluso prender el dispositivo.

Este modelo resulta muy útil para trabajar con este tipo de proyecto pues su implementación es fácil y práctica. Cuenta también con la posibilidad de conectarle **paneles solares** de manera tal que se recargue la batería y aumentar la autonomía del dispositivo. El mismo fabricante de PiJuice ofrece un simulador de descarga de batería<sup>59</sup> para estimar la autonomía según el modelo de Raspberry Pi, la batería utilizada y la función a la que se le esté forzando. Como ejemplo, para el modelo utilizado en este proyecto (2B), una batería de 5000 mAh y mirando un video de 1080p, la autonomía se estima que es de 5,34 horas.

Es importante aquí destacar el uso de los paneles pues pueden existir dos escenarios, como se mencionó antes. El primero es aquel donde el dispositivo tiene acceso a la red eléctrica y ante un corte de suministro, podría funcionar con batería algunas horas. El segundo escenario implica que el dispositivo esté a campo abierto y necesite de autonomía propia. Es aquí donde se puede pensar en los paneles solares como alternativa de carga de la Raspberry.

### *Batería con protección*

Existen otras alternativas más económicas para entregar autonomía propia a la Raspberry. En estos casos, pueden no ser compatibles con el modelo utilizado y se necesite utilizar circuitos intermedios de protección, lo que en PiJuice viene incorporado. Existen diversos modelos de protección que desconectan la carga en caso de un evento de rápida descarga. No es la intención de este proyecto ahondar en detalles de diseño. Sin embargo, estas consideraciones son necesarias en caso de optar por una alternativa de este estilo.

### *Baterías AA*

Se podría también optar por una opción con baterías AA (esta alternativa no se recomienda en un proyecto que no cuente con acceso a la red). Para esto es necesario un conversor DC/DC con el fin de obtener una tensión de continua mayor a la salida que a la entrada, con una corriente menor, de forma tal de entregar la tensión de continua necesaria (5 V) para alimentar a la Raspberry Pi.

A continuación se presenta una tabla con los cálculos aproximados de autonomía de la Raspberry Pi 3B alimentada con baterías de litio (3,7 V) de distinta carga; y con paneles solares para la recarga de las mismas, de distinta potencia. Vale aclarar que para el alcance de esta solución se podría utilizar un modelo más viejo, incluso el más básico de Raspberry, el cual consume considerablemente menos, pues no es necesario el uso de grandes aplicaciones. En este caso, se hace el análisis con este modelo pues es el que se utilizó a lo largo de todo el trabajo. El cálculo de la autonomía se estima con las siguientes ecuaciones:

$$C_{batería} \frac{V_{out}}{P_{consumida}} \quad (6.2)$$

Donde  $C_{batería}$  es la capacidad (mAh) de la batería,  $V_{in}$  es la tensión de salida de la batería (V) y  $P_{consumida}$  es la potencia (W) consumida por el mismo. Como se mencionó antes, el consumo medio de una Raspberry Pi 3B es 0,5 A y tiene una tensión de entrada de 5 V, consumiendo 2,5 W. Entonces, se puede estimar la autonomía usando baterías de distintas capacidades:

Capacidad de la batería (mAh)	Autonomía media (h)
1300	1,92
1820	2,7
2300	3,4
5000	7,4
12000	17,76

Tabla 6.4: Autonomía de la Raspberry Pi 3B para distintas baterías de Litio (3,7 V)

Esta tabla no considera el uso de los paneles solares. Aplicando la misma fórmula, se puede estimar el tiempo de carga de las mismas baterías y con distintos de ellos. Siendo la potencia consumida, la misma que entrega el panel:

Capacidad de la batería (mAh)	Potencia del panel (W)	Potencia efectiva del panel (W)	Tiempo de carga (h)
1300	12	10	0,48
	22	20	0,24
1820	12	10	0,67
	22	20	0,34
2300	12	10	0,85
	22	20	0,43
5000	12	10	1,85
	22	20	0,93
12000	12	10	4,44
	22	20	2,22

Tabla 6.5: Tiempos de carga de baterías de litio con distintos paneles solares

Vale aclarar que los cálculos se hacen suponiendo que el panel tiene incidencia solar durante todo el período de carga. Además, la potencia del panel es el valor comercial. La que realmente puede entregar es la potencia efectiva.

## 6.8 Resumen

Inicialmente se detallan los puntos técnicos del proyecto que dada su complejidad y el resultado deseado pueden presentar mayores dificultades a lo largo del desarrollo, resumidamente estos son:

- El hardware a utilizar en el equipo remoto (considerando facilidad en la instalación, consumo energético y capacidad de cómputos del equipo),
- canal y protocolo de comunicación tanto hacia el sensor como hacia la nube,
- almacenamiento de datos y
- visualización de los mismos.

Luego a partir de estos se analizó el estado del arte y el contexto tecnológico actual en cada ámbito pudiendo esclarecer las soluciones posibles y, finalmente, tomar una decisión informada tanto sobre las tecnologías a utilizar como de la arquitectura del sistema.

# Capítulo 7

## Ingeniería en detalle

En este capítulo se presenta la implementación de cada una de las etapas del proyecto, brindando el detalle de cada tecnología utilizada, los protocolos, la arquitectura de los mismos, como el protocolo IoT elegido, por ejemplo. La idea básica de este capítulo es hacer el repaso completo de la implementación final del proyecto en cuestión, analizando cada paso de punta a punta y justificando la elección de cada ítem elegido para cada módulo. Se explica también la entrada y salida de cada módulo y los requisitos necesarios para que la salida de uno sea compatible con la entrada del siguiente. En la siguiente imagen se pueden encontrar los detalles de cuáles son las etapas y módulos que se explican.

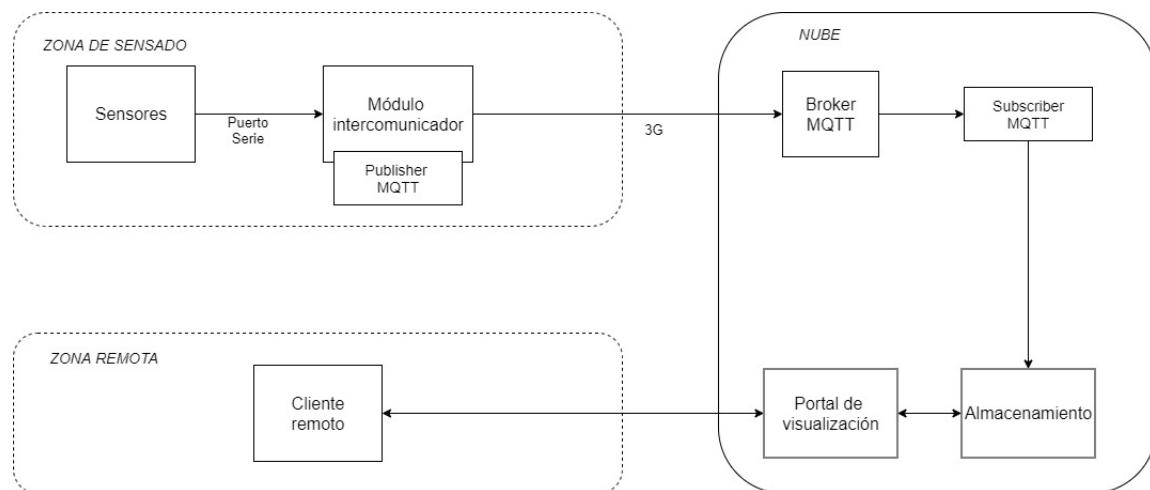


Figura 7.1: Diagrama en bloques de los módulos a analizar

## 7.1 Transmisión Sensores - Modulo

### 7.1.1 Comunicación serial

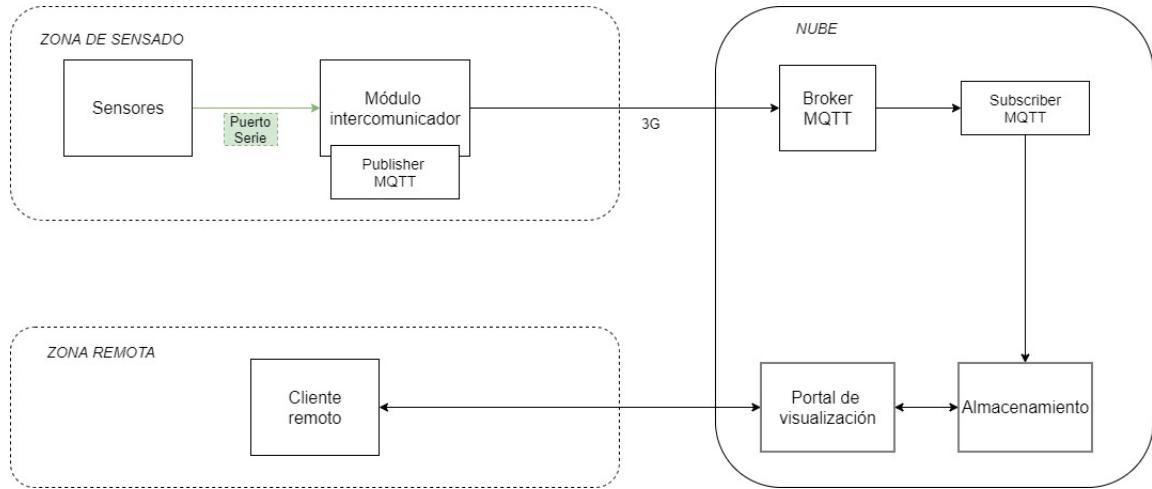


Figura 7.2: Sección de transmisión serial

#### Protocolo UART

Un puerto serie es una interfaz de comunicaciones de datos digitales, frecuentemente utilizado por computadoras y periféricos, donde la información es transmitida bit a bit, enviando secuencialmente un solo bit a la vez; este se contrasta con el puerto paralelo que envía varios bits simultáneamente sobre múltiples canales o buses.

Especificamente, la comunicación serie suele llevarse a cabo utilizando el protocolo UART ([Cap. 13.6](#)). Dado que esta comunicación es asincrónica, se establece una estructura fija para la trama:

- Bit de inicio: Indica que comienza la comunicación serie y por convención siempre se presenta con un valor lógico bajo.
- Paquete de bits: Pueden ser entre 5 y 9 bits, pero por lo general se usan 8, el tamaño definido para un byte.
- Bit de alto: Indica la finalización de la trama y por convención siempre se presenta con un valor lógico alto.



Figura 7.3: Ejemplo de trama UART.

## Configuración de la comunicación Serie

Por defecto, el puerto serie de la Raspberry Pi<sup>51</sup> se encuentra configurado para usarse con la consola de sistema. Esto es útil para arreglar problemas de arranque o poder ingresar al sistema si interfaz gráfica no se encuentra disponible pero para poder utilizar la interfaz serie en una comunicación con un dispositivo externo, esto debe deshabilitarse.

Para esto primero se valida la documentación y se ve el estado actual de la interfaz utilizando:

```
dmesg | grep tty
```

Donde *dmesg* lista los mensajes de diagnóstico e inicio del sistema y *grep tty* muestra solo los resultados que tengan *tty* en el nombre ya que según la documentación del fabricante, ambos puertos serie se representan en el sistema operativo como */dev/ttys0* y */dev/ttyAMA0*. Adicionalmente, de ahí también se obtiene que el puerto serie principal es */dev/ttyAMA0*, por lo que este es el puerto a utilizar.

```
[pi@raspberrypi:~ $ dmesg | grep tty
[    0.000000] Kernel command line: coherent_pool=1M bcm2708_fb.fbwidth=656 bcm2708_fb.fbheight=416 bcm2708_fb.fbswap=1 vc_mem.mem_b
ase=0x3ec00000 vc_mem.mem_size=0x4000000 dwc_otg.lpm_enable=0 console=tty1 root=PARTUUID=88a27704-02 rootfstype=ext4 elevator=dead
line fsck.repair=yes rootwait quiet splash plymouth.ignore-serial-consoles
[    0.000382] console [tty1] enabled
[    0.793770] 3f201000.serial: ttyAMA0 at MMIO 0x3f201000 (irq = 81, base_baud = 0) is a PL011 rev2
```

Figura 7.4: *dmesg | grep tty*

En la imagen superior se ve el resultado de este comando donde vemos que la consola se encuentra habilitada en el puerto */dev/ttyAMA0*. Para cambiar esto se inicia la configuración del sistema con **sudo raspi-config** y desde las opciones de interfaces se puede deshabilitar la terminal y habilitar para el uso con hardware externo. De esta manera si se corre nuevamente el comando *dmesg | grep tty* vemos que ya no se indica que la consola este funcionando en ese puerto.

En el modelo utilizado, los pines de transmisión y recepción respectivamente destinados al protocolo UART son el 14 y el 15. Estos son los que se utilizaron para las pruebas (Ver Capítulo **Construcción del prototipo**).

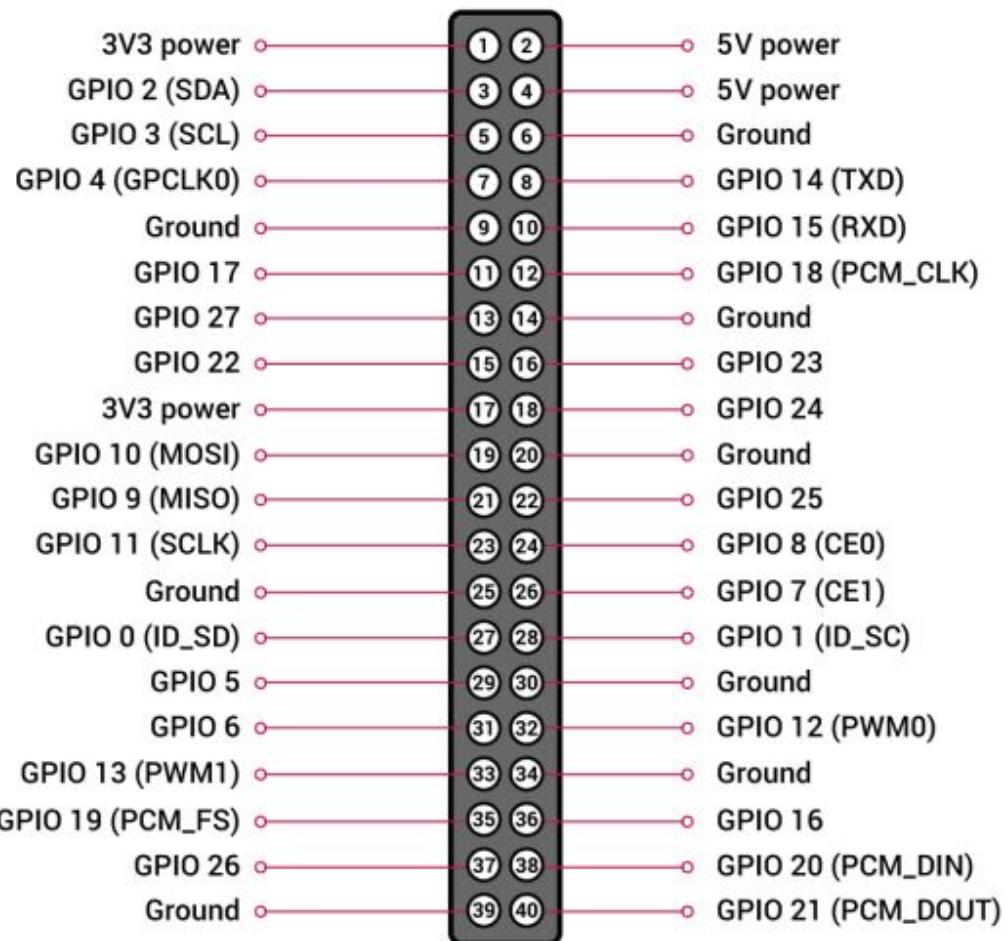


Figura 7.5: Diagrama de Pin Out - Rasapberry Pi 2B

### 7.1.2 Módulo intercomunicador

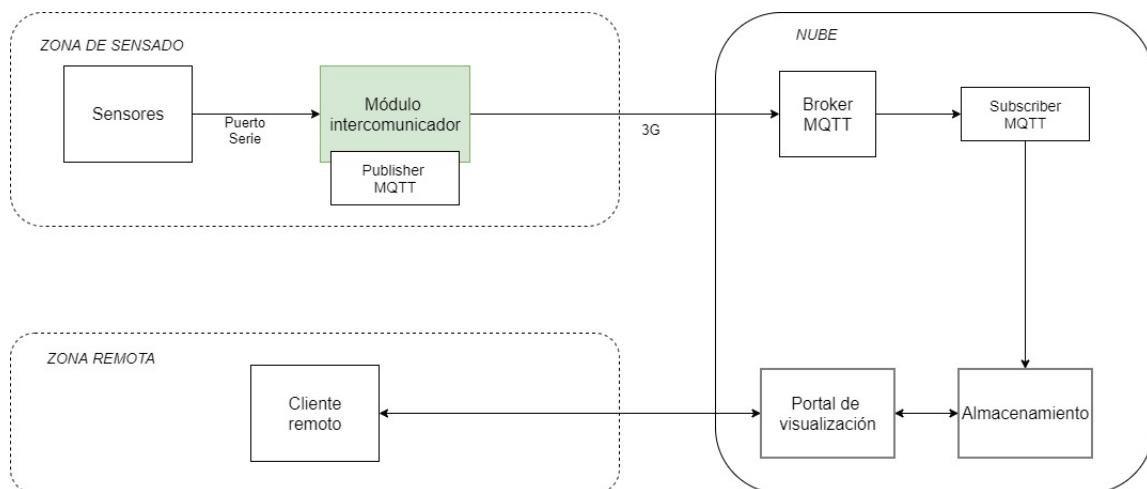


Figura 7.6: Sección módulo intercomunicador

Se trata del dispositivo fundamental para la comunicación entre los sensores externos al proyecto y la nube. Su funcionamiento para cumplir con requisitos de este trabajo se basa en:

- **Recepción** de los datos enviados por puerto serie desde el sensor genérico.
- **Procesamiento** de los datos crudos, de forma de estructurarlos en un formato adaptable para su transmisión
- **Sistema operativo** compatible con Python, de forma de poder establecer una conexión MQTT como cliente.
- Capacidad de **conexión a internet** con datos móviles de forma de poder acceder al punto de acceso donde el Broker MQTT está alojado (en la nube). Esto se relaciona con la **transmisión** de los datos recibidos.

El elegido por los autores fue la **Raspberry Pi 2 Modelo B V1.1**. La misma, como se menciona en la sección 6.2.2, cuenta con un sistema operativo completo que le da versatilidad para desarrollar distintos tipos de tareas, entre ellas incluidas las necesarias para intercomunicar la red de sensores con la nube.

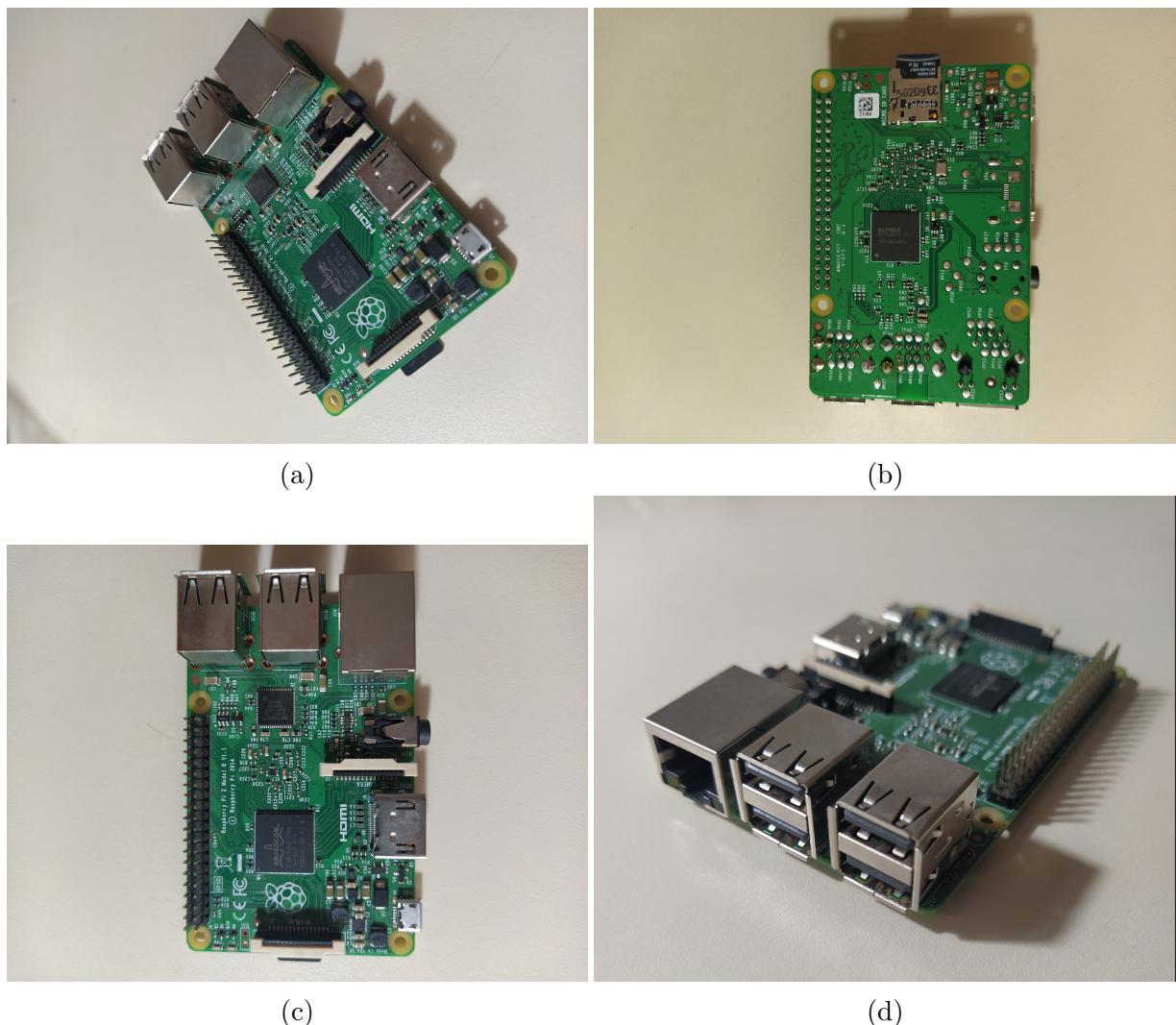


Figura 7.7: Raspberry Pi 2B V1.1

### 7.1.2.1 Sistema operativo Raspbian

Raspbian es el sistema operativo recomendado para Raspberry Pi (al estar optimizado para su hardware) y se basa en una distribución de GNU/Linux llamada Debian.

Al ser una distribución de GNU/Linux las posibilidades son vastas. Todo software de código abierto puede ser recompilado en la propia Raspberry Pi para que pueda ser utilizado en el propio dispositivo en caso de que el desarrollador no proporcione una versión ya compilada para esta arquitectura. Además esta distribución, como la mayoría, contiene repositorios donde el usuario puede descargar multitud de programas como si se tratase de una distribución de GNU/Linux para equipos de escritorio. Todo esto hace de Raspberry Pi un dispositivo que además de servir como placa con microcontrolador clásica, tenga mucha de la funcionalidad de una computadora personal.

Para la instalación del sistema operativo en el dispositivo, se necesita una computadora con Windows/Linux/MacOS, una tarjeta de memoria, un cable HDMI, un mouse (con USB), un teclado (con USB), un monitor y un instalador. Para éste último se usa el NOOBS (New Out Of Box Software).<sup>56</sup> Los pasos para la instalación son:

1. Descargar NOOBS en la PC.
2. Insertar o conectar la tarjeta SD en ese mismo PC.
3. Formatear la tarjeta SD.
4. Descomprimir NOOBS que se descargó como fichero zip en esa SD
5. Una vez termina de copiarse todo se saca la tarjeta SD y se la inserta en la Raspberry Pi
6. Conectar el cable HDMI, teclado y el mouse al monitor.
7. Conectar el cable de alimentación eléctrica y empezará a arrancar la Raspberry Pi

Una vez accedido, se empiezan a crear las particiones necesarias. A continuación aparece el menú con los sistemas operativos disponibles para instalar.

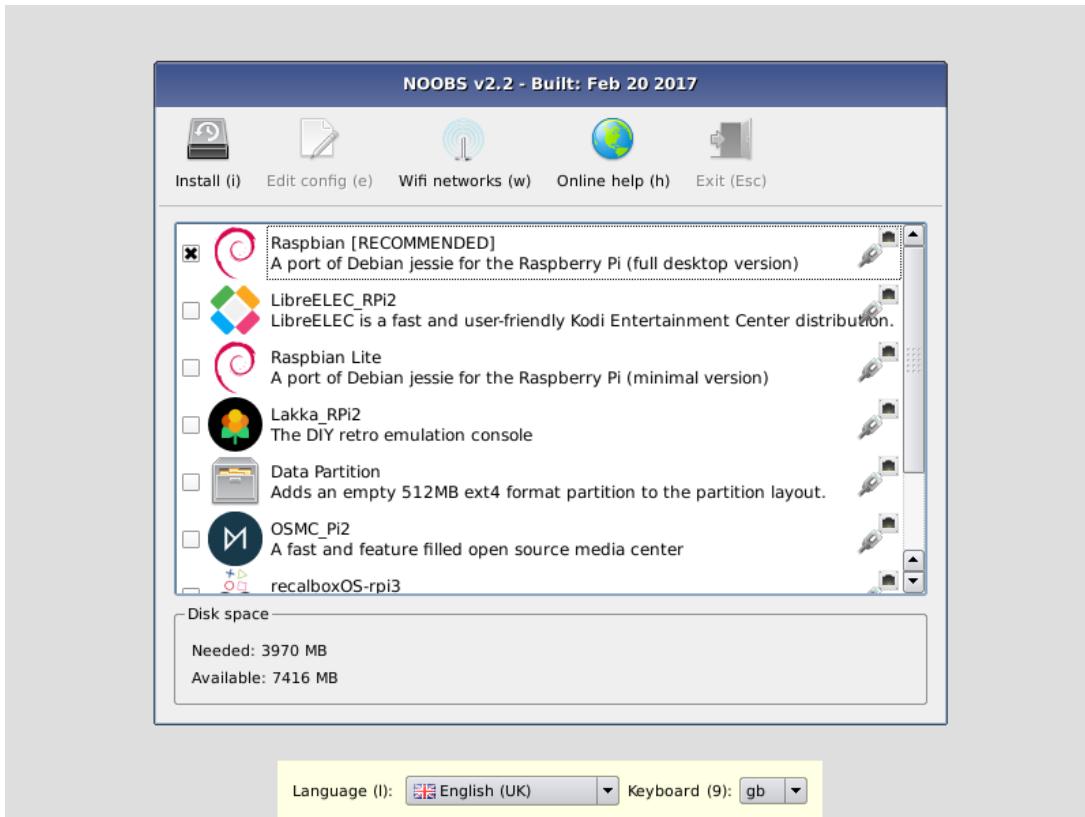


Figura 7.8: Opciones de instalación

En el mismo menú se selecciona Raspbian. Una vez seleccionada se elige el idioma y comienza la instalación. Terminada la misma, se realiza el primer boot y el dispositivo está listo para usar con Raspbian.

Para el procesamiento y transmisión de los datos es necesario contar con Python (desde la versión 2). La instalación en Raspbian es trivial. Abriendo una consola de comandos se utiliza la herramienta para gestionar la instalación y eliminación de programas y paquetes en Linux *Advanced Packaging Tool* (apt):

```
1 sudo apt update
2 sudo apt install python
```

Fragmento de código 2: Instalación de Python por línea de comando

Una vez instalado, el sistema está listo para cumplir con los requerimientos de procesamiento mínimos para el alcance de este proyecto.

## 7.2 Transmisión Módulo - Nube

### 7.2.1 Conexión 3G

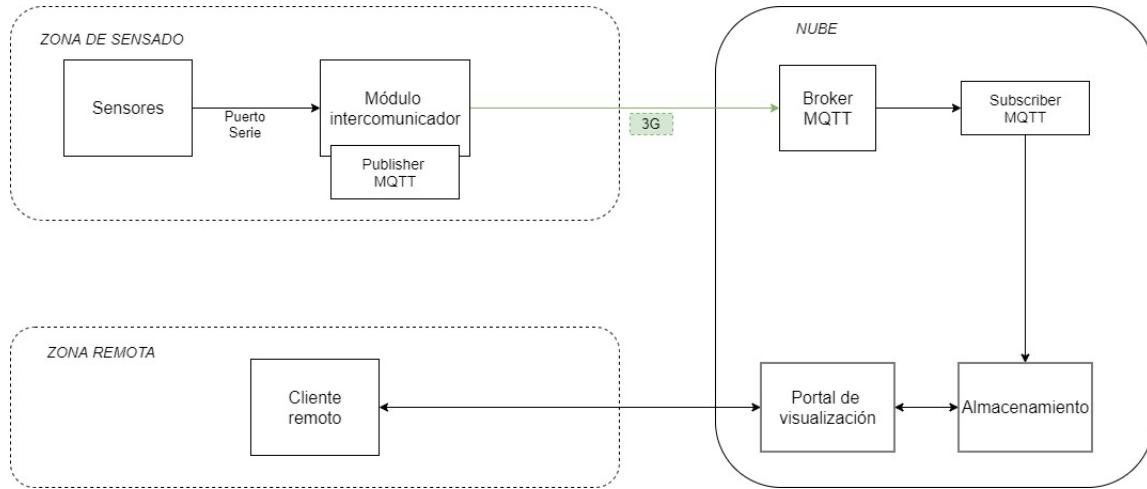


Figura 7.9: Sección conexión 3G

3G es la abreviación de tercera generación de transmisión de voz y datos a través de telefonía móvil mediante UMTS (Universal Mobile Telecommunications System o servicio universal de telecomunicaciones móviles). Las tecnologías de la tercera generación se categorizan dentro del IMT-2000 (International Mobile Telecommunications-2000)<sup>55</sup> de la ITU (Internacional Telecommunication Union), que marca el estándar para que todas las redes 3G sean compatibles unas con otras.

Los servicios asociados con la tercera generación proporcionan la posibilidad de transferir voz y datos no-voz (como la descarga de programas, intercambio de correos electrónicos, y mensajería instantánea). En todos los casos requieren una tarjeta SIM para su uso, aunque el uso del número de teléfono móvil asociado a la tarjeta para realizar o recibir llamadas pueda estar bloqueado o estar asociado a un número con contrato 3G. En este caso se utilizó un chip con una tarjeta de la compañía Tuenti, con un plan pre-pago, teniendo acceso a los datos móviles correspondientes a la cantidad de MB de descarga que ofrece por el precio pagado, por un mes.

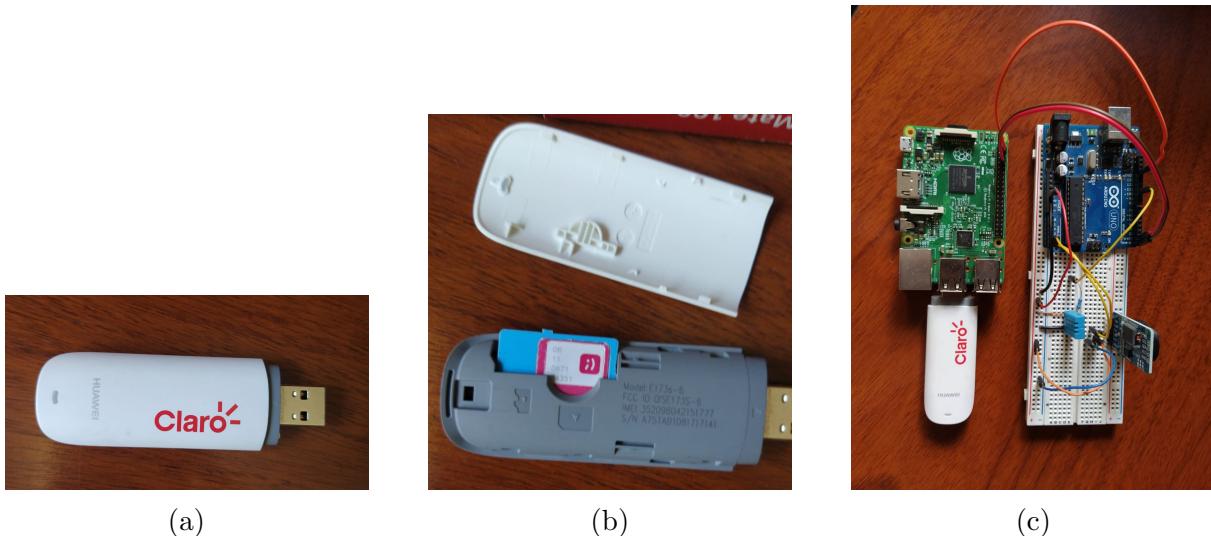


Figura 7.10: Módem 3G utilizado

En la figura 7.10 se aprecia el diseño del módem y cómo se conecta a la Raspberry Pi. Como se puede ver, el dispositivo es lo que se conoce como un *dongle* el cual posee un conector USB. Se utilizó el modelo de Huawei E173s-6 (ver hoja de datos en Anexo).

Para que el módem pueda conectarse a la red móvil del proveedor es necesario instalar el paquete `wvdial`<sup>57</sup> y un archivo de configuración para el mismo que contiene información básica sobre el puerto del módem, la velocidad, información sobre el proveedor de servicios, número de teléfono y nombre de usuario y contraseña en caso de ser necesario.

De esta manera, el archivo de conexión se presenta en el anexo 14.4.3 y en la figura 7.11 se muestra la salida cuando la conexión del módem es exitosa:

```
[pi@raspberrypi:/etc $ sudo wvdial Tuenti
--> WvDial: Internet dialer version 1.61
--> Initializing modem.
--> Sending: ATZ
ATZ
OK
--> Sending: at+cgdcont=1,"ip","internet.movil"
at+cgdcont=1,"ip","internet.movil"
OK
--> Modem initialized.
--> Sending: ATDT*99#
--> Waiting for carrier.
ATDT*99#
CONNECT 7200000
--> Carrier detected. Starting PPP immediately.
--> Starting pppd at Sun Feb  9 21:03:24 2020
--> Pid of pppd: 838
--> Using interface ppp0
--> local  IP address 10.132.216.191
--> remote IP address 10.64.64.64
--> primary   DNS address 200.81.35.1
--> secondary  DNS address 200.81.41.1
```

Figura 7.11: Conexión del módem 3G.

Esta figura demuestra la conexión exitosa a la red celular que cumple con el requerimiento **L0.2 - L1.1**.

## 7.2.2 Solución Cloud

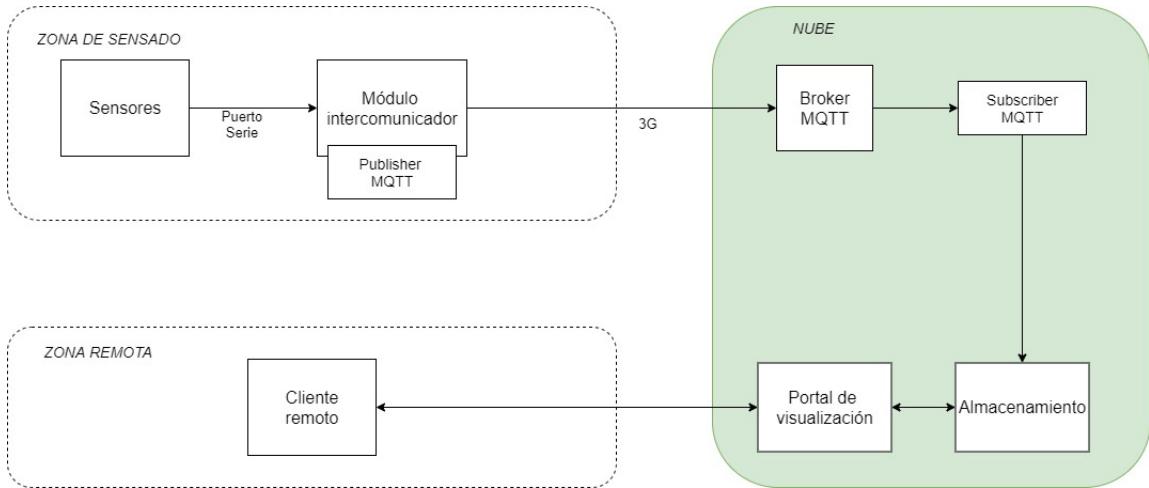


Figura 7.12: Sección Nube

Para hacer que el envío de datos sea exitoso, es necesario contar con un servicio de hosting que pueda almacenar los datos y ser accedido para el uso pertinente de los mismos. En este caso se eligió un servicio en la nube (**Heroku**<sup>61</sup>). El mismo ofrece distintos servicios gratuitos que sirven para la solución propuesta. En una versión 2.0, de ser necesario, se puede extender el servicio a una solución Cloud (**Cap. 12.6**) paga que ofrezca mayores garantías tanto de seguridad como de almacenamiento y disponibilidad.

### 7.2.2.1 Heroku CLI

Como su nombre lo indica, se trata de la línea de comandos de Heroku. El uso de la misma permite administrar los recursos de la aplicación y realizar casi cualquier tarea relacionada con Heroku directamente desde la terminal:

- Crear nuevas aplicaciones de Heroku
- Escalando los recursos de acuerdo a las necesidades de consumo
- Ejecutar scripts y aplicaciones
- Realizar copias de seguridad de la base de datos
- Configurar y administrar complementos

En lo que respecta a este proyecto, esta herramienta se usó habitualmente para la creación, despliegue y configuración del cliente de suscripción, como así también monitoreo del mismo a través de los logs como historial de recepción del mismo. Adicionalmente se usó también para monitoreo de las bases de datos ya que también fueron desplegadas y configuradas en la plataforma Heroku.

### 7.2.3 Características técnicas del protocolo MQTT

En el objetivo de enviar los datos desde el módulo intercomunicador hacia un cliente remoto se requiere la participación activa de tres agentes. Como se ve en la Fig. 6.1 estos son: un *publisher* que publique los datos, un *subscriber* que se suscriba al tópico que publica el primero y así recibir los datos, y un *broker* (**Cap. 12.6**) que actúe como entidad central.

Las características de este protocolo se corresponden con lo solicitado en el requerimiento **L0.2 - L1.2**, siendo este un protocolo ligero de IoT. A continuación se presentan sus características.

#### 7.2.3.1 Estructura de paquete

MQTT<sup>52</sup> es un protocolo binario y tiene un formato de comando y reconocimiento de comandos. De esta manera, cada vez que un cliente envía un comando al broker, este envía un acuse de recibo. Dado que a nivel de la capa de transporte se basa en el protocolo TCP, primero habrá un establecimiento de conexión TCP y luego habrá un establecimiento de conexión MQTT seguido de la transferencia de datos.

Para poder llevar a cabo las operaciones como se describió el paquete MQTT consta de un encabezado fijo de 2 bytes + un encabezado variable y una carga útil o *payload*. Este primer encabezado fijo de 2 bytes siempre estará presente en todos los paquetes mientras que el encabezado variable y la carga útil no siempre lo estarán.

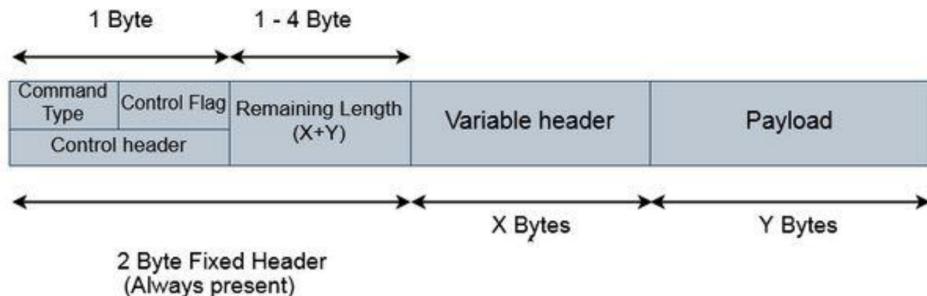


Figura 7.13: Paquete MQTT.

#### Tipo de comando

Del encabezado fijo de dos bytes, el primer byte es el campo de control. Este campo de 8 bits se divide en dos campos de 4 bits de los cuales, los primeros 4 más significativos son el campo de tipo de comando.

Los valores del tipo de comando están definidos por la tabla 7.1, donde por ejemplo el valor del comando de conexión es 1. Eso significa que para el comando de conexión el campo de tipo de conexión debe ser 0001. Para el comando de publicación, el valor es 3, lo que se traduce en 0011.

Nombre	Valor	Dirección	Descripción
Reservado	0	Prohibido	Reservado
CONNECT	1	Cliente a Broker	Pedido de conexión del cliente
CONNACK	2	Broker a Cliente	Reconocimiento de conexión
PUBLISH	3	Bidireccional	Mensaje de publicación
PUBACK	4	Bidireccional	Reconocimiento de publicación
PUBREC	5	Bidireccional	Asegura parte 1 de la entrega
PUBREL	6	Bidireccional	Asegura parte 2 de la entrega
PUBCOMP	7	Bidireccional	Asegura parte 3 de la entrega
SUBSCRIBE	8	Cliente a Broker	Pedido de suscripción del cliente
SUBACK	9	Broker a Cliente	Reconocimiento de suscripción
UNSUBSCRIBE	10	Cliente a Broker	Pedido de desuscripción
UNSUBACK	11	Broker a Cliente	Reconocimiento de desuscripción
PINGREQ	12	Cliente a Broker	solicitud ping
PINGRESP	13	Broker a Cliente	reconocimiento de ping
DISCONNECT	14	Cliente a Broker	desconexión de cliente
Reservado	15	Prohibido	Rservado

Tabla 7.1: Tipos de comandos

#### *Flag de control*

Los siguientes 4 bits son los bits de la bandera de control y son utilizados por el comando de publicación. Para el resto de los comandos el valor está reservado y siempre será el mismo: 2 para el caso de PUBREL, SUBSCRIBE y UNSUBSCRIBE y 0 para todos los demás.

Para el comando de publicación el bit 0 denota si el mensaje que se publica debe ser retenido. Si un cliente publica un mensaje con el flag de retención establecido en 1, el broker guardará ese mensaje como último mensaje retenido y este será recibido por cualquier cliente tan pronto como se suscriba a ese tema, incluso si la publicación fue previa a la suscripción.

Los bits 1 y 2 se utilizan para seleccionar la calidad del servicio si es 0, 1 o 2. Por último, el tercer bit indica si es un mensaje duplicado.

#### *Longitud restante*

El segundo byte del encabezado fijo contiene la longitud restante, que es la longitud del encabezado variable + la longitud de la carga útil. La longitud restante puede utilizar hasta 4 bytes en los que cada byte utiliza 7 bits para la longitud y el bit más significativo es una bandera de continuación.

Si el bit del indicador de continuación es 1, significa que el siguiente byte también forma parte de la longitud restante. En cambio, si el bit del indicador de continuación es 0, significa que ese byte es el último de la longitud restante.

Por ejemplo, si la longitud variable del encabezado es 10 y la longitud de la carga útil es 20, la longitud restante debe ser 30.

### *Encabezado variable*

El encabezado variable no está presente en todos los paquetes MQTT. Algunos mandatos o mensajes MQTT utilizan este campo para proporcionar información adicional o banderas y varían según el tipo de paquete. Uno de los mensajes que lo utiliza es el paquete CONNECT.

### *Carga útil*

Al final, el paquete puede contener una carga útil. Incluso la carga útil es opcional y varía según el tipo de paquete.

P.ej. Para el paquete CONNECT, la carga útil es el ID de cliente y el nombre de usuario y contraseña, si están presentes. Para el paquete PUBLISH, es el mensaje que se publicará.

### *Ejemplo de conexión*

Primeramente se presenta el formato del encabezado variable y de la carga útil del paquete CONNECT:

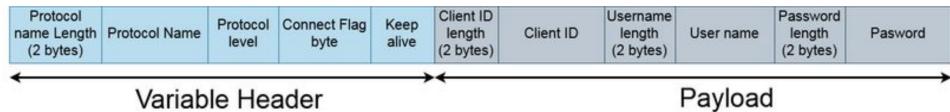


Figura 7.14: Encabezado variable y carga útil de paquete CONNECT.

En el encabezado variable, primero debe estar el nombre del protocolo. Y para esto, los primeros 2 bytes deben mencionar la longitud del nombre del protocolo seguido del nombre del protocolo. En nuestro caso, el nombre del protocolo es MQTT, que tiene una longitud de 4. Si se encuentra un protocolo no válido, el servidor puede rechazar la conexión.

Después del nombre del protocolo, está el nivel de protocolo. Esto determina qué versión de MQTT admite. Para la versión 3.1.1, el protocolo es de nivel 4. Y si el mismo protocolo no es compatible con el servidor, se desconecta enviando un acuse de recibo con el código de retorno 01.

Después de eso, está el byte de bandera de conexión donde se indica, entre otras cosas, si la conexión usará usuario, contraseña y el QoS. Para este ejemplo el valor será 02.

Los siguientes dos bytes se utilizan para mencionar la duración de mantener vivo en segundos. Durante 60 segundos, el valor será 003C en hexadecimal.

Después del encabezado variable se encuentra la carga útil y contendrá: ID de cliente, nombre de usuario y contraseña. Para el ejemplo no habrá nombre de usuario ni contraseña, por lo que solo estará presente la identificación del cliente. Al igual que se hizo para el nombre del protocolo, los primeros 2 bytes denominarán la longitud de la identificación del cliente, en este ejemplo, PQRST.

De esta forma, con un encabezado fijo de valor 10 por el tipo de comando y flags de control el paquete de conexión será:

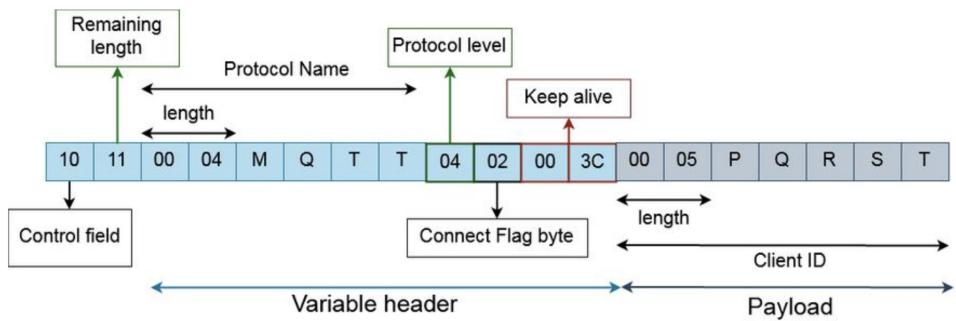


Figura 7.15: Paquete de conexión.

#### *Ejemplo de publicación*

En el siguiente ejemplo se publicará el mensaje "HOLA" al tópico OPENLABPRO. Para el paquete de publicación, el valor del comando es 3. Con QoS nivel 0 y el indicador de retención de mensajes será 0. En la sección de encabezado variable los primeros 2 bytes denotarán la longitud del tópico y luego el nombre del mismo. De manera similar, en la sección de carga útil, los primeros 2 bytes denotarán la longitud del mensaje seguido por el mensaje. De esta manera el paquete será:

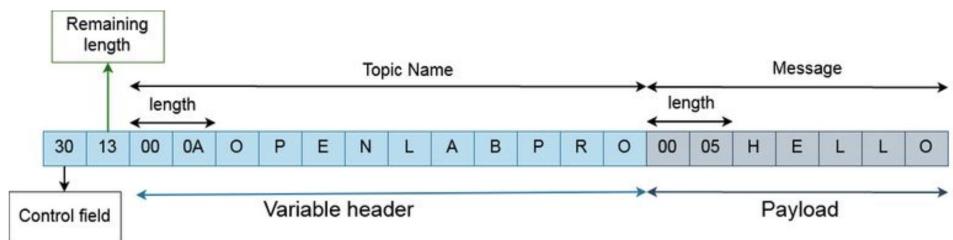


Figura 7.16: Paquete de publicación.

#### *Ejemplo de suscripción*

Una vez que el mensaje está publicado, si hay suscriptores para ese tópico, recibirán el mensaje. Para suscribirse a uno el cliente debe enviar el paquete SUBSCRIBE. El valor de comando del paquete de suscripción es 8 y el flag de control está reservado y deberá ser 2. El encabezado variable contendrá un ID de paquete de 16 bits distinto de cero y como carga útil estará el tópico para suscribirse seguido del nivel de QoS solicitado.

Para suscribirse al tema OPENLABPRO con QOS 0:

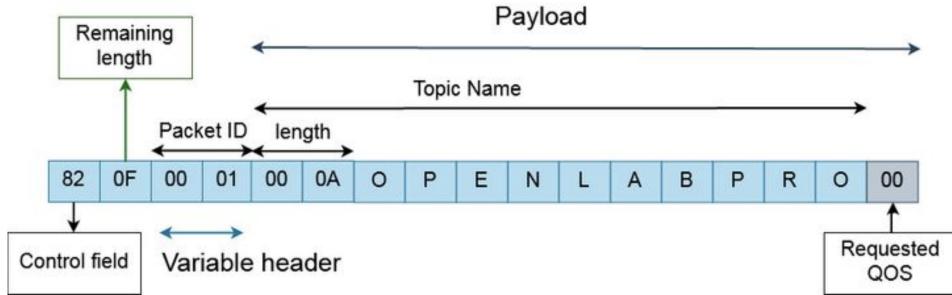


Figura 7.17: Paquete de suscripción.

### 7.2.3.2 Calidad de Servicio (QoS)

Los tres niveles diferentes de QoS determinan cómo el contenido es administrado por el protocolo MQTT. Aunque los niveles más altos de QoS son más confiables, tienen más requisitos de latencia y ancho de banda, por lo que los clientes suscritos pueden especificar el nivel deseado.

El nivel más simple de QoS es el servicio no reconocido. Este nivel de QoS utiliza una secuencia de paquetes PUBLISH; el publisher envía un mensaje al broker una vez, y este pasa el mensaje a los suscriptores una vez. No existe un mecanismo para asegurarse de que el mensaje se haya recibido correctamente, y el broker no guarda el mensaje. Este nivel de QoS también puede denominarse como máximo una vez, *QoS 0* o disparar y olvidar.



Figura 7.18: QoS nivel 0.

El segundo nivel de QoS es servicio reconocido. Este nivel de QoS utiliza una secuencia de paquetes PUBLISH/PUBACK entre el publisher y el broker, así como entre el broker y los suscriptores. Un paquete de confirmación verifica que se ha recibido contenido, y un mecanismo de reintento enviará el contenido original nuevamente si no se recibe una confirmación de manera oportuna. Esto puede provocar que el suscriptor reciba múltiples copias del mismo mensaje. Este nivel de QoS también puede denominarse al menos una vez o *QoS 1*.

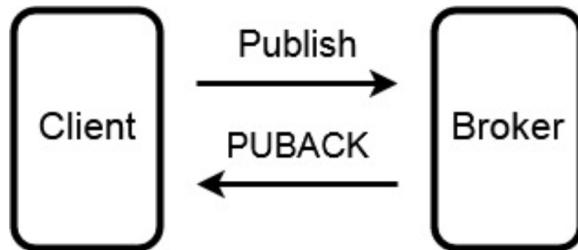


Figura 7.19: QoS nivel 1.

El tercer nivel de QoS es servicio asegurado. Este nivel de QoS entrega el mensaje con dos pares de paquetes siendo el primer par PUBLISH/PUBREC y el segundo PUBREL/PUBCOMP. Los dos pares aseguran que, independientemente del número de reintentos, el mensaje solo se entregará una vez. Este nivel de QoS también puede denominarse exactamente una vez o *QoS 2*. El gran inconveniente de utilizar este nivel de servicio es que cuadriplica la cantidad de paquetes necesarios y, con esto, eleva múltiples veces también el tiempo de una transmisión.

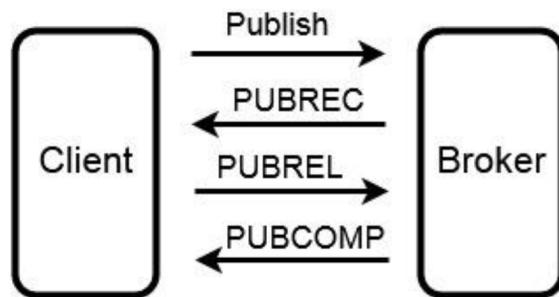


Figura 7.20: QoS nivel 2.

Los niveles de servicio, en suma con la estructura del paquete, determinan la ligereza del protocolo, que permite una transmisión remota exitosa con un protocolo de IoT que permite llevarlo a cabo. Esto se ve reflejado en el requerimiento **L0.2 - L1.2**

#### 7.2.3.3 Análisis con Wireshark

Para complementar el estudio teórico del protocolo se ejecutaron pruebas de publicación y suscripción a un broker local capturando los paquetes de la comunicación con la herramienta *Wireshark*.<sup>60</sup>

En la imagen 7.21 se observa un intercambio de paquetes MQTT entre el broker y un cliente donde lo primero que ocurre es el pedido y reconocimiento de conexión. Luego el cliente publica con QoS nivel 2 por lo que no se envía solo el paquete de publicación sino también PUBREC, PUBREL y PUBCOMP como se explicó en la sección 7.2.3.2. Por último el cliente se desconecta.

No.	Time	Source	Destination	Protocol	Length	Info
86	10.044728	192.168.1.103	192.168.1.255	DB-LSP-DISC	198	Dropbox LAN sync Discovery Protocol
50	6.037947	192.168.1.111	192.168.1.103	MQTT	118	Connect Command
52	6.038273	192.168.1.103	192.168.1.111	MQTT	70	Connect Ack
54	6.048420	192.168.1.111	192.168.1.103	MQTT	96	Publish Message (id=1) [test/topic]
56	6.048729	192.168.1.103	192.168.1.111	MQTT	70	Publish Received (id=1)
57	6.065245	192.168.1.111	192.168.1.103	MQTT	72	Publish Release (id=1)
59	6.065529	192.168.1.103	192.168.1.111	MQTT	70	Publish Complete (id=1)
60	6.086945	192.168.1.111	192.168.1.103	MQTT	68	Disconnect Req
12	3.042564	192.168.1.115	239.255.255.250	SSDP	143	M-SEARCH * HTTP/1.1

Figura 7.21: Captura de Wireshark de conexión y publicación.

Luego es posible hacer un análisis del paquete de conexión, en la figura 7.22 se presenta el detalle del mismo.

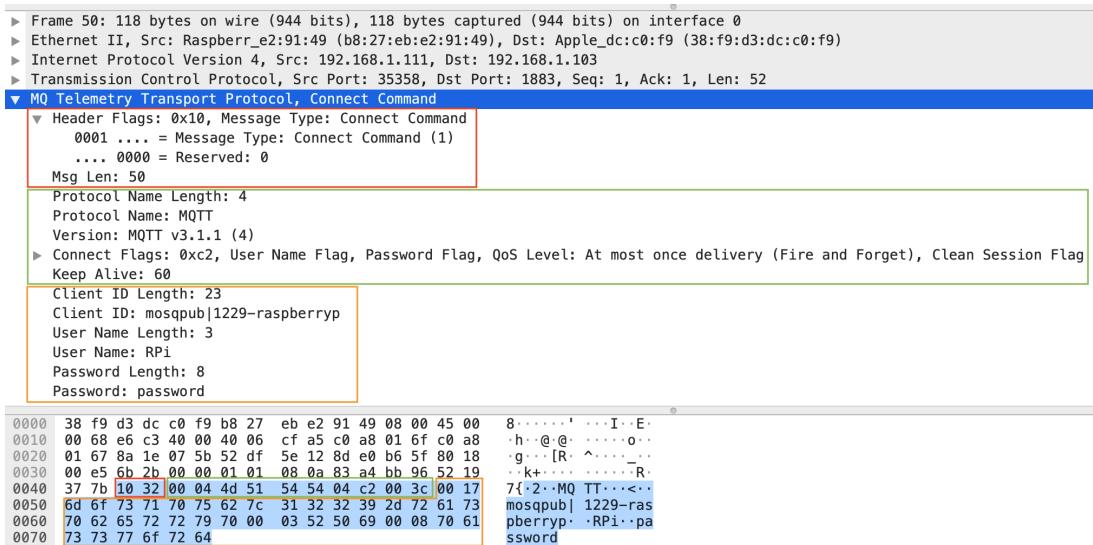


Figura 7.22: Detalle del paquete de conexión.

En rojo se muestra el encabezado fijo y el largo restante del mensaje, en verde el encabezado variable y en naranja la carga útil. Notar que en la parte inferior se encuentran marcados con los mismos colores los bytes que representan a esos colores.

Al igual que en la imagen 7.15 y en concordancia con la tabla 7.1, el encabezado fijo es 10 (0001 0000). Luego, el encabezado variable y la carga útil siguen la misma estructura que la figura 7.14, primero el largo y nombre del protocolo, los flags de conexión y luego la carga útil donde se envían el ID del cliente, usuario y contraseña enviando previo a cada uno el largo del valor.

De igual manera a partir de la misma captura es posible analizar el paquete de publicación a continuación.

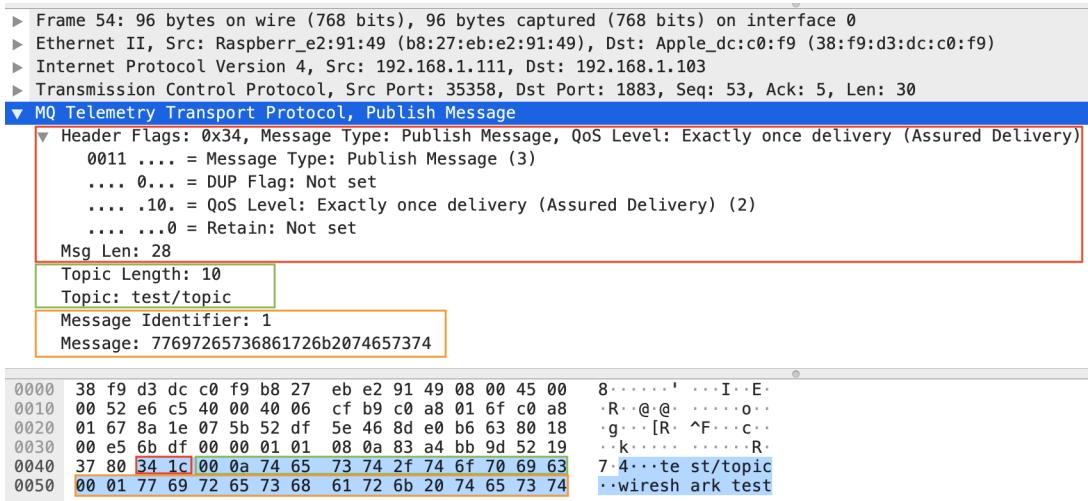


Figura 7.23: Detalle del paquete de publicación.

Comenzando con el encabezado estático, el valor es 34 difiere del 30 del ejemplo 7.16 ya que, si bien ambos son paquetes de publicación (y eso se refleja en la primera mitad del byte de valor 3), este caso es un intercambio con QoS de nivel 2, por lo que la segunda mitad del byte es 0100 (o 4 en decimal y hexadecimal) y por eso además se encuentran presentes los paquetes PUBREC, PUBREL y PUBCOMP posteriormente.

A continuación el paquete mantiene la misma estructura que el de la imagen 7.16, el encabezado variable contiene el largo y el nombre del tópico donde se publica y la carga útil un ID de mensaje y el valor del mismo, en este ejemplo "wireshark test" representado en la imagen en base hexadecimal.

Finalmente se observa la otra mitad de la comunicación, la suscripción.

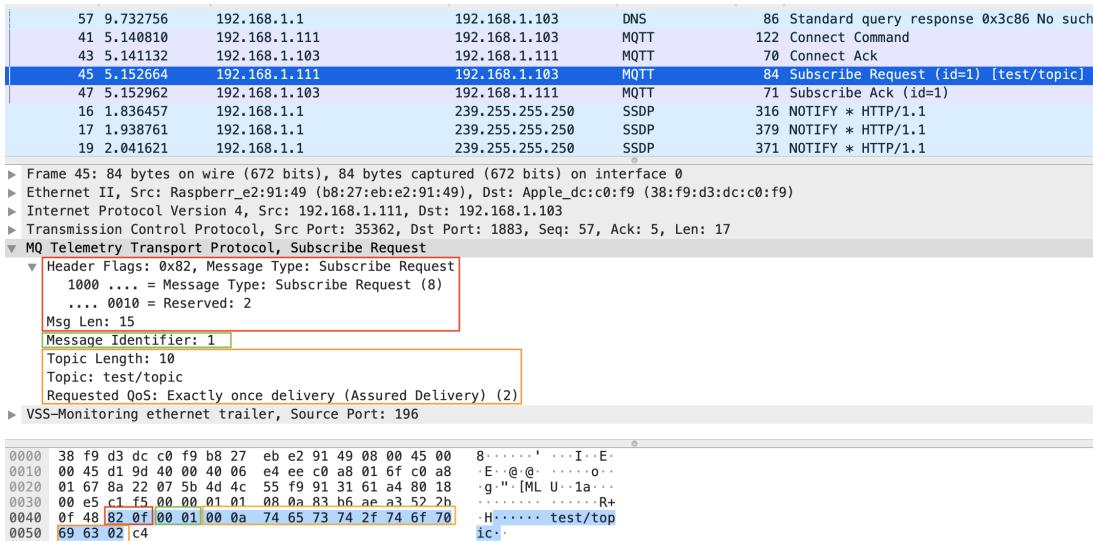


Figura 7.24: Captura de Wireshark de la suscripción.

Lo primero a observar es el intercambio de pedido de suscripción y reconocimiento del mismo. Luego se puede ver en detalle la trama de suscripción, marcado en la figura 7.24 se tiene en rojo el encabezado fijo junto con la longitud restante, en verde el encabezado

variable y en naranja la carga útil.

Haciendo referencia a la tabla 7.1 se ve que en el caso de la suscripción, el valor del tipo de comando es 8 y como se menciona también, el valor reservado del flag de control es 2, lo que coincide con lo capturado. Luego se tiene el largo restante (encabezado variable + carga útil) y en el encabezado variable se tiene un ID del paquete al igual que en la figura 7.17. Por último, en la carga útil se tiene el largo del tópico, el nombre del mismo y el nivel de QoS, todo al igual que el caso teórico de la figura 7.17.

#### 7.2.4 Broker - Heroku CloudMQTT

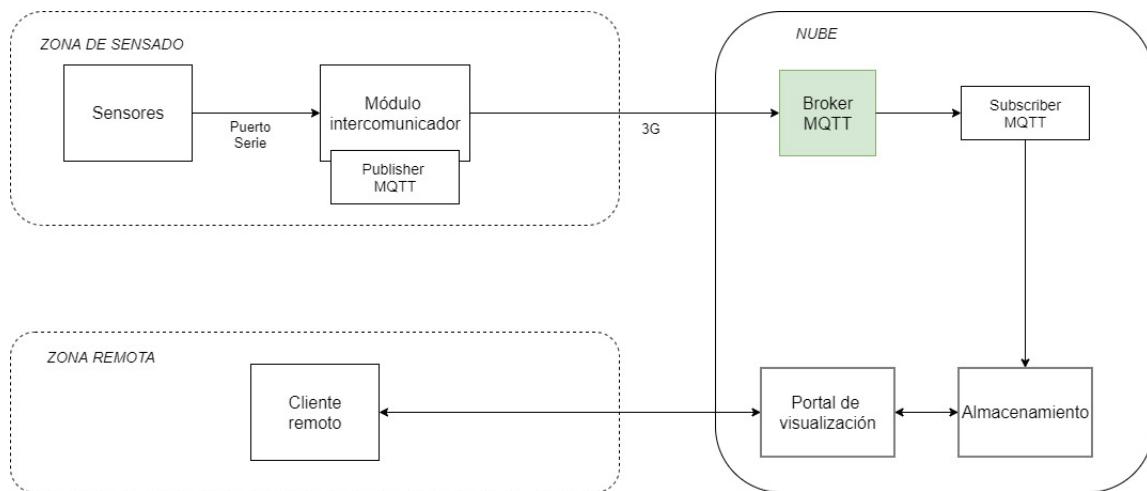


Figura 7.25: Sección Broker MQTT

El primer servicio que se explica consiste de la solución para la comunicación MQTT entre los correspondientes clientes y el broker. En particular, esta solución sirve como host para el broker.<sup>62</sup> Es decir que los clientes que se conectan a los distintos tópicos deberán utilizar el broker alojado en la nube como intermediario.

Para poder conectarse al mismo, es necesario autenticación. Los datos son brindados por la misma solución y consisten en cuatro parámetros básicos:

- Nombre del servidor
- Nombre de usuario
- Contraseña
- Puerto de conexión

Estos se pueden encontrar en la sección *Details* del servicio.

Dentro del mismo también existen distintas secciones para monitorear todo lo que envuelve al servicio. Dentro de la opción *Users & ACL* se pueden crear los tópicos para que los clientes se suscriban.

Type	Pattern	Read/Write	
topic	pi - /pi/test	true/true	<button>Delete</button>

Pattern
Topic
Pattern
 Read Access?
 Write Access?
+ Add

Figura 7.26: Ejemplo de prueba de un tópico.

Otra sección interesante es la de *Websocket UI*, dentro de la cual se puede elegir un tópico y crearlo en el momento para enviar mensajes de prueba. Luego, desde un cliente que actúe como suscriber, se recibe este mensaje. Esta opción sirve para hacer pruebas de conexión entre el suscriber y el broker.

Dentro del tab de *Connections* se pueden monitorear las conexiones existentes actuales, lo que permite ver cuántos clientes están conectados al broker y verificar tanto si hay un suscriber y/o un publisher. Luego, los detalles de estas conexiones y los datos recibidos por el broker pueden verificarse dentro de *Log*, donde se lleva un track de todas las transacciones realizadas en el puerto del broker.

### 7.2.5 Clientes

El cliente de la librería *paho-mqtt*<sup>63</sup> que fue utilizado posee múltiples opciones que el usuario puede definir para adaptar la librería a su solución. A continuación se presentarán los puntos básicos de la librería para su uso y las características puntuales de más relevancia para el funcionamiento del proyecto.

Lo primero a cubrir es la inicialización del cliente, este posee un constructor de la forma:

```
Client(client_id="", clean_session=True, userdata=None, protocol=MQTTv311,
transport="tcp")
```

Los valores que se encuentran luego del signo igual en cada parámetro son los valores que toman por defecto en caso de no indicarse.

Los primeros 2 son de especial utilidad para asegurar que no se pierdan datos en la transmisión, tener un `client_id` único asegura que los mensajes siempre se reciban por el mismo cliente. Se comprobó que si se da el caso de 2 conexiones con mismo id, el comportamiento sería errático y los mensajes llegarían aleatoriamente a alguno de los clientes compartiendo id. Cuando no se indica un id fijo el broker crea un *guid* para la sesión que asegura esa unicidad. Por otro lado, el parámetro `clean_session` es booleano e indica al broker si debe guardar información del cliente cuando este se desconecte o si debe olvidar todo, limpiar la sesión.

Una vez instanciado el cliente y previo a la conexión y suscripción se le puede definir diversas opciones para adecuar su comportamiento a través de funciones. El total de funciones a utilizar para esto son 13 pero algunas de las más relevantes son:

- `max_inflight_messages_set(self, inflight)`: Define la cantidad máxima de men-

sajes con QoS>0 que pueden fluir al mismo tiempo por la red del cliente. Por defecto es 20, incrementarlo consumiría más recursos pero puede aumentar el throughput.

- `max_queued_messages_set(self, queue_size)`: Define el valor máximo de mensajes con QoS>0 que pueden estar pendientes en la cola de salida, cuando la cola esta llena, los mensajes que no logran entrar son desechados. Por defecto es 0 e indica que no hay límite de mensajes.
- `reconnect_delay_set(min_delay=1, max_delay=120)`: El cliente reintenta conectarse automáticamente. Entre intentos de conexión esperará un número de segundos entre `min_delay` y `max_delay`.
- `will_set(topic, payload=None, qos=0, retain=False)`: Define un mensaje o *testamento*, en caso de desconexión abrupta e involuntaria (es decir, sin el llamado del método `disconnect()`), el broker publicaría este mensaje testamento en nombre del cliente desconectado.

Como última medida previo a la conexión y suscripción se definen las *funciones de callback*. Estas son disparadas por eventos, las más comunes son:

- `on_connect()`: Se dispara cuando el cliente recibe el mensaje CONNACK por parte del broker.
- `on_disconnect()`: Se dispara cuando el cliente envía un mensaje de desconexión hacia el broker.
- `on_publish()`: Se dispara cuando el menaje es enviado hacia el broker.
- `on_subscribe()`: Se dispara cuando el broker confirma la suscripción.
- `on_message()`: Se dispara cuando se recibe un mensaje en un tópico al cual el cliente esta suscripto.

#### 7.2.5.1 Publisher

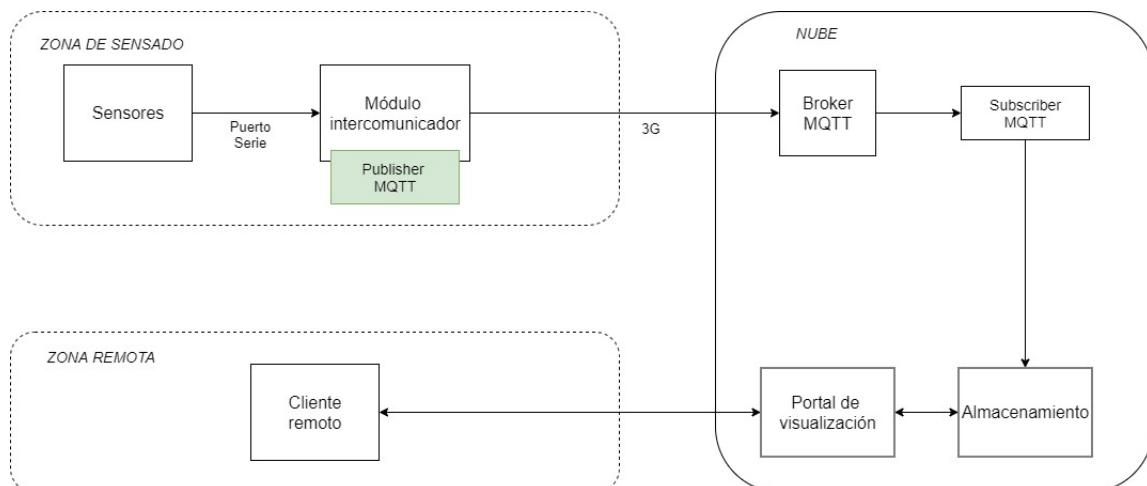


Figura 7.27: Sección Publisher MQTT

Previo a la publicación en si se utiliza lo enunciado en la sección 7.1 para la recepción del paquete por parte del equipo de sensado. Una vez recibido el paquete completo se inicia la secuencia de publicación mostrada en el diagrama 7.28. Lo primero que ocurre es la inicialización del cliente, para este se utilizan las opciones por defecto y solo se definió la función de callback `on_publish()` para poder loggear si el envío de datos fue exitoso o no.

Una vez creado el cliente, conectado al broker online y suscrito al canal correspondiente (con la necesidad de credenciales de autenticación) de la publicación, se divide el mensaje dado un tamaño predeterminado con el objetivo de evitar tanto la saturación del medio de transporte como del broker mismo a la hora de recibirlo.

Finalmente se envían los paquetes y se indica el éxito o fracaso del mismo.

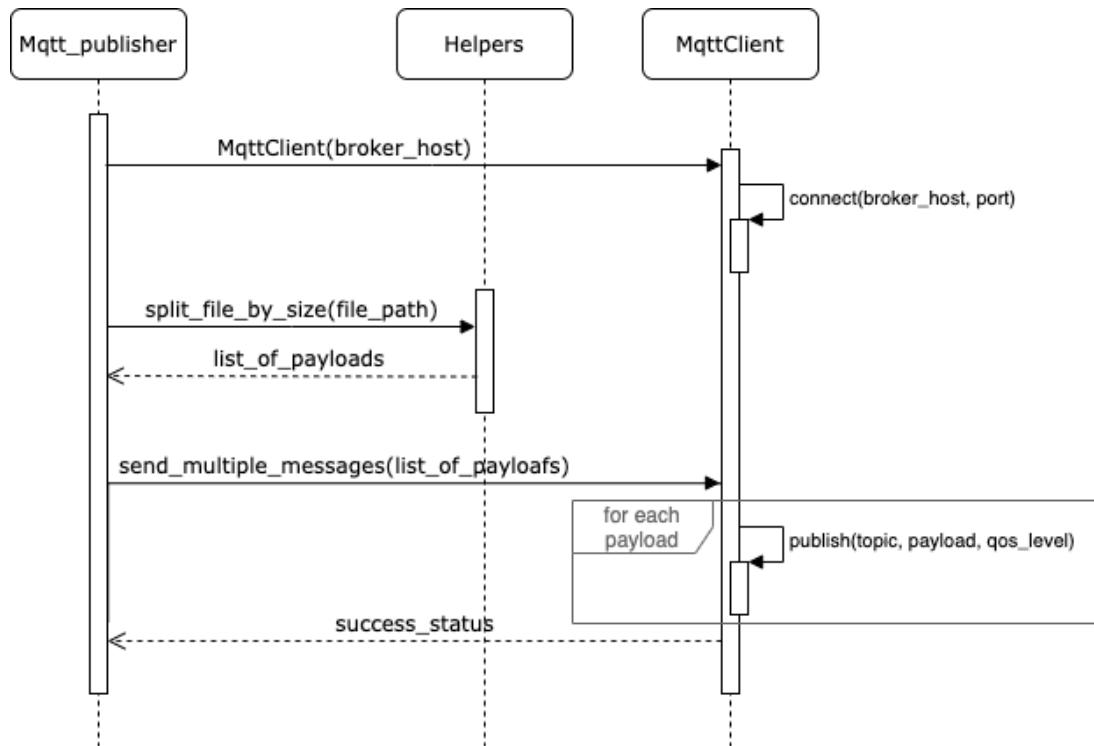


Figura 7.28: Diagrama UML del algoritmo de publicación.

### 7.2.5.2 Automatización de la publicación

Como la mayoría de los sistemas operativos, los de base Linux poseen servicios y procesos que corren por detrás para llevar acabo tareas esenciales. Cuando el sistema se inicia, estos servicios también lo hacen y se mantienen en funcionamiento hasta que el sistema es apagado.

**systemd** es un conjunto de bibliotecas y herramientas diseñados como una plataforma de administración y configuración central para interactuar con el núcleo del Sistema Operativo.

Conjuntamente, el grupo de herramientas para controlar los servicios del sistema operativo es **systemctl**. A través de estas herramientas se puede no solo administrar servicios

sino también verificar estados, cambiarlos y trabajar con los archivos de configuración.

Estos recurso que el sistema sabe cómo operar y administrar se conocen como ‘unidades’. Este es el objeto principal con el que las herramientas systemd saben cómo lidiar. Estos recursos se definen utilizando archivos de configuración llamados archivos de unidad que, por convención, se les agrega el sufijo *.service* a la hora de definirlos.

Siguiendo esta linea, se creó una unidad (o servicio) *publisher.service* con el siguiente contenido:

```
[pi@raspberrypi:~/PythonScripts/MQTT/mqtt_publisher $ cat /lib/systemd/system/publisher.service
[Unit]
Description=MQTT Publishing Service
After=multi-user.target

[Service]
User=pi
WorkingDirectory=~
Type=idle
Restart=on-failure
RestartSec=60
ExecStart=/usr/bin/python3.7 /home/pi/PythonScripts/MQTT/mqtt_publisher/main.py /pi/temp

[Install]
WantedBy=multi-user.target
```

Figura 7.29: Definición del servicio de publicación

Donde se puede destacar que:

1. Se le indica al servicio que se auto reinicie luego de un minuto al fallar en la publicación.
2. Se lo define como tipo *idle*, de forma que funcione por detrás y el dispositivo pueda realizar otras funciones en paralelo.
3. Se define además, que guarde todos logs de la tarea en el mismo dispositivo (además de los logs reflejados en el Broker de MQTT) en un archivo *publishing.log*.

De forma adicional, unos comandos típicos para el manejo de archivos de unidad son, para las tareas básicas de inicio, detención y reinicio:

```
systemctl start application.service
systemctl stop application.service
systemctl restart application.service
```

Para indicarle al sistema que inicie un servicio automáticamente en el encendido:

```
systemctl enable application.service
```

Y para deshabilitar ese inicio automático:

```
systemctl disable application.service
```

Por ultimo, para revisar el estado de un servicio:

```
systemctl status application.service
```

Donde se presenta información de todo tipo, utilizando de ejemplo el servicio creado para la publicación:

```
[pi@raspberrypi:~/PythonScripts/MQTT/mqtt_publisher $ sudo systemctl status publisher.service
● publisher.service - MQTT Publishing Service
  Loaded: loaded (/lib/systemd/system/publisher.service; enabled; vendor preset: enabled)
  Active: active (running) since Sat 2021-07-03 16:58:42 BST; 8min ago
    Main PID: 899 (python3.7)
      Tasks: 1 (limit: 2200)
     Memory: 12.6M
        CGroup: /system.slice/publisher.service
                  └─899 /usr/bin/python3.7 /home/pi/PythonScripts/MQTT/mqtt_publisher/main.py /pi/temp

Jul 03 16:58:42 raspberrypi systemd[1]: Started MQTT Publishing Service.
pi@raspberrypi:~/PythonScripts/MQTT/mqtt_publisher $ ]
```

Figura 7.30: Servicio de publicación en funcionamiento

### 7.2.5.3 Subscriber

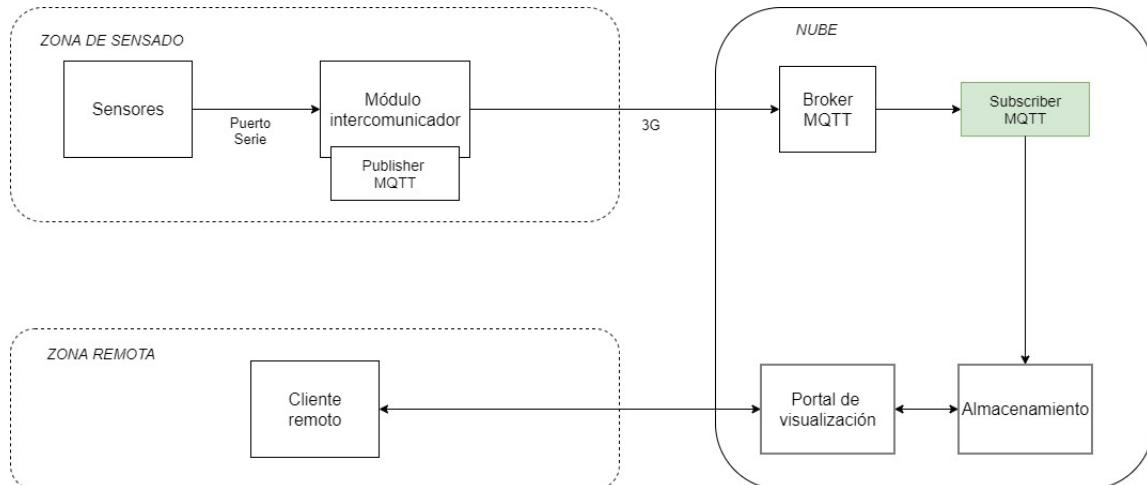


Figura 7.31: Sección Subscriber MQTT

En la inicialización del cliente Suscriber se utiliza un `client_id` preestablecido y `clean_session=False`. Con estas medidas se asegura que en caso de que el Suscriber pierda súbitamente la conexión, el broker mantenga registro de la cola de mensajes del mismo (relacionada al cliente por su id) y cuando el cliente consigue reconectarse con el mismo id fijo, continúa con la recepción de mensajes donde había dejado.

Además de las opciones mostradas en la sección anterior se definió achicar el tiempo de reintento de conexión para asegurar un funcionamiento continuo y no acumular gran cantidad de paquetes en las colas de espera, disminuyendo el riesgo de pérdidas.

Dado que este cliente debe estar continuamente disponible para la recepción de datos, se optó por tenerlo dentro de la misma solución Cloud. Para esto se configuró un *pipeline* de despliegue automático desde el repositorio en Github a una aplicación en Heroku. Esto permite que cualquier actualización que se impacte en el código fuente

sea automáticamente desplegada en el suscriptor en la nube, asegurando el correcto funcionamiento en su última versión sin necesidad de intervención manual.

Una vez recibido el paquete y previo a la carga en la base de datos, el algoritmo valida que lo recibido sea congruente con el tipo de paquete esperado. Es decir, si el paquete proviene del tópico de mediciones de temperatura y humedad, se espera que las mediciones cuenten con 2 valores numéricos en un formato de payload preestablecido como puede ser: '30T50H'. Esto lleva a que se deba predefinir (y mantener) un formato conocido para cada sensado que se desee agregar a la plataforma y consecuentemente también a la extensión (no modificación) del código fuente para incorporar este nuevo formato. Esta forma de trabajar brinda al algoritmo una estructura cerrada a las modificaciones pero extensible a nuevos comportamientos como indica el segundo principio enunciado en 1.5. Además le da el carácter de extensible al producto, de forma que más de un sensor pueda enviar paquetes al módulo intercomunicador y que este los envíe de manera exitosa, distinguiendo entre cada uno. Esto último se corresponde con el requerimiento **L0.1 - L1.1**.

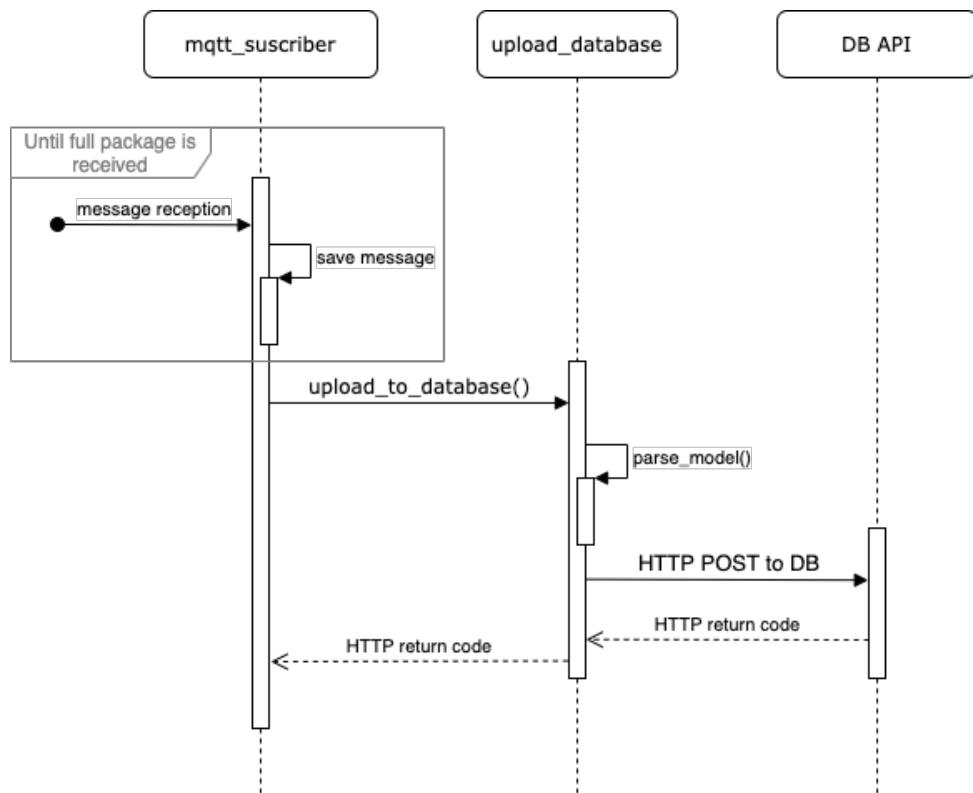


Figura 7.32: Diagrama UML del subscriber.

Luego de una validación exitosa se envía el paquete a la API de almacenamiento que se encarga de la gestión de la base de datos, asegurando una clara división de tareas y responsabilidades unitarias, nuevamente como se indica en el primer principio de 1.5.

## 7.3 Almacenamiento de datos

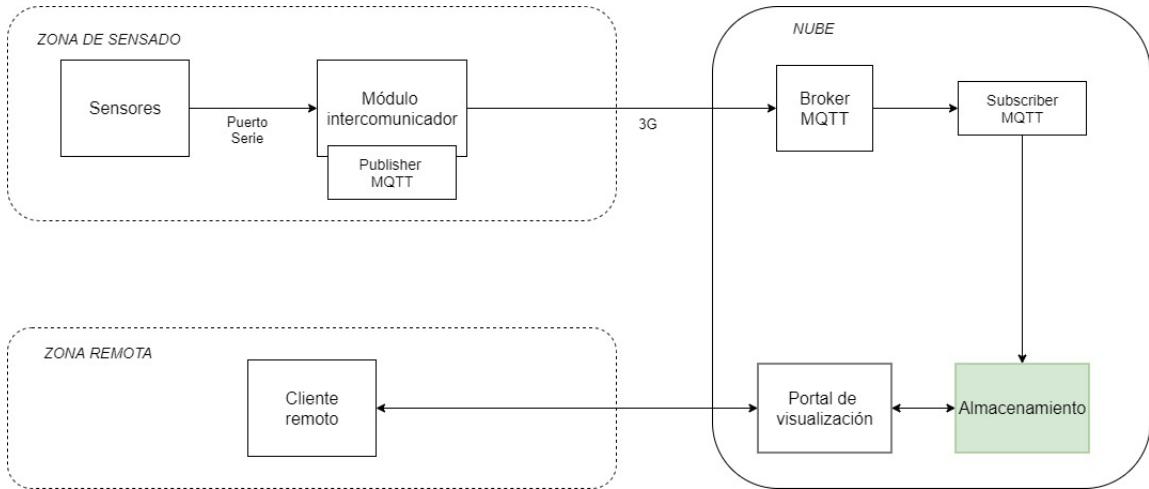


Figura 7.33: Sección Base de Datos

### 7.3.1 Heroku Postgres

Uno de los elementos principales del proyecto punta a punta es lograr tener los datos crudos almacenados (Requerimiento **L0.5**) de forma tal que puedan consumirse desde una aplicación, un portal web o simplemente poder descargarlos desde un cliente remoto. Para esto, se utiliza una base de datos PostgreSQL. La misma está hosteada dentro de la misma solución Cloud utilizada en el resto del proyecto. De esta forma, para lograr la conexión a la misma, la misma solución provee de un usuario y una contraseña para poder autenticarse a la base y realizar consultas.

En base a los datos recibidos, se cuenta con tres tablas:

- **Paquete:** representa la información del paquete de datos.
- **Segmentos:** información de los segmentos de cada paquete. Para el caso de los precursores sísmicos, por cada paquete se tiene 60 segmentos, con lo cual se tiene 60 filas de segmentos por cada fila de la tabla *paquete*.
- **Data:** los datos correspondientes a cada segmento de datos dentro del paquete. Esta es la tabla que indexa la información cruda de forma tal que pueda ser consumida.

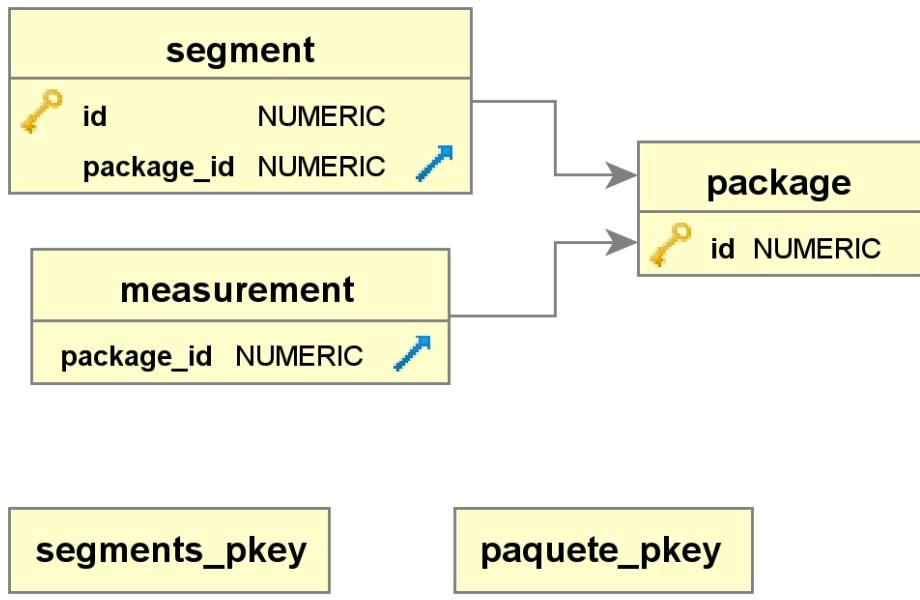


Figura 7.34: Diagrama relacional de la base de datos

Como se puede apreciar en la imagen, se trata de una base de datos relacional. Esto significa que dentro de cada tabla existen atributos que la vinculan con otra. Esto se aplica con el fin de tener datos asociados a segmentos y segmentos asociados a paquetes. Dentro de la tabla *Data* existen dos atributos que son **Foreign Key** es decir que apuntan hacia un atributo de otra tabla. Estos son el ID del paquete y el ID del segmento, de forma de poder diferenciar un dato del segmento x y del segmento y, como así también de los distintos paquetes. De la misma forma se eligió una Foreign key dentro de la tabla de segmentos que apunta hacia el ID del paquete. Si se eliminara un paquete con ese ID, se eliminarían todos los registros que apuntan al mismo.

### 7.3.2 API de comunicación

Con el fin de cumplir con el requerimiento **L0.5**, se desarrollo una API ([Cap. 13.6](#)) para la interacción con la base de datos de forma tal que los distintos bloques funcionales del proyecto que requieran tanto insertar como consumir información se comuniquen con una única estructura con mensajes HTTP sin la necesidad de hacer querys SQL donde no corresponde. Esto a su vez brinda abstracción y re-usabilidad manteniendo la esencia de micro servicios, todas características deseadas en una plataforma que busca ser extensible a múltiples aplicaciones. Es un punto centralizado para las comunicaciones que además brinda seguridad tanto al prohibir el acceso directo a la base de datos como al predefinir a través de sus métodos las operaciones realizables en la base. Esto es de suma importancia ya que usuarios con conocimiento técnico o futuros desarrolladores externos a la plataforma que la utilicen para sus proyectos podrían eliminar de forma maliciosa el contenido o acceder a información para el cual no están autorizados.

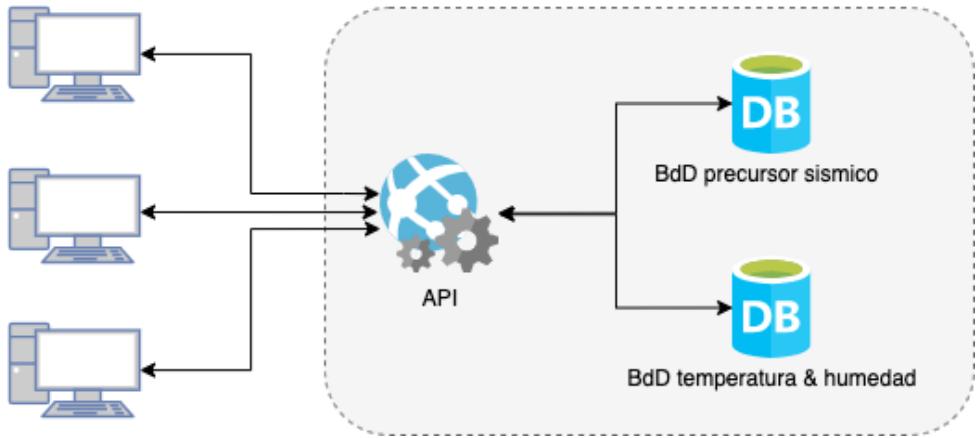


Figura 7.35: Diagrama de alto nivel del uso de la API

Generalmente, las funciones básicas hacia la base de datos que se ofrecen a través de la interfaz son las de Crear, Leer, Actualizar y Borrar (o CRUD de sus siglas en inglés). A su vez, el protocolo HTTP define una serie de métodos de petición (también referido como "verbos") que pueden utilizarse y tiene flexibilidad para ir añadiendo nuevos métodos y así añadir nuevas funcionalidades. Cada método indica la acción que desea que se efectúe sobre el recurso identificado, los más comunes y que se corresponden con las funciones CRUD expresadas previamente son: POST, GET, PUT, DELETE.

En este caso solo se ha implementado la creación de datos en la base como un método POST. Los datos a crear en la base se deben incluir en el cuerpo de la petición HTTP en formato JSON con una estructura como se muestra en el anexo 14.4.4.

En él mismo lo primero que se debe incluir son las credenciales de la base de datos, de esta manera se evita un acceso no autorizado y la posibilidad de introducir información maliciosa por un agente externo. A continuación se debe incluir el paquete en si de datos a insertar.

Para indicar el estado de la petición la API el protocolo HTTP también define un listado de códigos de retorno:

- Códigos con formato 1xx: Respuestas informativas. Indica que la petición ha sido recibida y se está procesando.
- Códigos con formato 2xx: Respuestas correctas. Indica que la petición ha sido procesada correctamente.
- Códigos con formato 3xx: Respuestas de redirección. Indica que el cliente necesita realizar más acciones para finalizar la petición.
- Códigos con formato 4xx: Errores causados por el cliente. Indica que ha habido un error en el procesado de la petición a causa de que el cliente ha hecho algo mal.
- Códigos con formato 5xx: Errores causados por el servidor. Indica que ha habido un error en el procesado de la petición a causa de un fallo en el servidor.

En particular, los códigos utilizados son *200:OK* cuando se consigue insertar en la base de forma correcta, *401:UNAUTHORIZED* cuando las credenciales provistas no tienen acceso a la base de datos y *400:BAD REQUEST* cuando se da algún error inesperado durante el proceso.

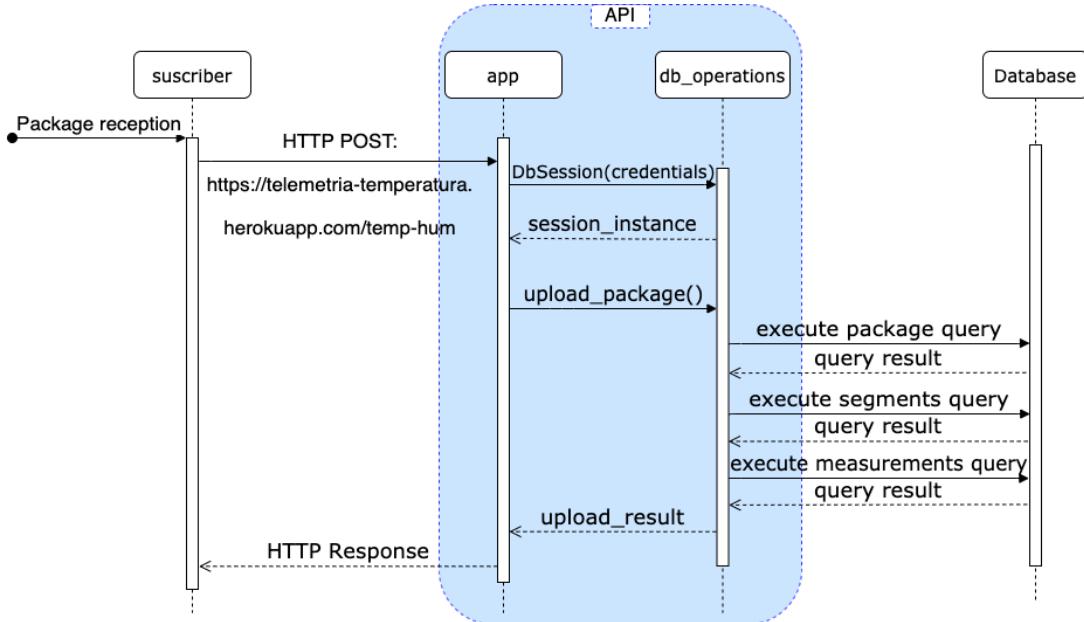


Figura 7.36: Diagrama UML del guardado en la base de datos

## 7.4 Portal web

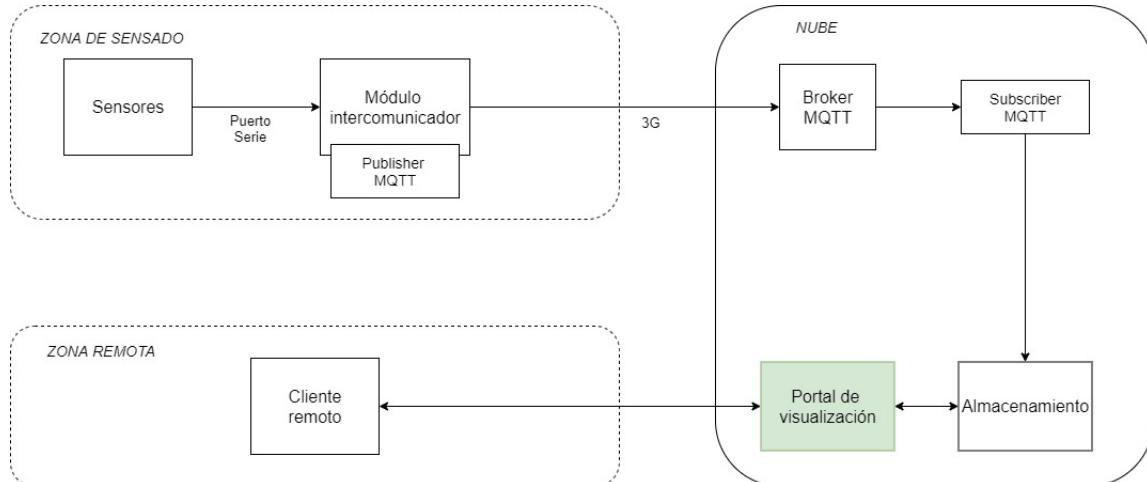


Figura 7.37: Sección Portal Web

En esta sección se presentan los detalles de el front end utilizado para la visualización de las mediciones, cubriendo lo solicitado en el requerimiento **L0.5 - L1.1**. Se analizaron distintas opciones esto, teniendo en cuenta las funcionalidades básicas que se consideraban que debía presentar:

- Descarga de datos

- Visualización en el tiempo
- Valores máximos y mínimos

Para cumplir estos requisitos, se optó por la opción más simple, que fue utilizar una herramienta de monitoreo llamada **Grafana**,<sup>64</sup> hecha en software libre, específicamente con licencia Apache 2.0. Está escrita en Lenguaje Go (creado por Google) y Node.js LTS. Sirve para visualizar información, la cual es recolectada y/o procesada por aplicaciones de terceros. El único objetivo de Grafana es presentar los datos de monitoreo de una manera más fácil de usar y agradable. A continuación se analizan las opciones que presenta Grafana en términos de hosting.

#### 7.4.1 Grafana Enterprise

Existe una versión Enterprise ([grafana.com](http://grafana.com)) que usa complementos para más fuentes de datos. La misma cuenta con una versión Open Source que puede ser configurada dentro de una red local. Esto se logró simplemente configurando el archivo de configuración de la aplicación (Ver anexo 14.4.5), indicando el puerto (por defecto Grafana utiliza el 3000) y la URL con la que se accede a la misma.

Con esta configuración establecida, es posible acceder desde la misma PC apuntando a la dirección IP de loopback (localhost - 127.0.0.1). Sin embargo, para que pueda ser accedido por otro cliente dentro de la misma red local, es necesario montar un servidor Web que hostee la instancia de Grafana.

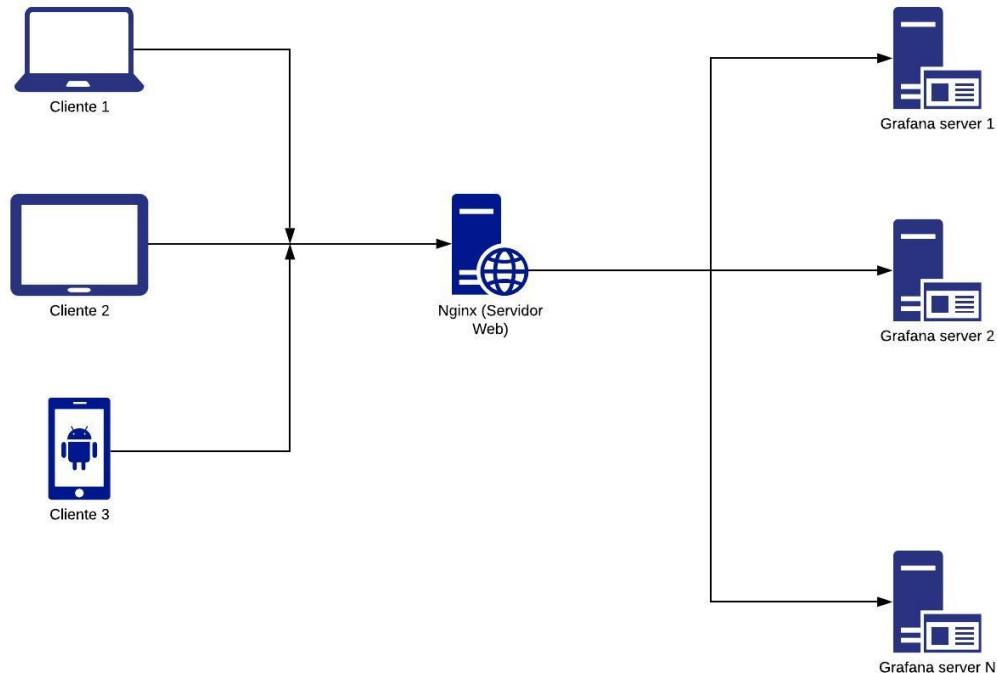


Figura 7.38: Arquitectura del servidor Web y Grafana

El servidor Web elegido fue **Nginx**, una solución Open Source que funciona como servidor y como *reverse proxy*, el cual es un tipo de servidor proxy que recupera recursos

en nombre de un cliente desde uno o más servidores. Estos recursos son entonces regresados al cliente como si se originaran en el propio servidor Web. Contrariamente a un proxy forward, que es un intermediario para que sus clientes asociados se pongan en contacto con cualquier servidor, un proxy inverso es un intermediario para que sus servidores asociados sean contactados por cualquier cliente. Esto significa que se pueden configurar varias instancias de Grafana, como pueden ser, una para los precursores sísmicos y otra para la evaluación de nivel de agua, y que los clientes utilicen siempre el mismo servidor para accederlas.

Para lograr esto, es necesario modificar el archivo de configuración de nginx (ver Anexo 14.4.5) para poder establecer la conexión con las instancias de Grafana. Posteriormente, cualquier cliente dentro de la red local, será capaz de acceder utilizando la dirección IP privada del servidor que contenga nginx y el puerto configurado para accederlo.

Dentro de la aplicación, se pueden configurar distintas visualizaciones, alertas, permisos, entre otras cosas que permiten un uso fácil de la misma. Para el prototipo de temperatura y humedad, se crearon unos gráficos de ejemplo:

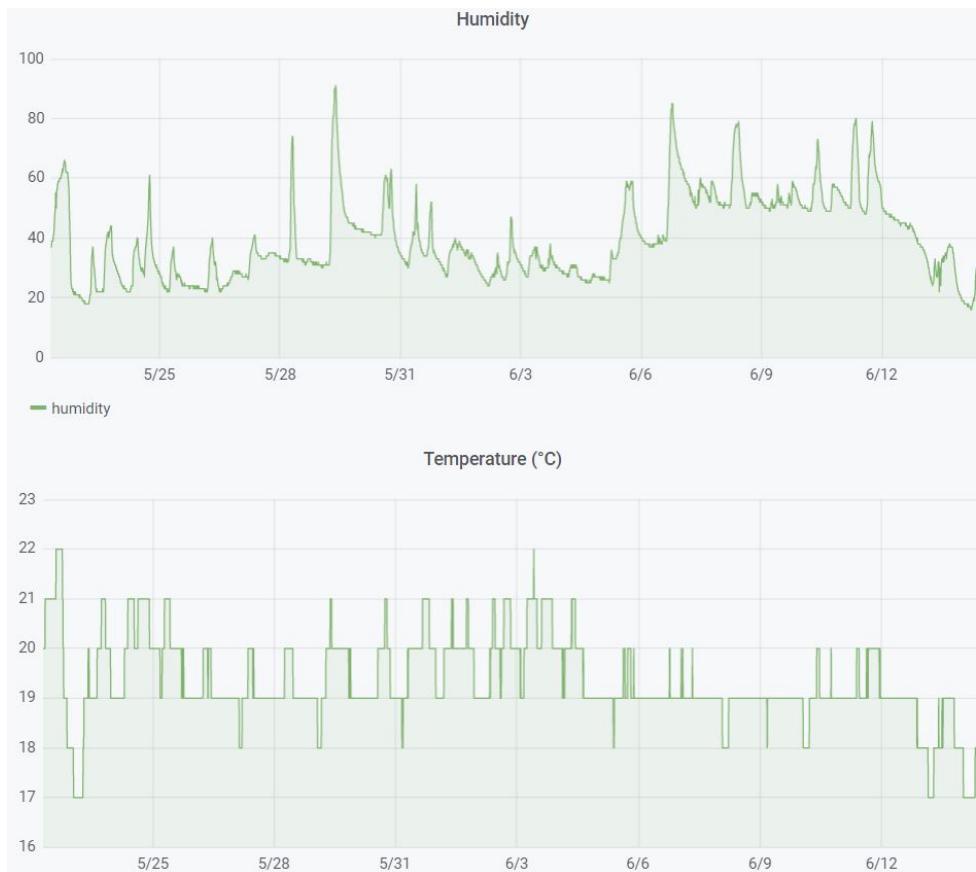


Figura 7.39: Ejemplo de visualizaciones en grafana (Para la creación de alertas ver sección 9.7)

Cada panel es configurable según los datos que presenta. En este caso, Grafana cuenta con un *plug-in* que permite acceder a bases de datos Postgres, lo cual resulta muy útil para este proyecto. Para poder realizar esto, simplemente se ingresan las credenciales de la misma junto con la query correspondiente que permite traer los datos de la métrica a

visualizar. Una vez indexados, se elige la visualización deseada. En la imagen la misma es un gráfico en el tiempo de temperatura y humedad medidas con el prototipo.

Se puede apreciar en la figura 7.39 que los datos tomados son de un mes de sensado. Dentro de la base de datos se puede confirmar que los datos recibidos son iguales a todos los enviados, lo que logra verificar la disponibilidad mayor al 99,9% planteada en el requerimiento **L0.4 - L1.1**.

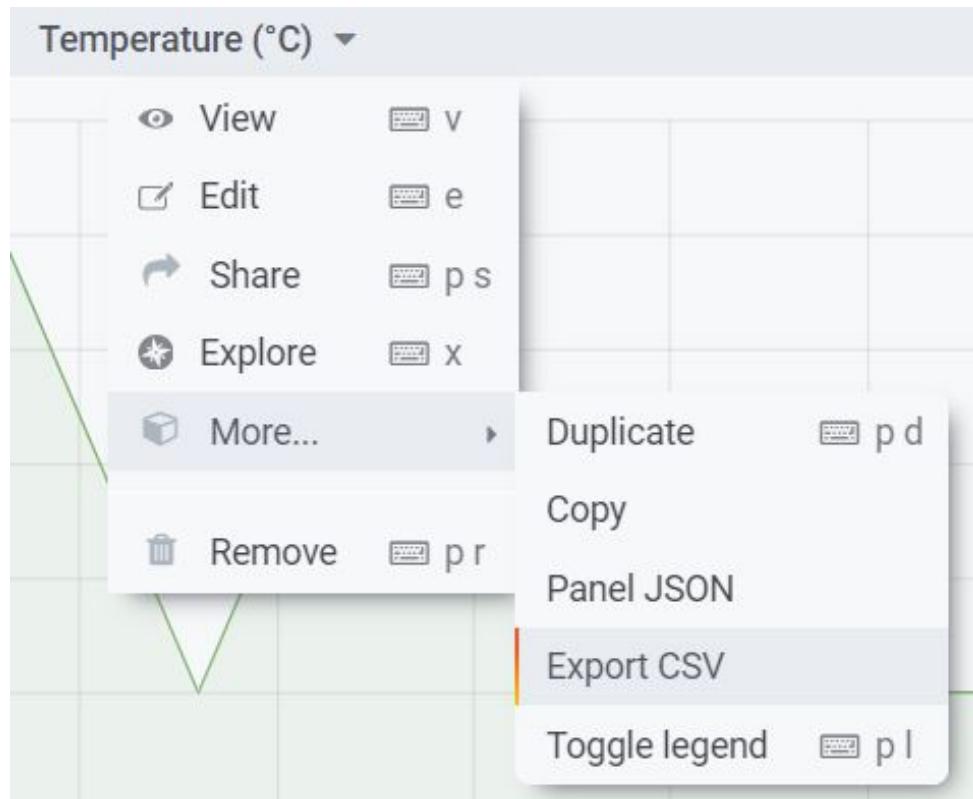


Figura 7.40: Opciones de descarga de datos en Grafana

En la figura 7.40 se muestra las opciones de exportación de datos que provee Grafana. Fácilmente desde el gráfico se pueden descargar los datos en formato CSV para su procesamiento posterior.

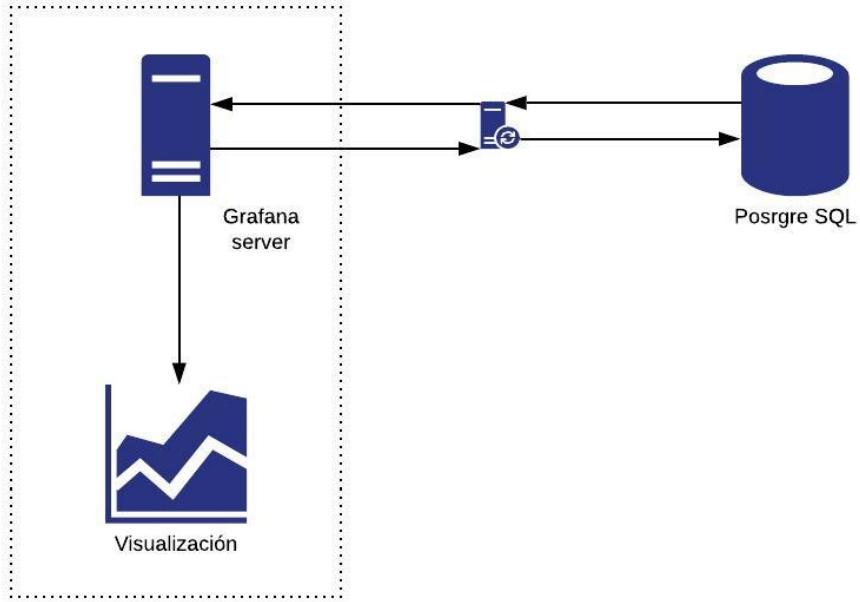


Figura 7.41: Arquitectura Grafana - PostgreSQL

Como bien lo dice el nombre, esta instancia es de código abierto y no tiene costo alguno (ver tabla 5.1). Se necesita sin embargo, contar con un servidor que pueda hostear esta instancia para que pueda ser accedida por otros usuarios. Existe también una versión personalizable de Grafana Enterprise que permite agregar distintas funcionalidades y cuyo precio es variable según la cantidad de adicionales contratados y su configuración.

#### 7.4.2 Grafana Cloud

En este caso, se configura una instancia que es hosteada en la nube. Esto significa que no se necesita configurar ningún archivo ni servidor para poder hacerla funcionar. Una vez creada la instancia, se accede con una URL y una vez dentro, se pueden configurar dashboards y visualizaciones de la misma forma que se mostró en la sección anterior.

Existen en este caso, dos opciones de contratación, que son la versión **Starter** y la versión **Standard**. En la tabla 5.1 se pueden ver los costos de cada una. Vale aclarar que la primera es gratis. Entre las principales diferencias que conciernen a este proyecto, se pueden destacar las siguientes:

	<b>Starter</b>	<b>Standard</b>
<b>Cantidad de Usuarios</b>	1	10
<b>URL personalizable</b>	NO	SI
<b>Cantidad de Dashboards</b>	1	5
<b>Alertas ilimitadas</b>	SI	SI

Tabla 7.2: Diferencias Grafana Cloud

## 7.5 Resumen

La sección analizada es de gran importancia para entender el funcionamiento de punta a punta del proyecto. Se puede entender la elección del protocolo MQTT para la transmisión de los datos. Resulta un protocolo robusto para el tipo de solución presentada, pues con la QoS (calidad de servicio) se asegura una transmisión ligera y rápida de los datos y se asegura la correcta recepción de los mismos. Esto se logra gracias a la estructura del *header* de MQTT, cuyos comandos permiten interpretar las tramas del protocolo. Esto se evidencia en las pruebas realizadas con Wireshark.

Por otro lado, resultó importante la elección de los servicios de Heroku en la nube para el *broker* como para el *subscriber*, pues reducir la configuración y el mantenimiento de los mismos, lo cual queda del lado del proveedor del servicio.

Uno de los pilares más importantes dentro de esta sección fue el desarrollo de la API, la cual sirve como intermediario entre los clientes de MQTT y la base de datos donde se almacenan los datos, pues los clientes consumen de ella y las funciones desarrolladas para escribir o leer de la base. Una vez terminado este paso, todo queda listo para que los datos sean visualizados.

Por último, se destaca la separación de esta solución en módulos o etapas, lo que facilita el mantenimiento de cada una. Para lograr que todo funcione de manera correcta, fue muy importante establecer la compatibilidad entre la entrada y la salida de cada una de ellas. Para eso, la designación del formato YAML como estándar usado, fue lo que permitió poner los datos dentro de un paquete cuyo formato es compatible en gran parte de la industria y facilitar así la compatibilidad de la entrada y salida de cada etapa.

# Capítulo 8

## Módulos complementarios

En el presente capítulo se listan y explican aquellas partes del trabajo que estaban por fuera de los objetivos planteados y se agregaron para darle un complemento extra al proyecto. Principalmente se presenta la caja de calibración que emula la generación de datos del precursor sísmico; también se explica la función de telemando creada, la cual se agregó de forma tal de darle la posibilidad al usuario de enviar comandos hacia el módulo intercomunicador, haciendo la comunicación bidireccional.

### 8.1 Telecomando

La construcción de esta funcionalidad es un extra que se le agregó a este proyecto con el fin de entregar una solución bi-direccional. Esto significa que el usuario no solo recibe los datos enviados desde el módulo intercomunicador, sino que también se le da la posibilidad de enviar comandos al mismo. En la figura 1.1 se aprecia el flujo en una sola dirección. A continuación se presenta el diagrama con el comando agregado.

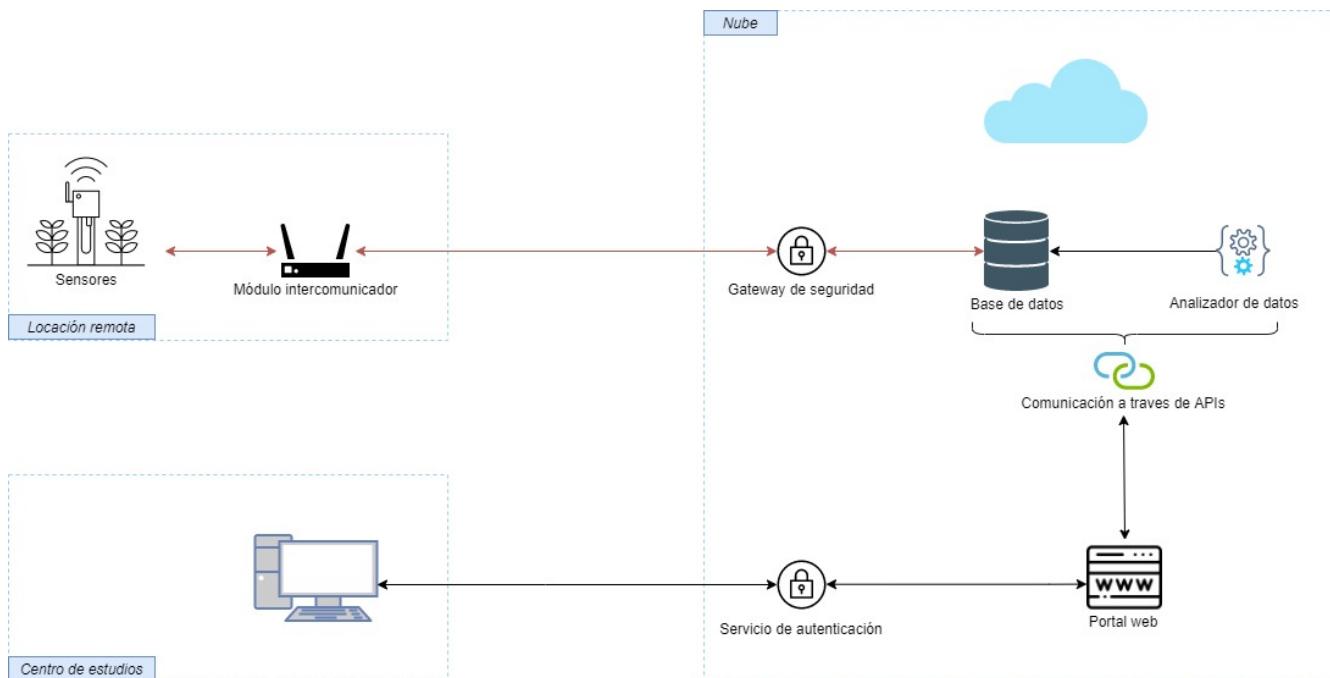


Figura 8.1: Modelo bidireccional del sistema

### 8.1.1 Funcionamiento

La implementación del telecomando se llevó a cabo utilizando MQTT, al igual que en el resto del proyecto, y re-utilizando los scripts de publicación y suscripción. En este caso, en el sentido inverso.

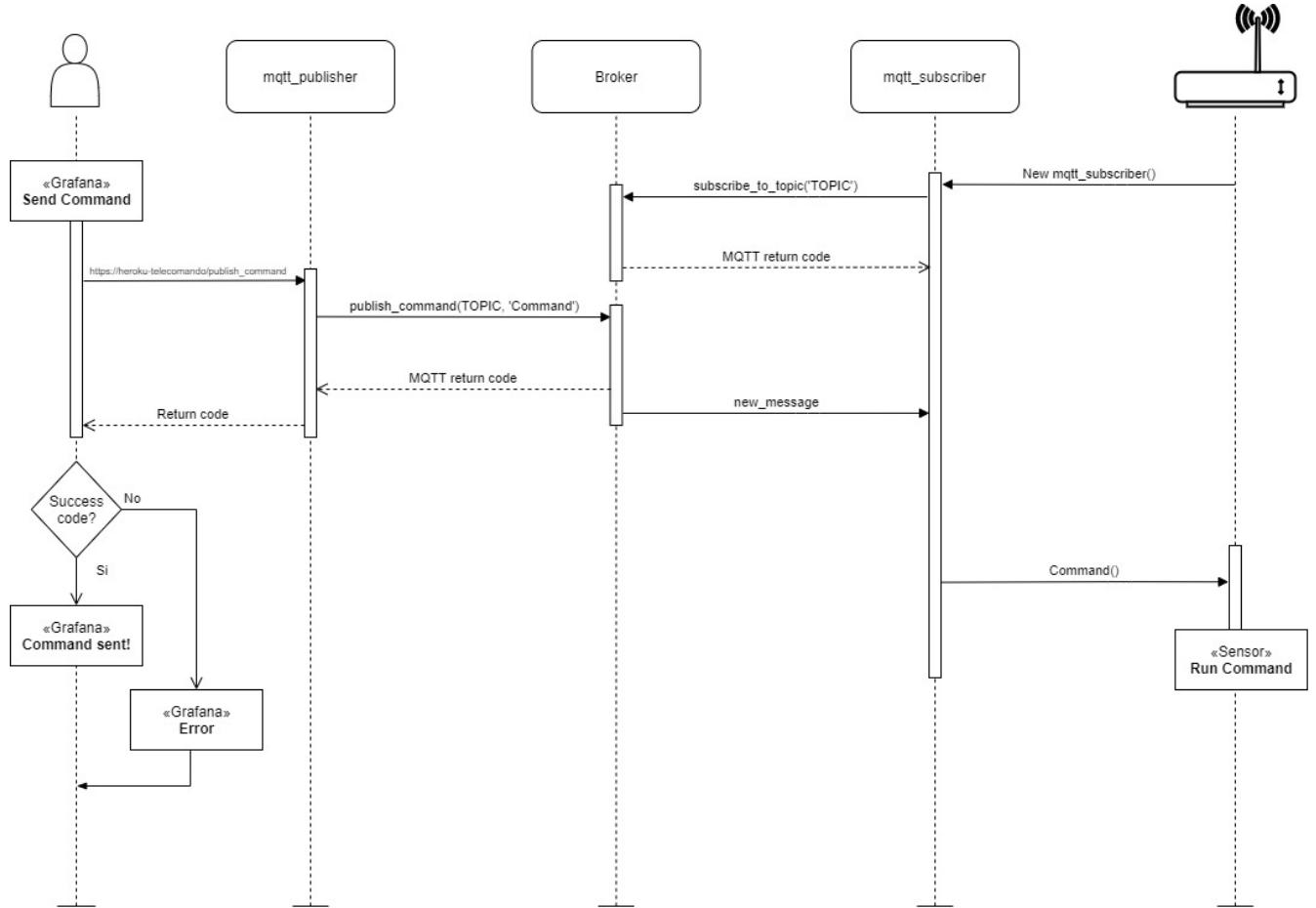


Figura 8.2: Funcionamiento del telecomando - Diagrama UML

El usuario es el encargado esta vez de accionar la publicación del comando. Mediante un botón creado en lenguaje html dentro de la instancia de Grafana, se llama a la misma función de publicación utilizada en el proyecto a través de una API separada en Heroku, dedicada exclusivamente para el telecomando. Como se ve en la figura 8.2, al accionar el botón, la función de html apunta al endpoint `https://heroku-telecomando.com/publish-command`. En el mismo, se define una función que llama al publisher. Se publica el mensaje con el comando que se le quiera enviar al sensor y con un tópico exclusivo para esta función. En este caso, ambos dos se definieron de manera genérica, con el fin para verificar la correcta transmisión de los mismos. Al igual que en el envío de datos desde el sensor, el broker recibe el mensaje publicado y devuelve un código de éxito validando la correcta recepción, o uno de error si algo sucedió. El cliente recibe esto, y mediante el uso de html, se le informa al usuario la correcta o no correcta publicación del comando.

```

<button class="btn btn-success"
onclick="
window.open('https://telecomando.herokuapp.com/publish_command','_blank')
"
> Enviar comando</button>

```

Fragmento de código 3: Fragmento html para la configuración del botón de Telecomando



Figura 8.3: Botón utilizado en Grafana para Telecomando

Desde el lado del módulo intercomunicador, se reutiliza el código de suscripción. En el caso de uso del sensor que envía datos, como se explicó en este informe, el cliente suscriptor está alojado en la nube; el mismo se suscribe a un tópico y recibe los datos que publica el módulo intercomunicador con ese mismo tópico. Una vez recibidos, los publica en una base de datos. En este caso, el módulo intercomunicador es quien se suscribe a un tópico nuevo (el utilizado para los comandos). Una vez que el usuario publica un mensaje, a través del broker, el cliente, alojado en el módulo intercomunicador, recibe el mismo y en lugar de publicar en una base de datos, dispara una acción sobre el/los sensor/es. Para la etapa de pruebas, se utilizó un simple comando '*Shutdown*' que se envió desde una PC de usuario. Una vez recibido en la Raspberry Pi, la misma acciona un script en *Shell*, el cual se utilizó para verificar el correcto funcionamiento del punta a punta. Para una implementación con un sensor real, este script podría enviar una instrucción a un pin de salida, con el fin de enviar un comando a un sensor.

Los **posibles usos** de esto pueden variar entre comandos simples, como encender, o apagar el sensor, como también cambiar algún parámetro del mismo o algún límite de medición, o hasta realizar alguna actualización de código, cambiar alguna aplicación. Todo esto es realizable **sin necesidad de acceder manualmente al equipo**.

## 8.2 Calibrador

Para simular la integración con el sensor se desarrolló un módulo adicional que sensa información analógica cada cierto período de tiempo y lo transmite de forma serial en el formato establecido. Simula lo mismo que se espera del sensor de precursor sísmico pero con otro tipo de dato, no mediciones de campo magnético como ese sensor llevará a cabo.

Además se le agrega un botón que envía un paquete preestablecido, sin una medición real de forma que el módulo pueda funcionar como calibrador y ante un error en el funcionamiento integral punta a punta esté adicional permitiría descartar errores en el sistema de transmisión de información creado.

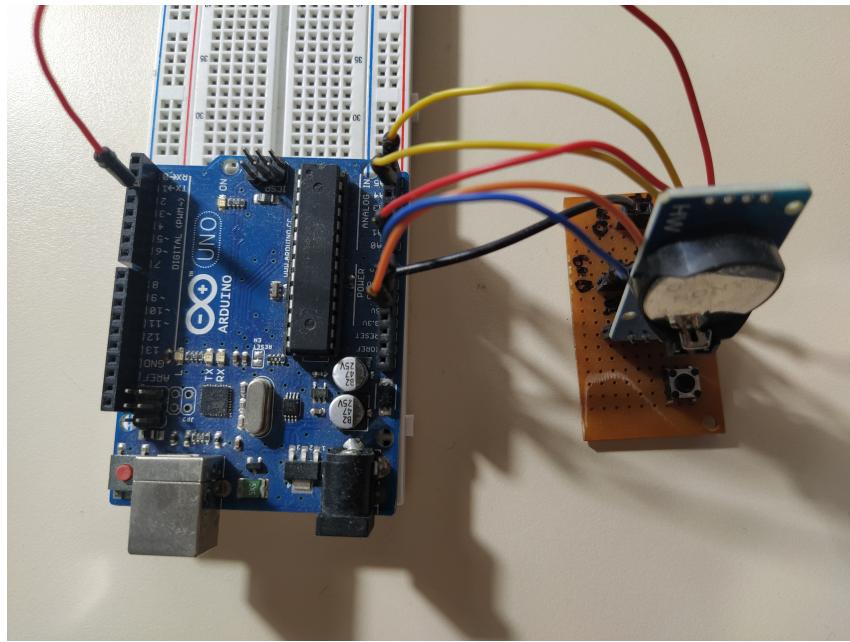


Figura 8.4: Módulo CALSIM.

Para un primer prototipo del mismo se utilizó de base un circuito *Arduino Uno*<sup>65</sup> para el procesamiento y armado del paquete estandarizado utilizando las mediciones de una de sus entradas analógicas. El circuito requiere una alimentación de 5V que puede proveerse desde el mismo módulo intercomunicador a través del puerto USB y su entrada analógica acepta tensiones entre 0 y 5V. Dado que el conversor analógico digital de la placa Arduino utiliza 10 bits se tiene una precisión de la medición de 5mV.

Luego de un poco más de trabajo se logró armar una versión final del CALSIM como se muestra en la figura 8.5. En la misma se ve como tiene conexiones serie de entrada y salida para el módulo intercomunicador, entradas analógicas para una estación de sensado del mismo carácter y por último un botón para enviar un paquete de prueba de calibración.



Figura 8.5: Módulo CALSIM.

En la sección 14.5.1 del Anexo se puede encontrar el ICD ([Cap. 13.7](#)) que describe la relación entre el CALSIM y la plataforma de telemetría. Este documento sirve también de referencia para cualquier estación de sensado genérica que se conecta con la plataforma misma.

## 8.3 Resumen

Los módulos complementarios forman parte del trabajo no como parte de los objetivos planteados sino como funcionalidades extra. En el primer caso, completa la comunicación bidireccional entre el usuario y el módulo intercomunicador. En el caso del calibrador, dada la situación mundial que se vivió como contexto durante la realización del trabajo, sirvió como simulador del sensor con el que se había planeado trabajar inicialmente, dado que el mismo no pudo tener avances significativos debido a la situación mencionada. Esto permitió que los autores trabajaran bajo una plataforma que emula un caso real.

# Capítulo 9

## Construcción del prototipo

En este capítulo se presentan los pasos realizados hasta arribar al prototipo presentado finalmente. Se listan las pruebas realizadas en cada etapa para cada módulo individual así como también los pasos necesarios para su integración. Se demuestra el crecimiento incremental del proyecto desde un punto de vista técnico y se verifican los requerimientos descriptos en el capítulo *Definición del producto*.

### 9.1 Primeras pruebas con puerto serie

Se comunicaron entre si 2 Raspberry Pi con los pre requisitos explicados en la sección 7.1.1 y con Python instalado. De esta forma, en uno de los sistemas se corre el script 14.4.1 para transmitir información por el puerto serie y en el otro 14.4.2 para leer del mismo. Ambos se encuentran en el anexo.

Como resultado, el equipo que corre el primer script actuará de transmisor de la comunicación serie enviando una secuencia creciente de números mientras que el equipo que corre el segundo script actuará de receptor e imprimirá en pantalla la misma secuencia numérica. De esta manera se puede verificar que ambos equipos se encuentran correctamente configurados.

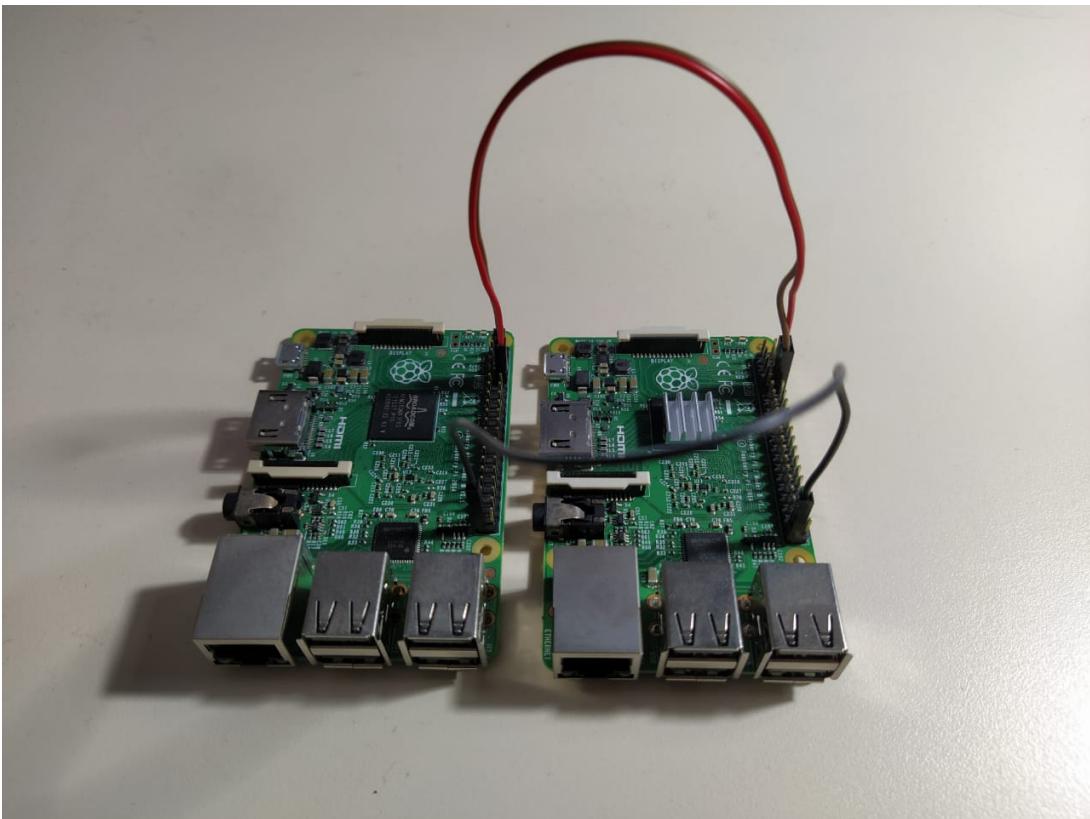


Figura 9.1: Prueba comunicación serial - Conexión

La conexión a los pines es la se corresponde con los mostrados en la figura 7.5, y con los pines de GND conectados.

## 9.2 Primera publicación

Lograda la comunicación por puerto serie, el siguiente paso fue lograr una publicación exitosa utilizando la Raspberry Pi como *publisher* hacia el broker alojado en la nube. Para esto se utilizó un módulo WiFi conectado al puerto USB del dispositivo, con el fin de probarlo sobre una red genérica. Esto además verificaría el requerimiento L0.2 relacionado a la transmisión remota de los datos.



Figura 9.2: Raspberry Pi con módulo WiFi

Para configurar la red a la que se conecta el dispositivo se configura un archivo .CONF con el nombre de la red (SSID) y la contraseña. Ese mismo archivo se guarda en el directorio principal de la tarjeta de memoria que se conecta a la Raspberry con el nombre de *wpa\_supplicant.conf*. En el fragmento de código 13 se aprecia la configuración mencionada. Una vez que bootea, el sistema busca el archivo de configuracion en el directorio */etc/wpa\_supplicant/*. Si la lectura es exitosa podrá conectarse a la red indicada, encendiéndose el LED azul del módulo como se observa en la figura 9.2.

Desde la aplicación Heroku en la nube se configuró un broker MQTT básico *Cute Cat* (ver Tabla 5.1), que permite conexiones de 5 conexiones totales y enlaces de 10 kbps.

Las credenciales provistas son las que posteriormente se utilizan para conectarse al broker desde los clientes. A continuación se definió un tópico de prueba utilizado para los testeos de conexión. En la instancia del broker existe una sección llamada *Users&ACL* donde se pueden definir los tópicos y los permisos sobre los mismos. En este caso se creó */pi/test* como tópico de prueba.

## ACLs

### Note:

- There are two types of ACL rules, topic and pattern. Topic ACLs is applied to a given user. Pattern ACLs is applied to all users.
- Use # for multi level wildcard acl.
- Use + for single level wildcard acl.
- Creating and deleting users and ACLs are asynchronous tasks and may take up to a minute. Poll list APIs to see when ready.

For API docs look at [HTTP API](#)

Type	Pattern	Read/Write	
pattern	/pi/sys-ops	true/true	<button>Delete</button>
pattern	/pi/temp	true/true	<button>Delete</button>
pattern	/pi/test	true/true	<button>Delete</button>

Pattern
Topic
Pattern
 Read Access?
 Write Access?
+ Add

Figura 9.3: Tópicos agregados sobre la instancia del Broker

Configurado esto, se procedió a realizar la primera prueba de publicación utilizando una versión temprana del *publisher* y un paquete de datos de ejemplo provisto (Ver Script 12 del Anexo).

```
pi@raspberrypi:~ $ python mqttPublisher.py
Done
Done
Done
Done
Done
Done
Done
2.47393894196
```

Figura 9.4: Resultados de la primera publicación

En la figura 9.4 se ven los resultados del envío del paquete hacia el broker. Los indicadores *Done* marcan el correcto envío de los paquetes, los cuales se dividieron en *chunks* separados por *END\_SEGMENT* (ver script 12 del Anexo). La última línea indica el tiempo en segundos que tardó en enviar el paquete completo. En este caso se probó con un paquete de 810 bytes.

Del lado del subscriber, como se explica en la sección 7.2.5.3, se montó un script con un pipeline automático en Heroku conectado con Git. Es decir que al actualizar el código, se sincronizaría con Heroku. Para lograr que el script corra de manera independiente se define un archivo llamado *Procfile* sin extensión en el cual se define un *worker*, el cual será un comando que llama al script. El mismo se puede encontrar en el fragmento de código 14 del anexo.

```

2020-10-06T08:06:43.391647+00:00 heroku[worker.1]: Starting process with command `python mqttSubscriber.py`
2020-10-06T08:06:44.199987+00:00 heroku[worker.1]: State changed from starting to up
2020-10-06T08:06:48.566720+00:00 app[worker.1]: Subscribed to /pi/test with result code 0
2020-10-06T08:06:48.566880+00:00 app[worker.1]: Subscribed to /pi/temp with result code 0
2020-10-06T08:06:48.567091+00:00 app[worker.1]: Connected with result code 0

```

Figura 9.5: Subscriber conectado desde Heroku

Corriendo el comando *heroku logs* se pueden obtener los logs de la aplicación de Heroku. Allí se puede verificar la correcta conexión al broker y la suscripción al tópico */pi/test*.

```

1 2020-10-06T17:21:51.596919+00:00 app[worker.1]: Message received:
2 2020-10-06T17:21:51.597131+00:00 app[worker.1]: /pi/test      20191011172018
3 2020-10-06T17:21:51.597133+00:00 app[worker.1]: ID:000001
4 2020-10-06T17:21:51.597133+00:00 app[worker.1]: LEN:018008580
5 2020-10-06T17:21:51.597133+00:00 app[worker.1]: BEGIN_SEGMENT
6 2020-10-06T17:21:51.597134+00:00 app[worker.1]: SEGMENT_ID:001
7 2020-10-06T17:21:51.597134+00:00 app[worker.1]: 20191011172018
8 2020-10-06T17:21:51.597134+00:00 app[worker.1]: MICROSEC:000000
9 2020-10-06T17:21:51.597135+00:00 app[worker.1]: RATE:0500
10 2020-10-06T17:21:51.597135+00:00 app[worker.1]: FLAG1:001
11 2020-10-06T17:21:51.597135+00:00 app[worker.1]: FLAG2:002
12 2020-10-06T17:21:51.597135+00:00 app[worker.1]: FLAG3:003
13 2020-10-06T17:21:51.597136+00:00 app[worker.1]: FLAG4:004
14 2020-10-06T17:21:51.597136+00:00 app[worker.1]: BEGIN_DATA
15 2020-10-06T17:21:51.597136+00:00 app[worker.1]: 1111111111
16 2020-10-06T17:21:51.597136+00:00 app[worker.1]: END_DATA
17 2020-10-06T17:21:51.597137+00:00 app[worker.1]: END_SEGMENT
18 2020-10-06T17:21:51.597441+00:00 app[worker.1]: Amount of receptions so far: 1
19 2020-10-06T17:21:53.384927+00:00 app[worker.1]: Message received:
20 2020-10-06T17:21:53.385097+00:00 app[worker.1]: /pi/test      Finish
21 2020-10-06T17:21:53.385180+00:00 app[worker.1]: Amount of receptions so far: 2

```

Fragmento de código 4: Heroku logs - Mensaje recibido por el subscriber

Arriba se pueden ver los logs de Heroku una vez que se envió la primera publicación. Se observa que el subscriber recibe correctamente el mensaje y lo imprime en pantalla. Al final recibe un mensaje de *Finish* que le indica que el mensaje finalizó. Con esto se verifica el correcto funcionamiento de los tres end points: publisher, broker y subscriber.

## 9.3 Operacionalización de los clientes

Habiendo superado las primeras pruebas, el siguiente paso fue diseñar módulos de forma tal de que ambos clientes, *subscriber* y *publisher* estén operativos y puedan enviar y recibir paquetes genéricos y con las velocidades indicadas en los requerimientos. Para ambos dos se utilizaron los métodos definidos en la sección 7.2.5.

Desde el lado del cliente que publica, como se menciona en la sección 7.2, el mismo utiliza métodos para dividir los paquetes según un límite de tamaño de datos para evitar

mandar todo junto y saturar el enlace. En este caso se usaron porciones de 100 kB. La función se puede encontrar en el fragmento de código 15 del Anexo. Además se definió un QoS 1 para la publicación de los mensajes.

```
piece_of_file size: 100033
chunk number 178
piece_of_file size: 100033
chunk number 179
piece_of_file size: 100033
chunk number 180
piece_of_file size: 9451
chunk number 181
piece_of_file size: 33
amount of chuncks to send: 181

Sending multiple payloads...
Published message with id: 1
Published message with id: 2
Published message with id: 3
Published message with id: 4
Published message with id: 5
Published message with id: 6
Published message with id: 7
Published message with id: 8
```

```
Published message with id: 179
Published message with id: 180
Published message with id: 181
```

```
All segments sent!
Elapsed time: 363.49287819862366 seconds
Published message with id: 182
```

(a)

(b)

Figura 9.6: Resultados de la publicación con publisher operativo

En la figura 9.6 se pueden ver los resultados de la operacionalización del *publisher*. Se ve que se indican los tamaños en bytes de los subpaquetes enviados y luego el cliente indica su correcta publicación. Una vez concluido se envía el mismo mensaje de *Finish* para indicar la finalización del paquete completo. Se puede ver que en la prueba se envió un paquete con 181 segmentos, los cuales en total conformaban un total de 17,1 MB. Esta prueba se realizó utilizando el módulo WiFi. Con el tiempo en que tardó en publicar todo, se puede hacer un estimado de la velocidad de envío, la cual es cercana a unos 400 kbps.

Desde el lado del *subscriber* no se realizaron cambios significativos ni de funcionalidades en esta etapa. Simplemente se lo estructuró con el paradigma de Programación Orientada a Objetos, definiendo clases y métodos. En este caso, los métodos son los mismos que los definidos en la sección 7.2.5.

## 9.4 Reemplazo del módulo WiFi por módem 3G

El siguiente paso fue realizar las mismas pruebas reemplazando el módulo WiFi por el módem presentado en la sección 7.2.1. En esta misma sección se explica la configuración del archivo wvdial.conf que fue necesaria para lograr la conexión exitosa de la Raspberry a internet.

Además, se realizó la misma prueba de publicación pero esta vez sobre la red 3G.

```
Published message with id: 704
Published message with id: 705
Published message with id: 706
Published message with id: 707
Published message with id: 708
Published message with id: 709
Published message with id: 710
Published message with id: 711
Published message with id: 712
Published message with id: 713
Published message with id: 714
Published message with id: 715
Published message with id: 716
Published message with id: 717
Published message with id: 718
Published message with id: 719
Published message with id: 720
Published message with id: 721

All segments sent!
Elapsed time: 1445.0094361305237 seconds
Published message with id: 722
pi@raspberrypi:~/PythonScripts/MQTT/mqtt_publisher $ scrot
```

Figura 9.7: Prueba ded publicación sobre red 3G

Se ve que el tiempo de publicación total del paquete es de 1445 segundos. Teniendo en cuenta los headers, la velocidad estimada es de 98 kbps, logrando enviar un paquete de una hora de datos en aproximadamente 24 minutos.

El éxito en el envío de los datos verifica el requerimiento **L0.2 - L1.1** relacionado a la transmisión de los datos por red celular, en particular por 3G.

## 9.5 Incorporación de la base de datos

Seguidamente, para el almacenamiento de los datos se incorporó dentro de la misma instancia de Heroku una base de datos, es decir en la nube. Esto permite verificar el requerimiento **L0.2 - L1.3** que plantea el almacenamiento de los datos en la nube. Los detalles del diseño, acceso e implementación de la misma se pueden encontrar en la sección 7.3. Se adaptó el script del subscriber de forma de darle la funcionalidad de escribir los datos recibidos en la base de datos, utilizando los métodos de la API de comunicación.

Dentro de la misma sección se puede ver como funciona la misma.

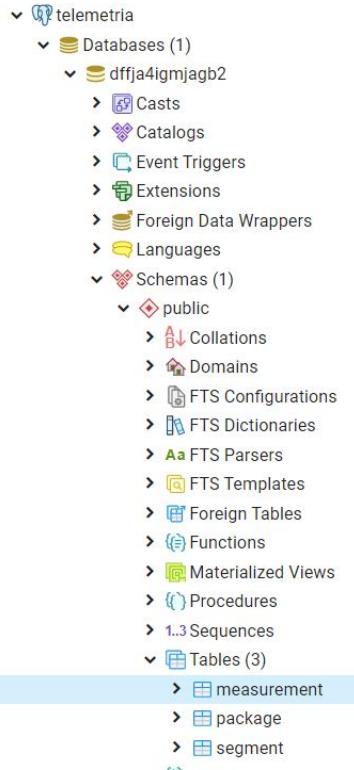


Figura 9.8: Esquema de la base de datos creada en PosgreSQL UI.

Se procedió a realizar la misma prueba con el mismo paquete de prueba. Una vez que el subscriber recibe los datos, el mismo realiza la conexión con el servidor con las credenciales necesarias. Una vez hecho esto, escribe los datos leídos con métodos definidos. En particular se utiliza la función que se encuentra en el Fragmento de código 16 del Anexo.

## 9.6 Escalabilidad a múltiples aplicaciones

Una vez listas las distintas partes del proyecto, fue necesario adaptarse a nuevas aplicaciones previo a poder hacer una prueba integral ya que dado el contexto mundial del COVID (sección 1.2) la obtención de mediciones del precursor sísmico debió posponerse. Esto va en concordancia con el requerimiento **L0.1**.

Con este objetivo se adoptó el estándar YAML para la comunicación de entrada, el mismo se explicó en la sección 6.4.1 y el paquete de ejemplo se muestra en el anexo 14.3.1. A partir de este punto, se pudieron adaptar los scripts de forma estándar sin la necesidad de parsear el paquete de datos que se usaba de ejemplo para su correcto envío, debido a que con el formato YAML simplemente hubo que utilizar las librerías de Python correspondientes. Esto ahorró mucho trabajo sobre cada módulo medible en líneas de código. El fragmento de código 17 del Anexo se puede ver el método utilizado para el parseo del paquete.

## **9.7 Incorporación de visualización de datos**

A continuación se incorporó el portal web para la visualización de los datos enviados. En la sección 7.4 se explican los detalles del mismo. Lo más importante a resaltar es la capacidad de conexión a la base de datos con el fin de realizar distintos tipos de operaciones.

Se desarrolló una visualización de prueba en la cual se monitorea los datos explicados en la subsección siguiente. Con esto se pudo validar el correcto envío de los datos y verificar la disponibilidad de la solución de forma visual. Al mismo tiempo se comprobó la posibilidad de generar alertas. Esto es posible en base a criterios elegidos por el usuario. Es decir que en el caso de establecer alertas en base a un outlier, como marca uno de los requerimientos, esto es posible.

The screenshot shows the Grafana Alerting interface. At the top, there is a bell icon and the title "Alerting" followed by "Alert rules & notifications". Below this, there are two tabs: "Alert Rules" and "Notification channels", with "Notification channels" being the active tab. The main area is titled "Edit Notification Channel" and shows a configuration form for a channel named "Alerta genérica" of type "Email". The configuration includes several toggle switches: "Default (send on all alerts)" (off), "Include image" (off), "Disable Resolve Message" (on), and "Send reminders" (off). Below this, there is a section titled "Email settings" with a "Single email" toggle switch (off) and an "Addresses" field containing "email@ejemplo.com". A note at the bottom states: "You can enter multiple email addresses using a ";" separator".

(a)

Alert

The screenshot shows the configuration of an alert rule named "Alerta de exceso de Temperatura (°C)". The rule is set to evaluate every 1m for 5m. The conditions section contains a single condition: "WHEN avg () OF query (A, 5m, now) IS ABOVE 23". The "No Data & Error Handling" section defines two actions: "If no data or all values are null" (Set State To: No Data) and "If execution error or timeout" (Set State To: Alerting).

(b)

Figura 9.9: Ejemplo de alertas en Grafana

En la figura 9.9 se muestra un ejemplo de la configuración de un alerta en base a un criterio que se toma en base a los datos obtenidos por la query sobre la base de datos. El criterio es de ejemplo a raíz de los datos de temperatura tomados. Si la temperatura supera los 23 grados centígrados, se enviará un correo a la dirección seleccionada. Esto

verifica el requerimiento **L0.3**, aunque en lugar de alertas por SMS se pueden enviar alertas por email, Telegram y otros canales de comunicación instantáneos.

Posteriormente, como se explica en la sección 8.1, se agregó la opción para poder enviar un comando remoto desde el mismo portal. Esto le agregó una funcionalidad extra a la herramienta de visualización para que el usuario pueda interactuar con el módulo intercomunicador.

## 9.8 Prueba punta a punta y automatización del flujo

Para realizar una prueba punta a punta integral se desarrolló un sensor de temperatura y humedad. A su vez esto sirvió para probar la escalabilidad a múltiples aplicaciones.

En la imagen 9.10 se muestra el módulo de publicación a la izquierda y el módulo de sensado a la derecha.

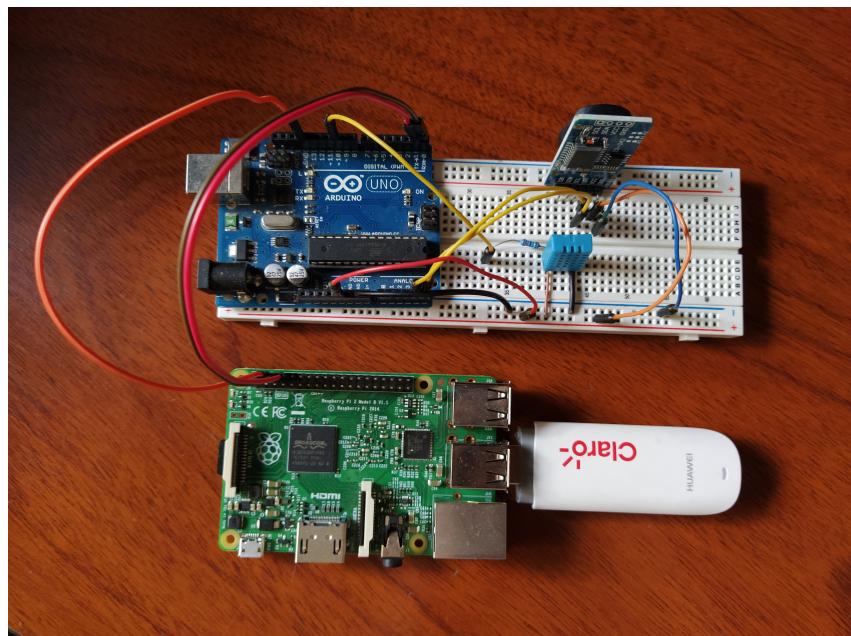


Figura 9.10: Prueba punta a punta.

Por último, para probarlo de forma continua y simulando un ambiente productivo, se automatizó el flujo y utilizando el mismo sensor de temperatura y humedad se lo mantuvo en funcionamiento durante un mes. La automatización de publicación se mostró en la sección 7.2.5.2 y en la sección 7.2.5.3 se indica la disponibilidad en la nube del cliente suscriptor.

Con el sensor conectado por puerto serie a la Raspberry Pi, se transmitieron los datos a través de la misma hacia la nube utilizando el publisher como cliente de envío de datos.

```

Starting serial comm...
OK!
('Starting to publish file:', 'reception.yml', '...')
('empty chunks size:', 36)
('piece_of_file size:', 616)
('chunk number', 1)
('piece_of_file size:', 21)
('amount of chuncks to send:', 1)

Sending multiple payloads...
('Published message with id:', 1)

All segments sent!
('Elapsed time:', 2.0069329738616943, 'seconds')
('Published message with id:', 2)
OK!

```

Figura 9.11: Publicación de datos de temperatura y humedad

De manera continua y cada diez minutos se tomaba una medición genérica de temperatura y humedad que transmitía como se ve en la figura 9.11. El promedio de tiempo de envío de cada mensaje fue de 2 segundos. Se verifica el requerimiento **L0.4 - L1.2**.

```

2020-10-15T01:39:17.741832+00:00 app[worker.1]: Message received:
2020-10-15T01:39:17.741849+00:00 app[worker.1]: /pi/temp package_id: 4
2020-10-15T01:39:17.741850+00:00 app[worker.1]: package_len: 18008580
2020-10-15T01:39:17.741850+00:00 app[worker.1]: package_date: 20201014223314
2020-10-15T01:39:17.741850+00:00 app[worker.1]: segments:
2020-10-15T01:39:17.741851+00:00 app[worker.1]: - segment_id: 1
2020-10-15T01:39:17.741851+00:00 app[worker.1]: segment_date: 20201014223314
2020-10-15T01:39:17.741852+00:00 app[worker.1]: segment_microsec: 0
2020-10-15T01:39:17.741852+00:00 app[worker.1]: segment_rate: 320
2020-10-15T01:39:17.741853+00:00 app[worker.1]: flag1: flag1
2020-10-15T01:39:17.741854+00:00 app[worker.1]: flag2: flag2
2020-10-15T01:39:17.741854+00:00 app[worker.1]: flag3: flag3
2020-10-15T01:39:17.741854+00:00 app[worker.1]: flag4: flag4
2020-10-15T01:39:17.741855+00:00 app[worker.1]: payload: T21.00H15.00
2020-10-15T01:39:17.741855+00:00 app[worker.1]: - segment_id: 2
2020-10-15T01:39:17.741855+00:00 app[worker.1]: segment_date: 20201014223514
2020-10-15T01:39:17.741856+00:00 app[worker.1]: segment_microsec: 0
2020-10-15T01:39:17.741856+00:00 app[worker.1]: segment_rate: 320
2020-10-15T01:39:17.741856+00:00 app[worker.1]: flag1: flag1
2020-10-15T01:39:17.741857+00:00 app[worker.1]: flag2: flag2
2020-10-15T01:39:17.741857+00:00 app[worker.1]: flag3: flag3
2020-10-15T01:39:17.741857+00:00 app[worker.1]: flag4: flag4
2020-10-15T01:39:17.741857+00:00 app[worker.1]: payload: T21.00H16.00
2020-10-15T01:39:17.741858+00:00 app[worker.1]: - segment_id: 3
2020-10-15T01:39:17.741858+00:00 app[worker.1]: segment_date: 20201014223714
2020-10-15T01:39:17.741858+00:00 app[worker.1]: segment_microsec: 0
2020-10-15T01:39:17.741858+00:00 app[worker.1]: segment_rate: 320
2020-10-15T01:39:17.741859+00:00 app[worker.1]: flag1: flag1
2020-10-15T01:39:17.741859+00:00 app[worker.1]: flag2: flag2
2020-10-15T01:39:17.741859+00:00 app[worker.1]: flag3: flag3
2020-10-15T01:39:17.741859+00:00 app[worker.1]: flag4: flag4
2020-10-15T01:39:17.741860+00:00 app[worker.1]: payload: T21.00H16.00
2020-10-15T01:39:17.741908+00:00 app[worker.1]: Amount of receptions so far: 9
2020-10-15T01:39:19.578296+00:00 app[worker.1]: Message received:
2020-10-15T01:39:19.578458+00:00 app[worker.1]: /pi/temp Finish
2020-10-15T01:39:19.578552+00:00 app[worker.1]: Amount of receptions so far: 10
2020-10-15T01:39:19.684378+00:00 app[worker.1]: Package stored successfully

```

Figura 9.12: Recepción de los datos desde la nube

Desde el subscriber ubicado en la nube se imprimieron por pantalla los resultados para cada uno de los paquetes enviados.

	temperature numeric	humidity numeric	date timestamp without time zone	package_id numeric	segment_id numeric
1	[null]	[null]	2000-01-01 03:38:55	1	1
2	23.00	46.00	2000-01-01 03:48:55	1	2
3	24.00	45.00	2000-01-01 03:58:56	1	3
4	24.00	44.00	2000-01-01 04:08:57	2	1
5	24.00	43.00	2000-01-01 04:18:57	2	2
6	24.00	44.00	2000-01-01 04:28:58	2	3

Figura 9.13: Almacenamiento de los datos recibidos

En la figura 9.13 se observa el correcto almacenamiento de los datos en la base sobre la que escribe el subscriber.

```
2020-10-15T01:33:18.797065+00:00 app[worker.1]: Failed to insert records with error: duplicate key value violates unique constraint "package_pkey"
2020-10-15T01:33:18.797068+00:00 app[worker.1]: DETAIL: Key (id)=(3) already exists.
```

Figura 9.14: Ejemplo de error de envío

Como caso de ejemplo se muestra el error que se imprime a través del script de suscripción en caso de enviar un paquete cuyo ID ya existe en la base de datos. Se ve que eso genera un error y no se permite la escritura de la entrada.

En la figura 7.39 se aprecia un ejemplo de mediciones tomadas con el sensor durante un período cercano a un mes y el resultado mostrado desde el portal de Grafana, lo que termina de completar el proceso automático de punta a punta.

## 9.9 Resumen

La construcción de este prototipo verifica el cumplimiento de los objetivos planteados a lo largo de este proyecto. Nuevamente se destaca el diseño modular y la posibilidad de ir construyendo cada una de las etapas por separado. Esto permitió validar el correcto funcionamiento de cada una de ellas y una mayor facilidad a la hora de integrarlas.

Por último, la posibilidad de construir un sensor genérico con materiales económicos permitió verificar de manera exitosa la adaptabilidad del proyecto a cualquier tipo de medición y con esto se logró tomar mediciones simulando un sensor productivo que funcione a lo largo de un período largo de tiempo.

# **Capítulo 10**

## **Conclusiones**

En el presente trabajo se trabajaron sobre distintos objetivos alineados hacia un fin común. Los mismos fueron planteados en la presentación del Trabajo Profesional (sección 14.1 del Anexo), previo a la realización del mismo, y detallados en este informe. El fin común fue el envío exitoso de los datos desde sensores genéricos hacia una estación de trabajo remota. Para lograr esto, se contaba con la necesidad de que estos datos llegaran de manera veloz, íntegra y consistente, de forma de asegurar la validez de los mismos en el mismo tiempo en que fueran sensados.

La realización de este objetivo general fue exitosa a través de la solución presentada en las secciones de este informe, consistiendo de distintas etapas que sirvieron para asegurar cada uno de los objetivos planteados. La compatibilidad de estos módulos fue importante para asegurar la integridad de los datos. Las conexiones entre los mismos, mediante herramientas de enlace (puerto serie, 3G, entre otros) fueron los que aseguraron la velocidad de los mismos. Por último, las herramientas de visualización sirvieron para asegurar tanto la integridad como la consistencia de la información.

A continuación se analizan los objetivos específicos planteados en la sección 2.3 y la resolución de los mismos presentadas en este trabajo.

## 10.1 Resultados obtenidos

En base al alcance del proyecto planteado en el Plan de TP Profesional (sección 14.1 del Anexo), se plantearon una serie de objetivos en la sección 2.3. Los resultados fueron los siguientes:

**OBJ.1** *Recibir en la RP los datos del sensor a través de una comunicación por puerto serie.*

Mediante la utilización scripts codificados en Python, tanto en la Raspberry Pi como en el dispositivo que envía los datos, se logró con éxito el envío por este protocolo. Los detalles se explican en la sección 7.1. Cabe destacar que para este objetivo particular, se contó con una idea original de trabajo en conjunto con el proyecto de "EstaciÓn de Sensado para Emisiones de Campo Magnético a Frecuencias Extremadamente Bajas ysu Aplicación en la Predicción de Sismos" llevado a cabo por el Laboratorio de Radiación Electromagnética de forma de poder transmitir los datos generados por los sensores implicados en este proyecto. Debido al contexto mundial bajo el que se realizó este proyecto, por la pandemia de COVID-19, el proyecto mencionado no pudo continuar con los avances pertinentes y que conciernen al trabajo presentado en este informe. Es por eso, que para la realización de este objetivo, se diseñó una caja de calibración para la simulación de los datos que se debían obtener originalmente del precursor sísmico, de forma de poder transmitir un paquete de datos genérico que cumpliera con los requisitos. Los detalles se encuentran en la sección 8.2.

La completitud del objetivo presentado se relaciona con el S.1 planteado en el Plan y detallado en la sección 4.1, donde se plantea la necesidad de la correcta captura de datos. El mismo está asociado a los requerimientos L0.1 y L0.2. Al mismo tiempo contribuye a los requerimientos técnicos (RT) RT.1, al ser la primera etapa o módulo del trabajo.

**OBJ.2** *Decodificar los datos recibidos en crudo para convertirlos en algún formato adaptado.*

Para asegurar la consistencia de los datos, fue importante la utilización de estándares para estructurar los mismos, de manera que el formato con el que se enviaran fuera compatible en la entrada y salida de cada uno de los módulos. En este caso se optó por la utilización del formato de serialización de datos YAML, muy utilizado en la industria. En la sección 6.4.1.

Este objetivo se asocia tanto con el S.1 como con el S.2, siendo que la necesidad de contar con formato estandarizado de datos se plantea tanto para la recepción como para el envío de paquetes desde el módulo intercomunicador.

**OBJ.3** *Procesar los datos para obtener una visualización de los mismos en el tiempo.*

A través del almacenamiento de los mismos en la base de datos, los mismos resultan accesibles para su visualización. Esto se logró mediante la creación de la base alojada en la nube (sección 7.3), junto con el desarrollo de la API que permite realizar

pedidos por HTTP sobre la misma (sección 7.3.2). Por último, para lograr la visualización, se utilizó la herramienta de monitoreo Grafana, que permite consumir de la base de datos y realizar distintos gráficos (sección 7.4).

Completoando esto, se contribuye con lo planteado en S.3, donde se plantea la visualización de los datos en formato de tiempo.

**OBJ.4** *Procesar datos para detectar anomalías y establecer alertas.*

Al igual que en el punto anterior, la solución que ofrece la herramienta Grafana provee un set de configuraciones que permite el establecimiento de alertas. Las mismas funcionan de forma tal que se establece un criterio para la misma y se envía un mensaje al usuario mediante email o Telegram como los medios más relevantes. A este objetivo, se le suma la funcionalidad de Telecomando explicada en la sección 8.1, la cual, ante la detección de estas anomalías, permite enviar un comando al sensor de forma de realizar distintas operaciones sobre el mismo, como apagarlo por ejemplo.

La relación de este objetivo con los Smart se asocia con el objetivo adicional (Stretch) S.5, el cual plantea la opción de incorporar alertas en base a criterios del usuario.

**OBJ.5** *Compresión de datos y transmisión a un servidor en la nube.*

El cumplimiento de este objetivo fue gracias al aprovechamiento de los servicios Cloud que Heorku provee. Esto permitió recibir los datos enviados por el módulo intercomunicador y a partir de allí operar todo en la nube. La realización de este punto en particular se llevó a cabo en tres pasos, que consistieron en establecer el Broker MQTT (sección 7.2.4) que recibe los tópicos publicados; el script de suscripción, el cual se suscribe al tópico en el Broker para obtener la información publicada y realizar operaciones (sección 7.2.5.3); y la creación de la base de datos para almacenar esos datos (sección 7.3). Estos tres elementos fueron los principales para lograr tener una correcta transmisión a la nube.

Este objetivo se asocia con el S.2, donde se plantea la necesidad de un sistema automático para el envío de los datos desde el módulo y la recepción desde otra ubicación, que en este caso es la nube.

**OBJ.6** *Publicar página web en un host alojado en la nube (datos extra + botón para descargar datos). El servidor podría ser uno propio o un hosting gratuito.*

La realización de este punto fue gracias a las distintas alternativas que el software Grafana ofrece al usuario. Entre ellas se cuenta con la posibilidad de usar la versión *Enterprise* que permite levantar una instancia de la misma de manera local y ser accedida por clientes dentro de la misma red a través de un proxy reverso. Estos detalles se explican en la sección 7.4.1. Por otro lado, se ofrece también una versión Cloud, la cual los autores consideran más pertinente si se desea acceder desde cualquier locación con acceso a internet. Las funcionalidades son las mismas

pero con una instancia hosteada por la misma empresa que ofrece el servicio, lo que reduce el mantenimiento. La visualización de datos es parte de la instancia como también lo es la opción de descarga de datos. Fue fundamental también el desarrollo de la base de datos para que la instancia de Grafana pueda tomar datos de ella.

La completitud de este objetivo se relaciona con el S.3, donde se plantea la necesidad del procesamiento y almacenamiento de las imágenes obtenidas. Así también guarda relación con el S.4, donde se presenta la opción de descarga de datos mediante una llamada del usuario.

Todos estos fueron completados de manera exitosa y en el tiempo planteado en la sección 4.1. Los objetivos SMART planteados aquí engloban los objetivos específicos listados arriba. Y la finalización de estos se corresponde con la finalización de los SMART, planteados también en el Plan de TP Profesional.

Los requerimientos presentados en la sección 3.1 guardan relación con los objetivos. El detalle de los mismos se presenta en la siguiente sección. El tiempo correspondiente a la finalización de todos los objetivos fue de un año y medio, desde la presentación del Plan del proyecto. Esto se corresponde con lo planteado al principio del proyecto y destaca el trabajo realizado por los autores al haberlo realizado bajo el contexto del COVID-19 y las limitaciones detalladas en la sección 1.2.

## 10.2 Matriz de verificación de requerimientos

En la sección 3.1 se presentaron los requerimientos específicos de los usuarios, planteados en tres niveles, correspondientes al nivel técnico que implicaba cada uno de los tres. En la siguiente tabla se presenta la matriz de verificación de cada uno de ellos, completados desde el nivel más bajo hacia el más alto, es decir del más al menos detallista.

En la columna "Sección/Capítulo" se referencian la/s sección/es en donde se completó el requerimiento citado. En ellas se presenta el detalle del módulo o sección del trabajo y se referencia al requerimiento correspondiente, justificando la finalización del mismo.

Nivel	Requerimiento	Cumple	Sección/Capítulo	Comentarios
<b>L0.1</b>	<b>Versatilidad</b>	✓	6.4, 9.6	
<b>L1.1</b>	<b>Formato estandarizado de datos</b>	✓	6.4.2	
<i>L2.1</i>	<i>Paquete YAML</i>	✓	6.4.1	
<b>L1.2</b>	<b>Extensibilidad</b>	✓	7.2.5.3	
<b>L0.2</b>	<b>Transmisión remota</b>	✓	9.2	
<b>L1.1</b>	<b>Transmisión por red celular</b>	✓	9.4	
<i>L2.1</i>	<i>3G</i>	✓	9.4, 7.2.1	
<b>L1.2</b>	<b>Protocolo ligero de IoT</b>	✓	7.2.3	
<i>L2.1</i>	<i>MQTT</i>	✓	9.2	
<b>L1.3</b>	<b>Datos en la nube</b>	✓	9.5	
<i>L2.1</i>	<i>Acceso para suscriptores</i>	✓	9.5, 9.2	
<b>L0.3</b>	<b>Sistema de alertas</b>	✓	9.7	
<b>L1.1</b>	<b>Alertas por SMS</b>	✗	9.7	La alternativa son alertas por email o Telegram
<i>L2.1</i>	<i>Detección de outliers para envío</i>	✓	9.7	A criterio del usuario. Se establecen en la aplicación de Grafana
<b>L0.4</b>	<b>Tiempo real</b>	✓	9.8	
<b>L1.1</b>	<b>Alta disponibilidad</b>	✓	7.4	
<i>L2.1</i>	<i>Disponibilidad &gt; 99,9%</i>	✓	7.4	
<b>L1.2</b>	<b>Alta tasa de envío</b>	✓	9.8	
<i>L2.1</i>	<i>Tasa &gt; 40kbps</i>	✓	9.4	
<b>L1.3</b>	<b>Tamaño adecuado del paquete</b>	✓	6.4.3, 9.2	
<i>L2.1</i>	<i>Paquete &lt; 20 MB</i>	✓	6.4.3, 9.2	Depende de los datos sensados
<b>L0.5</b>	<b>Accesibilidad de los datos</b>	✓	7.3	
<b>L1.1</b>	<b>Visualización</b>	✓	9.7, 7.4	
<i>L2.1</i>	<i>Datos en gráfico de tiempo</i>	✓	7.4	
<b>L1.2</b>	<b>Posibilidad de descarga</b>	✓	7.4	
<b>L1.3</b>	<b>Almacenamiento</b>	✓	7.3	
<i>L2.1</i>	<i>GB ilimitados de almacenamiento</i>	✓	5.3	Depende del servicio de Base de Datos contratado
<i>L2.2</i>	<i>Servidor PostgreSQL</i>	✓	7.3.1	
<b>RT.1</b>	<b>Diseño modular</b>	✓	7	Cada una de las secciones representa un módulo del diseño
<b>RT.2</b>	<b>Autonomía (hs)</b>	✓	6.7	
<b>RT.3</b>	<b>Tamaño de paquetes (KB)</b>	✓	6.4.3	
<b>RT.4</b>	<b>Velocidad de transmisión (kbps)</b>	✓	9.8	
<b>RT.5</b>	<b>Costo (USD)</b>	✓	5	
<b>RT.6</b>	<b>Almacenamiento (GB)</b>	✓	7.3.1 , 5.3	
<b>RT.7</b>	<b>Consumo (mAh)</b>	✓	6.7	
<b>RT.8</b>	<b>Paquetes en formato estandarizado</b>	✓	6.4.1	

Tabla 10.1: Matriz de verificación de requerimientos

## **10.3 Aporte del trabajo realizado**

A lo largo del presente informe, se han destacado factores importantes que demuestran la contribución del trabajo realizado. Por un lado, desde el punto de vista competitivo, en donde, a diferencia de otros productos existentes, esta solución logra ser lo suficientemente genérica para poder adaptarse a sensores de distintos tipos. Ejemplos de esto son los Precursores Sísmicos con los que originalmente se planeó este proyecto y para los cuales se realizaron las pruebas, con los paquetes de ejemplo proveídos. Otro ejemplo es el del Instituto Nacional del Agua, para los cuales se planean mediciones de menor frecuencia que las del precursor, pero que aún así, el producto es compatible. Esto se logra en parte gracias al formato de paquete elegido y también al protocolo de IoT MQTT que permite el envío de estos datos con una velocidad de transmisión acorde a las necesidades del cliente.

Todos estos aportes contribuyen a la industria tanto del software, de forma de introducir un producto nuevo e innovador al mercado, como también a los estudios de ciencia y tecnología a lo largo del país, tanto en instituciones públicas y privadas, como las ya mencionadas. En la carrera de querer seguir mejorando este rubro, muchas de éstas buscan una solución tecnológica que les permita estar conectados de forma instantánea con los datos que les son relevantes, de forma de poder tomar acción de inmediato. Este proyecto forma parte de esa solución. Como consecuencia de esto, habrá una mejora en el proceso productivo, reduciendo tiempos de acceso a los datos, costos indirectos como viajes, equipamiento y hardware para la visualización de los mismos. Esto se traduce en ahorros de capital, tanto para lo que sería una empresa privada como para un instituto público, y también tiene un aporte social, al formar parte del proceso que permite monitorear variables que afectan a barrios, ciudades y grupos de personas de bajos recursos (ver sección 2.4.2).

Por último, este proyecto contribuye al ámbito académico, brindando conocimiento sobre la aplicación de distintas tecnologías disponibles en la industria, aplicando los conocimientos adquiridos en la carrera de Ingeniería Electrónica, utilizando recursos de análisis, cálculo y mediciones aprendidas a lo largo de la misma. Desde este punto de vista, se buscó a lo largo de toda la realización del proyecto la aplicación del pensamiento racional, actuando profesionalmente como ingenieros, buscando superar cada dificultad presentada y utilizando el mismo como herramienta principal para la resolución de los problemas encontrados. El ejemplo más claro de superación es el contexto en el que se realizó el proyecto. Como se mencionó en diversas secciones del informe, ante la pandemia del COVID-19, distintas pruebas, mediciones, e incluso el proyecto paralelo con el que se buscaba trabajar en conjunto se vieron afectadas por la misma, lo que llevó a los autores a buscar alternativas con el fin de cumplir con los objetivos planteados.

## **10.4 Lecciones aprendidas y recomendaciones**

Es importante destacar varios puntos en esta sección. En primer lugar tener en cuenta que este comprende el trabajo final de una carrera universitaria. Esto implica tener la oportunidad de aplicar los conocimientos adquiridos durante la misma hacia un proyecto del interés del alumno y que pueda dar un aporte a algún sector de la sociedad, ya sea dentro o fuera del ámbito universitario, e incluso al mismo alumno, dado que el proyecto

podría potencialmente ser comercializado. En pocas palabras, la búsqueda de un objetivo es importante a la hora de planear un proyecto y aplicar conceptos que uno considera importantes.

Como segundo punto, la idea de realizar un trabajo profesional implica llevarlo a cabo como el título mismo lo dice, de manera *profesional*. Esto quiere decir que el alumno debe tener en cuenta diversos aspectos para realizarlo. Como se mencionó, empezar por un objetivo, guiado por las necesidades del cliente y las capacidades de quien lo realiza; estudiar todos los aspectos técnicos y económicos necesarios para su realización; negociar con los clientes, de manera de entender sus necesidades, abriendo el diálogo, y manteniéndolo al tanto de los avances; evaluar la competencia, de manera de asegurarse de brindar una idea fresca que entregue un valor agregado al mercado y que no sea algo que el cliente pueda ir a buscar fácilmente por un competidor; mantenerse actualizado de los productos que existen en la industria y que puedan ayudar al desarrollo del proyecto.

Este último ítem es importante para evitar aumentar tiempos de desarrollo de ciertas partes del trabajo que podrían resolverse con un producto ya existente, evitando lo que vulgarmente se conoce como "reinventar la rueda". Como ejemplo de aprendizaje propio de este proyecto, se puede mencionar el de la herramienta de visualización utilizada. La idea original de ésto fue comenzar con el desarrollo completo de una interfaz de usuario con la capacidad de comunicarse con la base de datos y visualizar los datos en forma de gráficos, además de tener la opción de poder descargar los mismos y generar alertas. A las pocas semanas del comienzo del desarrollo, cuando aún no se había completado ni un 20 % del mismo, los autores encontraron la opción de Grafana y de muchas otras herramientas de monitoreo que ofrecían las mismas funciones (y más) que se buscaban para esta parte del trabajo. Es aquí cuando se decidió finalizar el desarrollo y optar por la herramienta ya existente, siempre haciendo las evaluaciones técnicas y económicas correspondientes, lo que redujo mucho tiempo y esfuerzo de los autores.

Por último, los autores destacan nuevamente el contexto bajo el que se realizó el trabajo, que si bien consistió en un hecho inédito a nivel mundial, sirve como ejemplo de las contingencias a las que uno se puede enfrentar a la hora de realizar un proyecto de ingeniería. Como bien se aprendió a lo largo de la carrera, el mundo ingenieril consiste en parte a la resolución de problemas y sistemas complejos, tratando de encontrar la alternativa más efectiva para poder llevar adelante una solución. Este caso no fue la excepción y la solución se llevó adelante. Así como en este caso, en la carrera de cada uno los problemas serán habituales. La mejor recomendación que los autores pueden brindar es la utilización del pensamiento racional para analizar la causa y las posibles soluciones a esas contingencias, utilizar los recursos disponibles y con los que no se cuentan, salir a buscarlos, y cada problema podrá ser resuelto.

## 10.5 Líneas futuras

En base a lo trabajado, este proyecto tiene un alcance del cual se puede sacar provecho en su implementación, ya sea para la aplicación en precursores sísmicos como también para otro tipo de proyectos de distinta índole. El alcance es equivalente para cada uno de ellos, sin ultimar detalles.

Existen de todas formas, líneas de trabajo futuras para aquel que quiera continuar con el presente trabajo y que harán del mismo, un diseño más robusto en su implementación. Las mismas no formaron parte de los objetivos del mismo, sin embargo, se analizaron en distintas secciones del informe para su posterior análisis. A continuación se presentan:

### *Consumo*

En la sección 6.7 se detalló un análisis del consumo del módulo intercomunicador. Como bien se detalla en la sección 14.2 del Anexo, el dispositivo que toma muestras debe hacerlo en lugares remotos con acceso limitado a la red eléctrica. Es por eso que será importante el uso de baterías y paneles solares (detalles en sec. 6.7) para evitar la intervención humana. La implementación del uso de las mismas resultará importante para completar el diseño completo de los sensores con el sistema presentado en este trabajo. En el caso de otros proyectos, con mayor acceso a la red, el uso de baterías puede no ser necesario.

Por otro lado, para reducir el consumo existe la alternativa de cambiar el hardware utilizado para el módulo intercomunicador. En lugar de usar la Raspberry Pi, la opción más viable es el ESP-32, como se mencionó en la sección 6.2.

### *Alertas*

En la sección 9.7 se muestra la disponibilidad de alertas sobre la plataforma de visualización elegida. Las mismas se pueden establecer mediante un criterio elegido por el usuario y mediante un canal de comunicación establecido, como por ejemplo por correo electrónico.

Para la implementación con el sistema capturador de datos presentado en el paper de la sección 14.2 del Anexo, será necesario el establecimiento oficial de estas alertas, tomando criterios en base a las mediciones y estableciendo alertas por los canales necesarios.

### *Seguridad*

En la tabla 5.1 se presentan los costos para la implementación de brokers con SSL/TLS (**Cap. 13.8**), que permiten una autenticación segura para así asegurar la privacidad de los datos. Existen librerías que permiten la interacción de los clientes MQTT con SSL/TLS, lo que hace muy simple su implementación, y varía mucho del código implementado en este proyecto. Para el uso de las mismas es necesario la adquisición de los certificados correspondientes.

# Capítulo 11

## Agradecimientos

La finalización de este proyecto tuvo muchas influencias y gente que colaboró para que el mismo pueda ser llevado a cabo. A continuación se hace mención de aquellos.

- Al decano de FIUBA, **Ing. Alejandro M. Martínez**
- Al director del departamento de Electrónica, **Ing. Carlos Fernando Belaustegui Goitia**.
- Al tutor del Trabajo Profesional, **Ing. Pablo Carlos Marino Belcaguy**.
- Al co-tutor del proyecto, **Ing. Ramiro Alonso**.
- A los colaboradores del proyecto ”Sistema de adquisición de datos para el estudio de precursores sísmicos”, **Ing. Leonardo Carducci**, **Ing. Ramiro Alonso** y **Ing. Walter G. Fano**.
- A los miembros del **LABi (Laboratorio Abierto)**
- A **Pablo García**, miembro del INA (Instituto Nacional del Agua) y sus colaboradores.
- A **Francisco Odeón**, miembro de Ingeniería sin Fronteras y sus colaboradores.
- A los docentes de FIUBA.
- A **Maximiliano Torre**, estudiante de FIUBA.

# Capítulo 12

## Definiciones

### 12.1 Capítulo 1

- *Hosting*: el alojamiento web (en inglés: web hosting) es el servicio que provee a los usuarios de Internet un sistema para poder almacenar información, imágenes, vídeo, o cualquier contenido accesible vía web.
- *Clase*: una clase es una construcción que permite crear tipos personalizados propios mediante la agrupación de variables de otros tipos, métodos y eventos. Una clase es como un plano. Define los datos y el comportamiento de un tipo. Si la clase no se declara como estática, el código de cliente puede utilizarla mediante la creación de objetos o instancias que se asignan a una variable. La variable permanece en memoria hasta todas las referencias a ella están fuera del ámbito. En ese momento, CLR la marca como apta para la recolección de elementos no utilizados. Si la clase se declara como estática, solo existe una copia en memoria y el código de cliente solo puede tener acceso a ella a través de la propia clase y no de una variable de instancia.<sup>66</sup>
- *Objeto*: una definición de clase o estructura es como un plano que especifica qué puede hacer el tipo. Un objeto es básicamente un bloque de memoria que se ha asignado y configurado en función del plano. Un programa puede crear muchos objetos de la misma clase. Los objetos también se denominan instancias y se pueden almacenar en una variable con nombre o en una matriz o colección. El código cliente es el código que utiliza estas variables para llamar a los métodos y obtener acceso a las propiedades públicas del objeto. En un lenguaje orientado a objetos como C#, un programa típico se compone de varios objetos que interactúan dinámicamente.<sup>67</sup>
- *Interfaz*: una interfaz contiene las definiciones de un grupo de funcionalidades relacionadas que una clase o una estructura no abstracta deben implementar. Una interfaz puede definir métodos static, que deben tener una implementación. A partir de C# 8.0, una interfaz puede definir una implementación predeterminada de miembros. Una interfaz no puede declarar datos de instancia, como campos, propiedades implementadas automáticamente o eventos similares a las propiedades.

Mediante las interfaces puede incluir, por ejemplo, un comportamiento de varios orígenes en una clase. Esta capacidad es importante en C# porque el lenguaje no

admite la herencia múltiple de clases. Además, debe usar una interfaz si desea simular la herencia de estructuras, porque no pueden heredar de otra estructura o clase.<sup>68</sup>

- *Clase Abstracta*: el propósito de una clase abstracta es proporcionar una definición común de una clase base que múltiples clases derivadas pueden compartir. Por ejemplo, una biblioteca de clases puede definir una clase abstracta que se utiliza como parámetro para muchas de sus funciones y solicitar a los programadores que utilizan esa biblioteca que proporcionen su propia implementación de la clase mediante la creación de una clase derivada.<sup>69</sup>

## 12.2 Capítulo 2

- *Stakeholder*: cualquier individuo o grupo que puede afectar o verse afectado por las actividades de la organización. Este concepto conlleva una visión muy amplia de los grupos de interés de la empresa asumiendo a los competidores y la sociedad donde la entidad desarrolla sus actividades como parte integrante de los mismos, asumiendo que las compañías “se insertan” en una comunidad su correcta gestión puede beneficiar la vida de la misma.<sup>70</sup>
- *Microservicios*: La arquitectura de microservicios (en inglés, Micro Services Architecture, MSA) es una aproximación para el desarrollo de software que consiste en construir una aplicación como un conjunto de pequeños servicios, los cuales se ejecutan en su propio proceso y se comunican con mecanismos ligeros (normalmente una API de recursos HTTP). Cada servicio se encarga de implementar una funcionalidad completa del negocio. Cada servicio es desplegado de forma independiente y puede estar programado en distintos lenguajes y usar diferentes tecnologías de almacenamiento de datos.<sup>71</sup>

## 12.3 Capítulo 3

- *Versatilidad*: capacidad de algo o alguien de adaptarse con rapidez y facilidad a distintas funciones.
- *Plug & Play*: es la tecnología o avance que permite a un dispositivo informático ser conectado a una computadora sin tener que configurar, mediante jumpers o software específico (no controladores) proporcionado por el fabricante, ni proporcionar parámetros a sus controladores. Para que sea posible, el sistema operativo con el que funciona el ordenador debe tener soporte para dicho dispositivo.
- *Disponibilidad*: la cantidad del tiempo en que el servicio está disponible para ser utilizado. Se puede expresar de muchas maneras, ya sea como un porcentaje de tiempo disponible, o bien garantizar la disponibilidad del servicio en una franja horaria determinada.

## 12.4 Capítulo 5

- *Market share*: en el área de dirección estratégica y mercadotecnia, cuota de mercado es la fracción o porcentaje que se tendrá del total de mercado disponible o del segmento del mercado que está siendo suministrado por la compañía.<sup>18</sup>

## 12.5 Capítulo 4

- *Work Breakdown Structure*: una estructura de desglose del trabajo (EDT), también conocida por su nombre en inglés Work Breakdown Structure o WBS, es una herramienta fundamental que consiste en la descomposición jerárquica, orientada al entregable, del trabajo a ser ejecutado por el equipo de proyecto, para cumplir con los objetivos de éste y crear los entregables requeridos, donde cada nivel descendente de la EDT representa una definición con un detalle incrementado del trabajo del proyecto.<sup>19</sup>
- *Diagrama de Gantt*: es una herramienta gráfica cuyo objetivo es exponer el tiempo de dedicación previsto para diferentes tareas o actividades a lo largo de un tiempo total determinado. A pesar de esto, el diagrama de Gantt no indica las relaciones existentes entre actividades.<sup>20,21</sup>

## 12.6 Capítulo 7

- *Cloud*: es un modelo que permite, convenientemente, el acceso bajo demanda a redes ubicuas para compartir un conjunto configurable de recursos de computación (por ejemplo, redes, servidores, almacenamiento, aplicaciones y servicios) que se pueden proveer y liberar rápidamente con un mínimo esfuerzo de administración o interacción del proveedor del servicio.<sup>22</sup>
- *Publisher - Subscriber*: Un cliente MQTT es cualquier dispositivo (desde un microcontrolador hasta un servidor) que corre una librería MQTT y se conecta a un MQTT broker sobre una red.<sup>23</sup>
- *Broker*: Un broker MQTT es un servidor que recibe todos los mensajes de los clientes y luego los rutea a los clientes de destino apropiados.<sup>24</sup>

# Capítulo 13

## Acrónimos

### 13.1 Capítulo 1

- **CALSIM**: Calibrador Simulador
- **SOA**: Arquitectura Orientada a Servicios
- **SRP**: Principio de responsabilidad única
- **OCP**: Principio de abierto/cerrado
- **LSP**: Principio de sustitución de Liskov
- **ISP**: Principio de segregación de la interfaz
- **DIP**: Principio de inversión de la dependencia

### 13.2 Capítulo 2

- **RP**: Raspberry Pi
- **TLF**: Frecuencia tremadamente baja
- **ELF**: Frecuencia extremadamente baja

### 13.3 Capítulo 3

- **GSM**: Sistema global para las comunicaciones móviles
- **3G**: Tercera Generación
- **MQTT**: MQ *Telemetry Transport* (Transporte de telemetría)
- **L0**: Nivel 0
- **L1**: Nivel 1
- **L2**: Nivel 2

- **YAML**: *YAML Ain't Markup Language* - YAML no es un lenguaje de marcado
- **RT**: Requerimiento Técnico

## 13.4 Capítulo 5

- **LPSC**: Laboratorio de Procesamiento de Señales en Comunicaciones
- **LRAD**: Laboratorio de Radiación Electromagnética
- **LSE**: Laboratorio de Sistemas Embebidos
- **IEEE**: Instituto de Ingeniería Eléctrica y Electrónica
- **INA**: Instituto Nacional del Agua
- **SIM**: *Subscriber Identity Module* - Módulo de identificación de abonado
- **CMOS**: Semiconductor complementario de óxido metálico
- **LIDAR**: *Light Detection and Ranging or Laser Imaging Detection and Ranging* - Detección y Rango de Luz o Imagen, Detección y Rango por Laser

## 13.5 Capítulo 4

- **WBS**: *Work Breakdown Structure* - Estructura de desglose del trabajo.

## 13.6 Capítulo 7

- **UART**: *Universal Asynchronous Reception and Transmission* - Transmisión-Recepción Asíncrona Universal
- **API**: *Application Programming Interface* - Interfaz de programación de aplicaciones

## 13.7 Capítulo 8

- **ICD**: *Interface Control Document* - Documento de Control de Interfaz

## 13.8 Capítulo 10

- **TLS**: *Transport Security Layer* - Seguridad de la Capa de Transporte
- **SSL**: *Secure Sockets Layer* - Capa de Puertos Seguros

# Capítulo 14

## Anexos

### 14.1 Presentación del Plan del TP Profesional



UNIVERSIDAD DE BUENOS AIRES

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE ELECTRÓNICA

---

#### Plan de trabajo profesional de Ingeniería Electrónica

---

*Alumnos:*

Belgrano, Mateo - 96208  
Luraschi, Sebastian - 97177

*Tutores:*

Marino, Pablo (Tutor)  
Alonso, Ramiro (Co-tutor)

1 de noviembre de 2020

# Índice

1. Descripción del proyecto	2
2. Antecedentes	3
3. Áreas profesionales de relevancia	3
4. Objetivos	3
5. Definiciones de necesidades y soluciones	4
5.1. Necesidad detectada . . . . .	4
6. Alcance del proyecto	4
7. Entregables	5

# 1. Descripción del proyecto

**Título:** Sistema de telemetría IoT aplicado al estudio de precursores sísmicos

El proyecto consiste en la comunicación de datos generados por un sistema de sensores y un servidor remoto. Los mismos permiten ver en tiempo cuasi real a la distancia, desde una computadora conectada a internet, la evolución de determinados parámetros. En este caso particular la aplicación es precursores sísmicos pero el desarrollo se realizará de forma que sea fácilmente adaptable a otras aplicaciones, o sensores.

La **motivación** de este grupo por la realización de este trabajo se basa en la amplitud de áreas que podrían ser abarcadas por el mismo, como así también la relación que guarda con el campo laboral desempeñado por los miembros del equipo, siendo que nuestro trabajo de hoy en día está ligado a la industria petrolera. Éste proyecto es por tanto, un impulso que sirve para negociar con posibles stakeholders involucrados en esta área, dado su fuerte relación con el estudio de suelos. Como se mencionó, se hará especial foco en mantener una estructura de microservicios reutilizables, de forma de que solo se necesiten cambios menores a ciertos bloques funcionales para que la solución se pueda reaplicar a un campo totalmente nuevo.

El sistema completo se compone de esta forma:

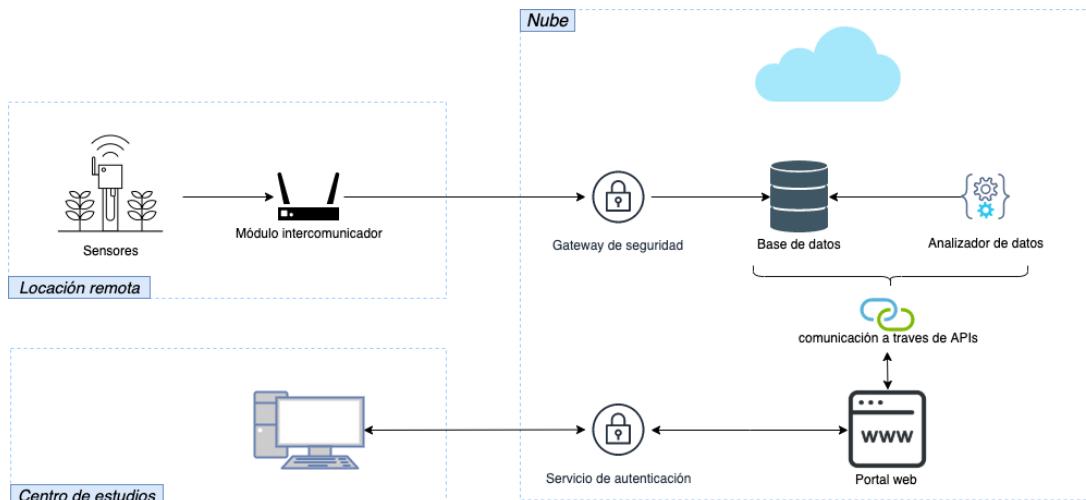


Figura 1: Modelo del sistema

Desde un módulo intercomunicador localizado en el mismo espacio físico que los sensores se envían los datos crudos tomados del campo directo a la nube, con la particularidad de que la red de sensores puede estar ubicada en lugares de poca cobertura. Además, estos datos serán procesados y publicados con un formato específico en un servidor también en la nube para luego ser consumidos en tiempo real por un usuario remoto a través de un servicio web implementado por los alumnos de este proyecto.

## 2. Antecedentes

Si bien hay diversos proyectos independientes similares, no se ha encontrado uno comercializado con el mismo foco sismológico que este proyecto presenta. Se encontraron empresas que con las mismas tecnologías brindan servicios de sensado pero, a diferencia de nuestro producto, estos funcionan en áreas con buena conectividad. Algunas de estas empresas son:

- **Pycno:** Al igual que nuestro producto, el suyo logra comunicar los datos sensados en áreas sin WI-FI utilizando una tarjeta SIM embebida en sus sensores. La diferencia es que se dedican al sensado, análisis y control con el foco puesto en la mejora de la producción agrícola. Si bien toman datos del suelo estos se relacionan con la humedad, PH y fertilización. Sus sensores hechos especialmente para la aplicación los comercializan por U\$S 350 (cada uno) y la base de control centralizado por U\$S 150, además de un posible costo mensual por la tarjeta SIM.[1]
- **SlantRange:** También se dedican al análisis de los suelos pero desde el cielo. En este caso comunican y almacenan los datos sensados en una SD y luego los traspasan de forma manual para su estudio. La plataforma de análisis se renueva anualmente y el monto puede subir hasta U\$S 3000 por año, según que se contrate. Adicionalmente venden sensores especiales para su aplicación por U\$S 5000.[2]

## 3. Áreas profesionales de relevancia

- Comunicaciones digitales
- Procesamiento de señales
- Programación Web
- Manejo de bases de datos y grandes volúmenes de información
- Sistemas embebidos
- IoT y servicios Cloud
- Sensores

## 4. Objetivos

El principal objetivo de este proyecto es lograr el envío exitoso de los datos desde el sensor instalado en el campo hasta una estación de trabajo remota. Luego, el correcto procesamiento de los mismos, teniendo en cuenta que corresponden a distintas variables, con distintas magnitudes y frecuencias.

Como objetivo particular, se busca que con la publicación de los datos procesados en la nube se pueda realizar una prueba de concepto de la comunicación remota que sirva como disparador para la industria dedicada a la investigación de movimientos generados en la tierra (petroleras, mineras, laboratorios de investigación).

## 5. Definiciones de necesidades y soluciones

### 5.1. Necesidad detectada

Existen distintos proyectos de monitoreo aplicados a varios campos de la industria y la investigación. Muchos de estos, como el caso analizado, utilizan sensores que deben ser instalados en lugares remotos que pueden llegar a ser inaccesibles. Esto genera una dificultad a la hora del acceso y el procesamiento de los datos, dado que para levantarlos en un dispositivo, uno estaría obligado a permanecer cerca del sensor para permitir la comunicación entre los mismos (sensor y computadora por ejemplo). Esto significa un problema para el caso de grandes investigaciones que requieren del trabajo de científicos o profesionales en áreas lejanas al sensado, como puede ser un laboratorio ubicado en otra ciudad o provincia.

Un ejemplo claro de estos casos son las empresas petroleras, las cuales deben tomar datos sísmicos y de movimientos de suelos con el fin de realizar los correspondientes análisis exploratorios para la extracción del petróleo en lugares de poca cobertura. Adicionalmente, en el campo de la investigación aun se busca una forma de detectar sismos antes de que ocurran. Distintas corrientes han surgido en base a esto, como el análisis de temperatura o humedad generado por movimientos, otra de estas corrientes basada en el análisis de pequeños movimientos también esta contemplada por la prueba de concepto desarrollada en este trabajo.

## 6. Alcance del proyecto

A continuación se presenta el alcance del proyecto con la intención de presentar objetivos **SMART** (Específicos -S-, Medibles -M-, Alcanzables -A-, Relevantes -R-, Tiempo Orientados -T-)

1. *Utilizar un módulo intercomunicador para capturar datos de baja y alta frecuencia enviados desde la red de sensores, en paquetes con un formato determinado por el grupo y el equipo de investigación que toma los datos. El envío se realiza por el protocolo adecuado para el tamaño de los paquetes.*

**Tiempo estimado:** 1,5 meses.

Para este objetivo, las tareas más importantes consisten en el armado de los paquetes de datos con el formato adecuado, implementando los scripts de captura correspondientes que lean los mismos en forma periódica. Establecidas estas pautas, la integridad de los datos será verificada almacenándolos en archivos dentro de un repositorio, de forma de tener un registro de lo que se envió.

2. *Desarrollar un sistema de software automático que envíe los datos periódicamente mediante un protocolo de envío ligero desde el módulo intercomunicador hacia la nube. Desarrollar un sistema análogo que reciba los datos desde otra ubicación con el formato que fueron enviados y bajo la misma tasa.*

**Tiempo estimado:** 2 meses.

Las claves de este punto están en el montaje de los tres agentes principales para la comunicación: el **publicador**, el **servicio** y el **suscriptor**. El primero será el

que envía los datos, es decir, el mismo intercomunicador que los obtiene de la red de sensores. El servicio será montado en la nube utilizando un host gratuito [3] y funcionará como intermediario entre los clientes. Quien suscribe será cualquier dispositivo conectado a la red desde cualquier lugar remoto. La calidad de este objetivo se verificará mediante la comprobación de que los datos publicados por el módulo intercomunicador se mantengan íntegros a la hora de leerlos desde el suscriptor.

3. *Procesar los datos utilizando los filtros correspondientes similar para obtener una visualización en tiempo y frecuencia del comportamiento de las señales obtenidas. Las imágenes serán guardadas en un repositorio compartido por los miembros del grupo, sus tutores y colaboradores.*

**Tiempo estimado:** 2 meses.

Utilizando una PC con los recursos suficientes, se procederá a realizar una biblioteca de filtros necesarios para el procesamiento de los datos recibidos. El objetivo la obtención de imágenes en tiempo y frecuencia para el análisis posterior por parte del usuario final.

4. *Publicar una página web utilizando un hosting alojado en la nube de forma de publicar los datos procesados previamente y que brinde la opción de descargar y visualizar los datos correspondientes a un período de tiempo. El servidor puede ser propio (LABI) o un hosting gratuito.*

**Tiempo estimado:** 3 meses.

Las tareas principales para la realización de este objetivo consistirán en la obtención de un hosting para el servicio web, como así también el correcto envío de los datos en forma periódica al mismo. Una vez que los mismos estén subidos, se debe establecer un mecanismo de autenticación para quien desee visualizarlos/descargarlos, y verificar su integridad.

5. *Stretch: mediante algoritmos de detección de anomalías y detección de outliers, generar alertas que se disparen periódicamente desde el servidor hacia los usuarios mediante email o SMS al teléfono móvil.*

**Tiempo estimado:** 1.5 meses

Se define a este objetivo como *stretch*, dado que se lo piensa como un objetivo ampliado, que se busca lograr siempre y cuando se logren los anteriores. Para lograrlo, se deben elegir al menos dos algoritmos que sirvan para la detección de anomalías o outliers y la configuración de un endpoint para el envío de alertas a los dispositivos que se suscriban a las mismas.

## 7. Entregables

A continuación se indican **orientativamente** los distintos entregables en una estructura WBS (*Work Breakdown Structure*) indicando a su vez el encargado de la tarea con los marcadores MB - Mateo Belgrano; SL - Sebastian Luraschi:

1. Comunicación sensores - módulo intercomunicador

- a) Muestreo de baja frecuencia
  - 1) Establecimiento del formato del paquete - MB y SL
  - 2) Establecimiento de velocidad de transmisión - MB y SL
  - 3) Prueba integral (con sensores) de envío sincrónico - MB
- b) Muestreo de alta frecuencia
  - 1) Establecimiento del formato del paquete - MB y SL
  - 2) Establecimiento de velocidad de transmisión - MB y SL
  - 3) Prueba integral (con sensores) de envío sincrónico - MB y SL

## 2. Comunicación intercomunicador - nube

- a) Infraestructura y pruebas
  - 1) Adquisición de módulos Wi-Fi y 3G - SL
  - 2) Publicación de datos por Wi-Fi - SL
  - 3) Publicación de datos por 3G - MB
  - 4) Implementar medidas de seguridad y privacidad - SL
- b) Arquitectura [4, 5]
  - 1) Crear broker MQTT en la nube - MB
  - 2) Desarrollo de scripts de suscripción - MB
  - 3) Desarrollo de scripts de publicación - MB
  - 4) Generación de tareas programadas para la publicación - SL

## 3. Procesamiento de datos

- a) Acondicionamiento de señales
  - 1) Creación de biblioteca de filtros - SL
  - 2) Desarrollo de espectrogramas - MB
  - 3) Análisis gráfico de lo sensado - MB
  - 4) Análisis de resonancia Shumann - SL
  - 5) Pruebas con diversas señales - SL
- b) Almacenamiento de resultados
  - 1) Análisis de almacenamiento en la nube - SL
  - 2) Análisis de almacenamiento local - SL
  - 3) Análisis de persistencia de datos - MB
  - 4) Armado del storage - SL

## 4. Publicación Web

- a) Hosting
  - 1) Estudio de posibles hostings - MB
  - 2) Desarrollo de UI - SL y MB
  - 3) Desarrollo de APIs de comunicación - SL y MB

## 5. Alertas y predicciones

- a) Alertas automáticas
  - 1) Análisis de datos obtenidos - SL
  - 2) Estudio de los canales de alerta - MB
  - 3) Desarrollo de algoritmos de detección y estimación - MB

## Referencias

- [1] <https://es.pycno.co/>
- [2] <https://slantrange.com/>
- [3] <https://www.heroku.com/>
- [4] <http://mqtt.org/>
- [5] <https://www.ibm.com/developerworks/ssa/library/iot-mqtt-why-good-for-iot/index.html>

## 14.2 Sistema de adquisición de datos para el estudio Precursores Sísmicos - Paper

# Data acquisition system for the study of earthquakes precursors by measuring magnetic field emissions

Leonardo M. Carducci  
Facultad de Ingeniería  
Universidad de Buenos Aires  
Buenos Aires, Paseo Colón 850  
Email: lcarducci@fi.uba.ar

Ramiro Alonso  
Facultad de Ingeniería  
Universidad de Buenos Aires  
IEEE Member  
Email: ralonso@fi.uba.ar

Walter G. Fano  
Facultad de Ingeniería  
Universidad de Buenos Aires  
Senior Member IEEE  
Email: gustavo.fano@ieee.org

**Abstract**—There are several evidence of magnetic field alterations that have been registered instants before seismic activity, in different parts of the world. Under the hypothesis that there is a relation between both phenomena, a multiple disciplinary project is being developed for verifying this statement. In this paper, a prototype for a data acquisition system is presented for obtaining valuable information in order to find this evidence. It's main objective is to sense and register electromagnetic variations as well as other relevant variables found in high seismic activity areas. This system is completed with several communication technologies, for remote monitoring of the information.

**Resumen**—En diversas partes del planeta se han registrado múltiples evidencias de alteraciones electromagnéticas momentos previos a una actividad sísmica. Bajo la hipótesis de que existe una relación entre ambos fenómenos, y en el marco de un proyecto interdisciplinario, en este trabajo se presenta el prototipo para un sistema de adquisición de datos con el que se podrá obtener valiosa información para este tipo de investigaciones. Su objetivo principal es sensar tanto las variaciones electromagnéticas como otras variables relevantes en zonas con una potencial actividad sísmica. Además, el sistema se complementa con distintas tecnologías de comunicación para el monitoreo remoto del mismo.

**Keywords**—seismic precursors; data acquisition; magnetic field; extremely low frequencies; sensors.

### I. INTRODUCCIÓN

Los terremotos son uno de los fenómenos naturales capaces de producir gran pérdida de vidas y daños a la civilización. Es por ello que para obtener un pronóstico anticipado a estos cataclismos la humanidad ha ingeniado distintos instrumentos capaces detectar la actividad sísmica de manera temprana. En este sentido, el primer sismógrafo registrado se remonta al siglo II d.C., creado por el filósofo chino Chang Heng [1]. Actualmente los sismógrafos utilizados proporcionan un alto grado de precisión para detectar los movimientos de la corteza terrestre, empleando para ello sensores mecánicos, electromagnéticos, mediante acelerómetros, etc. [2].

Por otra parte, desde hace tiempo que los investigadores están realizando mediciones de los precursores electromagnéticos con la finalidad de vincularlos a los terremotos. Karakelian trabajó con mediciones de campo magnético en las frecuencias de ULF (*Ultra Low Frequency*) [3], porque a frecuencias mayores la profundidad de penetración disminuye y hace difícil las mediciones ya que hay demasiada atenuación. Para

hacer dichas mediciones es conveniente aislar el instrumento de interferencias ULF de tipo: ionosférico, geomagnético, variaciones de temperatura y otras variables ambientales, resonancia de Schumann [4], etc.. También hay que discriminar los micropulsos de origen geomagnético que se pueden medir en diferentes latitudes cuyos períodos van desde 0,2 s hasta 10 min [5]. Lo que interesa es detectar las señales producidas por los movimientos sísmicos desde las placas tectónicas, que son señales del orden de 7 segundos de período [3]. Hata y Yabashi [6], en la ciudad de Ito, Shizuoka, y en Monte Fugen, Nagasaki, han detectado señales de campo magnético en la banda de frecuencias ELF (*Extremely Low Frequency*), de 225 Hz aproximadamente, antes y después de un terremoto. Otra experiencia fue la de Bleier [7], que colocó dos sensores de campo magnético en la banda de ULF separados por 20 km entre sí, detectando en Loma Prieta una frecuencia de 0.01 Hz, con una intensidad pico de campo magnético de 3 nT. Existen muchas otras evidencias de cambios ionosféricos debido a variaciones de campo magnético, posiblemente causadas por la trituración de rocas cristalinas [8] [9] durante actividades sísmicas intensas. Estas alteraciones se han manifestado como fenómenos electromagnéticos inexplicables, minutos, horas e incluso días previos a un terremoto. Estos efectos se han observado, por ejemplo, como estática producida en radios, encendido de tubos fluorescentes, alteraciones en la imagen de televisores de rayos catódicos, imanes cayéndose de las heladeras, entre otras [10] [11].

El objetivo de este trabajo es contribuir con un proyecto de investigación enfocado en determinar la presencia de estos fenómenos electromagnéticos, y su posible correlación con los movimientos sísmicos. En este contexto, el trabajo presenta el desarrollo de un sistema de adquisición cuyo propósito es medir las perturbaciones del campo magnético en el rango de 0.01 Hz a 200 Hz aproximadamente, y al mismo tiempo monitorear la actividad sísmica empleando sensores convencionales. Además, debido a que la onda electromagnética que se desea detectar se encuentra en la zona de campo cercano, es conveniente que la estación de medición siente las tres componentes del campo magnético [12] [13].

En la sección II de este trabajo se plantea un breve resumen de los sensores que componen al sistema de adquisición y las variables que se desean adquirir. En la sección III, se presenta

la arquitectura de *hardware* y sus distintos subsistemas. En la Sección IV se muestran los resultados obtenidos en distintas capturas. Finalmente se presentan las conclusiones en la sección V y algunas propuestas a futuro en la sección VI.

## II. MEDICIÓN DE MAGNITUDES RELEVANTES

Como se mencionó en la sección anterior, en este proyecto se desean estudiar las perturbaciones electromagnéticas que pudieran estar relacionadas con los movimientos sísmicos. Por esta razón, el instrumento principal consiste de 3 antenas de tipo cuadro, perpendiculares entre sí, diseñadas para captar las variaciones de campo magnético a muy bajas frecuencias e intensidades. Junto con esto se desea conocer el comportamiento de distintas variables que pudieran aportar información relevante sobre la actividad sísmica y otras condiciones ambientales. Entre ellas, el campo magnético terrestre estático, utilizando para ello un magnetómetro de protones. También se desea registrar la presencia de gas radón liberado desde el interior de las rocas tras su fractura. Además, se colocará un sismógrafo mecánico tradicional para detectar vibraciones mecánicas que pudieran estar relacionadas con posibles movimientos sísmicos. Por otra parte, se tomarán muestras de otras variables como la conductividad del aire, humedad y temperatura, para descartar errores que pudieran producirse debido a variaciones bruscas de las mismas.

### A. Sensor de perturbaciones electromagnéticas

Es necesario detectar perturbaciones del orden de los nanoteslas, y para ello se utilizarán 3 bobinas de cable, como la que se observa en la Fig. 1, de 78 cm lado en 3 ejes. Cada una de ellas estará conectada a un amplificador con una ganancia de hasta 100 dB en tensión y un filtro pasa bajos que garantice el rango de frecuencias de interés (0.01-200 Hz), el cual se utilizará para evitar el aliasing durante el muestreo del conversor analógico digital.

### B. Detector de gas radón

Cuando el radón decae se liberan partículas alfa con una energía de 5.5 MeV. Estas partículas se desplazan entre 4 y 7



Figura 1. Bobina de cuadro prototipo: 78 cm de lado y un arrollamiento de 220 vueltas de alambre esmaltado de 0,5 mm.

centímetros antes de disipar su energía ionizando moléculas de aire. Básicamente, el detector produce un campo eléctrico con una fuente de alta tensión, y muy bajo consumo de corriente, con electrodos ubicados en el interior de un recipiente cilíndrico. Este campo eléctrico captura los electrones producto de la ionización, para luego ser detectados en forma de pulsos eléctricos. De esta manera, se podrá calcular la concentración de radón contando los pulsos generados [14] [15].

### C. Magnetómetro de protones

Mediante un campo magnético aplicado se alinean las moléculas de un fluido, por ejemplo agua destilada, y se registra el tiempo que tardan en volver a su estado original. Este tiempo es proporcional al campo magnético presente en el ambiente [16]. Sin embargo, debido a que este instrumento aún está en proceso de desarrollo por el equipo de investigación, por el momento no se encuentran completamente definidas todas sus especificaciones.

### D. Sensor de vibraciones

Si bien existe una gran variedad de sensores específicos para esta medición disponibles en el mercado, a los fines prácticos de este trabajo se desarrolló un detector de vibraciones mecánicas usando un transductor piezoelectrónico [17]. Para ello, uno de los extremos del sensor se adhiere a una superficie sólida fijada al terreno. En el otro extremo, se fija una masa en suspensión. Ante las vibraciones del terreno se producen tensiones mecánicas en el piezoelectrónico debido a la inercia de la masa suspendida. Luego estas vibraciones se traducen a una señal eléctrica que será acondicionada por medio de un amplificador, filtros y un diodo zener para limitar la amplitud en la entrada del conversor analógico digital.

## III. SISTEMA DE ADQUISICIÓN

En la sección II se mencionó una breve descripción de las magnitudes a medir y los sensores requeridos en el marco de este proyecto. Por consiguiente, en la presente sección se desarrollan los aspectos relacionados a la plataforma de *hardware* propuesta para la adquisición de dichos datos.

El sistema completo, cuyo diagrama de bloques se indica en la Fig. 2, debe cumplir con un conjunto de requerimientos mínimos para un adecuado análisis de la información recolectada, permitiendo así la búsqueda de evidencias asociadas a los fenómenos descritos en la sección I. En términos generales, entre los objetivos del proyecto también se consideran todas aquellas mejoras y optimizaciones que pudieran aplicarse a futuros prototipos de este sistema. De esta manera, será posible mejorar su desempeño tanto en aspectos de *firmware* como de *hardware* en función de las observaciones y los resultados que se obtengan de ellas durante períodos de tiempo prolongados. Asimismo, este sistema se destaca por ser una plataforma de relativamente bajo costo, con tasas de muestreo apropiadas para cada una de las variables sensadas, pero no excesivamente sobredimensionadas en relación a los rangos de frecuencia de observación (0.01-200 Hz).

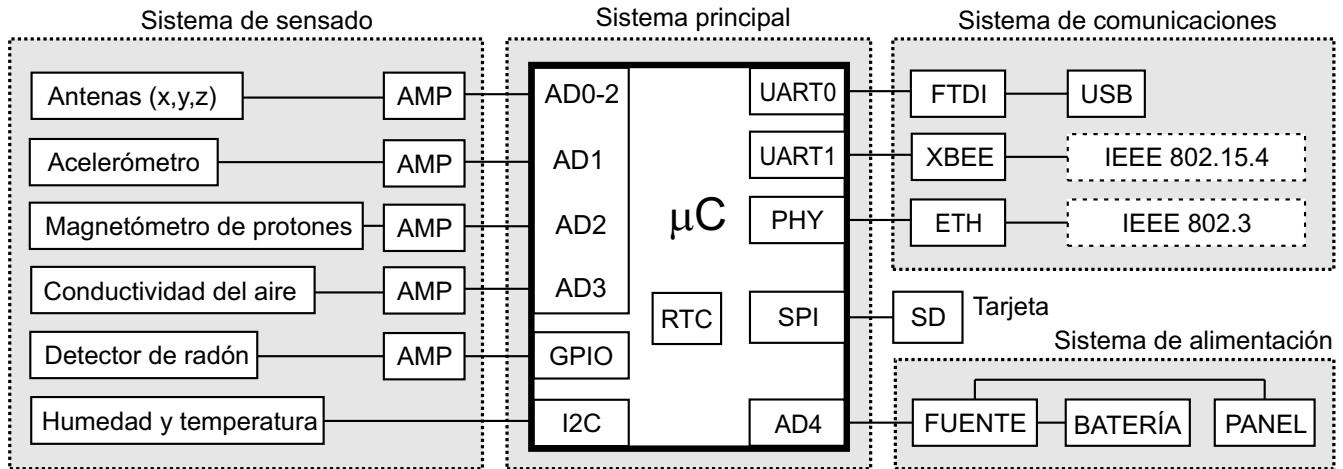


Figura 2. Diagrama en bloques del sistema de adquisición.

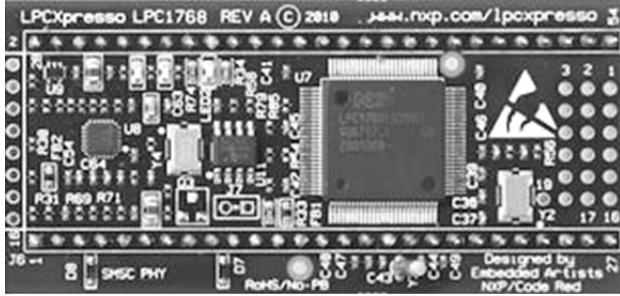


Figura 3. Placa de evaluación *LPC1769 LPCXpresso Board*: microcontrolador LPC1769, circuito LAN8729, cristales para el microcontrolador y RTC.

#### Cuadro I

REQUERIMIENTOS MÍNIMOS PARA LOS SENSORES DEL SISTEMA, CONSIDERANDO: LA CANTIDAD, EL TIPO, LA FRECUENCIA DE MUESTREO (Fs) Y LA RESOLUCIÓN (Res).

Sensores (x cantidad)	Tipo	Fs	Res
Antenas (x3)	Analógico	1 kHz	10-bit
Magnetómetro de protones (x1)	Analógico	100 Hz	8-bit
Acelerómetros (x1)	Analógico	100 Hz	10-bit
Detector de Radón (x1)	Digital	INT*	1-bit
Humedad y Temperatura (x1)	Digital	3.3 mHz	8-bit
Conductividad del aire (x1)	Analógico	3.3 mHz	8-bit

\*INT corresponde a sensado por interrupción.

### A. Sistema principal

El elemento central de este bloque es un microcontrolador de 32-bit, de arquitectura ARM cortex-M3, modelo LPC1769. Algunas características son: frecuencia de reloj de 120 MHz, 512 KB para memoria de programa, 64 KB de RAM, 8 canales ADC (*Analog to Digital Converter*), interfaces UART, I2C, SPI y Ethernet, entre otras. Las funciones principales de este componente abarcan: el muestreo de las señales mediante los conversores ADC y entradas digitales (I2C y GPIO), manejo de un RTC (*Real Time Clock*) para garantizar los datos de fecha y hora, lectura/escritura de una tarjeta de memoria SD mediante la interfaz SPI, manejo de módulos inalámbricos a través de interfaces de comunicación serie (UART) y una interfaz Ethernet. En particular, para este prototipo se utiliza la placa de evaluación *LPC1769 LPCXpresso Board*, indicada en la Fig. 3, que consta principalmente del microcontrolador con sus cristales y un circuito integrado LAN8720 para la capa física de Ethernet. Por su parte, el medio de almacenamiento es una tarjeta de memoria SD para lo cual se utiliza la biblioteca estándar FatFS [18] adaptada al *firmware* compatible con el modelo de microcontrolador especificado.

### B. Sistema de sensado

El conjunto de sensores descritos en la sección II, además de un circuito externo al módulo principal para la amplificación y filtrado de las señales, debe cumplir una serie de requerimientos mínimos, según se indica en el Cuadro I. De esta manera podrá garantizarse la utilidad de los datos dentro de los rangos válidos para el estudio de las magnitudes de interés. En el caso de las antenas, para capturar adecuadamente señales en el rango de 0.01-200 Hz se considera suficiente una tasa de muestreo de 1 kHz. Los acelerómetros deberán muestrearse por lo menos a una tasa de 100 Hz, al igual que magnetómetro de protones. Para el caso del detector de radón, éste sólo requerirá una entrada digital que realice el conteo de los pulsos generados por este sensor. Por su parte, para los datos de conductividad de aire, humedad y temperatura, bastará con una frecuencia de muestreo de 3.3 mHz, equivalente a una muestra cada 5 min, ya que son magnitudes que varían muy lentamente. Estos requerimientos serán fijados por defecto, no obstante, el sistema principal podrá ser configurado en tiempo real para modificar cualquiera de estos parámetros si fuera necesario.

### C. Sistema de comunicación

Si bien el principal acceso a los datos sensados, una vez transcurrido el período de tiempo estipulado para su recolección, es mediante un dispositivo de almacenamiento masivo, en este caso una tarjeta de memoria SD, el sistema además se complementa con distintas interfaces de comunicación. El propósito de éstas es transferir información de diagnóstico, como el estado de la batería, alarmas, o cualquier otro evento particular que deseé determinarse en tiempo real. También se contempla la posibilidad de que el sistema envíe una parte de los datos almacenados durante un intervalo de tiempo dado, en un día y hora particulares. Al mismo tiempo, el sistema podrá recibir comandos de control y configuración de manera remota, ya sea para modificar tiempos de muestreo, activar o desactivar sensores, pasivar al sistema y ponerlo en modo de bajo consumo, reajustar la fecha y hora del RTC, solicitar información específica o cualquier tipo de configuración requerida. Para lograr que la comunicación tenga mayor flexibilidad en cuanto a la capacidad de adaptarse a cada medio en particular en que se encuentren situado el equipo, se incluyen para este prototipo dos tecnologías de red que permitirán efectuar un enlace entre la estación de monitoreo y el sistema: LAN (*Local Area Network*) y WPAN (*Wireless Personal Area Network*). No obstante, también se contempla al menos una interfaz USB principalmente para acceso en campo y configuración de *firmware*. En la Fig. 4 se puede apreciar la topología típica empleada en la comunicación para los dos tipos de tecnologías mencionadas.

1) *Enlace LAN*: La placa principal incluye una interfaz Ethernet que permitirá la comunicación entre el sistema de adquisición y cualquier otro dispositivo compatible con el estándar IEEE 802.3. De esta manera, podrá accederse a la información desde cualquier lugar del planeta a través de la red global, en caso que la estación disponga de un proveedor de internet, o limitarse específicamente a los dispositivos conectados al enlace local. Si bien una de las desventajas más evidentes es que el equipo podría estar instalado en zonas muy aisladas, donde no se disponga de una estación cercana para un enlace directo, puede igualmente aprovecharse esta interfaz. Aún en esos casos más extremos, podría conectarse a un sistema satelital que garantice la comunicación, ya que una de las principales ventajas de esta tecnología es su uso universal y estándar, compatible con múltiples dispositivos disponibles en el mercado. Por otra parte, el *firmware* implementado en el microcontrolador para garantizar este esquema de comunicación, hace uso de la biblioteca denominada LWIP [19] que implementa una versión del protocolo TCP/IP pero de tamaño reducido y apropiado para dispositivos con poca memoria de programa, como es el caso de microcontroladores, soportando los protocolos: ARP, IP, TCP, UDP, ICMP y DHCP.

2) *Enlace WPAN*: En este caso se utiliza un módulo de comunicación para *redes inalámbricas de área personal* (WPAN) compatible con el estándar IEEE 802.15.4, siendo éste la base de la especificación Zigbee ampliamente utilizada en aplicaciones *redes de sensores inalámbricas* (WSN,

*Wireless Sensor Networks*). Para ello, en la placa principal se incorpora un módulo marca *Xbee* que permite una conectividad inalámbrica de corto a mediano alcance, 30-100 m en zonas urbanas y varios cientos de metros en zonas rurales [20]. Éste *transceiver* opera en la banda ISM (2.4 GHz), con una tasa de datos máxima de 250 kbps, una interfaz serie para la comunicación con el microcontrolador y modos de bajo consumo para un uso eficiente de energía, entre otras características. De esta manera, la estación podrá ubicarse a varios cientos de metros, dependiendo del modelo del *transceiver*, la antena y la geometría del lugar. Algunas de las ventajas que ofrece este módulo inalámbrico son que puede funcionar en modo punto a punto sin la necesidad de implementar ningún protocolo adicional en el *firmware* del microcontrolador, un bajo consumo de energía y bajo costo económico. Según la Fig. 4, la conexión punto a punto entre sistema de adquisición y la estación es a través de un nodo *Gateway*. Éste último, además de un módulo inalámbrico *Xbee* posee también una interfaz Ethernet, necesaria para permitir la conectividad del sistema con internet.

3) *Interfaz USB*: Adicionalmente, se agrega una interfaz USB que emula un puerto serie virtual, ya que mediante el conversor serie-USB FT232RL permite adaptar la comunicación a través de una interfaz UART del microcontrolador a un puerto USB al que podrá accederse desde cualquier computadora, previa instalación de *drivers* FTDI. De esta manera, permitirá la configuración del sistema, depuración de *firmware* y la obtención de información de diagnóstico en campo, accediendo al dispositivo en forma directa.

### D. Sistema de alimentación

Uno de los requerimientos más importantes de todo el sistema es su autonomía energética. Esto se debe a que el equipo requiere de períodos muy prolongados de tiempo sin intervención humana, que podrían ser varios meses, ya que en la mayoría de los casos estará situado en lugares despoblados o sin acceso a la red de suministro eléctrico. Por lo tanto, si bien no ha sido implementado para el presente trabajo, es indispensable el desarrollo de un sistema de alimentación capaz de cubrir estas necesidades. La solución contemplada para este caso, es la inclusión de un panel solar, que junto a

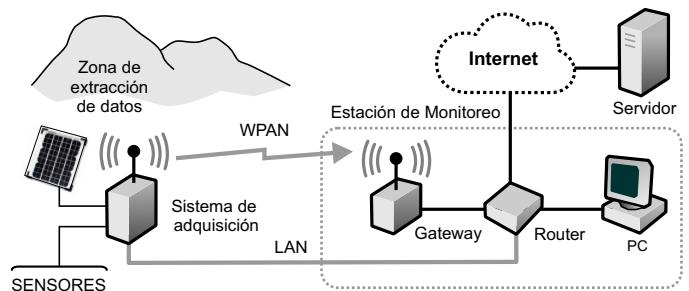


Figura 4. Topología típica para el sistema de comunicación. En la figura se muestran los dos tipos de conexiones implementadas, LAN y WPAN, ubicando al sistema de adquisición en la zona de interés para la extracción de los datos.

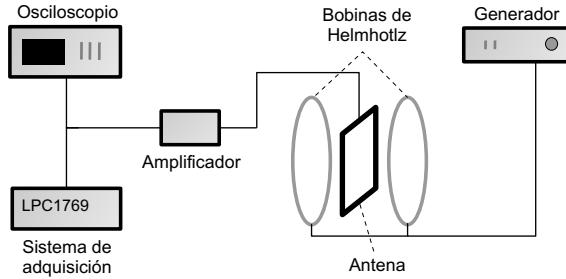


Figura 5. Banco de medición para la adquisición de ondas de prueba.

un circuito de fuente y regulación podrán mantener un ciclo de carga adecuado para la batería y así garantizar su autonomía, optimizando su rendimiento mediante un monitoreo permanente desde el sistema principal.

#### IV. RESULTADOS

Para verificar el desempeño del dispositivo de adquisición, se realizaron mediciones con dos tipos de sensores: primero se utilizó la antena descripta en la sección II, y luego el sensor de vibraciones piezoeléctrico. Para el primer caso, empleando el banco de medición de la Fig. 5, se ha medido la tensión inducida en una antena cuadro para dos frecuencias:  $f=10$  Hz y 100 Hz, para un determinado campo magnético constante impuesto mediante bobinas de Helmholz [21]. En primer lugar, la Fig. 6 muestra la captura con osciloscopio de la señal medida por la antena para la onda de 10 Hz, mientras que en la Fig. 7 (a) se indica la misma onda capturada en la tarjeta SD por el sistema de adquisición y luego procesada mediante el software *Octave*, apreciándose un resultado satisfactorio si se la compara con la Fig. 6. Por otra parte, la Fig. 7 (b) muestra el espectro de la señal con la componente de 10 Hz inducida, pero observando también la presencia de una señal interferente de 50 Hz producida por la red de alimentación de 220 VAC. Esto hace imperativa la necesidad de incorporar un filtro tipo *Notch*, teniendo en cuenta que para mediciones en campo las señales de interés podrían ser varios órdenes de magnitud más bajas que la interferencia de 50 Hz, ya que ésta podría producir un enmascaramiento de las señales bajo estudio. También se observan varios armónicos, múltiplos de 10 Hz y 50 Hz, que se originan debido a la alinealidad del amplificador que proporciona la ganancia para la señal inducida por la antena. Del mismo modo, en la Fig. 8, se representan las capturas efectuadas pero esta vez para una señal inducida de 100 Hz, observándose similares características que en el caso anterior.

Por otra parte, para el sensor de vibraciones presentado en la sección II, se obtuvieron las capturas que se indican en la Fig. 9. Allí se pueden observar dos escalas de tiempo distintas de la misma señal capturada, indicando el instante en que se producía una perturbación intencional durante la adquisición.

#### V. CONCLUSIONES

En este trabajo se presentó el desarrollo de un sistema de adquisición de datos cuyo propósito es medir campo magnéti-

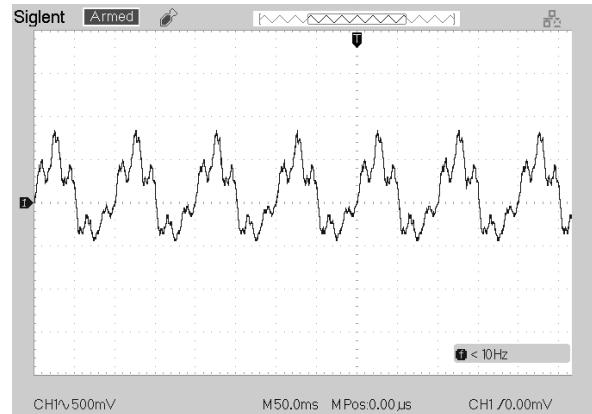


Figura 6. Tensión inducida para la onda de 10 Hz: captura de osciloscopio.

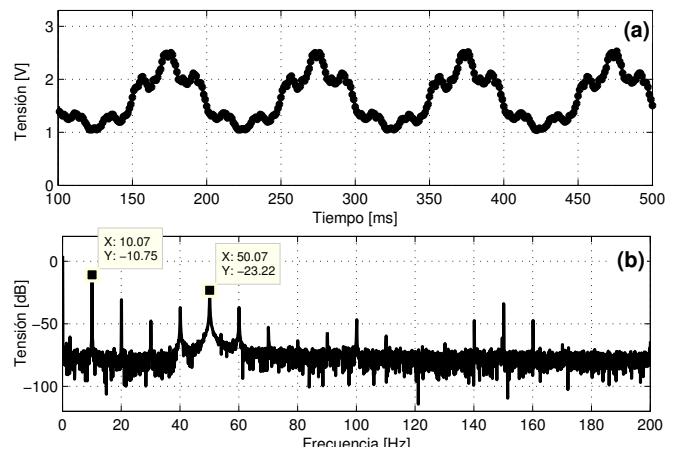


Figura 7. Tensión inducida a 10 Hz: (a) dominio del tiempo, (b) dominio de la frecuencia. Se observa interferencia de 50 Hz y armónicos por distorsión

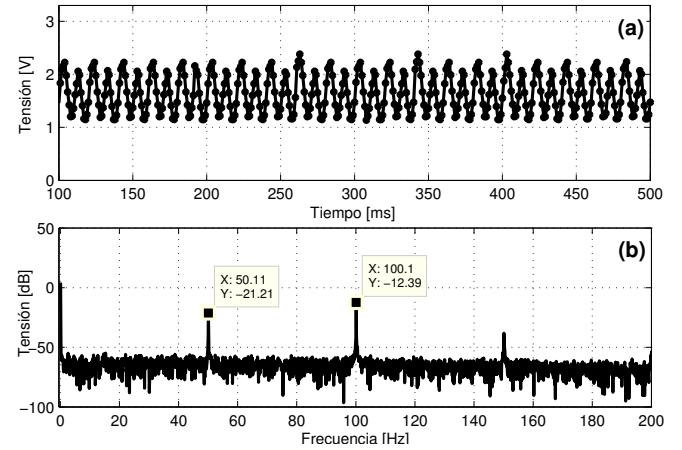


Figura 8. Tensión inducida a 100 Hz: (a) dominio del tiempo, (b) dominio de la frecuencia. Se observa interferencia de 50 Hz y armónicos por distorsión.

co en el rango aproximado de 0.01-200 Hz, monitoreando a su vez la actividad sísmica y recopilando la información para su posterior análisis.

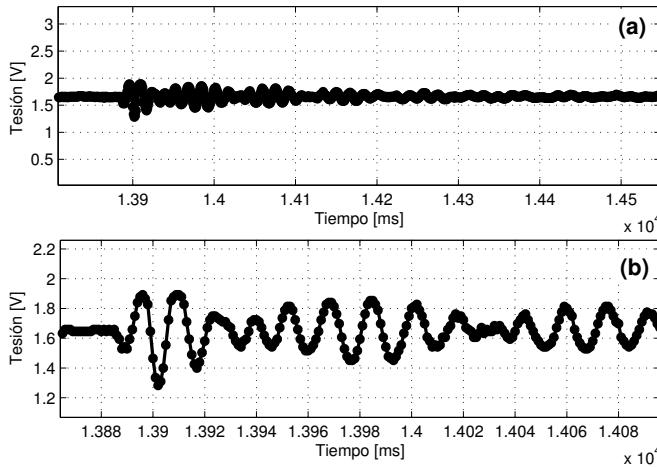


Figura 9. (a) Tensión producida por el sensor piezoelectrónico frente a una perturbación. (b) ampliación de la escala de tiempo de la señal en (a).

Se ha presentado una descripción sobre: los sensores requeridos para esta aplicación, la arquitectura de *hardware* del sistema de adquisición, las capturas realizadas en pleno funcionamiento, tanto en el dominio del tiempo como de la frecuencia, mostrando una buena convergencia de los resultados al contrastarlos con las mediciones en osciloscopio.

En las mediciones de campo magnético, se ha visualizado una señal interferente de 50 Hz proveniente de la red eléctrica en el ámbito de las mediciones. También se observó la presencia de armónicos causados por alinealidades del amplificador de acondicionamiento de señal.

## VI. TRABAJOS FUTUROS

Uno de los trabajos pendientes a futuro es el diseño del sistema de alimentación que garantice la autonomía del sistema completo mediante un panel solar y batería. Además, el equipo deberá estar protegido para soportar condiciones ambientales y meteorológicas extremas, tales como amplias variaciones de temperatura, lluvias, humedad, viento, descargas eléctricas, etc.. propias de la ubicación donde se instalará. Por ejemplo, se contempla algunas zonas de interés como el norte de Catamarca y Copahue, Neuquén, con colaboración de otros grupos de investigación.

Para la adquisición de señales, se prevé la realización de un filtro *Notch* que elimine posibles interferencias de 50 Hz evitando así el enmascaramiento de la señal de interés. Además resulta necesario corregir las alinealidades del amplificador de acondicionamiento de señal con el fin de reducir el contenido armónico que produce.

Resulta también de gran importancia estudiar los mecanismos de encriptación y seguridad para la protección de los datos

que pudieran transmitirse de forma inalámbrica. Por otra parte, el sistema de comunicación podrá extenderse incorporando un módulo GSM para aprovechar la red de telefonía celular, mejorando así la posibilidad de conectarse con el sistema en zonas aisladas. En el mismo sentido, para casos más extremos en donde ni siquiera exista cobertura de telefonía móvil, podrá considerarse la utilización de un modem satelital conectado al equipo mediante la interfaz Ethernet, pero debido a su mayor costo, será una opción contemplada como último recurso.

## REFERENCIAS

- [1] D. James and B. Perry, "History of seismometry (to 1900)," *Bulletin of the SSA Vol. 59, No. 1, pp. 183-227*, 1969.
- [2] B. Shannon, "Vibration measurement systems and guidelines for centrifugal fans - a field perspective," *AMCA International Engineering Conference Las Vegas, NV, USA 2 – 4 March*, 2008.
- [3] D. Karakelian, S. L. Klemperer, A. C. Fraser Smith, and G. C. Beroza, "A transportable system for monitoring ultra low frequency electromagnetic signals associated with earthquakes," *Seismological Research Letters*, vol. 71, no. 4, pp. 423–436, 2000.
- [4] E. Maffia, V. Trainotti, W. G. Fano, and N. Trench, "Medición de la resonancia de schumann," *Latinmag Letters*, vol. 1, no. A02, pp. 1–8, 2011.
- [5] W. H. Campbell, "Concurrent equatorial and high-latitude geomagnetic pulsations," *Radio Science*, vol. 4, no. 9, pp. 859–868, 1969.
- [6] M. Hata and S. Yabashi, "Pre and after-sign detection of earthquake through elf radiation," *Geoscience and Remote Sensing Symposium, 1993. IGARSS '93. Better Understanding of Earth Environment., International*.
- [7] T. Bleier, C. Dunson, S. Roth, J. Heraud, L. A., F. Freund, and R. Dahlgren, "Ground-based and space-based electromagnetic monitoring for pre-earthquake signals," *Earthquake Prediction Studies: Seismo Electromagnetics*, pp. 1–15, 2013.
- [8] J. Rosen, "Can electric signals in earth's atmosphere predict earthquakes?" *DOI: 10.1126/science.aae0148, Science*, 2015.
- [9] H. V. Alvan and A. F. H., "Satellite remote sensing in earthquake prediction. a review," *National Postgraduate Conference (NPC)*, 2011.
- [10] M. Ikeya, "Earthquakes and animals: From folk legends to science," *World Scientific Publishing Co. Pte. Ltd. 5 Toh Tuck Link, Singapore 596224*, 2004.
- [11] T. Bleier and F. Freund, "Ground-based and space-based electromagnetic monitoring for pre-earthquake signals," *IEEE Spectrum*, pp. 1–8, 2005.
- [12] W. G. Fano, "Rf emissions of compact fluorescent lights," *Interference Technology*, 2012.
- [13] W. G. Fano and G. Escrivá, "Near field characteristics of mf monopole with a parasitic," *Progress in Electromagnetic Research Letters*, vol. 5.
- [14] P. Neher, "Radon monitor," *Electronics Now*, pp. 56–70, 1994.
- [15] P. Richon and J. C. Sabroux, "Radon anomaly in the soil of taal volcano, the philippines: A likely precursor of the m 7.1 mindoro earthquake (1994)," *Geophysical Research Letters*, 2003.
- [16] L. Huggard, "Proton magnetometer," *Practical Electronics Oct.*, 1995.
- [17] J. R. Santalucía and E. Gargiulo, "Transductor de aceleración para la detección de sismos fuertes," *ISBN: 9974-0-0337-7, IBERSENSOR, Montevideo*, 2006.
- [18] S. Labs, "Fat on sd card," *AN0030 - Application Note*, 2013, 2013.
- [19] A. Dunkels, "Design and implementation of the lwip tcp/ip stack," *Swedish Institute of Computer Science*, 2001.
- [20] Digi, "Xbee and xbee-pro zigbee, rf module," *User Guide*, 2017.
- [21] E. L. Bronaugh, "Helmholtz coils for calibration of probes and sensors: Limits of magnetic field accuracy and uniformity," *Conference: Electromagnetic Compatibility*, 1995.

## 14.3 Factibilidad técnica

### 14.3.1 Formato YAML

```
1 package_id: '00001'  
2 package_len: 018008580  
3 package_date: '20191011172018'  
4 segments:  
5   - segment_id: '001'  
6     segment_date: '20191011172018'  
7     segment_microsec: '00000'  
8     segment_rate: '0500'  
9     flag1: flag1  
10    flag2: flag2  
11    flag3: flag3  
12    flag4: flag4  
13    payload: 'T25.00H30.00'  
14   - segment_id: '002'  
15     segment_date: '20191011172018'  
16     segment_microsec: '00000'  
17     segment_rate: '0500'  
18     flag1: flag1  
19     flag2: flag2  
20     flag3: flag3  
21     flag4: flag4  
22     payload: 'T27.00H32.00'
```

Fragmento de código 5: Paquete YML definido

## 14.4 Ingeniería en detalle

### 14.4.1 Escritura serial

```
1  #!/usr/bin/env python
2
3  import time
4  import serial
5
6  ser = serial.Serial(
7      port='/dev/ttyAMA0',
8      baudrate = 9600,
9      parity=serial.PARITY_NONE,
10     stopbits=serial.STOPBITS_ONE,
11     bytesize=serial.EIGHTBITS,
12     timeout=1
13 )
14 counter=0
15
16 while 1:
17     ser.write('Write counter: %d \n'%(counter))
18     time.sleep(1)
19     counter += 1
```

Fragmento de código 6: serial\_print.py

#### 14.4.2 Lectura serial

```
1 #!/usr/bin/env python
2
3 import time
4 import serial
5
6 ser = serial.Serial(
7     port='/dev/ttyUSB0',
8     baudrate = 9600,
9     parity=serial.PARITY_NONE,
10    stopbits=serial.STOPBITS_ONE,
11    bytesize=serial.EIGHTBITS,
12    timeout=1
13 )
14 counter=0
15
16 while 1:
17     x=ser.readline()
18     print x
```

Fragmento de código 7: serial\_read.py

#### 14.4.3 Configuración 3G

```
1 [Dialer Tuenti]
2 Init1 = ATZ
3 Init2 = at+cgdcont=1,"ip","internet.movil"
4 Modem Type = Analog Modem
5 Phone = *99#
6 Baud = 460800
7 New PPPD = yes
8 Modem = /dev/ttyUSB0
9 ISDN = 0
10 Username = XXXX
11 Password = XXXX
12 Stupid Mode = 1
```

Fragmento de código 8: Archivo de configuración wvdial.conf

#### 14.4.4 Paquete JSON de la API a la base de datos

```
1  {
2      "Db_connection": {
3          "db_id": "id",
4          "db_pass": "pass"
5      },
6      "package": {
7          "package_id": "00001",
8          "package_len": "018008580",
9          "package_date": "2019911172018",
10         "segments": [
11             {
12                 "segment_id": "001",
13                 "segment_date": "20191011172018",
14                 "segment_microsec": "00000",
15                 "segment_rate": "0500",
16                 "flag1": "flag1",
17                 "flag2": "flag2",
18                 "flag3": "flag3",
19                 "flag4": "flag4",
20                 "payload": {
21                     "temperature": "25.00",
22                     "humidity": "30.00"
23                 }
24             },
25             {
26                 "segment_id": "002",
27                 "segment_date": "20191011172018",
28                 "segment_microsec": "00000",
29                 "segment_rate": "0500",
30                 "flag1": "flag1",
31                 "flag2": "flag2",
32                 "flag3": "flag3",
33                 "flag4": "flag4",
34                 "payload": {
35                     "temperature": "27.00",
36                     "humidity": "32.00"
37                 }
38             }
39         ]
40     }
41 }
```

Fragmento de código 9: Cuerpo JSON para subir temperatura y humedad

#### 14.4.5 Configuración de Grafana Enterprise

```
1 ##### Server #####
2 [server]
3 # Protocol (http, https, h2, socket)
4 protocol = http
5
6 # The ip address to bind to, empty will bind to all interfaces
7 http_addr =
8
9 # The http port to use
10 http_port = 3000
11
12 # The public facing domain name used to access grafana from a browser
13 domain = localhost
14
15 # Redirect to correct domain if host header does not match domain
16 # Prevents DNS rebinding attacks
17 ;enforce_domain = false
18
19 # The full public facing url
20 root_url = %(protocol)s://%(domain)s/grafana/
21
22 # Serve Grafana from subpath specified in `root_url` setting.
23 # By default it is set to `false` for compatibility reasons.
24 serve_from_sub_path = true
25
26 # Log web requests
27 router_logging = false
28
29 # the path relative working path
30 static_root_path = public
31
32 # enable gzip
33 enable_gzip = false
34
35 # https certs & key file
36 cert_file =
37 cert_key =
38
39 # Unix socket path
40 socket = /tmp/grafana.sock
```

Fragmento de código 10: Archivo de configuración custom.conf de Grafana

```
1 server {
2     listen 80;
3     root /usr/share/nginx/www;
4     index index.html index.htm;
5
6     location /grafana/ {
7         proxy_pass http://localhost:3000/;
8     }
9 }
```

Fragmento de código 11: Configuración del proxy nginx - config.conf

## **14.5 Módulos complementarios**

### **14.5.1 Interface Control Document**

---

# **Plataforma de telemetría IoT aplicada al estudio de precursores sísmicos**

---

## **Interface Control Document**

---

**Version 1.1**

**13/02/2021**

## Table of Contents

<b>1.</b>	<b>Propósito del control de interfaz .....</b>	<b>1</b>
<b>2.</b>	<b>Introducción.....</b>	<b>2</b>
<b>3.</b>	<b>Plataforma de telemetría.....</b>	<b>3</b>
3.1.	Interfaz plataforma de telemetría - modulo de sensado .....	3
3.1.1.	Comunicación.....	3
3.1.2.	Limitaciones de la interfaz.....	5
<b>4.</b>	<b>CALSIM.....</b>	<b>6</b>
4.1.	Interfaz CALSIM - sensor analógico .....	7
4.1.1.	Comunicación.....	7
4.1.2.	Limitaciones .....	7
4.2.	Interfaz CALSIM - módulo intercomunicador .....	8
4.2.1.	Comunicación.....	8
4.2.2.	Limitaciones .....	8

## 1. Propósito del control de interfaz

---

Este Documento de control de interfaz (ICD) identifica y especifica la información necesaria para definir de manera efectiva las interfaces del proyecto “*Plataforma de telemetría IoT aplicada al estudio de precursores sísmicos*”. El propósito de este ICD es comunicar claramente todas las posibles entradas y salidas del sistema para todas las acciones potenciales, ya sean internas al sistema o transparentes para los usuarios del sistema.

Este control de interfaz se crea durante las fases de planificación y diseño del proyecto y ayuda a garantizar la compatibilidad entre los componentes y los segmentos del sistema.

## 2. Introducción

---

Este ICD describe la relación entre el sensor o módulo de sensado (el sistema de origen) y la “*Plataforma de telemetría IoT aplicada al estudio de precursores sísmicos*” (el sistema de destino).

Adicionalmente también describe la relación entre la plataforma de telemetría y el módulo adicional *CALSIM*.

Este ICD especifica los requisitos de interfaz que deben cumplir los sistemas participantes, define la estructura del mensaje y los protocolos que gobiernan el intercambio de datos e identifica las rutas de comunicación por las que el equipo del proyecto espera que fluyan los datos.

Para cada interfaz, el ICD proporciona la siguiente información:

- Una descripción del formato de intercambio de datos y el protocolo para el intercambio.
- Una descripción general de la interfaz.
- Supuestos cuando corresponda

## 3. Plataforma de telemetría

### 3.1. Interfaz plataforma de telemetría - modulo de sensado

La interfaz con el módulo de sentado es la entrada principal a la plataforma. A partir de esta se realiza el ingreso de los datos aún en formato correcto para su próximo almacenamiento y consumo visual.

#### 3.1.1. Comunicación

La plataforma espera de entrada un paquete en formato YAML con ciertas características, y es el módulo de senado quien en esta interfaz está encargado de generarlo. Este formato se presenta a continuación, donde el payload es efectivamente las mediciones tomadas:

```
package_id: '00001'
package_len: 018008580
package_date: '20191011172018'
segments:
- segment_id: '001'
  segment_date: '20191011172018'
  segment_microsec: '00000'
  segment_rate: '0500'
  flag1: flag1
  flag2: flag2
  flag3: flag3
  flag4: flag4
  payload: 'XXXX'
- segment_id: '002'
  segment_date: '20191011172018'
  segment_microsec: '00000'
  segment_rate: '0500'
  flag1: flag1
  flag2: flag2
  flag3: flag3
  flag4: flag4
  payload: 'XXXX'
```

El intercambio se realiza en 2 pasos, en una primera instancia el módulo de senado comunica a la plataforma el tamaño en bits del paquete, esta envía una señal de recepción y luego el módulo finaliza enviando el paquete propiamente dicho.

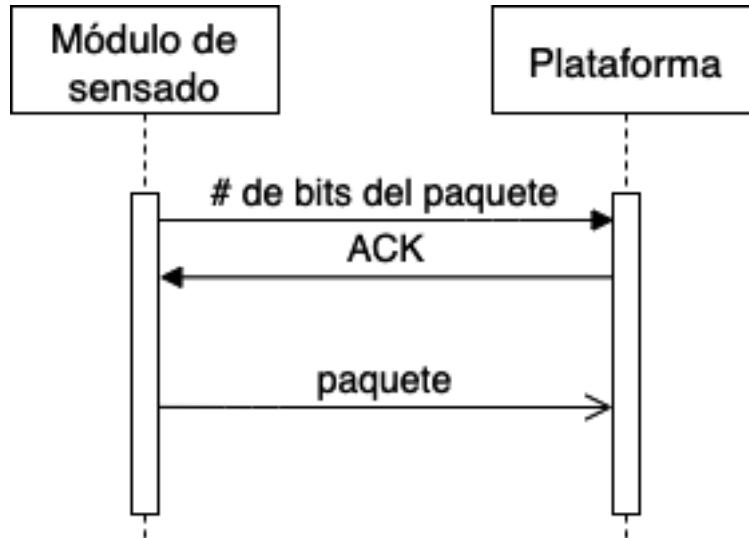


Figura 1: Recepción del paquete

Finalmente, la comunicación se realiza por protocolo serie directamente al puerto UART de la plataforma. Más precisamente esto la conexión es a los pines 8 y 10 de la placa Raspberry Pi, donde el 8 será el transmisor serie para el mensaje de ACK y el 10 será el receptor del paquete.

BOARD	GPIO		GPIO	BOARD
01	3.3v DC Power		DC Power 5v	02
03	GPIO02 (SDA1 , I <sup>2</sup> C)		DC Power 5v	04
05	GPIO03 (SCL1 , I <sup>2</sup> C)		Ground	06
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14	08
09	Ground		(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)		Ground	14
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23	16
17	3.3v DC Power		(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)		Ground	20
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08	24
25	Ground		(SPI_CE1_N) GPIO07	26
27	ID_SD (I <sup>2</sup> C ID EEPROM)		(I <sup>2</sup> C ID EEPROM) ID_SC	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12	32
33	GPIO13		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40

Figura 2: Pinout de la Raspberry Pi

### 3.1.2. Limitaciones de la interfaz

- El modulo de sentado no solo debe poder transmitir, sino también recibir por puerto serie.
- El modulo de sensado debe armar el paquete correspondiente previo a la transmisión y no solamente enviar las mediciones.
- Todos los puertos UART en la Raspberry Pi son de 3,3 V; se producirán daños si se conectan a sistemas de 5 V.

## 4. CALSIM

El agregado del CALSIM ofrece 2 funciones:

- Funciona como módulo calibrador conectándose a la plataforma como módulo de sensado permitiendo enviar un paquete conocido. De esta manera los datos que contiene serán almacenados, visualizados y últimamente podrán compararse con el resultado esperado para validar que la plataforma esta funcionando correctamente.
- Provee una interfaz para conectarle un sensor analógico y envía su mediciones en el paquete requerido por la plataforma de telemetría, sin necesidad de que el sensor lo haga.



## 4.1. Interfaz CALSIM - sensor analógico

### 4.1.1. Comunicación

El módulo CALSIM se comunica a través de 2 puertos con una estación de sentado analógica, uno para la conexión con el sensor y otra para la tensión de referencia. Los mismos son los que se encuentran a la derecha en la imagen.

El sensor únicamente debe transmitir las tensiones en el rango aceptado y el módulo CALSIM es capaz de formar el paquete en el formato necesario y enviarlo a través de su conexión serie a la plataforma de telemetría siguiendo las especificaciones de la sección 3 de este documento.



### 4.1.2. Limitaciones

- El circuito requiere una alimentación de 5V que puede proveerse desde el mismo módulo intercomunicador a través del puerto USB.
- La entrada analógica del CALSIM acepta tensiones entre 0 y 5V.
- Posee una precisión de la medición de 5mV.

## 4.2. Interfaz CALSIM - módulo intercomunicador

### 4.2.1. Comunicación

El módulo CALSIM se comunica con el módulo intercomunicador siguiendo los lineamientos explicados en la sección 3.1 de este documento, posee puertos serie de recepción y transmisión respectivamente ubicados a la izquierda del dispositivo.



### 4.2.2. Limitaciones

- El circuito requiere una alimentación de 5V que puede proveerse desde el mismo módulo intercomunicador a través del puerto USB.
- La comunicación serie en los puertos TX / RX utiliza niveles lógicos TTL de 5V.

## 14.6 Construcción del prototipo

### 14.6.1 Primera publicación

```
1 20191011172018
2 ID:000001
3 LEN:018008580
4 BEGIN_SEGMENT
5 SEGMENT_ID:001
6 20191011172018
7 MICROSEC:000000
8 RATE:0500
9 FLAG1:001
10 FLAG2:002
11 FLAG3:003
12 FLAG4:004
13 BEGIN_DATA
14 XX
15 END_DATA
16 END_SEGMENT
```

Fragmento de código 12: Paquete de ejemplo utilizado para pruebas - paquete\_ejemplo.txt

```
1 country=ar
2 update_config=1
3 ctrl_interface=/var/run/wpa_supplicant
4
5 network={
6   scan_ssid=1
7   ssid="Wifi Network Name"
8   psk="password"
9 }
```

Fragmento de código 13: Archivo de configuración para conectar Raspberry a WiFi wpa-supplicant.conf

```
1 worker: python mqttSubscriber.py
```

Fragmento de código 14: Archivo Procfile

## 14.6.2 Operacionalización de los clientes

```
 1 from collections import Mapping, Container
 2 from itertools import islice
 3 from sys import getsizeof
 4
 5 CHUNK_SIZE = 100 * 1000 # 100kb
 6
 7 def split_by_size(file_path):
 8     with open(file_path, "rb") as f:
 9         i = 0
10         chuncks = list()
11         print('empty chunks size:', getsizeof(chuncks))
12
13     while True:
14         piece_of_file = f.read(CHUNK_SIZE)
15         print('piece_of_file size:', getsizeof(piece_of_file))
16
17         if piece_of_file.decode() == "":
18             break
19         chuncks.append(piece_of_file)
20         i = i + 1
21         print('chunk number', i)
22
23     print('amount of chuncks to send:', len(chuncks))
24     return chuncks
```

Fragmento de código 15: Porción de helpers.py

### 14.6.3 Incorporación de la base de datos

```
1  def upload_to_database(topic, msg):
2      try:
3          package_to_send = parse_model(topic, msg)
4      except Exception as e:
5          print("Data was incomplete and model was not created:")
6          print(e)
7          return -1
8
9      if topic == mqtt_client.Topics.SEISMIC.value:
10          db_creds = db_credentials.DbCredentials(base64.b64decode
11                                         (os.environ['SEISMIC_USR'])
12                                         .decode('utf-8'),
13                                         base64.b64decode
14                                         (os.environ['SEISMIC_SCT'])
15                                         .decode('utf-8'))
16          endpoint = BASE_ENDPOINT + "seismic-data"
17      elif topic == mqtt_client.Topics.T_AND_H.value:
18          db_creds = db_credentials.DbCredentials(base64.b64decode
19                                         (os.environ['TH_USR'])
20                                         .decode('utf-8'),
21                                         base64.b64decode
22                                         (os.environ['TH_SCT'])
23                                         .decode('utf-8'))
24          endpoint = BASE_ENDPOINT + "temp-hum"
25
26      data = form_data(package_to_send, db_creds)
27      r = requests.post(url=endpoint, data=data, headers=
28                         {'content-type':
29                          'application/json'})
30      print(r.text)
31      return r.status_code
```

Fragmento de código 16: Función de Registro en la Base de Datos

#### 14.6.4 Escalabilidad a múltiples aplicaciones

```
1 import yaml
2 from model import package
3
4 def parse_model(topic, msg):
5     pack_as_dict = yaml.safe_load(msg)
6     pack = package.Package(topic, **pack_as_dict)
7     return pack
```

Fragmento de código 17: Función de parseo del mensaje a formato YAML

# Capítulo 15

## Referencias

<sup>1</sup> Dave Evans. Cisco Internet Business Solutions Group (IBSG), abril de 2011 (p. 2).  
*Internet de las cosas - Cómo la próxima evolución de Internet lo cambia todo*

<sup>2</sup> Mohammadi Zanjireh, Morteza & Larijani, Hadi. (2015).  
*A Survey on Centralised and Distributed Clustering Routing Algorithms for WSNs*

<sup>3</sup> Conner, Margery (9 de mayo de 2010).  
*Sensors empower the "Internet of Things"* (Issue 10). pp. 32-38. ISSN 0012-7515.

<sup>4</sup> Philippe Gautier  
*RFID y adquisición de datos Evenementielles: retours d'expérience chez Benedicta* - páginas 94 a 96, Systèmes d'Information et Management - revista trimestral N ° 2 vol. 12, 2007, ISSN 1260-4984 / ISBN 978-2-7472-1290-8, éditions ESKA.

<sup>5</sup> HP News.  
*HP Study Reveals 70 Percent of Internet of Things Devices Vulnerable to Attack*  
<https://www8.hp.com/us/en/hp-news/press-release.html?id=1744676#.X57K7ohKhPY>

<sup>6</sup> Naciones Unidas  
*Una población en crecimiento*  
<https://www.un.org/es/sections/issues-depth/population/index.html>

<sup>7</sup> Project Management Institute.  
*Project Management Body of Knowledge*, 5th Edition.

<sup>8</sup> University of Technology Sidney.  
*Program Guide to Project Management*, Sidney, Australia.

<sup>9</sup> Alberto Allami, Certified PMP.  
*La gestión exitosa de proyectos según el estándar del Project Management Institute (PMI)*

<sup>10</sup> Stanley E. Portny, Certified PMP.  
*Project Management For Dummies*, 3rd Edition, Wiley Publishing, Inc.

<sup>11</sup> BBC Mundo News - ”¿Qué son los misteriosos destellos de luz que aparecieron en el cielo de México durante el terremoto?” <https://www.bbc.com/mundo/noticias-41201049>

<sup>12</sup> L. M. Carducci, R. Alonso, F. Luna, J. Zola, E. Zothner, W. G. Fano.

*Medición y procesamiento de señales para la detección de anomalías de campo magnético de muy bajas frecuencias* Universidad de Buenos Aires, Facultad de Ingeniería. Buenos Aires, Argentina, 2019.

<sup>13</sup> L. M. Carducci, R. Alonso, W. G. Fano.

*Estación de Sensado para Emisiones de Campo Magnético a Frecuencias Extremadamente Bajas y su Aplicación en la Predicción de Sismos* Universidad de Buenos Aires, Facultad de Ingeniería. Laboratorio de radiación electromagnética - Laboratorio de procesamiento de señales y comunicaciones, 2018.

<sup>14</sup> Ishikawa, Kaoru; traducción del japonés al inglés por David J. Lu ; traducción Margarita Cárdenas (1997)

*Qué es el control total de calidad? : la modalidad japonesa* (11 reimpr. edición). Bogotá: Editorial Norma. p. 78. ISBN 9580470405.

<sup>15</sup> John R. Hauser (1993) How Puritan-Bennet used the house of quality  
*Sloan Management Review*, Spring, 61-70

<sup>16</sup> John R. Hauser & Don Clausing (1988) The house of quality  
*Harvard Business Review*, May–June, 63-73

<sup>17</sup> Doran, G. T.

*There's a S.M.A.R.T. way to write management's goals and objectives.* Management Review. 70 (11): 35–36. 1981.

<sup>18</sup> Philip Kotler (1992).

*Dirección de Marketing: Análisis, planificación, gestión y control*, McGraw-Hill

<sup>19</sup> Project Management Institute.

*Practice Standard for Work Breakdown Structure*, 2da Edición.

<sup>20</sup> Diagrama de Gantt - Definición,

[https://es.wikipedia.org/wiki/Diagrama\\_de\\_Gantt#cite\\_ref-1](https://es.wikipedia.org/wiki/Diagrama_de_Gantt#cite_ref-1)

<sup>21</sup> Edward R. Marsh.

*The Harmonogram of Karol Adamiecki*. Academy of Management Proceedings, 1974

<sup>22</sup> P. Mell, T. Grance,

*The NIST Definition of Cloud Computing*, National Institute of Standards and Technology, NIST Special Publication 800-145, 2011.

<sup>23</sup> Client, Broker / Server and Connection Establishment - MQTT Essentials: Part 3, hivemq.com. Tomado el 13 de Octubre de 2019.

<sup>24</sup> Yuan, Michael

*Getting to know MQTT*,

IBM Developer. Tomado el 13 de Octubre de 2019.

<sup>25</sup> LPSC Laboratorio de Procesamiento de Señales en Comunicaciones - Facultad de Ingeniería - UBA

<http://electronica.fi.uba.ar/laboratorios/lrad>

<sup>26</sup> LRAD Laboratorio de radiación electromagnética - Facultad de Ingeniería - UBA  
<http://electronica.fi.uba.ar/laboratorios/lrad>

<sup>27</sup> LSE Laboratorio de Sistemas Embebidos - Facultad de Ingeniería - UBA  
<http://laboratorios.fi.uba.ar/lse/>

<sup>28</sup> IEEE Instituto de Ingeniería Eléctrica y Electrónica  
<https://www.ieee.org/>

<sup>29</sup> INA: Instituto Nacional del Agua  
<https://www.ina.gov.ar/>

<sup>30</sup> Pycno Sensors  
<https://es.pycno.co/>

<sup>31</sup> SlantRange  
<https://slantrange.com/>

<sup>32</sup> Telemetrik - Telemetría Industrial  
<https://telemetrik.co/>

<sup>33</sup> Grafana Cloud - Precios y Productos  
<https://grafana.com/products/cloud/>

<sup>34</sup> Heroku CloudMQTT - Planes y Servicios  
<https://elements.heroku.com/addons/cloudmqtt>

<sup>35</sup> Heroku Postgres - Planes y Servicios  
<https://elements.heroku.com/addons/heroku-postgresql>

<sup>36</sup> Center for Strategic & International Studies (CSIS)  
“*Defense industrial initiatives. Current issues : Cost-plus Contracts*” (Iniciativas industriales de defensa. Temas actuales: Contratos de costo-plus)

<sup>37</sup> Brealey Myers (2008)  
*Principios de Finanzas Corporativas.* Mc. Graw Hill.

<sup>38</sup> Tasa Efectiva Mensual Vencida - Banco de la Nación Argentina,  
<https://www.bna.com.ar/Home/InformacionAlUsuarioFinanciero>

<sup>39</sup> Lopez Dumrauf, G. (2006)  
*Cálculo Financiero Aplicado, un enfoque profesional*, 2a edición, Editorial La Ley, Buenos Aires.

<sup>40</sup> Ehrhardt, Michael C.; Brigham, Eugene F. (2007)  
*Finanzas Corporativas.* Cengage Learning Editores. p. 672. ISBN 978-97-0686-594-6.

<sup>41</sup> Gustavo Mercado, Carlos Tafernaberry, Marcela Orbiscay, Marcelo Ledda, Raúl Moralejo.  
*Survey de Protocolos Normalizados por IETF para Aplicaciones de Internet of Things (IoT).* Facultad Regional Mendoza de la Universidad Tecnológica Nacional y Centro Científico Tecnológico - CONICET Mendoza, 2018.

<sup>42</sup> Z. Shelby, K. Hartke, C. Bormann.

*RFC-7252: The Constrained Application Protocol (CoAP)*. Centro tecnológico de ciencias computacionales y tecnologías de la información, Universidad Bremen TZI, Junio 2014

<sup>43</sup> Especificación de COAP por C. Bormann,

<http://coap.technology/spec.html>.

Centro tecnológico de ciencias computacionales y tecnologías de la información, Universidad Bremen TZI, 2014-2016

<sup>44</sup> Jussi Haikara.

*Publish-Subscribe Communication for CoAP*. KTH Royal Institute Of Technology, Stockholm, Suecia 2017.

<sup>45</sup> Charlie Wang, Google Cloud Solutions Architect.

*HTTP vs. MQTT: A tale of two IoT protocols* Noviembre 26, 2018.

<sup>46</sup> Mónica Martí, Carlos Garcia-Rubio, Celeste Campo.

*Performance Evaluation of CoAP and MQTT\_SN in an IoT Environment*. Departamento de Ingeniería Telemática, Universidad Carlos III de Madrid, Madrid, España. Noviembre 20, 2019.

<sup>47</sup> Lars Dürkop, Björn Czybik, Jürgen Jasperneite.

*Performance Evaluation of M2M Protocols Over Cellular Networks in a Lab Environment*. Institute Industrial IT, Ostwestfalen-Lippe University of Applied Sciences, Alemania, 2015.

<sup>48</sup> Roberto Morabito, Zakaria Laaroussi, Jaime Jiménez.

*Evaluating the Performance of CoAP, MQTT, and HTTP in Vehicular Scenarios*. Publicado en "IEEE Conference on Standards for Communications and Networking" – IEEE CSCN, 2018.

<sup>49</sup> Especificación del protocolo HTTP,

<https://www.w3.org/Protocols/>

<sup>50</sup> Norma ISO/IEC 20922:2016,

<https://www.iso.org/standard/69466.html>

<sup>51</sup> Comunicación UART en Raspberry Pi,

<https://www.raspberrypi.org/documentation/configuration/uart.md>

<sup>52</sup> Documentación oficial MQTT,

<http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html>

<sup>53</sup> Documentación Oficial de Raspberry Pi,

<https://www.raspberrypi.org/documentation/hardware/raspberrypi/power/README.md>

<sup>54</sup> Especificación de YAML 1.2,

<https://yaml.org/spec/1.2/spec.html>

<sup>55</sup> Estándar de comunicaciones IMT-2000,

<https://www.itu.int/en/ITU-T/imt-2000/Pages/default.aspx>

<sup>56</sup> Documentación de NOOBS

<https://github.com/raspberrypi/noobs/blob/master/README.md>

<sup>57</sup> Especificación del manual de Linux de wvdial,

<https://linux.die.net/man/1/wvdial>

<sup>58</sup> Pi Dramble,

<https://www.pidramble.com>

<sup>59</sup> Simulador de descarga de Raspberry Pi,

<https://learn.pi-supply.com/battery-levels/>

<sup>60</sup> Wireshark,

<https://www.wireshark.org/>

<sup>61</sup> Plataforma Heroku Cloud,

<https://www.heroku.com/home>

<sup>62</sup> MQTT Broker,

<https://www.cloudmqtt.com/docs/index.html>

<sup>63</sup> Librería Paho MQTT,

<https://pypi.org/project/paho-mqtt/>

<sup>64</sup> Plataforma de visualización Grafana,

<https://grafana.com/grafana/>

<sup>65</sup> Arduino,

<https://www.arduino.cc/>

<sup>66</sup> Microsoft Docs - Documentación Oficial

*Clases (Guía de Programación C#)*

<https://docs.microsoft.com/es-es/dotnet/csharp/programming-guide/classes-and-structs/classes>

<sup>67</sup> Microsoft Docs - Documentación Oficial

*Objetos (Guía de Programación C#)*

[https://docs.microsoft.com/es-es/previous-versions/visualstudio/visual-studio-2008/ms173110\(v=vs.90\)](https://docs.microsoft.com/es-es/previous-versions/visualstudio/visual-studio-2008/ms173110(v=vs.90))

<sup>68</sup> Microsoft Docs - Documentación Oficial

*Interfaces (Guía de Programación C#)*

<https://docs.microsoft.com/es-es/dotnet/csharp/programming-guide/interfaces>

<sup>69</sup> Microsoft Docs - Documentación Oficial

*Clases y miembros de clase abstractos y sellados (Guía de Programación C#)*

<https://docs.microsoft.com/es-es/dotnet/csharp/programming-guide/classes-and-structs/abstract-and-sealed-classes-and-class-members>

<sup>70</sup> R. E. Freeman

*Strategic Management: An Stakeholders Approach*

Fecha de publicación original: 1984

<sup>71</sup> Martin Fowler 2014

*Microservices a definition of this new architectural term.*