CAPÍTULO

3 La Aplicación

3.1 INTRODUCCION

Una vez fijados unos conocimientos básicos sobre el funcionamiento de los GPS, podemos abordar el proyecto en sí. Este capítulo lo dedicaremos en su totalidad a explicar el funcionamiento de nuestra aplicación.

En el primer punto volveremos a tratar los antecedente de nuestra aplicación y dedicaremos un poco más de atención a cada uno de estos programas. El resto de apartados se centrarán en un determinado punto de InterGPS y nos describirán su funcionamiento interno, su estructura software, las tramas de comunicación que han sido necesarias para su implementación y posibles problemas o particularidades que hayan presentado en su realización. Por último, se verá el funcionamiento de la aplicación con equipos diferentes a los Novatel OEM4 RT20 para los que fue diseñada.

3.2 ANTECEDENTES

Como ya dijimos en la introducción de esta memoria, antes de iniciar el proyecto se contaba con una serie de programas orientados a trabajar con estos terminales. En el presente apartado estudiaremos con mayor profundidad el funcionamiento de esto software y veremos tanto las carencias como las virtudes que presentan cada uno de ellos.

3.2.1 El programa WinSat

WinSat se presenta como una interfaz gráfica diseñada específicamente para los receptores de Novatel, con el fin de poder hacer uso y explotar la gran mayoría de las capacidades de la tarjeta receptora.

Podemos dividir las funciones de dicho Software en cinco grupos:

- Comunicaciones a bajo nivel por su puerto serie.
- Comunicaciones a alto nivel a través del puerto serie.
- Procesamiento de datos.
- Monitorización de los datos.
- Almacenamiento de los datos.

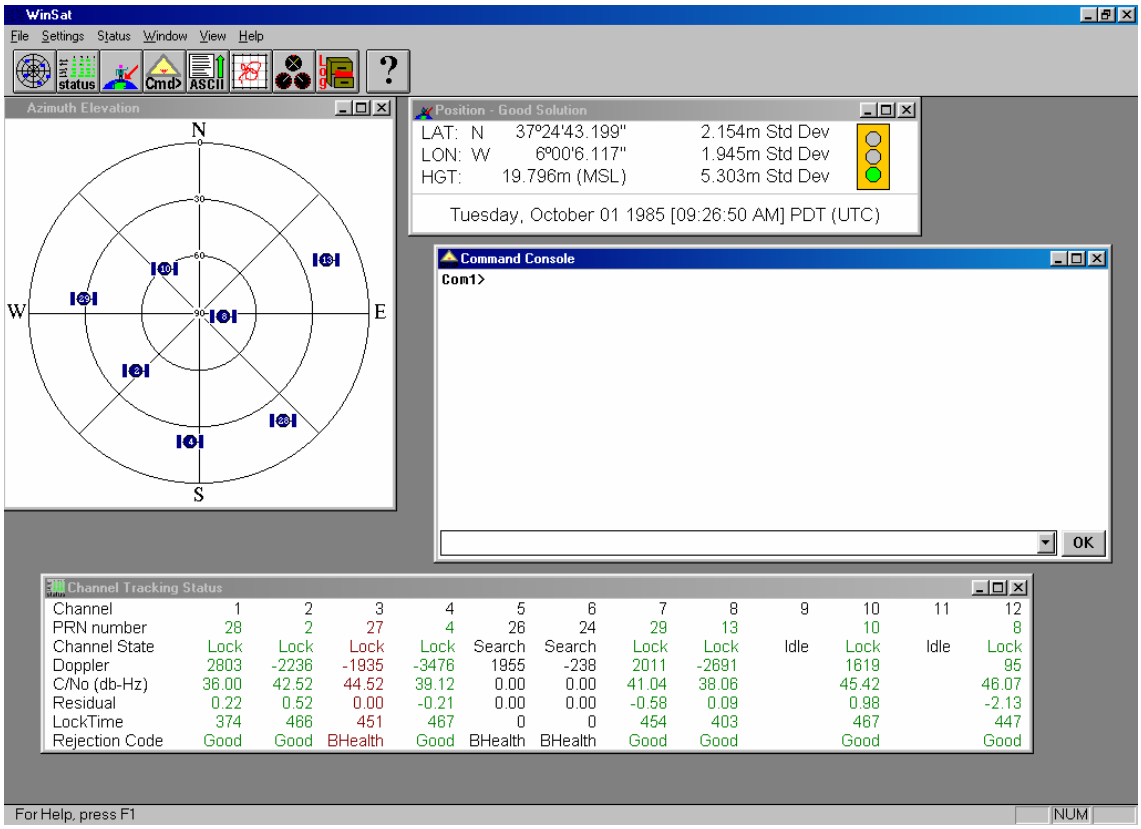

Imagen 3.1 : El programa WinSat

3.2.1.1 Comunicaciones a bajo nivel por puerto serie

Sobre este aspecto el WinSat se limita a automatizar la conexión del puerto serie, permitiendo configurar los parámetros de dicha conexión: puerto, velocidad, paridad, numero de bits por dato... Al inicio de la ejecución del programa, WinSat rastrea las posibilidades más comunes de comunicación con el GPS, de modo que ni siquiera sea necesario para el usuario tener que preocuparse por conocer los parámetros de la conexión.

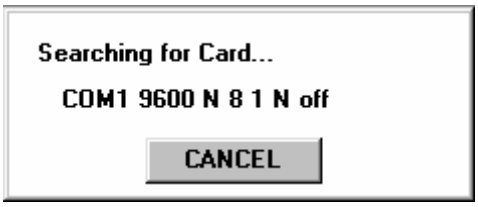

Imagen 3.2 : Búsqueda secuencial de parámetros de comunicación

3.2.1.2 Comunicaciones de alto nivel a través del puerto serie

Sobre los servicios proporcionados por la capa de comunicaciones inferior, WinSat implementa una nueva capa de comunicaciones encargada de recibir y enviar tramas siguiendo un protocolo concreto. Esta nueva capa es totalmente transparente para el usuario, ya que en ningún momento se hace necesario ningún tipo de comunicación. A esta segunda capa de comunicaciones, WinSat incorpora las pertinentes funciones de control de errores y de flujo, de manera que sea posible hacer un seguimiento del estado de las comunicaciones.

3.2.1.3 Procesamiento de los datos

Éste es sin duda el punto débil de la aplicación, ya que la mayoría de los datos que WinSat presenta al usuario son proporcionados directamente por el receptor de GPS, pudiendo hacer una excepción sólo con los datos de la velocidad y dirección del movimiento de la antena. El procesamiento que sufren el resto de datos está principalmente orientado a adecuarlos a una representación gráfica más eficiente para el usuario.

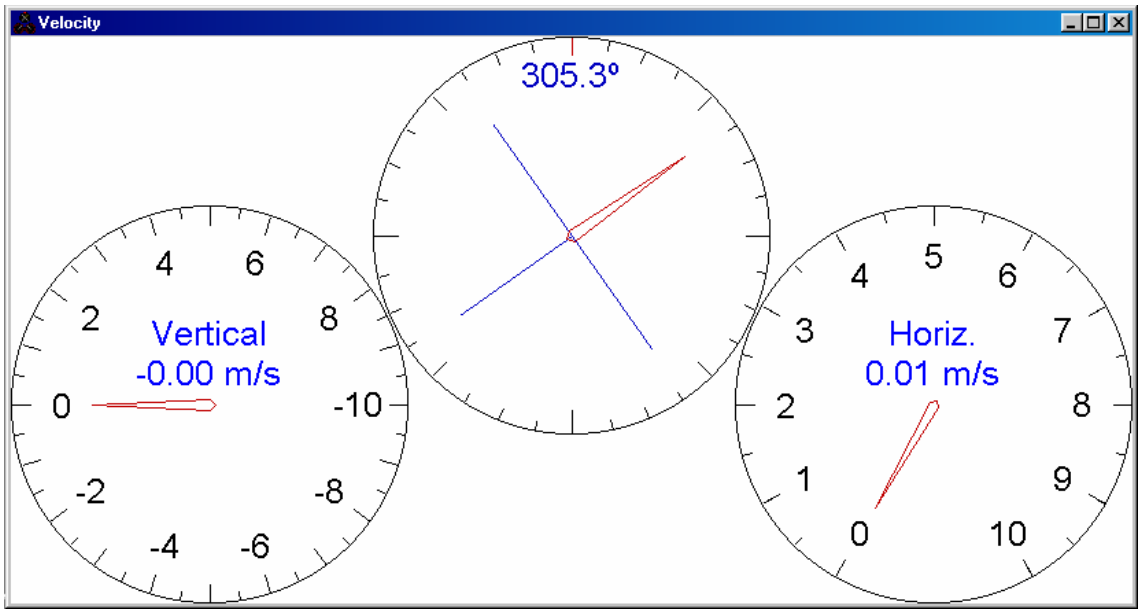

Imagen 3.3 : Velocidad y dirección del movimiento

3.2.1.4 Monitorización de los datos

Al contrario que el punto anterior, este punto sí está ampliamente explotado por la aplicación del fabricante. WinSat permite obtener, mediante multitud de indicadores visuales, la práctica totalidad de los datos que han sido procesados por el receptor de GPS, a partir de los distintos satélites que tenga a la vista en ese instante.

Por ejemplo, es capaz de mostrar información detallada sobre cada uno de los doce canales de que dispone para capturar la señal de algún satélite, dando como dato lo que usualmente se conoce como “salud” del canal, para indicar la calidad de la señal recibida. También aporta el desvío de frecuencia de cada satélite ocasionado por el efecto Doppler y otros datos referentes a las características del enlace Receptor-Satélite.

Channel Tracking Status

Channel	1	2	3	4	5	6	7	8	9	10	11	12
PRN number	28	2	27	4	26	24	29	13		10		8
Channel State	Lock	Lock	Lock	Lock	Search	Search	Lock	Lock	Idle	Lock	Idle	Lock
Doppler	2803	-2236	-1935	-3476	1955	-238	2011	-2691		1619		95
C/No (db-Hz)	36.00	42.52	44.52	39.12	0.00	0.00	41.04	38.06		45.42		46.07
Residual	0.22	0.52	0.00	-0.21	0.00	0.00	-0.58	0.09		0.98		-2.13
LockTime	374	466	451	467	0	0	454	403		467		447
Rejection Code	Good	Good	BHealth	Good	BHealth	BHealth	Good	Good		Good		Good

Imagen 3.4 : Estado de las comunicaciones con los satélites

Es capaz también de mostrar la posición actual de la antena receptora, de dos modos distintos: mediante dígitos o mediante un gráfico, en el que se puede observar, por ejemplo, la desviación que sufren medidas tomadas en las mismas condiciones y en instantes muy cercanos.

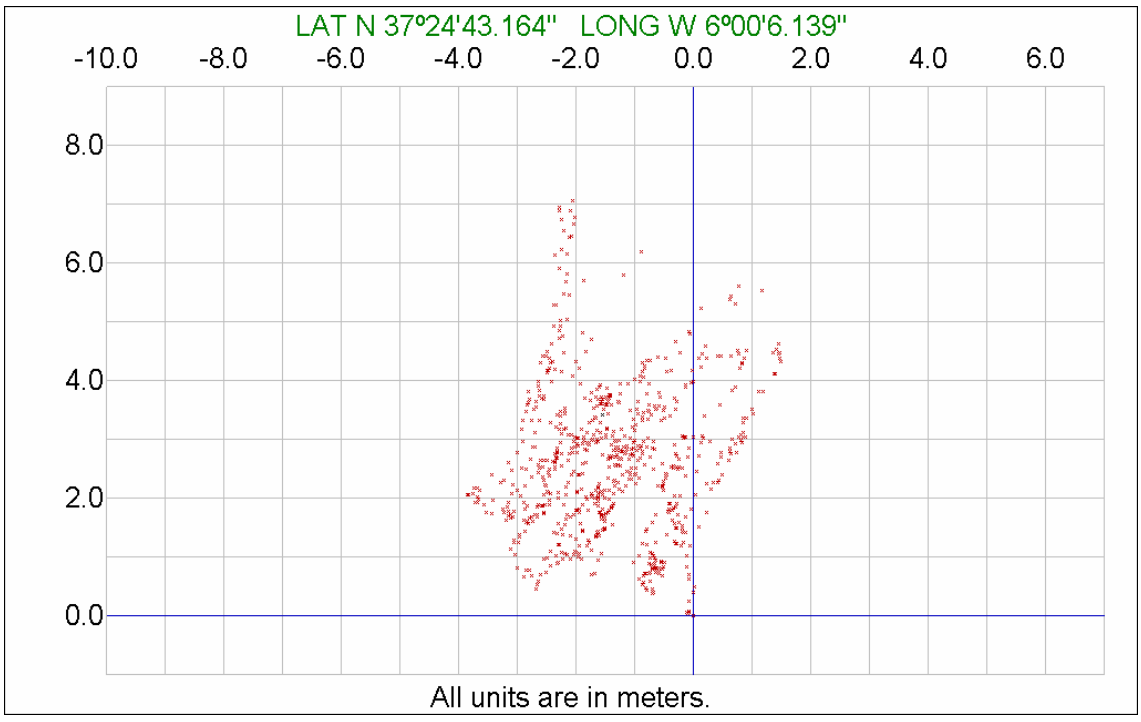

Imagen 3.5 : Histórico de posiciones calculadas.

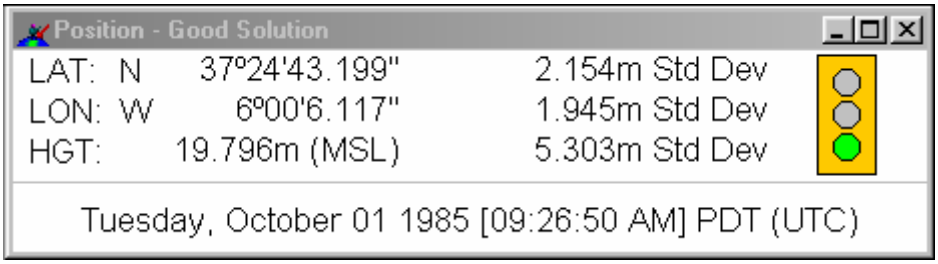

Imagen 3.6 : Posición actual y desviación estándar.

Y por último podemos destacar también la posibilidad de ver la disposición espacial de los satélites que se tienen "a la vista", en un plano que identifica cada satélite por su PRN, que no es más que un número asociado a cada satélite a modo de identificador.

3.2.1.5 Almacenamiento de los datos

El fabricante ha diseñado su aplicación para que pueda salvar en un archivo de texto todas las tramas recibidas y enviadas a través de WinSat, de forma que posteriormente puedan ser exportadas a otra aplicación. Además de esto, junto con WinSat (su aplicación principal) adjunta otra aplicación denominada File Converter que es capaz de transformar estos archivos almacenados a varios formatos como pueden ser ASCII, binario, comprimidos, RINEX.

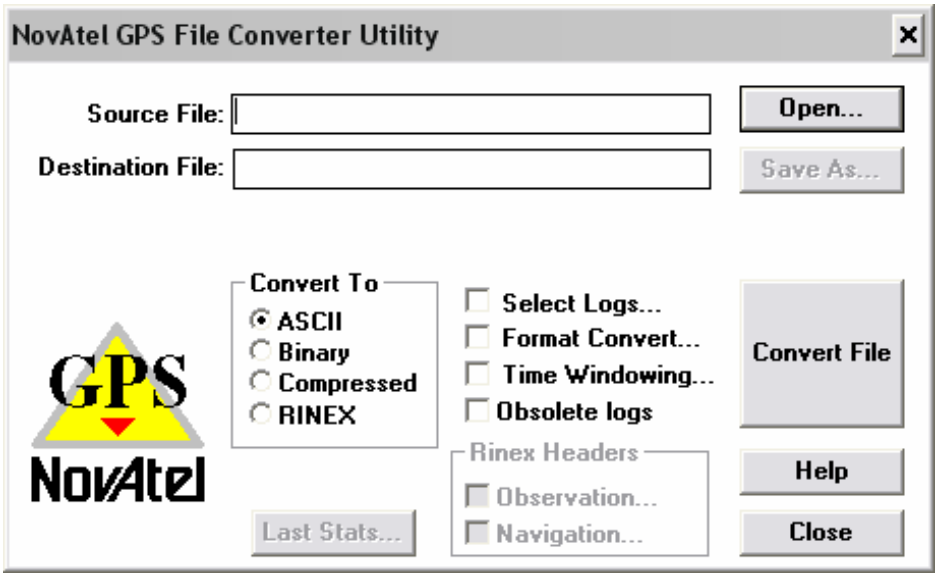

Imagen 3.7 : Utilidad de conversión de ficheros.

Como hemos comentado antes, con WinSat el PC se usa como una simple consola, en la que presentan los datos recibidos por el GPS, añadiendo funciones sencillas que permiten configurar el receptor, y almacenar los datos. Quizá éste sea el motivo más

importante por el cuál se hace necesario el diseño de otra aplicación que, además de tener todas las capacidades de WinSat, permita un mejor aprovechamiento de las ventajas que nos proporciona el usar un PC conjuntamente con el receptor.

A modo de resumen, podríamos decir que WinSat se diseñó como una aplicación de capacidades básicas, destinada únicamente a hacer una muestra de las funciones que incorpora el receptor.

3.2.2 El programa Sateyo

En el apartado anterior, hemos citado ciertas carencias que presenta la aplicación WinSat. Con el objetivo de cubrir estas necesidades, se desarrolló un software que permitía un mejor aprovechamiento de las ventajas proporcionadas por el uso conjunto del receptor y del PC. Este programa ha sido citado anteriormente, y no es otro que Sateyo.

Como ya hemos visto, el problema que presentaba Sateyo, era que había sido creado para trabajar con los receptores Novatel RT-20, mientras que en el presente proyecto, se estaba trabajando con equipos Novatel RT-2.

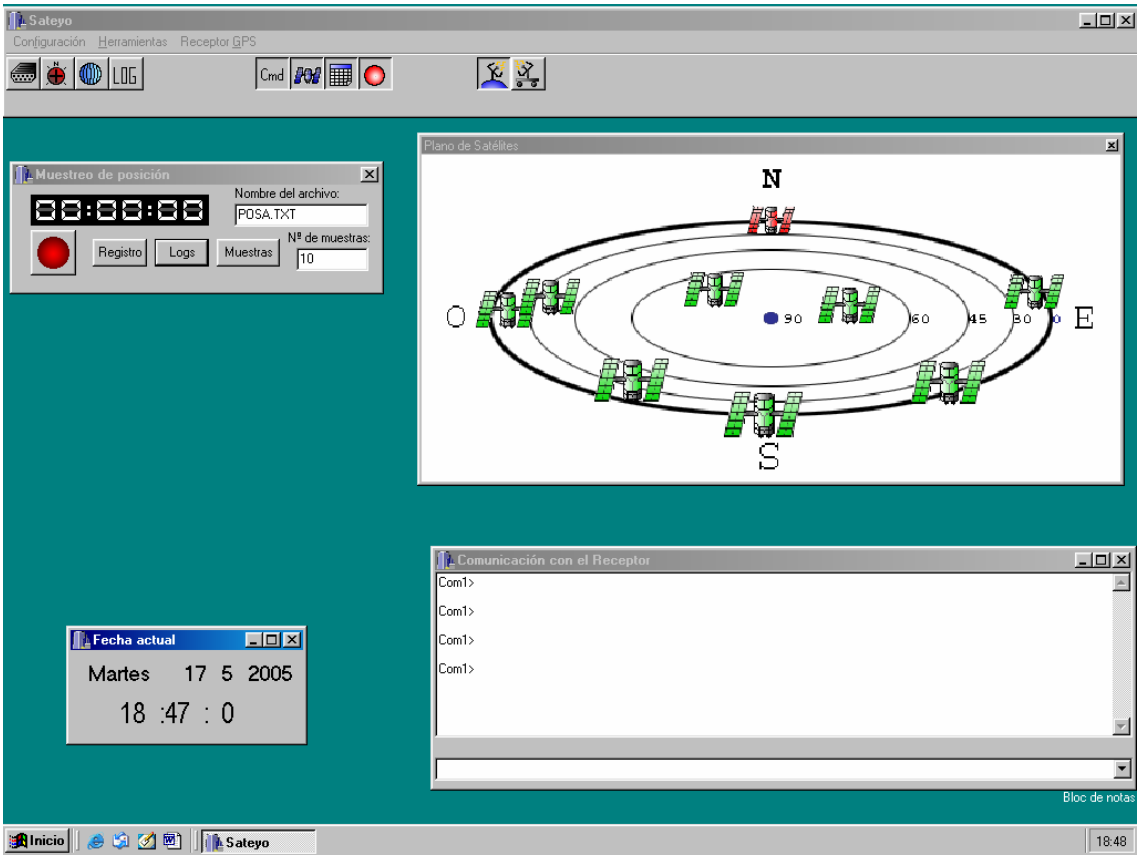

Imagen 3.8 : Aspecto de la aplicación Sateyo

Muchas de las opciones que proporciona Sateyo ya estaban presentes en WinSat, de ahí que nos centremos con especial interés en aquellas que no lo estaban, como pueden ser la configuración del equipo en modo base o remoto (para un funcionamiento en modo diferencial), la captura de datos durante el muestreo de la posición y la posibilidad de solicitar periódicamente al receptor ciertos datos como la posición de los satélites, la hora y la posición de la antena.

3.2.2.1 Receptor GPS

Esta opción sólo estará disponible si se ha conseguido establecer una comunicación válida con el receptor de GPS, de no ser así, la opción aparece clareada en el menú y será imposible ejecutarla.

Lo que ocurra al pulsar esta opción depende, de si estamos tratando con el receptor base o el receptor remoto. En cualquiera de los dos casos el cometido de esta función es poder configurar al receptor dentro del papel que se haya elegido para él, ya sea remoto o base.

En el caso en que estemos tratando con el receptor base, la ventana que se muestra es la siguiente:

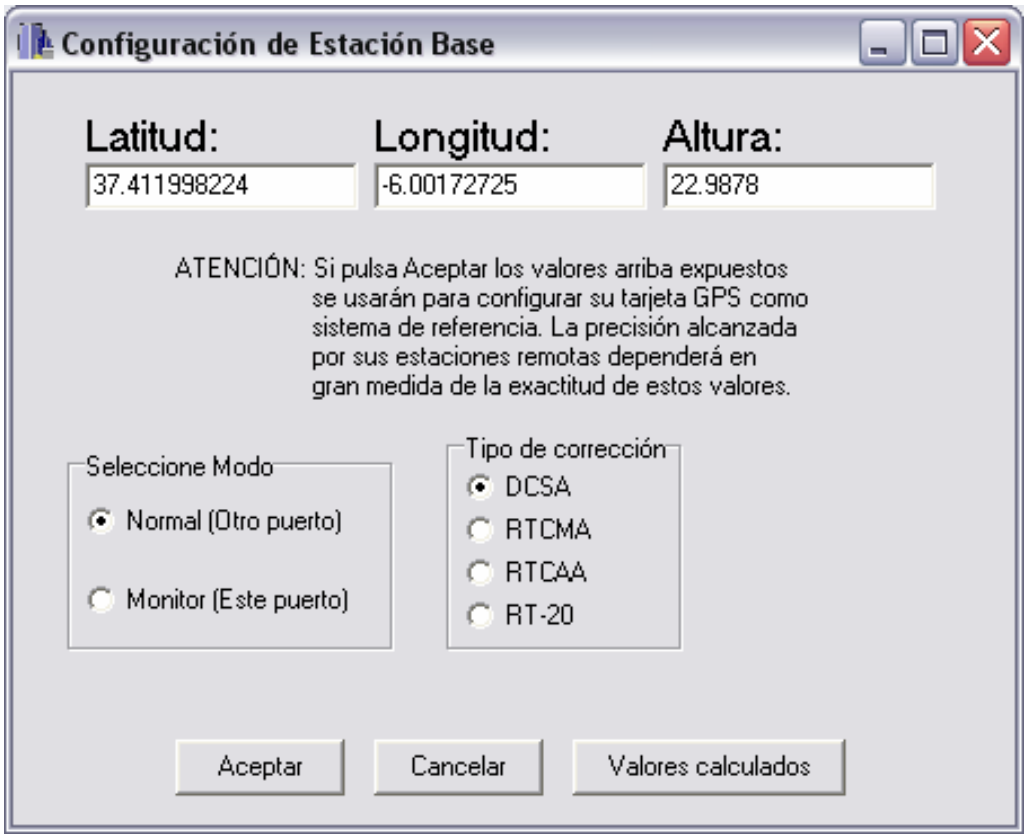

Imagen 3.9 : Configuración de la estación base.

Una vez elegidas todas las opciones, basta con pulsar sobre aceptar para que el receptor comience a comportarse como un receptor base y empiece a emitir correcciones diferenciales por el puerto seleccionado.

Por orden cronológico, debe siempre configurarse el receptor base antes que el remoto. Ya que las opciones que se elijan para configurar el receptor base serán las que luego se usen para configurar el receptor remoto sin dar opción al usuario a cambiarlas, salvo que quiera cambiar la configuración del receptor base.

En el caso de estar tratando con el receptor remoto, la ventana que se muestra es la siguiente, y como ya se ha dicho no tiene ninguna opción, basta con pulsar sobre el botón aceptar para que automáticamente el receptor de GPS, conectado al puerto serie, comience a comportase como un receptor de GPS remoto.

Imagen 3.10: Configuración del receptor remoto.

Los receptores usados, para base y remoto pueden ser exactamente el mismo modelo, de modo que la única forma que la aplicación tiene para diferenciar cuál de los dos receptores tiene conectado, es que el usuario se lo indique.

3.2.2.2 Muestreo de la posición

Este es el que permite capturar tramas y realizar los cálculos pertinentes para poder configurar los receptores de GPS.

Su aspecto es el siguiente inicialmente:

Imagen 3.11 : Módulo de muestreo.

Al pulsar sobre el botón Registro, se ofrece al usuario la opción de seleccionar qué datos quiere registrar en el archivo. Para poder elegir los datos aparece la siguiente ventana:

Imagen 3.12 : Selección de datos a registrar.

Las casillas que se encuentren marcadas serán los datos que quedarán registrados en el archivo de datos de posición (POSA.txt por defecto). Una opción especial es la de "Datos completos (sat.txt)", si se marca esta opción además del archivo de datos de posición se creará otro archivo distinto llamado sat.txt en el que se incluirán todos los datos referentes a todos los satélites captados en cada momento.

Para dar paso a la ejecución del muestreo no hay más que pulsar sobre el botón redondo y rojo que simboliza la acción de grabar.

Imagen 3.13 : Muestreando.

Una vez finalizado el proceso de muestro, automáticamente se pulsará el botón de parada y se mostrará la ventana de configuración de GPS, la que hemos tratado en el apartado anterior, con los valores calculados para la posición en los lugares correspondientes a la Latitud, Longitud y Altura.

3.2.2.3 Captura de datos

Además de lo ya mencionado, es posible capturar periódicamente medidas de la posición del equipo, la posición de los satélites y la hora. Esta opción no es más que solicitar al GPS una serie de comandos LOG (consultar el Anexo C) mediante los cuales el equipo nos devolverá periódicamente la información solicitada.

Imagen 3.14 : Selección de Logs

3.2.3 HyperTerminal de Windows

Este programa, como el lector sabrá, no ha sido creado explícitamente para su funcionamiento con equipos de GPS, aun así y gracias a su sencillez a la hora de mantener comunicaciones series, ha sido utilizado a lo largo de todo el proyecto como interfaz gráfica entre el PC y el GPS. La comunicación mediante este programa quedaba sujeta única y exclusivamente al envío y recepción de tramas entre un equipo y otro, por lo que la interpretación de estas se hacia bastante compleja y pesada.

En las fases iniciales del proyecto, el HyperTerminal nos permitió un primer acercamiento a los equipos GPS. La comunicación en este período se basó simplemente en algo tan simple como solicitar cierta información al receptor y en el posterior estudio de los datos recibidos, familiarizando así al programador con los receptores de GPS.

También fue muy útil el HyperTerminal durante la implementación de la aplicación, tanto en las primeras versiones en las que sirvió para chequear el funcionamiento de las comunicaciones series, como en las versiones intermedias en las que se utilizó para capturar las distintas respuestas que ofrecía el GPS ante los posibles estímulos de nuestra aplicación.

En las versiones finales el uso del HyperTerminal fue algo distinto. En este caso también fue utilizado para capturara los datos recibidos, pero con un objetivo diverso. Se trataba de estudiar ciertos comportamientos extraños del GPS, los cuales no se ceñían a la información suministrada por el fabricante o bien, no concordaban con lo esperado. Gracias la captura de estos datos y a su posterior estudio, se llegaron a una serie de conclusiones lógicas, que nos permitieron entender mejor el funcionamiento del GPS.

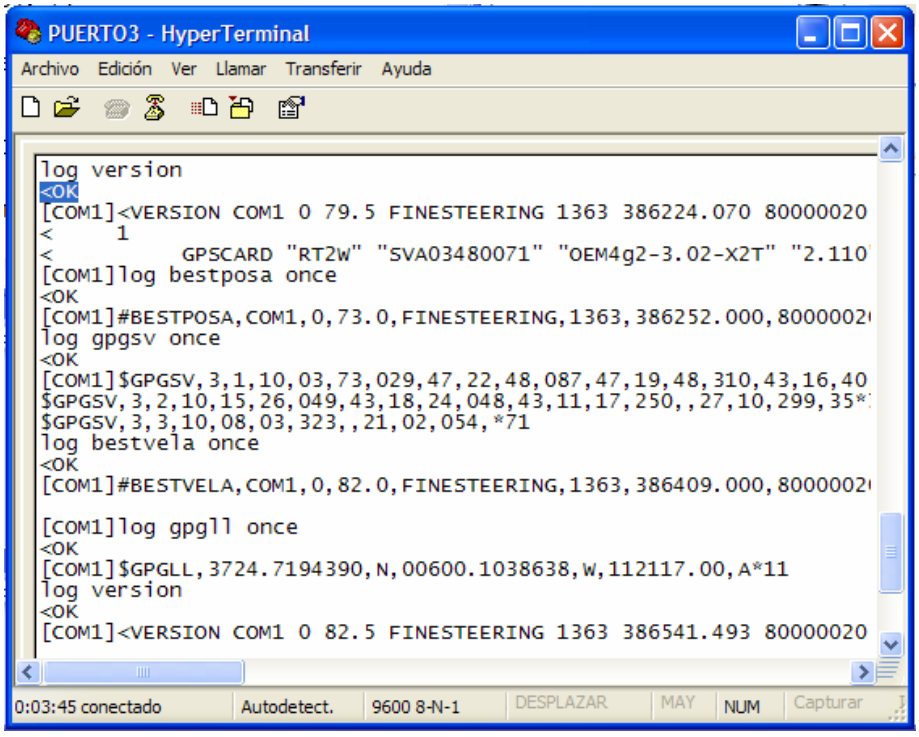

Imagen 3.15: HyperTerminal trabajando con el GPS

A pesar de todo esto, al comienzo de este punto se ha dicho que el hyperTerminal tenía una serie de limitaciones, debidas a las cuales se optó por implementar una aplicación auxiliar, llamada TxonRxonSerie (ver Anexo B). Esta aplicación nació con el único objetivo de simular el funcionamiento del GPS, cosa que se hacia especialmente difícil con el Hyperterminal, por la imposibilidad de copiar y pegar las largas y complejas tramas que debía enviarnos el equipo receptor.

3.2.4 El programa GPSolution.

GPSolution es una aplicación Windows de 32-bits. La aplicación proporciona al usuario una interfaz gráfica que le permite configurar y monitorizar a la hora de trabajar con los receptores de Novatel de la familia OEM4, como es el caso del equipo con el que vamos a trabajar en este proyecto. Este programa fue suministrado por el fabricante junto con los nuevos equipos RT-2.

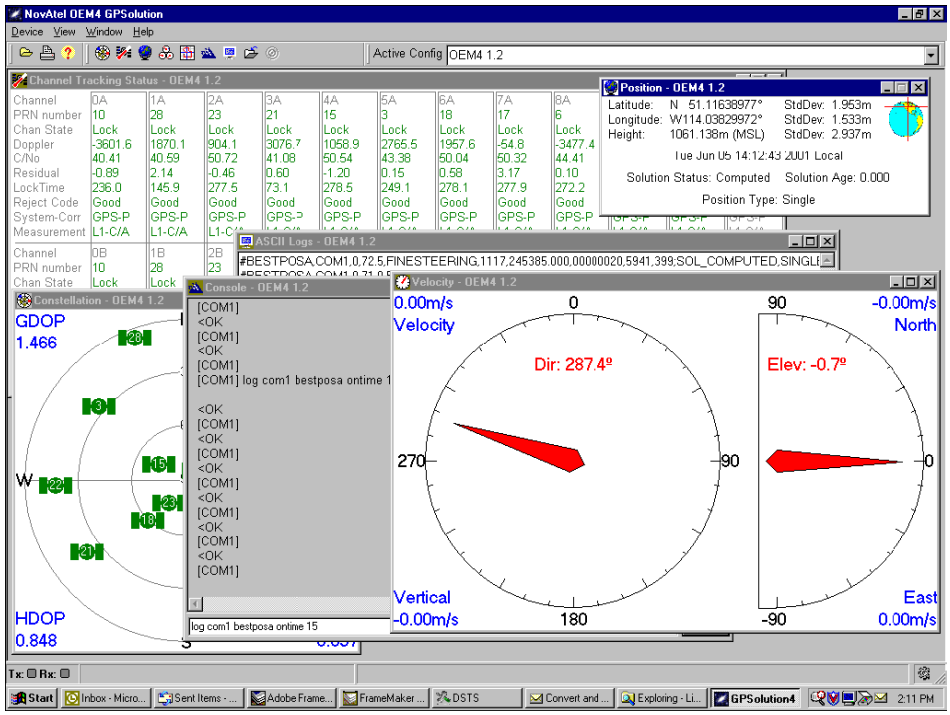

Imagen 3.16 : Aplicación GPSolution

La mayoría de las ventanas tiene un menú asociado que aparece al hacer clic en el botón derecho. Estos proporcionan al usuario un modo de gestionar el funcionamiento de dichas ventanas a la hora de trabajar con la aplicación. Algunas de las funciones que nos permite realizar este software son las siguientes.

- **Constellation Window**: Esta ventana muestra la posición de los satélites que está a la vista. Los números PRN de cada satélite son mostrados en el centro de

estos. Haciendo doble clic en uno de los satélites aparecerá una ventana con toda la información relativa a este satélite.

- **Channel Tracking Status Window**: Esta ventana proporciona al usuario la posibilidad de ver el estado de los satélites a la vista en forma de tabla.
- **Position Window**: Al activar esta venta podemos visualizar la Latitud, Longitu y Altura, entrono a una cierta desviación estándar, en la que está situada nuestra antena de GPS.
- **Velocity and Heading Window**: Esta ventana muestra la velocidad y la dirección en la que se desplaza nuestro equipo.
- **Console Window**: permite al usuario comunicarse directamente con el equipo receptor de GPS. Para ello, únicamente es necesario teclear el comando que se quiere enviar, pulsar <Enter> y esperar la respuesta.

Podemos ver que este programa se adapta bastante bien a los objetivos que nos marcamos al iniciar el proyecto, no obstante GPSolution no presenta la posibilidad de configurar de un modo automatizado los equipos para su funcionamiento en modo diferencial. Otro motivo que nos impulso a la creación de InterGPS, era la idea de disponer en el laboratorio de un software libre y totalmente realizado por nosotros, de tal modo que pudiera ser ampliado y modificado según nuestras necesidades presentes y futuras.

3.3 ESTRUCTURA SOFTWARE DE LA APLICACIÓN

La estructura software de nuestra aplicación esta formada por diferentes hilos: el hilo de lectura, el hilo de escritura y el hilo principal (este último aparece en el cuadro como HiloPpal). Gracias a estos tres hilos funcionando en paralelo y al flujo principal de la ejecución, hemos dotado de mayor velocidad y potencia al programa (ver figura siguiente).

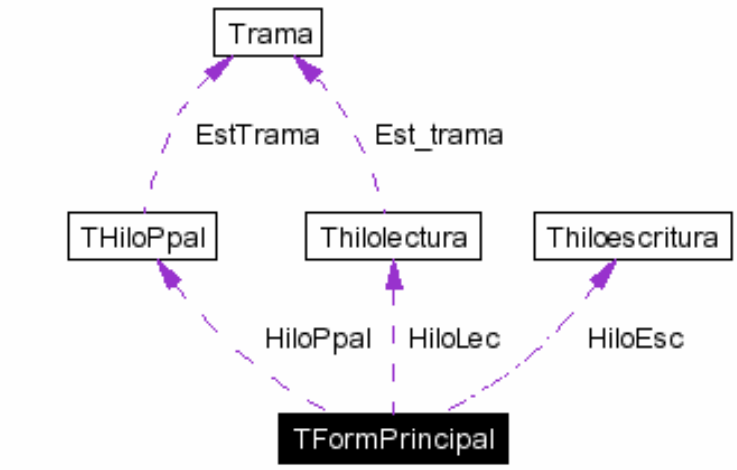

Imagen 3.17 : Estructura Software de InterGPS

Los tres hilos con los que cuenta InterGPS son creados justo antes de iniciar la comunicación serie, pero únicamente los de escritura y lectura son lanzando antes de que de que el correcto funcionamiento de las comunicaciones se haya confirmado. Una vez que el estado de la conexión se ha comprobado y ha resultado correcto, es cuando se lanza el tercero de los hilos, el hilo principal. El hecho de que los dos primero comiencen a funcionar con anterioridad, es tan simple como que estos son los que se encargan de la transmisión y recepción de tramas por parte del PC, de ahí que la confirmación de las comunicaciones no sea posible si estos hilos aun permanecen dormidos.

- El **hilo de lectura**, como su propio nombre indica, irá procesando todo los caracteres que se reciban por el puerto serie. En un primer momento, el hilo permanece esperando uno de los caracteres de inicio de trama, que en este caso sarán '#' y '$'. Una vez que se ha recibido uno de estos caracteres, lo almacena en una cadena y a medida que van llegando nuevos los va añadiendo al final de dicha cadena. Si una vez recibido un carácter de inicio trama llega otro de los considerados caracteres de inicio, el hilo borrará toda la cadena recibida con anterioridad y comenzará a almacenar caracteres en una nueva trama. De este modo, si durante la recepción de un mensaje ocurre un error y este se interrumpe, el equipo podrá seguir gestionando las tramas que nos lleguen con posterioridad. Por otro lado, si estando recibiendo una cadena nos llega el carácter de retorno de carro (código ASCII 13), el hilo interpretara que la trama se ha terminado y comenzará a interpretarla.

Una vez completada la recepción de una trama de información, lo primero que haremos será comprobar si los caracteres que la forman se han recibido correctamente. Para ello lo que haremos será comprobar si el código CCR de dicha trama coincide con el calculado por nuestra aplicación. Si ambos códigos CCR coinciden, la trama será gestionada carácter a carácter almacenando su información en unas estructuras determinadas. Ya hemos dicho que los caracteres de inicio de trama son dos ('#' y '$'), pues bien, según el tipo de carácter de inicio, tiene un determinado formato de CCR u otro. Las tramas de tipo $, posee un CCR de 8 bits que se corresponden con dos caracteres ASCII y las tramas de tipo #, uno de 32 bits que se corresponden 8 caracteres ASCII. La relación existente entre los caracteres ASCII y la secuencia de bits ha sido descrita en el Anexo B, donde se explica el funcionamiento de una aplicación auxiliar (CalculaCRC) que sirvió para comprobar el funcionamiento de nuestro programa en lo referente al cálculo de los CCR.

Una vez que la trama ha sido considerada correcta, lo primero que haremos será identificar su cabecera. Si dicha cabecera coincide con alguna de las que nuestro programa es capaz de interpretar, todos los datos contenidos en la trama serán extraído uno por uno y almacenados en una estructura (llamada trama) que contiene las distintas variables propias de cada tipo de mensaje. Cuando hayamos terminado de procesar y almacenar lo datos de la cadena, la estructura en la que ha sido guardada toda la información será apilada en una cola de estructura para su posterior tratamiento.

- El **hilo de escritura** es el más simple de los tres. Este se encarga de comprobar continuamente si hay algo para transmitir por el puerto serie y en caso de que lo haya lo enviará. La información a transmitir se le pasa al hilo a través de una cadena. Esta cadena es pública por lo que se podrá acceder a ella desde diferentes partes del código de nuestro programa.

- El **hilo principal** (HiloPpal) como ya hemos dicho no comienza a funcionar hasta que las comunicaciones no han sido establecidas y comprobado su correcto funcionamiento. Este hilo se encargará única y exclusivamente de gestionar la cola de estructuras. Ya dijimos que a medida que íbamos recibiendo e identificando tramas, los datos de estas eran almacenados en unas estructuras que posteriormente se apilaban para su procesado. El hilo principal está continuamente comprobando si se ha depositado alguna estructura en la cola y en el supuesto caso de que se haya hecho la extraerá, identificará de qué tipo de trama se trata y posteriormente llamará a las funciones encargadas de trabajar con dicho datos, una vez que esto se haya hecho, eliminará la estructura trama correspondiente.

- El **flujo principal** de ejecución es el encargado de abrir y cerrar las distintas ventanas, actualizar la información y refrescar los datos que se muestran por pantalla, además de ser el encargado de crear y lanzar los diferentes hilos.

3.4 CONFIGURACIÓN DE LAS COMUNICACIONES

Las comunicaciones constituyen el punto principal en el desarrollo de la aplicación. En base a como se implemente dicha parte girarán el resto de las capacidades del programa. El código mediante el cual se lleva a cabo estas tareas debe ser flexible, eficiente en velocidad pero a su vez robusto. Este es el motivo de que esta sección de código sea la que más modificaciones ha sufrido a lo largo del proyecto.

El receptor de GPS se comunica con el exterior mediante algunos de sus dos puertos regidos por la norma RS-32. El primer paso en el desarrollo de un programa de comunicaciones es siempre el mismo, conseguir transmitir y recibir datos. Para ello y como ya hemos dicho anteriormente, se utilizo el HyperTerminal de Windows, que nos permitió verificar el buen funcionamiento de nuestro programa.

En las primeras versiones de código se establecía una comunicación fija, es decir, el programa no nos permitía cambiar los parámetros con los que se configuraba el puerto. Posteriormente, y una vez que se comprobó el buen funcionamiento de nuestro programa con una configuración fija, se implementó una nueva sección del programa en la que se nos permitía abrir los puertos con diferentes parámetros e incluso una vez abiertos poder reconfigurar las comunicaciones. No obstante, el programa nos permite en todo momento elegir una configuración predeterminada con sólo hacer click en un botón. Esta configuración predeterminada consiste en 19200 bits por segundo, sin paridad ni control de flujo y con 1 bit de parada por cada 8 bits de datos.

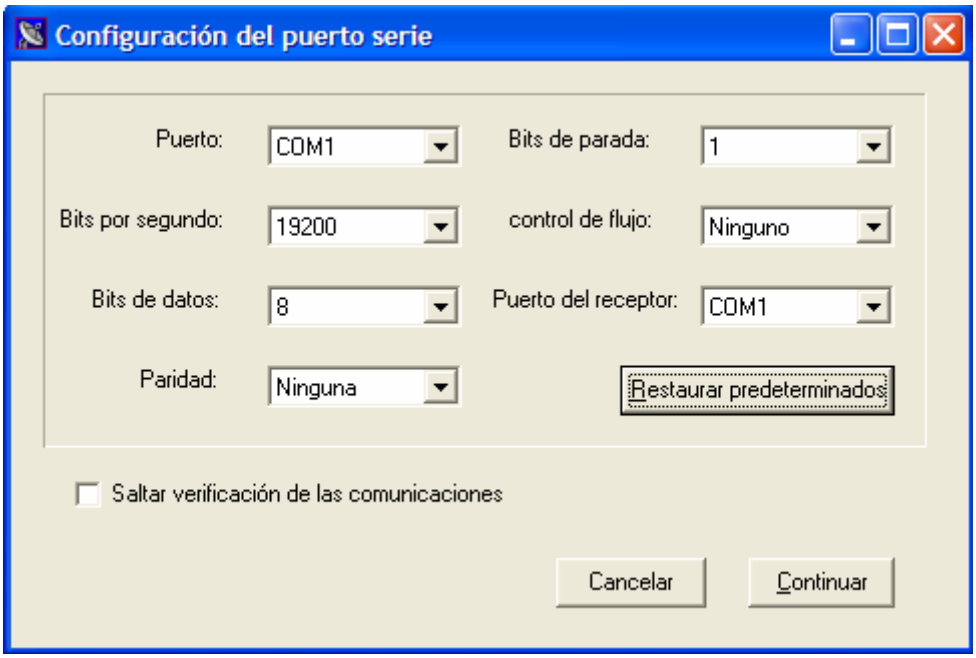

Imagen 3.18 :Ventana de configuración de las comunicaciones

En la siguiente tabla se muestran todas las opciones que nos ofrece nuestro programa a la hora de configurar el puerto serie.

Parámetro	Posibles valores
Puerto Com	**Com1**, Com2, Com3, Com4
Bits por segundo	2400, 4800, 9600, **19200**
Paridad	**Ninguna**, par, impar
Bits de datos	5, 6, 7, **8**
Bits de stop	**1**, 1.5, 2
Control de flujo	**Ninguno**, Xon/Xoff, Hardware
Puerto de receptor	**Com1**, Com2

Tabla 3.1: Configuraciones del puerto serie

Puerto Com, hace referencia al puerto del PC por el que se establecerá la comunicación. El parámetro “Puerto del receptor” indica el puerto del GPS al que hemos conectado el PC. Este último no es estrictamente necesario ni a la hora de establecer la comunicación, ni a la hora de trabajar con los equipos Novatel RT-2, pero si se hace primordial cuando estamos trabajando con otros equipos, como pueden ser los Novatel RT-20.

Para comunicarnos con los distintos GPS y solicitar la información necesaria, utilizamos el comando “LOG”. Como podemos ver en el Anexo C donde se explica detalladamente este comando, en los equipos Novatel RT-2 no es necesario indicar el

puerto por el que queremos que se nos envíe la información requerida. Sin embargo, esto no es así para todos los equipos. Por ejemplo, para los antiguos equipos GPSCard Novatel RT-20 es necesario en todo momento indicar porque puerto se nos enviará la información. Debido a esto y para que nuestra aplicación sea capaz de funcionar con estos equipos, se hace necesario añadir un parámetro en el que se indique dicho puerto.

3.4.1 Comprobación de las comunicaciones

Antes de establecer la conexión serie todas las funciones de nuestro programa están deshabilitadas. Por tanto no podremos acceder a dichas funciones hasta que la comunicación no se haya establecido correctamente.

La fase de comprobación es algo tan sencillo como solicitar una información concreta al equipo receptor de GPS y esperar durante un tiempo prudencial a que este nos la devuelva. Si pasado este tiempo no se ha recibido nada o lo que se ha recibido no corresponde con lo esperado, se dará un mensaje por pantalla en el que se indicará que algo no funciona como esperábamos o que simplemente no se ha recibido nada. Si realizado el proceso de comprobación todo ha salido correctamente, el programa habilitará todas la opciones de nuestra aplicación pudiendo entonces acceder a cada una de ellas.

La trama que se solicita al equipo para este proceso de verificación es la trama GPZDA. Al solicitar este mensaje el equipo nos devuelve una trama en la que estará contenida la fecha y la hora a la que se ha realizado la petición. Para solicitar la trama, nuestro programa enviará por el puerto serie el siguiente comando

LOG COMx GPZDA ONTIME 5

Con este comando estamos solicitando que se nos envíe una trama GPZDA cada 5 segundos a través del puerto Com x del equipo de GPS. Una vez que hemos recibido al menos una de estas tramas el equipo la interpretará comprobará si es correcta y mostrará por pantalla los datos pertinentes.

Imagen 3.19 : Ventana de confirmación de las comunicaciones

Confirmado el buen funcionamiento de las comunicaciones el equipo solicita la interrupción del envío de tramas GPZDA mediante el siguiente comando.

UNLOG COMx GPZDA

A continuación se ha presentado una tabla suministrada por el fabricante. En este cuadro podemos apreciar tanto la estructura del mensaje GPZDA como la información contenida en cada uno de los bloques que forman la trama

Field	Structure	Field Description	Symbol	Example
1	$GPZDA	Log header		$GPZDA
2	utc	UTC time	hhmmss.ss	220238.00
3	day	Day, 01 to 31	xx	15
4	month	Month, 01 to 12	xx	07
5	year	Year	xxxx	1992
6	null	Local zone description - not available	xx	, ,
7	null	Local zone minutes description - not available [1]	xx	, ,
8	*xx	Checksum	*hh	*6F
9	[CR][LF]	Sentence terminator		[CR][LF]

1 Local time zones are not supported by the OEM4 family receiver. Fields 6 and 7 will always be null.

Tabla 3.2 : Estructura del mensaje GPZDA

Ejemplo:

$GPZDA,113732.00,23,02,2006,,*64

En el ejemplo anterior podemos ver que no todos los campos tienen porque estar presentes y que los caracteres que van detrás del * forman parte del código CCR de la trama. Como era de esperar, al ser un mensaje de tipo '$' el CCR está formado por dos caracteres ASCII (Ver anexo B).

Por último añadir que es posible que en ciertas ocasiones no sea necesario comprobar la buena configuración de las comunicaciones, como puede ser el caso de trabajar con un equipo de otro fabricante que no se comporte del modo esperado. Por este motivo se ha añadido una opción en el formulario de configuración que nos permite saltar estar la fase de verificación, no obstante esta opción vendrá siempre desactiva por defecto.

3.5 MONITORIZACIÓN DE LA INFORMACIÓN

Como ya hemos mencionado con anterioridad el flujo principal de ejecución se encarga de ir abriendo y cerrando las distintas ventanas de las que consta nuestro proyecto. Cada vez que una de ellas es abierta, el flujo principal solicitará al GPS todos los datos requeridos por la ventana en cuestión. Esto último lo hace volcando sobre el hilo de escritura los comandos pertinentes para la solicitud de la información, siendo este último hilo el que envía los mensajes por el puerto serie. Una vez que el GPS ha recibido la solicitud de envió comienza a transmitir los datos requeridos, de modo que

nuestra aplicación los vaya recibiendo y almacenando para su posterior procesado y monitorización.

El encargado de recoger y almacenarla la información a medida que nos va llegando por el puerto serie será el hilo de lectura. Una vez almacenados los datos, el hilo principal se encargará de procesar y monitorizar la información recibida.

En este apartado vamos a presentar los datos que nuestra aplicación es capaz de interpretar y las tramas que utiliza para que esto sea posible. No vamos a tratar la monitorización de la fecha y la hora puesto que ya ha sido explicada en el punto 3.3.1

3.5.1 Monitorización de la posición

Una de las opciones de nuestra aplicación es la monitorización la posición, entendiendo por esta la altitud, longitud y latitud del punto en el que está situada nuestra antera receptora. Aunque a simple vista esta información se consigue con tan solo solicitar una trama, veremos que la realización de esta aplicación ha sido algo más problemática de lo esperado.

En un primer momento se implemento una función en la que se solicitaba la trama GPGLL mediante el comando:

LOG COMx GPGLL ONTIME 30

El problema fue que una vez terminada su implementación este comando no nos proporcionaba información alguna sobre la altitud. Debido a que el conocimiento exacto de la posición de la antena receptora era de gran importancia a la hora de configurar los equipos para su funcionamiento en modo diferencial, y que un pequeño error en su determinación podía causar pérdidas de precisión en el cálculo de las posiciones de las estaciones remotas, se optó por abandonar esta trama por un segundo tipo de mensaje, el GPGGARTK. En la siguiente tabla se muestra la información suministrada por la trama GPGLL

Field	Structure	Field Description	Symbol	Example
1	$GPGLL	Log header		$GPGLL
2	lat	Latitude (DDmm.mm)	llll.ll	5106.7198674
3	lat dir	Latitude direction (N = North, S = South)	a	N
4	lon	Longitude (DDDmm.mm)	yyyyy.yy	11402.3587526
5	lon dir	Longitude direction (E = East, W = West)	a	W
6	utc	UTC time of position (hours/minutes/seconds/ decimal seconds)	hhmmss.ss	220152.50
7	data status	Data status: A = Data valid, V = Data invalid	A	A
8	*xx	Checksum	*hh	*1B
9	[CR][LF]	Sentence terminator		[CR][LF]

Tabla 3.3 : Estructura del mensaje GPGLL

Ejemplo:

$GPGLL,3724.7194390,N,00600.1038638,W,112117.00,A*11

La segunda trama utilizada, la GPGGARTK, si nos proporcionaba información sobre la altitud, por lo que una vez más implementamos la aplicación funcionando ahora con el mensaje GPGGARTK. Para la solicitud de esta trama se utilizaba el siguiente comando.

LOG COMx GPGGARTK ONTIME 30

Posteriormente ha mostrado la tabla relativa al comando GPGGARTK

Field	Structure	Field Description	Symbol	Example
1	$GPGGA	Log header		$GPGGA
2	utc	UTC time of position (hours/minutes/seconds/ decimal seconds)	hhmmss.ss	220147.50
3	lat	Latitude (DDmm.mm)	llll.ll	5106.7194489
4	lat dir	Latitude direction (N = North, S = South)	a	N
5	lon	Longitude (DDDmm.mm)	yyyyy.yy	11402.3589020
6	lon dir	Longitude direction (E = East, W = West)	a	W
7	GPS qual	GPS Quality indicator 0 = fix not available or invalid 1 = GPS fix 2 = Differential GPS fix 4 = RTK fixed ambiguity solution 5 = RTK floating ambiguity solution 9 = WAAS [2]	x	1
8	# sats	Number of satellites in use (00-12). May be different to the number in view	xx	08
9	hdop	Horizontal dilution of precision	x.x	0.9
10	alt	Antenna altitude above/below mean sea level (geoid)	x.x	1080.406
11	units	Units of antenna altitude (M = meters)	M	M
12	null	(This field not available on OEM4 family receivers)		,,
13	null	(This field not available on OEM4 family receivers)		,,
14	age	Age of Differential GPS data (in seconds) [1]	xx	,,
15	stn ID	Differential base station ID, 0000-1023	xxxx	,,
16	*xx	Checksum	*hh	*48
17	[CR][LF]	Sentence terminator		[CR][LF]

1 The maximum age reported here is limited to 99 seconds.

2 An indicator of 9 has been temporarily set for WAAS. The NMEA standard for WAAS has not been decided yet.

Tabla 3.4 : Estructura del mensaje GPGGARTK

Ejemplo:

$GPGGA,113621.00,3724.7203,N,00600.1041,W,1,08,1.0,18.71,M,51.10,M,,*4D

Podemos ver que la cabecera de este mensaje no es, como cabría esperar GPGGARTK, sino únicamente GPGGA. Esto se debe a que en realidad son la misma trama salvo que la primera, tiene algo más de precisión en lo que se refiere a medidas de altitud, longitud y latitud.

Una vez terminado el proyecto y comprobado ya, su uso con los equipos OEM4 RT-2 se empezó a trabajar con otros equipos para ver la versatilidad que mostraba InterGPS. Este hecho nos obligo a cambiar una vez más la trama utilizada para la monitorización de la posición. El motivo por el que tuvimos que volver a sustituir este tipo de mensaje fue el hecho de que los antiguos equipos Novatel GPSCard RT-20 no incluían entre sus tramas las ya mencionadas GPGGARTK. Debido a este inconveniente tuvimos que cambiar por tercera y última vez la trama a solicitar. En este caso se recurrió a la GPGGA, que sí era interpretada por los receptores RT-20 y además nos proporcionaban todo la información necesaria para el funcionamiento de nuestra aplicación. A continuación se muestra el comando para la solicitud de trama y la tabla de información suministrada por el fabricante Novatel.

LOG COMx GPGGARTK ONTIME 30

Field	Structure	Field Description	Symbol	Example
1	$GPGGA	Log header		$GPGGA
2	utc	UTC time of position (hours/minutes/seconds/ decimal seconds)	hhmmss.ss	220147.50
3	lat	Latitude (DDmm.mm)	llll.ll	5106.7194489
4	lat dir	Latitude direction (N = North, S = South)	a	N
5	lon	Longitude (DDDmm.mm)	yyyyy.yy	11402.3589020
6	lon dir	Longitude direction (E = East, W = West)	a	W
7	GPS qual	GPS Quality indicator 0 = fix not available or invalid 1 = GPS fix 2 = Differential GPS fix 4 = RTK fixed ambiguity solution 5 = RTK floating ambiguity solution 9 = WAAS [2]	x	1
8	# sats	Number of satellites in use (00-12). May be different to the number in view	xx	08
9	hdop	Horizontal dilution of precision	x.x	0.9
10	alt	Antenna altitude above/below mean sea level (geoid)	x.x	1080.406
11	a-units	Units of antenna altitude (M = meters)	M	M
12	undulation	Undulation	x.x	-16.271
13	u-units	Units of undulation (M = meters)	M	M
14	age	Age of Differential GPS data (in seconds) [1]	xx	,,
15	stn ID	Differential base station ID, 0000-1023	xxxx	,,
16	*xx	Checksum	*hh	*48
17	[CR][LF]	Sentence terminator		[CR][LF]

1 The maximum age reported here is limited to 99 seconds.

2 An indicator of 9 has been temporarily set for WAAS. The NMEA standard for WAAS has not been decided yet.

Tabla 3.5 : Estructura del mensaje *GPGGA*

Ejemplo:

$GPGGA,220147.50,5106.7194489,N,11402.3589020,W,1,08,0.9,1080.406,M,,,,*48

Podemos ver que la tabla es exactamente la misma para los dos mensajes, la única diferencia es que los equipos RT-2 son capaces de trabajar con ambos mensajes y los Novatel RT-20 no son capaces de trabajar con las tramas GPGGARTK.

En la siguiente tabla se han mostrado las diferencias de precisión existente entre usar una u otra trama. Como podemos ver hemos perdido un poco de precisión al usar la trama GPGGA, pero esto ha sido obligatorio a consecuencia de lo ya explicado anteriormente.

NMEA Log	Latitude (# of decimal places)	Longitude (# of decimal places)	Altitude (# of decimal places)
GPGGA	4	4	2
GPGGARTK	7	7	3
GPGLL	7	7	N/A
GPRMC	7	7	N/A

Tabla 3.6 : Diferencias de precisión

3.5.2 Monitorización de los satélites.

Otras de las posibilidades que ofrece nuestro programa es la posibilidad de mostrar por pantalla la posición en la que se encuentran los distintos satélites que tiene a la vista nuestro equipo.

Para la realización de este apartado se han utilizados dos mensajes distintos, el SATVISA y el GPGSV. Ambos mensajes dan aproximadamente la misma información sobre los satélites, el problema es que el mensaje SATVISA es un mensaje propio de Novatel y más concretamente de los equipo OEM4 RT-2, por lo que si únicamente se trabajara con este tipo de tramas nuestro programa no sería capaz de funcionar con los equipos de otros fabricantes. Por otro lado los antiguos receptores de GPS, los Novatel RT-20, tampoco son capaces de trabajar con las tramas SATVISA de ahí que haya sido necesario el uso de las tramas GPGSV, perteneciente al estándar NMEA. Posteriormente se muestra la tabla que suministra el fabricante para cada una de las dos tramas solicitadas y se estudiarán con mayor detenimiento ambos tipos de mensajes.

La primera trama que vamos a abordar será la más general, la GPGSV. Hemos añadido a continuación el comando que se ha enviado por el puerto serie, a modo de solicitud de este tipo de tramas y además también se ha introducido la tabla de estructura de este mensaje.

LOG COMx GPGSV ON TIME 600

Field	Structure	Field Description	Symbol	Example
1	$GPGSV	Log header		$GPGSV
2	# msg	Total number of messages, 1 to 3	x	3
3	msg #	Message number, 1 to 3	x	1
4	# sats	Total number of satellites in view	xx	09
5	prn	Satellite PRN number	xx	03
6	elev	Elevation, degrees, 90 maximum	xx	51
7	azimuth	Azimuth, degrees True, 000 to 359	xxx	140
8	SNR	SNR (C/N_0) 00-99 dB, null when not tracking	xx	42
... ...	... ...	Next satellite PRN number, elev, azimuth, SNR, ... Last satellite PRN number, elev, azimuth, SNR,		
variable	*xx	Checksum	*hh	*72
variable	[CR][LF]	Sentence terminator		[CR][LF]

Tabla 3.7 : Estructura del mensaje GPGSV

Ejemplo:

$GPGSV,3,1,11,06,78,273,44,10,54,070,44,17,40,255,40,30,30,206,38*78
$GPGSV,3,2,11,24,21,071,37,13,20,058,37,22,13,323,39,05,07,182,35*73
$GPGSV,3,3,11,26,02,143,,23,02,203,,18,01,014,*46

Como podemos ver esta trama puede enviar información de hasta 12 satélites divididos en tres submensajes. Cada uno de ellos, contiene una cabecera con el número de submensajes total y el número de submensaje que ocupa dentro de la cadena.

La segunda trama utilizada es la SATVISA. Para solicitar este tipo de mensajes se ha enviado por el puerto serie el siguiente comando.

LOG COMx SATVISA ONTIME 600

Esta trama nos da información de todos los satélites de GPS que orbitan en el espacio. A primera vista esto debería ser imposible puesto que el número de canales que posee nuestro equipo es de 12, mientras que los satélites que orbitan alrededor de la tierra de un modo activo desde el punto de vista de GPS son 24. Otro aspecto por lo que esto debería resultar imposible es el hecho de que en un determinado momento el número máximo de satélites que un receptor de GPS puede tener a la vista son 12. No obstante, todo esto tiene lugar gracias a la transferencia de información entre los satélites y las estaciones de referencia, es decir, podemos saber la situación y el estado de todos los satélites orbitantes gracias a los datos contenidos en los almanaques y en las efemérides de los mismos. Esto presenta un problema a la hora de monitorizar la información, puesto que no queremos representar todos los satélites existentes sino solo aquello que tenemos a la vista en cada momento. Para poder solucionar este problema se estuvo capturando tramas GPGSV y tramas SATVISA durante varios periodos de tiempo, con el fin de estudiar los resultados obtenidos de modo que se estableciera una relación entre los satélites capturados por cada una de las distintas tramas.

Finalmente se llegó a la siguiente conclusión. Todos los satélites observados por la trama GPGSV, presentaban una característica común cuando se les estudiaba a partir de

los parámetros dados por las tramas SATVISA, esta propiedad en común era la de poseer una altitud mayor que cero, mientras que el resto de satélites, los que no aparecían en las trama GPGSV presentaban una altitud negativa. Por otro lado no es de extrañar esta característica, puesto que los satélites que estaban situados por debajo de nuestro horizonte no podrían en ningún momento conectarse con nuestro equipo. Llegada a esta conclusión se tomo la siguiente media, por cada trama SATVISA recibida se tomarían y se representarían los datos pertenecientes a todos aquellos satélites con altitud mayor que cero y por el contra todos aquellos satélites que no cumplieran esta especificación serían despreciados. Como consecuencia de la decisión tomada, el resultado mostrado por pantalla resultaba ser el mismo si trabajamos con un tipo de mensaje u otro, con la única diferencia de que trabajando con los SATVISA tendríamos cierta información adicional.

Para terminar, mostraremos la tabla suministrada por el fabricante correspondiente a la trama SATVISA.

Field #	Field type	Data Description	Format	Binary Bytes	Binary Offset
1	header	Log header		H	0
2	sat vis	Is satellite visibility valid? 1 = TRUE 0 = FALSE	Enum	4	H
3	comp alm	Was complete almanac used? 1 = TRUE 0 = FALSE	Enum	4	H+4
4	#sat	Number of satellites with information to follow	Ulong	4	H+8
5	PRN	GPS satellite PRN number of range measurement.	Short	2	H+12
6	Reserved		Short	2	H+14
7	health	Satellite health[a]	Ulong	4	H+16
8	elev	Elevation (degrees)	Double	8	H+20
9	az	Azimuth (degrees)	Double	8	H+28
10	true dop	Theoretical Doppler of satellite	Double	8	H+36
11	app dop	Apparent Doppler for this receiver	Double	8	H+44
12	Next satellite offset = H + 12 + (#sat x 40)				
variable	xxxx	32-bit CRC (ASCII and Binary only)	Hex	4	H+12+ (#sat x 40)
variable	[CR][LF]	Sentence terminator (ASCII only)	-	-	-

a. Satellite health values may be found in ICD-GPS-200. To obtain copies of ICD-GPS-200, see ARINC in the appendix on *Standards and References* in *Volume 1* of this manual.

Tabla 3.8 : Estructura del mensaje SATVISA

Ejemplo:

#SATVISA,COM1,0,44.0,FINESTEERING,1039,490308.000,00000028,6002,0;
TRUE,TRUE,27,
14,0,0,74.5,267.4,458.2,458.926672761,
25,0,0,61.3,73.7,-1252.6,-1251.902056196,
...
26,0,0,-82.0,114.8,-188.9,-188.237459086*bf8c9522

3.6 CONFIGURACIÓN DE LA ESTACIÓN BASE

Hemos dicho repetidas veces que la finalidad de este proyecto es la de implementar una aplicación que nos permita configurar nuestros equipos de una manera rápida y fácil para su uso en modo diferencial. En este apartado explicaremos cómo y mediante que mensajes hemos conseguido realizar este objetivo.

Sateyo, el programa antecedente a nuestra aplicación, hacía una gran diferenciación entre configurar un equipo base y configurar un equipo móvil. InterGPS, por el contrario, solo hace referencia a la configuración de los equipos como estación base. El motivo por el que no se ha tenido en cuenta la configuración de la estación móvil en este programa, es tan sencillo como que en los nuevos equipos de Novalte, los EOM4 RT-2, la configuración de los equipos como estaciones móviles ya viene realizada por defecto, por lo que podemos prescindir de su configuración.

Por otro lado, en lo referente a la configuración de un equipo como estación base, también se ha simplificado mucho este proceso. Esto se debe a que gran parte del proceso de configuración se ha introducido directamente en la memoria EEPROM contenida en la tarjeta de nuestra estación base, de ahí, que para la aplicación InterGPS configurar una estación base se haya reducido simple y llanamente a fijar, de un modo lo más preciso posible, la posición en la que ésta se encuentra. La precisión con la que se calcula dicha posición es de suma importancia puesto pequeños errores de a la hora de fijar la posición de la antena receptora del equipo base puede aumentar considerablemente los errores obtenidos en los cálculos realizados por los equipos móviles.

Nuestra aplicación nos ofrece dos opciones para fijar la posición de la estación base. La primera de ellas, la más sencilla, consiste en introducir directamente los valores que definen el punto donde nos encontramos en términos de latitud, longitud y altitud. La segunda se basa en un cálculo previo de estos valores antes de introducirlos como punto de localización de la estación base.

3.6.1 Fijar posición

Como su propio nombre indica este método consiste única y exclusivamente en introducir la latitud, longitud y altitud de la posición en que se encuentra nuestra estación base. El modo por el que se han obtenido estos parámetros es irrelevante para la aplicación. Hay que destacar que este método está sujeto a ciertas comprobaciones, por lo que no podemos fijar la situación de la antena de un modo aleatorio. Según esto, si la posición que queremos dar se aleja en demasía de la que el GPS ha estimado como propia, los datos introducidos serán despreciados y la posición dada no vendrá fijada.

Para fijar la posición se han utilizado conjuntamente varios mensajes. A continuación mostraremos las secuencias de comando que envía nuestra aplicación al receptor de GPS para realizar esta operación.

Lo primero que hacemos será cancelar la posición fijada inicialmente. Para ello utilizaremos el comando FIX. Hemos añadido la tabla de este comando de modo que se proporcione al lector un mayor conocimiento sobre él.

FIX type [param1 [param2 [param3]]]

Field	Field Type	ASCII Value	Binary Value	Description	Binary Format	Binary Bytes	Binary Offset
1	header	-	-	This field contains the command name or the message header depending on whether the command is abbreviated ASCII, ASCII or binary, respectively.	-	H	0
2	type	See *Table 22 on Page 70*		Fix type	Enum	4	H
3	param1	See *Table 21*		Parameter 1.	Double	8	H + 4
4	param2			Parameter 2.	Double	8	H + 12
5	param3			Parameter 3.	Double	8	H + 20

Tabla 3.9 : Estructura del comando FIX

Como podemos ver en la tabla, si lo que queremos es cancelar la posición fijada anteriormente en nuestro receptor, debemos transmitir el siguiente comando.

FIX NONE

Una vez cancelada esta posición, fijaremos la posición a partir de los datos introducidos por el usuario mediante el siguiente mensaje.

FIX POSITION LAT LON ALT

Hecho esto, si los valores que se han facilitado al GPS están dentro de los límites que el receptor considera oportunos, la posición introducida vendrá fijada como localización de la estación base.

3.6.2 Obtener y fijar posición

El segundo métodos no sólo consiste en introducir los datos de latitud, longitud y altura, sino que además consta de una fase previa en la que se que estimará el punto donde nos encontramos. Existen dos modos de calcular esta posición, ya sea usando los mensajes POSAVE o mediante el comando LOG.

3.6.2.1 Posave

POSAVE es un tipo de trama propio de los equipos Novatel RT-2, por lo que esta opción no será posible realizar con otros receptores. El funcionamiento de este comando es bastante sencillo por lo que para su explicación no centraremos en la estructura del mismo.

POSAVE [Estado] Tmax DesH DesvV

Podemos ver que el comando POSAVE está formado por varios parámetros. Cada uno de estos parámetros no servirá para delimitar tanto la precisión como la duración del promediado.

El parámetro “Estado” nos indica si el proceso de promedio se encuentra iniciado o parado. En caso de que este esté iniciado, “Estado = ON”, podremos detenerlo enviando simplemente mensaje siguiente.

POSAVE OFF

Tmax indica la duración máxima que debe tardar el equipo de GPS en alcanzar los valores de precisión establecidos.

DesH y DesV sirven para indicar al GPS la desviación estándar máxima horizontal y vertical que queremos conseguir.

Este proceso terminará una vez se hayan alcanzado los límites de precisión establecidos o bien una vez se haya superado el tiempo máximo de duración. Llegados a este momento se ofrecerá al usuario la posibilidad de fijar los resultados obtenidos como punto de localización de la estación base.

A continuación se adjunta la tabla perteneciente a este mensaje

Field	Field Type	ASCII Value	Binary Value	Description	Binary Format	Binary Bytes	Binary Offset
1	header	-	-	This field contains the command name or the message header depending on whether the command is abbreviated ASCII, ASCII or binary, respectively.	-	H	0
2	state	ON	1	Enable or disable position averaging. (default = ON)	Enum	4	H
		OFF	0				
3	maxtime	0.01 - 100 hours		Maximum amount of time that positions are to be averaged. Only becomes optional if State = OFF.	Float	4	H+4
4	maxhstd	0 - 100 m		Desired horizontal standard deviation. (default = 0)	Float	4	H+8
5	maxvstd	0 - 100 m		Desired vertical standard deviation. (default = 0)	Float	4	H+12

Tabla 3.10 : Estructura del POSAVE

Ejemplo:

POSAVE 12 2 4

Ya hemos explicado por encima el funcionamiento del comando POSAVE, posteriormente veremos como nuestra aplicación hace uso de este tipo de mensaje para obtener y fijar la posición, de la que será nuestra estación base.

Una vez que se ha transmitido el comando POSAVE con los parámetros deseados, lo siguiente que se hará, será envía un comando Log con la finalidad de obtener los

resultados que se van obteniendo a medida que se realiza el cálculo. Estos mensajes son procesados por el hilo principal y posteriormente presentados por pantalla para facilitar al usuario el seguimiento de la operación. El comando de petición enviado es el siguiente.

LOG AVEPOSA ONCHANGE

Este mensaje solicita al receptor el envío de una trama tipo AVEPOSA cada vez que se hayan producido cambios a lo largo del proceso. Para más detalles sobre este comando se ha introducido la siguiente tabla proporcionada por Novatel.

Field #	Field type	Data Description	Format	Binary Bytes	Binary Offset
1	header	Log header		H	0
2	lat	Average WGS84 latitude (degrees)	Double	8	H
3	lon	Average WGS84 longitude (degrees)	Double	8	H+8
4	ht	Average height above sea level, or geoid (m)	Double	8	H+16
5	lat σ	Estimated average standard deviation of latitude solution element, in meters	Float	4	H+24
6	lon σ	Estimated average standard deviation of longitude solution element, in meters	Float	4	H+28
7	hgt σ	Estimated average standard deviation of height solution element, in meters	Float	4	H+32
8	posave	Position averaging status (see *Table 40*)	Enum	4	H+36
9	ave time	Elapsed time of averaging (s)	Ulong	4	H+40
10	samples	Number of samples in the average	Ulong	4	H+44
11	xxxx	32-bit CRC (ASCII and Binary only)	Hex	4	H+48
12	[CR][LF]	Sentence terminator (ASCII only)	-	-	-

Tabla 3.11 : Estructura del mensaje AVEPOSA

Ejemplo:

#AVEPOSA,COM1,0,65.0,FINESTEERING,1058,272851.000,80000000,3a72,32800;51.11635672780,-114.03824509533,1046.6118,5.0299,3.4216,8.0996, INPROGRESS, 13800,24*a2f7b21b

Atendiendo a la tabla anterior y a la información introducida por el usuario podemos ver que mientas el usuario habla en términos de desviación máxima vertical y horizontal el comando AVEPOS habla en términos de desviación de latitud, longitud y altitud. Según esta falta de coherencia entre unos parámetros y otros, se hace necesario realizar una serie de operaciones bastante sencilla.

Por un lado la desviación vertical no es más que la desviación en altitud, por lo que en lo referente a estos parámetros no será necesaria operación alguna. El problema se presenta cuando hablamos de la desviación horizontal. Esta última hace referencia a la desviación de la posición entrono al plano horizontal, es decir entrono a la superficie de la tierra. Dicho esto, resulta fácil de identificar su relación con respecto a la desviación en latitud y longitud, mediante la siguiente fórmula.

$$\sigma_h = \sqrt{\sigma_{lat}^2 + \sigma_{lon}^2}$$

Una vez calculada la posición cumpliendo los límites deseados, no queda más que fijar la posición de la base, del mismo modo que se hizo anteriormente, es decir, utilizando el comando FIX POSITION.

Cabe mencionar dentro de este apartado un pequeño problema que apareció durante la elaboración de esta parte del código y al que no se le ha dado una explicación lógica. El problema era que al realizar un sondeo con el comando POSAVE, si el GPS no era capaz de alcanzar los límites de desviación máxima en un periodo de tiempo determinado (no siempre superior al límite temporal establecido) se detenía el proceso dando por completada la operación. En un primer momento se pensó que posiblemente era un error del programador, a la hora de interpretar la tabla del comando pertinente. No obstante, tal y como aparecía reflejado el límite temporal en la tabla, lo más intuitivo era pensar que dicho límite iba desde las 0.01 centésimas de hora hasta las 100 horas de espera. De todos modos y en vista a los resultados obtenidos, se pensó que tal vez lo decimales hacían referencia a minutos y no a centésimas de hora.

Partiendo de esta nueva interpretación se puso a sondear al receptor durante diferentes periodos de tiempo y con unos límites de desviación lo suficientemente pequeños como para que nunca los alcanzara. De este modo, siempre que se interrumpiera la operación esta habría sido a causa del límite temporal. Una vez tomadas estas muestras y estudiados los resultados, no se pudo llegar a una conclusión lógica. Las muestras obtenidas no presentaban ninguna correspondencia apreciable con el límite temporal impuesto y ni con el formato con el que estos límites habían sido introducidos. En las siguientes líneas se han detallado algunos de los resultados conseguidos a lo largo de este estudio, con la intención de hacer llegar al lector la incoherencia de los datos adquiridos.

En el primero de los sondeos se limitó el periodo temporal a 0.01 y las desviaciones a 1, por lo que la trama a enviar era la siguiente.

POSAVE 0.01 1 1

Tras varios mensajes aveposa se recibió el mensaje de completado del proceso:

#AVEPOSA,COM2,0,78.5,FINESTEERING,1371,229966.000,80000000,e3b4,1374;37.41199445682,-6.00173360337,18.044560589,6.0855,5.4844,13.6831, COMPLETE,60,2*6d6251df

Como podemos ver, siguiendo la tabla de este comando, el proceso termino a los 60 segundos de duración y sin llegar a los límites establecidos de desviación máxima. Según estos datos, podrían interpretarse los decimales como minutos. Con vistas a comprobar esta conclusión se prosiguió a enviar el siguiente comando.

POSAVE 0.10 1 1

Según lo acordado, el proceso tendría que detenerse a los 10 minutos de duración, es decir a los 600 segundo. Como veremos en la trama siguiente, este razonamiento no era

válido puesto que el proceso se detuvo tras varias tramas anteriores, a los 180 segundos sin que tampoco llegara a alcanzar los límites de desviación.

#AVEPOSA,COM2,0,68.5,FINESTEERING,1371,231667.000,80000000,e3b4,1374;37.41199106350,-6.00173922417,21.386167099,4.1004,4.6278,11.2944,COMPLETE,180,4*a929f03c

Tras estos resultados y con la intención de indagar más en estos extraños comportamientos, se realizaron sondeos con los siguientes límites temporales, 0.1 y 1. En ninguno de ellos se llegó a unos resultados lógicos. A continuación se han presentado las muestras obtenidos en ambos ejercicios.

POSAVE 0.1 1 1

#AVEPOSA,COM2,0,79.5,FINESTEERING,1371,230303.000,80000000,e3b4,1374;37.41199046312,-6.00173035152,20.662570203,4.5330,4.9956,11.8110,COMPLETE,120,3*5145244c

POSAVE 1 1 1

#AVEPOSA,COM2,0,60.0,FINESTEERING,1371,234523.000,80000000,e3b4,1374;37.41198427425,-6.00172860024,19.234985708,0.9901,0.9696,2.3404,COMPLETE,1800,4*2d7a01a8

Podemos apreciar que en el primero de los casos el proceso se para a los 2 minutos de duración y en el segundo a los 30 minutos, por tanto los resultados siguen sin tener un comportamiento lógico. Estos experimentos se realizaron también para otros periodos temporales como 0.2, 0.02, 0.20, 0.3, 0.03, y 0.30 resultado un comportamiento similar a los anteriores pero siempre fuera de toda lógica. No obstante, no se le ha dado mayor importancia a este problema puesto que existen métodos mejores de calcular la posición de nuestra antena. Un ejemplo de ellos sería el que hace uso del comando LOG y que se explicará a continuación.

3.6.2.2 Log

Un método más eficiente para el cálculo de la posición consiste en solicitar al GPS que nos envíe de forma periódica una medida de la posición en la que se encuentra. Una vez recibidas estas muestras, se calcula su valor medio y se toma el resultado obtenido como posición óptima de la estación base. Este algoritmo basa su funcionamiento en el uso del comando Log. Como podemos ver en el Anexo C de esta misma memoria, Log suele ir acompañado de una serie de parámetro que indican, entre muchas otras cosas, el tipo de mensaje que queremos recibir y con que frecuencia queremos recibirlos.

Cuando InterGPS intenta configurar una estación base trabajando con Logs, lo primero que hará será solicitar al usuario que identifique el tipo de mensajes con el que quiere trabajar. Estos mensajes pueden ser de tipo GPGLL, GPGGA, GPGGARTK. En apartados anteriores hemos visto las limitaciones que presentan cada uno de estos mensajes, por lo que el uso de un determinado tipo u otro vendrá limitado por las especificaciones que queramos cumplir. Hay que destacar, que en caso de estar

trabajando con lo equipos antiguos RT-20, no será posible utilizar los de tipo GPGGARTK, puesto que estos equipos no son capaces de procesar estos mensajes.

Una vez que se ha introducido el tipo de mensaje, el usuario deberá facilitar el periodo con el que se quieren recibir las muestras y la duración que tendrá el proceso. Hecho esto el programa enviará al GPS el correspondiente comando Log y dará inicio el proceso. Un posible ejemplo de comando Log sería el siguiente, en el que se solicita que se envíe una trama GPGGA cada 30 segundos por el puerto Com X.

LOG COMx GPGGA ONTIME 30

A medida que van llegando la trama con la posición, el hilo de lectura comprueba que hayan llegado correctamente y las almacena en la pila de tramas recibidas. Del mismo modo que han sido almacenadas, el hilo principal las va extrayendo de la pila para que la información contenida en ellas sea procesada. Una vez transcurrido el tiempo de duración del proceso, la aplicación ofrecerá al usuario la posibilidad de establecer los datos obtenidos como posición de la estación base.

Debido a la naturaleza de nuestro programa, en el que las tramas son borradas una vez que han sido procesadas, era imprescindible que para el cálculo de la media, no fuera necesario mantener almacenada todas las muestras anteriores. A consecuencia de esto se desarrollo un algoritmo distinto a la fórmula tradicional para el cálculo de la media, esto nos permitía obtener dicho valor medio en cada iteración sin necesidad de mantener todas las muestras anteriores almacenadas. La fórmula en la que se basa este algoritmo es:

$$\overline{p}_i = p_i \qquad \text{para } i = 1$$

$$\overline{p}_{i+1} = \overline{p}_i \cdot \frac{i}{i+1} + p_i \cdot \frac{1}{i+1} \qquad \text{para } i>1$$

El índice “i” indica el número de la última muestra recibida, de ahí que cuando i es igual a 1, el valor medio coincide con el valor de “p_i”, donde la variable “p_i” contiene el valor de la posición dado por la muestra número “i”. Según esta fórmula, a medida que nos van llegando tramas, podemos ir calculando el valor medio de la posición sin necesidad de ir almacenando cada uno de sus valores.

Para comprobar la veracidad de esta fórmula vamos a deducir el valor enésimo a partir de la ecuación tradicional del valor medio.

$$\overline{p}_n = \frac{\sum_{i=1}^{n} p_i}{n}$$

$$\overline{p}_n = \frac{\sum_{i=1}^{n-1} p_{i-1}}{n} + \frac{p_i}{n} = \frac{\overline{p}_{n-1} \cdot (n-1)}{n} + \frac{p_i}{n} = \overline{p}_{n-1} \frac{n-1}{n} + p_n \frac{1}{n}$$

Por último decir que una vez calculado el valor medio de la posición, el programa dará al usuario la posibilidad de fijar esta posición como el punto en el que se encuentra la estación base. Para realizar esto el programa seguirá los mismos pasos que se detallaron en el apartado 6.3.1 llamado “Fijar posición”.

3.7 FUNCIONAMIENTO CON OTROS EQUIPOS

Dedicaremos este punto de la memoria a estudiar el rendimiento de nuestra aplicación con ciertos equipos distintos de aquellos para los que ha sido explícitamente creada, que como sabemos eran los receptores Novatel OEM4 RT-2.

Este estudio viene limitado por el número de equipos con los que cuenta el laboratorio, o lo que es lo mismo por la cantidad de modelos de receptores de GPS distintos de los RT-2 con los que se podía trabajar.

En lo referente a los equipos GPSCard RT-20 de este mismo fabricante Novatel, el funcionamiento presentó grandes problemas al principio de su estudio. La aplicación no era capaz de trabajar con esto receptores a pesar de estar utilizando Log y comandos totalmente compatibles con estos aparatos. Para la resolución de este problema se volvió a hacer uso del ya mencionado HyperTerminal de Windows. Gracias a este programa, se fue comprobando las respuestas que presentaba el receptor al enviarle las diferentes tramas, tal y como lo hacia nuestra aplicación. Al hacerlo se comprobó que la respuesta era en todo momento un mensaje de ERROR.

Este error no se debía a que el equipo no fuera capaz de interpretar estos comandos, sino a que en los equipos Novatel RT-20 era necesario enviar en todo momento el puerto Com del receptor GPS, por el que se debía enviar la información. Esto no era así en los nuevos receptores (RT-2) en los que si no se facilitaba esta información, los datos eran enviados por defecto a través del mismo puerto por el que se había recibido la trama de solicitud, de tal modo que en nuestra aplicación nunca se enviaba dicho puerto.

Gracias a la información obtenida mediante el hyperTerminal, se consiguió resolver el problema añadiendo un nuevo parámetro a la hora de configurar el puerto, este parámetro no era otro que el Com por el que el receptor debía enviar la información.

Solventado este problema, se prosiguió con el estudio de estos receptores, resultando óptimo en todo momento salvo en la monitorización de la posición. Como ya se ha comentado, el motivo de este problema era que los equipos Novatel RT-20 no eran capaces de interpretar y generar comandos GPGGARTK, por tanto se abandonó el uso de esta trama y se sustituyo por la GPGGA que si era interpretada por estos receptores.

Terminado el estudio se llego a la siguiente conclusión, el funcionamiento de nuestra aplicación con los receptores de Novatel GPSCard RT-20 era óptimo en todas sus funciones salvo en las referentes a la configuración base usando el comando “Posave” o el comando Log junto con las tramas “GPGGARTK”.