Escola d'Enginyeria de Telecomunicació i Aeroespacial de Castelldefels
UNIVERSITAT POLITÈCNICA DE CATALUNYA

TRABAJO FINAL DE CARRERA

TÍTULO: UAS-ATM Simulated Integrated Scenario Technical Debriefer

TITULACIÓN: Ingeniería Técnica Aeronáutica, especialidad Aeronavegación

AUTORES: Rubén Centelles Carmona
Irene Peña Parras

DIRECTOR: Pablo Royo Chic
CODIRECTOR: Raúl Cuadrado Santolaria

FECHA: 27de junio de 2013

Título: UAS-ATM Simulated Integrated Scenario Technical Debriefer

Autores: Rubén Centelles Carmona
Irene Peña Parras

Director: Pablo Royo Chic
Codirector: Raúl Cuadrado Santolaria

Fecha: 27de junio de 2013

Resumen

Un sistema aéreo no tripulado es una aeronave que vuela sin personal a bordo. Tanto estos vehículos como los aviones tripulados están provistos de una caja negra que registra la totalidad de las variables de vuelo (altitud, velocidad, aceleración, etc…) y las conversaciones de cabina. Una de sus funciones es almacenar datos que, en caso de un accidente, permita analizar lo ocurrido en los momentos previos a este. Estos datos son almacenados con el objetivo de ser tratados a posteriori.

El grupo de investigación ICARUS está trabajando en un simulador de aviones no tripulados llamado ISIS, con el que se pretende evaluar la integración de éstos en el espacio aéreo no segregado. Este simulador, al igual que una aeronave, contiene una caja negra que genera y almacena los datos de vuelo para su posterior análisis. Las particularidades del simulador de aviones no tripulados hacen que, para poder analizar los datos en profundidad, sea necesario un software específico.

Este proyecto tiene como objetivo el diseño e implementación de un software para el análisis de ficheros de datos generados por este simulador. El sistema debe ser capaz de representar mediante gráficos cualquier magnitud que éste genere en función del tiempo. Además debe tener herramientas para detectar conflictos de tráfico aéreo con otros aviones comerciales y analizar parámetros de las distintas fases del plan de vuelo. La aplicación ha de permitir exportar sus graficas a formato vectorial y realizar conversiones de unidades. Además, el software, ha sido extendido para analizar cualquier tipo de fichero de datos. De este modo a parte dar las herramientas necesarias para analizar los datos propios del simulador, ofrece flexibilidad para poder analizar ficheros de otras plataformas.

Para comprobar el funcionamiento de dicha extensión se han realizado pruebas con datos de tres plataformas diferentes: datos de vuelo de un helicóptero real no tripulado, del simulador de aviones no tripulados y de vuelos realizados por un avión no tripulado de NASA.

Title: UAS-ATM Simulated Integrated Scenario Technical Debriefer

Authors: Rubén Centelles Carmona
Irene Peña Parras

Director: Pablo Royo Chic
Codirector: Raúl Cuadrado Santolaria

Date: 27 de junio de 2013

Overview

An unmanned aerial system is an aircraft that flies with no crew onboard. These vehicles and manned aircraft are provided with a black box that records all flight variables (altitude, speed, acceleration, etc...) and cockpit conversations. One of its functions is to store data in the event of an accident, to analyze what happened instants prior the incident. This data is stored in order to be treated afterwards.

ICARUS research group is working on an unmanned aerial system simulator called ISIS, used to evaluate the integration of the unmanned aerial systems into non-segregated airspace. This simulator, like an aircraft, has a black box that generates and stores flight data for later analysis. The particularities of the unmanned aerial system simulator require specific software to analyze the data in depth.

This project aims to design and implementation of software to analyze data files generated by this simulator. The system must be able to represent charts using any magnitude in function of time. It must also have tools to detect air traffic conflicts with other commercial aircraft and analyze parameters of the different flight plan sections. The application allows exporting these charts to vector format and performing unit conversions. In addition, the software has been extended to analyze any type of data file from any platform. Thus, the software is not specific to the ISIS simulator, but is able to process any type of data file.

To check the operation of this extension, it has been tested with data from three different platforms: flight data from a real unmanned helicopter, from an unmanned aircraft simulator and from unmanned flights performed by NASA.

ÍNDICE

ÍNDICE DE FIGURAS

GLOSARIO

UAS	Unmanned Aircraft Systems
UAV	Unmanned Aerial Vehicle
ICARUS	Intelligent Communications and Avionics for Robust Unmanned aerial Systems
PF	Windows Presentation Foundation
XML	Extensible Markup Language
D3	Dynamic Data Display
ISIS	ICARUS Simulation Integrated Scenario
UTM	Tiempo Universal Coordinado
KML	Keyhole Markup Language
SVG	Scalable Vector Graphics
EETAC	Escola d'Enginyería de Telecomunicació i Aeroespacial de Castelledefels
ATM	Air Traffic Management
UTC	Universal Time Coordinated

CAPÍTULO 1. INTRODUCCIÓN

ICARUS es un grupo de investigación formado por estudiantes, investigadores y profesores de la EETAC. Entre sus proyectos se encuentra una plataforma de simulación UAS-ATM (Unmanned Aircraft Systems – Air Traffic Management), llamada ISIS, que permite evaluarla integración de los UAS en el espacio aéreo no segregado. Un UAS es una aeronave que vuela sin personal a bordo. Durante cada simulación, este software genera un fichero de texto con todos los datos de telemetría, plan de vuelo y tráfico aéreo.

Hasta la fecha no se disponía de ninguna aplicación específica para poder realizar el análisis o la visualización de las distintas magnitudes que forman estos datos de telemetría. Por este motivo, este trabajo final de carrera tiene como objetivo el diseño e implementación de una herramienta versátil que sea capaz de realizar dicho análisis.

El proyecto tiene dos fases diferenciadas. La primera se divide en la desarrollo de tres herramientas de análisis:

- *Análisis del fichero de telemetría*, mostrando la evolución de las diferentes magnitudes (velocidad, posición, aceleración, etc…) respecto al tiempo en un gráfico de líneas.
- *Análisis del plan de vuelo por tramos*, visualizando la duración, distancia, velocidad vertical y altitud llevadas a cabo en cada uno de los tramos.
- *Análisis del tráfico aéreo*, estudiando las posiciones, velocidades y ángulos de las aeronaves que forman parte de él. En esta fase serán necesarios los ficheros de telemetría correspondientes a cada una de las aeronaves, además de la telemetría del UAS.

La segunda fase es una evolución de la primera. En ella se desarrolla un lenguaje para diseñar un fichero de configuración que defina cómo va a ser el fichero de telemetría a procesar. De este modo, el software no es específico para el simulador ISIS, sino que es capaz de procesar cualquier tipo de fichero de datos. En concreto, se han realizado pruebas con tres plataformas diferentes:

- Datos de vuelo de un helicóptero real no tripulado [17].
- Datos del simulador de aviones no tripulados ISIS [18].
- Datos de vuelos realizados por un avión no tripulado de NASA [21].

1.1 **Objetivos y motivaciones**

El objetivo de este Trabajo Final de Carrera es claramente práctico, ya que busca crear una aplicación necesaria para el grupo de investigación ICARUS.
Los objetivos de este proyecto se pueden resumir en dos grandes bloques:

- *La aplicación ha de ser capaz de realizar el análisis de los parámetros contenidos en un fichero de datos*.
  Para ello se confeccionará un lenguaje para definir el formato de este fichero, obteniendo de esta manera una aplicación que sirva para distintas plataformas (vuelos reales y simuladores).

- *La aplicación debe ser a medida del simulador de UAS, analizando los datos relacionados con el plan de vuelo y tráfico aéreo.*
  El estudio de los datos relacionados con el plan de vuelo permitirá testear las nuevas maniobras y rutas que se desarrollen en el simulador. De este modo, se podrán mostrar resultados en cuanto al tiempo y distancia empleados en cada una de ellas y evaluar dichas maniobras para una futura incorporación en aviones no tripulados reales. En cuanto al tráfico aéreo, esta herramienta permitirá probar la detección de conflictos con intrusos en el simulador y la maniobra de escape o evasión que éste genera.

La principal motivación para realizar este proyecto ha sido la unión de la programación con una vertiente aeronáutica. Creemos que es mucho más interesante dedicar esta última etapa de nuestra carrera a realizar una tarea que dé como fruto una interesante herramienta que vaya a ser usada por nuestros compañeros.

1.2 **Estudio de las tecnologías utilizadas**

Durante el desarrollo de la carrera, nuestros conocimientos en torno a la programación se limitaban a C++ y Microsoft Visual Basic. Aun así el lenguaje utilizado en este proyecto es C# puesto que es el lenguaje más usado en las aplicaciones desarrolladas por el grupo de investigación ICARUS. De este modo, será posible reutilizar en el futuro las estructuras de datos creadas durante el proyecto, así como aprovechar en este proyecto las estructuras existentes. No obstante, C# es suficientemente similar a C++ como para no tener que detenerse demasiado en su aprendizaje.

En cuanto a la parte gráfica, la tecnología utilizada es Windows Presentation Foundation (WPF). Esta tecnología tiene un gran atractivo y posee gráficos vectoriales, mejorando significativamente su visualización independientemente de la resolución de las pantallas de las maquinas donde se ejecuta. Además de ser el lenguaje de implementación de Windows Vista, 7 y 8.

Otro aspecto a abordar era la librería que se iba a usar para realizar gráficas. Se buscó una librería gratuita, que tuviera foros de consultas y fuera usada por programadores para poder sacarle el máximo partido sin detenerse demasiado en problemas aislados. Siguiendo estos requisitos se encontraron varias

librerías, siendo las más importantes Dynamic Data Display (D3) [6] y WPFToolkit [19]. La primera, es una librería muy sencilla y fácil de utilizar por lo cual es la usada en la mayor parte del proyecto. La segunda, en cambio, es utilizada en gráficos más específicos ya que ofrece una mayor variedad de recursos.

1.3 **Planificación temporal**

Debido a la falta de conocimientos de los lenguajes y librerías mencionados anteriormente, los primeros pasos a seguir en el transcurso del proyecto fueron encaminados en el aprendizaje de éstos. Por este motivo, lo primero que se hizo fue aprender C# con la ayuda de materiales docentes [5], tarea que duró dos semanas. Posteriormente, se dedicó un mes para abordar WPF [1] [2] [3] [4] y D3 [6] [7]. No obstante, el aprendizaje de la librería D3 fue más complicado debido a la poca documentación existente.

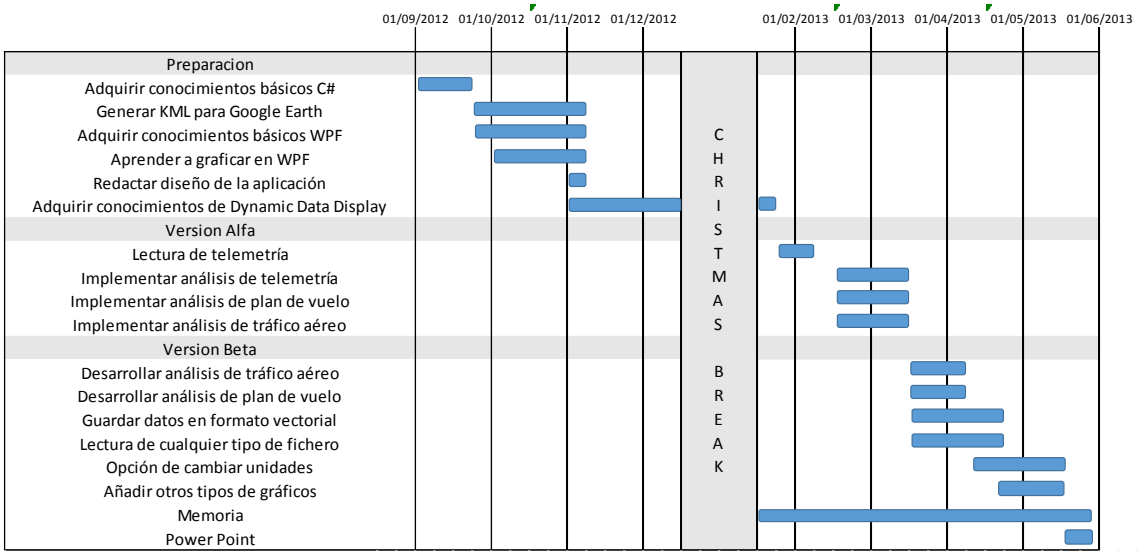


Figura 1.1 Planificación

En la Figura 1.1, está representada la planificación de las tareas. En ella se puede ver la tarea a realizar y el intervalo de tiempo que se le dedicó.

Una vez adquiridos los conocimientos necesarios, se realizó la tarea de generar archivos que mostrasen el recorrido de la aeronave en Google Earth. Para esta tarea se dedicó un mes.

A continuación, se redactó un diseño de la versión Alfa de la aplicación. En el diseño se requería que la aplicación fuera capaz de cargar un fichero de telemetría y representar las diferentes magnitudes que formaban parte de él (aceleración, velocidad, posición, etc…) en gráficos lineales.

Tras el parón de navidad, el siguiente objetivo fue la creación de la versión Alfa. Para esta versión se tuvieron que desarrollar las herramientas necesarias para

realizar el análisis de los datos de telemetría, de plan de vuelo y de tráfico aéreo. Tras seis semanas de desarrollo, la versión Alfa cumplía los requisitos redactados en el diseño de la aplicación y, por lo tanto, se dio por terminada.

La versión Alfa se considera esencial para demostrar el dominio de las tecnologías y una buena práctica para evaluar las librerías y testearlas.

Una vez terminada esta primera versión, el siguiente paso fue realizar la versión Beta. En ésta, el objetivo era desarrollar un lenguaje en base al cual, la aplicación pueda procesar cualquier tipo de fichero. De esta manera, la aplicación ganó utilidad debido al gran abanico de posibilidades de lectura de datos que ofrece. Además, esta versión permite la opción de exportar las gráficas a formato vectorial o SVG (Scalable Vector Graphics), cambiar de unidades y representar las magnitudes con otros tipos de gráficas. Para todas estas tareas se dedicaron dos meses.

Finalmente, se dedicó tiempo a confeccionar la memoria y el Power Point necesario para la presentación.

1.4 **Requisitos de calidad de la aplicación**

La aplicación ha de cumplir los siguientes requisitos que se van a detallar para que se dé por aceptada. Estos requisitos están distribuidos en cuatro grupos diferenciados:

- *La aplicación ha de ser gráficamente agradable y amigable*.
  Tiene que seguir un patrón de colores atractivos para el usuario y ha de disponer de suficientes mensajes que orienten al usuario sobre qué se requiere hacer previamente. Es decir, el usuario no debería tener ninguna duda sobre cómo interactuar con la aplicación, qué datos debe entrar y de qué modo, y cómo interpretar los resultados y mensajes de la aplicación.

- *La aplicación ha de ser robusta.*
  De este modo la aplicación puede resistir, sin bloquearse, todos los errores típicos que pueden aparecer. Se han de subsanar todos los errores que puedan bloquear la aplicación, por la posible pérdida de datos.

- *La aplicación ha de ser correcta y eficiente.*
  Esto significa que realiza todas las funcionalidades previstas y funciona bien en todas las pruebas realizadas utilizando el mínimo código posible. Esto se consigue reutilizando el máximo número de métodos y funciones que sean posibles.

- *El código está compuesto por bloques, es decir, bien organizado*.
  Ha de ser fácil encontrar el punto de la aplicación que hay que tocar para realizar alguna modificación. Cada procedimiento y función ha de estar correctamente documentado mediante un comentario inicial que explica lo que hace, y cuáles son los parámetros de entrada y salida. Además, los

puntos del código especialmente complicados tienen un comentario suficientemente clarificador.

CAPÍTULO 2. Arquitectura y diseño de la aplicación

2.1 Requisitos

Los requisitos de la aplicación están distribuidos en tres grupos según el tipo de análisis a realizar. De esta manera se podrán crear herramientas independientes que, cada una de ellas, puedan cumplir los requisitos de un grupo en concreto.

En la Figura 2.1 se muestra un esquema de los tres grupos de análisis del proyecto. El primer grupo de análisis es el de telemetría, para mostrar gráficamente los datos cargados. En segundo lugar el de tráfico aéreo, para realizar un análisis del conflicto entre el UAS y las aeronaves que interfieren en su plan de vuelo. Y por último el grupo encargado del análisis de todos los parámetros relacionados con el plan de vuelo del UAS.

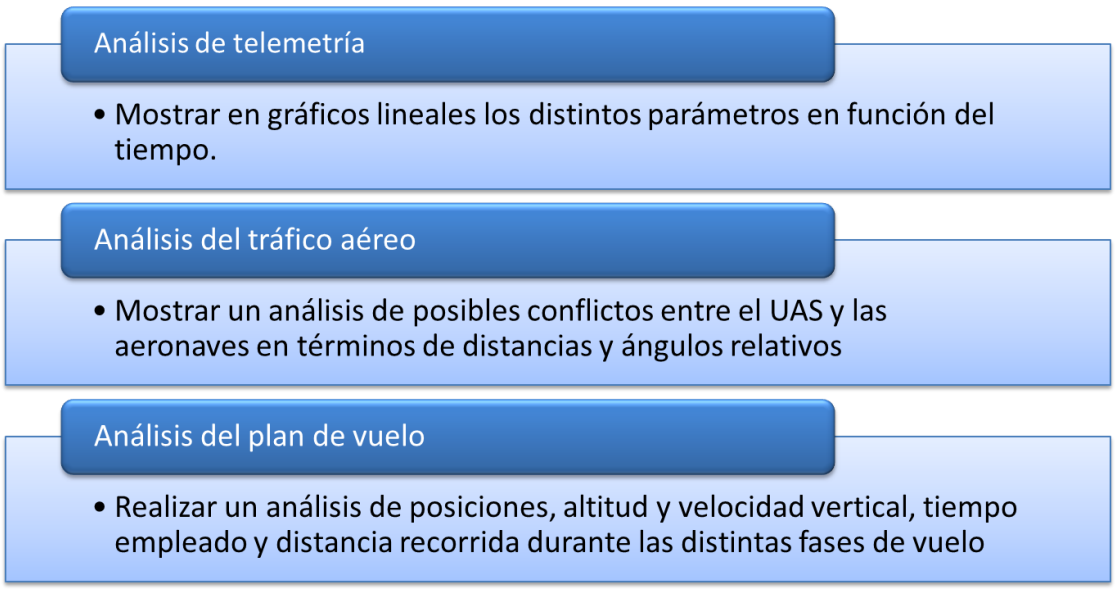


Figura 2.1 Herramientas generales de la aplicación

Estos grupos son explicados en detalle en los apartados siguientes.

2.1.1 Análisis de telemetría

El principal cometido de este grupo es analizar los datos de telemetría obtenidos de un fichero de texto. Este análisis debe ser aplicable para datos recibidos de cualquier plataforma. Por este motivo, esta herramienta debe estar adaptada para recibir cualquier tipo de dato de entrada. Sus requisitos principales son los siguientes:

- Mostar mediante una gráfica de líneas los datos de telemetría y su evolución respecto al tiempo.
- Exportar las gráficas visualizadas en formato vectorial con el objetivo de facilitar su posterior edición.
- Permitir que el usuario cambie las unidades de las distintas magnitudes que se muestran en las gráficas.
- Permitir seleccionar las diferentes líneas de los gráficos en colores distintos y en aquellos colores seleccionados por el usuario, con el objetivo de aclarar la visualización de varios datos al mismo tiempo.

2.1.2 Análisis de plan de vuelo

Un plan de vuelo puede estar separado por tramos, organizados según la fase de vuelo en la que se encuentra cada uno de ellos (despegue, iteraciones, aproximación, aterrizaje, etc…). En el simulador ISIS se registran los datos del plan de vuelo indicando en el fichero de texto la posición y el instante en el que el UAS entra en un nuevo tramo. Este registro de datos es muy particular y, por lo tanto, este tipo de análisis será específico para el simulador.

En este grupo de análisis se desea estudiar cada fase del vuelo de manera separada. Con este objetivo los requisitos son los siguientes:

- Mostrar gráficamente el recorrido del UAS en un tramo concreto del vuelo.
- Mostrar el tiempo, distancia y velocidad media empleada en cada tramo recorrido durante el vuelo, así como la tasa de ascenso y descenso y los cambios de altura de cada uno de estos.
- Mostrar la proporción de tiempo y distancia empleados en ese tramo concreto respecto al vuelo completo.
- Exportar las gráficas visualizadas en formato vectorial con el objetivo de facilitar su posterior edición.
- Permitir que el usuario cambie las unidades de las distintas magnitudes que se muestran en las gráficas.

2.1.3 Análisis de tráfico aéreo

Una de las opciones que permite el simulador ISIS, es la creación de un escenario virtual con tráfico aéreo. Como el simulador funciona en base al UAS, cualquier aeronave que interfiera en su plan de vuelo o trayectoria será considerada un intruso. De esta manera, se puede investigar y probar en el simulador las maniobras de evasión y escape en caso de que un intruso superé la distancia mínima con el UAS, en cuyo caso existirá un conflicto.

En esta herramienta se desea analizar y evaluar éstas maniobras de resolución de conflictos entre el UAS y el tráfico simulado. Este tipo de análisis es específico para el simulador ISIS. Concretamente tiene los siguientes requisitos:

- Cargar los ficheros de telemetría de los intrusos para poder compararlos con el del UAS, ya sea de uno en uno o seleccionando todos los intrusos desde una carpeta.
- Mostrar el recorrido de los distintos intrusos en Google Earth y así ver su trayectoria y comparar los resultados obtenidos en las gráficas. Deberán mostrarse los recorridos de cada uno de ellos con colores diferentes para que el usuario pueda diferenciarlos.
- Mostrar las distancias horizontales y verticales, así como las velocidades y ángulos relativos en su evolución con el tiempo de los distintos intrusos respecto al UAS.
- Poder resaltar los momentos de conflicto en función de una distancia de separación vertical y horizontal seleccionada por el usuario.
- Exportar las gráficas visualizadas en formato vectorial con el objetivo de facilitar su posterior edición.
- Permitir que el usuario cambie las unidades de las distintas magnitudes que se muestran en las gráficas.

2.2 **Arquitectura del proyecto**

En la Figura 2.2 se puede ver la relación de entradas-salidas de la aplicación. Ésta aplicación, llamada Debriefer, recibe como entrada cuatro tipos diferentes de ficheros y después de analizarlos, ofrece distintos tipos de salidas para visualizarlos.

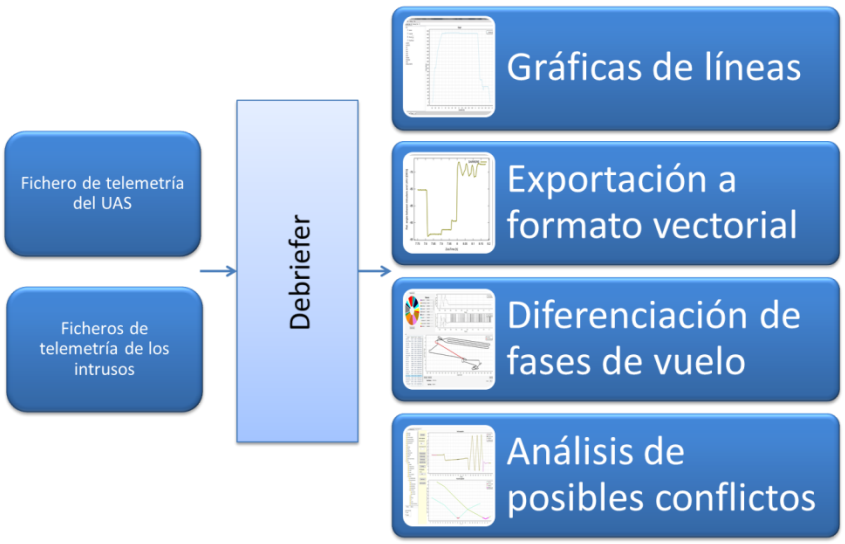


Figura 2.2 Arquitectura del proyecto

Como se aprecia la Figura 2.2, la aplicación representada como “Debriefer”, recibe y almacena los siguientes archivos de entrada para su posterior análisis:

- El fichero de telemetría del UAS.
- Los ficheros de telemetría de los intrusos.

Una vez que el programa recibe estos ficheros, éste los procesa y los almacena en estructuras de datos internas para después ofrecer múltiples salidas, de nuevo representadas en la Figura 2.2:

- Gráficos lineales de las diferentes magnitudes de telemetría (altitud, velocidad, posición, etc…) en la ventana principal.
- Ficheros con formato vectorial para procesar con otros programas de edición las gráficas.
- Análisis de las fases del plan de vuelo en la ventana principal.
- Análisis de los conflictos con intrusos en la ventana principal.
- Ficheros de tipo KML (Keyhole Markup Language). Ficheros necesarios para exportar los datos a Google Earth.

Una de las funcionalidades más importantes de este software es la capacidad de procesar varios patrones o estructuras diferentes de datos. Para ofrecer esta flexibilidad a la hora de leer los datos de telemetría será necesario definir un lenguaje que describa el formato en el que estos han sido guardados. Este lenguaje, está basado en el estándar XML (eXtensible Markup Language) y está explicado en detalle en el capítulo 3 de esta memoria. Con este lenguaje se crea un fichero de configuración que será usado por el programa para leer el fichero de telemetría.

Para comprobar el correcto funcionamiento de nuestra aplicación, se usarán ficheros de diversas plataformas (ver Figura 2.3), cada una de las cuáles tiene una estructura de datos distinta:

- *Datos de vuelo de un helicóptero real no tripulado*: el software comercial que almacena estos datos pertenece a UAVNAvigation [14]. En el fichero de telemetría generado se muestran todos los datos, que comparten el mismo instante de tiempo, en una sola línea.
- *Datos del simulador de aviones no tripulados*: este es el formato utilizado en los ficheros del UAS y los intrusos del simulador ISIS. En este tipo de ficheros, cada línea es un campo con un conjunto de datos, independientemente de si éste comparte el mismo instante de tiempo con otros conjuntos de datos.
- *Datos de vuelos realizados por un avión no tripulado de NASA*: de nuevo se muestran todos los datos relativos al mismo instante de tiempo en una sola línea. No obstante, no contiene los mismos datos que el helicóptero y, por lo tanto, se considera una plataforma diferente.

En la Figura 2.3 se puede ver un esquema de los diferentes tipos de fichero probados en la aplicación.

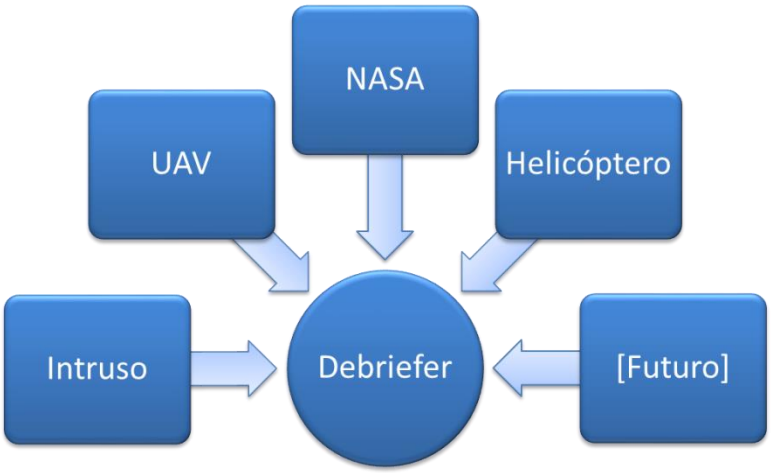


Figura 2.3 Ejemplos de datos a tratar

2.3 **Estructura del proyecto**

La estructura de la aplicación se puede desglosar en tres bloques generales, clasificados según la función que desarrollan.

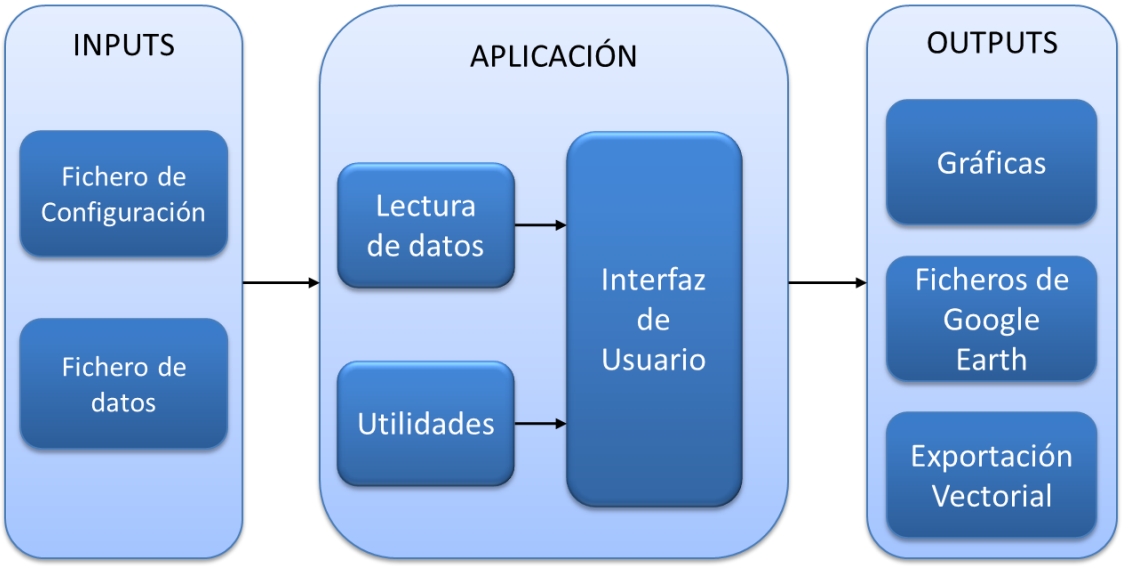


Figura 2.4 Estructura del proyecto

En la Figura 2.4 está representada la relación de entradas-salidas de la aplicación. El software recibe dos tipos de archivos de entrada, un fichero de configuración y otro con los datos. Estos ficheros son procesados por los diferentes componentes de la aplicación pada dar como resultado los diferentes sistemas de visualización y análisis implementados.

Dentro de la aplicación se pueden diferenciar tres grandes bloques. El primer bloque, realiza la *Lectura de datos*, es el encargado de recibir los datos de entrada y almacenarlos en estructuras internas para que puedan ser utilizados por el resto de bloques. El segundo bloque, *Utilidades*, provee al sistema de las herramientas necesarias para simplificar el procesado de los datos. Por último, el tercer bloque, *Interfaz de usuario*, contiene todos los componentes necesarios para generar la visualización adecuada de los datos. Finalmente, los outputs son gráficas, ficheros de Google Earth o ficheros en formato vectorial.

A partir del capítulo siguiente se entrará en detalle dentro de cada uno de estos bloques.

CAPÍTULO 3. LECTURA DE DATOS

El primer bloque de la aplicación tiene como objetivo realizar el procesado de los datos. Este bloque es el encargado de recibir unos ficheros de entrada, procesarlos y guardarlos en unas estructuras para su posterior análisis.

3.1 Introducción

Debido a que los datos de los ficheros de telemetría pueden estar organizados de diversas formas, un mal diseño en las estructuras de datos internas del programa o un mal planteamiento en su procesado, puede dar lugar al principal problema de la aplicación. Por lo tanto esta es una parte muy importante del programa. En la Figura 3.1 se muestra el diseño realizado para este bloque. En ella se pueden observar los ficheros de entrada del programa (el de configuración y el de datos), el componente encargado de procesar los ficheros (el *Procesador de datos*), y las estructuras de datos usadas (*Estructura de datos* y de configuración).

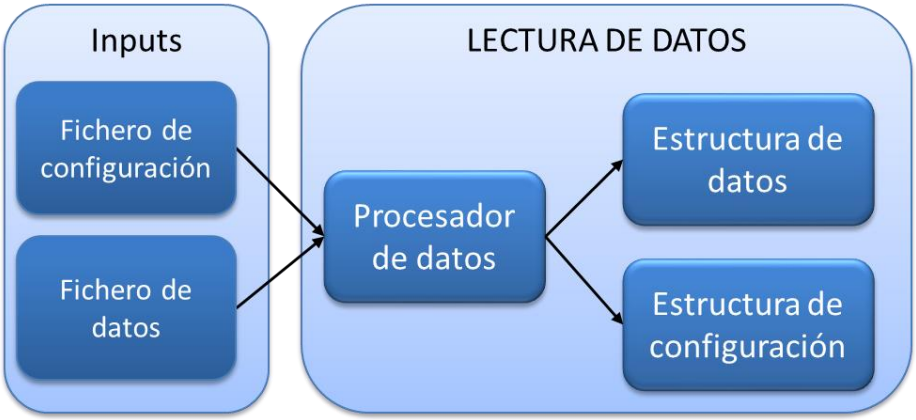


Figura 3.1 Diseño de la lectura de datos

Un fichero de telemetría puede agrupar los datos en función del campo al que hacen referencia, pudiendo encontrar varias líneas que pertenecen a un mismo instante de tiempo. Por otro lado, se pueden encontrar ficheros donde todos los datos de un mismo instante de tiempo se guardan en una única línea. Esto significa que cada una de las plataformas que generan ficheros de telemetría puede tener un formato o estructura de datos diferente. Para dar solución a este problema, se ha creado un lenguaje que permite definir la estructura de estos ficheros. Este lenguaje es empleado para crear un fichero de configuración que definirá el formato y la estructura de los datos de un determinado fichero de telemetría. Además, este lenguaje, permite definir cada uno de los datos del fichero de telemetría, otorgándoles un nombre y las unidades que tendrán en caso de tratarse de valores numéricos.

Una vez que el usuario ha definido el fichero de configuración, la aplicación almacena en una estructura interna los datos contenidos en él. De esta manera, el programa puede acceder en todo momento a los datos de configuración.

Por otro lado se encuentra la estructura que almacenará los datos del fichero de telemetría. Esta estructura está organizada en base al fichero de configuración y, por lo tanto, es muy similar a la estructura anterior.

Por último lugar se encuentra el procesador de datos, que es el encargado de crear las estructuras a partir de los ficheros necesarios para cada una de ellas (fichero de configuración y de telemetría). Además ofrece un mecanismo de filtrado de datos en caso de que el usuario no desee procesar todos los datos. Este mecanismo es muy útil en situaciones en las que el fichero de datos a analizar es muy extenso pero el análisis que se desea no es muy preciso. De esta manera, el programa realizará la lectura de los datos en menor tiempo.

3.2 **Lenguaje de configuración**

Como se ha explicado en el apartado anterior, se ha creado un lenguaje para definir el formato y la estructura de los datos del fichero de telemetría. Este lenguaje está basado en el estándar XML, eXtensible Markup Language. Esta definición se guarda en un fichero de configuración.

3.2.1 Diseño

El lenguaje XML es un lenguaje abstracto que está definido por etiquetas, las cuales se especifican entre los símbolos < y >. A estas etiquetas y su contenido se les conoce como elementos. Un elemento puede contener varias etiquetas anidadas, formando árboles. Gracias a esto, el lenguaje XML permite definir de una manera sencilla la estructura que tendrán los ficheros de telemetría.

Para definir un fichero de telemetría el primer paso es definir el formato de origen, es decir, el esquema utilizado al guardar los datos. De esta manera, la aplicación sabe de qué modo leer el fichero. Una vez definido el formato, se deben de definir los campos que tiene el fichero de telemetría, indicando en cada caso que datos componen ese campo. Por último, este lenguaje permite definir operaciones que la aplicación posteriormente realizará con los parámetros indicados en ellas.

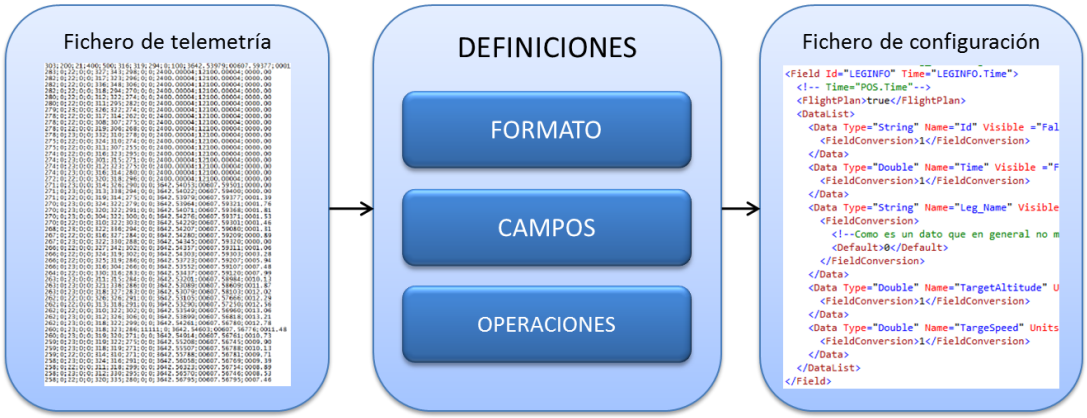


Figura 3.2 Definiciones fichero de configuración

En la Figura 3.2 está representada la relación entre el fichero de telemetría y el fichero de configuración. Definiendo el formato, los campos y las operaciones del fichero de telemetría, se crea el fichero de configuración.

3.2.2 Implementación

Se ha decidido dividir el fichero de configuración en tres partes, por un lado tenemos el elemento que contendrá la información de formato, especificado por la etiqueta “<Format>”, por otro lado tendremos el elemento con información de los datos contenidos en el fichero de telemetría, especificado por la etiqueta “<Field>”, y por último el elemento etiquetado como “<Operations>”, que nos permite definir como serán visualizados los datos.

El elemento “Format” debe incluir información sobre el tipo de fichero de telemetría (si los datos vienen ordenados en filas, columnas, si aparece el tiempo…), el separador de estos datos y el margen a aplicar en el fichero (línea o espacio a partir del cual empezar a leer). Para ello se crea una sintaxis específica.

Como se ha explicado en el apartado anterior, el lenguaje XML es un lenguaje que está definido por etiquetas, que se especifican entre los símbolos < y >. Por lo tanto, la definición del formato debe aparecer dentro de la etiqueta <Format>. Dentro de esta etiqueta, con etiquetas anidadas, se dará valor a los parámetros que lo definirán:

- <*Type*>: en esta etiqueta se define el tipo de fichero de telemetría. El fichero de telemetría puede ser de diferentes tipos:
    - Tipo 0: El fichero tiene los campos ordenados en filas, donde cada fila tiene un campo que contiene varios datos de un único instante de tiempo. La fila siguiente puede contener otro campo en el mismo instante de tiempo.
    - Tipo 1: El fichero tiene en la misma línea todos los valores de un campo de todos los instantes de tiempo.

    - Tipo 2: El fichero tiene todos los datos que comparten el mismo instante de tiempo en la misma fila, de este modo la fila siguiente tendrá un valor de tiempo diferente.
- <*Split*>: en esta etiqueta se define el separador de datos. Los datos pueden estar separados por un tabulador (valor 0), por un espacio (valor 1), o por una cadena de caracteres (valor 2).
- <*Margin*>: es el margen superior a partir del cual se comenzará a leer el fichero. Este margen es necesario en caso de que un fichero de telemetría contenga los nombres de los datos o las descripciones en las primeras líneas. Gracias al margen, en estos casos, las primeras líneas se obviarán.
- <*LerftMargin*>: es el margen lateral izquierdo a partir del cual se comenzará a leer el fichero. Este margen se aplica si no se desea leer el primer valor de cada línea.

En la Figura 3.3 se puede ver un ejemplo de la definición del formato, en el cual se muestra la manera correcta de usar las etiquetas. En este ejemplo el fichero es del tipo 0, es decir, cada línea representará un campo de datos en un único instante de tiempo. Además, el separador de datos es un tabulador y no existe margen superior ni margen izquierdo.

```
<Format>
   <Split Id="">0</Split>
   <Margin>0</Margin>
   <LeftMargin>0</LeftMargin>
   <Type>0</Type>
</Format>
```

Figura 3.3 Definición de formato del fichero de configuración

Una vez definido el formato, se deben definir la estructura de datos del fichero. Como se ha comentado anteriormente, esta estructura está formada por campos, cada uno de los cuales puede contener uno o varios datos que están asociados a él. La definición de cada campo se realiza con la etiqueta <Field>. Dentro de esta etiqueta encontramos dos atributos, su identificador o “Id”, es decir, el nombre del campo, y la ruta donde se encuentra el valor de tiempo asociado. La ruta de la variable tiempo viene especificada por el identificador del campo y el identificador del dato.

Una vez definido el campo, dentro de la etiqueta <Data>, con sus atributos, se definen los datos asociados a éste. Estos datos se definen con la etiqueta <Field> y tienen una serie de atributos como su nombre, el tipo de dato (si es numérico o no), las unidades que tiene, y una propiedad que indica si el dato ha de ser visible.

En la Figura 3.4 se muestra un ejemplo de la definición de campos. En ella se puede observar que con la etiqueta <Field> se define el campo que tiene como identificador “POS”. Además, el siguiente atributo indica que el tiempo al que se

asocia este campo está en la ruta POS.Time. Esto significa que dentro del campo POS, el tiempo estará en el dato Time.

En la Figura se puede observar también que el campo POS contiene los datos de Time, Latitud y Longitud, bajo la etiqueta <Data>. Estos datos están definidos por su nombre, el tipo de dato, sus unidades, y la propiedad Visible.

```
<Field ID="POS"  Time="POS.Time">
  <DataList>

      <Data Type="Double" Name="Time" Visible ="False" Units="h"/>
      <Data Type="Double" Name="Latitude" Units="rad"/>
      <Data Type="Double" Name="Longitude" Units="rad"/>

  </DataList>
</Field>
```

Figura 3.4 Definición de campos en el fichero de configuración

Por último, mediante la etiqueta <Operations>, se definen las operaciones que el usuario quiere realizar con los datos del fichero de telemetría.

Para definir una operación, es necesario indicar el tipo de operación a realizar. Actualmente, el programa dispone de tres opciones de operaciones. El primer tipo es la operación “KML” que sirve para generar ficheros de Google Earth. Por segundo lugar, la operación “PIE” que genera un gráfico especial, compuesto por un gráfico circular y una tabla que muestra los datos representados en él. Y por último lugar la operación “SERIES” que sirve para crear gráficos de series.

Otros parámetros a definir de las operaciones son el nombre de la operación, el nombre del método que realizará la operación, el conjunto de datos necesarios, y en caso de que la operación genere un fichero, el nombre de este fichero.

```
<Operations>
      <KML>
            <Name>Generate Dynamic KMLGraph</Name>
            <Function>generarKMLDynamic</Function>
            <Dato>POS.Latitude</Dato>
            <Dato>POS.Longitude</Dato>
            <Dato>POS.Altitude</Dato>
            <Dato>POS.Time</Dato>
            <OutputName>Simulator_temporal_flightplan</OutputName>
      </KML>
</Operations>
```

Figura 3.5 Definición de operaciones

En la Figura 3.5 está ejemplificada una operación de tipo “KML”. A continuación se nombra la operación, en este caso “GenerateDynamic KML Graph, y se declara la función que la llevará a cabo. Por último se especifican los datos que deben usarse y el nombre que tendrá el archivo creado, en este ejemplo “*Simulator_temporal_flightplan*”.

3.3 **Estructura de configuración**

Como se ha explicado en el apartado 3.2 (Lenguaje de configuración), el fichero de configuración determina el formato y los campos que posee el fichero de telemetría. Además, contiene una serie de operaciones que la aplicación debe realizar sobre él. Por este motivo, esta estructura se compone de dos partes, la estructura que define el fichero de telemetría y la estructura que contiene las operaciones a realizar.

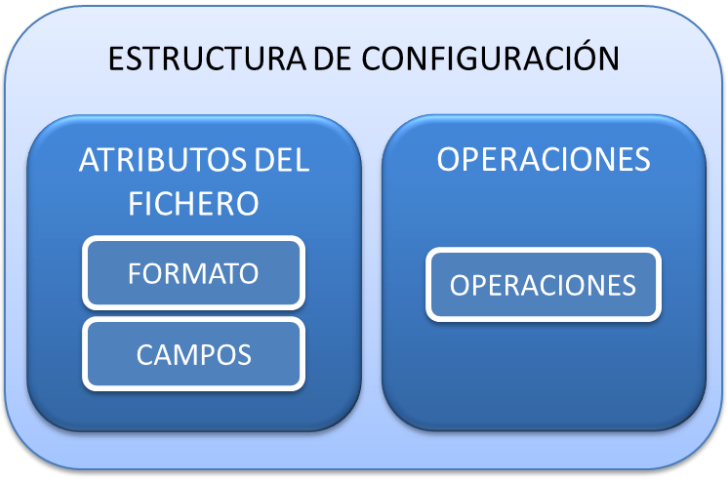


Figura 3.6 Diseño de la estructura de configuración

En la Figura 3.6 se muestra un esquema de los dos bloques que forman la estructura de configuración. El primer bloque contiene los atributos del fichero de telemetría (formato y campos) y el segundo las operaciones que se realizarán en él.

- Atributos del fichero

En esta parte de la estructura de configuración se almacena el formato (márgenes, separación de los datos, etc…) del fichero de telemetría al que hace referencia. Por otro lado, se almacena una lista con la definición de todos los datos que va a contener este fichero. A continuación se puede observar un ejemplo de este tipo de estructura.

Como se ha explicado en el apartado 3.2 (Lenguaje de configuración), los datos del fichero de telemetría pueden estar agrupados por campos. Estos campos contienen un identificador que dará nombre al campo y una ruta que

señalará el lugar del fichero dónde se encuentra el tiempo asociado a este campo. Además contienen la definición de cada uno de los datos que formarán parte de él.

Los datos individuales vienen definidos por el nombre del dato, las unidades en las que aparece en el fichero de telemetría y una propiedad que indica si debe representarse o no en un gráfico lineal.

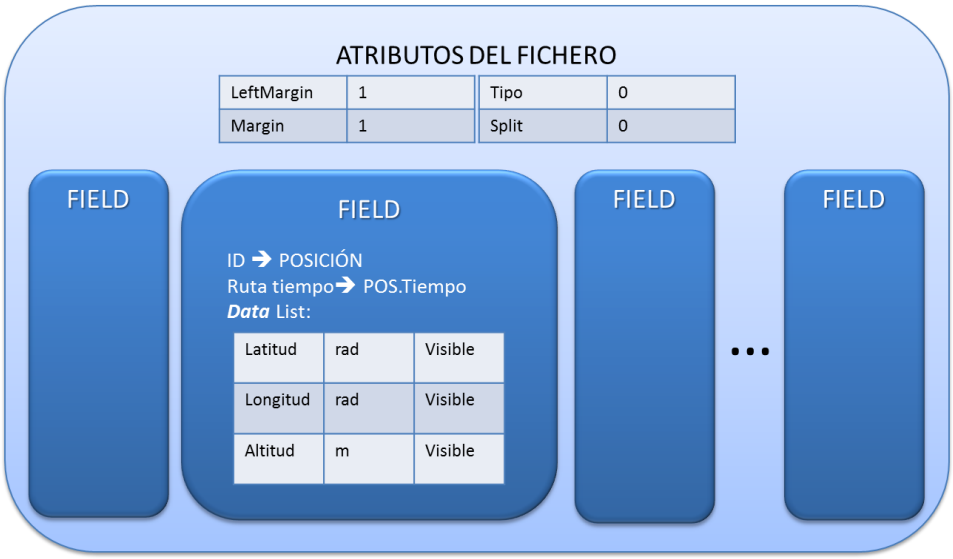


Figura 3.7 Estructura de configuración

La Figura 3.7 representa mediante un ejemplo la estructura de configuración. En la parte superior de la Figura se define el formato del fichero de telemetría a leer (márgenes, tipo de sintaxis, separación de los datos). En la parte inferior hay una lista de campos, llamados “FIELD”. Estos campos contienen la definición de cada uno de los datos asociados a ellos. En el ejemplo anterior el campo POSICIÓN contiene la definición de los datos de Latitud, Longitud y Altitud.

- Operaciones

Esta segunda parte de la estructura de configuración es la encargada de almacenar las operaciones o funciones que el usuario decide declarar en el fichero de configuración. Se compone de los elementos siguientes:

  - *Nombre*: el nombre de la operación a realizar.
  - *Tipo*: el tipo de operación a realizar:
  - *Datos*: el conjunto de datos necesarios para realizar esta operación.
  - *Función*: el nombre del método o función que llevará a cabo esta operación.
  - *OutputName*: en caso de que la operación lleve a cabo la creación de un fichero de salida, este será el nombre de este fichero.

Figura 3.8 Estructura de Operaciones

La Figura 3.8 contiene un ejemplo de una lista de *Operaciones*, donde se pueden apreciar todos los atributos explicados anteriormente. En esta Figura vemos como ejemplo, una operación para crear un archivo de tipo KML (archivo de Google Earth), llamado uav_dynamic, con sus funciones y datos correspondientes.

3.4 **Estructura de datos**

Este elemento es el encargado de almacenar los datos del fichero de telemetría. Esta estructura depende de la estructura de configuración y, por lo tanto, es muy similar a ésta.

Esta estructura está formada por una lista de campos donde cada campo representa un grupo de datos de telemetría. Es posible que en el fichero de telemetría aparezca más a menudo la información de un campo que la de otro. Esto es así porque hay campos de datos que solo es necesario mostrar cuando ha habido un cambio, por ejemplo, cuando hay un cambio en la fase de vuelo, o en el modo del piloto automático. Por este motivo, es necesario que cada campo tenga un valor de tiempo y así saber a qué instante pertenece.

En cuanto a los datos, estos están formados por su nombre y su valor numérico. A continuación se muestra un ejemplo de este tipo de estructura.

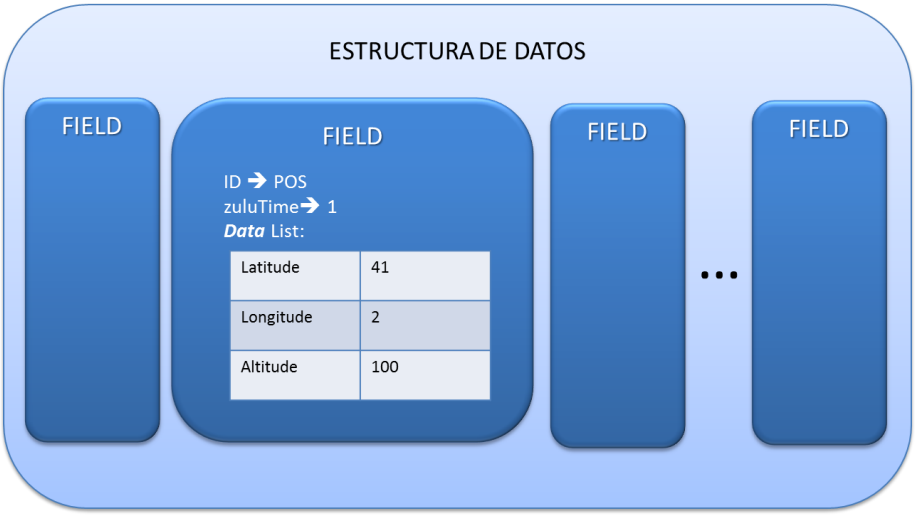


Figura 3.9 Estructura de datos

En la Figura 3.9 se puede ver la estructura de datos completa. Esta estructura está formada por una lista de campos, llamados “FIELD”. Estos campos contienen los datos del fichero telemetría y el tiempo asociado a ellos. En el ejemplo anterior el campo POSICIÓN contiene los datos de Latitud, Longitud y Altitud.

3.5 **Procesado de datos**

Este elemento es el componente encargado de recibir los ficheros de entrada y almacenar sus datos en las estructuras destinadas a ello.

Los ficheros de telemetría provienen de diferentes plataformas y, por lo tanto, su formato o sintaxis puede ser diferente. Por este motivo, el objetivo de este elemento es transformar esa sintaxis en una estructura común apropiada para su almacenamiento. Para ello se necesita el fichero de configuración que defina cuál es la sintaxis empleada en el fichero de telemetría.

Este elemento se compone de herramientas creadas específicamente para esta tarea con las siguientes funcionalidades:

- Mediante un fichero de configuración genera una estructura que permita definir la sintaxis del fichero de telemetría. En esa misma estructura almacena las operaciones o conversiones que el usuario desea aplicar en los datos de telemetría. Esta estructura ha sido explicada en el apartado 3.3 (Estructura de configuración).
- Mediante los datos almacenados en la estructura anterior, decodifica el fichero de telemetría y almacena los datos en las estructuras explicadas en el apartado 3.4 (Estructura de datos).
- Ofrece un mecanismo de filtrado de datos en caso de que el usuario no desee procesar todos los datos. Este mecanismo es muy útil en situaciones en las que el fichero de datos a analizar es muy extenso pero el análisis que

se desea no es muy preciso. De esta manera, el programa realizará la lectura de los datos en menor tiempo.

CAPÍTULO 4. Utilidades

Una vez generadas las estructuras con los datos de los ficheros de entrada, la aplicación necesita una serie de elementos que lleven a cabo las operaciones necesarias para proporcionar las funcionalidades específicas del software. Este conjunto de elementos se conoce por *Utilidades* y están esquematizados en la Figura 4.1.

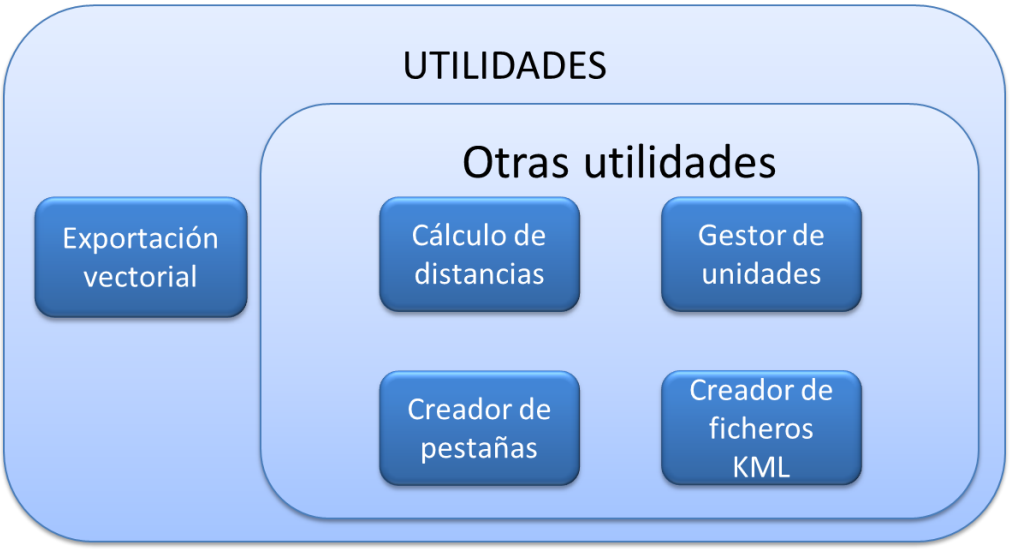


Figura 4.1 Estructura de las Utilidades

En este esquema se puede apreciar una clasificación de las utilidades. Por un lado, está la exportación a gráficos vectoriales, que es la encargada de convertir las gráficas en ficheros con formato vectorial y de este modo permitir editarlas con otros programas. Por otro lado están el resto de utilidades:

- El *creador de pestañas:* que permite crear y manipular las pestañas que aparecerán en la ventana principal.
- El *cálculo de distancias*: que, entre otras cosas, calcula la distancia entre dos puntos.
- El *creador de ficheros KML:* que genera ficheros de Google Earth.
- El *gestor de unidades*: que permite cambiar de unas unidades a otras.

A continuación se explican estas utilidades en detalle.

4.1 Exportación vectorial

Uno de los requisitos de la aplicación es poder exportar las gráficas a formato vectorial, concretamente a SVG (del inglés *Scalable Vector Graphics*). SVG es un formato de imagen vectorial basado en XML para gráficos de dos dimensiones que tiene soporte para la interactividad y animación. La principal ventaja de exportar las gráficas a formato SVG es ofrecer la posibilidad de

edición sin pérdida de calidad. De este modo se puede realizar un analisis más exahustivo de sus diferentes puntos o hacer un postprocesado.

Para poder exportar satisfactoriamente a formato vectorial se requiere una configuración básica en la máquina donde se ejecuta el programa. Esta configuración está descrita en las indicaciones técnicas situadas en el anexo de esta memoria.

Con la intención de que esta funcionalidad sea lo más versátil posible y que se pueda implementar fácilmente en toda la parte visual, se han creado tres componentes para estructurar los parámetros necesarios y poder realizar la exportación con los mismos atributos que se muestran por pantalla en ese momento.

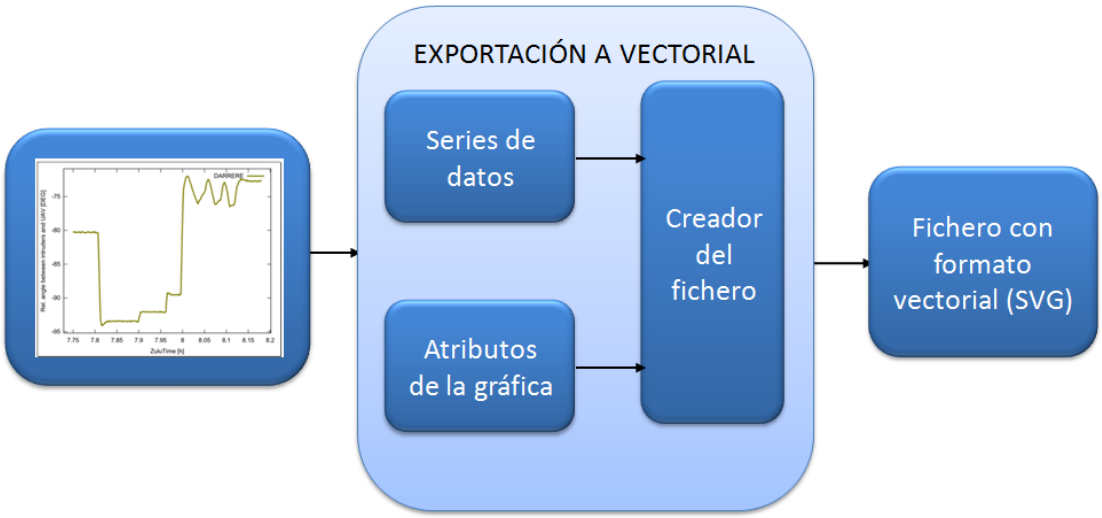


Figura 4.2 Componentes para la exportación a vectorial

En la Figura 4.2 está representada la relación entradas-salidas de la exportación a vectorial. En primer lugar aparece la gráfica que el usuario desea exportar. Con los datos contenidos en esta gráfica y las herramientas creadas para esta utilidad (mostradas en el centro de la Figura), se genera el fichero con formato vectorial.

El primer componente, nombrado *Series de datos*, es el encargado de almacenar los datos que se deben exportar al fichero SVG. No obstante, si el usuario quiere exportar únicamente una parte de la gráfica en la que se ha hecho zoom es necesario almacenar el rango de datos que muestra. Por este motivo se crea el segundo componente, *Atributos de la gráfica*, donde se almacena la parte de la gráfica que se está visualizando y la leyenda. Una vez almacenados los datos y los atributos de la gráfica, la herramienta llamada Creador del fichero realiza las operaciones necesarias para exportar estos datos.

- *Atributos de la gráfica*

En este componente se almacenan los atributos propios de la gráfica que se quiere exportar tales como las leyendas de los ejes vertical y horizontal, y los valores máximos y mínimos sobre los que se visualiza la gráfica. Estos datos son de especial relevancia para poder determinar la zona que se está visualizando una vez que se haya hecho un zoom sobre alguna parte de la gráfica. De este modo, el fichero vectorial generado ya está enfocado en esa parte de la gráfica en concreto.

- *Series de datos*

Es el componente en el que se almacenan las series de datos que componen la gráfica en cuestión. En él quedan definidos el conjunto de puntos mostrados, el color, grosor y tipo de línea. En esta herramienta, se guarda también el nombre de la serie de datos así como las unidades en las que se expresan dichos datos.

- *Creador del fichero*

Esta herramienta es la encargada de generar el fichero SVG con los datos almacenados previamente (series de datos y atributos de la gráfica).

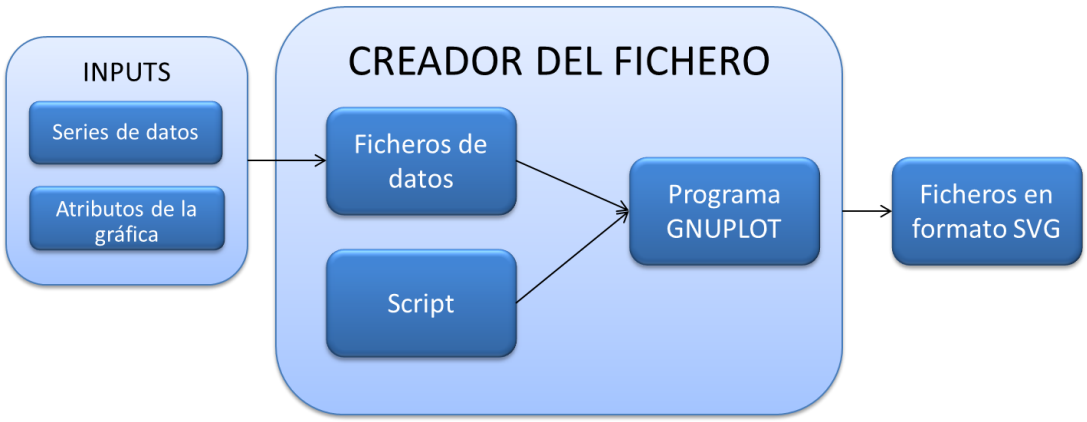


Figura 4.3 Creador del fichero de exportación a formato vectorial

Como se puede apreciar en la Figura 4.3, la herramienta creador del fichero, con los datos almacenados en *Series de datos* y *Atributos de la gráfica* genera unos *Ficheros de datos* en formato ".dat". Estos ficheros son imprescindibles y se almacenan en una carpeta llamada "SVG Files" que la propia aplicación crea. A continuación, el generador del fichero construye un *Script*, que es una serie de instrucciones que posteriormente serán ejecutadas. En estas sentencias, se llama a los *Ficheros de datos* previamente creados y se proporciona las indicaciones necesarias para que se respeten los rangos, títulos de ejes, colores, leyendas y formas de los puntos que visualiza o ha seleccionado el usuario. Finalmente, el creador del fichero ejecuta el script mediante el *programa GNUPLOT* y este último construye los *Ficheros en formato SVG* de manera automática. Estos archivos son también guardados en la carpeta "SVG Files".

4.2 **Otras utilidades**

A lo largo del desarrollo del software se ha realizado una búsqueda de componentes, herramientas, librerías o ejemplos que ayudaran en la realización de los objetivos marcados. Después de esta búsqueda ha sido necesario aprender el funcionamiento y entender cada uno de estos componentes para poder modificarlos e implementarlos en la aplicación.

- Creador de pestañas: para esta utilidad se ha utilizado la herramienta "*CloseableTabItem*" encontrada en un ejemplo [9]. Es una herramienta auxiliar que aporta funciones para manipular las distintas pestañas que componen la aplicación. Esta herramienta era necesaria puesto que la tecnología WPF no ofrece la posibilidad de manipular las pestañas y, en particular, de cerrarlas con un botón. No ha sido necesario modificar esta herramienta puesto que satisface las necesidades requeridas.

  En esta aplicación se sitúan las herramientas de análisis dentro de pestañas y de este modo se facilita una imagen más moderna y funcional permitiendo añadir un título a dicha pestaña y dando la posibilidad de cerrarlas. Se puede ver un ejemplo de estas pestañas en la Figura 4.4.

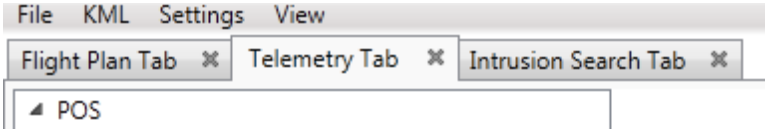


Figura 4.4 Creador de pestañas

- Creador de ficheros KML: para esta utilidad se ha utilizado la herramienta "*KMLWriter"* [13] del grupo ICARUS, que contiene los métodos necesarios para construir los ficheros con formato KML (ficheros utilizados en Google Earth). La herramienta "*KMLWriter"* ha sido modificada a medida para este proyecto con el objetivo de que sea capaz de trabajar con todas las unidades posibles. Estas unidades son tanto las de posición (radianes o grados sexagesimales) como las de tiempo. Además se ha modificado para mostrar los recorridos en varios colores y así poderlos diferenciar en caso de visualizar más de uno a la vez.

  En la Figura 4.5 se muestra el recorrido de un vuelo del simulador ISIS mediante el programa Google Earth.

Figura 4.5 Recorrido de un vuelo del simulador en Google Earth

Esta herramienta también permite visualizar el recorrido de forma dinámica, es decir, teniendo en cuenta el instante de tiempo en el cual sobrevuela cada punto. De esta manera, el usuario puede ver el recorrido real paso a paso. Esto es útil cuando se comparan dos aeronaves y así verificar si en algún momento se han cruzado, cosa que de otra manera no sería posible.

- Cálculo de distancias: para esta utilidad se utiliza la herramienta "*Point"* [15], una herramienta creada por el grupo ICARUS. Esta herramienta se usa, entre otras cosas, para hallar distancias entre dos puntos definidos por su latitud y longitud. Para esta aplicación ha sido necesario mejorar esta herramienta ya que se requerían más funcionalidades.

  Las mejoras en la herramienta permiten calcular las distancias horizontales y verticales entre dos aeronaves a lo largo de un intervalo de tiempo. Para ello, primero se ha de calcular en que intervalos de tiempo coinciden ambas. Una vez conocidos los instantes donde coinciden, se ha de calcular la distancia que los separa para poder mostrar su evolución en el tiempo. También permite calcular la longitud recorrida por el UAS a lo largo de un tramo, por adición de las distancias que separan los puntos que sobre vuela al recorrerlo.

  Las fórmulas empleadas están referenciadas en [11] y son extensamente explicadas en la siguiente sección.

- Gestor de unidades: para poder satisfacer el requisito de cambios de unidades, se utiliza la herramienta "*Units"* [16]. Esta herramienta está creada por el grupo ICARUS y ha sido adaptada con las necesidades específicas de la aplicación.

Debido a que la aplicación es muy genérica y puede recibir datos de varias plataformas, el software no conoce las unidades de origen. Por ese motivo se ha modificado la herramienta para que, gracias al fichero de configuración, determine las unidades de origen. De esta manera, además de permitir los cambios de unidades, identifica la unidad de cada dato y la clasifica en función de la magnitud que representa, y aplica los factores de conversión necesarios según la unidad seleccionada por el usuario.

4.3 **Cálculo de distancias**

En todo momento se informa al usuario de las unidades en las que se muestran las distintitas magnitudes pero resulta relevante informar sobre cómo se han realizado algunos cálculos, sobre todo a nivel del análisis de intrusos. De esta manera el usuario podrá analizar los resultados desde una perspectiva más crítica.

Se utiliza la “Haversine” fórmula [11] para calcular la distancia de círculo máximo u ortodrómica entre dos puntos, es decir, la distancia más corta que une esos puntos sobre la superficie de la Tierra.

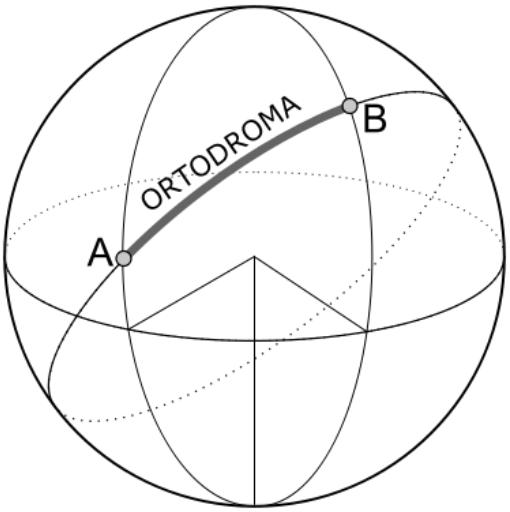


Figura 4.6 Distancia de círculo máximo

En la Figura 4.6 se puede ver la distancia ortodrómica o de círculo máximo, es decir la distancia mínima entre dos puntos sobre la superficie terrestre.

El “Haversine” fue publicado por Roger Sinnott en la revista Sky&Telescope en 1984 ("Virtudes del Haversine"). La fórmula de “Haversine” se expresa de la forma siguiente:

$$\mathrm{a} = \sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2) \quad \textbf{(4.1)}$$

$$c = 2 \cdot \mathrm{atan}\left(\frac{\sqrt{a}}{\sqrt{1-a}}\right) \quad \textbf{(4.2)}$$

$$d = R \cdot c \quad \textbf{(4.3)}$$

Donde φ es latitud, λ es longitud, d la distancia calculada, R el radio de la tierra (6.371 km). Los angulos han de ser expresados en radianes.

Las fórmulas empleadas son para cálculos sobre la base de una tierra esférica (ignorando los efectos elipsoidales) siendo suficientemente preciso para esta aplicación. De hecho, la tierra es ligeramente elipsoidal y utilizando un modelo esférico da errores con un máximo de un 0,3%.

El inconveniente de estas fórmulas es que no se toma en cuenta la altura y, por lo tanto, la distancia real entre las aeronaves es mayor a la distancia de círculo máximo. No obstante, para el análisis de tráfico aéreo es suficiente, puesto que dos aeronaves que entran en conflicto están lo suficientemente cerca para que dicha diferencia sea nula. Además, en el análisis de tráfico aéreo no es necesario tomar en cuenta las distancias verticales puesto que si dos aeronaves no se encuentran en el mismo nivel de vuelo no existe conflicto entre ellas.

Por último, para el análisis de plan de vuelo, la distancia recorrida por la aeronave no puede ser calculada desde el punto inicial hasta el final puesto que la fórmula de Haversine calcula la distancia mínima entre estos puntos. Para solucionar este inconveniente se calculan sucesivamente las distancias entre dos puntos contiguos del recorrido, y una vez sumadas equivalen a la distancia total realmente recorrida.

CAPÍTULO 5. INTERFAZ DE USUARIO

La interfaz de usuario es el elemento del proyecto que contiene los componentes gráficos de la aplicación. La tecnología utilizada en el desarrollo de la parte gráfica de la aplicación es WPF (Windows Presentation Foundation). WPF es una tecnología que se basa, entre otras cosas, en el uso de controles. Un control es un componente visual que contiene una o varias funciones específicas (un botón, un menú desplegable, una caja de texto…). Para simplificar la implementación y el desarrollo de una aplicación, WPF permite crear controles de usuario. Los controles de usuario son agrupaciones de controles básicos que se unen para formar un solo control. De esta manera, el programador puede reutilizar estos controles de usuario tantas veces como desee. Para esta aplicación en concreto, se han creado varios controles de usuario, que se explicarán en detalle en la siguiente sección.

Como se ha explicado el capítulo 2 (Arquitectura y diseño de la aplicación), el programa debe realizar tres tipos de análisis (análisis de telemetría, de plan de vuelo y de tráfico aéreo). Por este motivo, se han creado tres bloques visuales diferentes, llamados *Ventanas de análisis*, cada uno de ellos encargados de realizar uno de estos tres tipos de análisis. Dichos bloques constarán de uno o varios de los controles de usuario mencionados anteriormente.

En este capítulo se explican en detalle los controles de usuario creados para la aplicación, y las ventanas de análisis que están formadas por dichos controles.

5.1 Controles de usuario

Los controles de usuario permiten crear una aplicación gráfica compleja a partir de un conjunto de pequeños fragmentos relativamente simples. Esto resulta muy útil cuando el programador quiere aprovechar un conjunto de controles básicos o un diseño de ventana previamente creado para reutilizarlo en otras partes de una aplicación. En esta aplicación en particular, como se puede observar en la Figura 5.1, se han creado varios controles de usuario.

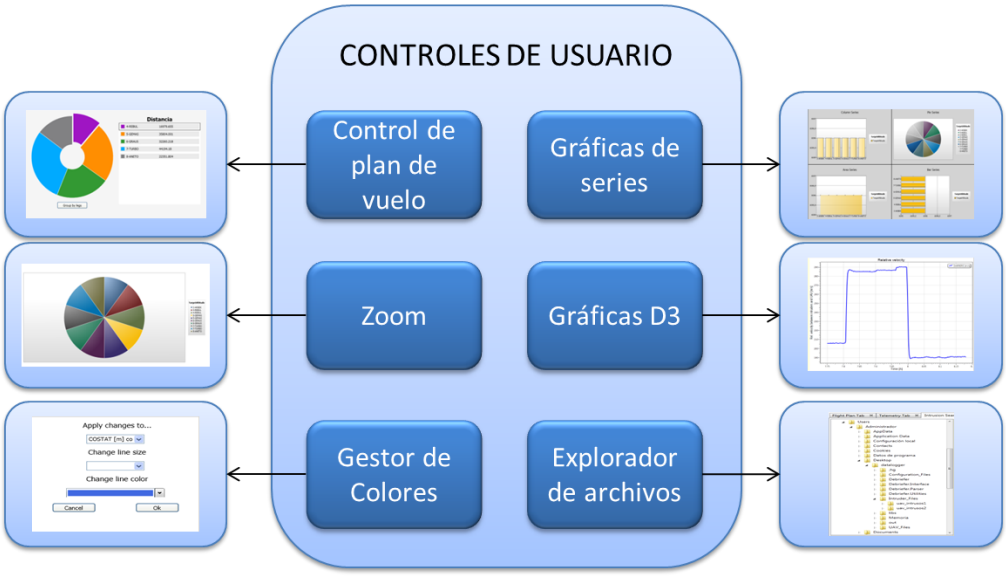


Figura 5.1 Controles de usuario de la aplicación

Estos controles de usuario serán utilizados a lo largo del proyecto en varias ocasiones. A continuación se detallan cada uno de ellos:

- Gestor de colores

Uno de los requisitos de la aplicación es permitir al usuario elegir el color o grosor de las líneas que se representan en el gráfico lineal. Para cumplir dicho requisito se ha creado el *Gestor de colores*.

El *Gestor de colores* permite al usuario elegir un color y un grosor de línea de la serie de datos que desee (en caso de que haya varias). Además, para que dicha elección no se pierda, se almacena el color y el grosor con el que se han representado gráficamente todas las series de datos dentro de una misma gráfica. De este modo, en caso de volver a representar una serie, siempre será con el mismo color y grosor de línea. En la figura 5.2 se muestra el control que permite al usuario realizar estos cambios.

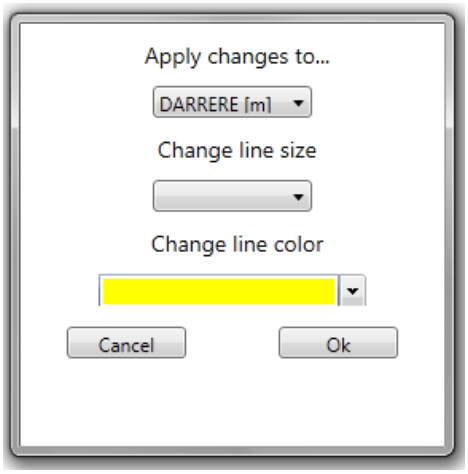


Figura 5.2 Gestor de colores

En la Figura anterior que sirve como ejemplo, el usuario ha elegido cambiar a color amarillo la línea que representa la serie de datos DARRERE [m].

- *Gráficas de series*

Para representar gráficamente una serie de valores numéricos asociados a palabras, es mejor utilizar gráficos de series. Para ello se ha utilizado la librería WPFToolkit [19]. Esta librería contiene controles que generan gráficos de series pero ha sido mejorada para poder generar cuatro gráficos al mismo tiempo. Los gráficos de series implementados son *Bar Chart, Area Chart, Column Chart o Pie Chart.* Se puede ver un ejemplo de estos gráficos en la Figura 5.3.

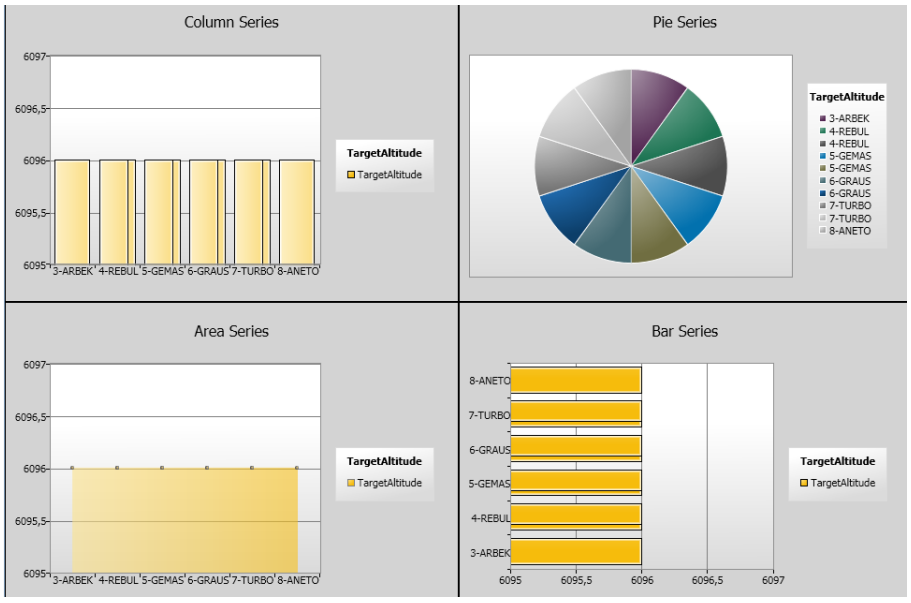


Figura 5.3 Gráficas de series

- Explorador de archivos

Para realizar la búsqueda de ficheros de telemetría de los intrusos para el análisis de plan de vuelo, es más útil elegir una carpeta en la que se encuentren todos los ficheros en lugar de seleccionarlos uno a uno. Por este motivo se ha decidido utilizar un explorador de archivos que muestre todas las carpetas del disco duro del usuario. Para la creación de dicho explorador, se ha utilizado un ejemplo ya existente [10]. Por lo tanto, este control de usuario crea un árbol de carpetas representando el disco duro del usuario para que éste seleccione la carpeta donde se encuentran los archivos de intrusos. En la Figura 5.4 se puede ver un ejemplo de este control de usuario.

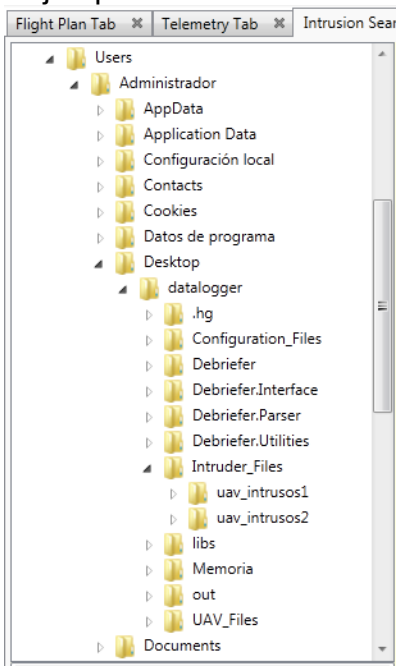


Figura 5.4 Explorador de archivos

- *Zoom*

Debido a que en algunas ventanas puede existir mucha información, en numerosas ocasiones es necesario mostrar algún elemento en un tamaño mayor para ver la información que dicho elemento ofrece. Para solucionar este problema, se ha creado un control de usuario llamado “Zoom”. Este control de usuario recibe una imagen o un gráfico y lo muestra en una ventana emergente para que sea visualizado en el tamaño que el usuario desee. Este control de usuario es muy útil para las gráficas de series que tienen una gran cantidad de valores y es difícil diferenciarlos en gráficos pequeños.

En la Figura 5.5 se muestra un ejemplo de “Zoom”. En este ejemplo, se ha abierto una ventana nueva con un gráfico de series para verlo más en detalle.

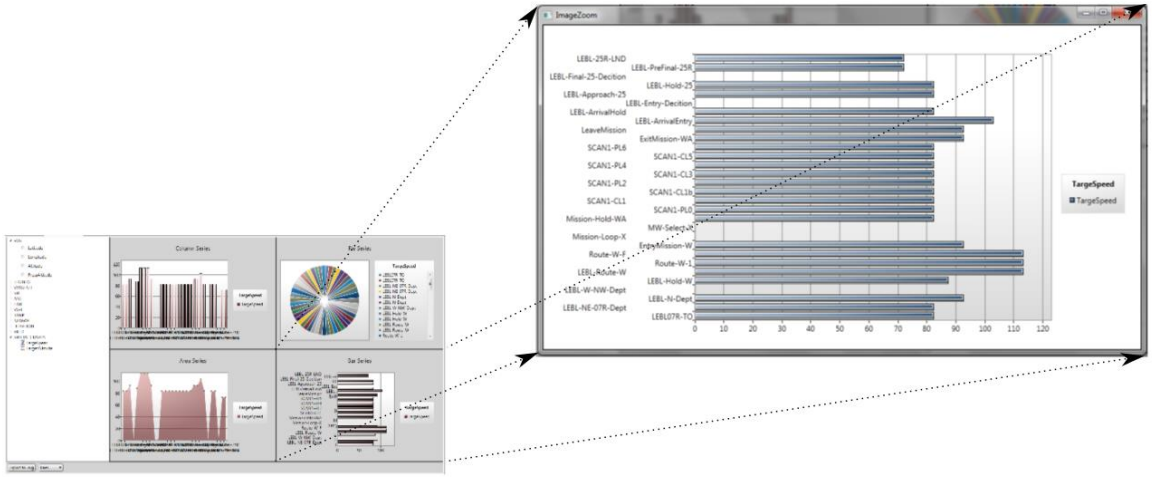


Figura 5.5 Ejemplo de Zoom

- Gráficas D3

Este control de usuario ha sido creado para realizar los gráficos lineales. Para ello se ha utilizado la librería D3 [6]. Esta librería genera un gráfico de líneas a partir de dos series de datos numéricos. Para poder utilizar dicha librería, tal y como se explica en la sección 1.3 (Planificación temporal), se tuvo que dedicar un mes a su aprendizaje. En la Figura 5.6 se muestra un gráfico lineal utilizando la librería D3.

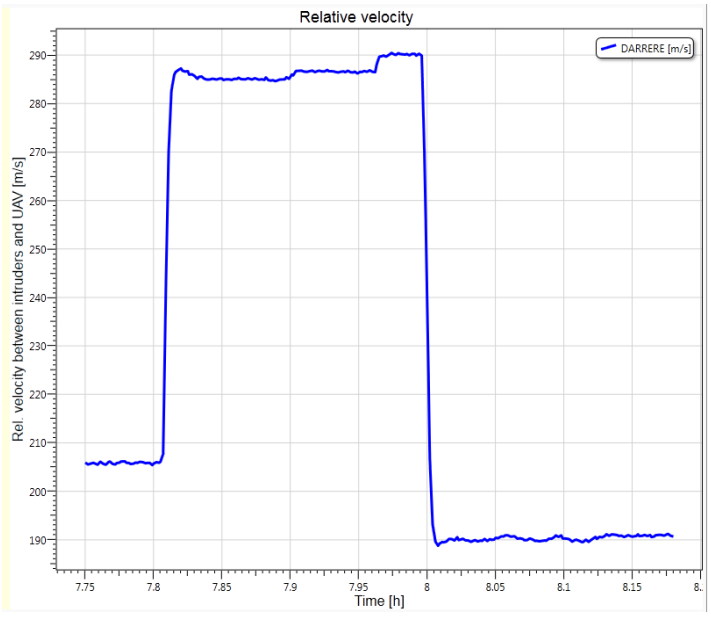
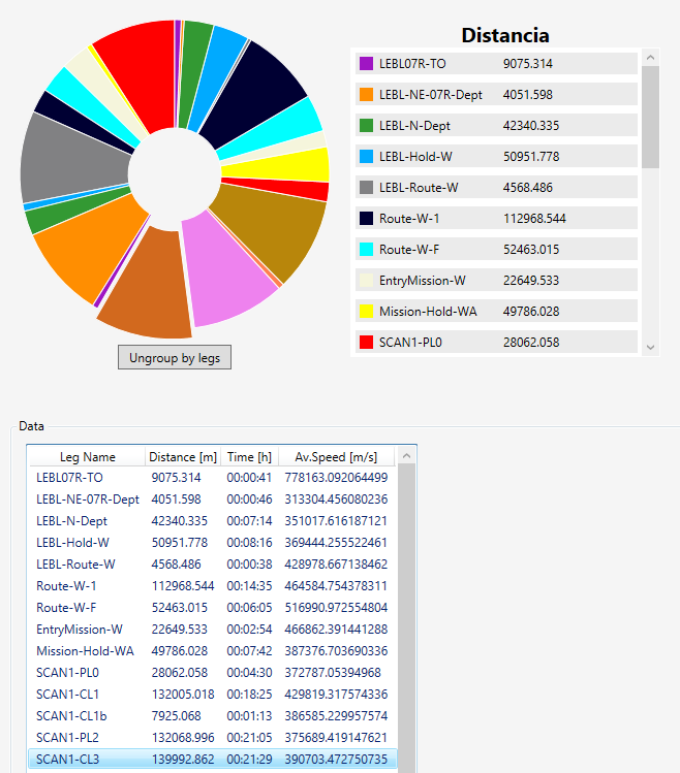


Figura 5.6 Gráficas lineales de la librería D3

- Control de plan de vuelo

Para crear este control de usuario se ha reutilizado un control de un ejemplo existente [8]. Este control contiene un gráfico circular, una leyenda y una tabla de datos que están relacionados entre sí. Esto significa que si se selecciona un valor de cualquiera de estos tres elementos, dicho valor se selecciona en los otros dos automáticamente.

Este control sacado del ejemplo [8] tenía el inconveniente de que únicamente trabajaba con dos series de datos, puesto que tenía esa relación automática entre sus elementos. Para poder añadir más series y visualizarlas al mismo tiempo, fue necesario estudiar a fondo este control de usuario y modificarlo. De esta manera, el control de plan de vuelo es una versión mejorada del control original. Se puede ver un ejemplo del control de plan de vuelo en la Figura 5.7.



Leg Name	Distance [m]	Time [h]	Av.Speed [m/s]
LEBL07R-TO	9075.314	00:00:41	778163.092064499
LEBL-NE-07R-Dept	4051.598	00:00:46	313304.456080236
LEBL-N-Dept	42340.335	00:07:14	351017.616187121
LEBL-Hold-W	50951.778	00:08:16	369444.255522461
LEBL-Route-W	4568.486	00:00:38	428978.667138462
Route-W-1	112968.544	00:14:35	464584.754378311
Route-W-F	52463.015	00:06:05	516990.972554804
EntryMission-W	22649.533	00:02:54	466862.391441288
Mission-Hold-WA	49786.028	00:07:42	387376.703690336
SCAN1-PL0	28062.058	00:04:30	372787.05394968
SCAN1-CL1	132005.018	00:18:25	429819.317574336
SCAN1-CL1b	7925.068	00:01:13	386585.229957574
SCAN1-PL2	132068.996	00:21:05	375689.419147621
SCAN1-CL3	139992.862	00:21:29	390703.472750735

Figura 5.7 Control de plan de vuelo

En esta Figura se aprecian los componentes que forman el control de plan de vuelo. En la esquina superior izquierda se encuentra el gráfico circular, a su derecha la leyenda de este gráfico, y en la parte inferior la tabla con todos los valores representados.

En la siguiente sección se explica en detalle las diferentes ventanas de análisis.

5.2 Ventanas de análisis

Las *Ventanas de análisis* son controles de usuario más avanzados, formados por los controles descritos en la sección anterior. Se han nombrado como *Ventanas de análisis* puesto que cada una de ellas tendrá la función de realizar cada uno de los análisis descritos en el capítulo 2 (Arquitectura y diseño de la aplicación). Dichos análisis son el de telemetría, el de plan de vuelo y el de tráfico aéreo. Por lo tanto existen tres *Ventanas de análisis*, esquematizadas en la Figura 5.8.

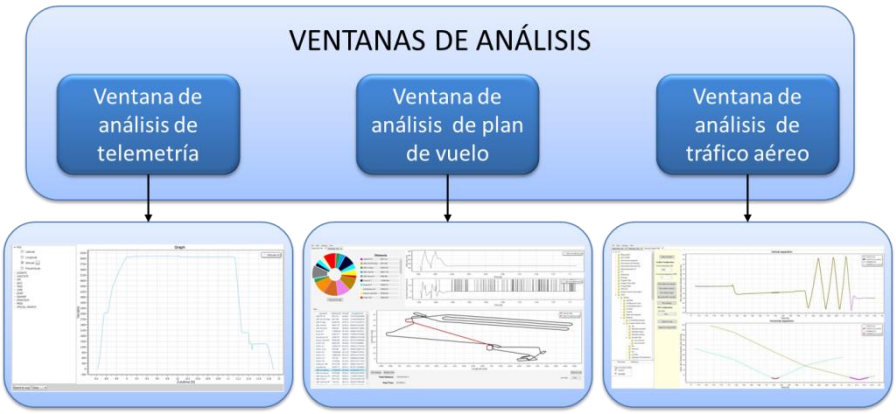


Figura 5.8 Ventanas de análisis

A continuación se explica en detalle cada una de las *Ventanas de análisis.*

5.2.1 Ventana de análisis de telemetría

Esta ventana es la encargada de hacer el análisis de la telemetría. Esta ventana está formada por una estructura en forma de árbol o “treeview” (control propio de WPF), que es rellenado con la estructura de datos. Junto a ella se encuentra el control de usuario *Gráficas D3,* que como se ha explicado en la sección anterior, es el encargado de generar gráficos lineales. De esta manera, cuando el usuario seleccione uno de los datos ubicados en el “treeview”, en la *Gráfica D3* se representará una función lineal. Esta función tendrá el dato seleccionado como valor de ordenadas, y el tiempo UTC (Universal Time Coordinated) como valor de abscisas. El tiempo UTC es el estándar por el cual se regula el tiempo en todo el mundo, motivo por el cual es el tiempo utilizado en las representaciones gráficas.

Por otro lado, esta ventana contiene otros controles de usuario que son necesarios para cumplir los requisitos del análisis de telemetría.

Figura 5.9 Arquitectura de la Ventana de análisis de telemetría

En la Figura 5.9 se puede ver la arquitectura de la *Ventana de análisis de telemetría*. Dicha ventana recibe la estructura de datos almacenada en memoria y, mediante las *Gráficas D3* y las *Gráficas de series*, representa los datos en pantalla. Además utiliza el control de usuario *Zoom* y el *Gestor de colores* para aportar funcionalidades a la ventana. El control *Zoom* permite aumentar un gráfico para verlo más en detalle, y el *Gestor de Colores* permite al usuario cambiar el grosor o el color de las líneas de las *Gráficas D3*.

Los requisitos iniciales del análisis de telemetría, detallados en el apartado 2.1.1, son los siguientes:

- Mostar mediante una gráfica de líneas los datos de telemetría y su evolución respecto al tiempo.
- Exportar las gráficas visualizadas en formato vectorial con el objetivo de facilitar su posterior edición.
- Permitir que el usuario cambie las unidades de las distintas magnitudes que se muestran en las gráficas.
- Permitir seleccionar las diferentes líneas de los gráficos en colores distintos y en aquellos colores seleccionados por el usuario, con el objetivo de aclarar la visualización de varios datos al mismo tiempo.

Gracias a las utilidades *Exportación vectoria*l y *Gestor de unidades,* explicadas en el capítulo 4, y a los controles de usuario utilizados en esta ventana, se han cumplido todos los requisitos asociados a este tipo de análisis.

Por último, en la Figura 5.10 se puede ver el estado final de la *Ventana de análisis de telemetría.*

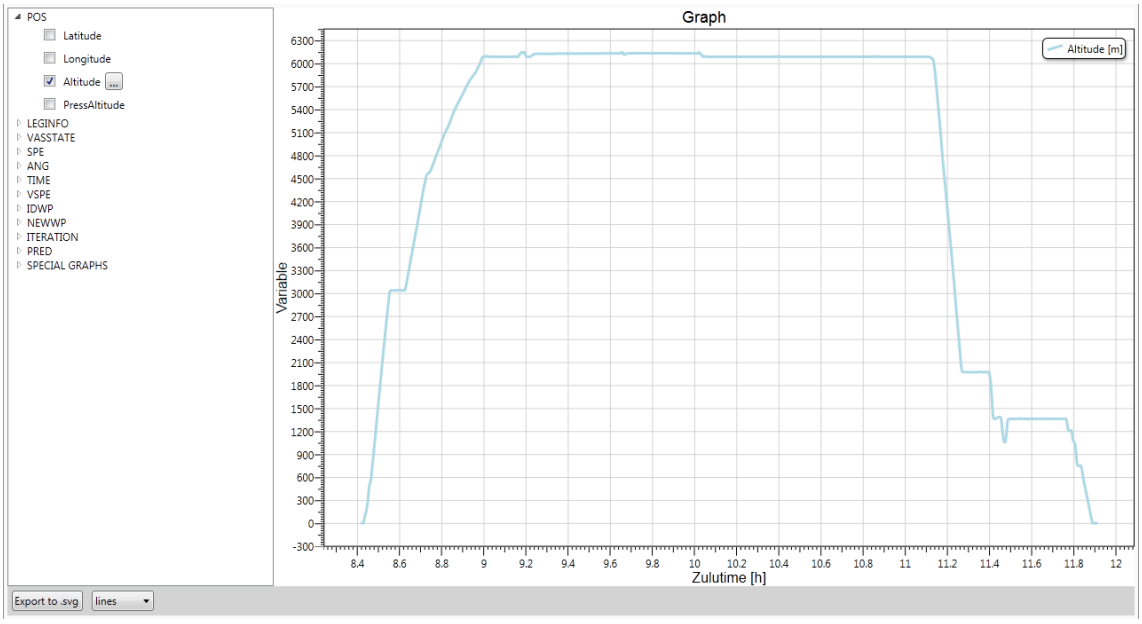


Figura 5.10 Ventana de análisis de telemetría

En esta Figurase puede ver como en el árbol se ha seleccionado la magnitud de "*Altitude*" y ésta se muestra en forma de gráfica lineal.

5.2.2 Ventana de análisis de plan de vuelo

La *Ventana de análisis de plan de vuelo* muestra el plan de vuelo por tramos, permitiendo un estudio del tiempo de misión, la distancia recorrida y la variación de altitudes y velocidades verticales en cada uno de ellos.

Figura 5.11 Arquitectura de la Ventana de análisis de plan de vuelo

En la Figura 5.11 se puede ver la arquitectura de la *Ventana de análisis de plan de vuelo*. Dicha ventana recibe la estructura de datos almacenada en memoria y, mediante las *Gráficas D3* y el *Control de plan de vuelo*, representa los datos en pantalla.

Los requisitos iniciales del análisis de plan de vuelo, detallados en el apartado 2.1.2, son los siguientes:

- Mostrar gráficamente el recorrido del UAS en un tramo concreto del vuelo.
- Mostrar el tiempo, distancia y velocidad media empleada en cada tramo recorrido durante el vuelo, así como la tasa de ascenso y descenso y los cambios de altura de cada uno de estos.
- Mostrar la proporción de tiempo y distancia empleados en ese tramo concreto respecto al vuelo completo.
- Exportar las gráficas visualizadas en formato vectorial con el objetivo de facilitar su posterior edición.
- Permitir que el usuario cambie las unidades de las distintas magnitudes que se muestran en las gráficas.

Gracias a las utilidades *Exportación vectoria*l y *Gestor de unidades,* explicadas en el capítulo 4, y a los controles de usuario utilizados en esta ventana, se han cumplido todos los requisitos asociados a este tipo de análisis.

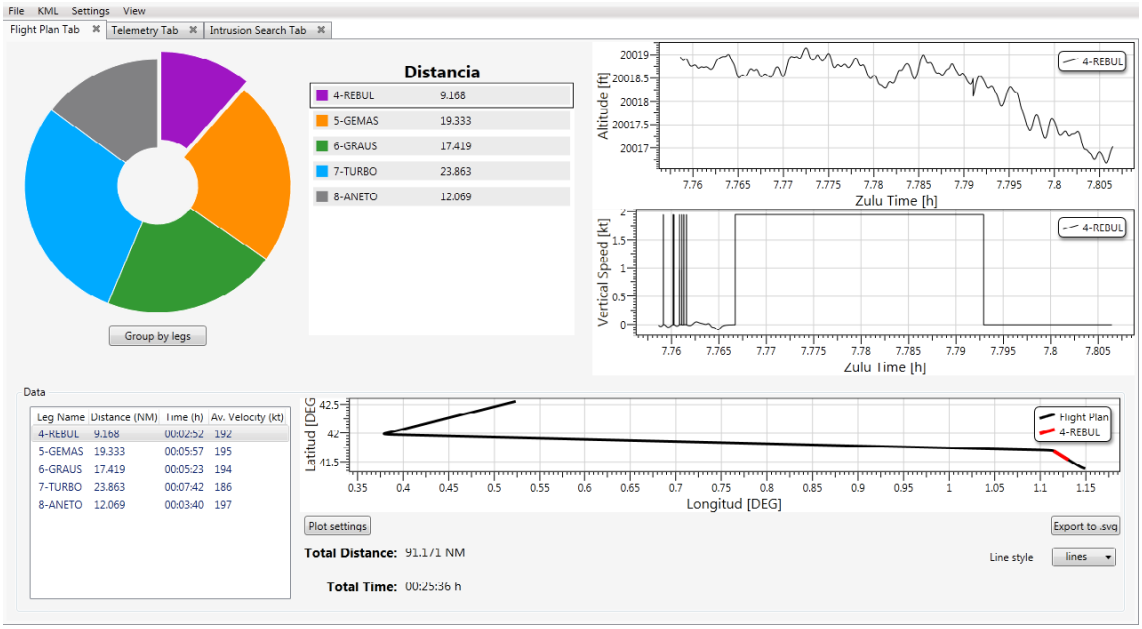


Figura 5.12 Ventana de análisis de plan de vuelo

En la Figura 5.12 se puede ver el estado final de la *Ventana de análisis de plan de vuelo*. En la parte izquierda de la Figura se encuentra el *Control de plan de vuelo* con todos sus elementos (el gráfico circular, su leyenda y la tabla de valores). En la parte derecha de la Figura se encuentran los tres *Gráficos D3*. Los dos primeros muestran la altitud y la velocidad vertical mientras que el tercero resalta, sobre el recorrido total del vuelo, el tramo seleccionado.

5.2.3 Ventana de análisis de tráfico aéreo

La *Ventana de análisis de tráfico aéreo* permite calcular distancias, velocidades y ángulos relativos entre los intrusos y el UAS, y mostrarlos gráficamente definiendo también una posible zona de conflicto.

Figura 5.13 Arquitectura de la Ventana de análisis de tráfico aéreo

En la Figura 5.13 se puede ver la arquitectura de la *Ventana de análisis de tráfico aéreo*. Dicha ventana recibe la estructura de datos del UAS y la estructura de datos de los intrusos almacenadas en memoria y, mediante las *Gráficas D3*, representa los datos en pantalla. En caso de que no exista una estructura de datos de los intrusos, mediante el *Explorador de archivos* el usuario puede seleccionar la ubicación de los ficheros de telemetría de éstos.

Los requisitos iníciales del análisis de tráfico aéreo, detallados en el apartado 2.1.3, son los siguientes:

- Cargar los ficheros de telemetría de los intrusos para poder compararlos con el del UAS, ya sea de uno en uno o seleccionando todos los intrusos desde una carpeta.
- Mostrar el recorrido de los distintos intrusos en Google Earth y así ver su trayectoria y comparar los resultados obtenidos en las gráficas. Deberán mostrarse los recorridos de cada uno de ellos con colores diferentes para que el usuario pueda diferenciarlos.
- Mostrar las distancias horizontales y verticales, así como las velocidades y ángulos relativos en su evolución con el tiempo de los distintos intrusos respecto al UAS.
- Poder resaltar los momentos de conflicto en función de una distancia de separación vertical y horizontal seleccionada por el usuario.
- Exportar las gráficas visualizadas en formato vectorial con el objetivo de facilitar su posterior edición.
- Permitir que el usuario cambie las unidades de las distintas magnitudes que se muestran en las gráficas.

Gracias a las utilidades *Exportación vectoria*l, *Gestor de unidades y Creador de ficheros KML,* explicadas en el capítulo 4, y a los controles de usuario utilizados en esta ventana, se han cumplido todos los requisitos asociados a este tipo de análisis.

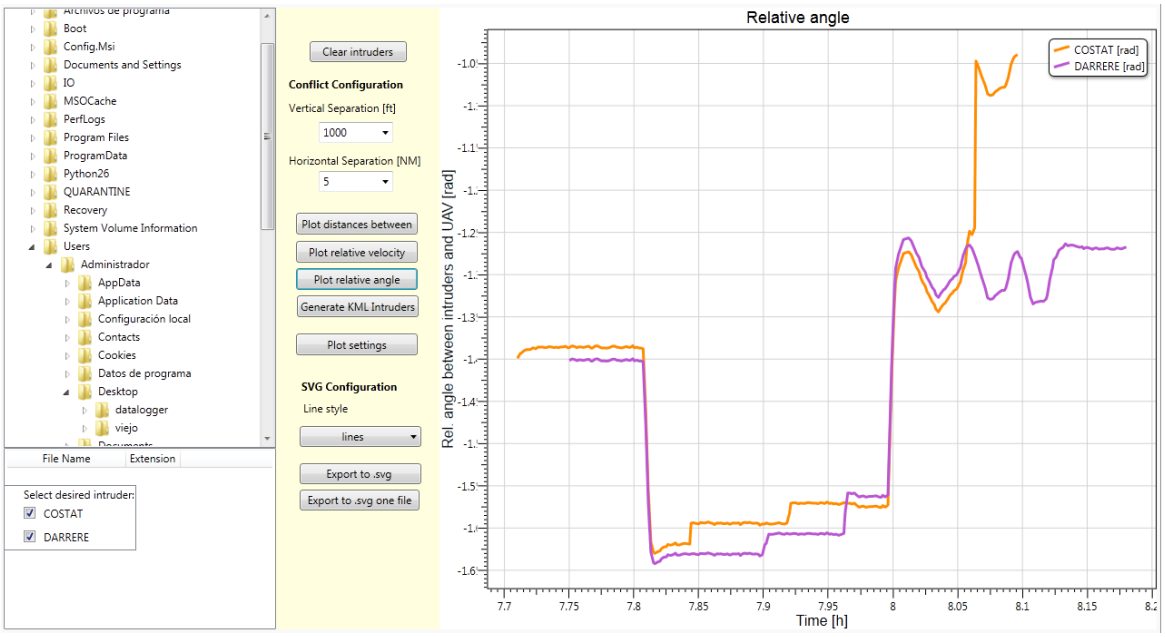


Figura 5.14 Ventana de análisis de tráfico aéreo

En la Figura 5.14 se puede ver el estado final de la *Ventana de análisis de plan de vuelo*. En ella se puede observar el ángulo relativo entre dos intrusos, llamados “COSTAT” y “DARRERE”, y el UAS que está siendo analizado.

CAPÍTULO 6. DEBRIEFER

Finalmente, todo el conjunto de los elementos descritos en los capítulos anteriores se implementa en una ventana. De este modo se obtiene la aplicación llamada Debriefer, mediante la cual se realiza el análisis de los datos. No obstante, este proyecto dispone de otra ventana, llamada Start Page. La única finalidad de esta ventana es servir como página de inicio y facilitar al usuario la entrada de datos.

6.1 Página de inicio

Al ejecutar el programa, aparece una ventana llamada *Start Page*. Esta ventana, representada en la Figura 6.1, consta de tres pestañas que proporcionan diferentes utilidades.

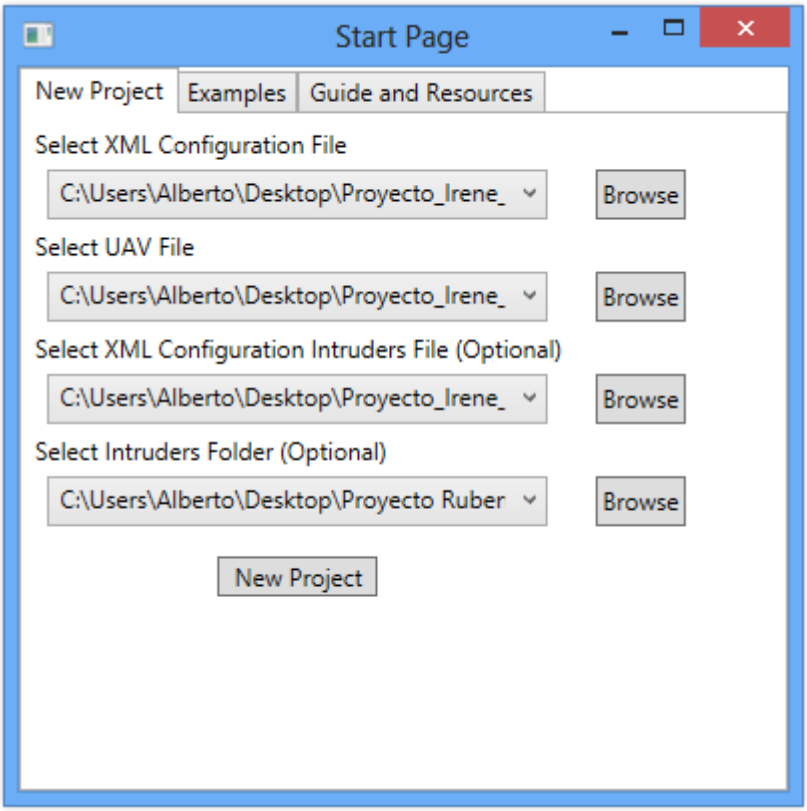


Figura 6.1 Página de inicio

En la Figura 6.1 se puede ver la primera pestaña, llamada New Project. Esta pestaña consta de cuatro menús desplegables donde se seleccionan los archivos que el usuario quiere analizar. En estos menús están guardadas las rutas de los últimos ficheros usados por el usuario. De igual modo, se pueden seleccionar los archivos a través del disco duro utilizando el botón "*Browse*".

La página de inicio cuenta también con dos utilidades muy sencillas, mostradas en la Figura 6.2.

Figura 6.2 Utilidades de la página de inicio

En esta Figura se encuentran las otras dos pestañas de la página de inicio. En la pestaña mostrada a la izquierda de la imagen, el usuario puede ver ejemplos de los diferentes tipos de análisis que hace el programa. En la segunda pestaña, situada a la derecha de la imagen, el usuario podrá ver una serie de guías y tutoriales para aprender a usar programa y referencias de otros productos del grupo ICARUS. Estas guías están en proceso de construcción.

6.2 **Menú y Ventana principal**

Esta es la ventana principal del programa, donde se realizará todo el análisis de datos. Cuenta con un menú desplegable en el cual se encuentran todas las opciones y utilidades del programa. Éstas están definidas en cuatro grupos:

- File

En este apartado, tal y como se puede apreciar en la Figura 6.3, se encuentra todo lo relacionado con los ficheros. En este menú desplegable se da la opción de crear un nuevo proyecto desde cero, cambiar un archivo (el de configuración o el de datos) abrir nuevas pestañas y cerrar el programa.

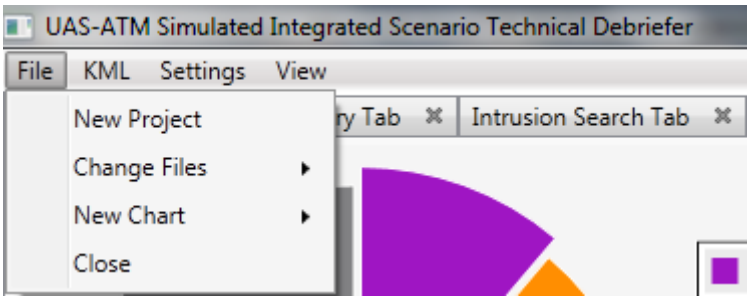


Figura 6.3 Menú desplegable File

- KML

En este menú desplegable se hallan las operaciones que define el usuario, explicadas en el apartado 3.3, relacionadas con la exportación de ficheros con extensión KML, los ficheros utilizados en el *Google Earth*.

Como se aprecia en el ejemplo de la Figura 6.4, en este caso en particular el usuario ha definido dos operaciones de este tipo: *Generate KML Static* y *Generate KML Dynamic*. Por lo tanto, la aplicación muestra estas operaciones en pantalla.

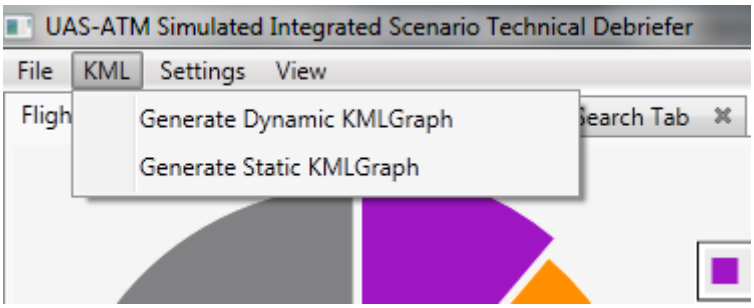


Figura 6.4 Menú desplegable KML

- Settings

En esta parte del menú se encuentran las opciones o preferencias que el usuario puede modificar. Estas nos permiten cambiar el número de puntos que el usuario desea graficar y, en el caso que se permita, cambiar las unidades en las que se muestran esos datos.

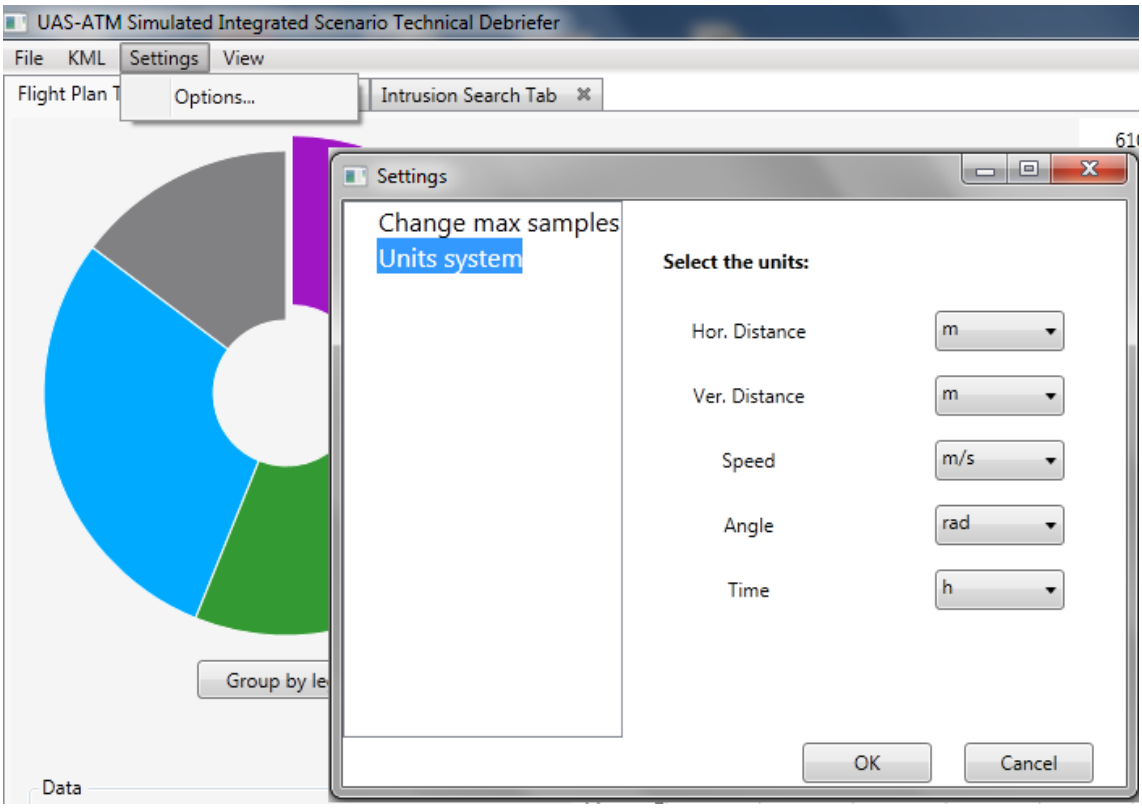


Figura 6.5 Menú desplegable Settings

Como se puede observar en la Figura 6.5, las opciones del menú abren una nueva ventana donde nos da la opción de cambiar el sistema de unidades o cambiar el número máximo de muestras.

- View

En este menú desplegable, ejemplificado en la Figura 6.6, el usuario puede cambiar el tamaño de la ventana del programa, pasando de modo ventana a pantalla completa.

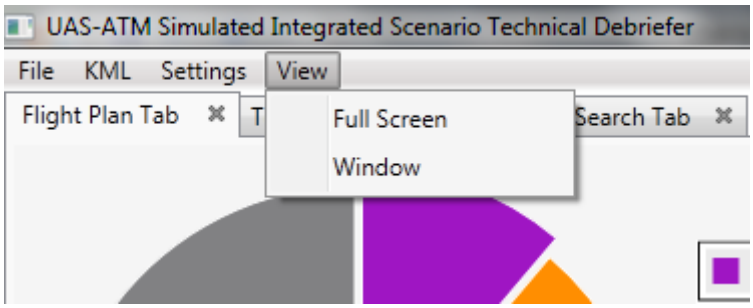


Figura 6.6 Menú desplegable View

Finalmente, la ventana principal del programa es únicamente un contenedor dentro del cual el usuario puede ir añadiendo pestañas tantas veces como desee. Estas pestañas son añadidas por la utilidad *Creador de pestañas*, explicada en el capítulo 4 (*Utilidades*). De este modo el usuario puede abrir un tipo de pestaña más de una vez para poder comparar resultados. Existen tres tipos diferentes de pestañas, cada una de las cuales contiene una V*entana de análisis* (la de telemetría, la de plan de vuelo y la de tráfico aéreo).

En la Figura 6.7 se puede ver el resultado final de la aplicación, donde están implementados todos los elementos descritos en esta memoria. En dicha Figura se puede observar la ventana principal con la pestaña de plan de vuelo seleccionada.

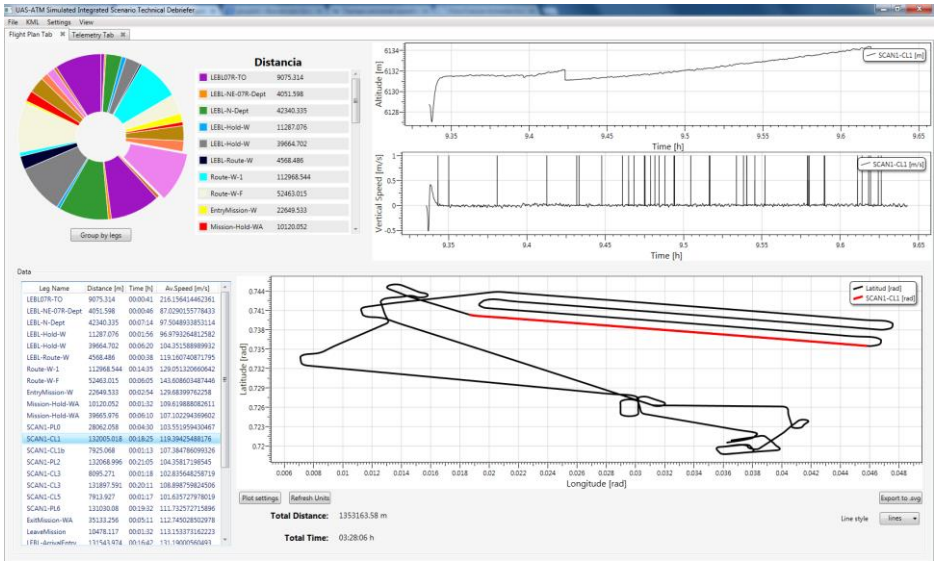


Figura 6.7 Ventana principal

CAPÍTULO 7. CONCLUSIONES Y TRABAJO FUTURO

7.1 Conclusiones

Una vez terminado el proyecto es necesario comprobar si los objetivos han sido conseguidos. Para ello se procede a compararlos con los resultados finales.

- *La aplicación ha de ser capaz de realizar el análisis de los parámetros contenidos en un fichero de datos*.

Este requisito ha sido cumplido y mejorado. En un principio el grupo ICARUS deseaba que el programa fuera capaz de leer un fichero de telemetría específico generado por ellos y analizarlo. En cambio, el programa es capaz de leer cualquier tipo de fichero, sea de la plataforma que sea, gracias a los ficheros de configuración.

- *La aplicación debe ser a medida del simulador de UAS, analizando los datos relacionados con el plan de vuelo y tráfico aéreo.*

El programa es capaz de discernir si el fichero que está leyendo tiene la estructura y los datos que genera el simulador del grupo ICARUS. Además, el programa dispone de dos pestañas, es decir, dos tipos de análisis diferentes hechos a medida. En esas pestañas se puede analizar el plan de vuelo realizado por el simulador y los posibles conflictos con intrusos. De nuevo, se ha logrado el objetivo deseado.

Finalmente, se ha obtenido una herramienta con un gran rendimiento en cuanto a velocidad de carga de datos, versatilidad en su proceso y en definitiva, utilidad. Además, cumple con todas las funcionalidades deseadas para cada una de las pestañas y para cada tipo de análisis de datos proporcionando unos gráficos o ficheros de salida interesantes que posteriormente pueden ser analizados.

En nuestro caso particular como autores del proyecto, estamos muy orgullosos de haber formado parte, como proyectistas, durante este tiempo del grupo ICARUS. Hemos logrado desarrollar una aplicación que puede ser de gran ayuda al grupo y hemos logrado dejarla en una versión estable y funcional.

También estamos muy contentos de haber tenido la oportunidad de aprender a utilizar lo que para nosotros son nuevas herramientas de desarrollo de software como es el caso C# y WPF. Creemos que ha sido una experiencia muy gratificante puesto que hemos seguido todos los pasos de creación de una aplicación desde cero.

7.2 **Trabajo futuro**

La aplicación ahora es capaz de leer un archivo de telemetría y mostrar los datos de este archivo mediante gráficas o ficheros de salida. Creemos que el futuro desarrollo de esta aplicación puede enfocarse al análisis a fondo de esos datos mostrados en el programa.

Una funcionalidad interesante sería que el programa fuera capaz, no solo de mostrar estos datos, sino analizarlos y determinar cuál ha sido la causa de un accidente o de una pérdida de sustentación. Analizar posibles diferencias entre órdenes transmitidas mediante plan de vuelo y reacción de los actuadores. Las distintas relaciones existentes en la evolución de la transmisión y recepción de paquetes, etc... En definitiva, crear una herramienta no solo de lectura sino de análisis y detección de posibles fallos y errores.

7.3 **Impacto medioambiental**

Respecto al impacto medioambiental, se ha considerado que debido a que el proyecto consiste en la implementación de software, no provoca impacto medioambiental alguno. No obstante, se han seguido las indicaciones de la maqueta elaborada por la *Escola d'Enginyeria de Telecomunicació i Aeroespacial de Castelldefels*, por la cual la presente memoria ha de estar impresa a doble cara minimizando este impacto.

BIBLIOGRAFIA

[1] Daniel M. Solis. *Illustrated WPF,* Apress. 2009

[2] Introducción a WPF
“http://msdn.microsoft.com/es-es/library/ms752299.aspx”

[3] Tutorial WPF
“http://www.wpftutorial.net/Home.html”

[4] Tutorial WPF
“http://www.codeproject.com/Articles/140611/WPF-Tutorial-Beginning”

[5] Material de la asignatura Informática 2, del grado de Ingeniería de aeronavegación e ingeniería de aeropuertos, de la Escuela de Ingeniería de Telecomunicación y Aeroespacial de Castelldefels, curso 2012-2013

[6] Dynamic Data Display Source Code and libraries
“https://dynamicdatadisplay.codeplex.com/”

[7] Tutorial Dynamic Data Display
“http://research.microsoft.com/en-us/um/cambridge/groups/science/tools/d3/dynamicdatadisplay.htm”

[8] Pie Chart
“http://www.codeproject.com/Articles/28098/A-WPF-Pie-Chart-with-Data-Binding-Support”

[9]CloseableTab
“http://geekswithblogs.net/kobush/archive/2007/04/08/closeabletabitem.aspx”

[10] File Explorer
“http://joshsmithonwpf.wordpress.com/2007/11/09/reaction-to-a-simple-wpf-explorer-tree/”

[11] Cálculo de distancias
“http://www.movable-type.co.uk/scripts/latlong.html”

[12] WPF Toolkit Charting Controls Samples
“http://www.codeproject.com/Articles/196502/WPF-Toolkit-Charting-Controls-Line-Bar-Area-Pie-Co”

[13] KMLWriter.cs
Proporcionado por el grupo de investigación ICARUS

[14] Web oficial de UAVNAVIGATION
http://uavnavigation.com

[15] Point.cs
Proporcionado por el grupo de investigación ICARUS

[16] Units.cs
Proporcionado por el grupo de investigación ICARUS

[17] Vuelo ICARUS Helicóptero
https://www.youtube.com/watch?v=-2H-Fsyddw8

[18] Vuelo Simulador ISIS
https://www.youtube.com/watch?v=W1JjwahAqBc

[19] WPFToolkit
http://wpf.codeplex.com/releases/view/40535

[20] GNUPlot
www.gnuplot.info

[21] UAV Sierra de NASA
http://airbornescience.nasa.gov/aircraft/SIERRA

ANEXO: Indicaciones técnicas

Para exportar las gráficas visualizadas en la aplicación es necesario:

- Tener instalado el programa GnuPlot [20] (http://www.gnuplot.info/download.html) en la misma máquina que se ejecuta el programa, teniendo como extensión asociada a dicho programa la ".gp".
- En la configuración regional y de idioma del ordenador donde se ejecuta, hay que configurar que el símbolo decimal utilizado es ".".

Para visualizar el fichero ".kml" en Google Earth:

- Es necesario tener instalado el programa Google Earth (http://www.google.es/intl/es/earth/index.html) y que dicha extensión esté asociada al mismo.

Indicaciones para la pestaña de plan de vuelo:

Las variables han de estar definidas en el fichero de configuración como están descritas a continuación para poder realizar un correcto análisis del plan de vuelo:

- Ha de existir el campo *POS* y dentro del él han de estar los datos *Latitude*, *Longitude*, *Altitude* para las distintas posiciones.
- Dentro del campo *SPE* ha de estar el dato *VerticalSpeed*.
- En el campo donde se encuentren los nombres de los tramos del plan de vuelo se debe añadir un "flag" o atributo para que el software lo reconozca.

Este "flag" es el siguiente:

```
<FlightPlan>true</FlightPlan>
```

- En el fichero de configuración se debe definir una operación de tipo PIE como ilustra el ejemplo siguiente:

```
<PIE>
<Dato>POS.Latitude</Dato>
<Dato>POS.Longitude</Dato>
<Dato>POS.Altitude</Dato>
<Dato>POS.Time</Dato>
<Name>Generate PieChart</Name>
<Function>generarPieChart</Function>
</PIE>
```

Indicaciones para la pestaña de intrusos:

- Ha de existir el campo *POS* y han de estar los datos *Latitude*, *Longitude*, *Altitude* para las distintas posiciones en todos los archivos tanto de intrusos como del UAS.
- Para el UAS ha de existir el campo *SPE* y dentro del él ha de estar el dato TrueSpeed (en modulo). Además, dentro del campo *ANG* ha de estar el dato *Yaw.*
- Para los intrusos ha de existir el campo *SPE* y dentro de él los datos *GroundSpeed_East* y *GroundSpeed_North* siendo así sus velocidades separadas en dos componentes.
- Las unidades de longitud y latitud de los intrusos y el UAS en él ".txt" han de estar en radianes [rad] para poder realizar las operaciones de cálculo de distancias. Además, es necesario que los tiempos, las velocidades y las altitudes estén en las mismas magnitudes en los diferentes archivos, ya sean horas o minutos, knots o metros/segundo y/o metros o millas náuticas. Posteriormente en los datos mostrados se puede seleccionar la unidad deseada.
- En los intrusos es necesario que las velocidades estén en componente este y componente norte. Para el UAS las velocidades han de estar en modulo y ángulo, siendo el ángulo en radianes.