

ESCUELA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

REDES INALÁMBRICAS Y MÓVILES

Documentación

Diego Amil González

Iván Martínez Fernández

Pedro Otero Rivas

Javier Otero Vizoso

Axel Valladares Pazó

Samuel Yañez Delgado

Vigo

2 de abril de 2023

Índice

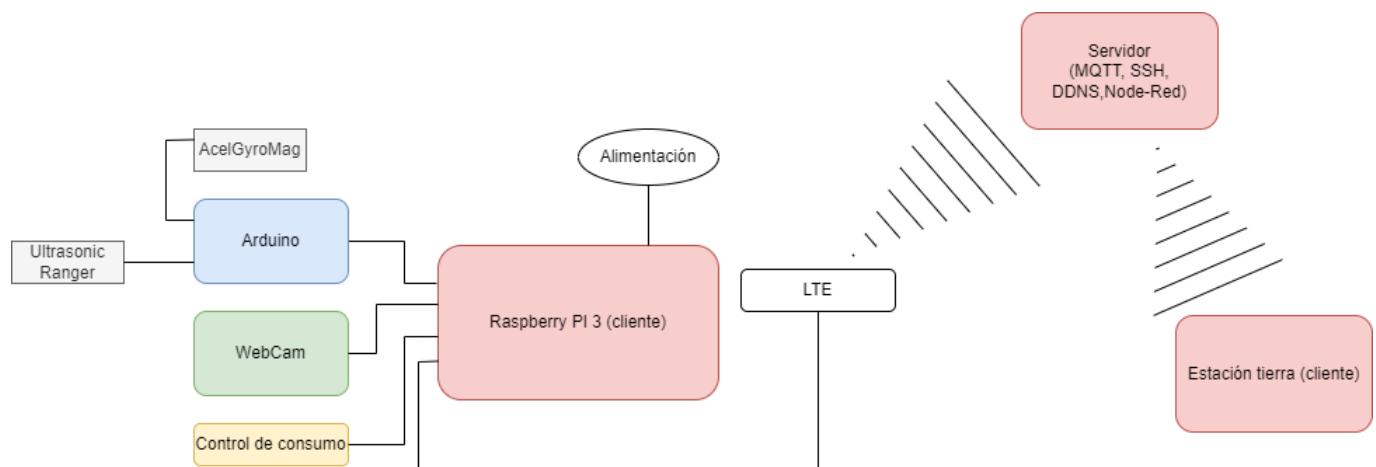
1	Explicación	3
2	Arquitectura	3
2.1	Sensores	4
2.2	Adafruit Metro	4
2.3	Cámara web	5
2.4	USB detector	5
2.5	Módem USB	5
2.6	Raspberry Pi 3 Modelo B	6
2.7	Streaming	6
2.8	Servidor	6
2.9	Interfaz Gráfica	7
2.10	Estación tierra	7
3	Tecnologías probadas	8
3.1	LTE	8
3.2	NB-IoT	8
3.3	WiFi	9
4	Protocolos probados	10
4.1	UDP	10
4.2	TCP	10
4.3	HTTP	10
4.4	MQTT	11
5	Evaluación de prestaciones	12
5.1	Plan Evaluación de Prestaciones	12
5.1.1	Latencia	12
5.1.2	Jitter	12
5.1.3	Batería/Consumo	13
5.1.4	Ancho de banda / Bit Rate	13
5.1.5	Calidad del canal	14
5.1.6	Alcance de la señal / Potencia	14
5.1.7	Coste monetario	14
5.1.8	Pérdidas	15
5.1.9	Seguridad	15
5.1.10	Frecuencias de trabajo (banda libre o licenciada)	16
5.1.11	Facilidad de instalación	16
5.2	Informe de Prestaciones	17
5.2.1	Latencia	17
5.2.2	Jitter	19
5.2.3	Batería/Consumo	20
5.2.4	Ancho de banda / Bit Rate	21
5.2.5	Calidad del canal	21
5.2.6	Alcance de la señal / Potencia	22

5.2.7	Coste monetario	22
5.2.8	Pérdidas	22
5.2.9	Seguridad	23
5.2.10	Frecuencias de trabajo (banda libre o licenciada)	23
5.2.11	Facilidad de instalación	24
6	Prototipo final	25

1. Explicación

Desarrollo de una solución óptima para la telemetría de datos generada por un barco solar. Se ha realizado un estudio sobre distintas tecnologías inalámbricas para decidir cual implementar, así como qué protocolos de comunicación usar.

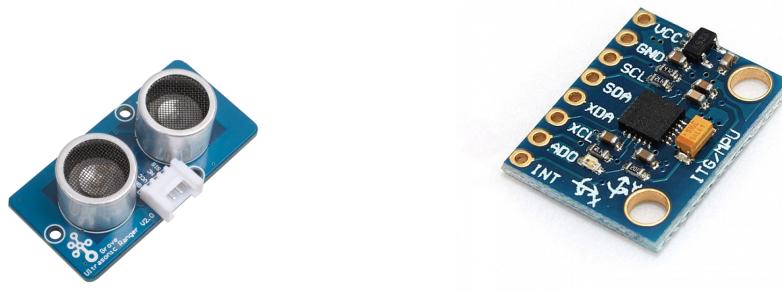
2. Arquitectura



2.1. Sensores

Se han implementado:

- Un sensor de ultrasonido para la medición de distancia con objetos con el fin de evitar colisiones.
- Un acelerómetro, giroscopio y magnetómetro para poder determinar la posición actual del barco.



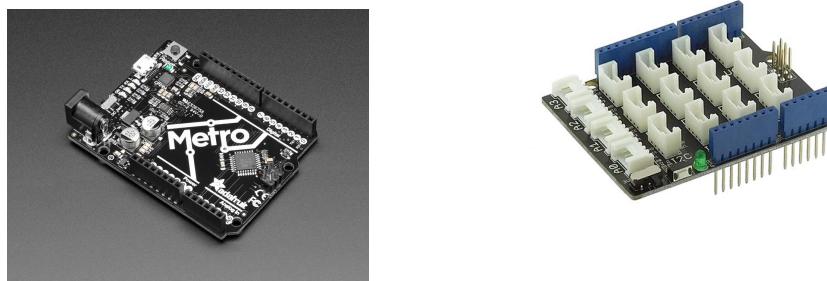
(a) Ultrasonic ranger

(b) Acelerómetro, giroscopio y magnetómetro

Figura 1: Sensores

2.2. Adafruit Metro

Mediante una placa arduino, con una base shield por encima para facilitar la conexión de dispositivos, se ha implementado un código que recoja los valores medidos por los sensores y los pase al puerto serial de la Raspberry.



(a) Metro

(b) Base Shield

Figura 2: Adafruit

2.3. Cámara web

Acoplamiento de una webcam para la realización de streaming de la visión del barco en cada momento, pudiendo obtener imágenes con una calidad de 720p a 30 fps.



Figura 3: Logitech C920

2.4. USB detector

Mediante una conexión usb controlaremos el consumo del sistema para evitar quedarnos sin suministro eléctrico.



Figura 4: USB

2.5. Módem USB

Haciendo uso de la tecnología LTE, podremos hacer conexiones de cliente a servidor, para la futura transmisión de datos.



Figura 5: LTE USB Huawei E3372

2.6. Raspberry Pi 3 Modelo B

Se ha usado una raspberry como cliente del servicio, la cual está almacenando y enviando los datos recogidos por los sensores, cámara y el USB detector a un servidor.



Figura 6: Raspberry Pi 3 B

2.7. Streaming

Como herramientas para poder transmitir el streaming, se han probado Motion y MJPG-Streamer. Motion está disponible en los repositorios de Ubuntu y MJPG-Streamer tiene su repositorio propio en GitHub. Nos hemos decantado por la segunda opción porque proporcionaba una mejor calidad que la primera. Se han instalado las herramientas que pedía en las instrucciones y se ha compilado todo. Este programa funciona ejecutándolo por línea de comandos metiendo los argumentos de output e input correctos. Se ha introducido una función en el código Python corriendo en la Raspberry para que se ejecute de forma automática cuando se inicia barco.py. Esta aplicación se ejecuta en el puerto 8080 de la Raspberry y se mapea por el 9090 en servidor, como se explicará en la siguiente sección.

2.8. Servidor

Para solventar los problemas de direccionamiento IP en redes LTE y también debido a que muchas operadoras no dejan alojar servidores en sus redes, se ha decidido usar un tercer equipo conectado a una red doméstica como servidor. Este equipo estará asociado a un nombre de dominio DDNS para solventar también los problemas del cambio de direcciones. No-IP permite asociar una ip a un nombre de dominio, además dispone de un software para refrescar la IP en caso de que cambie. En este servidor alojaremos el servidor Mosquitto, donde el barco publicará en los topics seleccionados y el puesto tierra se suscribirá a ellos. También realizaremos un mapeado de puertos para la visualización del streaming. Para este mapeado usaremos un servidor ssh instalado también en este equipo. Al acceder a barquitodepapel.no-ip.org:9090 estaremos accediendo realmente al puerto 8080 de la Raspberry, que es donde se está ejecutando el programa de la cámara.

2.9. Interfaz Gráfica

En el servidor se ha instalado Node-Red, una herramienta para aplicaciones IoT de este tipo. Su sencillez a la hora de programar con nodos entradas y salidas de MQTT, elementos del dashboard como botones o indicadores para los valores numéricos entregados por MQTT, etc. hace que sea la herramienta idónea para nuestro barco. Los nodos de MQTT vienen por defecto en Node-Red, pero para los elementos del dashboard, los nodos usados para manipular datos en la DB y para mostrar los datos de la DB mediante una tabla, se han instalado varias herramientas adicionales (dashboard, mongodb4, ui-table). En resumen, con estas herramientas hemos creado dos páginas, una donde se muestran los datos actuales de los sensores, el streaming y dos botones, uno para borrar la DB y otro para parar el barco; y otra para mostrar el histórico de datos. Además se han definido ciertos umbrales para alertar de valores peligrosos y sacarlos a modo de alerta en el dashboard.

2.10. Estación tierra

Como cliente se ha decidido usar un ordenador particular de un miembro del grupo para establecer conexión con el servidor cuando este lo solicite y poder acceder a los datos.

3. Tecnologías probadas

Se han estudiado distintas tecnologías posibles para la implementación del proyecto en nuestro escenario.

3.1. LTE

Es un estándar de comunicación el cual su evolución se ha producido en las redes 4G, sin embargo los datos de LTE se transfieren más rápido y con menor latencia. La conectividad LTE está disponible casi universalmente y proporcionará una continuidad a la red a largo plazo (no como las redes 2G y 3G que están desapareciendo), esto permite que en zonas donde las redes 5G no estén disponibles por un tiempo, las redes tecnología 4G LTE puede suplir esa migración.

Características:

LTE, más concretamente LTE-M o Cat M1, proporciona muy buenas velocidades de subida y bajada, de entre unos 100-300 Kbps, además de contar con un gran alcance y bajo consumo, permitiendo a los dispositivos con batería a durar hasta 10 años. Todo esto con una latencia bastante baja, que permite que esta tecnología sea viable en seguimiento de vehículos por ejemplo. Sin embargo, no está pensada para la transmisión de vídeo y audio debido a su tasa de bits.

Cat M1

- Usa canales 1.4 MHz en lugar de 20 MHz en LTE, por lo que proporciona una buena penetración de la señal a través de obstáculos
- Velocidades de subida y bajada esperadas 100-300 Kbps
- Latencia de 10 - 15 ms
- Una duración de la batería de hasta 10 años
- Cuenta con un bajo coste, entre 10 y 15 euros por módulo
- Orientado a dispositivos IoT

3.2. NB-IoT

Es una tecnología estándar abierta basada en LTE, centrada en conectar objetos cotidianos a Internet, que no necesiten transmitir grandes volúmenes de datos.

Entre los objetivos de esta tecnología encontramos:

- Proporcionar conexiones masivas con un bajo consumo de energía (NB-IoT surge en respuesta al auge de Low Power WAN).
- Integrarse y aprovechar su potencial de redes de telefonía móvil.
- Ofrecer amplia cobertura.

Características:

- Menor consumo energético que LTE.
- Menor complejidad, permitiendo plug-and-play
- Conexiones masivas por celda (+50000)
- Proporcionar mayor alcance en interiores y exteriores.
- Bajo costo de adquisición de componentes.
- Transferencia de información optimizada.
- Seguridad a nivel de autenticación y cifrado de los datos.
- Aprovecha el espectro de GSM y LTE.

En cuanto al pico de data rate, este no supera los 100kbps, contando también con una latencia de entre 1.5-10 segundos. Tampoco soporta voz ni movilidad.

3.3. WiFi

Es una tecnología que permite la interconexión inalámbrica de dispositivos electrónicos. Dichos dispositivos pueden conectarse entre sí o a Internet a través de un punto de acceso (AP).

Características:

- Alcance de unos 100 metros.
- Es una tecnología rápida, más aún cuanto más próximo estemos del AP.
- Tienen un límite de acceso.
- Se debe introducir una contraseña en el dispositivo que queremos conectar.

4. Protocolos probados

Se han probado varios protocolos de transporte diferentes a lo largo del desarrollo:

4.1. UDP

Abreviatura de User Datagram Protocol. Hace uso del protocolo TCP/IP. En las comunicaciones que utilizan UDP, un programa cliente envía un paquete a un servidor de destino el cual también funciona con UDP.

Propiedades:

- No garantiza la entrega de los paquetes. Se considera un protocolo poco fiable por esto.
- No hay flujo de datos entre un servidor UDP y un cliente UDP.
- Un cliente UDP puede enviar "n" paquetes distintos a un servidor UDP y también podría recibir "n" paquetes distintos como respuestas del servidor UDP.
- Dado que UDP es un protocolo sin conexión, la sobrecarga de UDP es menor en comparación con un protocolo basado en la conexión como TCP.

4.2. TCP

El Protocolo TCP/IP o Transfer Control Protocol consiste en un acuerdo estandarizado sobre el que se realiza la transmisión de datos entre los participantes de una red. A través de este protocolo se asegura que los datos lleguen a su destino en el mismo orden que se transfirieron y sin errores. Características:

- Funciona mediante la conexión mutua entre cliente y servidor.
- Ordena los segmentos provenientes del protocolo IP.
- Monitorea el flujo de los datos y permite evitar la saturación de la red.
- Entrega los datos al protocolo IP en forma de segmentos de longitud variable.
- Permite circular de forma simultánea a la información proveniente de diferentes fuentes.

4.3. HTTP

Conjunto de protocolos basados en la arquitectura cliente-servidor. Funciona como un protocolo de petición-respuesta entre el cliente y un servidor. Para solicitar una respuesta del servidor, existen 2 métodos:

- GET: para solicitar datos al servidor.
- POST: para enviar los datos a procesar al servidor.

Características:

- Pensado y desarrollado para ser leído y fácilmente interpretado por las personas, haciendo más fácil la depuración de errores y reduciendo la curva de aprendizaje.
- Es un protocolo sin estado (no guarda ningún dato entre dos peticiones en la misma sesión)
- Una conexión se gestiona al nivel de la capa de transporte, y por tanto queda fuera del alcance del protocolo HTTP.

4.4. MQTT

Es un protocolo de publicación/suscripción que facilita la comunicación entre un gran número de dispositivos y los gestiona mediante brokers. Los clientes publican los mensajes hacia un topic dentro del broker y/o se suscriben a un topic dentro del broker. MQTT es el protocolo ideal para aplicaciones IoT. Ocupa muy poco ancho de banda, tiene QoS configurable, acepta SSL aunque estoreste rendimiento y ancho de banda...

5. Evaluación de prestaciones

La evaluación de prestaciones se divide en el **Plan de Evaluación de Prestaciones** y en el **Informe de Prestaciones**.

En ellos se tendrán en cuenta el siguiente listado de parámetros:

- Latencia, Batería/Consumo, Ancho de banda, Bit rate, Calidad del canal, Alcance, Potencias, Coste monetario, Jitter, Pérdidas, Seguridad, Frecuencias de trabajo (banda libre o licenciada), Facilidad de instalación.

5.1. Plan Evaluación de Prestaciones

5.1.1. Latencia

- Descripción: tiempo que transcurre entre que el servidor genera unos valores aleatorios y el cliente los recibe.
- Motivo inclusión: es de mucha importancia saber los valores de los diferentes sensores, ya sea para saber si tenemos algún obstáculo cerca, para saber la posición del barco y si es necesario estabilizarlo... en un tiempo relativamente pequeño para poder reaccionar.
- Herramienta para llevar a cabo la medición: se ha utilizado Python, en el propio programa de conexión entre cliente y servidor mediante sockets, se recoge el momento en el que el servidor envía los datos y en el que el cliente los recibe.
- Descripción de las pruebas para llevar a cabo las medidas: ejecución del programa mencionado anteriormente múltiples veces y desde diversos dispositivos para ver las posibles variaciones.
- Rango de valores para los cuales el resultado de evaluación sería positivo: para los parámetros de mayor importancia, nuestro caso, sería aceptable una latencia de menos de 1 segundo.

5.1.2. Jitter

- Descripción: desincronización del servidor con el cliente.
- Motivo inclusión: saber el retardo entre barco y tierra es de suma importancia para tomar decisiones para su control y tenerlo en cuenta para la información que se intercambia.
- Herramienta para llevar a cabo la medición: wireshark, veremos la diferencia de retardo entre diferentes muestras.
- Descripción de las pruebas para llevar a cabo las medidas: medición del tiempo de retardo usando el streaming por webcam.
- Rango de valores para los cuales el resultado de evaluación sería positivo: En general, no debe superar los 30 ms como máximo en una dirección para evitar una mala calidad del envío de datos.

5.1.3. Batería/Consumo

- Descripción: mediciones de la autonomía de la batería del bote solar y su consumo medio para determinar la duración de la misma.
- Motivo inclusión: es fundamental conocer la duración de nuestra batería para poder asegurar que aguante durante un tiempo deseado.
- Herramienta para llevar a cabo la medición: se usará un medidor de batería el cual conectaremos a la Raspberry/Arduino.
- Descripción de las pruebas para llevar a cabo las medidas:
 - Ver cuánta batería es capaz de obtener de forma autónoma, velocidad de carga.
 - Realizar maniobras con el barco, comprobar su consumo y cuánto duraría.
 - Ver cuanto consume la Raspberry en activo. (y cuanto consumen los diferentes sensores).
- Rango de valores para los cuales el resultado de evaluación sería positivo:
 - Duración de la batería: entre 30 minutos y 1 hora (depende del clima)
 - Velocidad de carga: alrededor de una hora
 - Consumo medio de los diferentes componentes: La Raspberry Pi consume 700 mA bajo 5 voltios, es decir 3,5 W. Arduino UNO 46mA bajo 5 voltios, es decir 2,3 W. A esto hay que sumarle el consumo medio de los diferentes sensores.

5.1.4. Ancho de banda / Bit Rate

- Descripción: ancho de banda que tendremos disponible para las comunicaciones entre la estación de tierra y el barco y la tasa de datos que pueden ser procesados en un período de tiempo.
- Motivo inclusión: importancia de saber su valor para poder calcular y tomar decisiones en lo que concierne el intercambio de información entre barco y tierra.
- Herramienta para llevar a cabo la medición: usaremos una herramienta software, por ejemplo, speed test en Python, que permita visualizar los parámetros de la interfaz correspondiente.
- Descripción de las pruebas para llevar a cabo las medidas: Medición del ancho de banda en diferentes entornos (solo un usuario comunicándose, varios, con la red congestionada).
- Rango de valores para los cuales el resultado de evaluación sería positivo: El ancho de banda mínimo para poder transmitir streaming por webcam es 2,5 Mbps, a esto habría que sumarle el ancho de banda necesario para retransmitir los datos del resto de sensores. Suponemos que con 5 Mbps estaría solventado el problema.

5.1.5. Calidad del canal

- Descripción: comprobar si el canal por el cual nos vamos a conectar tiene interferencias con otras redes.
- Motivo inclusión: elegir un canal el cual no esté saturado o cuente con interferencias, minimizará las pérdidas, conseguirá mejores resultados respecto a la velocidad a la que se transmite la información y aportará mayor solidez.
- Herramienta para llevar a cabo la medición: emplearemos una aplicación Android llamada Wifi Analyzer.
- Descripción de las pruebas para llevar a cabo las medidas: Comprobación de la calidad del canal mediante herramientas software (aplicación en Android, por ejemplo) en diferentes escenarios.
- Rango de valores para los cuales el resultado de evaluación sería positivo: comprobar los distintos factores para los que el canal es bueno (lo concurrido que esté ese canal y las interferencias en el mismo).

5.1.6. Alcance de la señal / Potencia

- Descripción: distancia hasta la cual llega la señal entre el barco y la estación en tierra.
- Motivo inclusión: necesario conocer el alcance de nuestra señal para limitar la distancia hasta la cual podremos ir con el barco para no poderle transmitir información.
- Herramienta para llevar a cabo la medición: la app Wifi Analyzer o midiendo la intensidad desde un terminal Linux.
- Descripción de las pruebas para llevar a cabo las medidas: medir la intensidad de la señal según nos vayamos alejando.
- Rango de valores para los cuales el resultado de evaluación sería positivo: En exteriores la banda 2.4 GHz llega hasta los 90 metros. A partir de ahí se necesitarían repetidores Wifi.

5.1.7. Coste monetario

- Descripción: coste del hardware del barco y su estación en tierra (en caso de requerir algún componente adicional).
- Motivo inclusión: permite conocer los gastos por adelantado reduciendo la posibilidad de sobrepasar el presupuesto total. Además de poder ser un dato importante a futuro para empresas.
- Herramienta para llevar a cabo la medición: búsqueda online de los diferentes componentes en sus respectivos sitios oficiales/tiendas.
- Descripción de las pruebas para llevar a cabo las medidas: comparación en distintos sitios web.
- Rango de valores para los cuales el resultado de evaluación sería positivo: se considera que el coste es aceptable hasta los pocos cientos euros.

5.1.8. Pérdidas

- Descripción: cantidad de paquetes que no llegan a su destino y se pierden
- Motivo inclusión: una cantidad excesiva de pérdidas sería inaceptable para poder controlar e intercambiar información de forma satisfactoria.
- Herramienta para llevar a cabo la medición: software, monitorización a través de código de los ACK's perdidos
- Descripción de las pruebas para llevar a cabo las medidas: medir el porcentaje de los paquetes que se pierden en diferentes escenarios (muchos/pocos dispositivos conectados al mismo punto de acceso, distancia, interferencias).
- Rango de valores para los cuales el resultado de evaluación sería positivo: en torno a un 1 hasta 2,5 por ciento de pérdidas es aceptable para un buen streaming a través de webcam.

5.1.9. Seguridad

- Descripción: lo vulnerable/sensible que es nuestro sistema a sufrir ataques de terceros o que éstos puedan interceptar y comprender nuestros datos.
- Motivo inclusión: necesidad de un prototipo lo suficiente robusto como para resistir ciberataques básicos.
- Herramienta para llevar a cabo la medición: un tercer equipo con Kali Linux sería una buena opción.
- Descripción de las pruebas para llevar a cabo las medidas: simularíamos desde un tercer equipo que somos los atacantes y haríamos diversas pruebas donde se intentaría interceptar, suplantar, tirar la conexión...
- Rango de valores para los cuales el resultado de evaluación sería positivo: un tercero no debería poder ver el contenido de los paquetes por lo que tendríamos que implementar claves para encriptar la información. Pero considero que sería difícil defenderse de casos de suplantación.

5.1.10. Frecuencias de trabajo (banda libre o licenciada)

- Descripción: analizar la frecuencia que estamos empleando
- Motivo inclusión: conocer la frecuencia de trabajo del prototipo para conocer que banda empleamos, alcance que nos proporciona dicha banda, velocidad...
- Herramienta para llevar a cabo la medición: podríamos emplear una aplicación Android llamada Wifi Analyzer o hacerlo por terminal con los comandos iwlist o iwconfig.
- Descripción de las pruebas para llevar a cabo las medidas: comprobar con la aplicación que efectivamente la banda de frecuencias usada es la que debería corresponder a la tecnología empleada
- Rango de valores para los cuales el resultado de evaluación sería positivo: dependerá de la tecnología que usemos. Por ejemplo, para Wi-Fi sería un valor entre los 2.4 y 2.483 GHz

5.1.11. Facilidad de instalación

- Descripción: la rapidez y sencillez con la que un nuevo usuario pueda usar nuestra arquitectura/sistema de comunicación.
- Motivo inclusión: para que un tercero pueda probar/operar el prototipo (si se quisiera llevar al mercado) es importante que este sea intuitivo y fácil de instalar.
- Herramienta para llevar a cabo la medición: tercer equipo que pruebe y valore la dificultad de la instalación.
- Descripción de las pruebas para llevar a cabo las medidas: una vez tengamos un prototipo funcional, probar a instalarlo en un dispositivo en el que no hayamos probado antes.
- Rango de valores para los cuales el resultado de evaluación sería positivo: no debería llevar más de alrededor de una hora.

5.2. Informe de Prestaciones

5.2.1. Latencia

- **SOCKETS TCP:**

De las diferentes veces realizadas la prueba obtenemos los siguientes resultados:

TCP/WiFi																				
Sended	1.66837E+12	1.66834E+12																		
Received	1.66837E+12	1.66834E+12																		
Diff	37.15991211	36.41015625	36.04003906	37.2199707	27.2199707	29.92016602	30.34985352			31.5	32.54003906	33.43994141	35.89013672	36.41015625	36.0400					
Average	34.03 ms																			

Figura 7: Resultados TCP WiFi

- **SOCKETS UDP:**

De las diferentes veces realizadas la prueba obtenemos los siguientes resultados:

UDP/WiFi																				
Sended	1.66837E+12	1.66837																		
Received	1.66837E+12	1.66837																		
Diff	10.30004883	10.39990234	10.51000977	10.61987305	10.04003906	29.52001953	11.17993164	29.39013672	29.54003906	25.0300293	9.929931641	10.17993164	10.6599							
Average	16.07 ms																			

Figura 8: Resultados UDP WiFi

- **UDP / LTE:**

De las diferentes veces realizadas la prueba obtenemos los siguientes resultados:

UDP / LTE																				
Sended	1.66834E+12	1.66834E+12	1.668339,363,441	1.66834E+12	1.66834															
Received	1.66834E+12	1.66834E+12	1.668339,363,238	1.66834E+12	1.66834															
Diff	203.599853	202.7199707		202.9099121	203.0400391	202.6398926	203.4702148	203.5900879	221.9299316	203.7102051	220.8498535	221.0097656	221.1401367	221.27C						
Average	209.35 ms																			

Figura 9: Resultados UDP/LTE

- **NB-IoT (a la nube de hologram):**

De las diferentes veces realizadas la prueba obtenemos los siguientes resultados:

NB-IoT																				
Diff	3169.150635	3053.741455		3024.851074	2959.251709	2959.92749	2959.851318	2958.648193	2959.108398	2959.441406	2959.522949	2958.352295	2959.612793	2959.7						
Average	2984.07 ms																			

Figura 10: Resultados NB-IoT

- GRÁFICA COMPARANDO RESULTADOS:

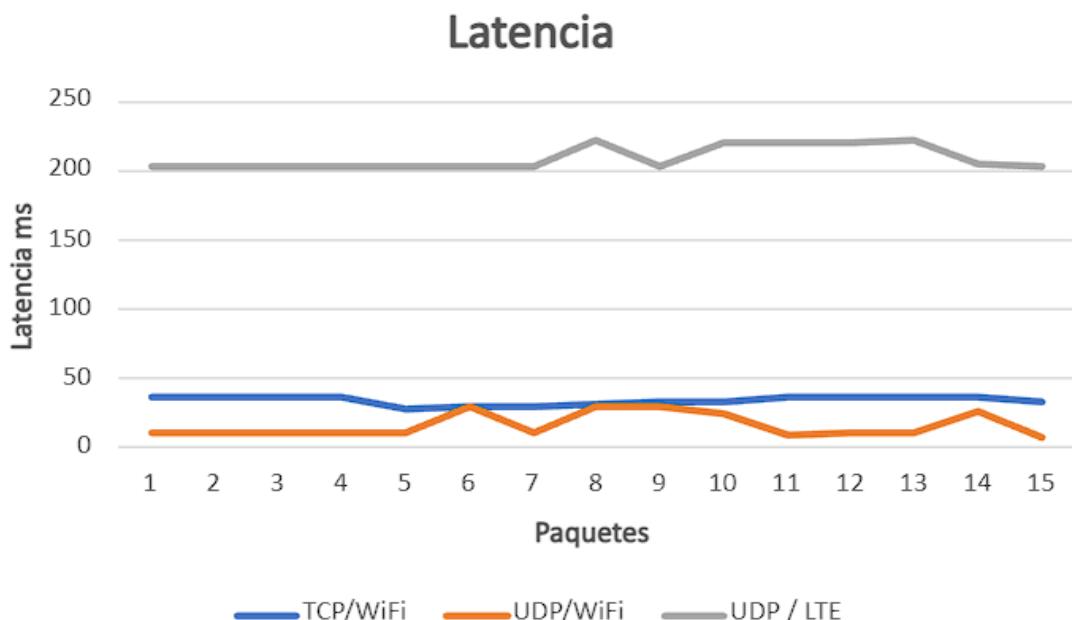


Figura 11: Resultados Gráfica

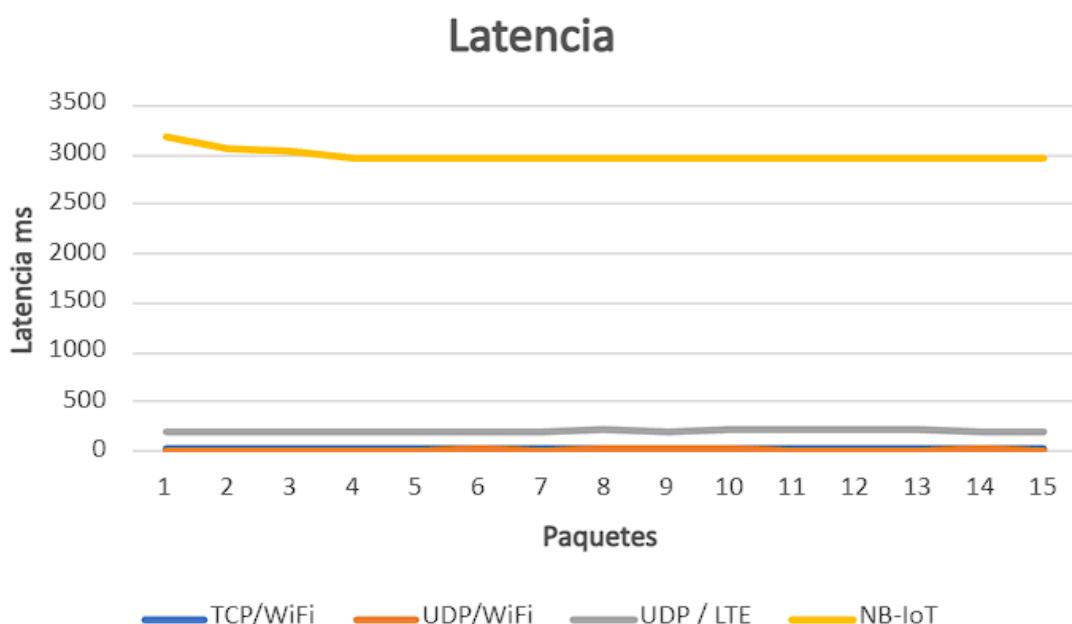


Figura 12: Resultados Gráfica

5.2.2. Jitter

- **SOCKETS TCP:**

De las diferentes veces realizadas la prueba obtenemos los siguientes resultados:

JITTER	TCP/WIFI														
	Diff	0.749756	0.370117	1.179932	10	2.700195	0.429688	1.150146	1.040039	0.899902	2.450195	0.52002	0.370117	1.179932	4.110107
	Average	1.939296	ms												

Figura 13: Resultados Jitter TCP WiFi

- **SOCKETS UDP:**

De las diferentes veces realizadas la prueba obtenemos los siguientes resultados:

JITTER	UDP/WIFI														
	Diff	0.099854	0.110107	0.109863	0.579834	19.47998	18.34009	18.21021	0.149902	4.51001	15.1001	0.25	0.47998	15.09009	17.77002
	Average	7.877145	ms												

Figura 14: Resultados Jitter UDP WiFi

- **UDP / LTE:**

De las diferentes veces realizadas la prueba obtenemos los siguientes resultados:

JITTER	UDP / LTE														
	Diff	0.879883	0.189941	0.130127	0.400146	0.830322	0.119873	18.33984	18.21973	17.13965	0.159912	0.130371	0.129883	16.19995	1.770264
	Average	5.331421	ms												

Figura 15: Resultados Jitter UDP/LTE

- **NB-IoT (a la nube de hologram):**

De las diferentes veces realizadas la prueba obtenemos los siguientes resultados:

JITTER	NB-IoT														
	Diff	115.4092	28.89038	65.59937	0.675781	0.076172	1.203125	0.460205	0.333008	0.081543	1.170654	1.260498	0.132813	0.758301	1.844727
	Average	15.56398	ms												

Figura 16: Resultados Jitter NB-IoT

■ GRÁFICA COMPARANDO RESULTADOS:

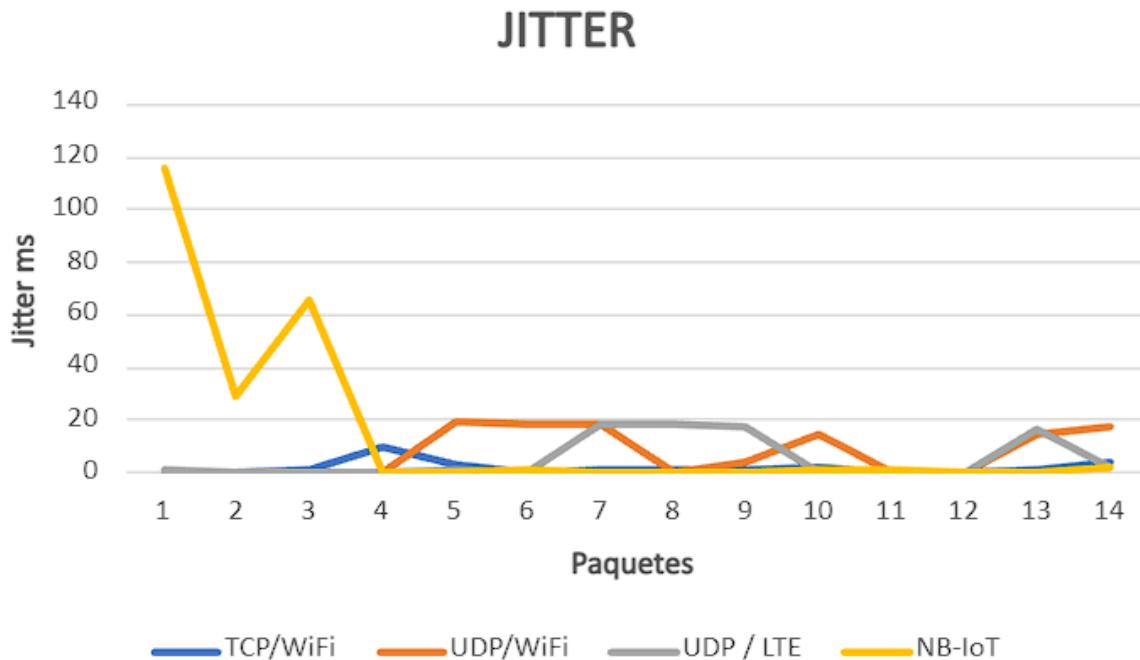


Figura 17: Resultados Gráfica

5.2.3. Batería/Consumo

Dependerá de la tecnología empleada, de momento pensamos que la mejor opción es LTE con MQTT, lo cual tendría un consumo bajo de batería, dependerá de que tipo de LTE elijamos, en el caso de Cat M1 la batería duraría hasta 10 años y con el servidor mosquito de MQTT también consumiría muy poco.

En el caso de utilizar NB-IoT el consumo de batería sería aún menor, pero la latencia de este protocolo y su alta periodicidad entre transmisiones de datos lo hace inviable a la hora de manejar un barco remotamente.

En el caso de usar Wi-Fi el consumo se dispara respecto a los casos anteriores, reduciendo su duración a pocos días (depende del uso de este) en la mayoría de los casos.

Por último, para Bluetooth dependerá si utilizamos o no el modo Low Energy, si lo utilizamos conseguiremos una duración de unos 4 o 5 años y sino su duración será muy similar a la del Wi-Fi.

5.2.4. Ancho de banda / Bit Rate

- Wi-Fi:

```
wlan0      IEEE 802.11 ESSID:"vodafoneBA9412"
Mode:Managed Frequency:2.422 GHz Access Point: 14:2E:5E:CA:84:98
Bit Rate=14.4 Mb/s Tx-Power=31 dBm
Retry short limit:7 RTS thr:off Fragment thr:off
Power Management:on
Link Quality=35/70 Signal level=-75 dBm
Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
Tx excessive retries:1077 Invalid misc:0 Missed beacon:0
```

Figura 18: Ancho de banda y Bit rate Wi-Fi

- Bluetooth
Ancho de banda de unos 2.4GHz.
- Para la tecnología LTE-M se espera un ancho de banda considerablemente menor de 1.4 MHz.
- Para NB-IoT sería de unos 200 KHz.

Nota: MQTT como protocolo está pensado para aplicaciones IoT, por lo que ocuparía poco ancho de banda.

5.2.5. Calidad del canal

Los resultados dependerán del canal a utilizar en ese momento, así como de los dispositivos conectados a esa frecuencia en ese momento. Un ejemplo de la calidad de canal empleada para llevar a cabo las pruebas de mediciones de los diversos parámetros es la de la siguiente figura:

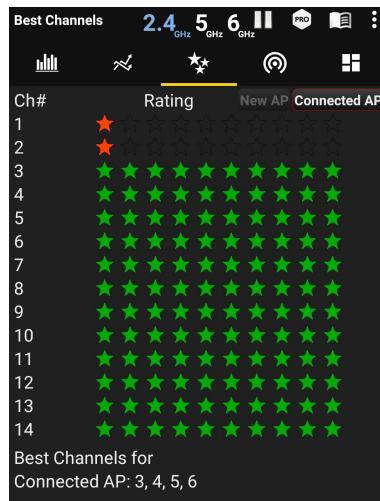


Figura 19:
Calidad del canal empleado según la herramienta Wi-Fi Analyzer
(Nuestro caso utilizando el canal 1)

5.2.6. Alcance de la señal / Potencia

- **Wi-Fi 2.4 GHz:** 45 metros en interiores y 90 metros en exteriores.
- **Bluetooth:** unos 10 metros, puede variar dependiendo de los obstáculos.
- **LTE:** hasta 10-15 km en entornos abiertos y 1.5 km en entornos urbanos.
- **NB-IoT:** hasta 15 km en entornos abiertos y 2 km en entornos urbanos.

5.2.7. Coste monetario

- **LTE:** Bajo coste, 40 euros por módulo.
- **NB-IoT:** Bajo coste, 100Mb 30 euros; 250Mb 62.50 euros; 500Mb 100 euros; 1Gb 175 euros; 2Gb 300 euros; 5Gb 700 euros al mes¹.

5.2.8. Pérdidas

▪ **SOCKETS TCP:**

El protocolo TCP es utilizado por los contenidos que no son de flujo continuo, como cuando se obtienen páginas web o imágenes. TCP está construido sobre UDP y proporciona muchas características adicionales, como la fiabilidad y el orden de los paquetes. Sin embargo, para conseguir estas características adicionales, un paquete TCP típico será más grande que uno UDP, lo que hace que TCP sea más lento y consuma más ancho de banda.

Debido a este mayor consumo y latencia, será preferible tener algunas pérdidas irrecuperables (UDP), a perder tiempo intentando recuperarlas.

▪ **SOCKETS UDP:**

El protocolo UDP es muy sencillo. Se conoce como 'datagrama', y funciona de forma muy parecida a la forma en la que un telegrama transporta mensajes a personas físicas. El paquete UDP contiene información sobre las redes de origen y destino de los datos, lo que permite a los proveedores de servicios de Internet entregar el paquete a su destinatario.

UDP es un protocolo muy ligero y, por lo tanto, es el método de transporte preferido para los contenidos en streaming, como el audio y el vídeo, ya que su uso reduce los requisitos de ancho de banda y la latencia.

Con una cobertura relativamente buena las pérdidas no deberían condicionar el funcionamiento de nuestras comunicaciones, y es preferible tenerlas a cambio de reducir los tiempos de latencia.

▪ **MQTT:**

Al igual que TCP, MQTT tiene la posibilidad de corregir errores.

¹<https://www.zoominformatica.com/huawei-e3372-modem-usb-4g-lte-libre-m150-banda-ancha-ranura-micro-sd.html>

5.2.9. Seguridad

- **SOCKETS TCP y SOCKETS UDP:**

En nuestro caso, al estar construidos de una forma muy similar tendrán la misma seguridad. Debido a que no tenemos un sistema de login cualquier usuario que sepa nuestra ip y se encuentre dentro del rango de cobertura, se podrá conectar a los sockets y recibir/enviar datos. Por lo que un posible ataque sería que alguien con malas intenciones realice muchas conexiones y las deje sin hacer nada. Una solución a este problema sería usar timeouts por inactividad, donde se eche a los usuarios que lleven más de X tiempo sin hacer nada. Otra solución en caso de que esos usuarios si estén haciendo algo sería limitar el número de ellos. Con estas últimas consideraciones conseguiríamos un servicio bastante estable y resistente a ataques o saturación.

- **MQTT:**

En el caso de MQTT se podría activar el SSL, pero esto conllevaría a una pérdida de prestaciones (latencia y ancho de banda). En nuestra aplicación no tendría sentido activar SSL porque queremos conseguir una baja latencia y es poco probable que alguien intente hackearnos.

5.2.10. Frecuencias de trabajo (banda libre o licenciada)

Como podemos ver en la siguiente imagen las bandas licenciadas serían las de LTE y NB-IoT

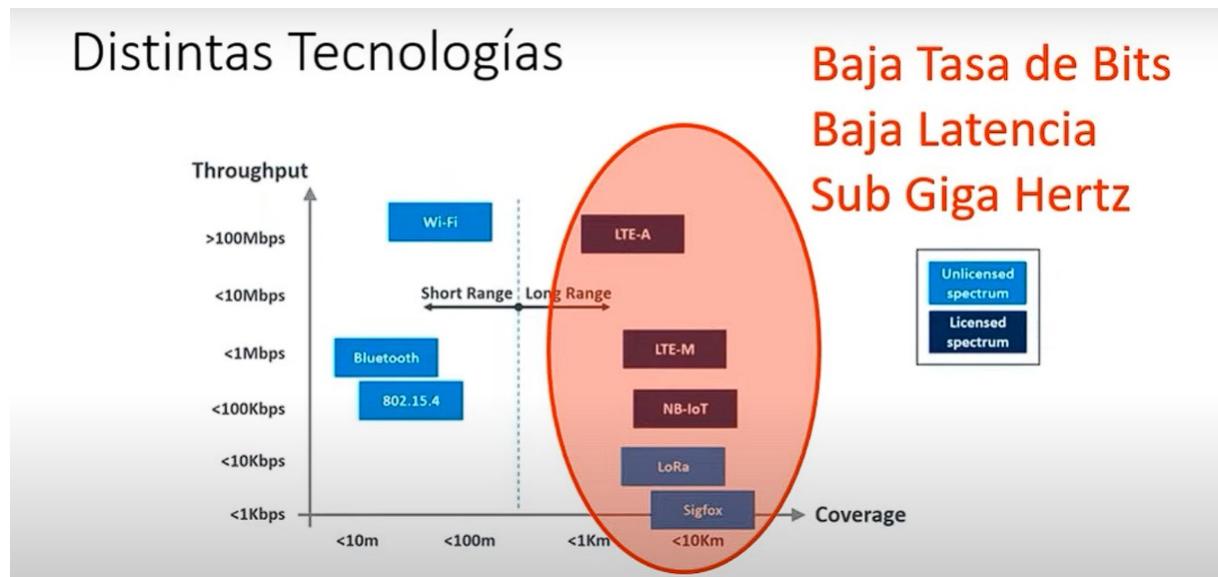


Figura 20: Alcance y tasa de diversas tecnologías

5.2.11. Facilidad de instalación

(NT: Esta sección no tiene como objetivo describir de forma precisa como llevar a cabo la instalación completa, sino que se describirán por encima los puntos de mayor importancia y se estimará el tiempo necesario para llevar a cabo todos ellos.)

Para medir la facilidad que un tercero tiene para instalar e implementar nuestro sistema tendremos en cuenta las siguientes consideraciones:

- El usuario no encontrará problemas adicionales a los mencionados durante la instalación.
- El usuario cuenta con un conocimiento tecnológico medio.
- El usuario cuenta con una guía de cómo llevar a cabo la instalación.

Con estas consideraciones el usuario deberá primero llevar a cabo la parte de montaje del hardware, la cual, en resumen, consistirá en alimentar y conectar la Raspberry, conectar los sensores (arduino) a la misma, la WebCam e introducir el pincho LTE.

Después teniendo en cuenta que la Raspberry del usuario no tiene nuestro código, necesario para las comunicaciones, el usuario deberá bajárselo de la web e introducirlo en la Raspberry. Hasta este punto tendremos montado la parte del barco con sus comunicaciones mediante LTE, ahora nos faltaría montar un servidor MQTT que escuche las transmisiones del barco y configurar nuestra estación en tierra.

En nuestro caso para montar este servidor hemos empleado otro equipo, en el cual al igual que en el barco será necesario bajar nuestro código correspondiente a esta parte y ejecutarlo para poner en marcha el servidor, además será necesario configurar las siguientes partes: envío y almacenamiento de los datos recibidos en una base de datos, y la comunicación con una interfaz que muestra el streaming de la WebCam, los datos recogidos por los sensores del barco y permite interactuar con algunos del mismo.

Por último el usuario deberá configurar la estación de tierra que será desde la cual podremos visualizar las cosas mencionadas anteriormente, para ello tendremos que bajar el código correspondiente a la estación de tierra e implementarlo.

Una vez tengamos ejecutándose estas tres partes: Raspberry, Servidor y Estación de tierra, y comunicándose, podremos dar por finalizada la instalación.

Tiempo total estimado: entre 1h 30min y 2h.

6. Prototipo final

Tras el estudio de las distintas tecnologías y prototipos, y tras realizar una serie de pruebas para evaluar las prestaciones, se ha decidido implementar:

- Como tecnología: **LTE**
- Como protocolo: **MQTT y HTTP**

Descartamos la tecnología NB-IoT por su baja periodicidad de transmisión de datos, ya que esto conlleva a una latencia del rango de 1.5 a 10 segundos. Además, se basa en LTE pero reduciendo al mínimo las funcionalidades del mismo, ya que se centra en lo imprescindible para IoT. Por ello se ha decidido implementar LTE. Además, se ha descartado la tecnología de WiFi por la necesidad de una infraestructura de repetidores repartidos por la ruta que sigue el barco para evitar la pérdida de comunicación.

Como protocolos, los resultados nos han hecho optar por MQTT para los datos de pitch, roll, yaw y distancia con objetos, y HTTP para el streaming de la webcam.

MQTT nos va a permitir tener un topic distinto para cada dato, pudiendo ser capaces de elegir en el servidor qué dato queremos obtener. HTTP permitirá un flujo de datos constante hacia el servidor, con una mayor capacidad en los paquetes que se envían, ya que MQTT hace uso de paquetes más pequeños que afectan directamente al visionada del streaming. Por ende, descartamos tanto TCP como UDP.

- Para la visualización de los datos, tanto en tiempo real como históricos, como el vídeo de la cámara; hemos decidido usar el software **Node-Red** para mostrarlo en una página web, ya que posee una programación gráfica muy fácil de utilizar y es muy potente para este tipo de aplicaciones IoT.

Se han usado los módulos *dashboard* , *mqtt*, *ui-table* y *mongodb*. A través de entregas MQTT se representan gráficamente su valor en elementos del *dashboard*, y con un botón podemos dar la orden a la raspberry para dejar de mostrar datos. Además se ha creado una base de datos en *mongodb atlas* (versión en la nube) para poder guardar los datos, y mediante un nodo tabla se representan en la interfaz gráfica. Tanto para insertar como para leer los datos de la tabla *mongodb* se han usado nodos *mongodb4* conectándolos al servidor y dandole la instrucción *insert* y *find.toArray*, respectivamente.

Para la orden tierra-barco se han usado nodos botones que crean el mensaje que queremos al pulsarlo y se conectan a un nodo *mqtt out* para enviarlos a nuestro servidor mqtt. Además, se ha modificado el código en Python de nuestro barco para que cuando reciba el mensaje de parar se termine el proceso. Para las alertas se han usado nodos funciones para permitir elegir un umbral de valores en el que hacerlas; y también nodos de notificación, que envían mensajes emergentes a la interfaz.