Universidad de Oviedo

Memoria del Trabajo Fin de Máster realizado por

Claudia Serrano Martínez

para la obtención del título de

Máster en Ingeniería de Automatización e Informática Industrial

Mejora del sistema de ayuda a la reducción de riesgos de colisión de buques en alta mar

SEPTIEMBRE 2020

AGRADECIMIENTOS

Quisiera agradecer, en primer lugar, a mis tutores, María de los Reyes Poo Argüelles, por su confianza y apoyo a lo largo de estos meses y, a Felipe Mateos Martín, por sus consejos y su ayuda. Gracias a ellos, ha sido posible realizar este trabajo.

Este trabajo se lo dedico especialmente a mi abuelo que aunque ya no esté aquí estoy segura que estaría muy orgulloso de su nieta y que disfrutaría igual o más que yo de este momento. Y, por último,a mis padres, por apoyarme cada día durante estos dos años y por brindarme, en los momentos malos, la fuerza que necesitaba para continuar. Sin ellos no hubiese sido posible estudiar este máster.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN

Los océanos son uno de los recursos ambientales más valiosos de nuestro planeta ya que cubren alrededor del 70 % de la superficie del mundo, proporcionando materias primas, energía, alimentos, empleo, un lugar para vivir, un lugar para relajarse y los medios para transportar alrededor del 80 % del comercio mundial. Por ello, el transporte marítimo es un usuario clave de los océanos. La ventaja competitiva del transporte marítimo sobre otros modos de transporte es que conduce a un mayor transporte de mercancías por mar. En consecuencia, nos enfrentamos a un aumento continuo de la intensidad del tráfico, el tonelaje y la velocidad que desarrollan algunos buques. Esto afecta negativamente la seguridad de las personas, los barcos, la carga y el ecosistema acuático.

Los informes y análisis sobre accidentes marítimos señalan a las colisiones entre buques como los accidentes de navegación más comunes. Estos abordajes suelen producirse entre dos embarcaciones o entre una embarcación y otros elementos, como pueden ser objetos flotantes. Numerosos estudios indican que las principales causas de estos abordajes suelen tener su origen en un error humano ya sea por falta de supervisión humana (un simple despiste, falta de vigilancia), no incumplimiento de la normativa (reglas COLREGs (*International Regulations for Preventing Collisions at Sea (COLREG, 1972)*) o por el mal entendimiento de algunos dispositivos (mala o nula comunicación entre los OONWs (*Officers in charge Of a Navigational Watch*)).

A lo largo de los años, para mejorar la seguridad y eficiencia de estos servicios de transporte marítimo, los buques están siendo equipados con dispositivos y sistemas cada vez más avanzados. Sin embargo, la creciente cantidad de información disponible a bordo y la mayor complejidad de estos nuevos sistemas implantados provoca que la gestión de la información sea extremadamente difícil, especialmente en la toma de decisiones en caso de situaciones de emergencia.

Entre algunas de las soluciones potencialmente positivas para mejorar la seguridad en la navegación y reducir el riesgo de colisión pueden ser:

- Introducción de herramientas de ayuda a los OONWs que, además de informar sobre la situación marítima puedan ayudar a la detección temprana y toma de decisiones ante situaciones de colisión.
- Establecimiento de comunicación entre los OONWs.

En los últimos años, se han desarrollado diferentes herramientas que tratan de asegurar el transporte marítimo. Entre ellos, se puede destacar el sistema PE (Programable Electrónico) que se presentó en Argüelles et al. (2019) y que se caracteriza por proporcionar a dos buques que estén en riesgo de colisión las instrucciones correctas y coordinadas de las maniobras a realizar tras contrastar la información que ambos tienen sobre la situación, es decir, la toma de decisiones no se produce de forma aislada por cada uno [1],[2]. En función de los datos estáticos y dinámicos recibidos del AIS (*Automatic Identification System*), el sistema PE de cada buque calcula distancia, demora (ángulo entre el Norte y la línea de la visual dirigida a un punto, 0°<=demora<360°), CPA (Closest Point of Approach) y TCPA (Time to CPA) con relación al resto de buques próximos y las maniobras a realizar respecto a cada uno basándose en las reglas. Estos sistemas PE se comunican enviando y recibiendo los datos de las maniobras y las señales de los OONWs, comparan los resultados de los cálculos de cada uno y llegan a decisiones comunes o bien comunican los desacuerdos. Este intercambio de información sobre datos de navegación (estáticos y dinámicos) y mensajes binarios definidos se realiza mediante sentencias o grupos de parámetros PGN (Parameter Group Number) codificados siguiendo estándares NMEA (NMEA 0183,2008; NMEA 2000,2016) [3],[4].

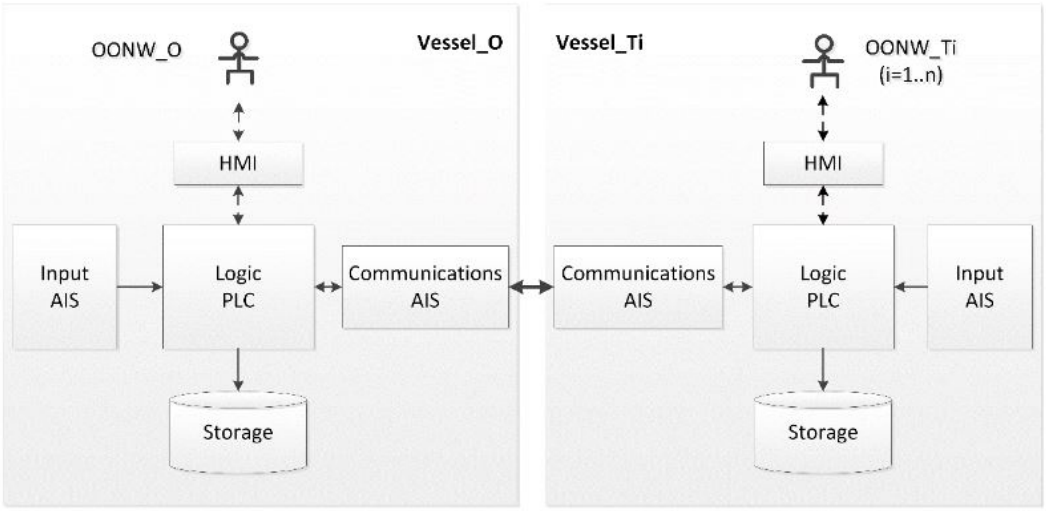


Fig. 1.1. Esquema sistema PE

El prototipo hasta el momento desarrollado, como precedente a este trabajo, ha utilizado dos plataformas software:

- CoDeSys v2.3, herramienta de programación de PLCs según la norma IEC 61131-3. En modo simulación, el programa no se procesa en un PLC real, sino en el ordenador en el que se está ejecutando la aplicación CoDeSys. Se puede así probar la lógica del programa sin disponer de unos PLCs concretos, y permite evitar inicialmente las complicaciones

derivadas de las configuraciones del hardware. Se ha creado también con CoDeSys un HMI (interfaz entre el operador y el sistema PE) básico.

- PCWorX v6.30, también programable según IEC 61131. Esta implementación se ha chequeado con PLCs reales, ILC130 de Phoenix Contact. En este caso, el HMI básico ha sido desarrollado con el editor WebVisit v6.21, software suministrado también por Phoenix Contact.

El prototipo incluye la programación de la simulación del movimiento de los buques y de simulación de las comunicaciones: bloques funcionales para la lectura y escritura de la información transmitida por el AIS.

Los objetivos planteados para este TFM incluyen mejoras del prototipo enfocadas en tres aspectos:

- Implementación de un interfaz con el OONW más completo.
- Programación de las comunicaciones AIS (*Automatic Identification System*) reales.
- Creación de un programa para PC de lectura y presentación de la información almacenada por en PLC en la base de datos.

1.1. Objetivos del proyecto

De forma más precisa, con el fin de completar y mejorar el prototipo desarrollado se va a desarrollar una nueva versión del sistema con las siguientes mejoras:

- **Uso de un nuevo PLC real (*PLCnext*):** uno de los objetivos de este proyecto es implementar este prototipo en un PLC de nueva generación, *PLCnext* de Phoenix Contact, que aporta más versatilidad y nuevas características que permitirán comprobar las comunicaciones con los sistemas AIS. Por ello, se ha optado por utilizar ***PLCnext Control AXC F 2152***.
- **Programación sistema:** en la versión actual la programación se ha realizado cumpliendo IEC 61131-3, en Codesys V2.3 y en PcWorx v6.30. En esta nueva versión la programación se realizará en PLCnext Engineer 2020.0 cumpliendo igualmente el estándar IEC 61131-3.
- **Comunicaciones:** en la versión actual las comunicaciones AIS están simuladas, unos bloques funcionales simulan el movimiento del buque y la lectura/ escritura de la información que se transmite vía AIS, mensajes 1, 5, 6 y 7 encapsulados según describe la

UIT (*Unión Internacional de Telecomunicaciones*). En esta nueva versión se integrarán señales de AIS reales para probar las comunicaciones NMEA entre un AIS y un PLC, concretamente se utilizará la herramienta de simulación *AIS Simulator Version 8.04a - October 2019*.

- **HMI:** en la versión actual existe un HMI básico con versiones Codesys y WebVisit por lo que se procederá a desarrollar una interfaz de usuario más atractiva con información relevante que sirva de ayuda para la navegación marítima. Se ha optado por desarrollar una interfaz gráfica web dinámica mediante Javascript y HMTL ya que estos lenguajes permiten crear contenido de actualización dinámica, control multimedia, animar imágenes, etc...Y, además, es compatible con el servidor Python desarrollado en el proyecto para la trasnferencia de los datos obtenidos de las estaciones AIS.
- **Base de datos:** En la nueva versión del sistema se desarrollará un programa para PC de lectura y presentación de la información almacenada por el PLC. En este caso, se ha utilizado *MySQL Workbench* que permite las siguientes funcionalidades:
  - Escritura periódica de los datos dinámicos recibidos y calculados.
  - Escritura de los valores, situaciones y escenarios cuando el sistema entra en prealerta.
  - Escritura de los mensajes intercambiados con el operador propio y con los demás buques.
  - Escritura de los mensajes al operador en caso de desacuerdos, en caso de que uno de los implicados en el diálogo no responda en un tiempo dado, o que falle la comunicación AIS.

1.2. Organización de la memoria

En consecuencia, la memoria se ha estructurado de la siguiente manera:

- **Capítulo 1: Introducción**
  Incluye una breve introducción del tema sobre el que trata el proyecto, indicando el fundamento y los objetivos a alcanzar.
- **Capítulo 2: Comunicaciones**
  En este capítulo se explica detalladamente las comunicaciones establecidas entre los diferentes componentes utilizados en el proyecto. Además, incluye una descripción paso a paso sobre cómo se ha realizado la obtención de los datos de cada buque para las diferentes sentencias NMEA a partir de mensajes codificados y, cómo se han realizado los cálculos de distancia, demora, velocidad relativa, rumbo relativo, CPA y TCPA a partir de dichos datos.

- **Capítulo 3: HMI**
  Se describen las herramientas utilizadas para el desarrollo de la interfaz de usuario y cómo se han utilizado para obtener un resultado final mucho más intuitivo y atractivo para el usuario. Se muestra el resultado final de dicha interfaz gráfica con sus funcionalidades principales y, se añaden ejemplos ante diferentes situaciones que pueden ocurrir para mostrar su funcionamiento real.

- **Capítulo 4: Base de datos**
  Se describe la estructura de la base de datos creada, tipo de información a almacenar y condiciones que deben darse para almacenar dicha información (situaciones de alerta/-prealerta). Además, se explica brevemente el programa para PC utilizado para la lectura y presentación de la información almacenada. Y, se añaden ejemplos para mostrar su funcionamiento ante algún caso real.

- **Capítulo 5: Planificación y presupuesto**
  Descripción detallada de las diferentes tareas que se han llevado a cabo durante el desarrollo del proyecto y, presupuesto para la realización del mismo.

- **Capítulo 6: Conclusiones y trabajos futuros**
  Incluye las principales conclusiones obtenidas tras la realización de este proyecto y, posibles mejoras que podrían plantearse.

2. COMUNICACIONES

Uno de los objetivos es utilizar el PLC real *PLCnext Control AXC F 2152* montado en el bastidor que se muestra en la Figura 2.1.

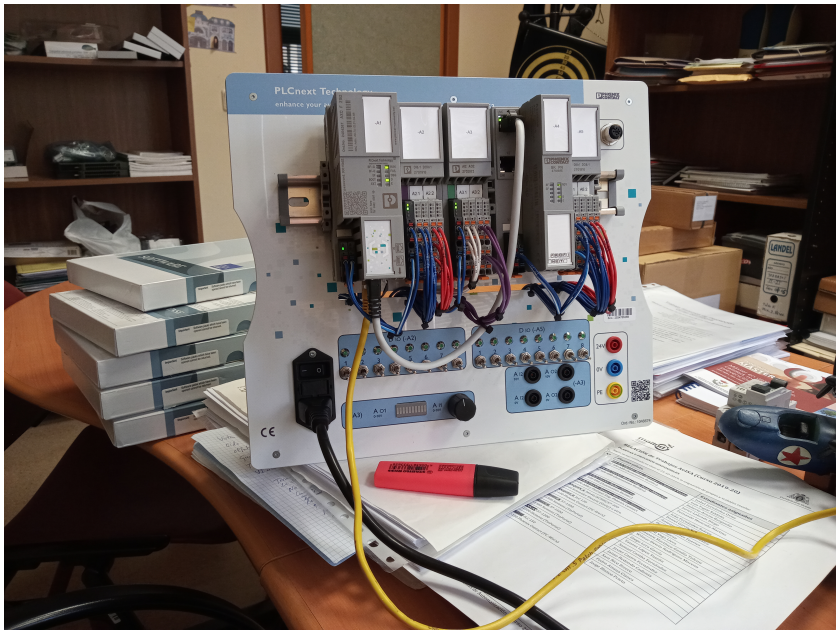


Fig. 2.1. PLCnext Control AXC F 2152.

Este kit de Inicio PLC CPU Phoenix Contact PLCnext proporcionado por el *Departamento Ingeniería Eléctrica, Electrónica, de Computadores y Sistemas* de la Universidad de Oviedo presenta los siguientes módulos:

- **PLCnext Control AXC F 2152:** que presenta las siguientes características [6]:
  - Soporte PROFINET para la comunicación de datos a través de Ethernet Industrial, principalmente utilizado para recopilar datos y controlar equipos en sistemas industriales.
  - Conexión a PROFICLOUD. Proficloud de Phoenix Contact es un sistema IoT abierto que permite crear soluciones adaptadas a cada aplicación. Puede recopilar y codificar los datos de sensores y/o procesos para transferirlos a las aplicaciones en la nube que continúan procesando estos datos o para implementar aplicaciones como Big Data, detección de modelos y Condition Monitoring.
  - Compatibilidad con numerosos protocolos como http, https, FTP, OPC UA, SNTP, SNMP, SMTP, SQL, MySQL, DCP, etc.

- Conexión a la PLCnext Store que, como adaptación al mundo IoT, facilita la descarga directa de nuevas funcionalidades de software que permite una fácil adaptación a los cambiantes requisitos y un uso eficiente del software (ingeniería modular).
- Hasta 63 módulos de E/S AXIO alineables directamente.
- 2x interfaces Ethernet (interruptor integrado).
- Resistencia CEM aumentada Margen de temperatura ampliado de -25...60 ^{o}C.
- Sistema operativo Linux.
- Compatibilidad con lenguajes estándar de alto nivel (diagrama de Bloques de Funciones (FBD), texto Estructurado (ST), etc...).

- **Módulo E/S - AXL F DI8/1 DO8/1 1H:** Axioline F, Módulo de entrada/salida digital, Entradas digitales: 8, 24 V DC, técnica de conexión: 1 conductor, Salidas digitales: 8, 24 V DC, 500 mA, técnica de conexión: 1 conductor, velocidad de transmisión en el bus local: 100 MBit/s, índice de protección: IP20, incluido módulo de zócalo de bus y conectores Axioline F [7]. Se utiliza para el registro y la salida de señales digitales.

- **Módulo E/S - AXL F AI2 AO2 1H:** Axioline F, Módulo de entrada/salida analógica, Entradas analógicas: 2, 0 V ... 5 V, -5 V ... 5 V, 0 V ... 10 V, -10 V ... 10 V, 0 mA ... 20 mA, 4 mA ... 20 mA, -20 mA ... 20 mA, técnica de conexión: 2 conductores, Salidas analógicas: 2, 0 V ... 5 V, -5 V ... 5 V, 0 V ... 10 V, -10 V ... 10 V, 0 mA ... 20 mA, 4 mA ... 20 mA, -20 mA ... 20 mA, técnica de conexión: 2 conductores, velocidad de transmisión en el bus local: 100 MBit/s, índice de protección: IP20, incluido módulo de zócalo de bus y conectores Axioline F [8]. Sirve para registrar y emitir señales de corriente y tensión analógicas.

- **Módulo de comunicación - AXL F RS UNI 1H:** xioline F, Módulo de comunicación, interfaz: RS-232, RS-485, RS-422, velocidad de transmisión en el bus local: 100 MBit/s, índice de protección: IP20, incluido módulo de zócalo de bus y conectores Axioline F [9]. Posibilita el funcionamiento de aparatos periféricos de tipo comercial con interfaz en serie en un sistema bus.

- **Acoplador del bus - AXL F BK PN:** Axioline F, Acoplador de bus, PROFINET, Hembra RJ45, velocidad de transmisión en el bus local: 100 MBit/s, índice de protección: IP20, incluidos módulo base de bus y conector Axioline F [10]. El acoplador de bus está previsto para el uso dentro de una red PROFINET.

Por esta razón, para abordar este proyecto se han llevado a cabo diferentes comunicaciones que se resumen en el siguiente esquema:

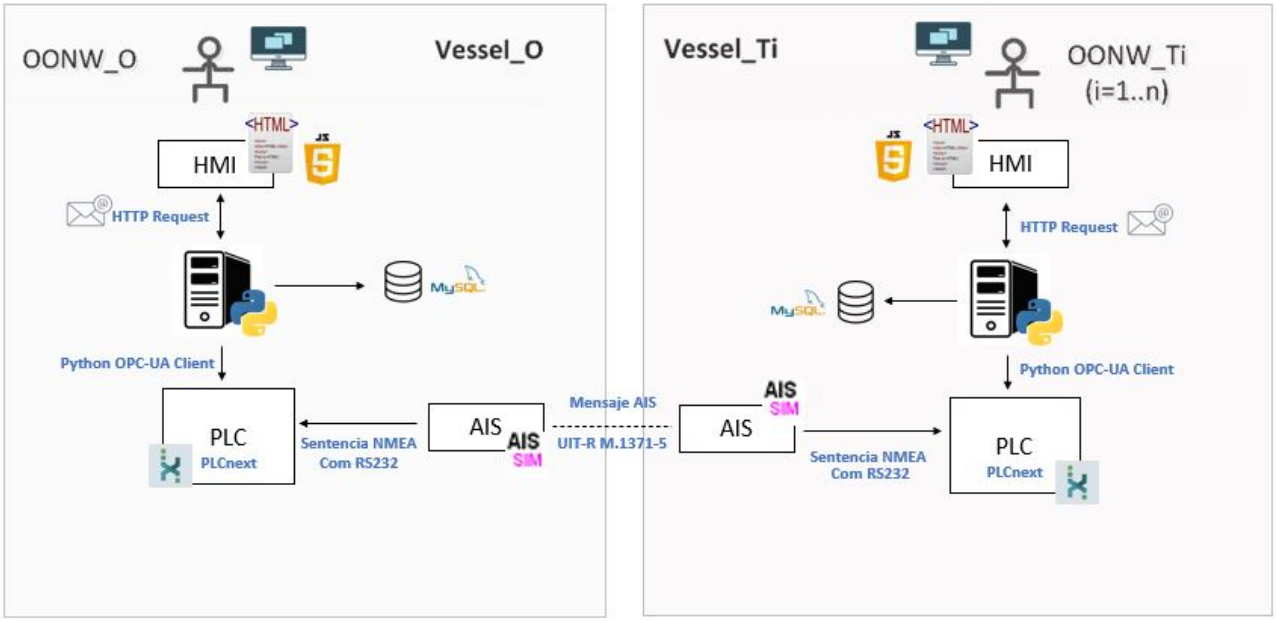


Fig. 2.2. Esquema comunicaciones implementado.

Para la parte gráfica se ha optado por desarrollar una interfaz gráfica con HTML y Javascript. Con el fin de permitir la comunicación entre el PLC y el HMI se ha desarrollado un servidor Python permitiendo así realizar peticiones HTTP con Javascript para la obtención de la información y, que a su vez permite almacenar información relevante mediante mySQL.

Por otra parte, para permitir la comunicación entre PLCnext y el servidor Python se ha implementado en el servidor un cliente OPC-UA para el intercambio de información entre ambos ya que en el PLC se ha configurado un servidor OPC-UA. Y, por último, para la comunicación entre PLCnext y la estación AIS se ha optado por establecer una comunicación serie RS232. A continuación, se explicará más detalladamente cada una de estas comunicaciones:

2.1. Comunicación AIS - PLC

La estación AIS (*Automatic Identification System*), es un sistema de comunicaciones capaz de intercambiar información sobre identificación, posición, rumbo, velocidad y otros datos entre las estaciones costeras y los buques [11]. El objetivo fundamental de este sistema es permitir a los buques comunicar su posición y otras informaciones relevantes al resto de buques o estaciones para que puedan conocerla y evitar colisiones. Proporciona al usuario la siguiente información:

- **Información dinámica:** información relativa a la posición del buque y la actividad que

está realizando en cada momento (la posición, la velocidad, el rumbo, el ratio de giro...).

- **Información estática:** información descriptiva del buque (tipo de embarcación, puerto de destino y hora de arribada al puerto destino, plan de ruta...).
- **Información del viaje:** información relativa al origen y destino del barco, así como la carga que lleva a bordo.
- **Recepción de mensajes binarios:** se trata de los mensajes AIS tipo 6 que son los mensajes que intercambia el operador propio con el resto de buques.

Existen dos tipos de AIS:

- **AIS Clase A:** son los que están obligados a llevar todos los buques afectados por la normativa del convenio SOLAS (*Safety of Life at Sea*) cuyo objetivo es establecer normas mínimas relativas a la construcción, el equipo y la utilización de los buques, compatibles con su seguridad [12]. Este sistema lo utilizan especialmente grandes embarcaciones o embarcaciones de gran tonelaje y se trata de un sistema completo y pesado que permite enviar y recibir una información estática, dinámica y relativa al viaje mucho más completa.
- **AIS Clase B:** son los utilizados por los buques que no están obligados por las normas SOLAS a llevar el sistema AIS a bordo, principalmente en la navegación de recreo o de pequeño tonelaje. Se trata de un sistema más ligero y económicamente asequible que tan solo emite la posición aunque algunos modelos pueden enviar información complementaria.

También existe un sistema **AIS Clase B+** que, a diferencia del AIS Clase B, utiliza la misma tecnología SOTDMA que la de los transpondedores AIS de Clase A y se caracteriza por ser capaz de modificar automáticamente las velocidades de transmisión en función de la velocidad. A medida que aumenta la velocidad de un barco, aumenta el número de transmisiones, de modo que otros barcos tienen una visión más clara y más actualizada de la posición del barco. Este tipo de sistema se suele utilizar en embarcaciones rápidas [13].

TABLA 2.1. TIPOS AIS.

Función	**Clase A**	**Clase B**	**Clase B+**
Potencia de transmisión	12.5W	2W	5W
Velocidad de transmisión (segundos)	2-3	30	5
Tecnología	SOTDMA	CSTDMA	SOTDMA
Datos del viaje	SI	NO	NO
Conexión de GPS externa	SI	NO	NO
Distancia transmisión (millas náuticas)	20-25	7-8	10-12
Interfaces digitales	2 entradas-salidas	Opcional	Opcional
Tipo mensajería específica	Transmisión Recepción	Transmisión Recepción	Recepción opcional No transmisión
Tipos datos transmitidos	Todo	No se muestra: Velocidad giro Estado navegación Destino ETA	No se muestra: Velocidad giro Estado navegación Destino ETA
Norma de certificación de la comisión electrotécnica internacional (IEC)	IEC 61993-2	IEC 62287-2	IEC 62287-1

En este proyecto, se ha optado por simular señales de sistemas AIS Clase A como se explicará más adelante. Una de las principales diferencias y que han permitido decantarse por el uso de estos sistemas de Clase A, es que proporcionan información mucho más completa que los de Clase B permitiendo así desarrollar un prototipo más óptimo. Además, el software utilizado para la simulación de estas señales, *AIS Simulator*, esta desarrollado para este tipo de estación.

Todo sistema AIS opera en la banda VHF del servicio móvil marítimo por lo que es capaz de identificar otros barcos detrás de obstáculos que con el uso del radar no se verían y, con ello, estar prevenidos antes de un contacto visual. Además, está basado en la tecnología TDMA (*Time- Division Multiple Access*) como se muestra en la Figura 2.3

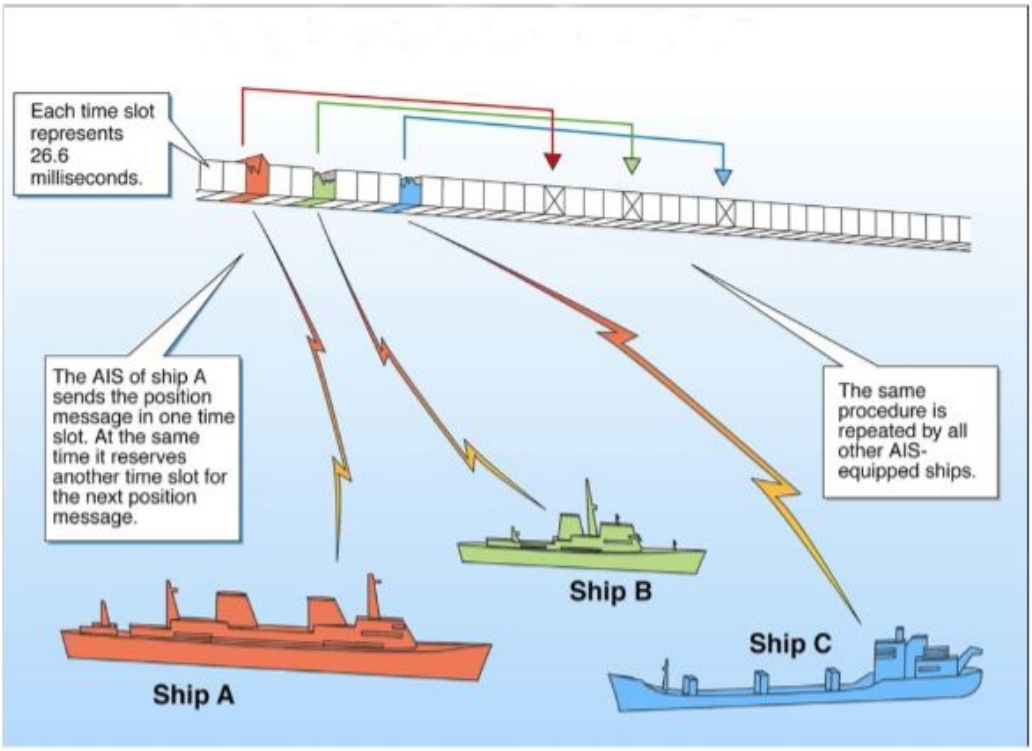


Fig. 2.3. Definición técnica de la tecnología TDMA.

Los transpondedores AIS de Clase A utilizan un sistema denominado Acceso Múltiple por División de Tiempo Autoorganizado (*SOTDMA*) donde múltiples transpondedores saben automáticamente cómo reclamar y reservar intervalos de tiempo, es decir, son capaces de reservar la misma ranura para retrasmisiones posteriores para evitar colisiones de datos con nuevos barcos. Para ello, todas las estaciones comparten una referencia de tiempo común (derivada del tiempo de GPS) asegurando que todas puedan determinar con precisión la hora de inicio de cada intervalo (*slot*) TDMA. En cada una de las transmisiones de datos de los buques se indica el intervalo TDMA que será utilizado por la estación transmisora para transmisiones posteriores permitiendo así a las estaciones receptores construir un "mapa" de qué ranuras están "en uso" por cada estación, y cuáles están "vacantes" y así evitar que otras estaciones transmitan al mismo tiempo mientras una de ellas está realizando sus propias transmisiones. Este sistema se basa dando prioridad a la distancia, es decir, cuando el sistema detecta una situación de saturación en un área determinada, es decir, existe un gran número de embarcaciones, los barcos más lejanos no obtienen un intervalo de tiempo. Esto permite que los buques más cercanos puedan ocupar las ranuras reservadas por otro que se encuentra más alejado de la zona.

Además, a medida que las estaciones móviles se mueven de un área a otra, permite que modifiquen su propia asignación ya que se pueden encontrar con nuevas estaciones con diferentes

asignaciones. Esto permite tener un sistema dinámico y autoorganizado en el tiempo y el espacio.

Sin embargo, los transpondedores de Clase B, utilizan una tecnología ligeramente diferente llamada TDMA (*Detección de portadora*), en la que el transpondedor de Clase B escucha los transpondedores de Clase A y tan pronto como detecta un intervalo de tiempo vacío, lo atrapa y realiza su transmisión. En este tipo de comunicación los transpondedores de Clase A siempre tienen prioridad sobre la Clase B, por lo que puede ocurrir que, ocasionalmente, un transpondedor de Clase A robe un intervalo de tiempo de un transpondedor de Clase B y este tenga que retrasar su transmisión y buscar otro intervalo. Los transpondedores de Clase B+ tienen la misma prioridad cuando se trata de reservar un intervalo de tiempo, lo que garantiza que siempre podrán transmitir [14].

Los sistemas AIS están compuestos por un receptor GPS y utilizan mensajes predefinidos para el intercambio de la información procedente del GPS y otra información importante. Estos mensajes son transmitidos a través de dos canales de frecuencias de banda marina de VHF dedicados al AIS (161.975 MHz y 162.025 MHz). Hay 27 mensajes estándar aprobados por la UIT (*Unión Internacional de Telecomunicaciones, UIT-R.M 1371-5, anexo 8) (ITU, 2014)* [15]. En este proyecto se utilizan los siguientes mensajes:

- Los mensajes tipo 1, 2, 3 incluyen información dinámica de la posición del buque. Cada buque difunde esta información en intervalos variables entre 2 segundos y 3 minutos.
- El mensajes tipo 5 incluye datos estáticos del buque. Cada buque difunde esta información en intervalos de 6 minutos.
- El mensaje tipo 6 incluye mensajes binarios direccionados (no difundidos).
- El mensaje tipo 7 es un acuse de recibo (ACKs) asociado al mensaje 6.

Mensajes tipo 1,2,3. Informes de posición

Los mensajes 1,2,3 tienen el formato mostrado en la siguiente tabla:

TABLA 2.2. CONTENIDO MENSAJES 1,2,3.PARTE 1.

Campo	**Bits**	**Descripción**	**Unidades**
0-5	6	ID del mensaje	Unsigned integer: 1..3
6-7	2	Nº de repeticiones	0..3; 0 = defecto; 3 = no repetir más
8-37	30	Número de identificación del servicio móvil marítimo o MMSI (Maritime Mobile Service Identity)	Unsigned integer: 9 digitos. Identifica inequívocamente a cada estación del servicio móvil digital (estaciones costeras y estaciones de barco)
38-41	4	Estado de navegación	0 = en camino con motor 1 = anclado 2=fuera de control 3=maniobrabilidad restringida 4=limitado por el calado 5=amarrado 6=encallado 7=pescando 8=en camino con vela 9=reservado para futuras enmiendas de estado de navegación para barcos que transportan materias peligrosas, sustancias dañinas o contaminantes marítimos, o contaminantes o peligrosde categoría C de la OMI (HSC) 10=reservado para futuras enmiendas de estado de navegación para barcos que transportan DG, HS o MP, o contaminantes o peligros decategoría A de la OMI (WIG) 11=barco de motor remolcado por popa (uso regional) 12=barco de motor en marcha o remolcado de costado (uso regional) 13=reservado para uso futuro 14=AIS-SART,MOB-AIS, RLS-AIS15 =no definido=defecto (también utilizado por el AIS-SART, MOB-AIS y RLS-AISen prueba)

TABLA 2.3. CONTENIDO MENSAJES 1,2,3. PARTE 2

Campo	**Bits**	**Descripción**	**Unidades**
42-49	8	Velocidad de giro (ROT)	Signed integer 0 a +126 = girando a la derecha a 708 grados por min o más; 0 a -126 = girando a la izquierda a 708 grados por min o más; Los valores entre 0 y 708 grados por min vienen codificados mediante: ROTAIS=4,733 SQRT(ROTsensor) grados por min, donde ROTsensor es la velocidad de giro introducida por un indicador de velocidad de giro externo (TI). ROTAIS está redondeado al valor entero más próximo. +127 = girando a la derecha a más de 5 grados por 30 s (no hay TI disponible) –127 = girando a la izquierda a más de 5 grados por 30 s (no hay TI disponible) –128 (80 hex) indica que no se dispone de información de giro =defecto. Los datos ROT no deben derivarse de la información COG
50-59	10	Velocidad sobre el fondo - efectiva (SOG)	Unsigned integer. Velocidad en pasos de 1/10 de nudo
60-60	1	Exactitud	1 = alto (<10 m) 0 = bajo (>10m) =defecto
61-88	28	Longitud	Minutos/10000. Valores entre -180 grados y + 180 grados. Este: positivo, Oeste: negativo (complemento a 2). 181= no disponible=defecto
89-115	27	Latitud	Minutos/10000. Valores entre -90 grados y + 90 grados. Norte: positivo, Sur: negativo (complemento a 2). 91= no disponible=defecto
116-127	12	Rumbo sobre el fondo - efectivo (COG)	Relativo al norte verdadero, 0.1 grados de precisión
128-136	9	Rumbo verdadero (HDG)	0 to 359 grados, 511 = no disponible=defecto
137-142	6	Indicación de tiempo	Segundos (UTC)

TABLA 2.4. CONTENIDO MENSAJES 1,2,3. PARTE 3

Campo	Bits	Descripción	Unidades
143-144	2	Indicador de maniobra especial	0 = no disponible = defecto 1 = no hay maniobra especial 2 = maniobra especial
145-147	3	Reservado	No empleado Debe ponerse en cero. Reservado para uso futuro
148-148	1	Bandera RAIM (supervisión de integridad autónoma de receptor) de dispositivo electrónico de determinación de posición	0 = RAIM no está en uso = defecto; 1 = RAIM en uso
149-167	19	Estado de comunicación	Ver más información en (Raymond, 2016)

Mensaje tipo 5. Datos estáticos y relativos al viaje del barco

El mensaje 5 incluye los datos estáticos o relativos al viaje mostrados en la siguiente tabla:

TABLA 2.5. CONTENIDO MENSAJE 5. PARTE 1

Campo	Bits	Descripción	Unidades
0-5	6	ID del mensaje	Unsigned integer: 5
6-7	2	Nº de repeticiones	0..3; 0 = defecto; 3 = no repetir más
8-37	30	MMSI	Unsigned integer: 9 digitos
38-39	2	Indicador de versión AIS	0 = estación conforme a la Recomendación UIT-R M.1371-1 1 = estación conforme a la Recomendación UIT-R M.1371-3 (o posterior) 2= estación conforme a la Recomendación UIT-R M.1371-5 (o posterior) 3 = estación conforme a ediciones futuras

TABLA 2.6. CONTENIDO MENSAJE 5. PARTE 2

Campo	Bits	Descripción	Unidades
40-69	30	Número OMI	Unsigned integer: 9 digitos 0 = no disponible = defecto – No aplicable a aeronave SAR 0000000001-0000999999 no se utiliza 0001000000-0009999999 = número OMI válido 0010000000-1073741823 = número de estado de bandera oficial
70-111	42	Distintivo de llamada	6 caracteres ASCII de 6 bit (ver Tabla 2.8) @@@@@@@ = no disponible = defecto
112-231	120	Nombre del buque	20 caracteres ASCII de 6 bits @@@@@@@@@@@@@@@@@@@@ = no disponible = defecto
232-239	8	Tipo de buque y tipo de carga	0..255. Ver Tabla 2.9
240-248	9	Distancia ref-proa	Unsigned integer (metros)
249-257	9	Distancia ref-popa	Unsigned integer (metros)
258-263	6	Distancia ref-babor	Unsigned integer (metros)
264-269	6	Distancia ref-estribor	Unsigned integer (metros)

TABLA 2.7. CONTENIDO MENSAJE 5. PARTE 3

Campo	**Bits**	**Descripción**	**Unidades**
270-273	4	Tipo de dispositivo electrónico de determinación de posición	0 = indefinido (defecto) 1 = GPS 2 = GLONASS 3 = GPS/GLONASS combinados 4 = Loran-C 5 = Chayka 6 = sistema de navegación integrado 7 = vigilado 8 = Galileo 9-14 = no empleado 15 = GNSS interno
274-277	4	ETA (Hora estimada de llegada) - mes	1..12; 0 = no disponible = defecto
278-282	5	ETA - día	1..31; 0 = no disponible = defecto
283-287	5	ETA - hora	0..23; 24 = no disponible = defecto
288-293	6	ETA - minutos	0..59; 60 = no disponible = defecto
294-301	8	Calado estático actual máximo	En 1/10 m 255 = calado 25.5 m o mayor 0 = no disponible = defecto
302-421	120	Destino	Máximo 20 caracteres ASCII de 6-bits; @@@@@@@@@@@@@@@@@@@@ = no disponible
422	1	DTE	Terminal de datos listo 0 = disponible 1 = no disponible = defecto
423	1	Reserva	No empleado. Debe ser cero. Reservado para uso futuro

TABLA 2.8. CODIFICACIÓN ESPECIAL ASCII DE 6 BITS.

Caracter	**Dec**	**Hex**	**Binario**	**Caracter**	**Dec**	**Hex**	**Binario**
@	0	0x00	000000	!	33	0x21	100001
A	1	0x01	000001	“	34	0x22	100010
B	2	0x02	000010	#	35	0x23	100011
C	3	0x03	000011	$	36	0x24	100100
D	4	0x04	000100	%	37	0x25	100101
E	5	0x05	000101	@	38	0x26	100110
F	6	0x06	000110	‘	39	0x27	100111
G	7	0x07	000111	(	40	0x28	101000
H	8	0x08	001000	)	41	0x29	101001
I	9	0x09	001001	*	42	0x2A	101010
J	10	0x0A	001010	+	43	0x2B	101011
K	11	0x0B	001011	,	44	0x2C	101100
L	12	0x0C	001100	-	45	0x2D	101101
M	13	0x0D	001101	.	46	0x2E	101110
N	14	0x0E	001110	/	47	0x2F	101111
O	15	0x0F	001111	0	48	0x30	110000
P	16	0x10	010000	1	49	0x31	110001
Q	17	0x11	010001	2	50	0x32	110010
R	18	0x12	010010	3	51	0x33	110011
S	19	0x13	010011	4	52	0x34	110100
T	20	0x14	010100	5	53	0x35	110101
U	21	0x15	010101	6	54	0x36	110110
V	22	0x16	010110	7	55	0x37	110111
W	23	0x17	010111	8	56	0x38	111000
X	24	0x18	011000	9	57	0x39	111001
Y	25	0x19	011001	:	58	0x3A	111010
Z	26	0x1A	011010	;	59	0x3B	111011
[	27	0x1B	011011	<	60	0x3C	111100
\	28	0x1C	011100	=	61	0x3D	111101
]	29	0x1D	011101	>	62	0x3E	111110
^	30	0x1E	011110	?	63	0x3F	111111
–	31	0x1F	011111				
Espacio	32	0x20	100000				

TABLA 2.9. ALGUNOS TIPOS DE BARCO Y TIPOS DE CARGA

Valor	**Descripción**
0	No disponible o ningún barco = defecto
30	Pesca
31	Remolcando
32	Remolcando y la longitud del remolque >200 m o la anchura >25m
33	Dragado u operaciones submarinas
34	Operaciones submarinas
35	Operaciones militares
36	Navegación a vela
37	Turismo
41	Barco de alta velocidad transportando mercancías peligrosas
50	Barco de práctico
51	Barcos de rescate
52	Remolcadores
53	Pilotos de puerto
54	Barcos con facilidades o equipos antipolución
55	Guardacostas
60..69	Barcos de pasaje
70..79	Barcos de carga
80..89	Buques tanque
90-99	Otros tipos
100-199	Reservado, para uso regional
200-255	Reservado, para uso futuro

Mensaje 6. Mensaje binario direccionado

El mensaje binario direccionado, contiene los parámetros resumidos en la siguiente tabla:

TABLA 2.10. CONTENIDO MENSAJE 6.

Campo	Bits	Descripción	Unidades
0-5	6	ID del mensaje	Unsigned integer: 5
6-7	2	Nº de repeticiones	0..3; 0 = defecto; 3 = no repetir más
8-37	30	ID de origen	Unsigned integer. 9 dígitos. MMSI del buque origen
38-39	2	Nº de secuencia	0..3
40-69	30	ID de destino	Unsigned integer. 9 dígitos. MMSI del buque destino
70	1	Bandera de retransmisión	0 = no hay retransmisión = defecto 1 = retransmitido
71	1	Reserva	No empleado. Debe ser 0
72-87	16	ID de aplicación	16 bits
88-	920	Datos de aplicación	Hasta 920 bits (115 bytes), divididos en intervalos: intervalo 1: 8 bytes intervalo 2: 36 bytes intervalo 3: 64 bytes intervalo 4: 94 bytes intervalo 5: 115 bytes

El campo ID de aplicación se divide en dos zonas, según muestra la Tabla 2.11.

TABLA 2.11. CAMPO ID DE APLICACIÓN DEL MENSAJE 6.

Bits	Descripción
15-6	Código de zona designada (DAC). Este código se basa en las cifras de identificación marítima (MID). Cero (prueba) y 1 (internacional) son excepciones
5-0	0..63. Identificador de función. El significado debe ser determinado por la autoridad responsable de la zona dada en el código de zona designada

El ID de aplicación de prueba (DAC = 0) con cualquier identificador de función (0 a 63) debe emplearse para pruebas. El identificador de función es arbitrario. El ID de aplicación internacional (DAC = 1) debe emplearse para aplicaciones internacionales de importancia mundial.

Para la aplicación que se desarrolla en este proyecto se utilizará DAC = 0 y se utilizarán los 6 bitsidentificadores de función para codificar los mensajes que un PLC va a enviar a otro. Se podrán codificar hasta 64 mensajes distintos.

Los datos de aplicación ocuparán el mínimo número de bits posible. Se enviará 1 intervalo, 8 bytes, incluyendo ID de aplicación, lo que deja 48 bits para los datos de aplicación. Si es necesario, 2 intervalos, hasta 36 bytes (hasta 34 bytes de datos).

Mensaje 7. Acuse de recibo binario

El Mensaje 7 se emplea como acuse de recibo de hasta cuatro Mensajes 6 recibidos y se transmite a través del canal por el que se ha recibido el mensaje direccionado del que se acusa recibo. El formato se resume en la Tabla 2.12

TABLA 2.12. CONTENIDO MENSAJE 7.

Campo	**Bits**	**Descripción**	**Unidades**
0-5	6	ID del mensaje	Unsigned integer: 7
6-7	2	Nº de repeticiones	0..3; 0 = defecto; 3 = no repetir más
8-37	30	ID de origen	Unsigned integer. 9 dígitos. MMSI del buque origen
38-39	2	Reserva	0 – No se utiliza
40-69	30	ID1 de destino	Unsigned integer. 9 dígitos. MMSI del buque destino1
70-71	2	Reserva	0 – No se utiliza
72-101	30	ID2 de destino	Unsigned integer. 9 dígitos. MMSI del buque destino2
102-103	2	Reserva	0 – No se utiliza
104-133	30	ID3 de destino	Unsigned integer. 9 dígitos. MMSI del buque destino3
134-135	2	Reserva	0 – No se utiliza
136-165	30	ID4 de destino	Unsigned integer. 9 dígitos. MMSI del buque destino4
166-167	2	Reserva	0 – No se utiliza

Además del protocolo TDMA utilizado para el intercambio de información a través de la radio VHF, las estaciones AIS utilizan interfaces digitales marítimos y estándares de comunicación de datos para el intercambio de datos con otros dispositivos, sistemas o redes. Esto facilita la visualización y el uso de la información AIS a bordo del buque. A través de estos interfaces digitales, el AIS de un buque comunica al sistema PE los datos de navegación (estáticos y dinámicos) propios y los recibidos de otros buques por los canales VHF, además de los mensajes binarios definidos. Esta comunicación digital se realiza mediante sentencias o grupos de parámetros PGN(Parameter Group Number) codificados siguiendo estándares NMEA (*NMEA 0183, 2008; NMEA 2000, 2016*) [3], [4].

2.1.1. NMEA 0183

El NMEA 0183 es un código de señales que utilizan los aparatos electrónicos para emitir sus propios datos y, al mismo tiempo comprender los que reciben de otros aparatos. Los datos son transmitidos en serie en forma asíncrona, de acuerdo con el estándar 2.1 de IEC 61162-1/2, conocido como NMEA 0183, mediante los siguientes parámetros de transmisión:

- Velocidad: 38,4 Kbps (IEC 61162-2) o 4,8 Kbps (IEC61162-1).
- Bits de datos: 8 (D7= 0); sin paridad.
- Bit de parada: 1.

El primer bit es el de arranque y está seguido por los bits de datos y el bit de parada y toda sentencia NMEA 0183 contiene, en el orden que se indica, los siguientes elementos:

- “$ ” or “!” - Inicio de sentencia
- <address field >- Identificador de la sentencia y del emisor
- “,” <data field >- Cero o más campos de datos
- ...
- “,” <data field >- Cero o más campos de datos

Existen diferentes sentencias NMEA 0183 pero, en este proyecto se han utilizado las siguientes:

Sentencia VDM

Transfiere el mensaje AIS recibido por VHF, encapsulado y codificado según lo definido en UIT-R.M 1371-5 (codificación de 6 bits).

!--VDM,x[1],x[2],x[3],a[4],s--[5]s,x[6]*hh[7]<CR><LF>

Fig. 2.4. Formato sentencia VDM.

1: número total de fragmentos necesarias para transferir el mensaje (1 a 9).
2: número del fragmento actual (1 a 9).
3: ID de la sentencia para mensajes multisentencia.
4: código del canal de radio. AIS utiliza la parte alta del dúplex de dos canales de radio.
VHF: Canal A es 161.975Mhz (87B); Canal B es 162.025Mhz (88B).
5: mensaje ITU-R M.1371 encapsulado.
6: número de bits de relleno (0 a 5).
7: suma de verificación (Checksum).

Sentencia VDO

Transfiere la información del buque propio. Formato idéntico a VDM.

Sentencia ABM

Mensaje binario o de seguridad direccionado. A la recepción de esta sentencia, el AIS enviará el mensaje (tipo 6 o tipo 12) al buque direccionado.

!AIABM,x,x,x,xxxxxxxxx,x,xx,s--s,x*hh<CR><LF>

Fig. 2.5. Formato sentencia ABM.

Sentencia ABK

Esta sentencia se genera cuando un AIS recibe un mensaje de ACK tipo 7, 13 o el fallo del envío del mensaje 6,12.

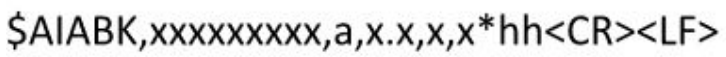


Fig. 2.6. Formato sentencia ABK.

2.1.2. NMEA 2000

El objetivo de esta norma es facilitar la interconexión entre equipos de diferentes fabricantes. NMEA 2000 esta basado en el protocolo industrial CANBUS (*Controller Area Network*), utilizado en automoción y automatización industrial desde hace algunos años.

La asociación NMEA creó este nuevo estándar debido al creciente número de productos electrónicos marinos a bordo que son necesarios conectar entre sí. Actualmente, prácticamente la mayoría de los productos electrónicos marinos se alimentan directamente de la red NMEA2000, todo nuevo instrumento electrónico marino viene con su conector NMEA 2000, normalmente al lado de la "vieja" toma NMEA 0183. Este nuevo estándar soluciona algunos de los problemas que presenta NMEA 0183, principalmente permite aumentar la velocidad de comunicación entre los productos electrónicos marinos y también facilita la instalación de productos electrónicos marinos ya que sus conectores están estandarizados entre todas las marcas de electrónica.

La principal desventaja que presenta este nuevo estándar es que es incompatible con NMEA 0183, mientras exista un viejo equipo NMEA 0183 ya que se trata de protocolos diferentes cuyas velocidades de transmisión son diferentes y, por tanto, los datos son ininteligibles entre sí y, además, los códigos de datos también son distintos y siempre requieren de un convertidor de datos.

Las sentencias que utiliza este nuevo protocolo se denominan PGNs. Incluyen información equivalente a los mensajes contenidos en las sentencias NMEA 0183. En la tabla 1 se incluyen los más relevantes para este proyecto.

2.1.3. Comunicación AIS-PLC implementada en el proyecto

Software AIS Simulator Version 8.04a - October 2019

Durante el desarrollo del proyecto, con el fin de implementar la comunicación NMEA entre un AIS real y un PLC, se barajaron diferentes opciones. La idea inicial era utilizar sistemas AIS reales para probar dicha comunicación con PLCnext por lo que se hizo un estudio inicial sobre varios de sistemas con el objetivo de escoger cuál era el más adecuado. Se optó por utilizar un

Transpondedor AIS Clase A Marca EM-TRAK Modelo A200 [16]:

Fig. 2.7. Transpondedor AIS Clase A Marca EM-TRAK Modelo A200.

TABLA 2.13. CARACTERÍSTICAS AIS CLASE A EM-TRAK MODELO A200.

Características
Totalmente certificado (Deep Sea & Navegación interior)
Resistente al agua y a la intemperie
Certificación IPx6 e IPx7
Alta resolución integrada pantalla a color
Carta de navegación electrónica completa motor y pantalla
WiFi de largo alcance integrado conectividad
Dual NMEA0183 y 2000

Finalmente no se ha podido disponer de estos sistemas reales por falta de presupuesto. Por esta razón, surgió la idea final, la cuál fue elegida, de utilizar la aplicación *AIS Simulator Version 8.04a - October 2019* [17] para simular señales de estaciones AIS reales.

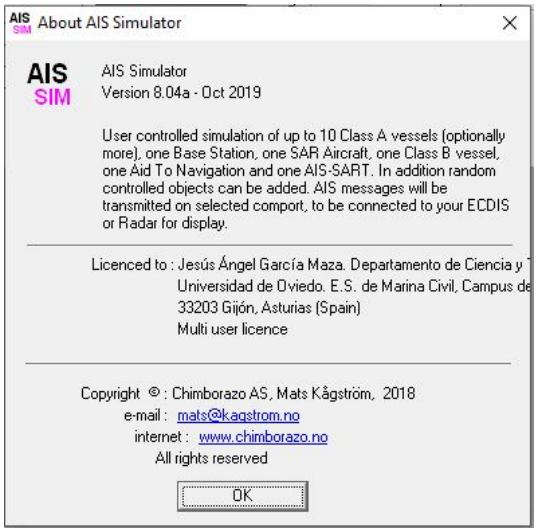


Fig. 2.8. AIS Simulator Version 8.04a - October 2019

Este software está diseñado para simular datos AIS para hasta 10 embarcaciones Clase A estándar (opcionalmente>10) controladas por el usuario como si se tratase de un AIS real ya que, para la entrada/salida de datos, permite establecer una comunicación serie RS232, real o virtual, TCP / IP o UDP en función de lo que el usuario considere adecuado.

Software PLCnext Engineer 2020.0 (Build 4.0.250.0)

Con el fin de implementar en PLCnext un programa que permita la decodificación de las sentencias NMEA recibidas de la estación AIS Simulator, se ha optado por utilizar el software *PLCnext Engineer 2020.0 (Build 4.0.250.0)* [18]. Se trata de una plataforma de software de ingeniería para los controles de automatización de Phoenix Contact y que cumple con IEC 61131-3.

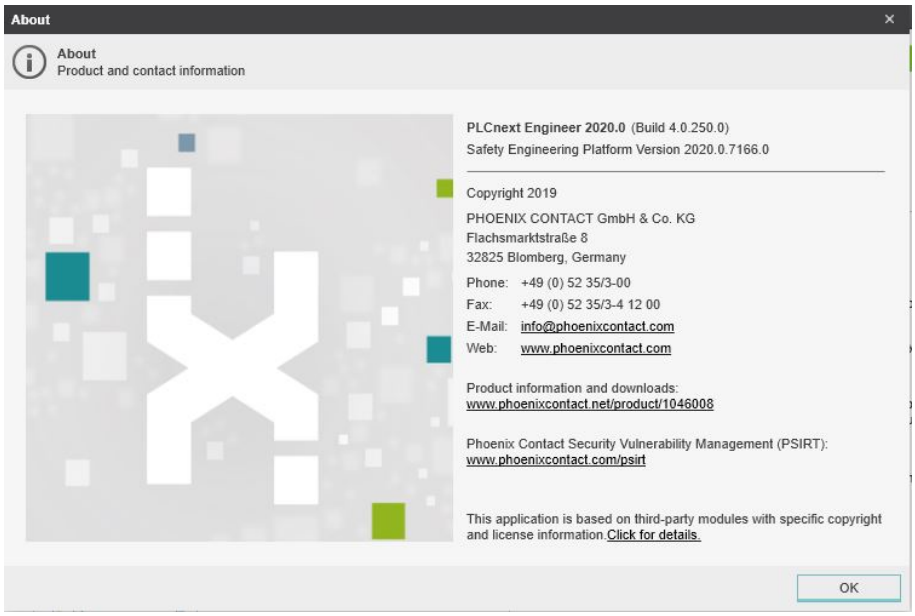


Fig. 2.9. Software *PLCnext Engineer 2020.0 (Build 4.0.250.0)*.

Además, presenta las siguientes ventajas:

- Ingeniería flexible mediante la integración de complementos funcionales individuales en la versión básica gratuita.
- Simplificación del proceso de ingeniería mediante el uso de módulos de automatización y programación orientada al objeto.
- Ahorros de tiempo y costes mediante una programación completa en una interfaz.
- Un software para todas las tareas de ingeniería.
- Menor esfuerzo de trabajo y formación gracias a la interfaz de usuario optimizada para el usuario Manejo único a nivel mundial gracias a la programación conforme a IEC 61131 Programación segura certificada según IEC 61508.
- Seguridad por diseño con los sistemas de control de la línea PLCnext Control.

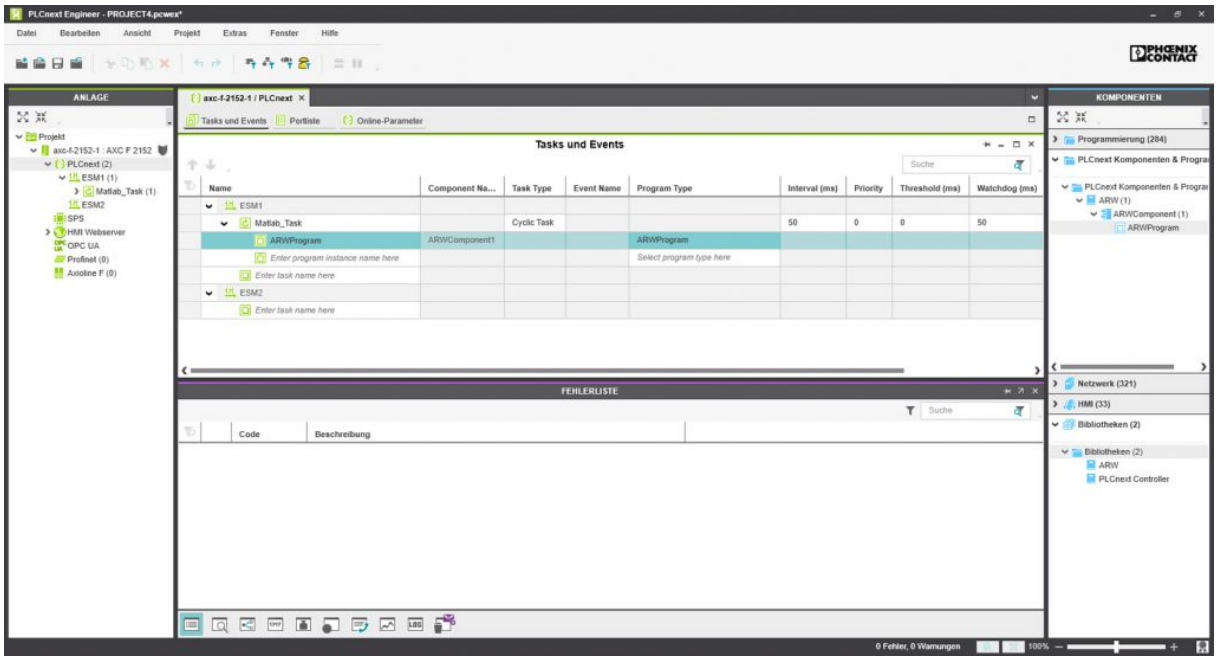


Fig. 2.10. Entorno de desarrollo PLCnext Engineer 2020.0.

Finalmente, en este proyecto, se ha optado por establecer una comunicación serie RS232 como se muestra en la Figura 2.11:

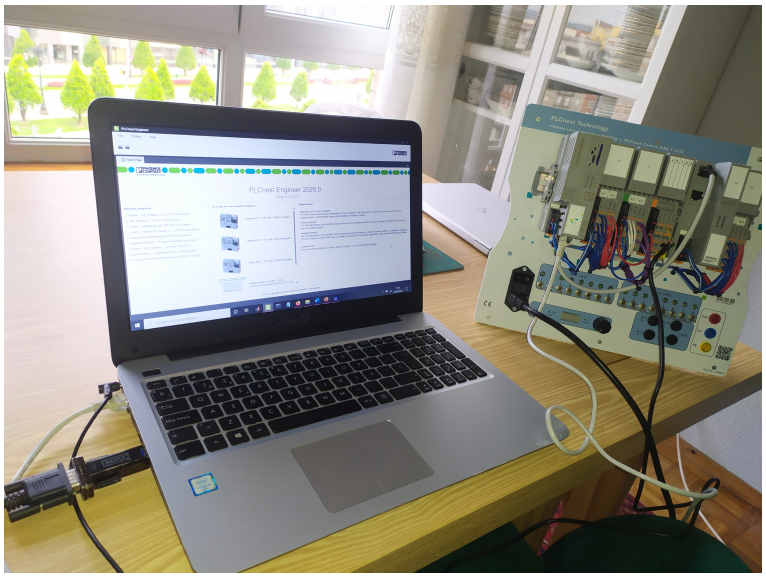


Fig. 2.11. Conexión comunicación serie PLC-AIS.

Para ello, se han utilizado y configurado los siguientes dispositivos y herramientas:

- **Adaptador USB 2.0 a Serial - Digitus DA-70156:** dado que el ordenador que se ha

utilizado para el desarrollo de este proyecto carece de conexión serie, se ha optado por utilizar este adaptador [19]. Se trata de un pequeño equipo que tiene un conector USB en un extremo y al menos uno, o quizás más, conectores serie en el otro extremo y que sirve para convertir entre señales USB y serie.

Fig. 2.12. AdaptadorDigitus DA-70156.

- **Módulo de comunicación Axioline F para datos en serie transmisión - AXL F RS UNI 1H:** para permitir esta comunicación es necesario conectar al módulo PLCnext Control F 2152 el módulo de comunicación mostrado en la Figura 2.13. Está diseñado para su uso dentro de una estación Axioline F y se utiliza para operar dispositivos de E / S estándar con interfaces seriales en un sistema de bus [9].

Fig. 2.13. Módulo AXL F RS UNI 1H.

- **Cable Serie RS232 de comunicaciones:** se ha utilizado para conectar el adaptador *Digitus DA-70156* y el *módulo AXL F RS UNI 1H* como se muestra en la Figura 2.14.

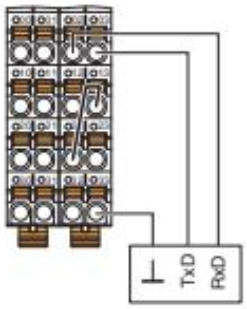


Fig. 2.14. Conexión establecida con cable serie RS232 en el *módulo AXL F RS UNI 1H*.

- **Configuración puerto serie ordenador (COM3):** se ha comprobado que la configuración del puerto serie del ordenador coincide con la configuración del software AIS Simulator y con la configuración serie fijada en PLCnext:

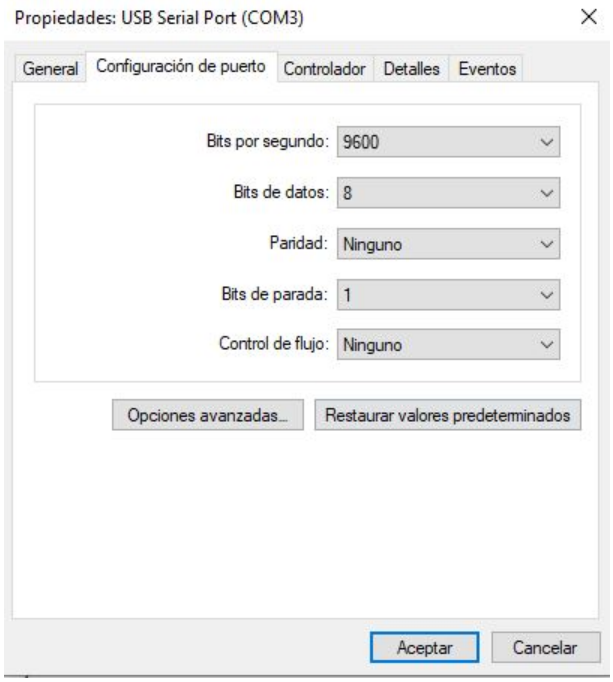


Fig. 2.15. Configuración puerto serie ordenador (COM3).

- **Configuración software AIS Simulator Version 8.04a - October 2019:** para permitir la comunicación serie fue necesario configurar la herramienta de acuerdo a la configuración del puerto serie que se muestra en la Figura 2.15:

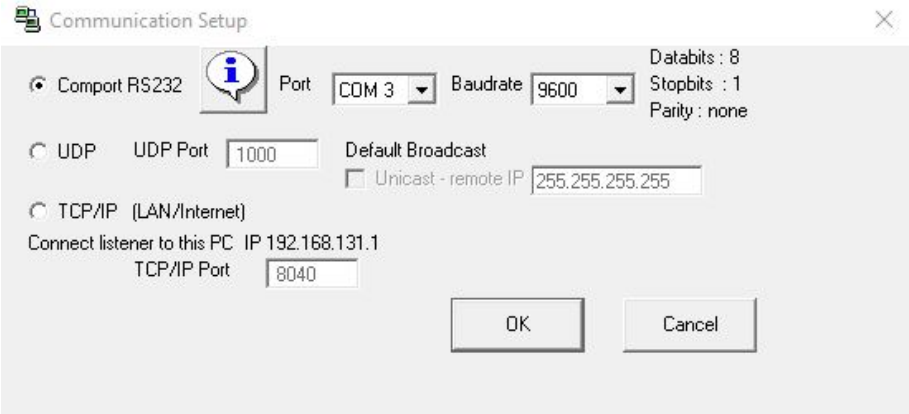


Fig. 2.16. Configuración comunicación software AIS Simulator.

- **Configuración módulo comunicación serie - AXL F RS UNI 1H:** para permitir la comunicación serie fue necesario configurar este módulo mediante la herramienta *PLCnext Engineer 2020.0* de acuerdo a la configuración del puerto serie que se muestra en la Figura 2.15:

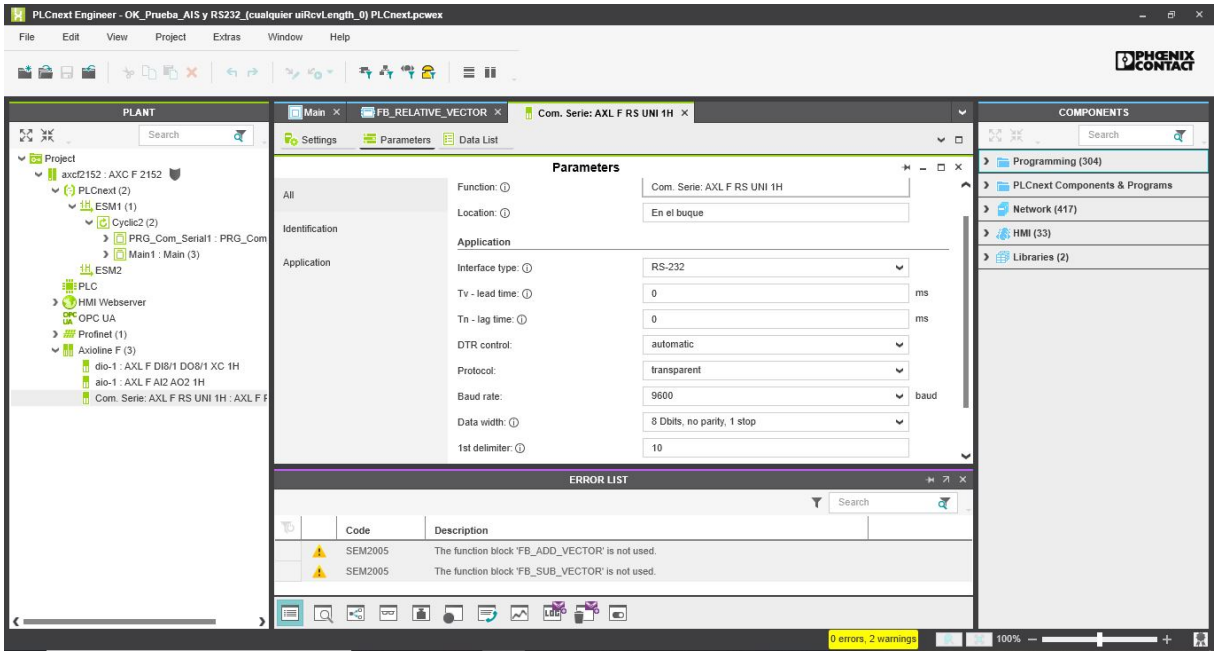


Fig. 2.17. Configuración comunicación serie en PLCnext.

- **Programa en PLCnext Engineer 2020.0:** se ha creado un programa que permite la decodificación de las sentencias NMEA recibidas de la estación AIS denominado *DecodingNMEASentences.pcwex*. Para recibir esta información se ha creado un programa adicional al programa principal que realiza la lectura del puerto serie. Para ello, es necesario utilizar

el bloque funcional *AXL_RSUNI_PD* proporcionado por la librería *AXL_ComSerial* [20]. Este bloque funcional proporciona al usuario una opción conveniente para configurar una conexión en serie cuando se utiliza el módulo *AXL F RS UNI 1H*.

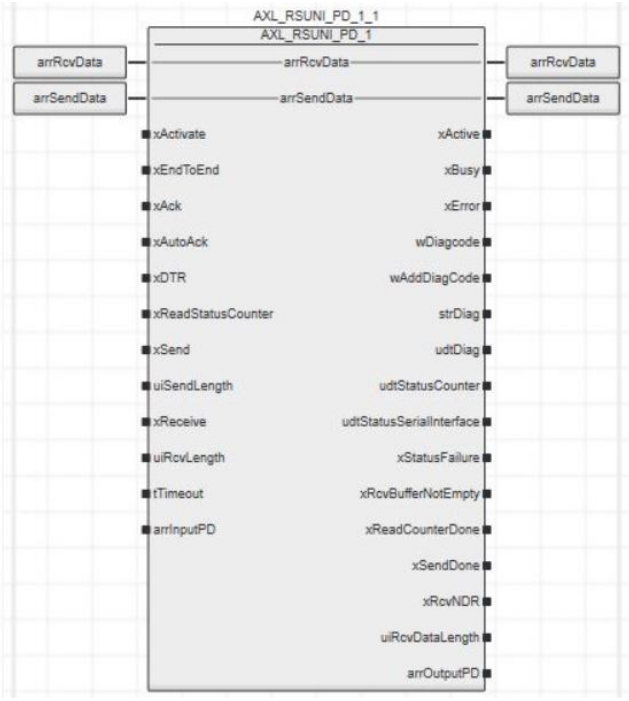


Fig. 2.18. Bloque funcional *AXL_RSUNI_PD*.

Para la recepción correcta de la información, se han configurado algunos de los parámetros de entrada del mismo de la siguiente manera:

- **xActivate (BOOL):** debe estar activa para que la comunicación sea posible.
- **xReceive (BOOL):** esta variable permite configurar la comunicación serie a “modo recepción". Para su activación se comprueba previamente que el buffer de recepción de datos no está vacío (variable salida *xRcvBufferNotEmpty*) y que no se han leído ya dichos datos (variable salida *xRcvNDR*). De esta forma permitirá activar y desactivar este modo de comunicación para una nueva recepción de datos.

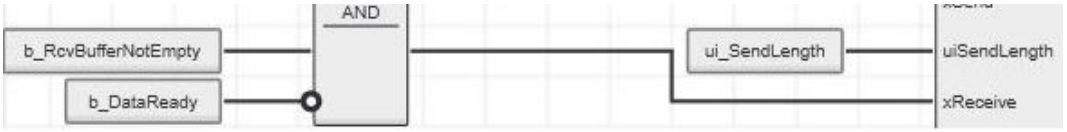


Fig. 2.19. Comprobación activación modo RX.

- **uiRcvLength (UNIT):** permite fijar el número de bytes de lectura. En este caso, se ha definido a 0 para permitir la lectura completa de todos los datos (*end-to-end*).

- **tTimeout (TIME):** los datos no se pueden recibir dentro del tiempo especificado. Se ha definido a T#150ms, un tiempo pequeño, para evitar que los mensajes se solapen en la recepción.
- **Resto de variables por defecto.**

2.1.4. Obtención de los datos - Sentencias NMEA

En el programa principal (*main*) definido en el PLC, tras la recepción de la sentencia NMEA a partir del bloque funcional *AXL_RSUNI_PD* se procede a la fragmentación de la misma para la obtención de la parte de datos. Y, posteriormente se obtiene el (*ID Mensaje*) para obtener los datos de acuerdo al tipo de mensaje recibido.

```
//recepción sentencia NMEA

FT (CLK:= x_RcvBufferNotEmpty);
IF  FT.Q THEN
   FOR i:=1 to ui_RcvDataLength do
      sarr_RcvData[i] := TO_STRING (arr_RcvData[i], '{0:c}');
   END_FOR;
   s_AIS := '';
   FOR i:= 1 TO (ui_RcvDataLength) DO
      s_AIS := CONCAT ( s_AIS, sarr_RcvData[i] );
      SR1 (SET1 := (i=83) , RESET:= x_Valor);
      xTest := SR1.Q1;
   END_FOR;
   i:=0;
   s_MsgAIS := s_AIS ;
END_IF;
```

Fig. 2.20. Obtención mensaje AIS.

```
//mensaje AIS (p.e  '!AIVDM,1,1,,B,13n34qP02FOWHNpIPm7P000:0L01,0*21')
s_xAIS:=s_MsgAIS;
//se obtiene el tipo sentencia (VDO, ABM)
s_IndMessage:=MID(s_xAIS, 5, 2);
//se elimina la parte inicial del mensaje (p.e '13n34qP02FOWHNpIPm7P000:0L01,0*2')
s_Message:=MID(s_xAIS, LEN(s_xAIS)-15, 15);

//se elimina la parte final del mensaje que no corresponde a datos
i_pos:=FIND(s_Message,',');
IF i_pos=1 THEN
   s_Message:=MID(s_xAIS, LEN(s_xAIS)-15, 16);
   i_pos:=FIND(s_Message,',');
END_IF;

//se obtiene la parte del mensaje que ya corresponde a los datos (p.e 13n34qP02FOWHNpIPm7P000:0L01)
 s_DataMessage:=MID(s_Message,i_pos-1,1);

//se procede a la decodificación de los datos
FB_Deco_Output_AIS1(iMessage := s_DataMessage, oDecoMessage => xDecoMessage);

//se obtiene el ID Mensaje para su posterior obtención de los datos a partir de los datos decodificados (msg tipo 1,5,6
ó 7)
ul_IDMessage:=F_BITS_TO_NUMBER(5, 0,xDecoMessage );
```

Fig. 2.21. Obtención parte de datos del mensaje AIS.

A continuación, para la decodificación de los datos del mensaje AIS, se ha definido el bloque funcional *FB_Deco_Output_AIS* como se muestra en la Figura 2.23. Este bloque funcional recibe la cadena de caracteres correspondiente a los datos y extrae los bits del mensaje de acuerdo a la Tabla 2.8 definida anteriormente.

Variables

in_Message	STRING	Input	Mensaje AIS
i_i	INT	Local	Variable bucle
i_Caracter	INT	Local	Valor caracter en códio ASCII
i_Division	INT	Local	Variable temporal cálculo valor decimal
s_DecoCaracter	STRING	Local	Variable almacenaje bits - caracter
s_Bit	STRING	Local	Variable temporal cálculo valor decimal - resto división
i_j	INT	Local	Variable bucle
oDecoMessage	MyArray	Output	Array de bits del mensaje AIS decodificado
i_k	INT	Local	Variable bucle
i_Tmp	INT	Local	Variable temporal almacenaje bits final

```
i_j:=0;
//se recorre el mensaje y se coge cada uno de los caracteres
FOR i_i := 1 TO LEN(in_Message) BY 1 DO
 //obtenemos el valor del caracter en codigo ASCII
 i_Caracter:=GET_CHAR(in_Message,i_i);
 //se obtiene su valor decimal y, para ello, si es mayor de 96 se le resta 56 sino 48
 IF i_Caracter>=96 THEN
     i_Caracter:=SUB(i_Caracter,56);
  ELSE
     i_Caracter:=SUB(i_Caracter,48);
 END_IF;
```

Fig. 2.22. Código *FB_Deco_Output_AIS*. Parte 1.

```
//una vez obtenido el valor decimal, se pasa a obtener su valor binario
i_Division:=i_Caracter;
s_DecoCaracter:='';
//para ello, se divide entre dos y se queda con el resto hasta que el número no se pueda dividir
más
WHILE NOT (i_Division = 0) DO
      s_Bit:= TO_STRING(MOD(i_Division,2),'{0:D}');
      i_Division:=DIV(i_Division,2);
    s_DecoCaracter:= CONCAT(s_Bit,s_DecoCaracter);
END_WHILE;
//dado que la decodificación de cada caracter debe ser de 6 bits se comprueba que es así,
//sino se añaden los ceros que sean necesarios hasta obtener dicha longitud
WHILE NOT (LEN(s_DecoCaracter)=6) DO
   s_DecoCaracter:=CONCAT('0',s_DecoCaracter);
END_WHILE;
//Por último, se añade cada uno de los bits al array final
FOR i_k := 1 TO LEN(s_DecoCaracter) BY 1 DO
   i_Tmp:=GET_CHAR(s_DecoCaracter,i_k);
   IF i_Tmp>=96 THEN
     i_Tmp:=SUB(i_Tmp,56);
   ELSE
     i_Tmp:=SUB(i_Tmp,48);
  END_IF
   oDecoMessage[i_j]:=i_Tmp;
  i_j:=i_j+1;
   END_FOR;
END_FOR;
```

Fig. 2.23. Código *FB_Deco_Output_AIS*. Parte 2.

Para la obtención de los datos a partir de los bits obtenidos tras la decodificación de la cadena de caracteres, fue necesario definir las siguientes funciones:

- **F_BITS_TO_CARACTER:** obtiene para un conjunto de bits el carácter asociado especificado en la Tabla 2.8.

Variables

i_j	INT	Local	Variable bucle
s_String	STRING	Local	Variable almacena caracter final
i_InitialRange	INT	Local	Bit inicial correspondiente a un caracter (6bits)
in_FinalBit	INT	Input	Bit final correspondiente al dato del mensaje final
i_FinalRange	INT	Local	Bit final correspondiente a un caracter (6bits)
i_Index	INT	Local	Variable bucle
in_InitialBit	INT	Input	Bit inicial correspondiente al dato del mensaje final
ul_Suma	ULINT	Local	Variable temporal suma
in_DecoMessage	MyArray	Input	Array de bits correspondiente al mensaje AIS
i_pos	INT	Local	Variable bucle

```
/Desde el bit final pasado por parámetro va cogiendo  bits de 6 en 6
//hasta alcanzar el bit inicial pasado por parámetro
i_InitialRange:=in_FinalBit-5; s_String:='';
i_FinalRange:=in_FinalBit;
WHILE (i_InitialRange>=in_InitialBit) DO
   //para cada grupo de 6 bits obtiene su valor decimal
   For i_Index := i_FinalRange To i_InitialRange By -1 DO
       ul_Suma:=ul_Suma+to_ulint(in_DecoMessage[ i_Index ])*to_ulint(EXPT(2.0, i_pos));
      i_pos:=i_pos+1;
   End_For;
   //le suma 16
   ul_Suma:=ADD(ul_Suma,16);
```

Fig. 2.24. Código *F_BITS_TO_CARACTER*. Parte 1.

```
  //en función del valor obtenido si es mayor de 96 le suma 56 sino le suma 48
  IF ul_Suma>=96 THEN
     ul_Suma:=ADD(ul_Suma,56);
   ELSE
     ul_Suma:=ADD(ul_Suma,48);
  END_IF
  //convierte a byte el valor obtenido para posteriormente obtener el carácter asociado
  // a dicho valor (codificación ASCII).
  s_String:=CONCAT(TO_STRING(TO_BYTE(TO_INT(ul_Suma)),'{0:C}'), s_String);
  i_pos:=0;
  ul_Suma:=0;
  i_FinalRange:=i_InitialRange-1;
  i_InitialRange:=i_FinalRange-5;
END_WHILE;

F_BITS_TO_CARACTER:=s_String;
```

Fig. 2.25. Código *F_BITS_TO_CARACTER*. Parte 2.

- **F_BITS_TO_NUMBER :** obtiene para un conjunto de bits su valor decimal.

Variables

ul_Suma	ULINT	Local	Valor numero final
i_pos	INT	Local	Variable bucle
i_Index	INT	Local	Variable bucle
in_FinalBit	INT	Input	Bit final correspondiente al dato del mensaje final
in_InitialBit	INT	Input	Bit inicial correspondiente al dato del mensaje final
in_DecoMessage	MyArray	Input	Array de bits correspondiente al mensaje AIS

```
For i_Index := in_FinalBit To in_InitialBit By -1 DO
   ul_Suma:=ul_Suma+to_ulint(in_DecoMessage[ i_Index ])*to_ulint(EXPT(2.0, i_pos));
   i_pos:=i_pos+1;
End_For;

F_BITS_TO_NUMBER:=ul_Suma;
```

Fig. 2.26. Código *F_BITS_TO_NUMBER*.

- **F_BITS_TO_COORDENATES:** obtiene el valor en grados para un valor decimal dado.

Variables

r_Valor	REAL	Local	Valor coordenada en grados
in_ValorInt	ULINT	Input	Valor Latitud/Longitud en minutos (millas)
s_ValorCoord	STRING	Local	Signo negativo Latitud/Longitud
in_ValorNeg	BOOL	Input	Indicador Latitud/Longitud negativo

```
//se obtiene la parte entera
r_Valor:=(TO_REAL(in_ValorInt)/10000.0)/60.0;
//comprueba si es un angulo positivo o negativo
IF in_ValorNeg THEN
   s_ValorCoord:='-';
END_IF;
s_ValorCoord:=CONCAT(s_ValorCoord,TO_STRING(r_Valor,'{0}'));
F_BITS_TO_COORDENATES:=s_ValorCoord;
```

Fig. 2.27. Código *F_BITS_TO_COORDENATES*.

Por último, en función del *ID Mensaje* se procede a la obtención de los datos utilizando las funciones definidas anteriormente y las Tablas 2.2,2.3,2.4, 2.5, 2.6, 2.7 y 2.10:

```
//Mensaje 5. Datos estáticos y relativos al viaje del barco
ELSIF ul_IDMessage=5 THEN
   ul_MMSI:=0; ul_NavState:=0; ul_ROT:=0; ul_SOG:=0; ul_Precision:=0; ul_OMINumber:=0;
ul_AISVersion:=0; s_Longitude:=''; s_Latitude:=''; ul_COG:=0; ul_HDG:=0;
   ul_TimeIndicator:=0; ul_ManeuverIndicator:=0; ul_Reserved:=0; ul_RAIMFlag:=0;
ul_CommStatus:=0;
   ul_IDOrigin:=0; ul_SequenceNumber:=0; ul_IDDestination:=0; ul_RetransmissionFlag:=0;
ul_Reservation:=0; ul_IDFunction:=0; ul_CodDAC:=0;
   ul_IDDestination1:=0; ul_IDDestination2:=0; ul_IDDestination3:=0; ul_IDDestination4:=0;
      r_PolarCoordAng:=0; r_PolarCoordMod:=0; r_CartesianCoordX:=0;
r_CartesianCoordY:=0;  s_ShipOwn:='';
   ul_RepetitionNumber:=F_BITS_TO_NUMBER(7, 6,xDecoMessage );
   ul_MMSI:=F_BITS_TO_NUMBER(37, 8,xDecoMessage );
   out_Mensaje5OPC[0]:=TO_STRING(ul_MMSI,'{0}');
   ul_AISVersion:=F_BITS_TO_NUMBER(39, 38,xDecoMessage );
   ul_OMINumber:=F_BITS_TO_NUMBER(69, 40,xDecoMessage );
   out_Mensaje5OPC[1]:=TO_STRING(ul_OMINumber,'{0}');
   s_DistinctiveCall:= F_BITS_TO_CARACTER(111, 70,xDecoMessage);
   out_Mensaje5OPC[2]:=s_DistinctiveCall;
   s_VesselName:=  F_BITS_TO_CARACTER(231, 112,xDecoMessage);
   out_Mensaje5OPC[3]:=s_VesselName;
   ul_VesselTypeLoad:=F_BITS_TO_NUMBER(239, 232,xDecoMessage );
   out_Mensaje5OPC[4]:=TO_STRING(ul_VesselTypeLoad,'{0}');
   ul_ProwDistance:=F_BITS_TO_NUMBER(248, 240,xDecoMessage );
   out_Mensaje5OPC[5]:=TO_STRING(ul_ProwDistance,'{0}');
   ul_SternDistance:=F_BITS_TO_NUMBER(257, 249,xDecoMessage );
   out_Mensaje5OPC[6]:=TO_STRING(ul_SternDistance,'{0}');
```

Fig. 2.28. Código ejemplo obtención datos mensaje tipo 5. Parte 1.

```
ul_LarboardDistance:=F_BITS_TO_NUMBER(263, 258,xDecoMessage );
out_Mensaje5OPC[7]:=TO_STRING(ul_LarboardDistance,'{0}');
ul_StarboardDistance:=F_BITS_TO_NUMBER(269, 264,xDecoMessage );
out_Mensaje5OPC[8]:=TO_STRING(ul_StarboardDistance,'{0}');
ul_ElectronicDeviceType:=F_BITS_TO_NUMBER(273, 270,xDecoMessage );
out_Mensaje5OPC[9]:=TO_STRING(ul_ElectronicDeviceType,'{0}');
ul_ETAMonth:=F_BITS_TO_NUMBER(277, 274,xDecoMessage );
out_Mensaje5OPC[10]:=TO_STRING(ul_ETAMonth,'{0}');
ul_ETADay:=F_BITS_TO_NUMBER(282, 278,xDecoMessage );
out_Mensaje5OPC[11]:=TO_STRING(ul_ETADay,'{0}');
ul_ETAHour:=F_BITS_TO_NUMBER(287, 283,xDecoMessage );
out_Mensaje5OPC[12]:=TO_STRING(ul_ETAHour,'{0}');
ul_ETAMinutes:=F_BITS_TO_NUMBER(293, 288,xDecoMessage );
out_Mensaje5OPC[13]:=TO_STRING(ul_ETAMinutes,'{0}');
ul_StaticDraft:=F_BITS_TO_NUMBER(301, 294,xDecoMessage );
out_Mensaje5OPC[14]:=TO_STRING(ul_StaticDraft,'{0}');
s_Destination:=F_BITS_TO_CARACTER(421, 302,xDecoMessage );
out_Mensaje5OPC[15]:=s_Destination;
ul_DTE:=F_BITS_TO_NUMBER(422, 422,xDecoMessage );
ul_Reservation:= F_BITS_TO_NUMBER(423, 423,xDecoMessage );
```

Fig. 2.29. Código ejemplo obtención datos mensaje tipo 5. Parte 2.

Cálculo distancia y demora entre buques

En primer lugar, se obtuvieron las coordenadas polares a partir de las coordenadas cartesianas (latitud, longitud) obtenidas del mensaje AIS.

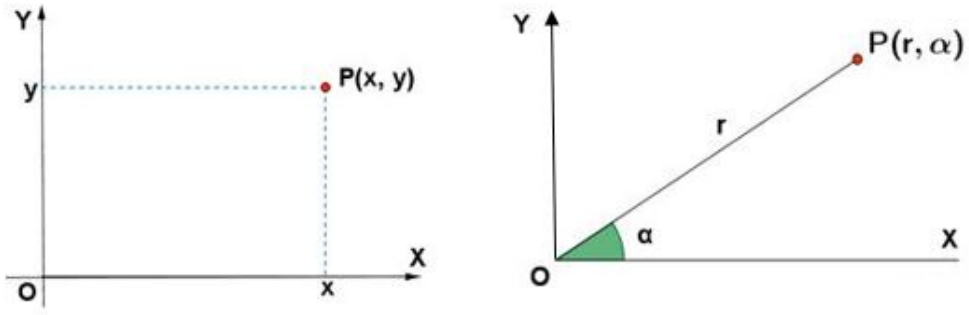


Fig. 2.30. Coordenadas cartesianas y polares.

Para obtener los componentes del vector correspondiente en polares (módulo, ángulo) a

partir de la posición en coordenadas cartesianas se crearon dos funciones. El módulo se corresponde con el valor de la distancia del punto P al origen de coordenadas (r) y el ángulo al que forma el segmento OP con la parte positiva del eje X (α): *F_CARTESIAN_TO_POLAR_MOD* y *F_CARTESIAN_TO_POLAR_ANG*. Esta última calcula el ángulo de las coordenadas polares a partir de los valores cartesianos obtenidos y está medido respecto al norte, en el sentido de las agujas del reloj.

Cálculo velocidad y rumbo relativo entre buques

A partir de dos posiciones en coordenadas polares (módulo, ángulo) se obtienen los valores de velocidad/rumbo o distancia/demora (bloque funcional *FB_RELATIVE_VECTOR*):

- Si ambas posiciones corresponden a ecos de un mismo buque, en dos instantes t1 y t2, se determina la velocidad (módulo/(t2-t1)) y rumbo (ángulo) efectivos del buque.
- Si corresponden a posiciones del buque propio (X1,Y1) y de un blanco (X2,Y2) en el mismo instante, se determina la distancia (módulo) y la demora (ángulo) de éste.

Cálculos de CPA y TCPA

El bloque funcional *FB_CALCULATE_CPA_TCPA* implementado devuelve CPA y TCPA (en segundos) de un blanco, dando distancia, demora, velocidad relativa y rumbo relativo. Un valor de -1.0 de TCPA indica que el blanco se está alejando.

2.2. Comunicación servidor web (Python) - PLC

La tecnología PLCnext cuenta con un servidor OPC-UA integrado. OPC-UA se emplea ya en la actualidad como estándar de comunicación prioritario en las instalaciones ya que habilita a la industria para las tendencias futuras de fabricación basadas en conceptos como Internet de las Cosas (IoT) y la Industria 4.0. Por esta razón, se ha optado por implementar este nuevo estándar de comunicación entre el servidor Python y el controlador lógico programable [21].

En primer lugar, a través de la herramienta *PLCnext Engineer 2020.0* se ha configurado la visibilidad de las variables del servidor OPC integrado en PLCnext para servir las variables deseadas. Particularmente se han definido las siguientes variables OPC:

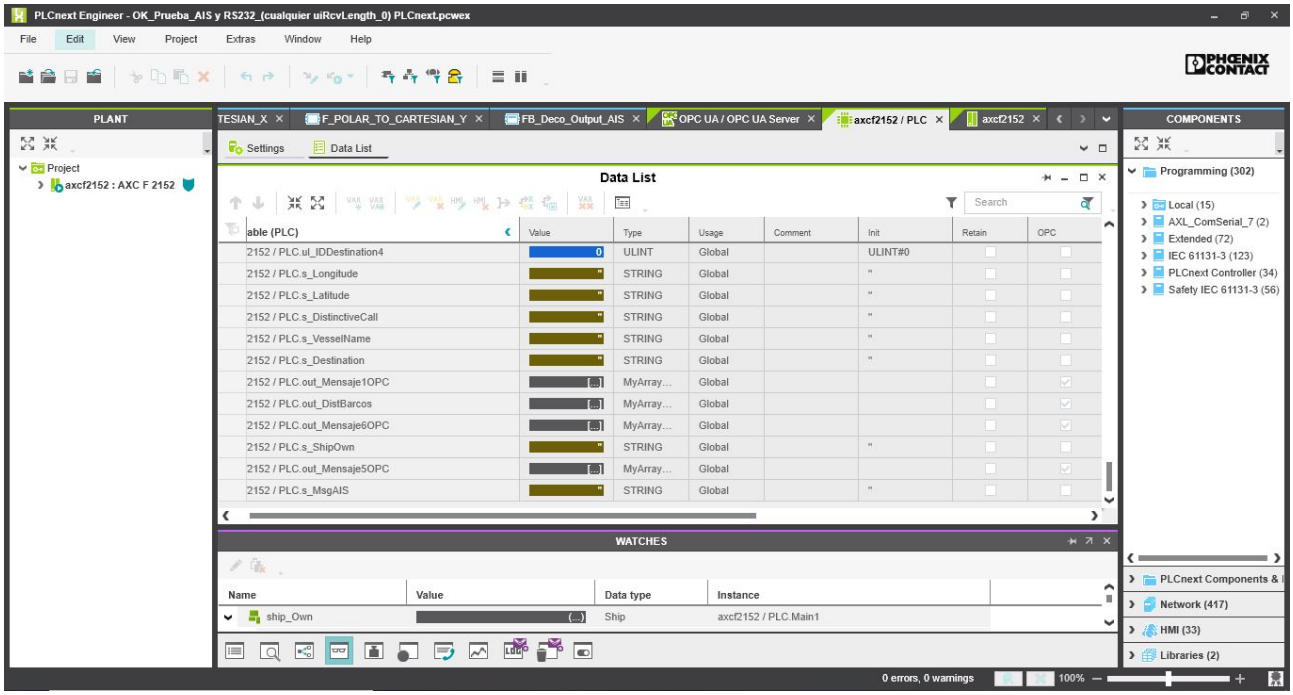


Fig. 2.31. Configuración variable OPC.

Por otra parte, en el servidor Flask implementado con Python se ha configurado el cliente OPC mediante el uso de la librería *Python OPC-UA / IEC 62541 Client and Server Python 2, 3* [22].

```
from opcua import Client


//conexión servidor OPC
client = Client("opc.tcp://192.168.1.10:4840") //ip PLC
client.set_user('admin') //usuario PLC
client.set_password('ee6caee1') //contraseña PLC
client.set_security_string("Basic256Sha256,SignAndEncrypt,certificate-example.der,private-key-example.pem") //configuración modo conexión segura
client.connect()
```

.

Fig. 2.32. Conexión cliente - servidor OPCUA

TABLA 2.14. VARIABLES OPC CREADAS.

Variable	**Tipo**	**Descripción**
out_Mensaje1OPC	ARRAY[0..20] OF STRING	Datos relevantes obtenidos de un mensaje tipo 1 (ID del mensaje, MMSI, estado navegación, ROT, SOG, longitud, latitud, COG, heading). Tablas 2.2, 2.3, 2.4.
out_Mensaje5OPC	ARRAY[0..20] OF STRING	Datos relevantes obtenidos de un mensaje tipo 5 (MMSI,número OMI, distintivo llamada,nombre buque, tipo de buque y tipo de carga, distancia ref-proa, distancia ref-popa, distancia ref-babor, distancia ref-estribor, tipo de dispositivo electrónico de determinación de posición, ETA (Hora estimada de llegada) mes, ETA - día, ETA - hora, ETA - minutos, calado estático actual máximo, destino). Tablas 2.5, 2.6, 2.7.
out_Mensaje6OPC	ARRAY[0..20] OF STRING	Datos relevantes obtenidos de un mensaje tipo 6 (ID del mensaje, ID origen, ID destino, ID función). Tabla 2.10.
out_DistBarcos	ARRAY[0..20] OF STRUCT MMSI_O:STRING; MMSI_T:STRING; DIST:REAL; BEAR:REAL; DIST_PREV:REAL; BEAR_PREV:REAL; REL_COURSE:REAL; REL_SPEED:REAL; CPA:REAL; TCPA:REAL; END_STRUCT	Cálculos obtenidos de distancia, demora, velocidad relativa, rumbo relativo, CPA y TCPA entre el buque propio y el resto de targets.

Para permitir la conexión con el servidor se ha especificado la dirección IP del controlador lógico programable que se ha fijado manualmente en su configuración (Figura 2.33)

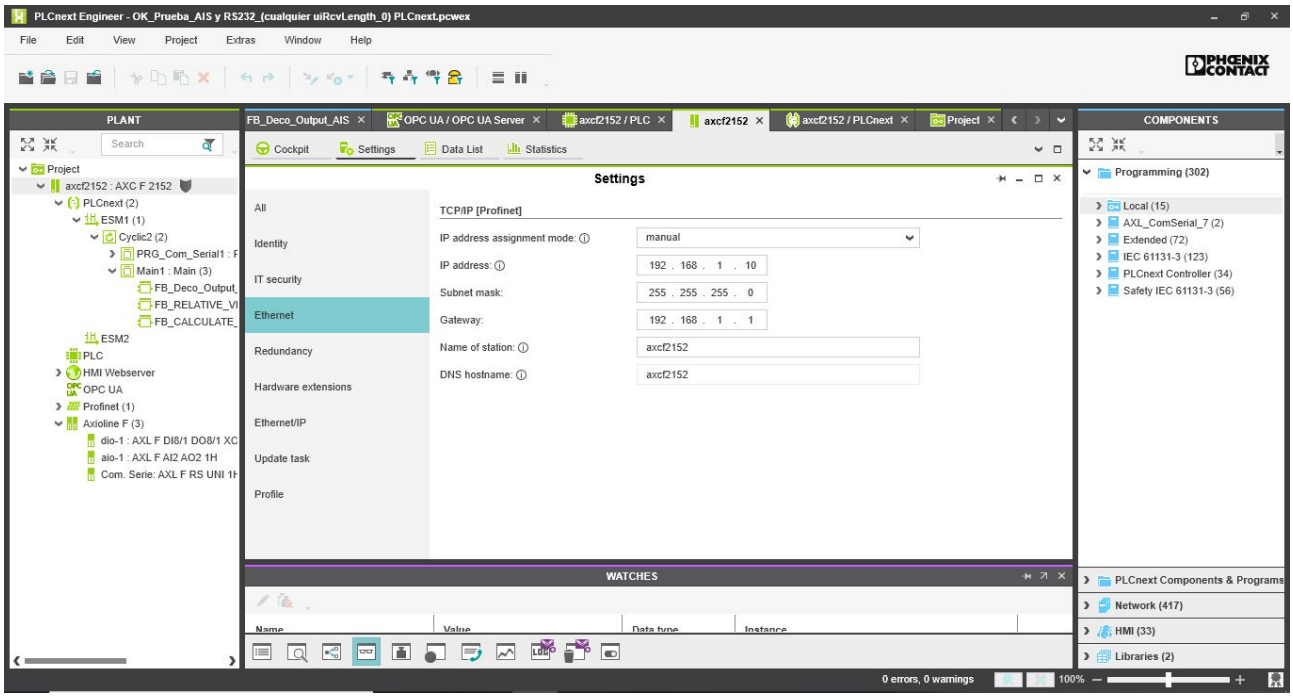


Fig. 2.33. Configuración TCP/IP - *PLCnext Control AXC F 2152*.

Y, como parámetros de conexión, se ha especificado el usuario y contraseña del controlador lógico programable que suele venir especificado en el módulo físico. Además, para permitir una conexión segura se ha especificado la política de seguridad utilizada (*Basic 256 Algorithm*) y el modo *SignAndEncrypt*. En este modo de conexión segura, el cliente debe enviar un certificado de instancia de aplicación para identificarse ante los socios de comunicación y una clave privada que se utiliza para firmar y / o cifrar mensajes (rutas a archivos .pem o .der). Cuando cliente y servidor OPC-UA establecen un canal seguro, intercambian una clave simétrica, la misma tanto para el cliente como para el servidor, y continúan encriptando mensajes con las claves simétricas.

Una vez establecido un canal de comunicación seguro, se procede a leer y acceder a las variables configuradas en el autómata a través del árbol de objetos del servidor.

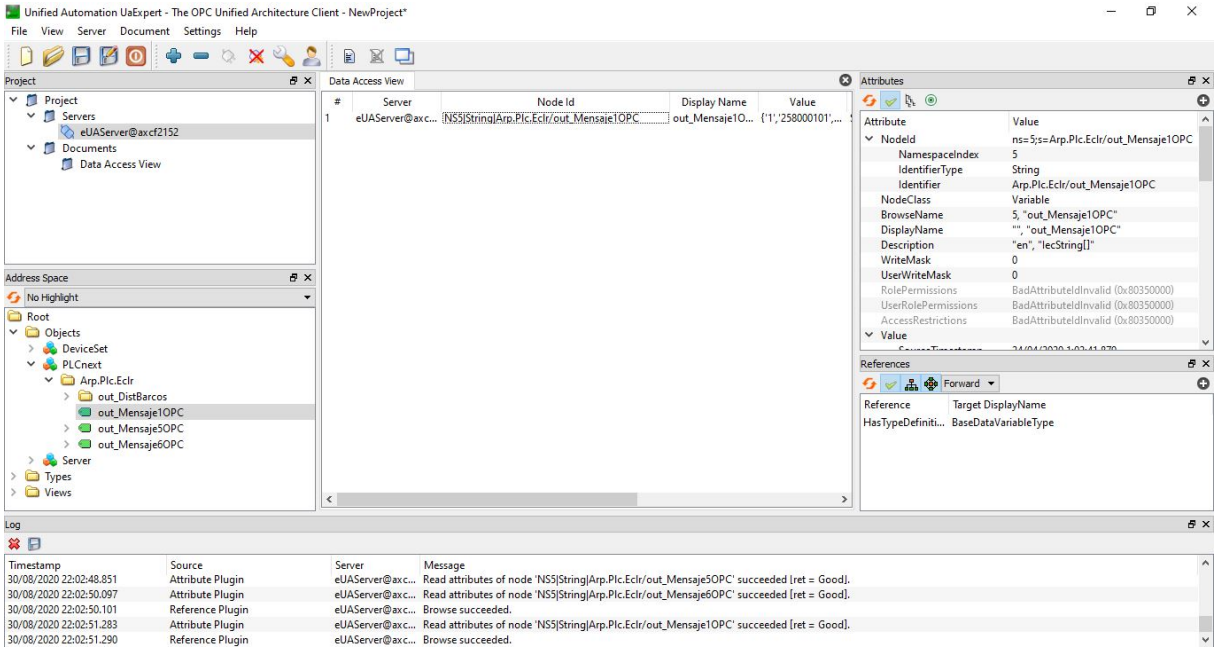


Fig. 2.34. Árbol de objetos del servidor.Nodo variable *out_Mensaje1OPC*.

```
//obtención variable out_Mensaje1OPC del árbol de objetos servidor
root = client.get_root_node()
objects = root.get_child(['0:Objects'])
barco=client.get_node("ns=5;s=Arp.Plc.Eclr/out_Mensaje1OPC")
```

Fig. 2.35. Obtención variable *out_Mensaje1OPC*.

2.3. Comunicación servidor web (Python) - Base de datos (mySQL)

Los intercambios de mensajes, además de los datos dinámicos deben ser almacenados por el sistema PE de cada buque, de forma que posteriormente puedan ser consultados. Por esta razón, se ha utilizado *MySQL (ORACLE, 2010)* como sistema de gestión de base de datos (SGBD). Se trata de una sistema de código abierto que funciona con un modelo cliente-servidor. Eso quiere decir que los ordenadores que instalan y ejecutan el software de gestión de base de datos se denominan clientes. Cada vez que necesitan acceder a los datos, los clientes se conectan al servidor del sistema de gestión de base de datos y le solicitan la información que necesitan.

Para que todo funcione correctamente, fue necesario crear un servidor MySQL local. En este proyecto se ha optado por utilizar la herramienta *XAMPP*. Se trata de un servidor independiente de plataforma de código libre que incluye servidores de bases de datos como MySQL.

Para poder acceder a la base de datos MySQL creada en este proyecto, *barcos.db*, se ha utilizado la extensión *Flask-MySQL* del servidor Python que permite trabajar con bases de datos MySQL [23]. Para establecer la conexión cliente-servidor MySQL se han definido las siguientes configuraciones de acuerdo a la configuración por defecto del servidor MySQL creado:

```
from flaskext.mysql import MySQL


mysql = MySQL()
app.config['MYSQL_DATABASE_USER'] = 'root'
app.config['MYSQL_DATABASE_PASSWORD'] = ''
app.config['MYSQL_DATABASE_DB'] = 'barcos'
app.config['MYSQL_DATABASE_Host'] = 'localhost'
mysql.init_app(app)
conn = mysql.connect() \\conexión
cursor =conn.cursor()
```

Fig. 2.36. Conexión base de datos MySQL.

2.4. Comunicación cliente y servidor web (Python)

Dado que, la interfaz gráfica se ha creado mediante Javascript y HTML, para permitir el intercambio de información entre el servidor Python y el cliente web definido con Javascript y HTML se ha optado por utilizar peticiones HTTP *Hypertext Transfer Protocol* [24] con el fin de proporcionar a la interfaz de usuario toda la información recibida de las estaciones AIS que dispone el servidor. Particularmente se han utilizado:

TABLA 2.15. MÉTODOS HTTP UTILIZADOS.

Método	**Descripción**
GET	El método más común. Se envía un mensaje GET y el servidor devuelve datos.
POST	Se utiliza para enviar datos de formularios HTML al servidor. El servidor no almacena en caché los datos recibidos por el método POST.

Para manejar solicitudes GET y POST en el servidor Python se ha utilizado el método app.route(). Este método debe recibir la URL de una ruta como parámetro y los métodos HTTP aceptados para dicha ruta. Por otra parte, para que el cliente Javascript pueda realizar este tipo de solicitudes al servidor, se ha decidido utilizar métodos *jQuery AJAX* [25].

TABLA 2.16. MÉTODOS JQUERY AJAX UTILIZADOS.

Método	**Descripción**
$ .get ()	Carga datos de un servidor mediante una solicitud HTTP GET de AJAX.
$ .post ()	Carga datos de un servidor mediante una solicitud POST HTTP AJAX.

3. HMI

El HMI (*Human Machine Interface*) muestra la información acerca de los buques que, dentro del radio dado, presentan unos valores de CPA y TCPA relativos al buque propio menores que los valores de seguridad dados. La idea principal para este proyecto fue que desde este HMI el operador pudiese modificar los valores establecidos para el radio de visualización, periodo de refresco de información, distancias y tiempos mínimos. Por esta razón, los datos que debe suministrar el operador se resumen en la Tabla 3.1 y la información a mostrar respecto a cada buque (target), en la Tabla 3.2.

TABLA 3.1. DATOS A PROPORCIONAR POR EL OPERADOR.

Parámetro	**Descripción**
LRS	Radio del universo visible, distancia para empezar a analizar encuentros.
CPASafe	Acercamiento mínimo considerado seguro.
TCPASafe	Tiempo considerado seguro mínimo para CPA.
Dist. PreAlert	Distancia para iniciar las maniobras si el buque no debe impedir el paso del otro buque.
Dist. Alert	Distancia para iniciar las maniobras si el buque propio es un buque cuyo paso no debe impedirse, en caso de que el otro buque no actúe correctamente.
Storage Time (mseg)	Tiempo establecido para actualizar la base de datos en situaciones seguras.
Storage Time Alert (mseg)	Tiempo establecido para actualizar la base de datos en situaciones de pre-alerta y alerta.

TABLA 3.2. INFORMACIÓN A MOSTRAR.

Parámetro	**Descripción**
Nombre MMSI	Nombre e identificador del target, extraídos del último mensaje estático recibido del mismo.
Latitude Longitude Heading SOG ROT	Datos extraídos del mensaje AIS con la última información dinámica del target: posición (latitud y longitud), rumbo de proa, velocidad sobre el fondo, velocidad de giro.
Distance	Distancia.
Bearing	Demora.
Relative speed	Velocidad relativa.
Relative Course	Rumbo relativo.
CPA	CPA.
TCPA	TCPA.

3.1. Características

Para la creación de la interfaz gráfica web se ha optado por emplear Javascript y HTML con el fin de ser compatible con el servidor Flask de Python en la transferencia de información relevante sobre los buques para su representación gráfica y permitir crear un contenido dinámico y más atractivo.

Cualquier página web está construída por HTML y CSS (un lenguaje de estilos). El primero de ellos es un lenguaje de marcas que permite construir todo el marcado de la página (contenido e información) mediante etiquetas HTML y, el segundo de ellos, es un lenguaje de estilos que permite construir una interfaz visual más agradable para el usuario dándole estilo a la página. Para interactuar con una página web y hacer el contenido de la misma más dinámico, se provee al lenguaje JavaScript que se trata de un lenguaje de secuencias de comandos que permite implementar funciones complejas en páginas web.

Particularmente, se han utilizado diferentes bibliotecas de Javascript para el desarrollo de una visualización web interactiva:

- **Leaflet.js:** biblioteca de código abierto que permite la creación de aplicaciones con mapas interactivos. En este trabajo se ha utilizado para crear un mapa y poder localizar en él

los diferentes buques, agregando marcadores con textos emergentes que muestran información relevante sobre ellos como el MMSI. Para ello, se ha utilizado como partida un ejemplo de código que permite la creación de un mapa de este tipo [26].

- **Heatmap.js:** biblioteca de JavaScript que permite crear mapas de calor basados en HTML5. En este trabajo, para la creación de las capas de calor sobre el mapa geográfico, se ha utilizado como código de partida un sencillo plugin de mapa de calor de Leaflet que utiliza una biblioteca de JavaScript pequeña, *simpleheat.js*, para dibujar mapas de calor con Canvas. Está inspirada por *heatmap.js* pero con un enfoque en la simplicidad y el rendimiento [27].

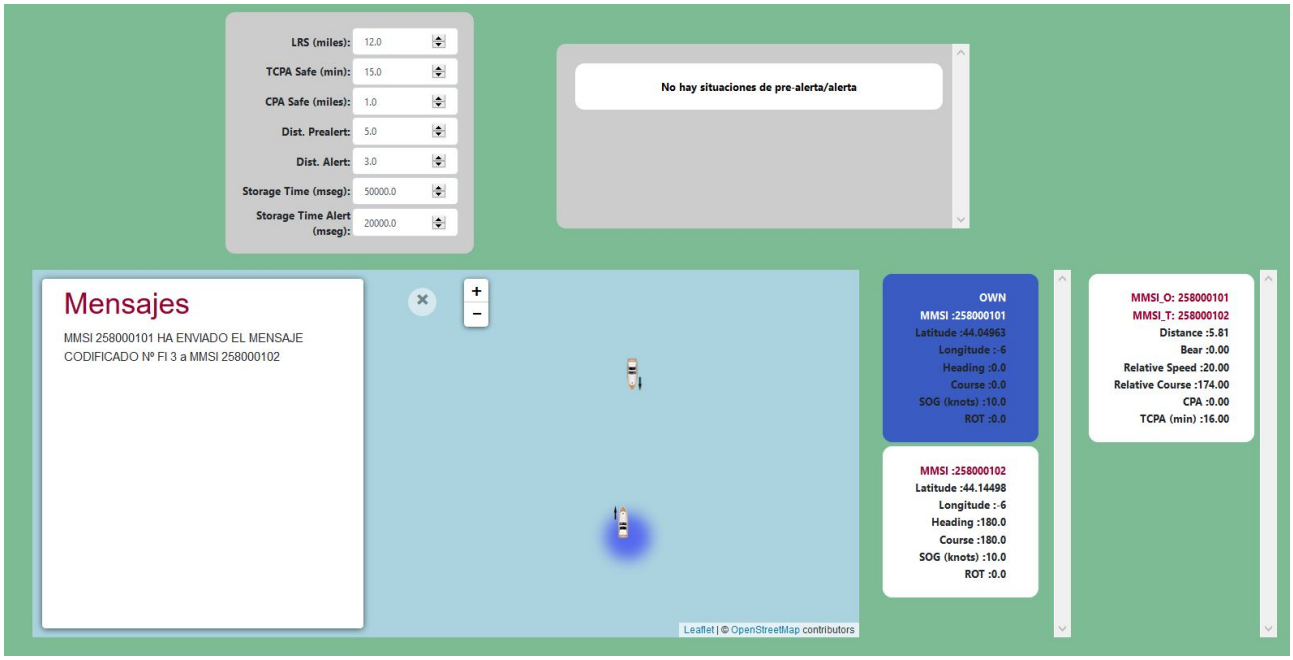


Fig. 3.1. Interfaz gráfica creada.

En la parte superior izquierda se ha definido un formulario con los datos que el usuario debe proporcionar según la Tabla 3.1. Y, en la parte derecha, se ha definido una zona en la que se mostrarán mensajes de aviso para el usuario sobre qué buques están en situación de pre-alerta y/o alerta (Tabla 3.3).

En el caso de que no se produzcan ninguna de las dos situaciones definidas, el sistema estará en una situación segura y se mostrará el mensaje: " *No hay situaciones de pre-alerta/alerta*".

TABLA 3.3. CONDICIONES SITUACIONES PRE-ALERTA Y ALERTA.

Situación	**Condición**
Pre-alerta	*CPA <= CPA Safe AND (Distancia <= Dist. Prealert (formulario) OR TCPA <= TCPA Safe).*
Alerta	*Prealerta AND Distancia <= Dist. Alerta (formulario).*

```
<form action = "http://127.0.0.1:5000//bbdd_dist" method = "post" id="dataForm"
style="width: 270px">

    <table><tr>

        <td align="right"><b>LRS (miles):</b></td>

        <td align="left"> <input id="LRS" name="LRS" class="form-control" type="number"
value="12.0" style="font-size: 11px"></td></tr>

        <tr><td align="right"><b>TCPA Safe (min):</b></td>

        <td align="left"> <input id="TCPA" name="TCPA" class="form-control" type="number"
value="20.0" style="font-size: 11px"></td></tr>

        <tr><td align="right"><b>CPA Safe (miles): </b></td>

        <td align="left"><input id="CPA" name="CPA" class="form-control" type="number"
value="1.0" style="font-size: 11px"></td></tr>

        <tr><td align="right"><b>Dist. Prealert: </b></td>

        <td align="left"><input id="Prealert" name="Prealert" class="form-control"
type="number" value="5.0" style="font-size: 11px"></td></tr>

        <tr><td align="right"><b>Dist. alert:</b></td>

       <td align="left"><input id="alert" name="alert" class="form-control" type="number"
value="3.0" step="0.1" style="font-size: 11px"></td> </tr>
```

Fig. 3.2. Código creación formulario HTML. Parte 1.

```
        <tr><td align="right"><b>Storage Time (mseg):</b></td>
        <td align="left"><input id="storageTime" name="storageTime" class="form-control"
type="number" value="50000.0" step="1.0" style="font-size: 11px"></td> </tr>
        <tr><td align="right"><b>Storage Time Alert (mseg):</b></td>
        <td align="left"><input id="storageTimeAlert" name="storageTimeAlert" class="form-
control" type="number" value="20000.0" step="1.0" style="font-size: 11px"></td></tr>
    </table>
</form>
```

Fig. 3.3. Código creación formulario HTML. Parte 2.

En la parte inferior de la interfaz gráfica mostrada en la Figura 3.1, se ha creado un mapa geográfico mediante el uso de la biblioteca *Leaflet.js* que ha permitido mostrar la ubicación de los diferentes buques en el mapa agregando marcadores con textos emergentes con el fin de mostrar cierta información sobre ellas como el número de identificación del servicio móvil marítimo (MMSI). Inicialmente se ha decidido centrar el mapa en Gijón para hacer este caso más real, por lo que se ha especificado la latitud y longitud de esta ubicación como se muestra en la Figura 3.4.

Sobre el mapa se han añadido diferentes mejoras para que la visualización sea más intuitiva. En primer lugar, para mostrar el rumbo y heading de cada buque y poder entender mejor la evolución de los movimientos que van a experimentar cada uno de ellos, se ha girado cada marcador en función del heading obtenido de los mensajes AIS de tipo 1 y, se ha añadido una flecha adicional a cada marcador que indica el rumbo del mismo. En el mapa. para poder diferenciar a simple vista cuál es el buque propio (OWN) a partir del cuál se calcularán los valores de distancia, demora, CPA y TCPA con respecto al resto de targets, se ha añadido una capa de calor de color azul sobre dicho buque. Y, para diferenciar qué buques están en situaciones de pre-alerta o alerta (Tabla 3.3) se han añadido capas de calor sobre los buques implicados:

TABLA 3.4. CAPAS DE CALOR EN FUNCIÓN DE LAS SITUACIONES PRE-ALERTA Y ALERTA.

Situación	**Capa de calor**
Pre-alerta	Naranja
Alerta	Rojo

```
map = L.map('map').setView([43.540635, -5.662464], 11);

var tiles = L.tileLayer('http://{s}.tile.osm.org/{z}/{x}/{y}.png', { attribution: '&copy; <a
href="http://osm.org/copyright">OpenStreetMap</a> contributors',}).addTo(map);

 var markersGroup = new L.LayerGroup();

//a partir de los datos (latitud, longitud) obtenidos para cada buque se añaden marcadores

//y se gira en función del heading

var boat=L.marker([datos[i]["lat"],datos[i]["lon"]], {icon: barcoIcon, rotationAngle:
datos[i]["HDG"]})

arrayMarkers.push(boat);

//si es el barco propio (OWN) se añade los datos en un recuadro azul para destacar respecto al
resto

if(datos[i]["MMSI"]==barcoOwn) {

    boat.bindPopup('<p><font color="blue"><b>MMSI: </b>'+datos[i]["MMSI"]+'</font></p>',
{maxWidth:100, closeOnClick: true});

//si es un target se añade un recuadro blanco con los datos correspondientes

} else{

    boat.bindPopup('<p><b>MMSI: </b>'+datos[i]["MMSI"]+'</p>', {maxWidth:100,
closeOnClick: true});

}
```

Fig. 3.4. Código creación mapa y marcadores.

```
markersGroup.addLayer(boat).addTo(map);
//si es el barco propio (OWN) se añade la capa de calor azul
if(datos[i]["MMSI"]==barcoOwn)
{
    datalayer=[];
    datalayer.push([datos[i]["lat"],datos[i]["lon"], 0.8])
    var heat = L.heatLayer(datalayer, {radius: 25, minOpacity:0.5, gradient:{0.5: 'blue'}});
     markersGroup.addLayer(heat).addTo(map);
 }
```

Fig. 3.5. Código capa de calor sobre el buque propio (OWN).

```
for (var j = 0; j < datosDist.length; j++) {
   for (var i = 0; i < datos.length; i++) {
        //se comprueba si no son datos innecesarios
        if(datosDist[j]["CPA"]>0 && datosDist[j]["CPA"]< parseFloat(CPA)){
            //si se produce una situación de alerta
            if(parseFloat(datosDist[j]["Dist"])<=parseFloat(DistAlert) &&
parseFloat(datosDist[j]["Dist"])<=parseFloat(DistPreAlert) && parseFloat(datosDist[j]["CPA"])<=parseFloat(CPA)){
                situationPreAlert=true;
                datalayer=[];
                if(parseInt(datos[i]["MMSI"])==parseInt(datosDist[j]["MMSI_O"])
||parseInt(datos[i]["MMSI"])==parseInt(datosDist[j]["MMSI_T"])){
                    //se añade una capa de calor sobre los buques implicados de color rojo
                    datalayer.push([datos[i]["lat"],datos[i]["lon"], 0.8])
                }
                var heat = L.heatLayer(datalayer, {radius: 25, minOpacity:0.5, gradient:{0.5: 'red'}});
                markersGroup.addLayer(heat).addTo(map);
        }
```

Fig. 3.6. Código capas de calor en situaciones de pre-alerta y alerta. Parte 1.

```
        // si se produce una situación de pre-alerta
        else if(parseFloat(datosDist[j]["Dist"])<=parseFloat(DistPreAlert) &&
parseFloat(datosDist[j]["CPA"])<=parseFloat(CPA)){
            situationPreAlert=true;
            var contenidoAlert='';
            datalayer=[];
             if(parseInt(datos[i]["MMSI"])==parseInt(datosDist[j]["MMSI_O"])||
parseInt(datos[i]["MMSI"])==parseInt(datosDist[j]["MMSI_T"])){
                datalayer.push([datos[i]["lat"],datos[i]["lon"], 0.0])
             }
             // se añade una capa de calor naranja
             var heat = L.heatLayer(datalayer, {radius: 25, minOpacity:0.5, gradient:{0.5: 'orange'}});
              markersGroup.addLayer(heat).addTo(map);
        }
    }
```

Fig. 3.7. Código capas de calor en situaciones de pre-alerta y alerta. Parte 2.

Además, en la parte izquierda del mapa se ha añadido un diálogo, llamado *Mensajes*, en el que aparecerán los mensajes AIS tipo 6 intercambiados entre el operador propio y los demás buques especificando el origen del mensaje, destino del mensaje e ID de la función a enviar (Tabla 2.10).

```
function ReadMsgExchange(){
    //petición HTTP para la obtención datos mensaje AIS tipo 6 al servidor Python
    $.get("/read_msj_exchange", function(data) {
        console.log(data);
        mnsj=data;
        var date= new Date();
        //si se ha recibido algún mensaje AIS tipo 6 se escribe en el diálogo Mensajes
        if(mnsj!=undefined){
            mensajesExchange='';
            for (var i = 0; i < mnsj.length; i++) {
                if(mnsj[i]["ID"]!=0){
                    mensajesExchange=mensajesExchange.concat('<p>MMSI '+ mnsj[i]["MMSI_ORI"]+' HA ENVIADO EL
MENSAJE CODIFICADO Nº FI '+ mnsj[i]["ID_FUNC"]+' a MMSI '+mnsj[i]["MMSI_DEST"]+'</p>')}
            }
            sidebar.setContent('<h1 style="color:#990033">Mensajes</h1>' + mensajesExchange)
            sidebar.show()}
    })
}
```

Fig. 3.8. Código inserción datos mensaje AIS tipo 6 en diálogo *Mensajes*.

En la parte inferior derecha de la interfaz gráfica, se han añadido dos bloques de texto. El primero de ellos muestra información importante de cada buque (MMSI, latitud (grados), longitud (grados), heading, rumbo, velocidad sobre el fondo - efectiva (SOG) y velocidad de giro (ROT)). Además, para diferenciar rápidamente los datos asociados al buque propio del resto de targets, se ha creado este cuadro de texto en color azul como se mostrará más adelante. El segundo bloque muestra los cálculos obtenidos de distancia, demora, velocidad relativa, rumbo relativo, CPA y TCPA entre el buque propio y cada target. Para evitar distraer al operador con datos innecesarios, se ha optado por ocultar esta última información cuando se cumplen las siguientes condiciones al mismo tiempo:

- Distancia >LRS.
- CPA >CPASafe.

3.2. Ejemplos

Tras la breve descripción de las diferentes funcionalidades que se pueden encontrar en la interfaz gráfica, se procederá a mostrar ejemplos sobre el funcionamiento de la misma ante diferentes situaciones que pueden aparecer. En primer lugar, para que la interfaz gráfica funcione correctamente es necesario ejecutar el servidor Flask de Python para poder permitir el intercambio de la información obtenida de los sistemas AIS y así poder visualizarla. Para ello, se ha utilizado el software *Anaconda* que se trata de una distribución libre y abierta que permite la ejecución de programas escritos en lenguaje Python.

El programa Python desarrollado en este proyecto para la creación del servidor Flask se denomina *WebServer.py* y para la ejecución del mismo se necesita ejecutar la siguiente instrucción en *Anaconda*:

Fig. 3.9. Ejecución servidor Python con Anaconda.

Como ejemplo, con *AIS Simulator* se han simulado señales de dos buques (se pueden simular tantos buques como se quiera, no se ha establecido ningún límite) con las siguientes características:

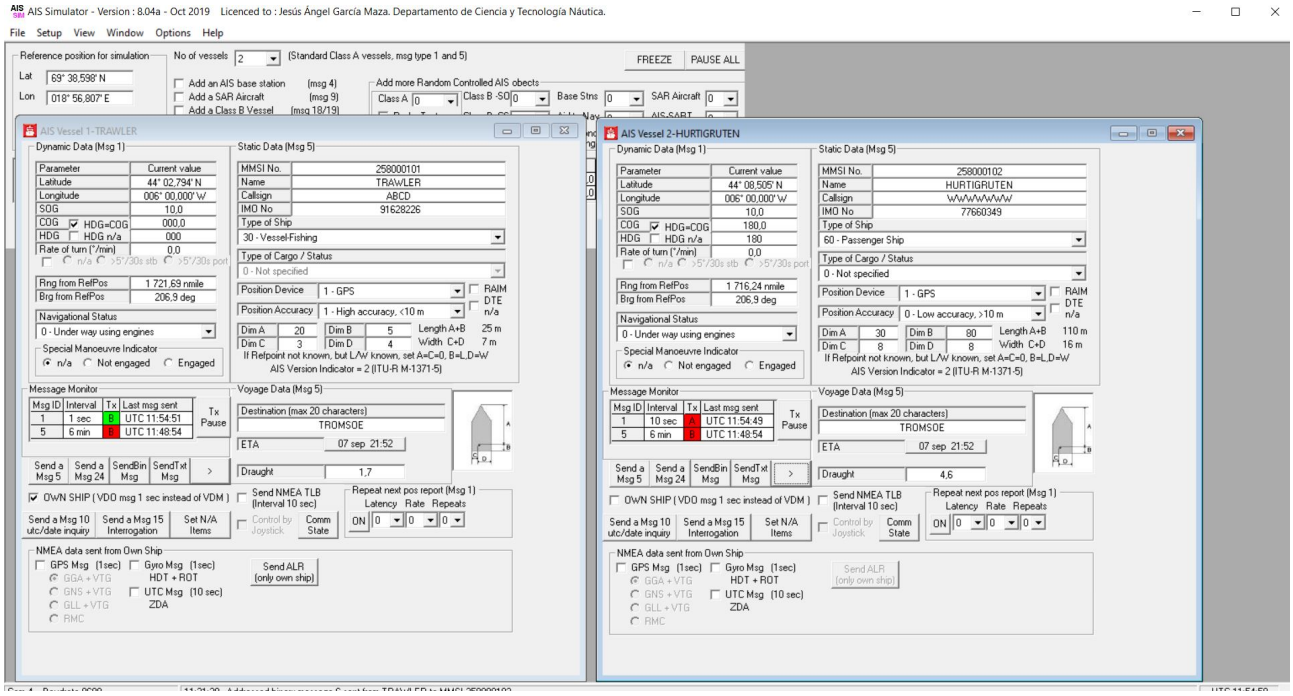


Fig. 3.10. Características buques simulados con *AIS Simulator*.

3.2.1. Situación segura

Una situación segura en la navegación marítima se da cuando se cumple la condición:

CPA >CPA Safe

En este caso, la intefaz gráfica tiene un funcionamiento normal. En el mapa se muestra el buque propio y los demás targets y, en la parte inferior derecha, la información importante asociada a cada buque y los cálculos de distancia, demora, velocidad relativa, rumbo relativo, CPA y TCPA entre el buque propio y el resto.

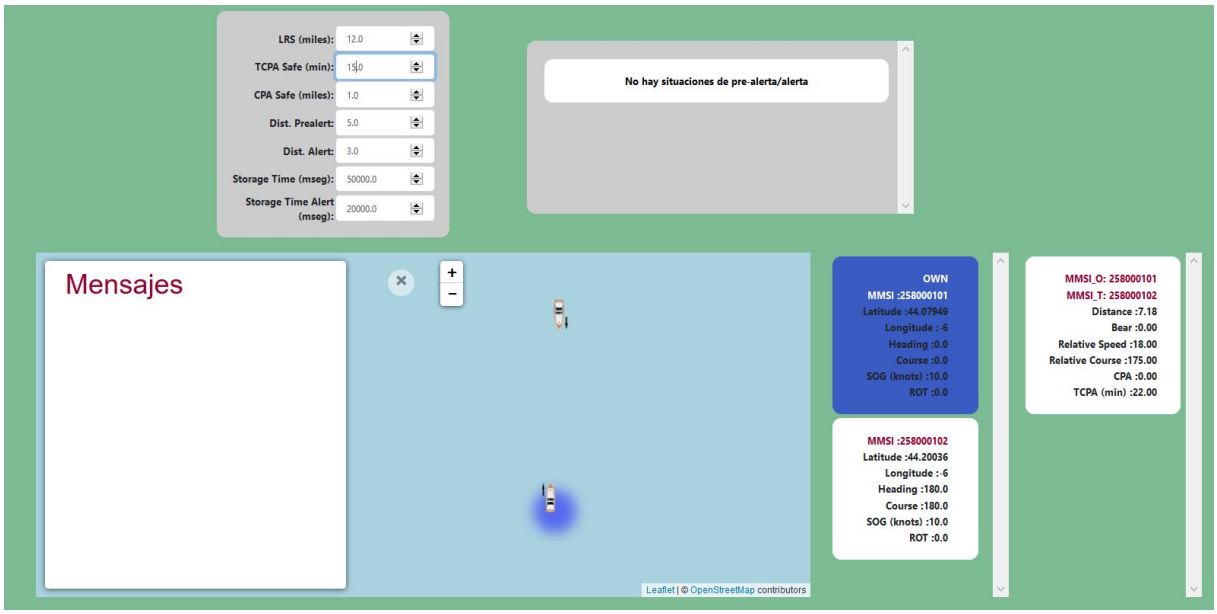


Fig. 3.11. Interfaz gráfica ante una situación segura de navegación.

3.2.2. Situación pre-alerta

La situación de pre-alerta, como ya se ha comentado anteriormente, se da cuando se cumple la condición:

CPA <= CPA Safe AND (Distancia <= Dis. PreAlert OR TCPA <= TCPA Safe)

Si se produce este tipo de situación, en la parte superior derecha de la interfaz gráfica aparecerá uno o varios mensajes en color naranja indicando qué buques son los que están implicados y, en el mapa aparecerán remarcados con un círculo naranja para que sean fácilmente identificables. Además, en la parte inferior derecha se seguirá mostrando la información importante asociada a cada buque y los cálculos obtenidos entre el propio y los targets.

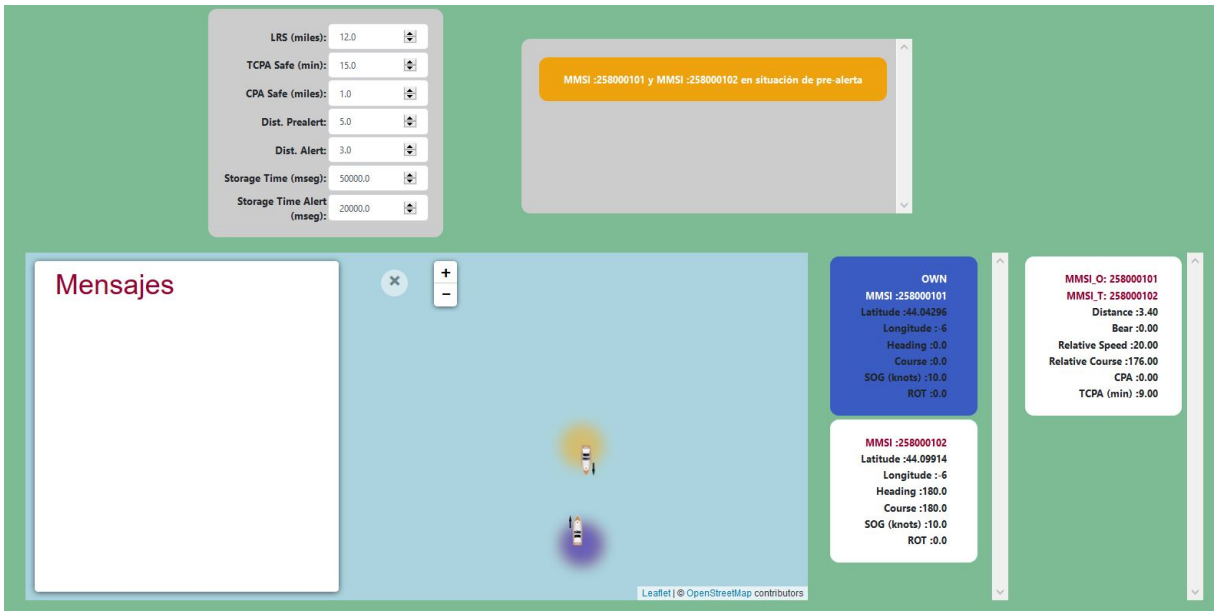


Fig. 3.12. Interfaz gráfica ante una situación de navegación de pre-alerta.

3.2.3. Situación alerta

Este penúltimo caso se produce si se cumple la siguiente condición:

Prealerta AND (Distancia <= Dist. Alert)

En este caso, parecido al anterior, en la parte superior derecha aparecerá uno o varios mensajes en color rojo indicando qué buques están implicados y, en el mapa aparecerán remarcados con un círculo rojo para que sean fácilmente identificables. Además, en la parte inferior derecha se seguirá mostrando la información importante asociada a cada buque y los cálculos obtenidos entre el propio y los targets.

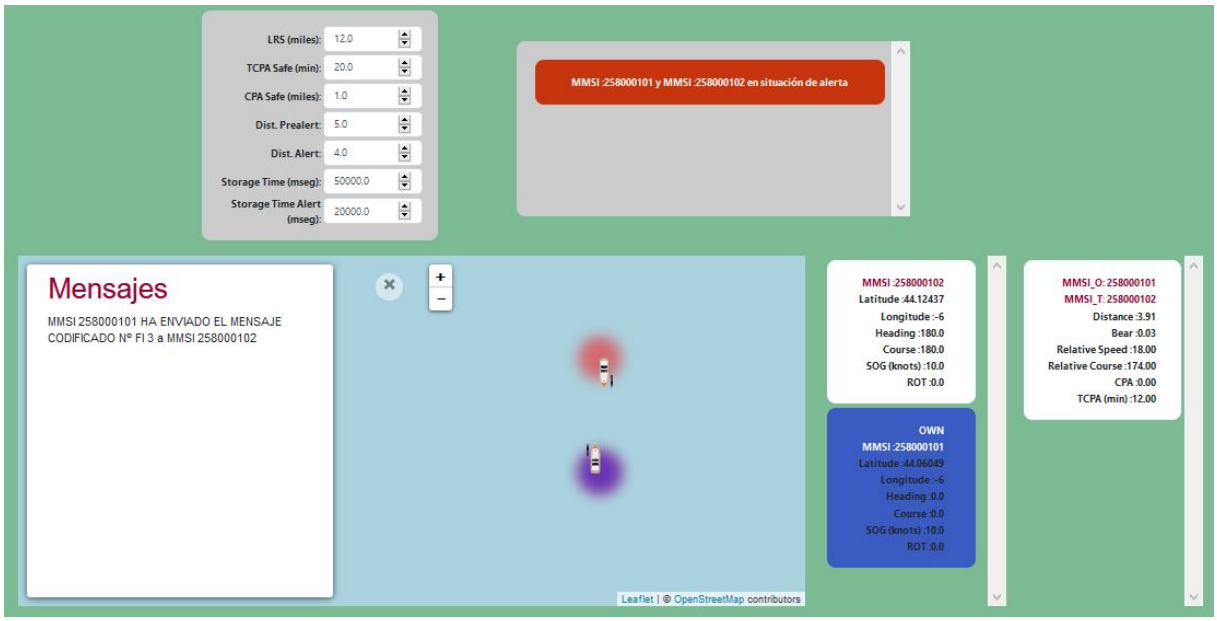


Fig. 3.13. Interfaz gráfica ante una situación de navegación de alerta.

3.2.4. Situación intercambio mensajes entre buques

Por último, en el caso de que un operador de cualquier buque intente intercambiar mensajes con otro buque (mensaje AIS tipo 6) en la interfaz gráfica del operador se mostrará el contenido de dicho mensaje en el diálogo izquierdo del mapa, llamado *Mensajes* como se muestra en la Figura 3.14.

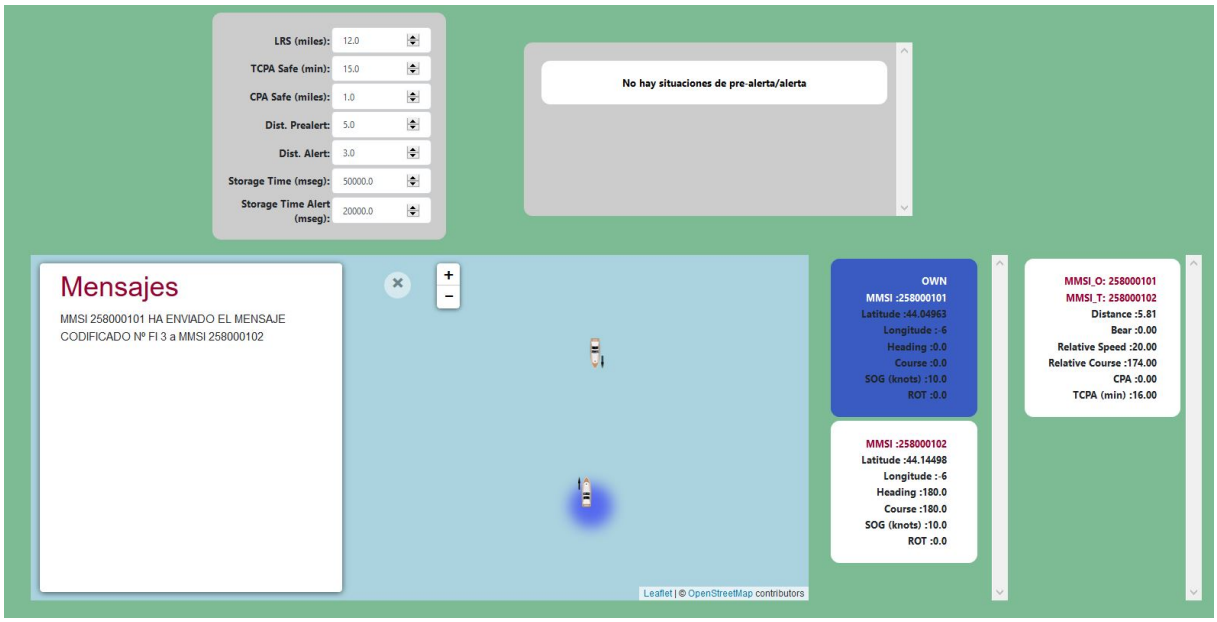


Fig. 3.14. Interfaz gráfica ante la detección de intercambio de mensajes entre buques.

4. BASE DE DATOS

Una base de datos permite almacenar gran número de información de una forma organizada para posteriormente realizar consultas, búsquedas rápidas de información o para su representación. En este trabajo la base de datos debe almacenar los datos dinámicos recibidos, datos estáticos recibidos , datos calculados a partir de los dinámicos y, por último, los mensajes intercambiados entre el operador propio y los demás buques. Todos ellos se deben almacenar tanto en situaciones seguras como en situaciones de pre-alerta y/o alerta y, en todos los casos se debe incluir el momento en el que se ha realizado dicha inserción para que, en el futuro, pueda ayudar al usuario en la búsqueda de la información.

Se ha optado por desarrollar una base de datos MySQL cuya estructura se explicará más adelante y, con el fin de poder leer la información almacenada en la base de datos y, que pueda ser representada en el futuro si el usuario lo desee se ha utilizado la aplicación *MySQL Workbench 8.0 CE* [28]. Se trata de una herramienta de diseño de bases de datos que integra desarrollo de software, administración, diseño, gestión y mantenimiento de bases de datos MySQL.

4.1. Configuración

Lógicamente, uno de los primeros pasos para crear la base de datos y obtener el esquema será conectar con el servidor MySQL. Para ello, es necesario crearlo mediante la herramienta *XAMPP* como se muestra en la Figura 4.1.

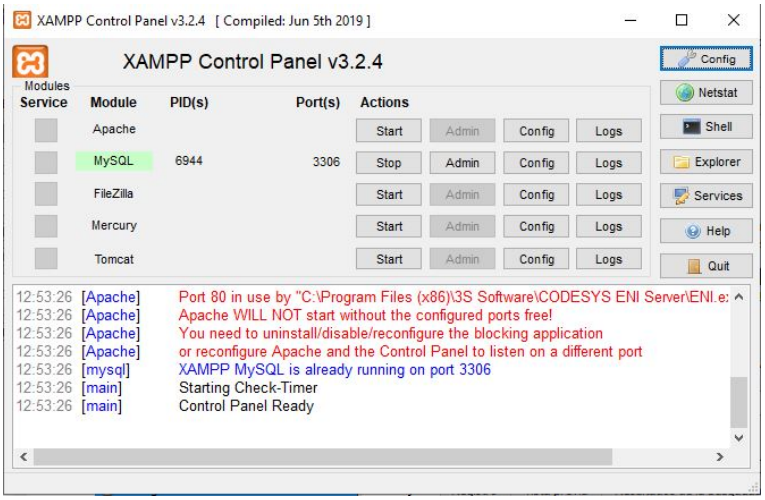


Fig. 4.1. Creación servidor MySQL local.

Una vez creado este servidor, mediante la herramienta *MySQL Workbench 8.0 CE* se debe enlazar con el mismo configurando los datos de conexión. Generalmente si se conecta a un servidor local se usa el protocolo TCP/IP, hostname: 127.0.0.1m usuario: root normalmente y la clave que tenga.

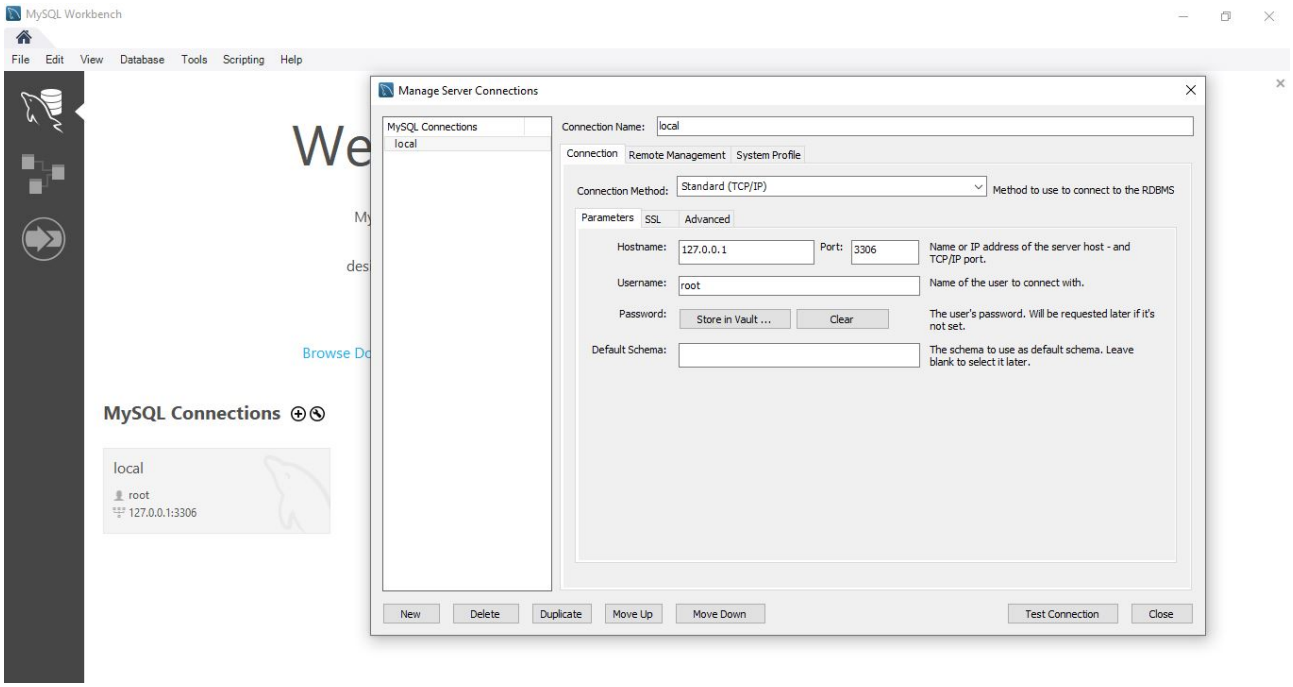


Fig. 4.2. Establecimiento conexión servidor MySQL local.

Una vez conseguida la conexión, se procede a la creación de la base de datos *barcos.db* mediante la ejecución de la siguiente query: *create database barcos;*. Otra forma para crearla es utilizar la opción de creación que presenta la aplicación *MySQL Workbench* que se muestra en la siguiente figura:

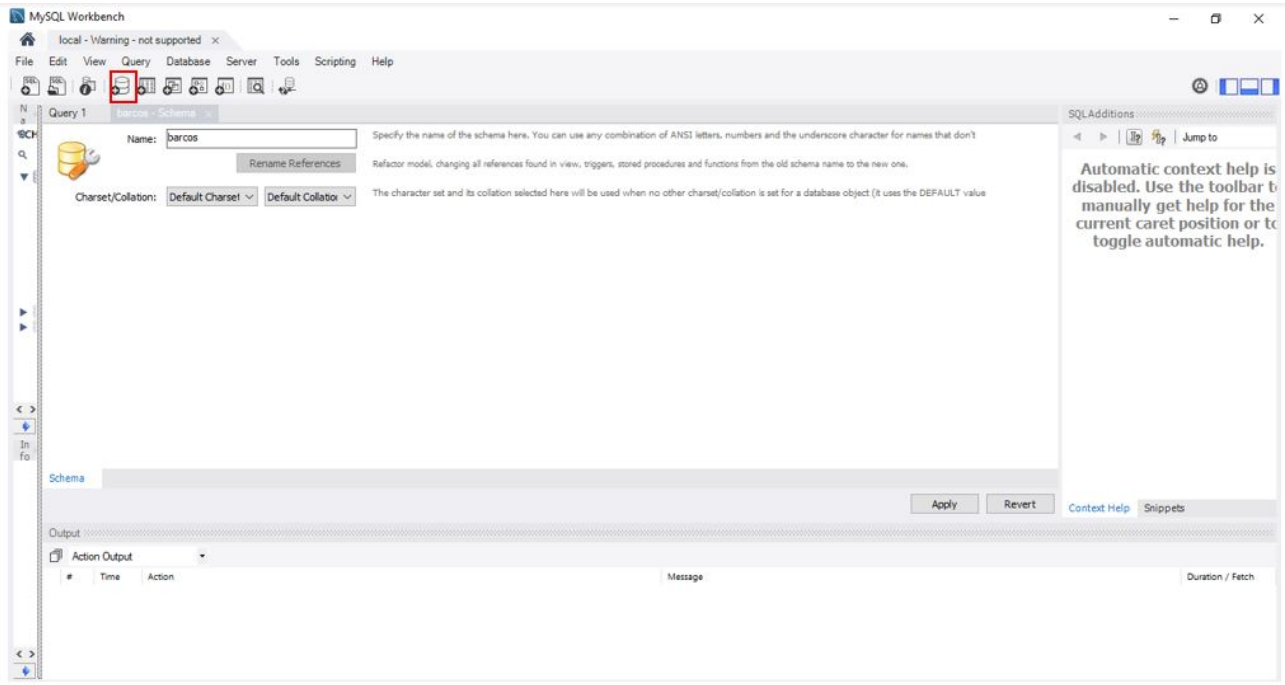


Fig. 4.3. Creación base de datos *barcos.db*.

A continuación, se ha procedido a la creación de la siguiente estructura de tablas:

- **Tabla boats:** almacenada para cada buque los siguientes datos: MMSI, latitud (grados), longitud (grados), estado de navegación, rumbo sobre el fondo - efectivo (COG), velocidad de giro (ROT), velocidad sobre el fondo - efectiva (SOG), heading (HDG) y fecha en la que se produce la inserción de los datos en la base de datos.
- **Tabla informationboats:** almacena información estática asociada a cada buque: MMSI, número OMI, distintivo de llamada, nombre del buque, tipo de buque y tipo de carga, distancia ref-proa, distancia ref-popa, distancia ref-babor, distancia ref-estribor, tipo de dispositivo electrónico de determinación de posición, fecha estimada de llegada, calado estático actual máximo, destino y fecha en la que se produce la inserción de los datos en la base de datos.
- **Tabla distancedata:** almacena los cálculos obtenidos de distancia, demora, CPA y TCPA entre el buque propio y los targets. Como en el resto de tablas, también se almacena la fecha en la que se produce la inserción de los datos en la base de datos.
- **Tabla messagesexchanged:** almacena información sobre los mensajes binarios intercambiados entre buques (mensaje AIS tipo 6): el MMSI de origen que envía el mensaje, el MMSI de destino que lo recibe y el ID de función que se envía. Y, también la fecha en la que se produce la inserción de los datos en la base de datos.

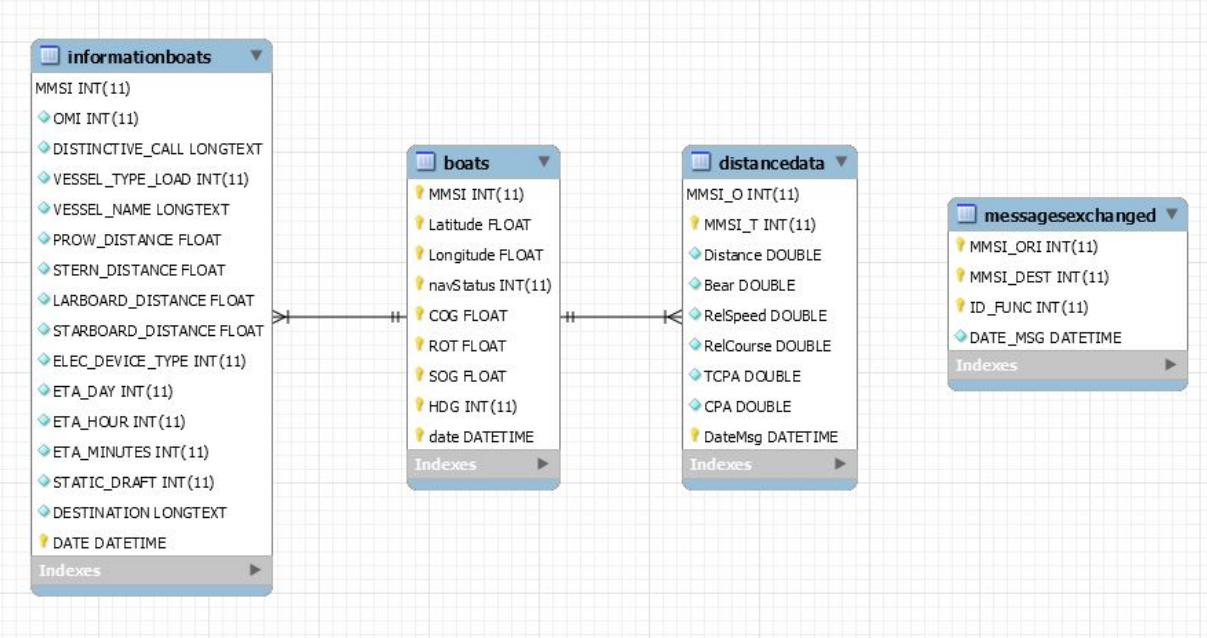


Fig. 4.4. Estructura base de datos *barcos.db*.

Para conseguir la creación de las tablas anteriores se puede ejecutar la siguiente query SQL:

```
CREATE TABLE `boats` (
  `MMSI` int(11) NOT NULL,
  `Latitude` float NOT NULL,
  `Longitude` float NOT NULL,
  `navStatus` int(11) NOT NULL,
  `COG` float NOT NULL,
  `ROT` float NOT NULL,
  `SOG` float NOT NULL,
  `HDG` int(11) NOT NULL,
  `date` datetime NOT NULL,
  PRIMARY KEY (`MMSI`,`date`,`Latitude`,`Longitude`,`navStatus`,`COG`,`ROT`,`SOG`,`HDG`)
);
```

Fig. 4.5. Creación ejemplo Tabla *boats*. Ejemplo 1.

Sin embargo, otra opción puede ser utilizar las opciones que presenta la aplicación *MySQL*

Workbench.

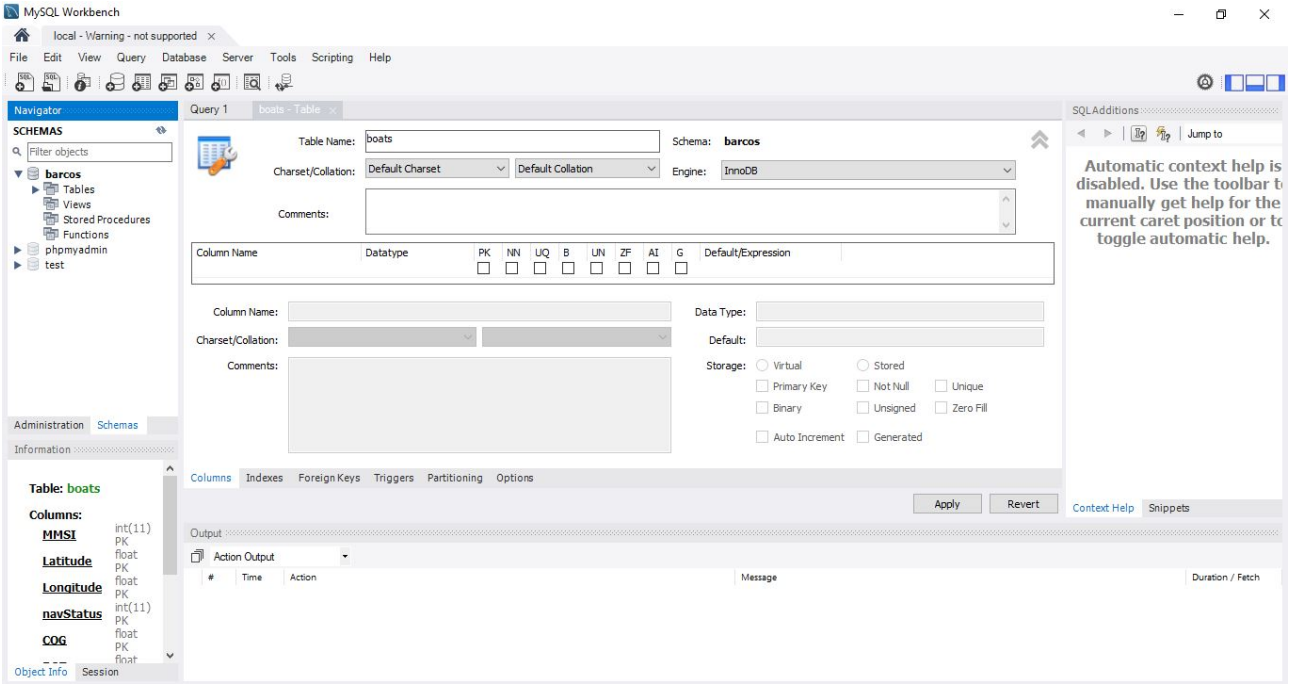


Fig. 4.6. Creación ejemplo Tabla *boats*. Ejemplo 2.

En función de la situación en la que nos encontremos, situación segura o situación de pre-alerta y/o alerta, la inserción en la base de datos se produce en tiempos diferentes. Estos tiempos los determina el usuario en el formulario incluido en la interfaz gráfica (*Storage Time (mseg)* y *Storage Time Alert (mseg)*).

Por otra parte, en el momento que se detecte el envío de un mensaje binario por parte de algún buque se almacenará dicha información en la Tabla *messagesexchanged*. Asimismo, la información estática asociada a cada buque sólo se almacenará una vez y se actualizará cada vez que se produzcan cambios.

```
@app.route('/read_msj_exchange', methods=['GET'])
def read_msj_exchange():
   try:
      //se obtienen los datos asociados a mensaje AIS tipo 6 mediante la conexión OPC-UA
establecida
      root = client.get_root_node()
      objects = root.get_child(['0:Objects'])
      msj=client.get_node("ns=5;s=Arp.Plc.Eclr/out_Mensaje6OPC")
      msjEx=[]
      existeMensaje=False
      //si se ha producido un intercambio de mensajes entre buques
      if (msj.get_value()[0]!=''):
         for i in range(len(mensajeExchange)):
            if mensajeExchange[i][1]==msj.get_value()[1] and
mensajeExchange[i][2]==msj.get_value()[2] and mensajeExchange[i][3]==msj.get_value()[3]:
               existeMensaje=True
         if not existeMensaje:
            mensajeExchange.append(msj.get_value())
```

Fig. 4.7. Ejemplo código inserción datos mensaje AIS tipo 6 en base de datos. Parte 1.

```
        // se conecta a la base de datos
         conn_msjex = mysql.connect()
         cursor_msjex =conn_msjex.cursor()
        // se obtiene la fecha y se ejecuta query inserción datos en bbdd
         date = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
         cursor_msjex.execute("INSERT INTO messagesexchanged
(MMSI_ORI,MMSI_DEST,ID_FUNC,DATE_MSG) VALUES (%d,%d,%d,str_to_date(\'%s\', '%%Y-%%m-%%d
%%H:%%i:%%s'))"%(int(msj.get_value()[1]),int(msj.get_value()[2]),int(msj.get_value()[3]),date))
        //se guardan los cambios y se cierra la conexión
         conn_msjex.commit()
         cursor_msjex.close()
         conn_msjex.close()
        //se crea la estructura json con los datos obtenidos para enviar a la interfaz gráfica
        for i in range(len(mensajeExchange)):
        msjEx.append(dict(ID=int(mensajeExchange[i][0]),MMSI_ORI=int(mensajeExchange[i][1]),MMSI_D
        EST=int(mensajeExchange[i][2]),ID_FUNC=int(mensajeExchange[i][3])))
        return jsonify(msjEx)
    except Exception as e:
      return e
```

Fig. 4.8. Ejemplo código inserción datos mensaje AIS tipo 6 en base de datos. Parte 2.

4.2. Ventajas uso de la aplicación *MySQL Workbench 8.0 CE*

MySQL Workbench 8.0 CE permite visualizar los datos almacenados en las diferentes tablas asociadas a las bases de datos creadas pero también contiene herramientas para la visualización y la mejora del rendimiento. Estas herramientas de monitorización, permiten a los administradores de dichas bases de datos ver fácilmente el rendimiento y así poder optimizar las consultas y el acceso a los datos. Algunas de estar herramientas son:

- **Panel de rendimiento:** permite mostrar el estado del servidor MySQL mediante diversas estadísticas de rendimiento:
  - Estadísticas del tráfico de red enviado y recibido por el servidor MySQL a través del cliente.

  - Gráficas que muestran la actividad del disco que genera el motor de almacenamiento InnoDB (el número de solicitudes de lectura y escritura por segundo, cantidad total de datos (en bytes) leídos, etc...)

- **Informes de esquema de rendimiento:** informan sobre las operaciones realizadas en el servidor MySQL. Estos informes sirven para descubrir declaraciones SQL de alto costo, enumerar las declaraciones que han generado errores o advertencias o todas las declaraciones que usan tablas temporales, etc...Estos informes de rendimiento se pueden exportar.

- **Plan de explicación visual:** los planes de explicación resaltan gráficamente cómo se ejecutan las sentencias SQL dentro de MySQL, es decir, muestra qué operaciones realiza MySQL cuando ejecuta sentencias SQL.

- **Estadísticas de consultas:** brindan estadísticas sobre las querys ejecutadas desde el Workbench Editor.

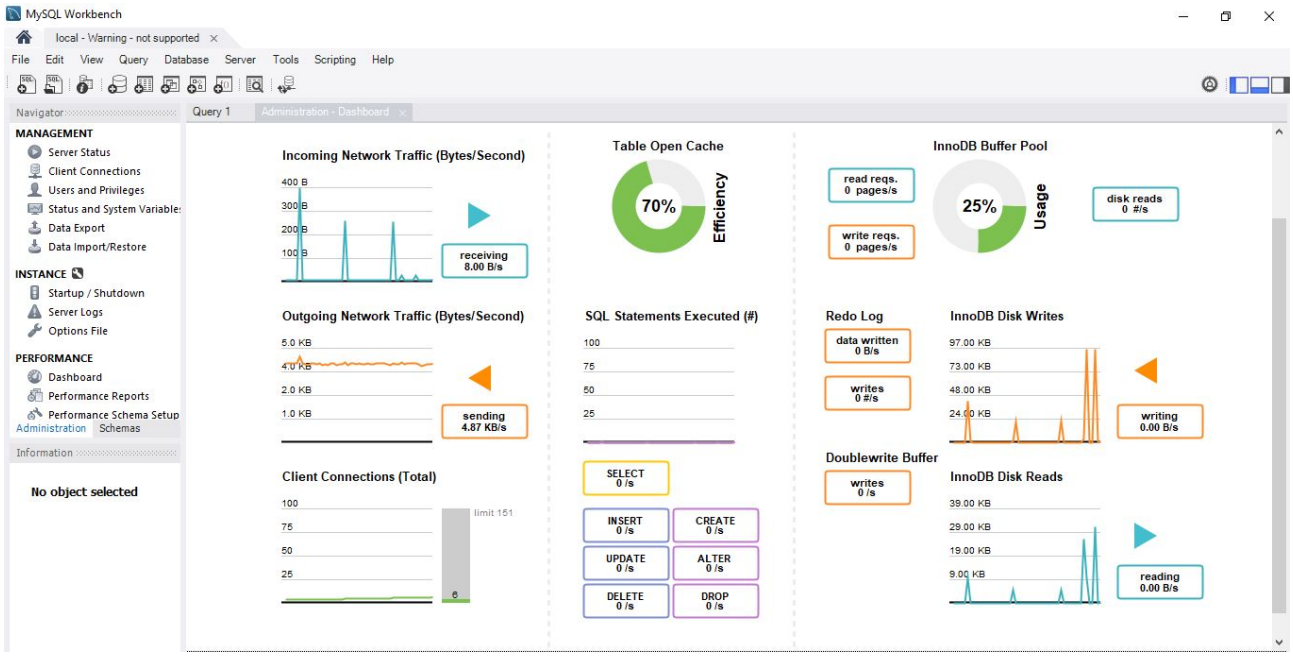


Fig. 4.9. Estadísticas *Panel de rendimiento*.

Además, permite exportar los datos y/o estructura de las diferentes bases de datos para utilizar por el usuario posteriormente en otras aplicaciones. Y, por último, ayuda con la migración de datos, permite migrar de MySQL a Microsoft SQL Server, Microsoft Access, Sybase ASE, SQLite, SQL Anywhere y PostreSQL.

4.3. Ejemplos

A continuación, se muestra ejemplos de almacenamiento de la información en la base de datos para posibles situaciones y, algunas ventajas que proporciona al usuario el uso de esta herramienta.

4.3.1. Situación segura

Como se muestra en la Figura 4.10 en una situación segura los datos deben almacenarse cada 50 segundos (*Storage Time (mseg)*).

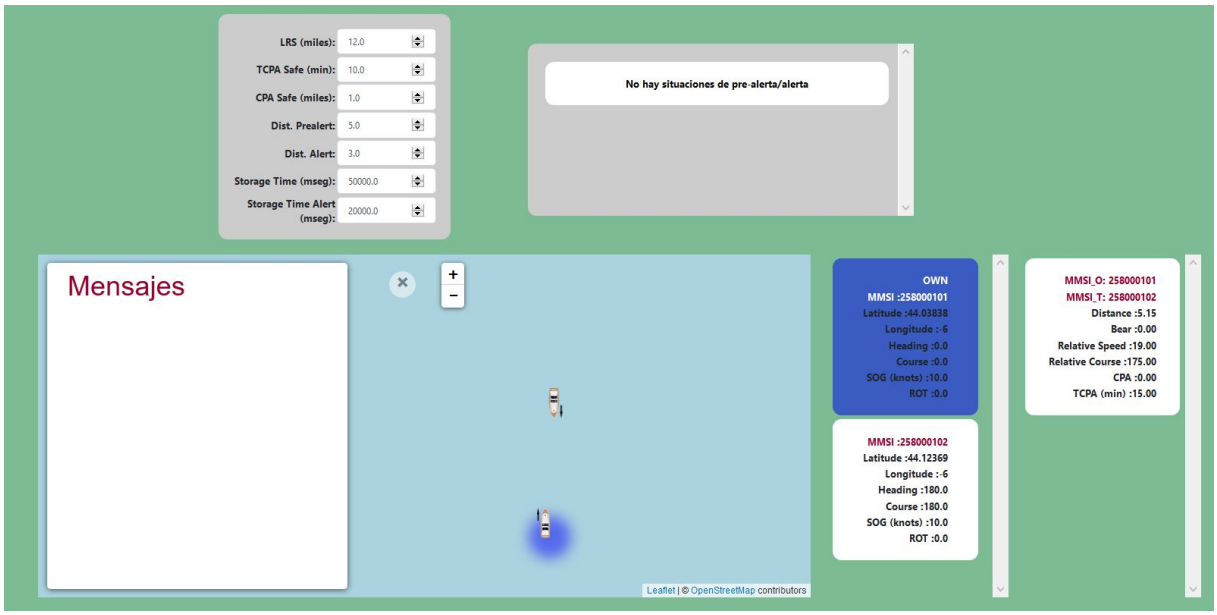


Fig. 4.10. Ejemplo situación segura.

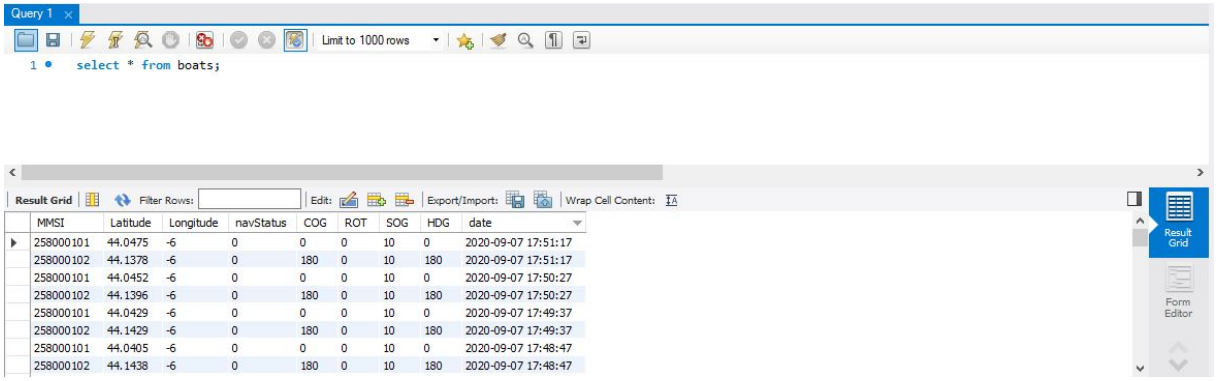
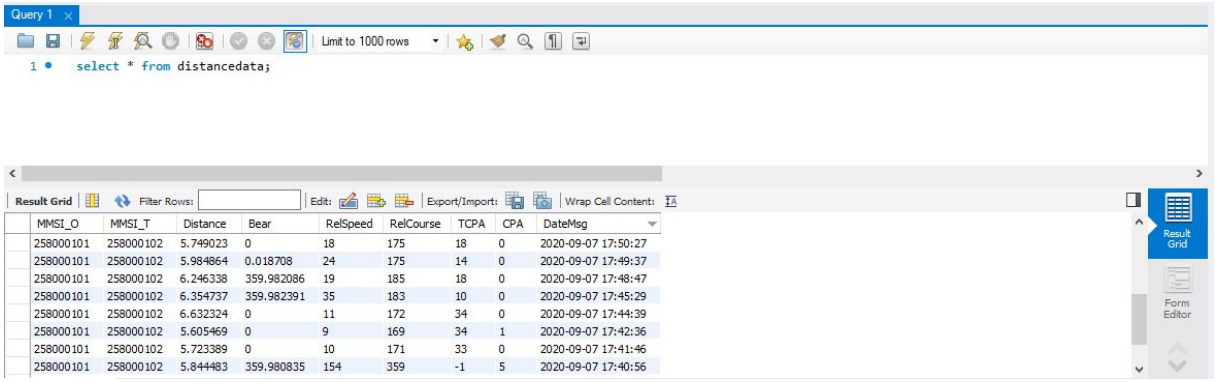


Fig. 4.11. Ejemplo almacenaje Tabla *boats* en situación segura.



MMSI_O	MMSI_T	Distance	Bear	RelSpeed	RelCourse	TCPA	CPA	DateMsg
258000101	258000102	5.749023	0	18	175	18	0	2020-09-07 17:50:27
258000101	258000102	5.984864	0.018708	24	175	14	0	2020-09-07 17:49:37
258000101	258000102	6.246338	359.982086	19	185	18	0	2020-09-07 17:48:47
258000101	258000102	6.354737	359.982391	35	183	10	0	2020-09-07 17:45:29
258000101	258000102	6.632324	0	11	172	34	0	2020-09-07 17:44:39
258000101	258000102	5.605469	0	9	169	34	1	2020-09-07 17:42:36
258000101	258000102	5.723389	0	10	171	33	0	2020-09-07 17:41:46
258000101	258000102	5.844483	359.980835	154	359	-1	5	2020-09-07 17:40:56

Fig. 4.12. Ejemplo almacenaje Tabla *distancedata* en situación segura.

4.3.2. Situación pre-alerta

Como se muestra en la Figura 4.13 en una situación segura los datos deben almacenarse cada 20 segundos (*Storage Time Alert(mseg)*).

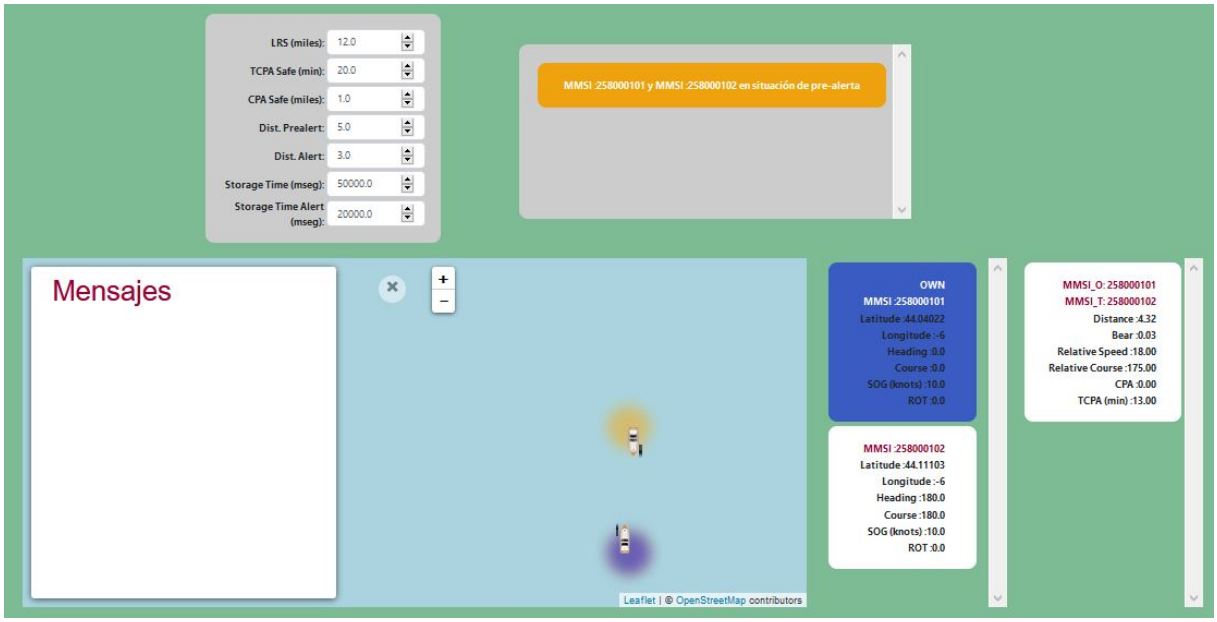
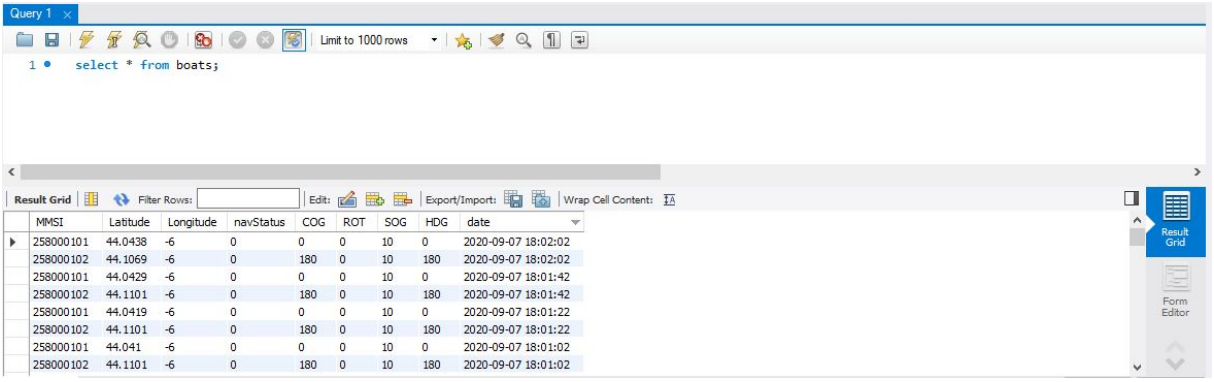
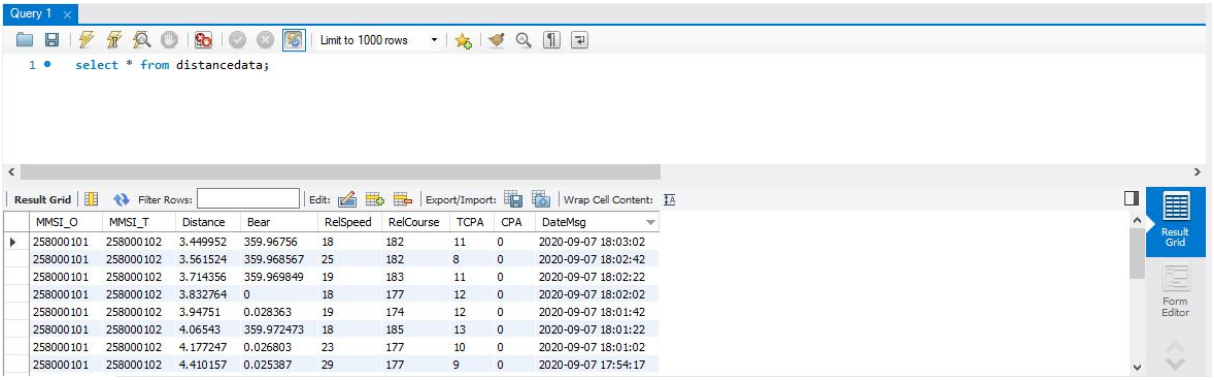


Fig. 4.13. Ejemplo situación pre-alerta.



MMSI	Latitude	Longitude	navStatus	COG	ROT	SOG	HDG	date
258000101	44.0438	-6	0	0	0	10	0	2020-09-07 18:02:02
258000102	44.1069	-6	0	180	0	10	180	2020-09-07 18:02:02
258000101	44.0429	-6	0	0	0	10	0	2020-09-07 18:01:42
258000102	44.1101	-6	0	180	0	10	180	2020-09-07 18:01:42
258000101	44.0419	-6	0	0	0	10	0	2020-09-07 18:01:22
258000102	44.1101	-6	0	180	0	10	180	2020-09-07 18:01:22
258000101	44.041	-6	0	0	0	10	0	2020-09-07 18:01:02
258000102	44.1101	-6	0	180	0	10	180	2020-09-07 18:01:02

Fig. 4.14. Ejemplo almacenaje Tabla *boats* en situación pre-alerta.



MMSI_O	MMSI_T	Distance	Bear	RelSpeed	RelCourse	TCPA	CPA	DateMsg
258000101	258000102	3.449952	359.96756	18	182	11	0	2020-09-07 18:03:02
258000101	258000102	3.561524	359.968567	25	182	8	0	2020-09-07 18:02:42
258000101	258000102	3.714356	359.969849	19	183	11	0	2020-09-07 18:02:22
258000101	258000102	3.832764	0	18	177	12	0	2020-09-07 18:02:02
258000101	258000102	3.94751	0.028363	19	174	12	0	2020-09-07 18:01:42
258000101	258000102	4.06543	359.972473	18	185	13	0	2020-09-07 18:01:22
258000101	258000102	4.177247	0.026803	23	177	10	0	2020-09-07 18:01:02
258000101	258000102	4.410157	0.025387	29	177	9	0	2020-09-07 17:54:17

Fig. 4.15. Ejemplo almacenaje Tabla *distancedata* en situación pre-alerta.

4.3.3. Situación intercambio mensajes entre buques

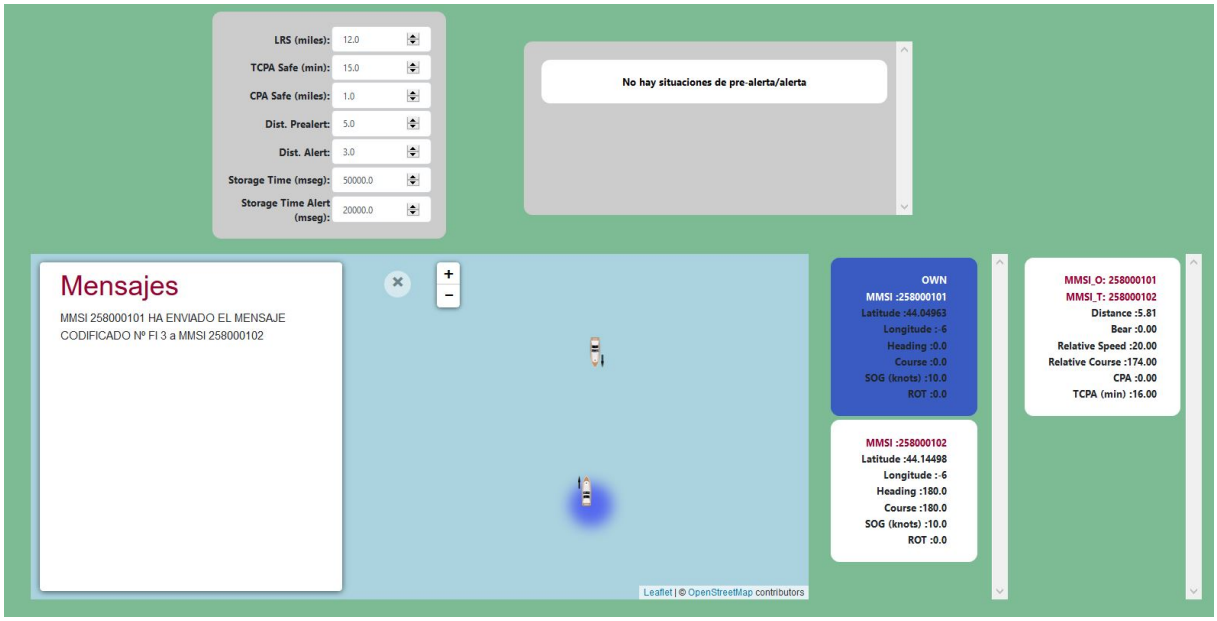


Fig. 4.16. Interfaz gráfica ante la detección de intercambio de mensajes entre buques.

En este caso, el buque propio envía un mensaje binario (mensaje AIS tipo 6) al target cuyo MMSI es 258000102.

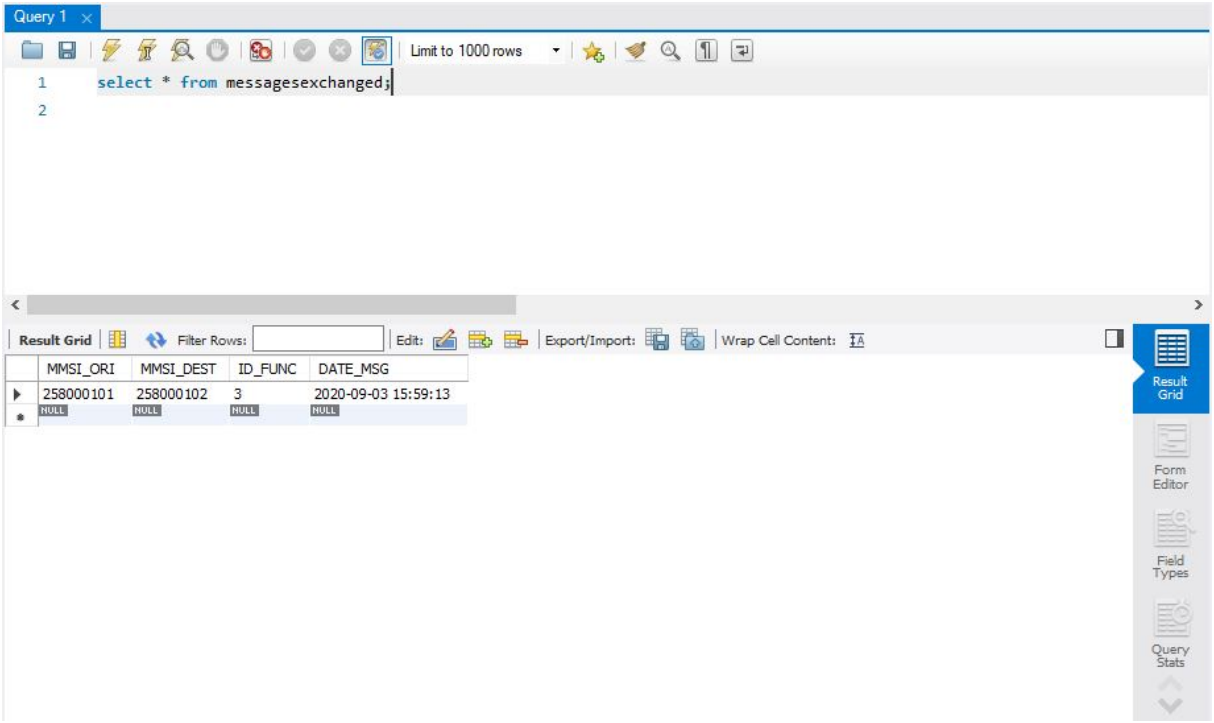


Fig. 4.17. Ejemplo almacenaje Tabla *mensajesexchanged*.

5. PLANIFICACIÓN Y PRESUPUESTO

5.1. Planificación

Se ofrece una planificación secuencial orientativa de las tareas que se han tomado para la realización del proyecto aunque algunas de ellas se pueden solapar y realizar al mismo tiempo.

El proyecto comienza en marzo y se va desarrollando a lo largo de los meses siguientes hasta la convocatoria de septiembre. Inicialmente, se llevaron a cabo varias reuniones con el tutor sobre la finalidad y alcance del trabajo en las cuáles se consiguió definir los objetivos del mismo. Una de las ideas fijas que se tenía desde el principio era utilizar el controlador lógico programable *PLCnext*, sin embargo, para el resto de ideas que se tenían sobre la mesa aún no se había estudiado su viabilidad. Por esta razón, en estas primeras reuniones se trataron y estudiaron temas como qué dispositivos AIS se podrían emplear, qué herramientas serían las más adecuadas para la creación de una interfaz gráfica más atractiva y, sobre todo, qué comunicaciones se deberían establecer para conseguir el funcionamiento total del sistema planteado. Fue en este momento cuando se produce la primera búsqueda de información a fin de enfocar el trabajo.

Tras estas reuniones, se consiguió enfocar el proyecto y se tomaron algunas decisiones como:

- Desarrollar el programa de decodificación de sentencias NMEA para el controlador *PLCnext* mediante la herramienta *PLCnext Engineer 2020.0*.
- Utilizar el software *AIS Simulator Version 8.04a - October 2019* en lugar de AIS reales ya que no ha sido posible disponer de estos dispositivos. Además, la funcionalidad de esta aplicación es similar al uso de una estación AIS real ya que permite simular las señales como si se tratasen de estaciones reales.
- Emplear una comunicación serie RS-232 entre el software *AIS Simulator Version 8.04a - October 2019* y *PLCnext* ya que es una de las formas de comunicación más adecuadas que presenta la herramienta de simulación. Y, además *PLCnext* dispone de un módulo adicional de comunicación serie, *AXL F RS UNI 1H*, que permite este tipo de contacto.
- Desarrollar una interfaz gráfica con Javascript y HTML para permitir que sea dinámica y más atractiva gracias a las opciones gráficas que presentan estas herramientas.
- Crear un servidor Python para permitir la comunicación y transferencia de datos entre la interfaz gráfica y *PLCnext* ya que dispone de herramientas que permiten la comunicación

OPC-UA con *PLCnext* y, el envío y recepción de peticiones HTTP con la interfaz web desarrollada.

- Utilizar comunicación OPC-UA entre *PLCnext* y el servidor Python desarrollado ya que, se trata de un estándar de comunicación más adecuado para las tendencias futuras de fabricación basadas en conceptos como Internet de las Cosas (IoT) y la Industria 4.0.
- Realizar peticiones HTTP entre la interfaz web desarrollada y el servidor Python ya que es una buena opción de comunicación entre estas dos tecnologías.

5.1.1. Programación PLCnext Engineer 2020.0

Antes de iniciar la programación del programa que permite la decodificación de las señales AIS simuladas y la realización de los cálculos de distancia, demora, velocidad relativa, rumbo relativo, CPA y TCPA, fue necesario comprender el funcionamiento de la herramienta para poder utilizarla de forma correcta. Por ello, se ha dedicado un tiempo a realizar programas de ejemplo para probar todas las funcionalidades que presenta.

Una vez desarrollado el programa, se llevaron a cabo pruebas de comunicación serie entre el controlador y la aplicación *AIS Simulator* para la recepción de las señales AIS con el fin de probar que el programa de decodificación funcionaba correctamente y que los cálculos obtenidos a partir de dichos datos fueron correctos.

5.1.2. Desarrollo servidor Python

Se ha dedicado un tiempo al estudio del establecimiento de la comunicación OPC-UA con PLCnext. Una vez conseguida dicha comunicación, se han realizado pruebas para obtener los datos y elaborar estructuras de datos que permitiesen la transferencia de la información obtenida a través de las peticiones HTTP que se iban a realizar desde la interfaz gráfica.

5.1.3. Creación peticiones HTTP

Una vez obtenida la información vía OPC-UA, se definieron las funciones asociadas a las peticiones HTTP en el servidor Python, es decir, se han definido los datos que se enviarán en cada petición. Además, se dedicó un tiempo a estudiar cómo crear este tipo de petición y cómo se puede manipular la información recibida desde Javascript.

5.1.4. Representación información vía Javascript y HTML

Por último, una vez conseguida la recepción de la información vía peticiones HTTP, se procede a manipular dicha información para su representación. Se estudiaron qué opciones de visualización presentan estas herramientas y, cómo era posible conseguir dicha visualización con los datos obtenidos. A partir de este estudio previo, se decidió utilizar las bibliotecas *Leaflet.js* y *Heatmap.js* de Javascript para el desarrollo de una interfaz web interactiva.

5.1.5. Pruebas finales

Se llevan a cabo varias pruebas finales sobre diferentes escenarios para comprobar el correcto funcionamiento del sistema completo.

5.1.6. Documentación

Finalmente, se realiza toda la documentación relativa al desarrollo del proyecto y los resultados obtenidos, así como toda la información recopilada que ha servido de ayuda para llevarlo a cabo.

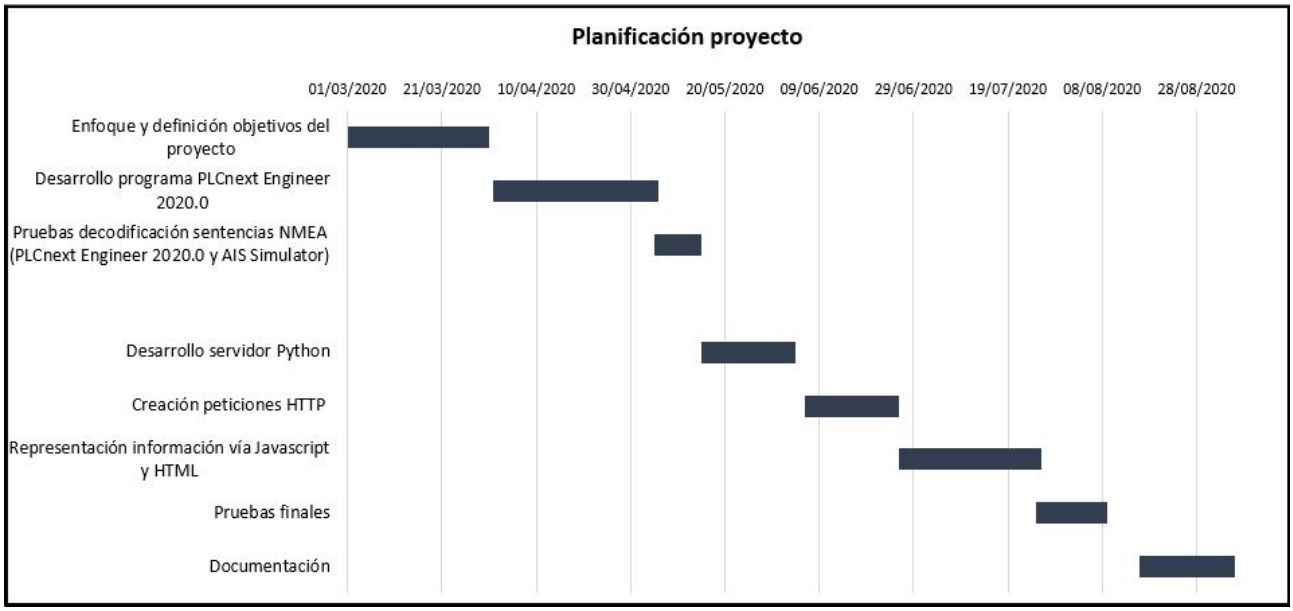


Fig. 5.1. Planificación desarrollo del proyecto.

5.2. Presupuesto

Y, a continuación, se desglosan todos los gastos que ha implicado la realización del proyecto:

5.2.1. Software

TABLA 5.1. PRESUPUESTO SOFTWARE.

Concepto	**Unidades**	**Precio ud (€)**	**Precio total (€)**
Software AIS Simulator Version 8.04a - October 2019	1	3000	3000
Software PLCnext Engineer 2020.0	1	-	-
Software MySQL Workbench 8.0 CE	1	-	-
Total			3000

5.2.2. Hardware

TABLA 5.2. PRESUPUESTO HARDWARE.

Concepto	**Unidades**	**Precio ud (€)**	**Precio total (€)**
PC Portátil	1	Amortización-12,5 600 (Precio ud(€) / 48 (vida útil (meses))	75 (12.5, 6 meses (tiempo uso))
Kit de Inicio PLC CPU Phoenix Contact PLCnext	1	Amortización-15,63 1500 (Precio ud(€) / 96(vida útil (meses))	93,75 (15.63, 6 meses (tiempo uso))
Módulo de comunicación - AXL F RS UNI 1H - 2688666	1	275	275
Acoplador del bus - AXL F BK PN - 2701815	1	300	300
Adaptador USB 2.0 a Serial - Digitus DA-70156	1	12	12
Cable Serie RS232 de comunicaciones	1	20	20
Total			775,75

Se supone que un portátil cuesta 600 €y su duración es de 48 meses; el valor de amortización por mes es de 12,50 €, y si se usa 6 meses su coste al proyecto es de 75 €.

Ocurre lo mismo para el controlador lógico programable, cuesta 1500 €y su duración es de 96 meses; el valor de amortización por mes es de 15,63 €, y si se usa 6 meses su coste al proyecto es de 93,75 €.

5.2.3. Mano de obra

Para la mano de obra se tienen en cuenta los siguientes precios unitarios:

TABLA 5.3. PRESUPUESTO MANO DE OBRA.

Concepto	**Total h**	**Precio h (€)**	**Precio total (€)**
Reuniones previas enfoque proyecto	3	20	60
Búsqueda y estudio de información	30	15	450
Familiarización con *PLCnext Engineer 2020.0*	15	15	225
Desarrollo programa con *PLCnext Engineer 2020.0*	100	20	2000
Desarrollo servidor Python	100	20	2000
Creación peticiones HTTP	50	20	1000
Creación interfaz usuario web	150	20	3000
Documentación para TFM	100	15	1500
Total			10235

- 15 €/h para trabajos de búsqueda de información, reuniones y documentación.
- 20 €/h para trabajo de ingeniería y programación.

5.2.4. Total

Total antes de impuestos

TABLA 5.4. PRESUPUESTO POR EJECUCIÓN DE CONTRATA.

Concepto	**Presupuesto (€)**
Total ejecución material	14010,75
Gastos generales (15 %)	2101,61
Beneficio industrial (6 %)	840,65
Total	16953,01

Presupuesto por ejecución de contrata

TABLA 5.5. PRESUPUESTO POR EJECUCIÓN DE CONTRATA.

Concepto	**Presupuesto (€)**
Total parcial	16953,01
Iva (21 %)	3560,13
Total	20513,14

6. CONCLUSIONES Y TRABAJOS FUTUROS

En el presente Trabajo Final de Máster se presenta una nueva versión de un prototipo de ayuda a la navegación marítima desarrollado con anterioridad que tiene una aplicación real práctica.

Durante la realización de este trabajo se han consolidado conceptos importantes estudiados durante este máster ya que ha permitido poner en práctica diferentes protocolos de comunicación industrial como ha sido OPC-UA, presentado en la asignatura *Comunicaciones Industriales* e *Integración de Sistemas*. También se han utilizado diferentes tecnologías como ha sido la visualización de los datos vía Javascript, estudiada en la asignatura *Análisis y Visualización de Datos* y, el uso del estándar IEC-61131 para la programación del controlador lógico programable, estudiado en la asignatura *Análisis e Implementación de Sistemas de Automatización*.

Una de las características que presenta es que ha permitido comprobar de forma práctica y real las comunicaciones que se establecen entre los OONWs actuales para garantizar una navegación marítima segura. Además, en cuánto a los resultados obtenidos, se ha podido comprobar que esta nueva versión desarrollada se trata de una buena herramienta de ayuda porque consigue mostrar rápidamente de forma visual la detección temprana de situaciones de colisión entre buques permitiendo así a los operarios de los mismos tomar decisiones tempranas ante estas situaciones.

Para terminar, se presentan posibles líneas de investigación que pueden ser objeto de interés en el futuro:

- Implementación del sistema con estaciones AIS reales, sin emplear un simulador de señales.
- Desarrollo de la interfaz gráfica mediante el uso de herramientas específicas para la creación de interfaces de usuario.
- Desarrollar en la interfaz gráfica la opción de permitir al operario enviar mensajes binarios (mensajes AIS tipo 6) a otros buques.
- Desarrollar en la interfaz gráfica la opción de permitir al operario enviar de órdenes de maniobra (cambio de velocidad, rumbo, heading, etc..) ante la detección de colisiones para poder evitarlas.
- Estudiar el uso de diferentes protocolos de comunicación entre las diferentes partes del

sistema implementado que permitan mejorar las comunicaciones actualmente establecidas.

- Mejorar la estructura y funcionamiento de la base de datos planteada en este proyecto.
- Estudiar el uso de otros programas PC para la visualización de los datos almacenados en la base de datos.

BIBLIOGRAFÍA

[1] Argüelles, R., Maza, J., and Martín, F. (2019). *Specification and Design of Safety Functions for the Prevention of Ship-to-Ship Collisions on the High Seas. Journal of Navigation*, 72 (1), 53-68.

[2] Argüelles, R., Maza, J., and Martín, F. (2019). *Testing a Safety Related System Model for the Prevention of Ship-to-Ship Collisions at High Seas. Enviado a Journal of Navigation.*

[3] NMEA 0183. (2008). *NMEA 0183 Standard For Interfacing Marine Electronic Devices.* https://www.nmea.org/content/nmea_standards/nmea_0183_v_410.asp.

[4] NMEA 2000. (2016). *NMEA 2000 Standard for Serial-Data Networking of Marine Electronic Devices.* https://www.nmea.org/content/nmea_standards/nmea_2000_ed3_10.asp.

[5] AIS_Guide. The Non-Idiot's Guide to AIS. http://www.mynewsdesk.com/uk/digitalyacht/news/ais-a-non-idiot-s-guide-91913.

[6] Phoenix Contact. *Control - AXC F 2152 - 2404267.* https://www.phoenixcontact.com/online/portal/es?uri=pxc-oc-itemdetail:pid=2404267&library=eses&tab=1.

[7] Phoenix Contact. *Módulo E/S - AXL F DI8/1 DO8/1 1H - 2701916.* https://www.phoenixcontact.com/online/portal/es?uri=pxc-oc-itemdetail:pid=2701916&library=eses&tab=1.

[8] Phoenix Contact. *Módulo E/S - AXL F AI2 AO2 1H - 2702072.* https://www.phoenixcontact.com/online/portal/es?uri=pxc-oc-itemdetail:pid=2702072&library=eses&tab=1.

[9] Phoenix Contact. *Módulo de comunicación - AXL F RS UNI 1H - 2688666.* https://www.phoenixcontact.com/online/portal/es?uri=pxc-oc-itemdetail:pid=2688666&library=eses&tab=1.

[10] Phoenix Contact. *Acoplador del bus - AXL F BK PN - 2701815.* https://www.phoenixcontact.com/online/portal/es?uri=pxc-oc-itemdetail:pid=2701815&library=eses&tab=1

[11] AIS_Guide. *The Non-Idiot's Guide to AIS*. http://www.mynewsdesk.com/uk/digitalyacht/news/ais-a-non-idiot-s-guide-91913.

[12] Convenio internacional para la seguridad de la vida humana en el mar, 1974 (Convenio SOLAS). http://www.imo.org/es/About/Conventions/ListOfConventions/Paginas/International-Convention-for-the-Safety-of-Life-at-Sea-(SOLAS),-1974.aspx.

[13] Delgado Moya S. (2016). *Utilización de datos AiS para el seguimiento de la actividad pesquera en Alicante. Trabajo Final de Grado. Universidad de Alicante.*

[14] IALA. (2016). *IALA Guideline No. 1082 On an Overview of AIS.* https://www.navcen.uscg.gov/pdf/IALA_Guideline_1082_An_Overview_of_AIS.pdf.

[15] ITU. (2014). *Recommendation ITU-R M.1371-5 Technical characteristics for an automatic identification system using time division multiple access in the VHF maritime mobile frequency band*. https://www.itu.int/rec/R-REC-M.1371-5-201402-I/es.

[16] A200 AIS Class A / Inland transceiver. Installation and operation manual. https://www.alphatronmarine.com

[17] AIS Simulator. http://www.kagstrom.no/ais_simulator.htm.

[18] Phoenix Contact. *Software - PLCNEXT ENGINEER - 1046008*. https://www.phoenixcontact.com/online/portal/es/?uri=pxc-oc-itemdetail:pid=1046008&library=eses&pcck=P-19-05-01&tab=5&selectedCategory=ALL

[19] Digitus. *DIGITUS USB 2.0 serial adapter*. https://www.digitus.info/en/products/computer-and-office-accessories/computer-accessories/usb-components-and-accessories/interface-adapter/da-70156/

[20] Phoenix Contact. *AXL_ComSerial Library*. https://www.plcnextstore.com/#/171

[21] ISAwiki. *Acceso a M241 mediante OPC UA*. http://isa.uniovi.es/wiki/isa/index.php/Acceso_a_M241_mediante_OPC_UA.

[22] Python OPC-UA Documentation. https://python-opcua.readthedocs.io/en/latest/.

[23] Flask-MySQL. https://flask-mysql.readthedocs.io/en/latest/.

[24] Quickstart — Flask Documentation (1.1.x). https://flask.palletsprojects.com/en/1.1.x/quickstart/.

[25] jQuery AJAX Methods. https://www.w3schools.com/jquery/jquery_ref_ajax.asp.

[26] Leaflet. https://github.com/Leaflet/Leaflet.

[27] Leaflet.heat. https://github.com/Leaflet/Leaflet.heat.

[28] MySQL Workbench 8.0 CE.