



Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

*Desarrollo de un asistente de voz para la generación de un
Cuaderno de Bitácora digital*

Grado en Ingeniería Mecánica

ALUMNO: Jose Antonio Cienfuegos Ardura

DIRECTORES: José P. González Coma
Milagros Fernández Gavilanes

CURSO ACADÉMICO: 2020-2021

Universida_{de}Vigo



Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

*Desarrollo de un Asistente de voz para la generación de un
Cuaderno de Bitácora digital*

Grado en Ingeniería Mecánica
Intensificación en Tecnología Naval
Cuerpo General

UniversidadeVigo

RESUMEN

Los asistentes de voz, como Alexa o Siri se están popularizando cada vez más. Actualmente, estos asistentes de voz son capaces de reconocer órdenes e interpretar lo escuchado. En este proyecto se plantea la aplicación de estas tecnologías al ámbito de la Armada, con el objetivo de generar un Cuaderno de Bitácora digital que permita facilitar las tareas de redacción y de almacenamiento. Este documento es uno de los más importantes del barco, con usos similares a la caja negra de los aviones. Al tratarse de un libro físico este se puede extraviar, se puede deteriorar o puede contener errores de redacción debidos a confusiones o problemas de comunicación. Mediante la utilización de un asistente de voz, se permitirá almacenar en un documento la información necesaria, sin mediación de un asistente humano, para luego poder ser transferido a una nube de datos, realizar copias con mayor facilidad o tener la opción de modificarlo todas las veces que uno desee. Con este fin se ha desarrollado, mediante el lenguaje de programación Python, una aplicación controlada mediante comandos de voz, que genera de forma automática los documentos que contienen la información relevante referente a la navegación, aliviando así las dificultades previamente descritas.

PALABRAS CLAVE

Reconocedor de voz, Python, Cuaderno de Bitácora, Comandos de voz

AGRADECIMIENTOS

Me gustaría agradecer en primer lugar a mi familia por la infinita paciencia que han tenido conmigo a lo largo de estos años, en segundo lugar, me gustaría agradecer a mis amigos por haberme apoyado durante este largo y duro viaje, en especial a la Camareta 214-B por haberme aguantado durante todas las pruebas que tuve que realizar con el reconocedor, compuesta por los AF Colsa, Conesa y Cuquerella; al estudio 214 compuesto por los AF Fernández, Díaz Amado, Delgado Pérez y Oller sin olvidarme tampoco de los 7 Magníficos, a PCG y a Rakos CDF. También me gustaría dedicarle unas palabras de agradecimiento a mis dos tutores Doña Milagros Fernández Gavilanes y Don José González Coma por haberme guiado a lo largo de este proyecto.

CONTENIDO

Contenido	1
Índice de Figuras	3
Índice de Tablas.....	7
1 Introducción y objetivos	8
1.1 Motivación	8
1.2 Objetivos del trabajo	9
1.3 Organización de la memoria	9
2 Estado del arte	11
2.1 Historia.....	11
2.1.1 Los asistentes de voz	11
2.1.2 Cuaderno de bitácora	13
2.2 Reconocedores de voz.....	18
2.3 Lenguajes de programación	20
2.3.1 Python.....	21
2.4 Entornos de desarrollo	23
2.4.1 Visual Studio Code	23
3 Desarrollo del TFG.....	25
3.1 Desarrollo previo.....	25
3.1.1 Instalación de módulos	26
3.2 Arquitectura del sistema.....	26
3.3 Implementación.....	27
3.3.1 Importación de módulos	28
3.3.2 Reconocedor de voz.....	29
3.3.3 Creación de los documentos	33
3.3.4 Captación de órdenes y relleno de ficheros	36
3.3.5 Solicitud de confirmación.....	38
3.3.6 Guardar documentos	40
3.3.7 Modo de consulta.....	40
3.4 Problemas detectados	42
4 Funcionamiento del sistema y pruebas.....	47
4.1 Manual de usuario	47
4.2 Pruebas	55
5 Conclusiones y líneas futuras	57

5.1 Conclusiones	57
5.2 Líneas futuras	57
6 Bibliografía.....	59
Anexo I: Versión de Google.....	63
Anexo II: Versión de Vosk.....	75

ÍNDICE DE FIGURAS

Figura 2-1: Dr. E. A. Quade mostrando <i>Shoebbox</i> (extraída de [6])	11
Figura 2-2: Distintos tipos de asistentes virtuales (extraída de [12])	12
Figura 2-3: Bitácora (extraído de [15])	13
Figura 2-4: Ejemplo de Cuaderno de Bitácora (extraída de [17])	14
Figura 2-5: Primera página de un Cuaderno de Bitácora	14
Figura 2-6: Apartado de crónica del Cuaderno de Bitácora	15
Figura 2-7: Segunda página de un Cuaderno de Bitácora	15
Figura 2-8: Apartado de acaecimientos del Cuaderno de Bitácora	16
Figura 2-9: Apartado de meteorología del Cuaderno de Bitácora	16
Figura 2-10: Apartado de máquinas del Cuaderno de Bitácora	17
Figura 2-11: Apartado de ronda mayor del Cuaderno de Bitácora	17
Figura 2-12: Uso de lenguajes de programación (extraído de [23])	21
Figura 2-13: Guido Van Rossum (extraído de [26])	21
Figura 2-14: Logotipo de Python (extraído de [27])	22
Figura 2-15: Logotipo Visual Studio Code (extraído de [30])	23
Figura 3-1: Arquitectura del proyecto	26
Figura 3-2: Diagrama de flujo del programa	28
Figura 3-3: Ejemplo de importación de módulos	29
Figura 3-4: Diagrama de flujo del funcionamiento del reconocedor de Google	30
Figura 3-5: Reconocedor de Google	31
Figura 3-6: Diagrama de flujo del funcionamiento del reconocedor de Vosk	32
Figura 3-7: Estructura para el funcionamiento del reconocedor de Vosk	33
Figura 3-8: Proceso de transcripción en el reconocedor de Vosk	33
Figura 3-9: Formato de nombre de los documentos	34
Figura 3-10: Creación documento txt	34
Figura 3-11: Comandos para la creación de la primera hoja	34
Figura 3-12: Comandos para la creación de la segunda hoja	35
Figura 3-13: Comandos para la creación de la tercera hoja	35
Figura 3-14: Comandos para la creación de la cuarta hoja	35
Figura 3-15: Aplicación de formatos al Excel	36
Figura 3-16: Parte de la estructura central del código	37
Figura 3-17: Ejemplo del contenido de la instrucción <code>if</code>	38
Figura 3-18: Contenido de la instrucción <code>if</code> 'crónica'	38

Figura 3-19: Estructura de la función 'respuesta'.....	39
Figura 3-20: Contenido de la función 'arrancar confirmación'.....	39
Figura 3-21: Contenido de la función 'parar confirmación y continuar'.....	39
Figura 3-22: Ejemplo de una instrucción de confirmación en el reconocedor de Vosk.....	40
Figura 3-23: Contenido de la función 'guardar'	40
Figura 3-24: Implementación de la función 'consulta' con el reconocedor de Google.....	41
Figura 3-25: Implementación de la función 'consulta' con el reconocedor de Vosk.....	41
Figura 3-26: Implementación de las instrucciones 'try' y 'except' y la función 'to_number'	42
Figura 3-27:Contenido de la instrucción 'latitud'	43
Figura 3-28: Función de identificación de número en español	44
Figura 3-29: Función de adaptación de números a español	44
Figura 3-30: Conversor para la función 'mar'	45
Figura 3-31: Conversión para la función 'crónica'	45
Figura 3-32: Implementación de la variable 'salir'	46
Figura 3-33: Función 'guardar' para la versión de Vosk.....	46
Figura 4-1: Inicio de programa.....	49
Figura 4-2: Solicitud de confirmación.....	49
Figura 4-3: Caso de confirmación positiva	50
Figura 4-4: Caso de confirmación negativa.....	50
Figura 4-5: Ejemplo de escritura de la crónica.....	51
Figura 4-6: Guardado de documentos	51
Figura 4-7: Ejemplo del modo consulta	52
Figura 4-8: Resultados finales del código	52
Figura 4-9: Hoja de crónica del documento Excel	53
Figura 4-10: Hoja de meteorología del documento Excel.....	53
Figura 4-11: Hoja de máquinas del documento Excel	54
Figura 4-12: Hoja de ronda mayor del documento Excel.....	54
Figura 4-13: Ejemplo de documento de texto	55
Figura 4-14: Comprobación del funcionamiento de Vosk	55
Figura 4-15: Función 'if' de 'la actitud'	56
Figura A1-1: Versión del código con reconocedor de Google 1/23.....	63
Figura A1-2:Versión del código con reconocedor de Google 2/23.....	63
Figura A1-3: Versión del código con reconocedor de Google 3/23.....	64
Figura A1-4: Versión del código con reconocedor de Google 4/23.....	64
Figura A1-5: Versión del código con reconocedor de Google 5/23.....	65
Figura A1-6: Versión del código con reconocedor de Google 6/23.....	65

Figura A1-7: Versión del código con reconocedor de Google 7/23.....	66
Figura A1-8: Versión del código con reconocedor de Google 8/23.....	66
Figura A1-9: Versión del código con reconocedor de Google 9/23.....	67
Figura A1-10: Versión del código con reconocedor de Google 10/23.....	67
Figura A1-11: Versión del código con reconocedor de Google 11/23.....	68
Figura A1-12: Versión del código con reconocedor de Google 12/23.....	68
Figura A1-13: Versión del código con reconocedor de Google 13/23.....	69
Figura A1-14: Versión del código con reconocedor de Google 14/23.....	69
Figura A1-15: Versión del código con reconocedor de Google 15/23.....	70
Figura A1-16: Versión del código con reconocedor de Google 16/23.....	70
Figura A1-17: Versión del código con reconocedor de Google 17/23.....	71
Figura A1-18: Versión del código con reconocedor de Google 18/23.....	71
Figura A1-19: Versión del código con reconocedor de Google 19/23.....	72
Figura A1-20: Versión del código con reconocedor de Google 20/23.....	72
Figura A1-21: Versión del código con reconocedor de Google 21/23.....	73
Figura A1-22: Versión del código con reconocedor de Google 22/23.....	73
Figura A1-23: Versión del código con reconocedor de Google 23/23.....	74
Figura A2-1: Versión del código con reconocedor de Vosk 1/39.....	75
Figura A2-2: Versión del código con reconocedor de Vosk 2/39.....	75
Figura A2-3: Versión del código con reconocedor de Vosk 3/39.....	76
Figura A2-4: Versión del código con reconocedor de Vosk 4/39.....	76
Figura A2-5: Versión del código con reconocedor de Vosk 5/39.....	77
Figura A2-6: Versión del código con reconocedor de Vosk 6/39.....	77
Figura A2-7: Versión del código con reconocedor de Vosk 7/39.....	78
Figura A2-8: Versión del código con reconocedor de Vosk 8/39.....	78
Figura A2-9: Versión del código con reconocedor de Vosk 9/39.....	79
Figura A2-10: Versión del código con reconocedor de Vosk 10/39.....	79
Figura A2-11: Versión del código con reconocedor de Vosk 11/39.....	80
Figura A2-12: Versión del código con reconocedor de Vosk 12/39.....	80
Figura A2-13: Versión del código con reconocedor de Vosk 13/39.....	81
Figura A2-14: Versión del código con reconocedor de Vosk 14/39.....	81
Figura A2-15: Versión del código con reconocedor de Vosk 15/39.....	82
Figura A2-16: Versión del código con reconocedor de Vosk 16/39.....	82
Figura A2-17: Versión del código con reconocedor de Vosk 17/39.....	83
Figura A2-18: Versión del código con reconocedor de Vosk 18/39.....	83

Figura A2-19: Versión del código con reconocedor de Vosk 19/39	84
Figura A2-20: Versión del código con reconocedor de Vosk 20/39	84
Figura A2-21: Versión del código con reconocedor de Vosk 21/39	85
Figura A2-22: Versión del código con reconocedor de Vosk 22/39	85
Figura A2-23: Versión del código con reconocedor de Vosk 23/39	86
Figura A2-24: Versión del código con reconocedor de Vosk 24/39	86
Figura A2-25: Versión del código con reconocedor de Vosk 25/39	87
Figura A2-26: Versión del código con reconocedor de Vosk 26/39	87
Figura A2-27: Versión del código con reconocedor de Vosk 27/39	88
Figura A2-28: Versión del código con reconocedor de Vosk 28/39	88
Figura A2-29: Versión del código con reconocedor de Vosk 29/39	89
Figura A2-30: Versión del código con reconocedor de Vosk 30/39	89
Figura A2-31: Versión del código con reconocedor de Vosk 31/39	90
Figura A2-32: Versión del código con reconocedor de Vosk 32/39	90
Figura A2-33: Versión del código con reconocedor de Vosk 33/39	91
Figura A2-34: Versión del código con reconocedor de Vosk 34/39	91
Figura A2-35: Versión del código con reconocedor de Vosk 35/39	92
Figura A2-36: Versión del código con reconocedor de Vosk 36/39	92
Figura A2-37: Versión del código con reconocedor de Vosk 39/40	93
Figura A2-38: Versión del código con reconocedor de Vosk 38/39	93
Figura A2-39: Versión del código con reconocedor de Vosk 39/39	94

ÍNDICE DE TABLAS

Tabla 1: Comandos de voz para la hoja de crónica	47
Tabla 2: Comandos de voz para la hoja de meteorología.....	48
Tabla 3: Comandos de voz para la hoja de máquinas	48
Tabla 4: Comandos de voz para la hoja de Ronda mayor	48

1 INTRODUCCIÓN Y OBJETIVOS

1.1 Motivación

Los asistentes de voz son una tecnología que permite a los usuarios realizar cualquier consulta verbalmente y devolverle una respuesta, ya sea buscando la respuesta en Internet o añadiendo un recordatorio al calendario. Forman parte de nuestras vidas desde hace años, siendo el más popular Siri [1], el cual tuvo su debut con su incorporación en el iPhone 4s, hasta el más moderno como Alexa [2], el cual puede controlar varios sistemas y hacer realidad el sueño de tener una casa inteligente.

El propósito de este TFG es poder aplicar las funcionalidades que nos aportan los asistentes de voz al ámbito de la Armada española. Un ejemplo de ello es desarrollar un asistente de voz el cual nos permita redactar un Cuaderno de Bitácora digital sin la necesidad de un escribiente apuntando cada una de las órdenes que se dan en el puente o las condiciones meteorológicas en las que nos encontramos en ese momento.

Con la creación de un Cuaderno de Bitácora digital podríamos prescindir de contar con un cuaderno físico, el cual podría perderse, estropearse; o en una situación complicada, como sería tener que abandonar el buque, evitar que la persona responsable de éste tenga que ir a por él. De este modo podríamos tener almacenada toda la información referente a la ruta del barco en un servidor del barco, algo parecido a las cajas negras de los aviones.

Teniendo esto en cuenta, se ha planteado la creación de un programa realizado mediante un lenguaje de programación escogido, en este caso Python, capaz de reconocer mediante la voz las órdenes proporcionadas por el correspondiente oficial para luego transcribir la información a texto, y de este texto extraer la información relevante antes de volcarlo a un documento Excel o Word mediante la utilización de una serie de comandos de voz. De esta manera, cada vez que se diga la palabra correspondiente, se le indicará al programa en que celda irá esa información dentro del documento Excel, si por el contrario esa información debe incluirse en el documento Word o si se desea guardar todo lo escrito anteriormente.

De esta forma se podrá escribir el Cuaderno de Bitácora a medida que se realiza la guardia sin necesidad de que el oficial encargado de redactarlo emplee más tiempo después de salir de la guardia y sin la necesidad de tener a un cronista apuntando todos los hechos que suceden durante la guardia, pudiendo así dedicar sus esfuerzos en otro puesto de mayor importancia. Como además el programa trabajará mediante comandos de voz, no incluirá información no relevante que pueda captar de una conversación.

1.2 Objetivos del trabajo

Como se ha comentado anteriormente, el principal objetivo de este proyecto es el diseño de una aplicación capaz de redactar un Cuaderno de Bitácora. Esta aplicación funcionará mediante la utilización de comandos de voz. Con el fin de alcanzar el objetivo final se introdujeron pequeños objetivos:

- Desarrollar un asistente de voz capaz de transcribir la información que se le transmite.
- Escritura en un documento de texto de la información transcrita
- Creación de un documento Excel
- Escritura en un documento Excel de la información transcrita
- Asignar casillas específicas a la hora de escribir la información transcrita
- Verificación de que la información captada es correcta
- Realización de pruebas para comprobar la robustez del código ante situaciones inesperadas

Para lograr todos los objetivos anteriormente expuestos será necesario una breve investigación para el correcto empleo del lenguaje de programación a la hora de desarrollar la aplicación, conociendo así las funciones principales del lenguaje, qué hacen y cómo se podrán implementar.

Por otro lado, también será necesario realizar una pequeña investigación acerca de los Cuadernos de Bitácora y en especial del utilizado en la Armada española. Se necesitará esta información para poder tener una referencia del modelo de Cuaderno de Bitácora digital que creará el programa como resultado final.

Con todo esto, y una vez logrados todos los objetivos expuestos anteriormente, se habrá conseguido la creación de un asistente de voz capaz de escribir datos en las celdas especificadas dentro de un documento Excel o en un documento de texto.

De esta manera, tendríamos creado un Cuaderno de Bitácora digital completo suprimiendo así el Cuaderno de Bitácora físico. Con el Cuaderno de Bitácora digital se podrá además acceder a él cada vez que se necesite, realizar copias y modificaciones en caso de que sea necesario.

1.3 Organización de la memoria

La memoria de este TFG se divide en seis apartados, complementados con dos anexos. En el capítulo actual se realiza una breve introducción hacia los asistentes de voz y sus funcionalidades y el modo en el que se podrían llegar a implementar en el ámbito de la Armada española.

En el Capítulo 2, está dedicado a la revisión estado del arte. En este estado del arte se incluirán algunos detalles acerca de la historia de los asistentes de voz, cómo se originaron, cómo fueron evolucionando, y qué asistentes son los más usados hoy en día. A continuación, se introducen algunos datos sobre la historia del Cuaderno de Bitácora, por qué se llama así y qué es lo que realmente contiene en su interior. Tras esto llegará el momento de introducir a los reconocedores de voz que se usarán para realizar este proyecto, junto con el lenguaje de programación y entorno de desarrollo escogidos.

En el Capítulo 3, se procederá a explicar cómo se fue desarrollando el trabajo, incluyendo la explicación de la implementación seguida, qué operaciones previas fueron necesarias realizar, cómo se fue escribiendo el código paso a paso y problemas y soluciones que se fueron tomando hasta crear el código final.

En el capítulo 4 se expondrán las pruebas realizadas para comprobar la solidez de los dos códigos propuestos, comprobando se ejecutan correctamente, su robustez frente a fallos y situaciones inesperadas y que cumplen la funcionalidad de recuperar toda la información relevante. Por otro lado, en el apartado de resultados se adjuntarán las imágenes de los resultados finales y cómo quedaría el documento Excel y como quedaría el documento de texto.

En el Capítulo 5 se realizará una conclusión sobre el trabajo y teniendo en cuenta los objetivos propuestos en el apartado 1.2. También se expondrán posibles líneas futuras que se podrán tomar para mejorar este proyecto.

Por último, en los Anexos, se adjuntará en formato imagen los dos códigos desarrollados a lo largo del proyecto.

2 ESTADO DEL ARTE

2.1 Historia

En este apartado se va a tratar un poco la historia de los asistentes de voz, con qué motivo se crearon, cómo surgió el primer asistente de voz y como han ido evolucionando a lo largo de los años. Del mismo modo se analizará el origen del nombre del Cuaderno de Bitácora, por qué se empezó a llamar así y qué es lo contiene y que se escribe en su interior.

2.1.1 Los asistentes de voz

Los asistentes de voz o asistentes virtuales inteligentes son un agente de software que ayuda a los usuarios facilitando la realización de tareas con la menor comunicación posible [3].

Es decir, un asistente virtual inteligente es un tipo de software que puede realizar tareas e incluso brindar servicios a un usuario basados en datos de entrada, reconocimiento de ubicación y la habilidad de acceder a información de una variedad de recursos en línea [4].

La primera herramienta que permitió llevar a cabo el reconocimiento de la voz fue *IBM Shoebox*, el cual tuvo su primera aparición en la Feria de Mundial de Seattle de 1962. Esta computadora era capaz de diferenciar dieciséis palabras y los números del 0 al 9, tal y como se puede apreciar en la Figura 2-1 [5].



Figura 2-1: Dr. E. A. Quade mostrando *Shoebox* (extraída de [6])

El siguiente acontecimiento en la historia del reconocimiento de voz tuvo lugar en 1970, en la Universidad de Carnegie Mellon con el apoyo del Departamento de Defensa de los Estados Unidos y su agencia DARPA. Consiguieron desarrollar una herramienta, *Harpy*, que llegó a dominar el equivalente de palabras al vocabulario de un niño de 3 años. Diez años después, el mismo grupo científico desarrolló un sistema que no sólo podía analizar palabras individuales sino frases y oraciones completas. De esta forma, los contestadores automáticos se convirtieron en los primeros asistentes virtuales junto al software médico de dictáfonos digitales [5].

El Asistente Personal Inteligente comenzó con un proyecto militar de inteligencia artificial al que le pusieron el nombre de CALO ‘*Cognitive Assitant that Learns and Organizes*’, con sede en el instituto de investigación de SRI International, una empresa estadounidense de investigación sin fines lucrativos. Los encargados del proyecto combinaron diversos enfoques, originalmente aislados, de la inteligencia artificial para intentar crear un programa de asistente personal que mejore al interactuar con su usuario. Este asistente de voz se acabaría llamando Siri [7].

La creación de los asistentes de voz, vino acompañado con su implementación en múltiples aspectos de nuestra vida cotidiana, siendo la más significativa la creación de los teléfonos inteligentes. El primer teléfono inteligente desarrollado, *IBM Simon*, fue lanzado en 1994 y sentó las bases de los asistentes virtuales inteligentes tal y como se conocen hoy en día. El primer asistente virtual que se instaló en un teléfono inteligente fue Siri, el cual se presentó como una característica especial del iPhone 4S el día 14 de octubre de 2011 [5].

Siri es capaz de ayudar a los usuarios a establecer recordatorios, alarmas o citas en el calendario, dar información actual y futura sobre el tiempo en la ciudad o abrir el navegador de para encontrar una respuesta a la pregunta realizada [8].

Meses más tarde del lanzamiento de Siri, Google sacó su propio asistente personal inteligente llamado Google Now [9], el cual puede ofrecer búsquedas por voz. Sin embargo, no participaba en conversaciones bidireccionales, por lo que finalmente en mayo de 2016 presentó Google Assistant [10].

En abril de 2013, Microsoft presentó por su parte a Cortana [11], que se lanzó en el mes de enero de 2015. Según Microsoft, Cortana es un asistente digital personal diseñado para ayudar a los usuarios a llevar a cabo tareas básicas y para poder responder a sus preguntas. Funciona en dispositivos con sistema operativo de Microsoft, pero es más popular en ordenadores.



Figura 2-2: Distintos tipos de asistentes virtuales (extraída de [12])

Finalmente, la última empresa en unirse al mercado de los asistentes virtuales fue Amazon, el cual presentó Alexa [2] y Amazon Echo [13], lanzándolos en 2015 en Estados Unidos y más tarde en el mercado europeo. Una característica especial de Alexa es que, a diferencia de los asistentes virtuales existentes hasta la época, Alexa era capaz de vincularse a un dispositivo independiente en lugar de integrarse en otros dispositivos electrónicos.

En la Figura 2-2 podemos observar los tres asistentes virtuales más famosos en la actualidad integrados en un dispositivo diseñado especialmente para ellos. Estos dispositivos requieren de conexión a Internet para poder trabajar correctamente y poder ayudar a los usuarios a resolver dudas. Hay que destacar que hay pequeñas diferencias entre los 3 asistentes. La primera, y la más evidente, es que el asistente de Siri solo puede trabajar en dispositivos de Apple. Por lo tanto, no podrá trabajar con ningún dispositivo que cuente con Android o Windows. Sin embargo, tanto Alexa como Google Assistant funcionan en cualquier dispositivo, ya sea Android, Apple o Windows. Por último, la principal diferencia entre Google Assistant y Alexa o Siri, es que el asistente de Google permite tener una conversación con el usuario, es decir, permite que el micrófono se vuelva a activar tras la respuesta del asistente sin necesidad de ejecutar el comando de voz inicial.

2.1.2 Cuaderno de bitácora

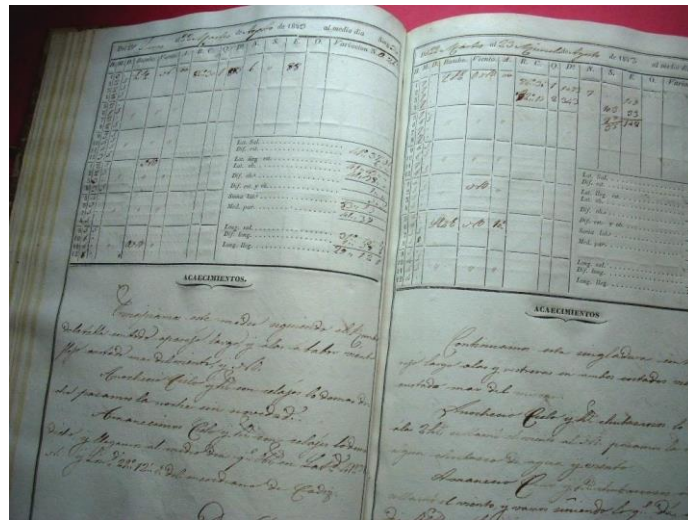
La bitácora (Figura 2-3) es un tipo de armario que se utiliza a bordo de los barcos. Es un instrumento que se encuentra en la cubierta, al lado del timón y de la aguja náutica y que facilita la navegación. Antiguamente, este dispositivo llevaba en su interior un cuaderno, el cuaderno de bitácora, en el cual se relataban el transcurso de los viajes para dejar constancia de todo lo ocurrido en el mismo y de los diversos problemas ocurridos junto con sus soluciones [14].



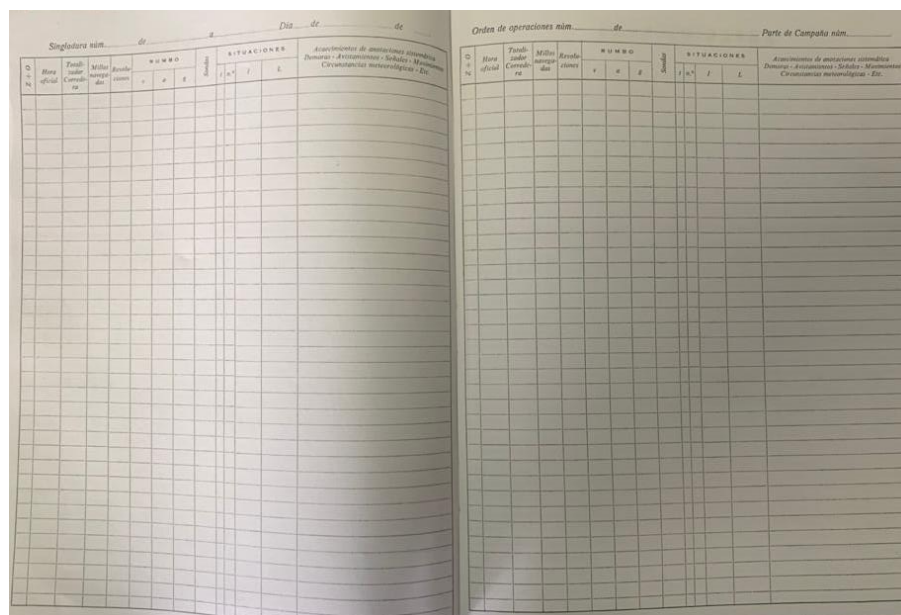
Figura 2-3: Bitácora (extraído de [15])

Este cuaderno se guardaba en la bitácora para así ser protegido de tormentas debido a que servía como libro de consulta ante las variaciones del viaje. Con el paso del tiempo, el término de bitácora pasó a asociarse únicamente al cuaderno, siendo en la actualidad un cuaderno o publicación que permite llevar un registro escrito [16].

En el Cuaderno de Bitácora, los pilotos anotan el estado de la atmósfera, los vientos que reinan, los rumbos que se hacen, la velocidad a la que se navega junto con el número de revoluciones del motor o el aparejo que se lleva en caso de tratarse de un buque de vela, distancia navegada y acaecimientos de importancia que ocurran durante la navegación, tal y como se puede apreciar en la Figura 2-4. Tiene una organización cronológica, facilitando así la revisión de los distintos acontecimientos.



Actualmente, el Cuaderno de Bitácora que se dispone en la Armada, se trata de un Cuaderno en el que se pueden diferenciar varios apartados, contando con un total de 4 páginas por cada día de navegación. Lo primero que encontramos al abrir una página de un Cuaderno de Bitácora, tal y como se observa en la Figura 2-5, es una tabla en la que aparecen distintas columnas, tal y como se ve en la Figura 2-6. Esta parte se hace relación a la crónica, en la que se detallan los acontecimientos más importantes transcurridos durante la guardia, los cuales se cubren en el apartado de acaecimientos, teniendo que indicar a qué hora se produjo, cuántas millas se habían recorrido, qué rumbo se llevaba y dónde nos encontrábamos cuando pasó.



14

Figura 2-6: Apartado de crónica del Cuaderno de Bitácora

Cuando pasamos la página, Figura 2-7, lo que nos encontramos es un apartado de acaecimientos, en el cual se escribe un breve resumen de todo lo acontecido durante la guardia. En este apartado no hay columnas, sino que directamente hay líneas sobre las que se escriben todo lo acontecido como se puede ver en la Figura 2-8.

En la página de al lado seguimos teniendo algunas líneas más y justo debajo de estas líneas se encuentra una pequeña tabla con 3 partes diferenciadas. Una de ellas se trata acerca de las observaciones meteorológicas, Figura 2-9, en la que se escribe estas condiciones (viento, estado de la mar, visibilidad, temperaturas y nubes) acontecidas cada 4 horas.

Figura 2-7: Segunda página de un Cuaderno de Bitácora

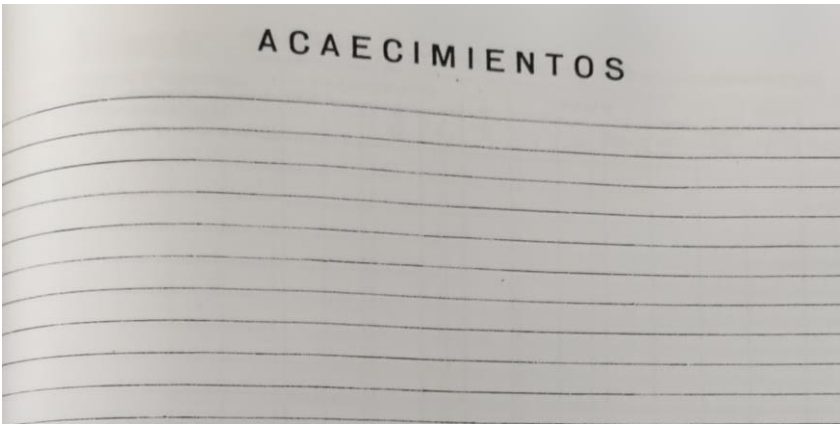


Figura 2-8: Apartado de acaecimientos del Cuaderno de Bitácora

OBSERVACIONES METEOROLOGICAS										
Hora T. M. C.	Dirección del viento en decenas de grados	Velocidad del viento. Nudos	Visibilidad	Tiempo presente	Barómetro en mm	PSICROMETRO		NUBES		Estado de la mar
						Bola seca	Bola húmeda	Cantidad	Clase	
1										
2										
3										
4										
5										
VI										
7										
8										
9										
10										
11										
XII										
13										
14										
15										
16										
17										
XVIII										
19										
20										
21										

Figura 2-9: Apartado de meteorología del Cuaderno de Bitácora

Al lado de esta tabla se encuentra la parte correspondiente a máquinas, Figura 2-10, en la que vendrá reflejado las revoluciones de los ejes de babor y estribor (o de un único eje en caso de no disponer ambos) y los consumos y existencias de gasolina, aceite y agua. Al igual que las observaciones meteorológicas, esto se rellenará cada 4 horas.

[illegible]

Figura 2-10: Apartado de máquinas del Cuaderno de Bitácora

Debajo de esta sección se encuentra el apartado de temperaturas de pañoles, el cual se corresponde con la ronda mayor, Figura 2-11. La ronda mayor consiste en una ronda que se da alrededor del barco comprobando los compartimentos y/o pañoles que contengan materiales o sustancias que puedan producir una explosión. Los más importantes son el pañol de pinturas, el pañol de pirotecnia y el pañol de munición. Para eso se lleva un registro diario de las temperaturas máximas, temperaturas mínimas y humedad relativa que hay en estos compartimentos.

TEMPERATURAS EN PAÑÓLES						
Número	HORAS					
	0400	0800	1200	1600	2000	2400
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						

Figura 2-11: Apartado de ronda mayor del Cuaderno de Bitácora

De todas las secciones comentadas anteriormente, el apartado de crónica, meteorología, máquinas y ronda mayor, se escribirán un documento en formato Excel y la sección de acaecimientos se redactará en un documento de texto.

2.2 Reconocedores de voz

Los reconocedores de voz son una parte de la inteligencia artificial que tiene como objetivo permitir la comunicación hablada entre el usuario y el ordenador. El problema en un sistema de este tipo es el combinar un conjunto de informaciones procedentes de diferentes fuentes en presencia de ambigüedades, incertidumbres y errores inevitables para poder llegar a una interpretación aceptable del mensaje [18].

Un sistema de reconocimiento de voz es una herramienta computacional capaz de procesar la señal de voz emitida por el ser humano y reconocer la información contenida en esta, convirtiéndola en texto o emitiendo órdenes que actúan sobre un proceso.

Un aspecto crucial en el diseño de un sistema de este tipo es la elección del tipo de aprendizaje que se utilice para construir las diversas fuentes de conocimiento. Existen 2 tipos de aprendizaje:

- Aprendizaje deductivo: Estas técnicas se basan en la transferencia de los conocimientos que un experto humano posee a un sistema informático.
- Aprendizaje inductivo: Estas técnicas se basan en que el sistema pueda conseguir de manera automática los conocimientos necesarios a partir de ejemplos reales sobre las tareas que se desea modelizar.

En la práctica, no existen técnicas que se basen únicamente en el aprendizaje inductivo, de hecho, se asume un compromiso deductivo-inductivo.

Otra parte importante de un sistema de reconocedor de voz es el decodificador acústico-fonético el cual es un módulo formado por las fuentes de información acústica, fonética y fonológica y por los correspondientes procedimientos interpretativos. La entrada al decodificador es la señal de voz y para ello es necesario que sufra un proceso de división para poder captar cada palabra de forma individual. En esta etapa es necesario asumir algún modelo físico, contándose con modelos auditivos y modelos articulatorios.

Y, por último, la última parte necesaria para un sistema de reconocimiento de voz es el modelo del lenguaje, en el cual van incluidas las fuentes de conocimiento sintáctico, semántico y pragmático. Los modelos de lenguaje más complejos necesitan grandes corpus de voz, los cuales son un conjunto amplio y estructurado de ejemplos reales de uso de la lengua, texto escrito para el aprendizaje y la evaluación de los correspondientes sistemas.

Algunos de los reconocedores de voz que se usarán durante este proyecto son:

- Vosk: es un reconocedor de voz *offline* de código abierto y permite modelos de reconocimiento de voz para 17 idiomas incluido el español. Este reconocedor funciona con Kaldi Recognizer, el cual se compone de un kit de herramientas para el reconocimiento de la voz y que se encuentra escrito en lenguaje C++. Sus modelos de idioma se diferencian entre pequeños y grandes, aunque rara vez superan los 50 Mb, y proporcionan una transcripción continua de vocabulario grande. Ofrece reconocimiento de voz para asistentes virtuales y puede crear subtítulos para las películas o transcripciones para conferencias y entrevistas. Debido a que Vosk se encuentra aún en desarrollo puede haber algunos problemas que puedan conducir a una mala precisión. La mejor solución para estas imprecisiones es adaptar el modelo existente, ya que los desarrolladores de Vosk se centran en modelos que funcionan en cualquier condición y dedican mucho tiempo a entrenarlos. Cuenta con cuatro niveles de adaptación que se pueden aplicar:

- Actualización del vocabulario del reconocedor en tiempo de ejecución
- Actualización del modelo de idioma: El modelo usado en Vosk es Kaldi, que se compila a partir de tres fuentes de datos, que son diccionario, modelo acústico y modelo de lenguaje. De esta forma se puede reconstruir los tres, pero a veces solo se necesitará ajustar la probabilidad de las palabras para mejorar el reconocimiento.
- Actualización de palabras y vocabulario en modelos grandes: Algunos idiomas (inglés, ruso, alemán) cuentan con modelos más grandes que el que tienen en español, y también se pueden reconstruir sin ningún problema.
- Adaptación del modelo acústico con ajuste fino: También cabe la posibilidad de adaptar el modelo acústico mediante la recopilación los datos y transformándolos en formato Kaldi [19].
- CMU Sphinx: es un reconocedor de voz *offline* de código abierto desarrollado en la universidad de Carnegie Mellon. Este reconocedor de voz también tiene modelos para el español. Incluye programas para reconocimiento de la voz y un entrenador de modelo acústico. Sin embargo, este reconocedor quedó obsoleto debido a que el equipo dejó de sacar actualizaciones para CMU Sphinx. Por el contrario, se involucraron en crear nuevos modelos y aplicaciones para mostrar la mejor manera de implementar el sistema de reconocimiento de voz, uniéndose así a Kaldi y abandonando CMU Sphinx [20].
- Google: es un reconocedor de voz de código abierto que funciona *online*, pero que también tiene una variante para trabajar de manera *offline* llamada Google Cloud, para la cual se necesita una clave API para poder utilizarlo. Para conseguir la transcripción, el reconocedor de Google aplica los algoritmos de aprendizaje profundo más avanzados para reconocer la voz automáticamente. Entre las principales características que posee hay que destacar:
 - Adaptación de voz: Permite personalizar funciones de reconocimiento de voz y poder transcribir términos de un dominio, palabras poco comunes, mejora la precisión con la que se transcriben frases específicas y transformar los números dichos en años, direcciones...
 - Modelos específicos para modelos entrenados: Estos modelos se encuentran optimizados con el objetivo de cumplir los requisitos de calidad de dominios específicos.
 - Reconocimiento de voz en *streaming*: Al igual que el reconocedor de Vosk y Sphinx, el reconocedor de Google también es capaz de recibir los resultados del reconocimiento en tiempo real procesando el audio captado por el micrófono o enviadas desde un archivo de audio grabado con anterioridad [21].

2.3 Lenguajes de programación

Un lenguaje de programación es un lenguaje formal que posibilita a un programador escribir un conjunto de órdenes, acciones y datos para crear programas que monitoricen el comportamiento de una máquina [22].

Gracias a este lenguaje se hace posible la existencia de una comunicación entre el programador y la máquina, de tal forma que se puede especificar, de una forma mucho más precisa, algunos aspectos como: con qué datos debe operar, cómo se almacenan y transmiten los datos e incluso las acciones que debe realizar según se cumplan unos requisitos u otros.

Es decir, es un sistema estructurado de comunicación formado por un conjunto de palabras clave que permiten la comprensión entre un programador y una máquina. No se debe confundir con el lenguaje informático, ya que este comprende otros lenguajes que dan formato a un texto, pero no son de programación en sí mismos.

El lenguaje de programación es la base para construir las aplicaciones digitales que se utilizan día a día y se pueden organizar en dos clases:

- Lenguaje de programación de bajo nivel: Son lenguajes orientados a la máquina. Sirven de interfaz y crean un vínculo entre hardware y software y ejercen un control directo sobre el equipo y su estructura física. Para aplicarlo es necesario conocer de una manera muy específica el hardware.
- Lenguaje de programación de alto nivel: Tiene como objetivo facilitar la comprensión de las instrucciones al programador. Además, permite escribir códigos en idiomas conocidos y luego se traduce al lenguaje máquina mediante traductores o compiladores.

Para realizar el programa se empleará un lenguaje de programación de alto nivel. Además, hay distintas herramientas que le permiten al programador crear, escribir códigos y depurar los proyectos. Algunas de estas herramientas son:

- Editores de texto: Autocompletan el código y marcan errores sintácticos.
- Compiladores: Traducen el código a lenguaje máquina.
- Depuradores: Optimizan el tiempo de desarrollo mediante la corrección de errores.
- Interpretadores o traductores: Carga el código y traduce las instrucciones para ejecutar el programa.
- Entorno de Desarrollo Integrado o IDE (*Integrated Development Environment*): Aplicación informática que proporciona una serie de servicios que facilitan la programación de software.

En la actualidad existen multitud de lenguajes de programación, pero hay 3 lenguajes en concreto que se usan más que el resto y que destacan por las funcionalidades que ofrecen. Estos 3 lenguajes son: Java, C y Python. En la Figura 2-12 podemos apreciar un top en el que se destaca la actividad segmentada por lenguajes de programación desde 2014 hasta 2020.

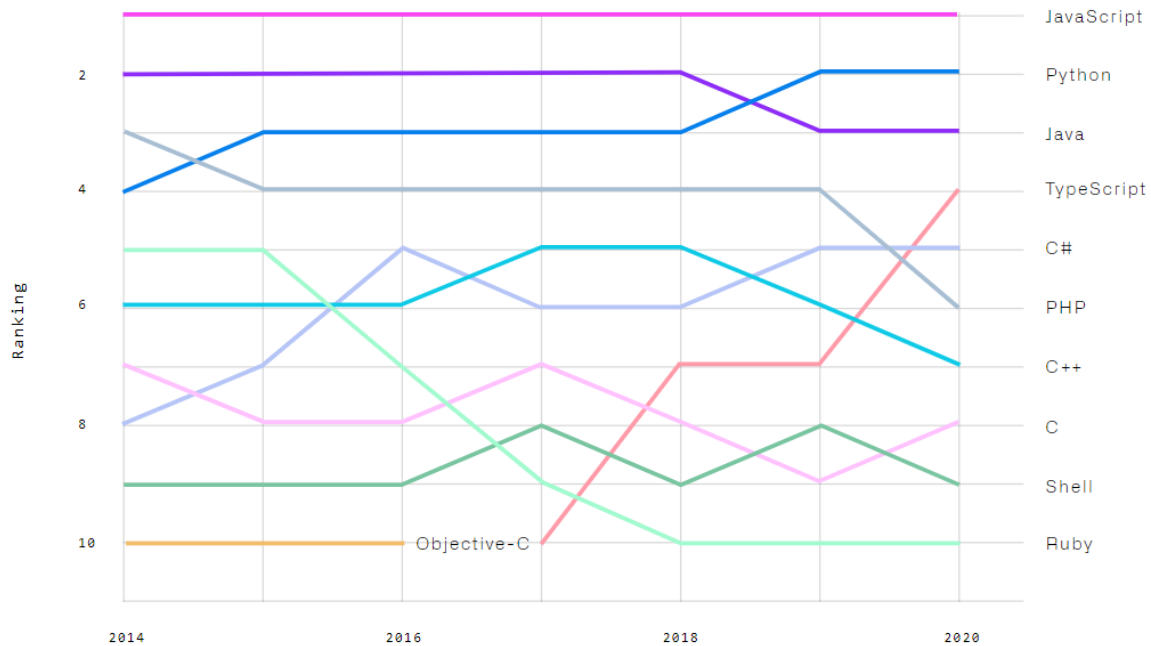


Figura 2-12: Uso de lenguajes de programación (extraído de [23])

Tal y como se puede apreciar, el lenguaje más usado desde 2014 hasta 2020 fue JavaScript, sin embargo, también se puede apreciar que Python, un lenguaje creado para programadores inexpertos, tuvo un auge en los últimos años llegando incluso a superar a la versión original de Java en cuanto a número de programadores por los que es usado.

2.3.1 Python

Python se ha convertido actualmente en el segundo lenguaje de programación más utilizado en el mundo por detrás del lenguaje de programación 'JavaScript'.

El creador de Python es Guido Van Rossum, a quien se puede ver en la Figura 2-13, un informático de origen holandés que fue encargado de diseñar Python y de pensar y definir todas las vías posibles de evolución. Guido le puso el nombre de Python a este lenguaje de programación en honor a los Monty Python, un famoso grupo de cómicos británicos [24] [25].



Figura 2-13: Guido Van Rossum (extraído de [26])

En las navidades de 1989 Van Rossum decidió empezar un nuevo proyecto como pasatiempo personal. Este pasatiempo consistió en darle continuidad a ABC, un lenguaje que se desarrolló a principios de 1980 como alternativa a Basic. Se trataba de un lenguaje pensado para programadores inexpertos por su facilidad de aprendizaje y uso con un código compacto pero legible.

Sin embargo, este proyecto no consiguió llegar lejos debido a las limitaciones hardware de la época, así que Van Rossum decidió darle una segunda vida a su idea y partiendo de la base que tenía, empezó a trabajar en Python (Figura 2-14). La primera implementación de Python fue en febrero de 1991 con la versión 0.9.0.

Esta primera versión incluyó clases con herencias, manejo de excepciones, funciones y una de las características fundamentales: funcionamiento modular. Esta función permitía que fuese un lenguaje mucho más limpio y accesible para la gente con pocos conocimientos de programación, característica que se mantiene hasta el día de hoy.

Van Rossum dirigió el desarrollo de este popular lenguaje hasta el año 2018, que fue cuando decidió apartarse y, desde 2019, quedó a cargo de cinco personas que son las que deciden cómo evoluciona y desarrolla Python.

A día de hoy, Python publicó tres versiones: La versión 1.0 se publicó en enero de 1994, la versión 2.0 se publicó en octubre de 2000 y la versión 3.0 se publicó en diciembre de 2008.

Python empezó a desarrollarse cuando Van Rossum se encontraba trabajando en CWI (Centro de investigación Wiskunde e Informática), un centro de investigación holandés centrado en las áreas de las matemáticas e informática teórica, donde se liberó en 1995 la versión 1.2 de Python. Desde entonces, desvinculado ya de CWI, Van Rossum hizo más accesible el código y para el año 2000 el equipo de desarrolladores de Python cambió a BeOpen Python Labs.



Figura 2-14: Logotipo de Python (extraído de [27])

En octubre del año 2000 se publica la segunda versión de Python. Una nueva versión que incluyó la generación de listas, una característica muy importante del lenguaje. En 2001 se crea la Python Software Foundation, y es a partir de la versión 2.1 cuando se convirtió en dueña de todo el código, documentación y especificaciones del lenguaje.

Esta nueva versión de Python también incluyó un nuevo sistema gracias al cual los programadores eran capaces de hacer referencias cíclicas y, gracias a esto, Python podría recolectar basura dentro del código.

La última gran actualización de la historia de Python se produjo en el año 2008, con el lanzamiento de la versión 3.0, la cual se dedicó a solventar los principales fallos en el diseño de este lenguaje de programación. Python ha ido acumulando formas nuevas y redundantes de programar un mismo elemento, de ahí la necesidad de nuevas versiones que eliminen estos constructores duplicados.

La última actualización de Python fue lanzada en octubre de 2020 con la versión 3.9. que es la versión con la que se trabajará en este proyecto. Por último, se prevé una nueva actualización de Python para el mes de octubre de 2021.

2.4 Entornos de desarrollo

Un entorno de desarrollo es un conjunto de herramientas utilizadas para desarrollar un código o programa. Este término se suele utilizar frecuentemente como un sinónimo de entorno de desarrollo integrado (IDE), que es la herramienta utilizada para escribir, probar y depurar un código. También proporciona a los desarrolladores una interfaz de usuario común para desarrollar y depurar en diferentes modos [28].

Los entornos de desarrollo tienen normalmente tres distintos niveles de servicios, clasificados como desarrollo, montaje y producción.

- Servicio de desarrollo: En este nivel es donde el desarrollador prueba el código y comprueba si la aplicación se ejecuta correctamente sin dar ningún fallo. Una vez que la implementación ha sido probada y el desarrollador considera que el código trabaja correctamente se pasa al siguiente nivel.
- Servicio de integración: En este nivel se prueba la aplicación para comprobar la fiabilidad y para asegurarse de que no falla en el servidor de producción real. Estas pruebas son el último paso antes de que la aplicación se despliegue en el servicio de producción.
- Servicio de producción: Tras la aprobación, la aplicación se convierte en una parte de este servicio.

Además, los entornos de desarrollo deben cumplir algunas características básicas para cumplir las expectativas del usuario. Deben ser multiplataforma y tener una interfaz atractiva y práctica para trabajar, deben facilitar el proceso integral de la programación y es importante que cuenten con un asistente de ayuda y foros donde los usuarios puedan plasmar sus dudas. Algunos de los entornos de desarrollo más usados son Eclipse, NetBeans o Visual Studio Core.

2.4.1 Visual Studio Code

Visual Studio Code (Figura 2-15) es un editor creado por Microsoft para sistemas operativos que usen Windows, Linux o macOS. Incluye soporte para depuración, control de Git integrado, resaltado de sintaxis, finalización de código inteligente, fragmentos de código y refactorización de código. También es personalizable, de modo que los usuarios pueden cambiar el tema del editor, los métodos abreviados de teclado y las preferencias [29].

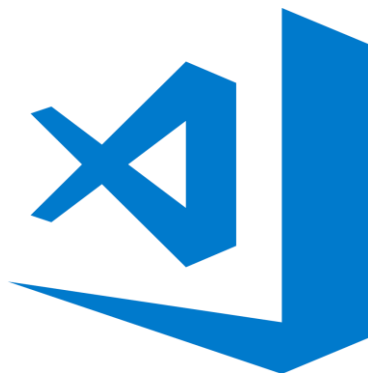


Figura 2-15: Logotipo Visual Studio Code (extraído de [30])

Se anunció el 29 de abril de 2015 en la conferencia Build de 2015 y poco después de esto lanzaron una versión preliminar. El 18 de noviembre de 2015 fue lanzado bajo la licencia MIT y su código fuente se publicó en GitHub. Finalmente, el 14 de abril de 2016 pasó de la etapa de vista previa pública y se lanzó a la web.

El código combina la interfaz de usuario optimizada de un editor moderno con asistencia y navegación de código enriquecido y una experiencia de depuración integrada, sin la necesidad de un IDE completo. Cuenta con herramientas de *Debug* hasta opciones para actualización en tiempo real de nuestro código en la vista del navegador y compilación en vivo de los lenguajes que lo requieran. Además de las extensiones, tendrá la posibilidad de optar por otros temas o bien configurarlo a nuestro gusto.

3 DESARROLLO DEL TFG

En esta parte del trabajo se va a explicar cómo se desarrollaron dos versiones del código, qué pasos se fueron siguiendo para su implementación, algunos problemas que se encontraron durante su desarrollo y por último las características de los reconocedores que se usaron, sus puntos fuertes, sus puntos débiles y las modificaciones que tuvo que sufrir el código para adaptarse a cada reconocedor.

3.1 Desarrollo previo

El lenguaje de programación que se decidió usar para escribir el código fue finalmente Python. Se decidió utilizarlo debido a que es un lenguaje que está en alza, tal y como se detalló en el apartado 2.3. El principal motivo es que es un lenguaje sencillo y en constante evolución, que, desde sus inicios fue enfocado hacia programadores inexpertos que estaban empezando a programar.

Concretamente, la versión de Python que se ha utilizado ha sido la 3.9.2. siendo esta la versión más moderna de Python hasta la fecha. Además, permite la correcta utilización y ejecución de los modelos de idiomas de los módulos de reconocimiento de Google [31] y Vosk [32], que se verán más adelante en este capítulo. Para descargarlo simplemente se accede a la página web de Python [33], dirigirse al apartado de descargas y se selecciona la versión de Python que se desea descargar.

Hay que destacar además que en este apartado de la página de Python aparecen todas las versiones de Python que están activas, junto con las acciones que se están llevando a cabo en esas versiones. En el caso de que se quiera descargar otra versión de Python que no sea la más moderna, existe un apartado que lo permite y que indica las distintas versiones de Python y la fecha en la que se lanzaron.

Por otra parte, el entorno de desarrollo elegido para escribir el código fuente fue Visual Studio Core, debido a que, entre otros factores, Visual Studio Core no se desarrolló de cara a trabajar con un único lenguaje de programación. Este programa se puede descargar desde la referencia [34]. Además, este editor de código fuente tiene varias ventajas, como, por ejemplo:

- Indica errores en el código antes de compilar el mismo.
- Permite ejecutar el programa en la misma aplicación sin necesidad de usar la ventana de símbolo de sistema de Windows.
- Indica los errores que ocurren una vez el código ha sido compilado.

3.1.1 Instalación de módulos

A la hora de comenzar el proyecto, lo primero que se hizo fue investigar que módulos se iban a necesitar y si se iban a tener que instalar algún programa adicional a parte de Python y Visual Studio Core.

Tras la investigación (durante la cual se visitaron las páginas de los enlaces [34], [35], [36], [37] y [38]) se vio que era necesario instalar algunos módulos como PyAudio, SpeechRecognizer o pytttsx3. Para poder instalarlos correctamente, se ejecutó la aplicación de Símbolo de sistema de Windows y se introdujo el siguiente comando:

```
'pip install pyaudio'
```

Mediante el comando 'pip' se indica al sistema que se trata un instalador que se encarga de incorporar módulos a Python, realizándose la misma operación con los módulos SpeechRecognition o pytttsx3.

3.2 Arquitectura del sistema

A continuación, se va a pasar a detallar la arquitectura seguida para el desarrollo de este trabajo. Esta se puede ver en la Figura 3-1. En esta figura se puede apreciar a un usuario que se encuentra hablando a un ordenador transmitiéndole información acerca de la guardia. El ordenador ejecutará la aplicación que captará lo que dice el usuario mediante el empleo de un reconocedor de voz. Dependiendo del reconocedor de voz usado, el programa funcionará de manera *online*, con conexión a Internet o de manera *offline*, sin conexión a Internet. Tras esto, el programa recopilará la información recibida por parte del usuario y creará dos documentos, uno en formato Excel y el otro en formato txt, que se elaborarán a partir de la información dicha por el usuario.

Dentro del documento en formato Excel se encontrarán las partes del Cuaderno de Bitácora relacionadas con la crónica, la meteorología y las máquinas y la ronda mayor. Por otro lado, en el documento en formato de texto se encontrará el apartado referido a los acaecimientos. (Una descripción más detallada de qué son cada una de ellas se encuentra disponible en el apartado 2.1.2).

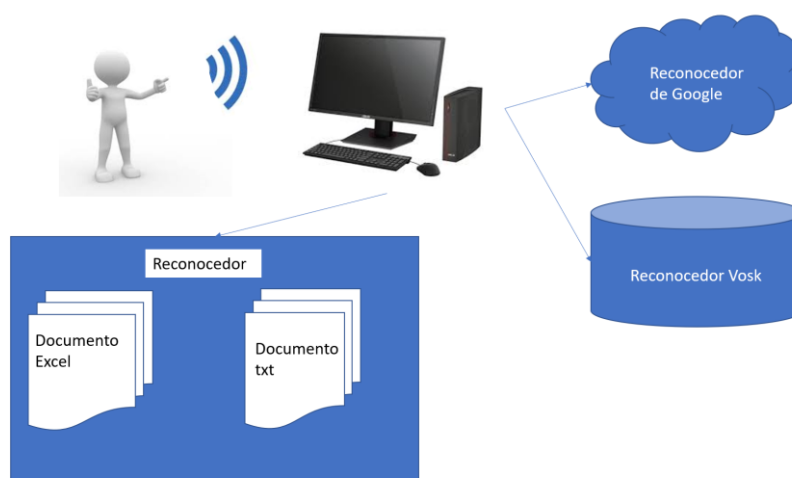


Figura 3-1: Arquitectura del proyecto

3.3 Implementación

Antes de entrar en más detalle con el código desarrollado, es necesario remarcar que se han creado dos implementaciones distintas, una que hace uso del reconocedor de Google y otra que requiere el reconocedor de Vosk. El principal motivo de estos dos desarrollos es debido a que el reconocedor de Google solo puede funcionar siempre y cuando se disponga de conexión a Internet y se tenga la suficiente velocidad de conexión, mientras que el reconocedor de Vosk puede trabajar, aunque se encuentre sin conexión a Internet.

Por otro lado, se decidió usar el reconocedor de Vosk antes que el reconocedor de CMU Sphinx ya que este se encuentra más evolucionado para nuestro idioma, siendo la última versión estable de CMU Sphinx del año 2015. A pesar de que ambos módulos realizan la misma función, su estructura varía. Más adelante se explicará con mayor profundidad cómo funciona cada uno y en qué se diferencian.

En la Figura 3-2 se puede apreciar el diagrama de flujo de la implementación realizada. Como se puede observar en ella, el primer paso va a ser importar los módulos, y tras esto se hará uso de uno de los reconocedores de voz. Después, el código continúa con la creación del documento Excel y el inicio de la detección de los comandos de voz. A partir de este momento, el programa empezará a captar la información proveniente del usuario y determinará las acciones a realizar en función de la información transcrita. Tras esto le pedirá una confirmación por voz al usuario para comprobar la fidelidad de la información recogida. Si esta confirmación es positiva, se ejecutará la función correspondiente al almacenamiento de los datos captados en la posición indicada del archivo de la crónica, pero si es negativa el programa considerará que existe algún error en los datos obtenidos, y le pedirá al usuario que repita la información.

El programa continuará realizando este proceso de forma indefinida, hasta que detecte el comando de voz ‘guardar’ y, una vez confirmada esta nueva orden guardará todo el progreso y cerrará el programa. A continuación, entrará en más detalle siguiendo como referencia los elementos de la Figura 3-2.

Tras haber importado los distintos módulos que se usarán en el código, lo primero que se hizo fue empezar a definir funciones. Esto permite encapsular partes del código para evitar redundancias y errores en él de tal forma que, cuando este código se tenga que repetir en diferentes partes del programa, sea suficiente con escribir el nombre de la función en lugar de escribir todo lo que esta contiene. Para llevar esto a cabo, se utilizó la instrucción ‘def’. La única función que ambos códigos tienen en común es *SpeakText*, la cual permite convertir texto en voz. Más adelante veremos las funciones adicionales que contiene la versión del código que usa el reconocedor de Vosk.

Las funciones relativas a Excel se refieren a funciones creadas con el nombre de la columna correspondiente de tal forma que, por ejemplo, la función ‘Millas totales’ se dedica única y exclusivamente a escribir datos en la columna de Millas totales. Se trata de un total de 30 funciones que van asociadas a las 30 columnas que contiene el Excel y que se dividen en las 4 hojas que este contiene.

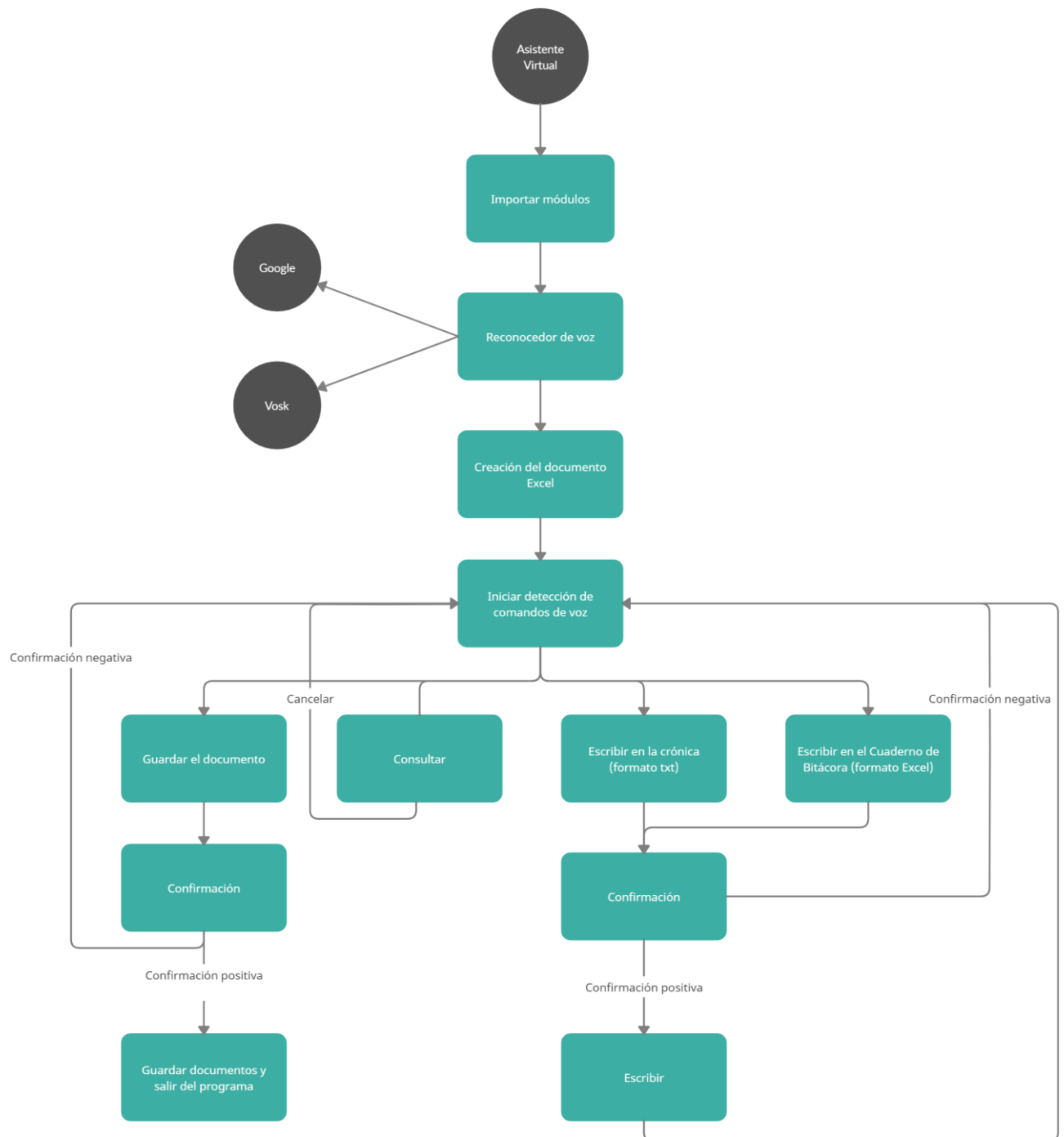


Figura 3-2: Diagrama de flujo del programa

3.3.1 Importación de módulos

Algunos de los módulos usados en este código son:

- `Pytttsx3`: El cual se usa para la conversión de texto a voz.
- `SpeechRecognition`: El cual se usa para la conversión de voz a texto y es la parte más importante del código que estamos realizando, usado para el código que contiene el reconocedor de Google.
- `Vosk`: Del cual se importaron los módulos `Model` y `KaldiRecognizer` para poder construir el reconocedor de voz de `Vosk`.

- `Datetime`: El cual se usa para captar el año, mes, día, hora, minuto y segundo en el que decimos algo.
- `Xlsxwriter`: El cual nos permite poder crear, escribir, modificar y guardar un documento Excel.
- `Os`: El cual nos permite realizar operaciones de sistema operativo como por ejemplo dirigirnos a una determinada carpeta dentro del ordenador o renombrar un archivo.
- `Json`: El cual nos va a permitir cambiar el tipo de dato que nos resulte de la transcripción de voz a texto de *string* a *dictionary*.
- `Spa2num.converter`: El cual nos va a permitir cambiar un texto a un número.
- `Pyaudio`: El cual se utiliza principalmente para reproducir o grabar audio y que será de gran ayuda en el código que utiliza el reconocedor de `Vosk`.
- `Math`: Módulo importado para poder detectar los diferentes errores que podían suceder en el reconocedor de Google.

Tras la importación de los módulos, se investigó cómo programar un receptor de audio en lenguaje Python. Para ello, lo primero que se tuvo que hacer fue incorporar al código los distintos módulos que se habían instalado previamente, y otros módulos que ya venían incluidos en el paquete de instalación de Python mediante la instrucción `'import'`.

Hay que destacar que la utilización de la instrucción `import` permite utilizar el nombre de lo importado de forma acortada, como por ejemplo ocurre con el `SpeechRecognition`. Este módulo se importó y se le puso de nombre `'sr'`, de tal forma que cada vez que se tenga que usar una función del módulo, solo se tendrá que escribir `'sr'` seguido de la función a la que se quiera referir. Se puede observar esta característica en la Figura 3-3.

```
from vosk import Model, KaldiRecognizer
import os
import json
import pyaudio
import xlsxwriter
import shutil
import datetime
import pyttsx3
from spa2num.converter import to_number
```

Figura 3-3: Ejemplo de importación de módulos

Además de esa utilidad, Python permite importar ciertas partes de un módulo sin la necesidad de importarlo al completo, como pasa por ejemplo con el módulo `Vosk`. De este módulo solo necesitamos importar las funciones `Model` y `KaldiRecognizer`, por lo que para ello se utilizará la instrucción `'from Vosk import Model, KaldiRecognizer'`. Se puede observar esta función en la Figura 3-3.

3.3.2 Reconocedor de voz

En este apartado se va a ver en mayor profundidad la estructura que es necesaria llevar a cabo para que los reconocedores de Google y `Vosk` funcionen correctamente y qué pasos seguir para poder implementarlos.

- *Reconocedor de Google*

El reconocedor de voz de Google tiene como principal ventaja que ha experimentado numerosas mejoras a lo largo del tiempo y que, por lo tanto, consta de un modelo de lenguaje con la gran mayoría de las palabras de la lengua española. Sin embargo, su principal desventaja es su necesidad de disponer de una conexión a Internet para que este pueda funcionar correctamente. En la Figura 3-4 se puede observar el funcionamiento de este reconocedor.

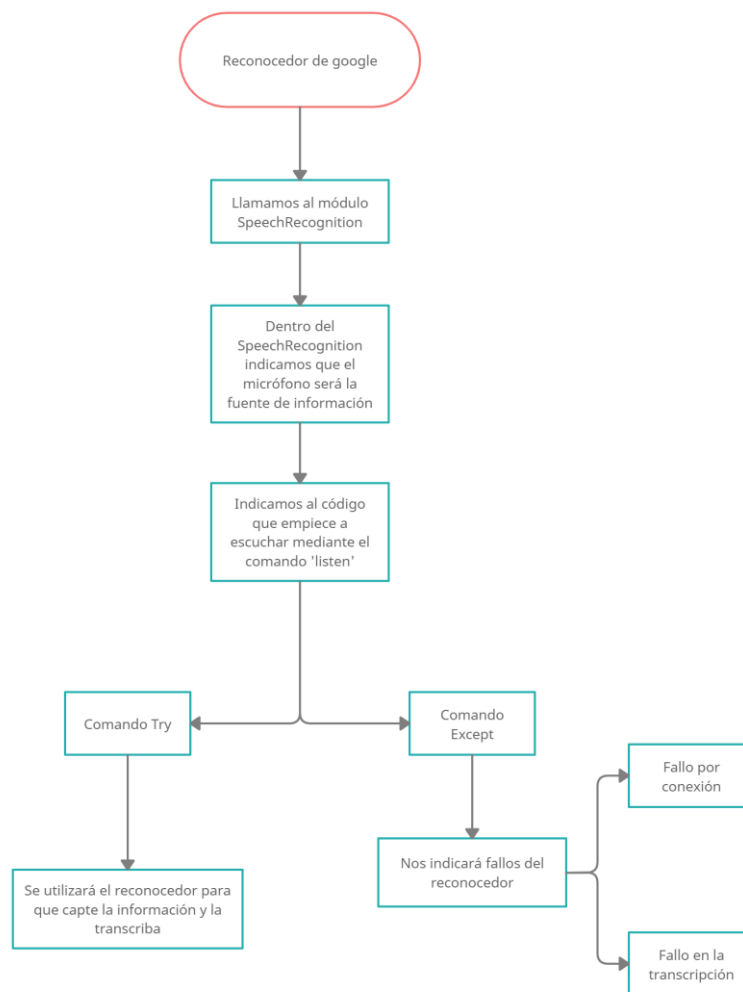


Figura 3-4: Diagrama de flujo del funcionamiento del reconocedor de Google

De entre todas las funciones definidas en el código en el que se usa este reconocedor, la más importante es la de 'takeCommand' ya que es donde se encuentra el núcleo del código, es decir, el código mediante el cual se activa el reconocedor de voz de Google.

Dentro de la función 'takeCommand', lo primero que se hace es definir la variable 'r=sr.Recognizer'. Tras esto se establece la fuente por la cual se captará el audio. En nuestro caso particular será el micrófono del ordenador gracias a sr.Microphone, que asignará como fuentes de datos 'source' con la instrucción 'with sr.Microphone() as source'. Hay que destacar que se puede seleccionar el micro por el cual se va a querer captar la información mediante la instrucción 'device_index='. En caso de no especificarlo se utilizará el que tenga el ordenador o plataforma por defecto.

Una vez definida la fuente por la que obtendrá la información del usuario, se define la información que recogida como 'audio' el cual consiste en mantener en escucha a la fuente seleccionada mediante la instrucción 'r.listen(source)'. Después se usa una instrucción 'try' para ejecutar el reconocedor voz y que, en caso de fallo, ya sea porque no captó bien la información, porque no se procesó bien la información dada o porque en ese momento no hay conexión a Internet, el sistema avise mediante audio de la necesidad de repetir debido al fallo gracias a su inclusión en la instrucción 'except'. Si no hay errores (dentro de la instrucción try) se reconocerá la frase que se pasa de audio a texto mediante la variable 'statement'.

Este proceso de transcripción de voz a texto lo hará el reconocedor usado, indicándole la variable donde se encuentra el audio a procesar y el idioma en el que se encuentra la información proporcionada. En la Figura 3-5 se puede ver la estructura de la función al completo.

```
def takeCommand():
    r=sr.Recognizer()
    with sr.Microphone(device_index=1) as source:
        print("Escuchando")
        audio = r.listen(source)

        try:
            statement=r.recognize_google(audio,language='es-ES')
            print(f"Has dicho:{statement}\n")
        except Exception as e:
            SpeakText("Disculpe, ¿Puede repetir?")
            return "None"
        return statement
```

Figura 3-5: Reconocedor de Google

- *Reconocedor de Vosk*

Vosk es otro reconocedor de voz que, al igual que el reconocedor de Google, está también bastante desarrollado, aunque no posee un modelo de lenguaje tan exhaustivo en lo que a cantidad de palabras se refiere, ni en cuanto a la cantidad de datos que han servido para su creación (en comparación con el de Google). Por otro lado, su principal ventaja es que puede funcionar tanto *online* como *offline* así que el código escrito con este reconocedor puede funcionar en cualquier parte del mundo.

Respecto a Vosk, la forma en la que se configura el reconocedor de voz es bastante diferente al mostrado con Google. En la Figura 3-6 se puede observar el diagrama de flujo de este reconocedor. Para empezar, la primera diferencia entre ambos reconocedores es que en Vosk es necesario importar los módulos Model y KaldiRecognizer. El módulo Model se encarga de buscar en el disco duro del equipo el modelo de idioma en el que se quiere que trabaje el reconocedor, mientras que el módulo KaldiRecognizer coge el modelo que se ha elegido pudiéndole indicar además la frecuencia de muestreo con la que trabajar.

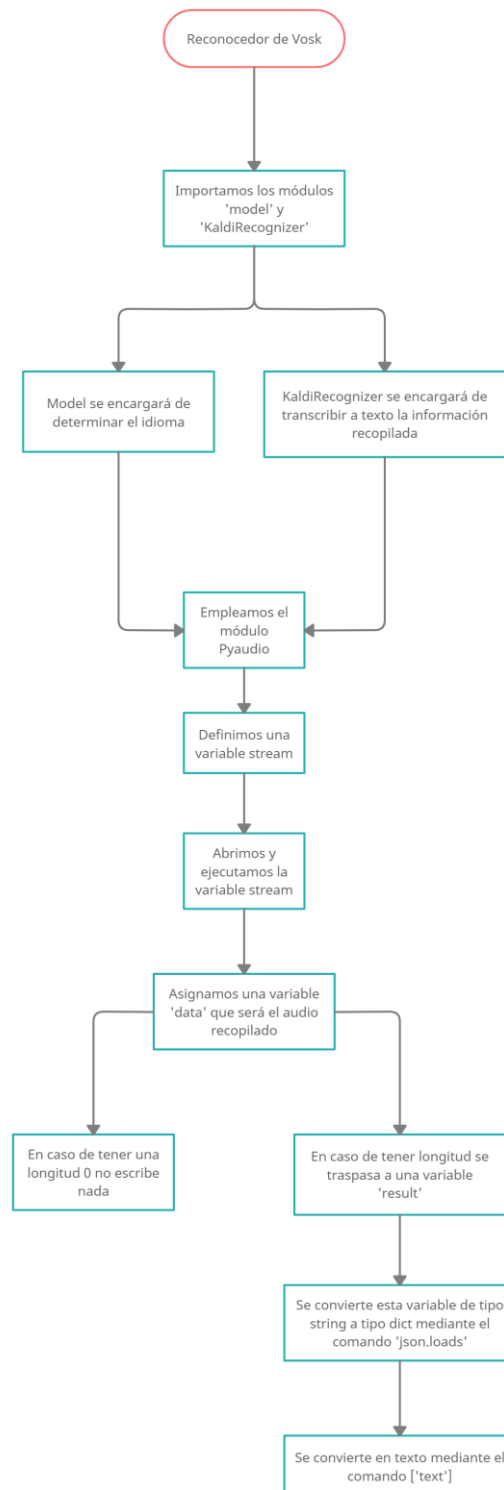


Figura 3-6: Diagrama de flujo del funcionamiento del reconocedor de Vosk

A continuación, se define una variable `p`, que resultará de llamar la función `PyAudio()` del módulo `pyaudio`. Esta permite reproducir o grabar audios para después ser transcritos. Tras esto, se define una variable `stream` que será la encargada de ejecutar la instrucción `p.open` permitiendo seleccionar algunas características antes de empezar a captar audio. A continuación, se empieza a captar audio con la instrucción `start_stream`. En la Figura 3-7 se puede observar todo este proceso.

```
model = Model("model")
rec = KaldiRecognizer(model,16000)

p = pyaudio.PyAudio()
stream = p.open(format=pyaudio.paInt16,channels=1,rate=16000,input=True,frames_per_buffer=8000)
stream.start_stream()
```

Figura 3-7: Estructura para el funcionamiento del reconocedor de Vosk

Ahora, llega el turno de transcribir la información que se está recibiendo, ya que, a diferencia del reconocedor de Google, es necesario realizar unos pasos intermedios para poder transcribir el audio a texto.

Esto se consigue, en primer lugar, con la función ‘stream.read’ al cual se le asigna una variable llamada ‘data’. A continuación, si se consigue captar la información transmitida, se transcribe con la función ‘rec.AcceptWaveform(data)’. El resultado se almacena en una variable cuyo nombre es ‘result’.

El último paso para que nos muestre solo el texto que ha captado por micrófono será cambiar el tipo de dato que nos da por defecto. Se trata de un dato tipo *string*, que se convertirá mediante la instrucción ‘json.loads’ a un dato tipo *dictionary*. Esto se tuvo que hacer debido a que los datos tipo *string* van almacenando todas las palabras que se detectan, imprimiéndolas por pantalla, y después imprime la frase final, sin embargo, los datos tipos *dictionary* muestran solo la frase captada, aunque dividida por palabras. A continuación, con este dato usaremos la instrucción ‘[‘text’]’ para que, finalmente, nos indique el texto transcrito, ya que si no nos aparecerá cada palabra por individual. En la Figura 3-8 se puede observar todo el proceso explicado anteriormente.

```
data = stream.read (4000)
if len(data)==0:
    break
if rec.AcceptWaveform(data):
    result = rec.Result()
    result = json.loads(result)
    result = result['text']
    print (result)
```

Figura 3-8: Proceso de transcripción en el reconocedor de Vosk

3.3.3 Creación de los documentos

Tras haber importado todos los módulos necesarios y tener definidas las funciones, se crean los documentos usados en este trabajo. Se comienza creando el Excel dentro del cual se redacta el Cuaderno de Bitácora, cuyo formato está basado en el Cuaderno de Bitácora usado en la Armada. De aquí se pudo extraer 4 apartados distintos, los cuales se pueden resumir en: crónica, meteorología, máquinas y ronda mayor.

Para la creación de los ficheros se comienza dándoles el nombre a ambos, de acuerdo con el siguiente formato “Cuaderno de bitácora del (día)-(mes) a las (hora) horas”. Para poder sacar el día, el mes y la hora se utilizó el módulo ‘datetime’ y se define una variable ‘fecha’ que contendrá la fecha del momento en el que se crea el documento mediante la función ‘datetime.datetime.now()’. De esta forma, tendremos la estructura de la Figura 3-9 para nombrar a los documentos.

```

fecha = datetime.datetime.now()
ficheroExcel="Cuaderno de bitácora del {}-{} a las {} horas.xlsx".format(fecha.day,fecha.month,fecha.hour)
ficheroTxt="Cuaderno de bitácora del {}-{} a las {} horas.txt".format(fecha.day,fecha.month,fecha.hour)

```

Figura 3-9: Formato de nombre de los documentos

Por otro lado, para crear el documento de texto es necesario darle la función ‘with open’ tal y como se puede ver en la Figura 3-10. Además, se le tiene que indicar que tiene que crearlo en modo ‘a’ para así poder escribir todo lo que se le dice y que no se quede con un único dato.

```

with open (ficheroTxt,mode='a') as file:

```

Figura 3-10: Creación documento txt

Una vez se tiene el nombre del documento, lo que viene a continuación es crear el Excel con la ayuda del módulo ‘xlsxwriter’ mediante la función ‘xlsxwriter.Workbook’. Esta función recibe como argumentos el nombre del documento, almacenado en la variable ‘ficheroExcel’. Al objeto devuelto por esta función le asignaremos la variable ‘workbook’, tal y como se observa en la Figura 3-11.

```

workbook = xlsxwriter.Workbook(ficheroExcel)
worksheet1 = workbook.add_worksheet("cronica")
worksheet1.write(0, 0,"Hora" )
worksheet1.write(0, 1,"Millas totales")
worksheet1.write(0, 2,"Millas navegadas")
worksheet1.write(0, 3,"Revoluciones")
worksheet1.write(0, 4,"Rumbo de aguja" )
worksheet1.write(0, 5,"Rumbo verdadero")
worksheet1.write(0, 6,"Latitud")
worksheet1.write(0, 7,"Longitud")
worksheet1.write(0, 8,"Acaecimientos")
worksheet1.write(0, 9,"Acaecimientos")

```

Figura 3-11: Comandos para la creación de la primera hoja

A continuación, se crean las 4 hojas dentro del documento Excel mediante la función ‘workbook.add_worksheet’. Cada una de estas hojas las designaremos mediante una variable llamada ‘worksheetX’, siendo X la página del cuaderno. De esta forma cada vez que se quiera escribir en la primera página, o lo que es lo mismo, la primera hoja del fichero Excel, se podrá hacer directamente mediante la función ‘worksheet1.write’, junto con la fila, la columna y lo que queramos escribir en esa celda. Este proceso se repetirá 4 veces hasta tener todas las hojas necesarias para completar un cuaderno de bitácora.

Hay que reseñar que en la hoja llamada ‘máquinas’ se añadió una función especial. Esto es debido a que esta hoja se rellena una vez cada cuatro horas, por lo que dependiendo de en la guardia en la que nos encontremos tendremos que poner una hora u otra. En la Figura 3-13 se puede ver cómo se ha realizado la creación de la hoja de máquinas junto con la función comentada anteriormente. Lo que hace esta función es que dependiendo de la hora a la que se cree el documento, pondrá una hora u otra en la columna de horas de la página de máquinas.

Estas hojas serán: crónica, meteorología, máquinas y ronda mayor. En la Figura 3-11, Figura 3-12 y Figura 3-14 se puede observar la creación de las otras tres hojas.

```
worksheet2 = workbook.add_worksheet("meteo")
worksheet2.write(0, 0, "Hora" )
worksheet2.write(0, 1, "Dirección del viento")
worksheet2.write(0, 2, "Velocidad del viento")
worksheet2.write(0, 3, "Visibilidad")
worksheet2.write(0, 4, "Barómetro" )
worksheet2.write(0, 5, "Bola seca")
worksheet2.write(0, 6, "Bola húmeda")
worksheet2.write(0, 7, "Cantidad de nubes")
worksheet2.write(0, 8, "Clase de nubes")
worksheet2.write(0, 9, "Estado de la mar")
worksheet2.write(0, 10, "Estado de la mar")
```

Figura 3-12: Comandos para la creación de la segunda hoja

```
worksheet3 = workbook.add_worksheet("máquinas")
worksheet3.write(0, 0, "Hora" )
ronda=datetime.datetime.now().hour
if ronda>=23 or ronda<3:
    worksheet3.write(1,0,"0000")
if ronda>=3 and ronda<7:
    worksheet3.write(1,0,"0400")
if ronda>=7 and ronda<11:
    worksheet3.write(1,0,"0800")
if ronda>=11 and ronda<15:
    worksheet3.write(1,0,"1200")
if ronda>=15 and ronda<19:
    worksheet3.write(1,0,"1600")
if ronda>=19 and ronda<23:
    worksheet3.write(1,0,"2000")
worksheet3.write(0, 1, "Revoluciones Br")
worksheet3.write(0, 2, "Revoluciones Er")
worksheet3.write(0, 3, "Gasolina consumida")
worksheet3.write(0, 4, "Gasolina restante")
worksheet3.write(0, 5, "Aceite consumido")
worksheet3.write(0, 6, "Aceite restante")
worksheet3.write(0, 7, "Agua consumida")
worksheet3.write(0, 8, "Agua producida")
worksheet3.write(0, 9, "Agua restante")
worksheet3.write(0, 10, "Agua restante")
```

Figura 3-13: Comandos para la creación de la tercera hoja

```
worksheet4 = workbook.add_worksheet("Ronda mayor")
worksheet4.write(0, 0, "Pañol")
worksheet4.write(0, 1, "Temperatura máxima")
worksheet4.write(0, 2, "Temperatura mínima")
worksheet4.write(0, 3, "Humedad relativa")
worksheet4.write(0, 4, "Humedad relativa")
```

Figura 3-14: Comandos para la creación de la cuarta hoja

Una vez creadas las 4 hojas con todas las columnas, le añadimos el formato deseado a cada hoja, de tal forma que cada columna tenga una longitud determinada, que la primera fila de cada columna se escriba en negrita, que el texto esté centrado y que se generen los bordes. Para ello, se crea una nueva variable llamada 'cell_format', dentro de la cual se encontrará la función 'workbook.add_format'. Para poner en marcha este formato se ejecutará las funciones 'worksheet1.set_row' y 'worksheet1.set_column', siendo el primero de estos el empleado para darle formato a una fila y el segundo para darle formato a una columna. En la Figura 3-15 se puede apreciar este proceso.

```
cell_format = workbook.add_format({'bold': True, 'center_across':True,'border':2})
cell_format1 = workbook.add_format({'border':1})
worksheet1.set_row(0,None, cell_format)
worksheet2.set_row(0,None, cell_format)
worksheet3.set_row(0,None, cell_format)
worksheet4.set_row(0,None, cell_format)
worksheet1.set_column(8, 8, 100,cell_format1)
worksheet1.set_column(1, 7, 20,cell_format1)
worksheet2.set_column(1, 9, 20,cell_format1)
worksheet3.set_column(1, 9, 20)
worksheet3.set_row(1, None,cell_format1)
worksheet3.set_row(2, None,cell_format1)
worksheet3.set_row(3, None,cell_format1)
worksheet3.set_row(4, None,cell_format1)
worksheet3.set_row(5, None,cell_format1)
worksheet3.set_row(6, None,cell_format1)
worksheet4.set_column(0, 3, 18,cell_format1)
worksheet1.set_column(0,0,None,cell_format1)
worksheet2.set_column(0,0,None,cell_format1)
```

Figura 3-15: Aplicación de formatos al Excel

Tras esto, se definen variables para cada columna del Excel para que actúen como un contador, de tal manera que cada vez que se rellene una celda se le sume una unidad a dicha variable y así pasar a indicando así que para escribir un nuevo dato en esa columna habrá de hacerse en la siguiente fila. Por último, cuando se quiera guardar el documento Excel, se tendrá que ejecutar la función 'workbook.close()' para que de esta manera se guarden todos los cambios realizados y se pueda acceder a él en caso de que se quiera modificar algún dato.

3.3.4 Captación de órdenes y relleno de ficheros

Finalmente viene la parte del código en la que se desarrollará la captación de las órdenes realizadas por el usuario. Esta parte empezará con la instrucción 'if __name__=="__main__"', una instrucción mediante el cual el programa podrá captar las palabras pronunciadas por el hablante. En función de la(s) palabra(s) captada(s) ejecutará una acción u otra. El programa estará continuamente en escucha hasta que se le dé la orden de guardar.

En función de la(s) palabra(s) clave captada(s) que determinará, se escribirá el dato que proporcione el reconocedor en la celda correspondiente. Se puede observar un ejemplo de esto en la Figura 3-16.

```
if __name__ == '__main__':  
    while True:  
        statement = takeCommand().lower()  
  
        if statement == 0:  
            continue  
  
        if 'millas totales' in statement: ...  
        if 'millas navegadas' in statement: ...  
        if 'revoluciones del eje' in statement: ...  
        if 'rumbo de aguja' in statement: ...  
        if 'rumbo verdadero' in statement: ...  
        if 'latitud' in statement: ...  
        if 'longitud' in statement: ...
```

Figura 3-16: Parte de la estructura central del código

Dentro de cada instrucción 'if' se encuentra la lógica correspondiente a la confirmación, por parte del usuario, de que desea almacenar la información recogida por el reconocedor. Se decidió añadir una confirmación en caso de que el usuario se confunda a la hora de hablar y lo pueda repetir de nuevo o en caso de que el reconocedor no haya sido del todo preciso al realizar la transcripción. A continuación, en caso de que la confirmación sea positiva, se escribirá la información captada en la correspondiente columna y fila y se sumará una unidad a su contador para pasar a la siguiente fila. En caso contrario, se le pedirá al usuario que repita la información.

Hay que destacar también que en las instrucciones if de 'Millas totales' y 'Dirección del viento' tendrán, en contraposición con el resto de las instrucciones, tres indicaciones mediante las cuales se escribe la hora a la que se captura el dato en la primera columna, la columna 'horas'. Esto se realiza para llevar un registro de cara a saber cuándo sucede un acontecimiento, en la hoja de crónica, y en qué momento cambia la meteorología de la zona, en la hoja de meteorología. Además, también se sumará una unidad al contador de dicha columna 'horas' como se puede observar en la Figura 3-17.

```

if 'millas totales' in statement:
    SpeakText(f"Ha dicho {statement}")
    SpeakText("¿Desea confirmar?")
    print ("esperando confirmación")
    pregunta=respuesta().lower()
    if 'afirmativo' in pregunta:
        statement1=statement.replace("millas totales","")
        try:
            float(statement1)
            worksheet1.write (mt,1,f"{statement1} NM")
            now=datetime.datetime.now()
            tiempo="{:}:{:}".format(now.hour,now.minute)
            worksheet1.write(t1,0,tiempo)
            mt= mt+1
            t1=t1+1
        except:
            SpeakText("Número no reconocido")
    if 'negativo' in pregunta:
        SpeakText("Repita")

```

Figura 3-17: Ejemplo del contenido de la instrucción `if`

En caso de captar la orden de ‘crónica’, el sistema trabajará con un documento de texto y volcará toda la información dictada por el usuario una vez pronunciada la palabra clave. Esta parte del código se corresponde con la parte del Cuaderno de bitácora denominada acaecimientos, la cual se trata de un breve resumen de lo sucedido en una guardia. En la Figura 3-18 se puede observar de manera más detallada este proceso.

```

if 'crónica' in statement:
    with open (ficherotxt,mode='a') as file:
        SpeakText(f"Ha dicho {statement}")
        SpeakText("¿Desea confirmar?")
        print ("esperando confirmación")
        pregunta=respuesta().lower()
        if 'afirmativo' in pregunta:
            statement2=statement.replace("crónica ","")
            statement2=statement2.replace("crónica","")
            file.write (f" {statement2}.")
        if 'negativo' in pregunta:
            SpeakText("Repita")

```

Figura 3-18: Contenido de la instrucción `if` ‘crónica’

3.3.5 Solicitud de confirmación

En ambos códigos la lógica para conseguir que se pueda realizar la confirmación es la misma. Es decir, se basa en añadir otra función que contenga al reconocedor de voz y llamar a la función una vez se haya transcrito el mensaje inicial, sin embargo, su estructura es distinta ya que la forma en la que se programa ambos reconocedores es distinta.

- *Confirmación en Google*

A la hora de implementar la confirmación con el reconocedor de Google fue tan simple como crear una función igual a la de 'takeCommand' llamada 'Respuesta' e implementarla una vez el primer reconocedor haya transcrito la información proporcionada. En la Figura 3-19 se puede ver la estructura de esta función, que es exactamente igual a la de 'takeCommand'.

```
def respuesta():
    r2=sr.Recognizer()
    with sr.Microphone(device_index=1) as source2:
        audio2=r2.listen(source2)

        try:
            pregunta=r2.recognize_google(audio2,language='es-ES')
            print(f"Has dicho:{pregunta}\n")
        except Exception as e:
            return "None"
            SpeakText("Disculpe,¿Puede repetir?")
        return pregunta
```

Figura 3-19: Estructura de la función 'respuesta'

- *Confirmación en Vosk*

Por otro lado, respecto al reconocedor de Vosk, cuando llegó el momento de implementar la confirmación mediante comando de voz, tuvo que ser necesario crear otro *stream* como el primero. La única diferencia con el reconocedor de Google fue que, en este caso, no se podía tener dos *streams* activados a la vez, por lo que hubo que detener el primero, arrancar el segundo, esperar a que se confirmara la información transcrita o se rechazara, parar y cerrar el segundo y arrancar otra vez el primero. De esta forma ya tendríamos implementada la confirmación mediante comando de voz en el segundo código. En la Figura 3-20, la Figura 3-21, y la Figura 3-22 se puede ver todo el proceso seguido para la implementación de la confirmación.

```
def ArrancarConfirmacion():
    p2 = pyaudio.PyAudio()
    stream2 = p2.open(format=pyaudio.paInt16, channels=1, rate=16000, input=True, frames_per_buffer=8000)
    stream2.start_stream()
    return stream2
```

Figura 3-20: Contenido de la función 'arrancar confirmación'

```
def PararConfirmacionYContinuar():
    stream2.stop_stream()
    stream2.close()
```

Figura 3-21: Contenido de la función 'parar confirmación y continuar'

```

if 'millas totales' in result:
    SpeakText(f"Usted ha dicho {result}")
    SpeakText("¿Desea confirmar?")
    print ("Esperando confirmación")
    stream.stop_stream()
    stream2=ArrancarConfirmacion()
    while True:
        data2=stream2.read(4000)
        if len(data2)==0:
            break
        if rec.AcceptWaveform(data2):
            result2=json.loads(rec.Result())
            result2=result2['text']
            print (result2)
            if 'afirmativo' in result2:
                stream.start_stream()
                try:
                    statement1 = result.replace ("millas totales ", "")
                    worksheet1.write(mt,1,f"{to_number(statement1)} NM")
                    now = datetime.datetime.now()
                    tiempo= "{}:{}".format(now.hour,now.minute)
                    worksheet1.write(t1,0,tiempo)
                    mt=mt+1
                    t1=t1+1
                    PararConfirmacionYContinuar()
                    break
                except:
                    print("Número no reconocido")
                    break
            if 'negativo' in result2:
                SpeakText("Repita la frase")
                stream.start_stream()
                PararConfirmacionYContinuar()
                break

```

Figura 3-22: Ejemplo de una instrucción de confirmación en el reconocedor de Vosk

3.3.6 Guardar documentos

Finalmente, en caso de que el reconocedor detecte la palabra clave ‘guardar’, se ejecutará el código mostrado en la Figura 3-23. En cuanto detecte esta palabra, el programa cerrará el Excel creado, lo moverá a una carpeta determinada y la aplicación terminará.

```

if 'guardar' in statement:
    SpeakText("¿Esta seguro de que desea guardar?")
    print ("esperando confirmación")
    pregunta=respuesta().lower()
    if 'afirmativo' in pregunta:
        workbook.close()
        SpeakText("Guardando su cuaderno de bitácora")
        SpeakText("Hasta la próxima")
        break
    if 'negativo' in pregunta:
        SpeakText("Repita")

```

Figura 3-23: Contenido de la función 'guardar'

3.3.7 Modo de consulta

Además de todas las funciones comentadas anteriormente, se decidió añadir una nueva implementación al código. Esta nueva implementación se activa con el comando de voz ‘consultar’. Una vez dicha esa palabra, el programa pasará a estar en un modo de consulta, en el que el usuario podrá preguntar en qué fila se encuentra una columna determinada o saber si toda la hoja se encuentra en la misma fila, en caso de que se le haya olvidado introducir algún dato.

Al igual que pasó con la confirmación, esta nueva implementación tendrá una estructura distinta dependiendo de si se usa con el reconocedor Vosk o si se usa con el reconocedor de Google.

Para llevar a cabo este nuevo modo, se siguió el mismo método que el usado en el apartado 3.3.5. Se puede ver la estructura de esta implementación en el código que usa el reconocedor de Google en la Figura 3-24 y en el código que utiliza Vosk en la Figura 3-25.

```
if 'consultar' in statement:
    consulta=False
    SpeakText ("Indique sobre qué columna desea saber en qué fila se encuentra. Si por el contrario le gustaría saber el estado")
    SpeakText("Si quiere cancelar diga cancelar")
    print("esperando")
    while not consulta:
        pregunta=respuesta().lower()
        if 'cancelar' in pregunta:
            consulta=True
            SpeakText("Opción cancelada")
        if 'millas totales' in pregunta:
            SpeakText(f"La columna de millas totales se encuentra en la fila {mt}")
        if 'millas navegadas' in pregunta:
            SpeakText(f"La columna de millas navegadas se encuentra en la fila {mn}")
        if 'revoluciones del eje' in pregunta:
            SpeakText(f"La columna de revoluciones del eje se encuentra en la fila {rev}")
        if 'rumbo de aguja' in pregunta:
            SpeakText(f"La columna de rumbo de aguja se encuentra en la fila {ra}")
        if 'rumbo verdadero' in pregunta:
            SpeakText(f"La columna de rumbo verdadero se encuentra en la fila {rv}")
        if 'latitud' in pregunta:
            SpeakText(f"La columna de latitud se encuentra en la fila {lat}")
        if 'longitud' in pregunta:
            SpeakText(f"La columna de longitud se encuentra en la fila {long}")
        if 'acaecimiento' in pregunta:
            SpeakText(f"La columna de acaecimiento se encuentra en la fila {aca}")
        if 'dirección del viento' in pregunta:
            SpeakText(f"La columna de dirección del viento se encuentra en la fila {dv}")
        if 'velocidad del viento' in pregunta:
            SpeakText(f"La columna de velocidad del viento se encuentra en la fila {vv}")
```

Figura 3-24: Implementación de la función 'consulta' con el reconocedor de Google

```
if 'consultar' in result:
    consulta=False
    SpeakText ("Indique sobre qué columna desea saber en qué fila se encuentra. Si por el contrario le gustaría saber el e")
    SpeakText("Si quiere cancelar diga cancelar")
    print("esperando")
    while not consulta:
        stream.stop_stream()
        stream2=ArrancarConfirmacion()
        while True:
            data2=stream2.read(8000)
            if len(data2)==0:
                break
            if rec.AcceptWaveform(data2):
                result2=json.loads(rec.Result())
                result2=result2['text']
                print (result2)
                if 'cancelar' in result2:
                    consulta=True
                    SpeakText("Opción cancelada")
                    stream.start_stream()
                    PararConfirmacionYContinuar()
                    break
                if 'millas totales' in result2:
                    SpeakText(f"La columna de millas totales se encuentra en la fila {mt}")
                if 'millas recorridas' in result2:
                    SpeakText(f"La columna de millas navegadas se encuentra en la fila {mn}")
                if 'revoluciones del eje' in result2:
                    SpeakText(f"La columna de revoluciones del eje se encuentra en la fila {rev}")
                if 'rumbo de aguja' in result2:
                    SpeakText(f"La columna de rumbo de aguja se encuentra en la fila {ra}")
                if 'rumbo verdadero' in result2:
                    SpeakText(f"La columna de rumbo verdadero se encuentra en la fila {rv}")
```

Figura 3-25: Implementación de la función 'consulta' con el reconocedor de Vosk

3.4 Problemas detectados

A la hora de implementar el reconocedor de Google dentro del programa no se percibió ningún problema aparente, ya que este posee un modelo de lenguaje con un vocabulario bastante extenso e interpreta los números de modo numérico y no textual, tal y como sucede en Vosk.

Cuando se empleó el reconocedor Vosk, lo primero que se pudo apreciar fue que, a la hora de captar los números este los transcribe como texto en lugar de números. Para lograr ese paso, se utilizó el módulo ‘spa2num.converter’, del cual se importó la función ‘to_number’ encargada de convertir los números en texto a número. Sin embargo, para conseguir que se realice correctamente es necesario que en el texto aparezca solo el número y sin ninguna palabra más.

Para solucionar esto, lo que se hizo fue utilizar la función ‘replace’ para que el resultado final de la variable ‘result’ solo contuviera el número expresado textualmente. A pesar de esto, en ocasiones, Vosk no capta bien el número si no está escrito de una forma específica, es decir, puede captar el número ‘doscientos’, pero no puede captar el número ‘doscientas’. Por lo que, si se le diera a confirmar habiendo dicho ‘doscientas’ saltará un error y se cerrará el programa.

Este comportamiento no deseado va en contra de la robustez de la aplicación desarrollada. Para evitar que ocurra esto, se ha modificado la versión inicial de este código incluyendo una instrucción ‘try’ y una instrucción ‘except’, de tal manera que siempre intente transcribir el número dicho de texto a número, y que en caso de error avise de la necesidad de repetir la información. Se puede apreciar un ejemplo de este desarrollo en la Figura 3-26.

```

if 'revoluciones del eje' in result:
    SpeakText(f"Usted ha dicho {result}")
    SpeakText("¿Desea confirmar?")
    print ("Esperando confirmación")
    stream.stop_stream()
    stream2=ArrancarConfirmacion()
    while True:
        data2=stream2.read(4000)
        if len(data2)==0:
            break
        if rec.AcceptWaveform(data2):
            result2=json.loads(rec.Result())
            result2=result2['text']
            print (result2)
            if 'afirmativo' in result2:
                stream.start_stream()
                try:
                    statement1 = result.replace("revoluciones del eje","")
                    worksheet1.write(rev,3,f"{to_number(statement1)} rpm")
                    rev=rev+1
                    PararConfirmacionYContinuar()
                    break
                except:
                    print("Número no reconocido")
                    break
            if 'negativo' in result2:
                SpeakText("Repita la frase")
                stream.start_stream()
                PararConfirmacionYContinuar()
                break

```

Figura 3-26: Implementación de las instrucciones 'try' y 'except' y la función ‘to_number’

A parte de esto existe otra pequeña particularidad en este código. Se trata de que en los apartados de latitud y longitud es necesario indicar más de un número y especificarle los grados y minutos.

Además, en caso de que se trate de latitud será necesario indicar de si se trata de Norte o Sur, y en caso de que sea longitud se indicará de Este u Oeste.

Con la función ‘to_number’ no es posible conseguir esto. La solución a la que se llegó fue intentar implementar una función específica para latitud y longitud. Se logró encontrar esta función llamada ‘text2int’, pero el único problema fue que estaba diseñada para números en inglés, por lo que hubo que adaptarlo al idioma español.

Para poder adaptarlo a nuestro idioma, se cambiaron los nombres de los números que contenía la función comentada anteriormente de inglés a español. Además, fue necesario crear otra función, denominada ‘frase’, para transcribir los números de texto en español a un formato mediante el cual estos pudiesen ser escritos como números. Se puede apreciar todo el procedimiento llevado a cabo, así como el contenido de todas las funciones mencionadas en la Figura 3-27, la Figura 3-28, y la Figura 3-29.

```
if 'latitud' in result:
    SpeakText(f"Usted ha dicho {result}")
    SpeakText("¿Desea confirmar?")
    print ("Esperando confirmación")
    stream.stop_stream()
    stream2=ArrancarConfirmacion()
    while True:
        data2=stream2.read(4000)
        if len(data2)==0:
            break
        if rec.AcceptWaveform(data2):
            result2=json.loads(rec.Result())
            result2=result2['text']
            print (result2)
            if 'afirmativo' in result2:
                stream.start_stream()
                try:
                    statement=frase(result)
                    worksheet1.write(lat,6,statement)
                    lat=lat+1
                    PararConfirmacionYContinuar()
                    break
                except:
                    print("Número no reconocido")
                    break
            if 'negativo' in result2:
                SpeakText("Repita la frase")
                stream.start_stream()
                PararConfirmacionYContinuar()
                break
```

Figura 3-27:Contenido de la instrucción 'latitud'

```

def text2int (textnum, numwords={}):
    if not numwords:
        units = [
            "cero", "uno", "dos", "tres", "cuatro", "cinco", "seis", "siete", "ocho",
            "nueve", "diez", "once", "doce", "trece", "catorce", "quince",
            "dieciseis", "diecisiete", "dieciocho", "diecinueve"]
        tens = ["", "", "veinte", "treinta", "cuarenta", "cincuenta", "sesenta", "setenta", "ochenta", "noventa"]
        scales = ["cien", "mil", "millón", "billón", "trillón"]
        numwords["y"] = (1, 0)
        for idx, word in enumerate(units): numwords[word] = (1, idx)
        for idx, word in enumerate(tens): numwords[word] = (1, idx * 10)
        for idx, word in enumerate(scales): numwords[word] = ([10 ** (idx * 3 or 2), 0])

    textnum = textnum.replace('-', ' ')
    current = result = 0
    curstring = ""
    onnumber = False
    for word in textnum.split():
        if word not in numwords:
            if onnumber:
                curstring += repr(result + current) + " "
                curstring += word + " "
                result = current = 0
                onnumber = False
            else:
                scale, increment = numwords[word]

                current = current * scale + increment
                if scale > 100:
                    result += current
                    current = 0
                onnumber = True

    if onnumber:
        curstring += repr(result + current)

    return curstring

```

Figura 3-28: Función de identificación de número en español

```

def frase(result):

    result=result.replace("con",",")
    result=result.replace("grados","º")
    result=result.replace("minutos","'")
    result=result.replace("norte","N")
    result=result.replace("sur","S")
    result=result.replace("oeste","W")
    result=result.replace("este","E")

    result=result.replace("longitud","")
    result=result.replace("latitud","")

    result=result.replace("una","uno")
    result=result.replace(["veinti","veinte y "])
    result=result.replace("doscientas","doscientos")
    result=result.replace("doscientos","dos cien")
    result=result.replace("trescientas","trescientos")
    result=result.replace("trescientos","tres cien")
    result=result.replace("cuatrocientas","cuatrocientos")
    result=result.replace("cuatrocientos","cuatro cien")
    result=result.replace("quinientas","quinientos")
    result=result.replace("quinientos","cinco cien")
    result=result.replace("seiscientas","seiscientos")
    result=result.replace("seiscientos","seis cien")
    result=result.replace("setecientas","setecientos")
    result=result.replace("setecientos","siete cien")
    result=result.replace("ochocientas","ochocientos")
    result=result.replace("ochocientos","ocho cien")
    result=result.replace("novecientas","novecientos")
    result=result.replace("novecientos","nueve cien")
    result=result.replace("ciento","uno cien")

    result=text2int(result)
    return result

```

Figura 3-29: Función de adaptación de números a español

Por otro lado, debido a la falta de vocabulario de Vosk, ha sido necesario indicar el estado de la mar con números para, más tarde, convertirlos en texto según su correspondiente estado, es decir, el número 0 se asociará con el estado en calma, el número 1 con el estado rizada, y así sucesivamente hasta llegar al número 9 que se asocia con enorme. En la Figura 3-30 se puede ver la función creada para realizar esta conversión.

```
def conversor(result):
    result=result.replace("estado de la mar cero","calma")
    result=result.replace("estado de la mar uno","rizada")
    result=result.replace("estado de la mar dos","marejadilla")
    result=result.replace("estado de la mar tres","marejada")
    result=result.replace("estado de la mar cuatro","fuerte marejada")
    result=result.replace("estado de la mar cinco","gruesa")
    result=result.replace("estado de la mar seis","muy gruesa")
    result=result.replace("estado de la mar siete","arbolada")
    result=result.replace("estado de la mar ocho","montañosa")
    result=result.replace("estado de la mar nueve","enorme")
    return result
```

Figura 3-30: Conversor para la función 'mar'

Este problema también se podría dar a la hora de redactar el apartado del Cuaderno de Bitácora en el documento de texto, por lo que también se implementó aquí. Con la intensidad del viento pasó lo mismo, por lo que se le aplicó el mismo método usado en el estado de la mar. En la Figura 3-31 se puede ver la función de conversión para los datos en la crónica.

```
def datoscronica(result):
    result=result.replace("mar cero","estado de la mar en calma")
    result=result.replace("mar uno","estado de la mar rizada")
    result=result.replace("mar dos","estado de la mar marejadilla")
    result=result.replace("mar tres","estado de la mar marejada")
    result=result.replace("mar cuatro","estado de la mar fuerte marejada")
    result=result.replace("mar cinco","estado de la mar gruesa")
    result=result.replace("mar seis","estado de la mar muy gruesa")
    result=result.replace("mar siete","estado de la mar arbolada")
    result=result.replace("mar ocho","estado de la mar montañosa")
    result=result.replace("mar nueve","estado de la mar enorme")
    result=result.replace("viento de intensidad cero","viento en calma")
    result=result.replace("viento de intensidad uno","viento ventolina")
    result=result.replace("viento de intensidad dos","viento flojito")
    result=result.replace("viento de intensidad tres","viento flojo")
    result=result.replace("viento de intensidad cuatro","viento bonancible moderado")
    result=result.replace("viento de intensidad cinco","viento frequito")
    result=result.replace("viento de intensidad seis","viento fresco")
    result=result.replace("viento de intensidad siete","viento frescachón")
    result=result.replace("viento de intensidad ocho","viento temporal")
    result=result.replace("viento de intensidad nueve","viento temporal fuerte")
    result=result.replace("viento de intensidad diez","viento temporal duro")
    result=result.replace("viento de intensidad once","viento temporal muy duro")
    result=result.replace("viento de intensidad doce","viento temporal huracanado")
    return result
```

Figura 3-31: Conversión para la función 'crónica'

Por último, debido a tener que activar un segundo 'stream' para implementar la confirmación por voz, se hizo necesario implementar una forma distinta de cerrar la captación de órdenes. Lo que se hizo fue asignar al primer bucle una variable llamada 'salir' de tal manera que siempre que estuviera en modo 'False' continuaría el bucle. De esta manera, al confirmar el guardado del documento, se cambiaría el valor de esta variable a 'True' y se cerraría el primer bucle. En la Figura 3-32 y la Figura 3-33 se puede ver la implementación de la variable 'salir' y el contenido de la función guardar.

```
salir=False

if __name__=='__main__':
    while not salir:
```

Figura 3-32: Implementación de la variable 'salir'

```
if 'guardar' in result:
    SpeakText("¿Quiere guardar el Cuaderno de Bitácora?")
    print ("Esperando confirmación")
    stream.stop_stream()
    stream2=ArrancarConfirmacion()
    while True:
        data2=stream2.read(4000)
        if len(data2)==0:
            break
        if rec.AcceptWaveform(data2):
            result2=json.loads(rec.Result())
            result2=result2['text']
            print (result2)

            if 'no' in result2:
                SpeakText("Repita la frase")
                stream.start_stream()
                PararConfirmacionYContinuar()
                break

            if 'si' in result2:
                stream.start_stream()
                PararConfirmacionYContinuar()
                workbook.close()
                salir=True
                break
```

Figura 3-33: Función 'guardar' para la versión de Vosk

4 FUNCIONAMIENTO DEL SISTEMA Y PRUEBAS

En este apartado se desarrollará un manual para explicar el funcionamiento del programa en el que se incluirán los resultados finales. A continuación, veremos las distintas pruebas realizadas para comprobar la robustez y efectuar la validación del código.

4.1 Manual de usuario

A continuación, explicaré como emplear de forma efectiva el programa creado. Al iniciar el programa este nos indicará que está arrancando y cuando podemos empezar a hablar mediante un comando de voz en la versión de Google y por pantalla en la versión de Vosk. Es a partir de este momento cuando podemos empezar a interactuar con él.

Para realizar la interacción se definieron una serie de comandos de voz asociados a cada columna del Excel creado. Estos comandos de voz irán asociados a cada una de las 4 hojas creadas con anterioridad. Los comandos de voz asociados con la primera hoja se pueden ver en la Tabla 1, los asociados con la segunda en la Tabla 2, los asociados con la tercera en la Tabla 3 y los asociados con la cuarta en la Tabla 4.

Palabras clave	Función asociada
Millas totales	Escribirá en la columna de Millas totales
Millas recorridas	Escribirá en la columna de Millas navegadas
Revoluciones del eje	Escribirá en la columna de Revoluciones del eje
Rumbo de aguja	Escribirá en la columna de Rumbo de aguja
Rumbo verdadero	Escribirá en la columna de Rumbo verdadero
Latitud	Escribirá en el apartado de Latitud
Longitud	Escribirá en el apartado de Longitud
Acaecimiento	Escribirá en el apartado de Acaecimiento

Tabla 1: Comandos de voz para la hoja de crónica

Palabras clave	Función asociada
Dirección del viento	Escribirá en la columna de Dirección del viento
Velocidad del viento	Escribirá en la columna de Velocidad del viento
Visibilidad actual	Escribirá en la columna de Visibilidad
Presión	Escribirá en la columna de Barómetro
Bola seca	Escribirá en la columna de Bola seca
Bola húmeda	Escribirá en el apartado de Bola húmeda
Cantidad de nubes	Escribirá en el apartado de Cantidad de nubes
Clase de nubes	Escribirá en el apartado de Clase de nubes
Estado de la mar	Escribirá en el apartado de Estado de la mar

Tabla 2: Comandos de voz para la hoja de meteorología

Palabras clave	Función asociada
Motor babor	Escribirá en la columna de Revoluciones Babor
Motor estribor	Escribirá en la columna de Revoluciones Estribor
Gasolina consumida	Escribirá en la columna de Gasolina consumida
Gasolina restante	Escribirá en la columna de Gasolina restante
Aceite consumido	Escribirá en la columna de Aceite consumido
Aceite restante	Escribirá en el apartado de Aceite restante
Agua consumida	Escribirá en el apartado de Agua consumida
Agua producida	Escribirá en el apartado de Agua producida
Agua restante	Escribirá en el apartado de Agua restante

Tabla 3: Comandos de voz para la hoja de máquinas

Palabras clave	Función asociada
Compartimento	Escribirá en la columna de Pañol
Temperatura máxima	Escribirá en la columna de Temperatura máxima
Temperatura mínima	Escribirá en la columna de Temperatura mínima
Humedad	Escribirá en la columna de Humedad

Tabla 4: Comandos de voz para la hoja de Ronda mayor

En la Figura 4-1 podemos ver un ejemplo de cómo empezaría el programa una vez iniciado. Los diálogos de la izquierda se corresponden con la información dicha por el usuario y los diálogos de la derecha se corresponden con el ordenador. Hasta que no detecte alguno de los comandos comentados anteriormente no activará la confirmación.

El proceso de confirmación lo podemos observar en la Figura 4-2. Si aceptamos la confirmación, diciendo la palabra afirmativo, se escribirá la información que se le ha transmitido, como se ve en la

Figura 4-3 y si se rechaza la confirmación, diciendo la palabra negativo, no escribirá nada y pedirá al usuario que vuelva a repetir el dato, como se ve en la Figura 4-4.

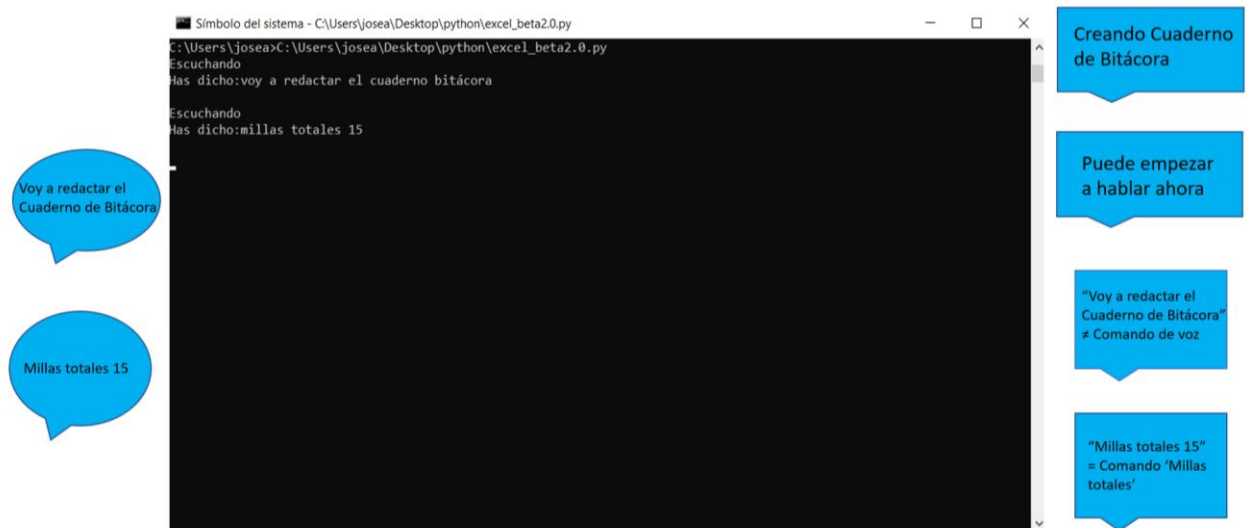


Figura 4-1: Inicio de programa

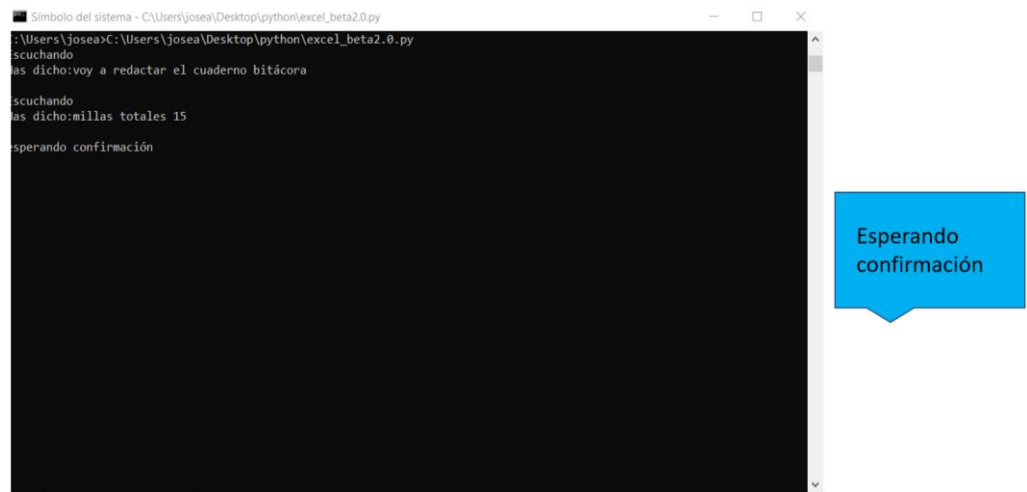


Figura 4-2: Solicitud de confirmación

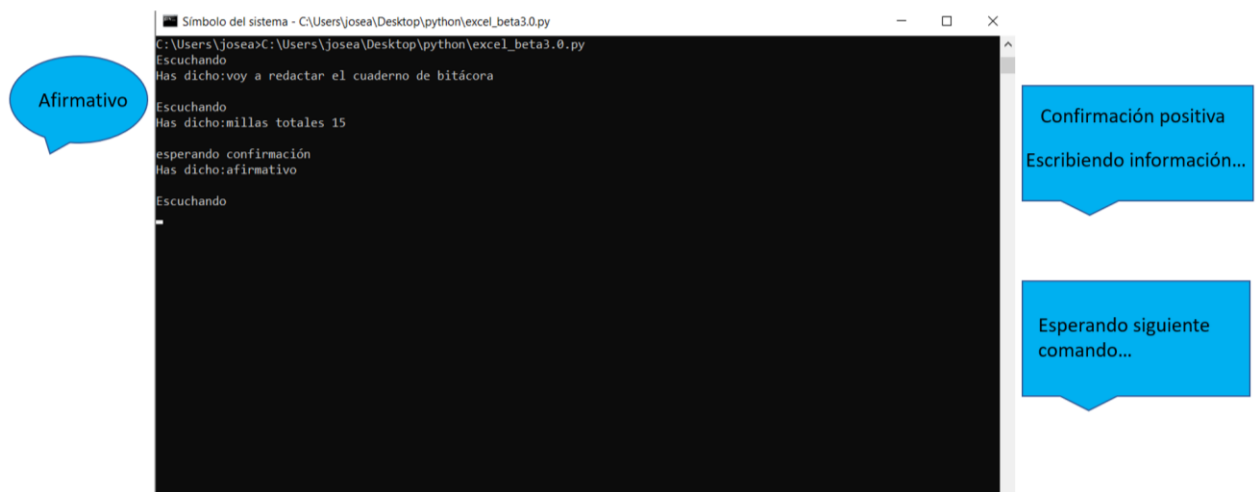


Figura 4-3: Caso de confirmación positiva

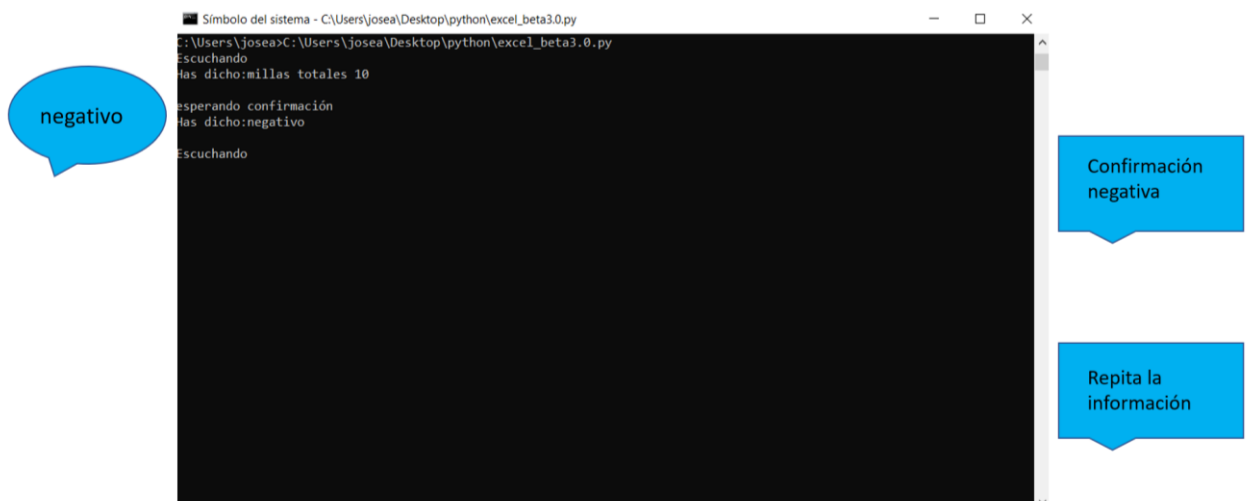


Figura 4-4: Caso de confirmación negativa

Por otro lado, cada vez que detecte el comando de voz ‘crónica’ plasmará en un documento de texto toda la información que venga después de ese comando de voz, como se puede ver en la Figura 4-5. Una vez se hayan terminado de rellenar todos los datos correspondientes, se puede proceder a guardar el documento. Para ello bastaría con pronunciar el comando de voz ‘Guardar’ y confirmar, como se puede ver en la Figura 4-6.

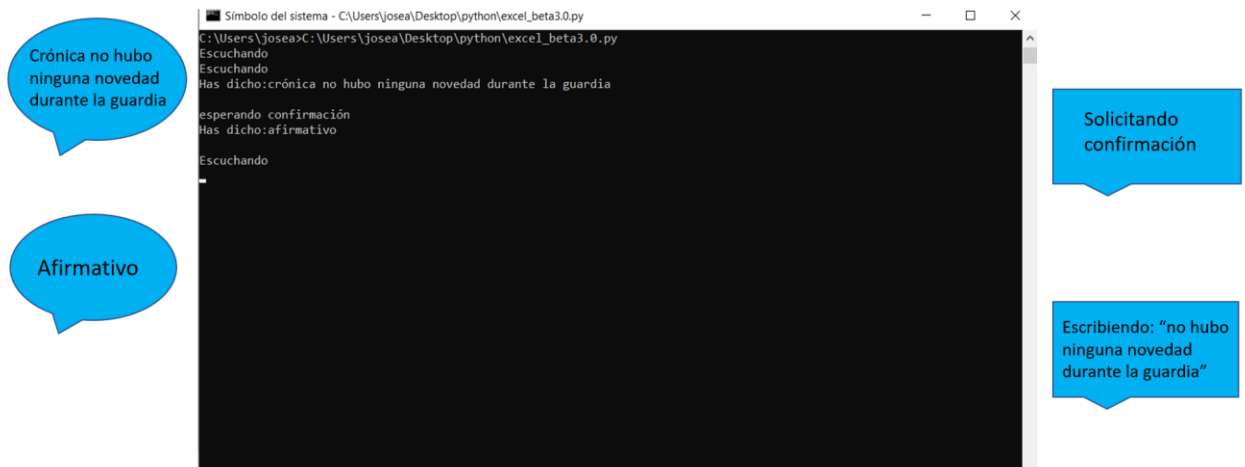


Figura 4-5: Ejemplo de escritura de la crónica



Figura 4-6: Guardado de documentos

Por otro lado, si se dice el comando de voz ‘consulta’ el programa entrará en un modo en el cual puede consultar la fila en la que se encuentra cada columna hasta que se diga el comando ‘cancelar’, momento a partir del cual el programa saldrá de este modo sin cerrarse. En este modo se podrá consultar el estado de una columna o de una hoja diciendo su nombre y ver si se encuentran todas las columnas en la misma fila para comprobar si nos hemos olvidado de transmitir un dato. En la Figura 4-7 podemos observar un ejemplo de este modo.

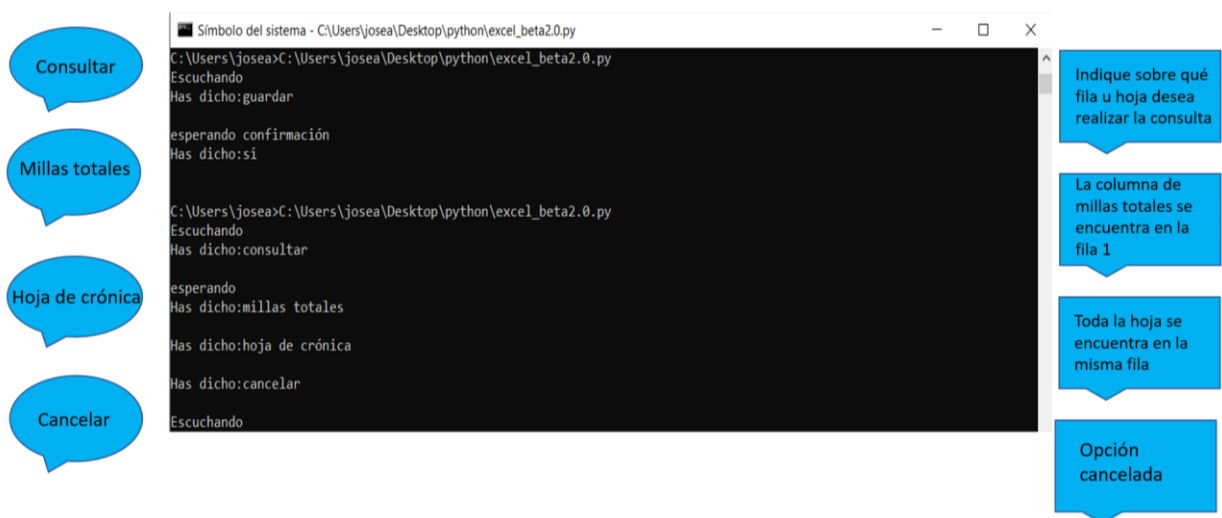


Figura 4-7: Ejemplo del modo consulta

En cuanto a los resultados, tras la finalización del programa podemos observar que tendremos un documento Excel y un documento txt, que aparecerán en la carpeta que hemos indicado en el código mediante la función 'os.chdir', el cual nos conduce hasta una carpeta específica dentro del ordenador. Ambos documentos tienen el mismo nombre tal y como podemos apreciar en la Figura 4-8.

Nombre	Fecha de modificación	Tipo	Tamaño
Cuaderno de bitácora del 9-3 a las 19 ho...	09/03/2021 19:50	Hoja de cálculo d...	12 KB
Cuaderno de bitácora del 9-3 a las 19 ho...	09/03/2021 19:49	Documento de tex...	1 KB

Figura 4-8: Resultados finales del código

Dentro del documento Excel podemos apreciar que se encuentran las 4 hojas mencionadas en el apartado de Implementación junto con sus columnas correspondientes y con el formato dado, tal y como se pueden apreciar en la Figura 4-9, la Figura 4-10, la Figura 4-11 y la Figura 4-12.

DESARROLLO DE UN ASISTENTE DE VOZ PARA LA GENERACIÓN DE UN CUADERNO DE BITÁCORA DIGITAL

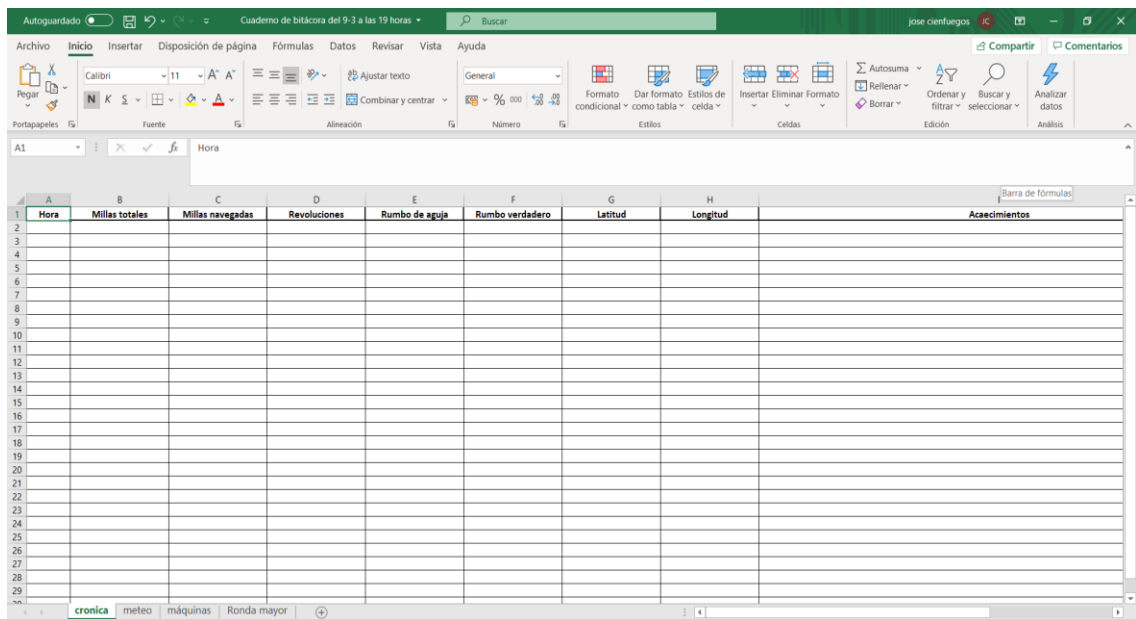


Figura 4-9: Hoja de crónica del documento Excel

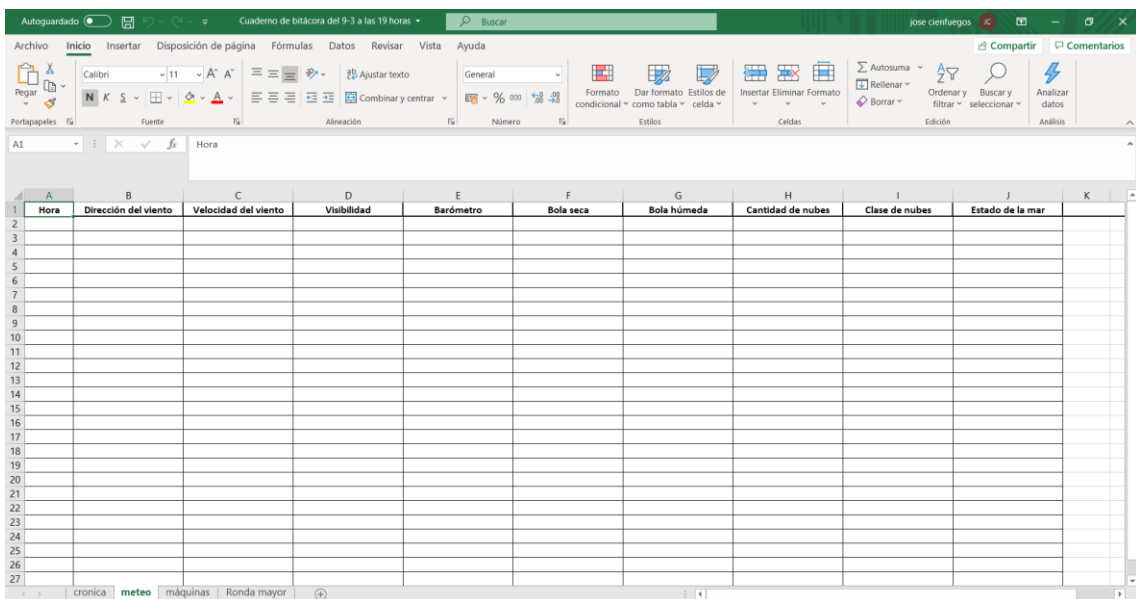


Figura 4-10: Hoja de meteorología del documento Excel

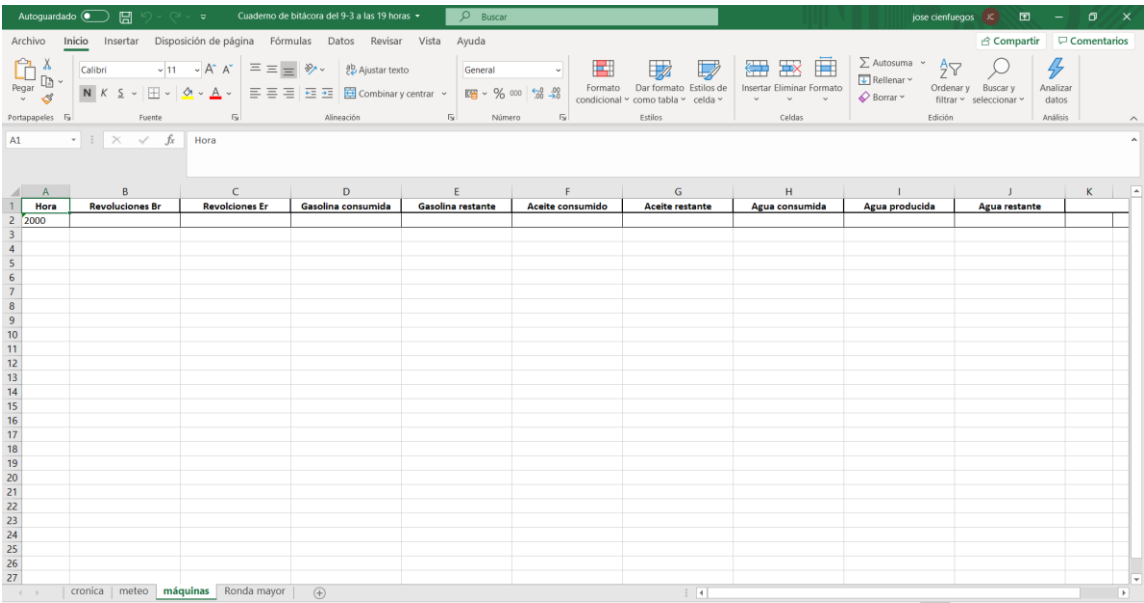


Figura 4-11: Hoja de máquinas del documento Excel

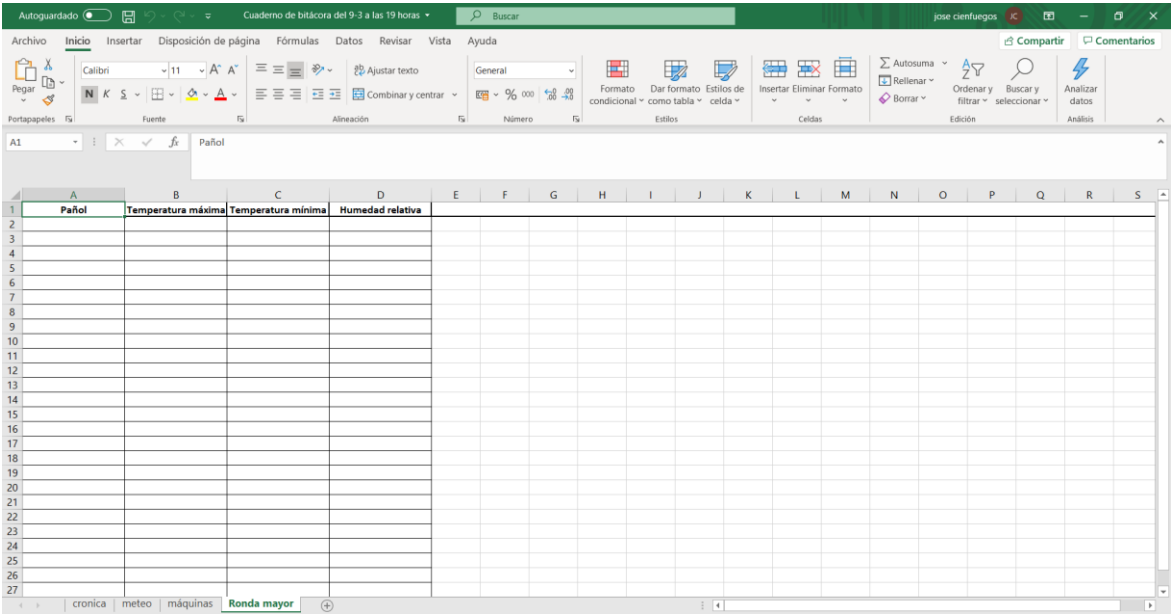


Figura 4-12: Hoja de ronda mayor del documento Excel

Y respecto al documento de texto, tendrá la apariencia que podemos apreciar en la Figura 4-13.

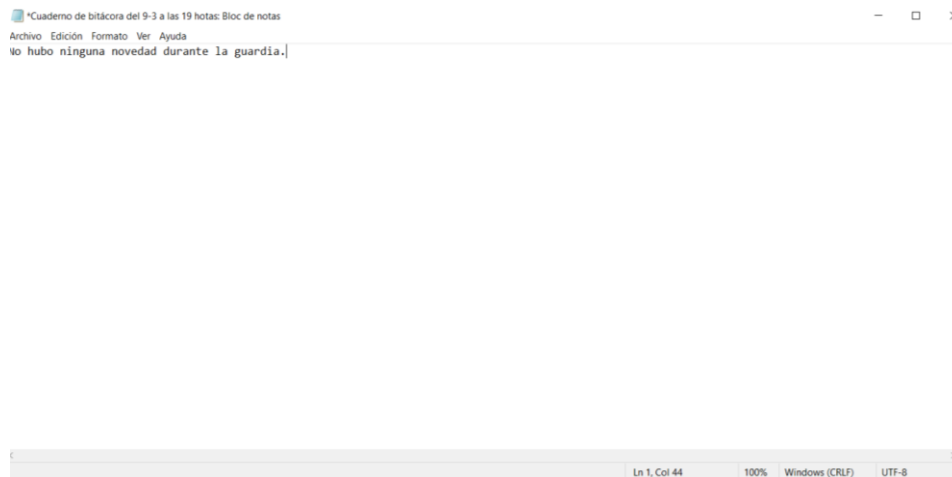


Figura 4-13: Ejemplo de documento de texto

4.2 Pruebas

Algunos requisitos no funcionales considerados para esta aplicación son la robustez, su usabilidad mediante comandos de voz, evitar distracciones y garantizar que la información recogida es la correcta. En este apartado realizaremos todas las pruebas oportunas para intentar comprobar la solidez del código y cómo es capaz de trabajar. Para comprobar esto se han realizado varias pruebas.

Una de ellas ha sido indicarle a la versión que usa el reconocedor de Vosk datos erróneos, como por ejemplo decir en apartados en los que debería ir un dato numérico palabras como casa o papel. De esta forma se ha realizado esta prueba en cada columna que era necesario introducir un dato numérico. Se pudo comprobar que cada vez que se decía una palabra en lugar de un número el programa devolvía un mensaje de “número no reconocido”, no escribía el dato que le habíamos proporcionado y le pedirá al usuario que repita el dato como se ve en la Figura 4-14.

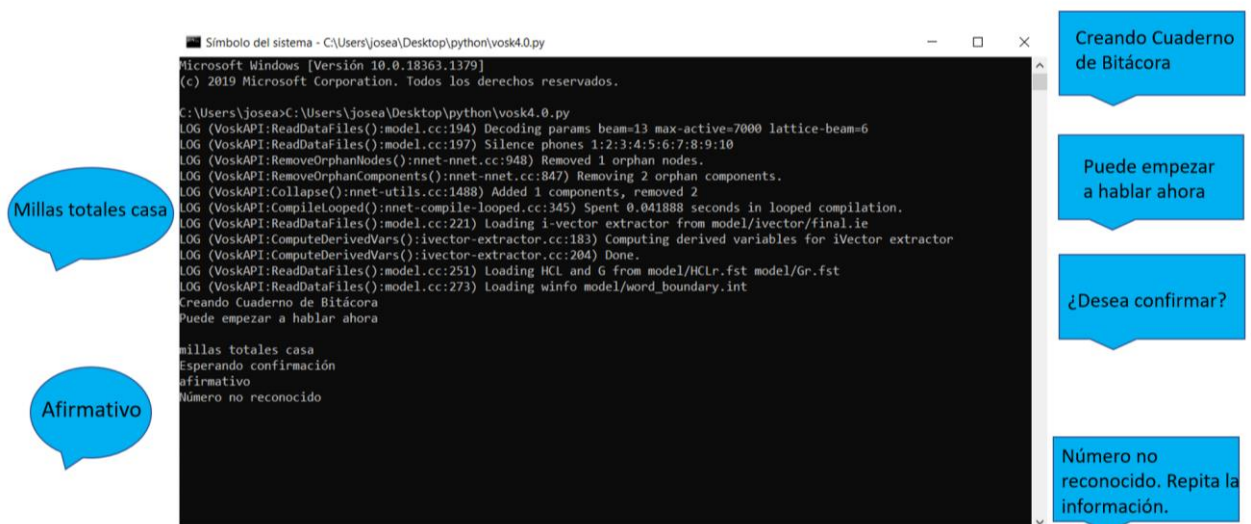


Figura 4-14: Comprobación del funcionamiento de Vosk

Otra prueba consistió en comprobar la efectividad del comando ‘consulta’. Este comando, como se explicó anteriormente, es un modo en el que el programa entra en un recordatorio para comprobar que se han rellenado todas las columnas necesarias. Se comprobó para cada columna y para cada hoja y se

pudo observar que funciona correctamente. Además de esto también se comprobó que permitía entrar y salir de este modo todas las veces que se quisiera.

A parte de lo comentado anteriormente, una vez terminado el programa se decidió probar la validez de este sometiéndolo a la escritura de datos de las dos primeras filas de cada columna usando, primero el reconocedor de Google y después el reconocedor de Vosk.

Se pudo apreciar que en el código que utiliza el reconocedor de Google, el código que trabaja de manera *online*, no tuvo ningún problema en la transcripción de palabras, ya que, transcribía toda la información dicha al completo o casi al completo, el único problema que se detectó en la transcripción es que en lugar de detectar pañol, detectaba español y como el reconocedor de Vosk tenía el mismo problema se decidió cambiar la palabra pañol por compartimento. A parte de esto, otro problema que se pudo detectar en esta versión del programa es la tardanza que a veces mostraba durante la transcripción debido a la velocidad de la conexión a Internet.

Por otro lado, con el código en el que se usó Vosk, se pudo apreciar una velocidad de transcripción mayor a la de Google, pero que, en algunos casos la información transcrita no era del todo precisa por lo que hubo que repetir la información varias veces.

Esto pasaba sobre todo en el apartado de 'latitud', debido a que en lugar de detectar 'latitud' detecta 'la actitud'. Por ello, y para evitar repetir la información varias veces hasta que lo detecte correctamente, se decidió incorporar una nueva instrucción 'if' con la palabra 'la actitud'. De esta manera, si el usuario pronunciaba la palabra 'latitud' y el reconocedor lo transcribía como 'la actitud' los datos se seguirán escribiendo en el Excel si el usuario lo confirma. Las correcciones aplicadas se pueden ver en la Figura 4-15.

```

812         if 'la actitud' in statement:
813             SpeakText(f"Ha dicho {statement}")
814             SpeakText("¿Desea confirmar?")
815             print ("esperando confirmación")
816             pregunta=respuesta().lower()
817
818             if 'afirmativo' in pregunta:
819                 Latitud()
820                 lat=lat+1
821                 SpeakText("Escrito correctamente")
822             if 'negativo' in pregunta:
823                 SpeakText("Repita")

```

Figura 4-15: Función if de 'la actitud'

Por otro lado, también se tuvo que sustituir la palabra clave 'millas navegadas' por 'millas recorridas' debido a que en el modelo de idioma de Vosk no se encuentra disponible la palabra 'navegadas', y siempre la incorporaba como 'navegas'. Lo mismo pasó con la palabra 'acaecimiento', la cual se tuvo que sustituir por 'suceso'. Salvo por estos pequeños inconvenientes no se detectaron más problemas derivados del modelo de lenguaje de Vosk.

También se comprobó la efectividad del programa mediante el uso de distintos tipos de micrófono, y se pudo comprobar que, cuanta mayor sea la calidad del micrófono, más claras y precisas serán las transcripciones.

5 CONCLUSIONES Y LÍNEAS FUTURAS

5.1 Conclusiones

Repasando los objetivos definidos en el apartado 1.2 podemos decir que:

- En este trabajo se ha desarrollado un asistente de voz capaz de transcribir la información que un usuario en el puente de un barco le va comunicando de viva voz.
- En este trabajo se ha conseguido que este asistente cree y escriba en un fichero Excel o un fichero txt la información que se le transmite por medio de unos comandos de voz especificados.
- En este trabajo se ha conseguido implementar una confirmación para que el asistente de voz pregunte antes de escribir la información recibida.
- En este trabajo se ha conseguido que el asistente sea capaz de detectar información errónea y avisar al usuario para que repita la información transmitida.

Una vez concluido el proyecto se puede afirmar que:

- El resultado final de la aplicación satisface todas las necesidades que vienen incluidas en el actual Cuaderno de Bitácora de la Armada española.
- La aplicación es capaz de funcionar únicamente mediante comandos de voz sin la necesidad de escribir ningún dato mediante teclado.
- Una vez finalizada la aplicación se podrá acceder a ambos ficheros para poder realizar consultas o copias de los archivos.

Si recordamos los objetivos que fijamos en el apartado 1.2 podremos afirmar que han alcanzado todos y cada uno de ellos ya que:

Este proyecto ha permitido que el alumno desarrolle cierto interés hacia la programación de aplicaciones y aplicar lo aprendido en dos asignaturas que cursó a lo largo del grado, profundizar en los distintos lenguajes de programación y entornos de desarrollo existentes y, además, acceder a una gran comunidad de programadores que, con su ayuda, le han permitido superar las dificultades encontradas a lo largo del proyecto.

5.2 Líneas futuras

Como posibles líneas futuras que se puedan desarrollar para complementar este TFG, se podrían encontrar:

- Intentar mejorar el modelo de idioma del reconocedor de Vosk incorporándole vocabulario exclusivo sobre la navegación o crear un modelo de idioma de Vosk con un vocabulario exclusivo de uso Armada. En la referencia [39] aparecen una serie de indicaciones mediante las cuales se podría llevar a cabo la incorporación de nuevas palabras dentro del reconocedor de Vosk.
- Implementación de estos códigos dentro del puente de un barco ya sea con un ordenador o con un pequeño dispositivo para que lo pueda almacenar y guardar en una nube exclusivamente de uso Armada. Por lo tanto, también se propone desplegar esta aplicación en un dispositivo reducido que sea fácilmente portable, como puede ser el ejemplo de una Raspberry [40], un Arduino [41] o incluso un smartphone.
- Implementar la conectividad con un casco que cuente con un micrófono y con un audífono para que el usuario que se encargue de redactar el Cuaderno de Bitácora pueda tener cierta libertad dentro del puente, y pueda transmitir la información hasta el dispositivo desde cualquier parte sin necesidad de tener que estar cerca de un micrófono estático cada vez que tenga que introducir algún dato en el cuaderno.
- Elaboración de un código más inteligente que incorpore la posibilidad de alternar el uso de un reconocedor u otro en función de la disponibilidad a Internet y su velocidad, comprobando ambos aspectos de manera periódica.

6 BIBLIOGRAFÍA

- [1] «Apple,» [En línea]. Available: <https://www.apple.com/es/siri/>. [Último acceso: 3 Marzo 2021].
- [2] «Amazon,» [En línea]. Available: <https://developer.amazon.com/es-ES/alexa>. [Último acceso: 3 Marzo 2021].
- [3] «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Asistente_virtual. [Último acceso: 31 Enero 2021].
- [4] «Equipo Blog Grupo Cajamar,» 10 Enero 2019. [En línea]. Available: <https://blog.grupocajamar.com/que-son-los-asistentes-virtuales-inteligentes/>. [Último acceso: 31 Enero 2021].
- [5] A. Palacios, «slideshare,» 2 Noviembre 2019. [En línea]. Available: <https://es.slideshare.net/MariaAlejandra348/historia-de-los-asistentes-virtuales>. [Último acceso: 31 Enero 2021].
- [6] «IBM,» [En línea]. Available: https://www.ibm.com/ibm/history/exhibits/specialprod1/specialprod1_7.html. [Último acceso: 14 Febrero 2021].
- [7] E. Naone, «Technology Review,» 24 Febrero 2009. [En línea]. Available: <http://www2.technologyreview.com/news/412191/tr10-intelligent-software-assistant/>. [Último acceso: 31 Enero 2021].
- [8] C. Cruz, «Blog think big,» 25 Abril 2019. [En línea]. Available: <https://blogthinkbig.com/asistentes-virtuales-usuarios>. [Último acceso: 31 Enero 2021].
- [9] «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Google_Now. [Último acceso: 3 Marzo 2021].
- [10] «Google,» [En línea]. Available: https://assistant.google.com/intl/es_es/. [Último acceso: 3 Marzo 2021].
- [11] J. A. P. Estapé, «Computer Hoy,» 5 Agosto 2015. [En línea]. Available: <https://computerhoy.com/noticias/software/que-es-como-funciona-cortana-windows-10-32429>.

[Último acceso: 3 Marzo 2021].

- [12] «Innovación y Tecnología,» [En línea]. Available: <https://innovacion-tecnologia.com/iot/que-es-alexa-google-home-y-siri-comparativa-y-diferencias/>. [Último acceso: 14 Febrero 2021].
- [13] «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Amazon_Echo. [Último acceso: 3 Marzo 2021].
- [14] J. P. P. y. A. Gardey, «Definición de,» 2013. [En línea]. Available: <https://definicion.de/bitacora/>. [Último acceso: 3 Febrero 2021].
- [15] «Wikipedia,» [En línea]. Available: <https://es.wikipedia.org/wiki/Bit%C3%A1cora>. [Último acceso: 14 Febrero 2021].
- [16] «Más de historia,» 27 Marzo 2009. [En línea]. Available: <http://masdehistoria.blogspot.com/2009/03/cuaderno-de-bitacora.html>. [Último acceso: 3 Febrero 2021].
- [17] J. A. Tomás, «Boletín Patrón,» [En línea]. Available: <http://boletinpatron.com/diario-de-navegacion-o-cuaderno-de-bitacora/>. [Último acceso: 3 Febrero 2021].
- [18] «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Reconocimiento_del_habla. [Último acceso: 16 Febrero 2021].
- [19] «Vosk,» [En línea]. Available: <https://alphacephei.com/vosk>. [Último acceso: 3 Marzo 2021].
- [20] «GitHub,» Sphinx, [En línea]. Available: <https://cmusphinx.github.io/>. [Último acceso: 3 Marzo 2021].
- [21] «Google,» [En línea]. Available: <https://cloud.google.com/speech-to-text?hl=es>. [Último acceso: 3 Marzo 2021].
- [22] «Rock Content,» [En línea]. Available: <https://rockcontent.com/es/blog/que-es-un-lenguaje-de-programacion/>. [Último acceso: 10 Febrero 2021].
- [23] «Octoverse,» GitHub, [En línea]. Available: <https://octoverse.github.com/>. [Último acceso: 10 Febrero 2021].
- [24] «Tokio School,» [En línea]. Available: <https://www.tokioschool.com/noticias/historia-python/>. [Último acceso: 3 Febrero 2021].
- [25] «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Historia_de_Python. [Último acceso: 3 Febrero 2021].
- [26] GitHub, [En línea]. Available: <https://gvanrossum.github.io/>. [Último acceso: 14 Febrero 2021].
- [27] «Cosas de devs,» [En línea]. Available: <https://cosasdedevs.com/python/>. [Último acceso: 14 Febrero 2021].
- [28] «Aritmetics,» [En línea]. Available: <https://www.arimetrics.com/glosario-digital/entorno-de-desarrollo>. [Último acceso: 10 Febrero 2021].
- [29] «EcuRed,» [En línea]. Available: https://www.ecured.cu/Visual_Studio_Code. [Último acceso: 10 Febrero 2021].
- [30] «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Visual_Studio_Code.

[Último acceso: 14 Febrero 2021].

- [31] «StackOverflow,» [En línea]. Available: <https://stackoverflow.com/questions/14257598/what-are-language-codes-in-chromes-implementation-of-the-html5-speech-recogniti/14302134#14302134>. [Último acceso: 20 Enero 2021].
- [32] «Alphacephei,» [En línea]. Available: <https://alphacephei.com/vosk/models>. [Último acceso: 10 Febrero 2021].
- [33] «Python,» [En línea]. Available: <https://www.python.org/>. [Último acceso: 25 Febrero 2021].
- [34] «Visual Studio Code,» [En línea]. Available: <https://code.visualstudio.com/>. [Último acceso: 8 Marzo 2021].
- [35] «StackOverflow,» [En línea]. Available: <https://stackoverflow.com/>. [Último acceso: 27 Febrero 2021].
- [36] «Geeks for Geeks,» [En línea]. Available: <https://www.geeksforgeeks.org/voice-assistant-using-python/>. [Último acceso: 19 Enero 2021].
- [37] «Xlsxwriter,» [En línea]. Available: <https://xlsxwriter.readthedocs.io/>. [Último acceso: 7 Febrero 2021].
- [38] «GitHub,» [En línea]. Available: <https://github.com/>. [Último acceso: 16 Febrero 2021].
- [39] «Alphacephei,» [En línea]. Available: <https://alphacephei.com/vosk/>. [Último acceso: 12 Febrero 2021].
- [40] «Raspberry,» [En línea]. Available: <https://www.raspberrypi.org/>. [Último acceso: 10 Marzo 2021].
- [41] «Arduino,» [En línea]. Available: <https://www.arduino.cc/>. [Último acceso: 10 Marzo 2021].
- [42] «Web de La Moncloa,» [En línea]. Available: <http://www.lamoncloa.gob.es>. [Último acceso: 13 enero 2015].
- [43] J. Rodríguez y V. Fernández, Cómo redactar el estado del arte de un trabajo, Editorial Genios, 2010.
- [44] P. Martínez y A. García, Cómo escribir una buena memoria de TFG, Publicaciones del 2000, 2013.
- [45] A. Pérez, Cómo escribir una bibliografía, Nuevas publicaciones.

ANEXO I: VERSIÓN DE GOOGLE

En este Anexo se adjuntará el código en formato imagen con la versión de Google.

```
1  from math import e
2  from operator import mod
3  import os
4  import speech_recognition as sr
5  import pyttsx3
6  import xlswriter
7  import datetime
8
9  os.chdir('C:/Users/josea/OneDrive/Escritorio/TFG/Documentos')
10
11 fecha = datetime.datetime.now()
12 ficheroExcel="Cuaderno de bitácora del {}-{} a las {} horas.xlsx".format(fecha.day,fecha.month,fecha.hour)
13 ficheroTxt="Cuaderno de bitácora del {}-{} a las {} horas.txt".format(fecha.day,fecha.month,fecha.hour)
14
15 # Esta fundión permite convertir text to speech
16 def SpeakText(command):
17
18     engine = pyttsx3.init()
19     engine.say(command)
20     engine.runAndWait()
21
22 def respuesta():
23     r2=sr.Recognizer()
24     with sr.Microphone(device_index=1) as source2:
25         audio2=r2.listen(source2)
26
27         try:
28             pregunta=r2.recognize_google(audio2,language='es-ES')
29             print(f"Has dicho:{pregunta}\n")
30         except Exception as e:
31             return "None"
32             SpeakText("Disculpe,¿Puede repetir?")
33     return pregunta
34
35 def takeCommand():
36     r=sr.Recognizer()
37     with sr.Microphone(device_index=1) as source:
```

Figura A1-1: Versión del código con reconocedor de Google 1/23

```
38         print("Escuchando")
39         audio = r.listen(source)
40
41         try:
42             statement=r.recognize_google(audio,language='es-ES')
43             print(f"Has dicho:{statement}\n")
44         except Exception as e:
45             SpeakText("Disculpe, ¿Puede repetir?")
46             return "None"
47         return statement
48
49 workbook = xlswriter.Workbook(ficheroExcel)
50 worksheet1 = workbook.add_worksheet("cronica")
51 worksheet1.write(0, 0,"Hora" )
52 worksheet1.write(0, 1,"Millas totales")
53 worksheet1.write(0, 2,"Millas navegadas")
54 worksheet1.write(0, 3,"Revoluciones")
55 worksheet1.write(0, 4,"Rumbo de aguja" )
56 worksheet1.write(0, 5,"Rumbo verdadero")
57 worksheet1.write(0, 6,"Latitud")
58 worksheet1.write(0, 7,"Longitud")
59 worksheet1.write(0, 8,"Acaecimientos")
60 worksheet1.write(0, 9,"Acaecimientos")
61
62 worksheet2 = workbook.add_worksheet("meteo")
63 worksheet2.write(0, 0,"Hora" )
64 worksheet2.write(0, 1,"Dirección del viento")
65 worksheet2.write(0, 2,"Velocidad del viento")
66 worksheet2.write(0, 3,"Visibilidad")
67 worksheet2.write(0, 4,"Barómetro" )
68 worksheet2.write(0, 5,"Bola seca")
69 worksheet2.write(0, 6,"Bola húmeda")
70 worksheet2.write(0, 7,"Cantidad de nubes")
71 worksheet2.write(0, 8,"Clase de nubes")
72 worksheet2.write(0, 9,"Estado de la mar")
73 worksheet2.write(0, 10,"Estado de la mar")
74
75
```

Figura A1-2: Versión del código con reconocedor de Google 2/23

```

75
76 worksheet3 = workbook.add_worksheet("máquinas")
77 ronda=datetime.datetime.now().hour
78 if ronda>=23 or ronda<3:
79     worksheet3.write(1,0,"0000")
80 if ronda>=3 and ronda<7:
81     worksheet3.write(1,0,"0400")
82 if ronda>=7 and ronda<11:
83     worksheet3.write(1,0,"0800")
84 if ronda>=11 and ronda<15:
85     worksheet3.write(1,0,"1200")
86 if ronda>=15 and ronda<19:
87     worksheet3.write(1,0,"1600")
88 if ronda>=19 and ronda<23:
89     worksheet3.write(1,0,"2000")
90 worksheet3.write(0, 1,"Revoluciones Br")
91 worksheet3.write(0, 2,"Revoluciones Er")
92 worksheet3.write(0, 3,"Gasolina consumida")
93 worksheet3.write(0, 4,"Gasolina restante")
94 worksheet3.write(0, 5,"Aceite consumido")
95 worksheet3.write(0, 6,"Aceite restante")
96 worksheet3.write(0, 7,"Agua consumida")
97 worksheet3.write(0, 8,"Agua producida")
98 worksheet3.write(0, 9,"Agua restante")
99 worksheet3.write(0, 10,"Agua restante")
100
101 worksheet4 = workbook.add_worksheet("Ronda mayor")
102 worksheet4.write(0, 0,"Pañol")
103 worksheet4.write(0, 1,"Temperatura máxima")
104 worksheet4.write(0, 2,"Temperatura mínima")
105 worksheet4.write(0, 3,"Humedad relativa")
106 worksheet4.write(0, 4,"Humedad relativa")
107
108
109 cell_format = workbook.add_format({'bold': True, 'center_across':True,'border':2})
110 cell_format1 = workbook.add_format({'border':1})
111 worksheet1.set_row(0,None, cell_format)

```

Figura A1-3: Versión del código con reconocedor de Google 3/23

```

112 worksheet2.set_row(0,None, cell_format)
113 worksheet3.set_row(0,None, cell_format)
114 worksheet4.set_row(0,None, cell_format)
115 worksheet1.set_column(8, 8, 100,cell_format1)
116 worksheet1.set_column(1, 7, 20,cell_format1)
117 worksheet2.set_column(1, 9, 20,cell_format1)
118 worksheet3.set_column(1, 9, 20)
119 worksheet3.set_row(1, None,cell_format1)
120 worksheet4.set_column(0, 3, 18,cell_format1)
121 worksheet1.set_column(0,0,None,cell_format1)
122 worksheet2.set_column(0,0,None,cell_format1)
123
124 mt = 1
125 mn = 1
126 rev = 1
127 ra = 1
128 rv = 1
129 lat = 1
130 long = 1
131 aca = 1
132 dv = 1
133 vv = 1
134 v = 1
135 bar = 1
136 bs = 1
137 bh = 1
138 cn = 1
139 cn2 = 1
140 m = 1
141 revbr = 1
142 rever = 1
143 gc = 1
144 gr = 1
145 ac = 1
146 ar = 1
147 agc = 1
148 agp = 1

```

Figura A1-4: Versión del código con reconocedor de Google 4/23

```
149 agr = 1
150 p = 1
151 ma = 1
152 mi = 1
153 h = 1
154 t1 = 1
155 t2 = 1
156
157
158
159
160 SpeakText ("Iniciando Cuaderno de Bitácora")
161 SpeakText ("Puede empezar a hablar ahora")
162
163
164 if __name__ == '__main__':
165     while True:
166         statement = takeCommand().lower()
167
168         if statement == 0:
169             continue
170
171         if 'millas totales' in statement:
172             SpeakText(f"Ha dicho {statement}")
173             SpeakText("¿Desea confirmar?")
174             print ("esperando confirmación")
175             pregunta=respuesta().lower()
176             if 'afirmativo' in pregunta:
177                 statement1=statement.replace("millas totales","")
178                 try:
179                     float(statement1)
180                     worksheet1.write (mt,1,f"{statement1} NM")
181                     now=datetime.datetime.now()
182                     tiempo="{:}:{:}".format(now.hour,now.minute)
183                     worksheet1.write(t1,0,tiempo)
184                     mt= mt+1
185                     t1=t1+1
```

Figura A1-5: Versión del código con reconocedor de Google 5/23

```
186         except:
187             SpeakText("Número no reconocido")
188         if 'negativo' in pregunta:
189             SpeakText("Repita")
190
191         if 'millas navegadas' in statement:
192             SpeakText(f"Ha dicho {statement}")
193             SpeakText("¿Desea confirmar?")
194             print ("esperando confirmación")
195             pregunta=respuesta().lower()
196             if 'afirmativo' in pregunta:
197                 statement1=statement.replace("millas navegadas","")
198                 try:
199                     float(statement1)
200                     worksheet1.write(mn,2,f"{statement1} NM")
201                     mn=mn+1
202             except:
203                 SpeakText("Número no reconocido")
204         if 'negativo' in pregunta:
205             SpeakText("Repita")
206
207         if 'revoluciones del eje' in statement:
208             SpeakText(f"Ha dicho {statement}")
209             SpeakText("¿Desea confirmar?")
210             print ("esperando confirmación")
211             pregunta=respuesta().lower()
212
213             if 'afirmativo' in pregunta:
214                 statement1=statement.replace("revoluciones del eje","")
215                 try:
216                     float(statement1)
217                     worksheet1.write(rev,3,f"{statement1} rpm")
218                     rev=rev+1
219                 except:
220                     SpeakText("Número no reconocido")
221             if 'negativo' in pregunta:
222                 SpeakText("Repita")
```

Figura A1-6: Versión del código con reconocedor de Google 6/23

```

223
224     if 'rumbo de aguja' in statement:
225         SpeakText(f"Ha dicho {statement}")
226         SpeakText("¿Desea confirmar?")
227         print ("esperando confirmación")
228         pregunta=respuesta().lower()
229
230     if 'afirmativo' in pregunta:
231         statement1=statement.replace("rumbo de aguja","")
232         try:
233             float(statement1)
234             worksheet1.write(ra,4,statement1)
235             ra=ra+1
236         except:
237             SpeakText("Número no reconocido")
238
239     if 'negativo' in pregunta:
240         SpeakText("Repita")
241
242     if 'rumbo verdadero' in statement:
243         SpeakText(f"Ha dicho {statement}")
244         SpeakText("¿Desea confirmar?")
245         print ("esperando confirmación")
246         pregunta=respuesta().lower()
247
248     if 'afirmativo' in pregunta:
249         statement1=statement.replace("rumbo verdadero","")
250         try:
251             float(statement1)
252             worksheet1.write(rv,5,statement1)
253             rv=rv+1
254         except:
255             SpeakText("Número no reconocido")
256
257     if 'negativo' in pregunta:
258         SpeakText("Repita")
259

```

Figura A1-7: Versión del código con reconocedor de Google 7/23

```

260     if 'latitud' in statement:
261         SpeakText(f"Ha dicho {statement}")
262         SpeakText("¿Desea confirmar?")
263         print ("esperando confirmación")
264         pregunta=respuesta().lower()
265
266     if 'afirmativo' in pregunta:
267         statement1=statement.replace("latitud","")
268         statement1=statement1.replace("grados","°")
269         statement1=statement1.replace("minutos","'")
270         statement1=statement1.replace("norte","N")
271         statement1=statement1.replace("sur","S")
272         worksheet1.write(lat,6,statement1)
273         lat=lat+1
274     if 'negativo' in pregunta:
275         SpeakText("Repita")
276
277     if 'longitud' in statement:
278         SpeakText(f"Ha dicho {statement}")
279         SpeakText("¿Desea confirmar?")
280         print ("esperando confirmación")
281         pregunta=respuesta().lower()
282
283     if 'afirmativo' in pregunta:
284         statement1 = statement.replace("longitud","")
285         statement1=statement1.replace("grados","°")
286         statement1=statement1.replace("minutos","'")
287         statement1=statement1.replace("este","E")
288         statement1=statement1.replace("oeste","W")
289         worksheet1.write(long,7,statement1)
290         long=long+1
291     if 'negativo' in pregunta:
292         SpeakText("Repita")
293
294     if 'acaecimiento' in statement:
295         SpeakText(f"Ha dicho {statement}")
296         SpeakText("¿Desea confirmar?")

```

Figura A1-8: Versión del código con reconocedor de Google 8/23

```
297     print ("esperando confirmación")
298     pregunta=respuesta().lower()
299
300     if 'afirmativo' in pregunta:
301         statement1 = statement.replace("acaecimiento","")
302         worksheet1.write(aca,8,statement1)
303         aca = aca+1
304     if 'negativo' in pregunta:
305         SpeakText("Repita")
306
307     if 'dirección del viento' in statement:
308         SpeakText(f"Ha dicho {statement}")
309         SpeakText("¿Desea confirmar?")
310         print ("esperando confirmación")
311         pregunta=respuesta().lower()
312
313         if 'afirmativo' in pregunta:
314             statement1=statement.replace("dirección del viento","")
315             try:
316                 float(statement1)
317                 worksheet2.write(dv,1,statement1)
318                 now=datetime.datetime.now()
319                 tiempo="{:}:{:}".format(now.hour,now.minute)
320                 worksheet2.write(t2,0,tiempo)
321                 t2=t2+1
322                 dv=dv+1
323             except:
324                 SpeakText("Número no reconocido")
325
326         if 'negativo' in pregunta:
327             SpeakText("Repita")
328
329     if 'velocidad del viento' in statement:
330         SpeakText(f"Ha dicho {statement}")
331         SpeakText("¿Desea confirmar?")
332         print ("esperando confirmación")
333         pregunta=respuesta().lower()
```

Figura A1-9: Versión del código con reconocedor de Google 9/23

```
334
335     if 'afirmativo' in pregunta:
336         statement1=statement.replace("velocidad del viento","")
337         try:
338             float(statement1)
339             worksheet2.write(vv,2,f"{statement1} kts")
340             vv=vv+1
341         except:
342             SpeakText("Número no reconocido")
343
344     if 'negativo' in pregunta:
345         SpeakText("Repita")
346
347     if 'visibilidad actual' in statement:
348         SpeakText(f"Ha dicho {statement}")
349         SpeakText("¿Desea confirmar?")
350         print ("esperando confirmación")
351         pregunta=respuesta().lower()
352
353         if 'afirmativo' in pregunta:
354             statement1=statement.replace("visibilidad actual","")
355             try:
356                 float(statement1)
357                 worksheet2.write(v,3,statement1)
358                 v=v+1
359             except:
360                 SpeakText("Número no reconocido")
361
362         if 'negativo' in pregunta:
363             SpeakText("Repita")
364
365     if 'barómetro' in statement:
366         SpeakText(f"Ha dicho {statement}")
367         SpeakText("¿Desea confirmar?")
368         print ("esperando confirmación")
369         pregunta=respuesta().lower()
370
```

Figura A1-10: Versión del código con reconocedor de Google 10/23

```

371         if 'afirmativo' in pregunta:
372             statement1=statement.replace("barómetro","")
373             try:
374                 float(statement1)
375                 worksheet2.write(bar,4,f"{statement1} hPa")
376                 bar=bar+1
377             except:
378                 SpeakText("Número no reconocido")
379
380         if 'negativo' in pregunta:
381             SpeakText("Repita")
382
383         if 'bola seca' in statement:
384             SpeakText(f"Ha dicho {statement}")
385             SpeakText("¿Desea confirmar?")
386             print ("esperando confirmación")
387             pregunta=respuesta().lower()
388
389         if 'afirmativo' in pregunta:
390             statement1=statement.replace("bola seca","")
391             try:
392                 float(statement1)
393                 worksheet2.write(bs,5,statement1)
394                 bs=bs+1
395             except:
396                 SpeakText("Número no reconocido")
397
398         if 'negativo' in pregunta:
399             SpeakText("Repita")
400
401         if 'bola húmeda' in statement:
402             SpeakText(f"Ha dicho {statement}")
403             SpeakText("¿Desea confirmar?")
404             print ("esperando confirmación")
405             pregunta=respuesta().lower()
406
407         if 'afirmativo' in pregunta:

```

Figura A1-11: Versión del código con reconocedor de Google 11/23

```

408             statement1=statement.replace("bola húmeda","")
409             try:
410                 float(statement1)
411                 worksheet2.write(bh,6,statement1)
412                 h=bh+1
413             except:
414                 SpeakText("Número no reconocido")
415
416         if 'negativo' in pregunta:
417             SpeakText("Repita")
418
419         if 'cantidad de nubes' in statement:
420             SpeakText(f"Ha dicho {statement}")
421             SpeakText("¿Desea confirmar?")
422             print ("esperando confirmación")
423             pregunta=respuesta().lower()
424
425         if 'afirmativo' in pregunta:
426             statement1=statement.replace("cantidad de nubes","")
427             try:
428                 float(statement1)
429                 worksheet2.write(cn,7,f"{statement1} octas")
430                 cn=cn+1
431             except:
432                 SpeakText("Número no reconocido")
433
434         if 'negativo' in pregunta:
435             SpeakText("Repita")
436
437         if 'clase de nubes' in statement:
438             SpeakText(f"Ha dicho {statement}")
439             SpeakText("¿Desea confirmar?")
440             print ("esperando confirmación")
441             pregunta=respuesta().lower()
442
443         if 'afirmativo' in pregunta:
444             statement1=statement.replace("clase de nubes","")

```

Figura A1-12: Versión del código con reconocedor de Google 12/23

```
445         worksheet2.write(cn2,8,statement1)
446         cn2=cn2+1
447     if 'negativo' in pregunta:
448         SpeakText("Repita")
449
450     if 'estado de la mar' in statement:
451         SpeakText(f"Ha dicho {statement}")
452         SpeakText("¿Desea confirmar?")
453         print ("esperando confirmación")
454         pregunta=respuesta().lower()
455
456     if 'afirmativo' in pregunta:
457         statement1=statement.replace("estado de la mar","")
458         worksheet2.write(m,9,statement1)
459         m=m+1
460     if 'negativo' in pregunta:
461         SpeakText("Repita")
462
463     if 'motor babor' in statement:
464         SpeakText(f"Ha dicho {statement}")
465         SpeakText("¿Desea confirmar?")
466         print ("esperando confirmación")
467         pregunta=respuesta().lower()
468
469     if 'afirmativo' in pregunta:
470         statement1=statement.replace("motor babor","")
471         try:
472             float(statement1)
473             worksheet3.write(revbr,1,f"{statement1} rpm")
474             revbr=revbr+1
475         except:
476             SpeakText("Número no reconocido")
477
478     if 'negativo' in pregunta:
479         SpeakText("Repita")
480
481     if 'motor estribor' in statement:
```

Figura A1-13: Versión del código con reconocedor de Google 13/23

```
482     SpeakText(f"Ha dicho {statement}")
483     SpeakText("¿Desea confirmar?")
484     print ("esperando confirmación")
485     pregunta=respuesta().lower()
486
487     if 'afirmativo' in pregunta:
488         statement1=statement.replace("motor estribor","")
489         try:
490             float(statement1)
491             worksheet3.write(rever,2,f"{statement1} rpm")
492             rever=rever+1
493         except:
494             SpeakText("Número no reconocido")
495
496     if 'negativo' in pregunta:
497         SpeakText("Repita")
498
499     if 'gasolina consumida' in statement:
500         SpeakText(f"Ha dicho {statement}")
501         SpeakText("¿Desea confirmar?")
502         print ("esperando confirmación")
503         pregunta=respuesta().lower()
504
505     if 'afirmativo' in pregunta:
506         statement1=statement.replace("gasolina consumida","")
507         try:
508             float(statement1)
509             worksheet3.write(gc,3,f"{statement1} L")
510             gc = gc+1
511         except:
512             SpeakText("Número no reconocido")
513
514     if 'negativo' in pregunta:
515         SpeakText("Repita")
516
517     if 'gasolina restante' in statement:
518         SpeakText(f"Ha dicho {statement}")
```

Figura A1-14: Versión del código con reconocedor de Google 14/23

```

519 SpeakText("¿Desea confirmar?")
520 print ("esperando confirmación")
521 pregunta=respuesta().lower()
522
523 if 'afirmativo' in pregunta:
524     statement1=statement.replace("gasolina restante","")
525     try:
526         float(statement1)
527         worksheet3.write(gr,4,f"{statement1} L")
528         gr=gr+1
529     except:
530         SpeakText("Número no reconocido")
531
532 if 'negativo' in pregunta:
533     SpeakText("Repita")
534
535 if 'aceite consumido' in statement:
536     SpeakText(f"Ha dicho {statement}")
537     SpeakText("¿Desea confirmar?")
538     print ("esperando confirmación")
539     pregunta=respuesta().lower()
540
541 if 'afirmativo' in pregunta:
542     statement1=statement.replace("aceite consumido","")
543     try:
544         float(statement1)
545         worksheet3.write(ac,5,f"{statement1} L")
546         ac=ac+1
547     except:
548         SpeakText("Número no reconocido")
549
550 if 'negativo' in pregunta:
551     SpeakText("Repita")
552
553 if 'aceite restante' in statement:
554     SpeakText(f"Ha dicho {statement}")
555     SpeakText("¿Desea confirmar?")

```

Figura A1-15: Versión del código con reconocedor de Google 15/23

```

556 print ("esperando confirmación")
557 pregunta=respuesta().lower()
558
559 if 'afirmativo' in pregunta:
560     statement1=statement.replace("aceite restante","")
561     try:
562         float(statement1)
563         worksheet3.write(ar,6,f"{statement1} L")
564         ar=ar+1
565     except:
566         SpeakText("Número no reconocido")
567
568 if 'negativo' in pregunta:
569     SpeakText("Repita")
570
571 if 'agua consumida' in statement:
572     SpeakText(f"Ha dicho {statement}")
573     SpeakText("¿Desea confirmar?")
574     print ("esperando confirmación")
575     pregunta=respuesta().lower()
576
577 if 'afirmativo' in pregunta:
578     statement1=statement.replace("agua consumida","")
579     try:
580         float(statement1)
581         worksheet3.write(agc,7,f"{statement1} L")
582         agc=agc+1
583     except:
584         SpeakText("Número no reconocido")
585
586 if 'negativo' in pregunta:
587     SpeakText("Repita")
588
589 if 'agua producida' in statement:
590     SpeakText(f"Ha dicho {statement}")
591     SpeakText("¿Desea confirmar?")
592     print ("esperando confirmación")

```

Figura A1-16: Versión del código con reconocedor de Google 16/23

```
593 pregunta=respuesta().lower()
594
595 if 'afirmativo' in pregunta:
596     statement1=statement.replace("agua producida","")
597     try:
598         float(statement1)
599         worksheet3.write(agp,8,f"{statement1} L")
600         agp=agp+1
601     except:
602         SpeakText("Número no reconocido")
603
604 if 'negativo' in pregunta:
605     SpeakText("Repita")
606
607 if 'agua restante' in statement:
608     SpeakText(f"Ha dicho {statement}")
609     SpeakText("¿Desea confirmar?")
610     print ("esperando confirmación")
611     pregunta=respuesta().lower()
612
613 if 'afirmativo' in pregunta:
614     statement1=statement.replace("agua restante","")
615     try:
616         float(statement1)
617         worksheet3.write(agr,9,f"{statement1} L")
618         agr=agr+1
619     except:
620         SpeakText("Número no reconocido")
621 if 'negativo' in pregunta:
622     SpeakText("Repita")
623
624
625 if 'compartimento' in statement:
626
627     SpeakText(f"Ha dicho {statement}")
628     SpeakText("¿Desea confirmar?")
629     print ("esperando confirmación")
```

Figura A1-17: Versión del código con reconocedor de Google 17/23

```
630 pregunta=respuesta().lower()
631
632 if 'afirmativo' in pregunta:
633     statement1=statement.replace("compartimento","pañol")
634     worksheet4.write(p,0,statement1)
635     p=p+1
636 if 'negativo' in pregunta:
637     SpeakText("Repita")
638
639 if 'temperatura máxima' in statement:
640     SpeakText(f"Ha dicho {statement}")
641     SpeakText("¿Desea confirmar?")
642     print ("esperando confirmación")
643     pregunta=respuesta().lower()
644
645 if 'afirmativo' in pregunta:
646     statement1=statement.replace("temperatura máxima","")
647     try:
648         float(statement1)
649         worksheet4.write(ma,1,f"{statement1}°C")
650         ma=ma+1
651     except:
652         SpeakText("Número no reconocido")
653
654 if 'negativo' in pregunta:
655     SpeakText("Repita")
656
657 if 'temperatura mínima' in statement:
658     SpeakText(f"Ha dicho {statement}")
659     SpeakText("¿Desea confirmar?")
660     print ("esperando confirmación")
661     pregunta=respuesta().lower()
662
663 if 'afirmativo' in pregunta:
664     statement1=statement.replace("temperatura mínima","")
665     try:
666         float(statement1)
```

Figura A1-18: Versión del código con reconocedor de Google 18/23

```

667         worksheet4.write(mi,2,f"{statement1}%C")
668         mi=mi+1
669     except:
670         SpeakText("Número no reconocido")
671
672     if 'negativo' in pregunta:
673         SpeakText("Repita")
674
675     if 'humedad' in statement:
676         SpeakText(f"Ha dicho {statement}")
677         SpeakText("¿Desea confirmar?")
678         print ("esperando confirmación")
679         pregunta=respuesta().lower()
680
681     if 'afirmativo' in pregunta:
682         statement1=statement.replace("humedad", "")
683         try:
684             float(statement1)
685             worksheet4.write(h,3,f"{statement1}%")
686             h=h+1
687         except:
688             SpeakText("Número no reconocido")
689
690     if 'negativo' in pregunta:
691         SpeakText("Repita")
692
693     if 'crónica' in statement:
694         with open (fichero.txt,mode='a') as file:
695             SpeakText(f"Ha dicho {statement}")
696             SpeakText("¿Desea confirmar?")
697             print ("esperando confirmación")
698             pregunta=respuesta().lower()
699             if 'afirmativo' in pregunta:
700                 statement2=statement.replace("crónica ", "")
701                 statement2=statement2.replace("crónica", "")
702                 file.write (f" {statement2}.")
703             if 'negativo' in pregunta:

```

Figura A1-19: Versión del código con reconocedor de Google 19/23

```

704         SpeakText("Repita")
705
706     if 'guardar' in statement:
707         SpeakText("¿Esta seguro de que desea guardar?")
708         print ("esperando confirmación")
709         pregunta=respuesta().lower()
710         if 'afirmativo' in pregunta:
711             workbook.close()
712             SpeakText("Guardando su cuaderno de bitácora")
713             SpeakText("Hasta la próxima")
714             break
715         if 'negativo' in pregunta:
716             SpeakText("Repita")
717
718     if 'consultar' in statement:
719         consulta=False
720         SpeakText ("Indique sobre qué columna desea saber en qué fila se encuentra")
721         SpeakText("Si por el contrario le gustaría saber el estado de una hoja diga el nombre de la hoja")
722         SpeakText("Si quiere cancelar diga cancelar")
723         print("esperando")
724         while not consulta:
725             pregunta=respuesta().lower()
726             if 'cancelar' in pregunta:
727                 consulta=True
728                 SpeakText("Opción cancelada")
729             if 'millas totales' in pregunta:
730                 SpeakText(f"La columna de millas totales se encuentra en la fila {mt}")
731             if 'millas navegadas' in pregunta:
732                 SpeakText(f"La columna de millas navegadas se encuentra en la fila {mn}")
733             if 'revoluciones del eje' in pregunta:
734                 SpeakText(f"La columna de revoluciones del eje se encuentra en la fila {rev}")
735             if 'rumbo de aguja' in pregunta:
736                 SpeakText(f"La columna de rumbo de aguja se encuentra en la fila {ra}")
737             if 'rumbo verdadero' in pregunta:
738                 SpeakText(f"La columna de rumbo verdadero se encuentra en la fila {rv}")
739
740

```

Figura A1-20: Versión del código con reconocedor de Google 20/23

```
741 if 'latitud' in pregunta:
742     SpeakText(f"La columna de latitud se encuentra en la fila {lat}")
743 if 'longitud' in pregunta:
744     SpeakText(f"La columna de longitud se encuentra en la fila {long}")
745 if 'acaecimiento' in pregunta:
746     SpeakText(f"La columna de acaecimiento se encuentra en la fila {aca}")
747 if 'dirección del viento' in pregunta:
748     SpeakText(f"La columna de dirección del viento se encuentra en la fila {dv}")
749 if 'velocidad del viento' in pregunta:
750     SpeakText(f"La columna de velocidad del viento se encuentra en la fila {vv}")
751 if 'visibilidad' in pregunta:
752     SpeakText(f"La columna de visibilidad se encuentra en la fila {v}")
753 if 'barómetro' in pregunta:
754     SpeakText(f"La columna de barómetro se encuentra en la fila {bar}")
755 if 'bola seca' in pregunta:
756     SpeakText(f"La columna de bola seca se encuentra en la fila {bs}")
757 if 'bola húmeda' in pregunta:
758     SpeakText(f"La columna de bola húmeda se encuentra en la fila {bh}")
759 if 'cantidad de nubes' in pregunta:
760     SpeakText(f"La columna de cantidad de nubes se encuentra en la fila {cn}")
761 if 'clase de nubes' in pregunta:
762     SpeakText(f"La columna de clase de nubes se encuentra en la fila {cn2}")
763 if 'estado de la mar' in pregunta:
764     SpeakText(f"La columna de estado de la mar se encuentra en la fila {m}")
765 if 'motor babor' in pregunta:
766     SpeakText(f"La columna de motor babor se encuentra en la fila {revbr}")
767 if 'motor estribor' in pregunta:
768     SpeakText(f"La columna de motor estribor se encuentra en la fila {rever}")
769 if 'gasolina consumida' in pregunta:
770     SpeakText(f"La columna de gasolina consumida se encuentra en la fila {gc}")
771 if 'gasolina restante' in pregunta:
772     SpeakText(f"La columna de gasolina restante se encuentra en la fila {gr}")
773 if 'aceite consumido' in pregunta:
774     SpeakText(f"La columna de aceite consumido se encuentra en la fila {ac}")
775 if 'aceite restante' in pregunta:
776     SpeakText(f"La columna de aceite restante se encuentra en la fila {ar}")
777 if 'agua consumida' in pregunta:
```

Figura A1-21: Versión del código con reconocedor de Google 21/23

```
778     SpeakText(f"La columna de agua consumida se encuentra en la fila {agc}")
779 if 'agua producida' in pregunta:
780     SpeakText(f"La columna de agua producida se encuentra en la fila {agp}")
781 if 'agua restante' in pregunta:
782     SpeakText(f"La columna de agua restante se encuentra en la fila {agr}")
783 if 'compartimento' in pregunta:
784     SpeakText(f"La columna de compartimento se encuentra en la fila {p}")
785 if 'temperatura máxima' in pregunta:
786     SpeakText(f"La columna de temperatura máxima se encuentra en la fila {ma}")
787 if 'temperatura mínima' in pregunta:
788     SpeakText(f"La columna de temperatura mínima se encuentra en la fila {mi}")
789 if 'humedad' in pregunta:
790     SpeakText(f"La columna de humedad se encuentra en la fila {h}")
791 if 'hoja de crónica' in pregunta:
792     if mt==mn==rev==ra==rv==lat==long==aca:
793         SpeakText("Toda la hoja se encuentra en la misma fila")
794     else:
795         SpeakText("Una columna no se encuentra en la misma fila")
796 if 'hoja de meteorología' in pregunta:
797     if dv==vv==v==bar==bh==bs==cn==cn2==m:
798         SpeakText("Toda la hoja se encuentra en la misma fila")
799     else:
800         SpeakText("Una columna no se encuentra en la misma fila")
801 if 'hoja de máquinas' in pregunta:
802     if revbr==rever==gc==gr==ac==agc==agp==agr:
803         SpeakText("Toda la hoja se encuentra en la misma fila")
804     else:
805         SpeakText("Una columna no se encuentra en la misma fila")
806 if 'hoja de ronda mayor' in pregunta:
807     if p==ma==mi==h:
808         SpeakText("Toda la hoja se encuentra en la misma fila")
809     else:
810         SpeakText("Una columna no se encuentra en la misma fila")
811
812 if 'la actitud' in statement:
813     SpeakText(f"Ha dicho {statement}")
814     SpeakText("¿Desea confirmar?")
```

Figura A1-22: Versión del código con reconocedor de Google 22/23

```
814     SpeakText("¿Desea confirmar?")
815     print ("esperando confirmación")
816     pregunta=respuesta().lower()
817
818     if 'afirmativo' in pregunta:
819         Latitud()
820         lat=lat+1
821         SpeakText("Escrito correctamente")
822     if 'negativo' in pregunta:
823         SpeakText("Repita")
```

Figura A1-23: Versión del código con reconocedor de Google 23/23

ANEXO II: VERSIÓN DE VOSK

En este Anexo se adjuntará el código mediante imágenes de la versión de Vosk.

```
1 from vosk import Model, KaldiRecognizer
2 import os
3 import json
4 import pyaudio
5 import xlswriter
6 import shutil
7 import datetime
8 import pyttsx3
9 from spa2num.converter import to_number
10
11
12 def text2int (textnum, numwords={}):
13     if not numwords:
14         units = [
15             "cero", "uno", "dos", "tres", "cuatro", "cinco", "seis", "siete", "ocho",
16             "nueve", "diez", "once", "doce", "trece", "catorce", "quince",
17             "dieciseis", "diecisiete", "dieciocho", "diecinueve"]
18         tens = ["", "", "veinte", "treinta", "cuarenta", "cincuenta", "sesenta", "setenta", "ochenta", "noventa"]
19         scales = ["cien", "mil", "millón", "billón", "trillón"]
20         numwords["y"] = (1, 0)
21         for idx, word in enumerate(units): numwords[word] = (1, idx)
22         for idx, word in enumerate(tens): numwords[word] = (1, idx * 10)
23         for idx, word in enumerate(scales): numwords[word] = (10 ** (idx * 3 or 2), 0)
24     textnum = textnum.replace('-', ' ')
25     current = result = 0
26     curstring = ""
27     onnumber = False
28     for word in textnum.split():
29         if word not in numwords:
30             if onnumber:
31                 curstring += repr(result + current) + " "
32                 curstring += word + " "
33                 result = current = 0
34                 onnumber = False
35             else:
36                 scale, increment = numwords[word]
37                 current = current * scale + increment
```

Figura A2-1: Versión del código con reconocedor de Vosk 1/39

```
38         current = current * scale + increment
39         if scale > 100:
40             result += current
41             current = 0
42             onnumber = True
43
44     if onnumber:
45         curstring += repr(result + current)
46
47     return curstring
48
49 def frase(result):
50
51     result=result.replace("con","")
52     result=result.replace("grados","º")
53     result=result.replace("minutos","")
54     result=result.replace("norte","N")
55     result=result.replace("sur","S")
56     result=result.replace("oeste","W")
57     result=result.replace("este","E")
58
59     result=result.replace("longitud","")
60     result=result.replace("latitud","")
61
62     result=result.replace("una","uno")
63     result=result.replace("veinti","veinte y ")
64     result=result.replace("doscientas","doscientos")
65     result=result.replace("doscientos","dos cien")
66     result=result.replace("trescientas","trescientos")
67     result=result.replace("trescientos","tres cien")
68     result=result.replace("cuatrocientas","cuatrocientos")
69     result=result.replace("cuatrocientos","cuatro cien")
70     result=result.replace("quinientas","quinientos")
71     result=result.replace("quinientos","cinco cien")
72     result=result.replace("seiscientas","seiscientos")
73     result=result.replace("seiscientos","seis cien")
74     result=result.replace("setecientas","setecientos")
```

Figura A2-2: Versión del código con reconocedor de Vosk 2/39

```

75     result=result.replace("setecientos","siete cien")
76     result=result.replace("ochocientos","ochocientos")
77     result=result.replace("ochocientos","ocho cien")
78     result=result.replace("novecientas","novecientos")
79     result=result.replace("novecientos","nueve cien")
80     result=result.replace("ciento","uno cien")
81
82     result=text2int(result)
83     return result
84
85 def frasecronica(result):
86     result=result.replace("una","uno")
87     result=result.replace("doscientas","doscientos")
88     result=result.replace("doscientos","dos cien")
89     result=result.replace("trescientas","trescientos")
90     result=result.replace("trescientos","tres cien")
91     result=result.replace("cuatrocientas","cuatrocientos")
92     result=result.replace("cuatrocientos","cuatro cien")
93     result=result.replace("quinientas","quinientos")
94     result=result.replace("quinientos","cinco cien")
95     result=result.replace("seiscientas","seiscientos")
96     result=result.replace("seiscientos","seis cien")
97     result=result.replace("setecientas","setecientos")
98     result=result.replace("setecientos","siete cien")
99     result=result.replace("ochocientas","ochocientos")
100    result=result.replace("ochocientos","ocho cien")
101    result=result.replace("novecientas","novecientos")
102    result=result.replace("novecientos","nueve cien")
103
104    result=text2int(result)
105    return result
106
107 def datoscronica(result):
108     result=result.replace("mar cero","estado de la mar en calma")
109     result=result.replace("mar uno","estado de la mar rizada")
110     result=result.replace("mar dos","estado de la mar marejadilla")
111     result=result.replace("mar tres","estado de la mar marejada")

```

Figura A2-3: Versión del código con reconocedor de Vosk 3/39

```

112     result=result.replace("mar cuatro","estado de la mar fuerte marejada")
113     result=result.replace("mar cinco","estado de la mar gruesa")
114     result=result.replace("mar seis","estado de la mar muy gruesa")
115     result=result.replace("mar siete","estado de la mar arbolada")
116     result=result.replace("mar ocho","estado de la mar montañosa")
117     result=result.replace("mar nueve","estado de la mar enorme")
118     result=result.replace("viento de intensidad cero","viento en calma")
119     result=result.replace("viento de intensidad uno","viento ventolina")
120     result=result.replace("viento de intensidad dos","viento flojito")
121     result=result.replace("viento de intensidad tres","viento flojo")
122     result=result.replace("viento de intensidad cuatro","viento bonancible moderado")
123     result=result.replace("viento de intensidad cinco","viento frequito")
124     result=result.replace("viento de intensidad seis","viento fresco")
125     result=result.replace("viento de intensidad siete","viento frescachón")
126     result=result.replace("viento de intensidad ocho","viento temporal")
127     result=result.replace("viento de intensidad nueve","viento temporal fuerte")
128     result=result.replace("viento de intensidad diez","viento temporal duro")
129     result=result.replace("viento de intensidad once","viento temporal muy duro")
130     result=result.replace("viento de intensidad doce","viento temporal huracanado")
131     return result
132
133 def SpeakText(command):
134
135     engine = pyttsx3.init()
136     engine.say(command)
137     engine.runAndWait()
138
139 def conversor(result):
140     result=result.replace("estado de la mar cero","calma")
141     result=result.replace("estado de la mar uno","rizada")
142     result=result.replace("estado de la mar dos","marejadilla")
143     result=result.replace("estado de la mar tres","marejada")
144     result=result.replace("estado de la mar cuatro","fuerte marejada")
145     result=result.replace("estado de la mar cinco","gruesa")
146     result=result.replace("estado de la mar seis","muy gruesa")
147     result=result.replace("estado de la mar siete","arbolada")
148     result=result.replace("estado de la mar ocho","montañosa")

```

Figura A2-4: Versión del código con reconocedor de Vosk 4/39

```
149     result=result.replace("estado de la mar nueve","enorme")
150     return result
151
152
153     os.chdir('C:/Users/josea/OneDrive/Escritorio/TFG')
154     fecha = datetime.datetime.now()
155     ficheroExcel="Cuaderno de bitácora del {}-{}.xlsx".format(fecha.day,fecha.month)
156     ficheroTxt="Cuaderno de bitácora del {}-{}.txt".format(fecha.day,fecha.month)
157     workbook=xlswriter.Workbook(ficheroExcel)
158     worksheet1=workbook.add_worksheet('crónica')
159     worksheet1.write(0, 0,"Hora" )
160     worksheet1.write(0, 1,"Millas totales")
161     worksheet1.write(0, 2,"Millas navegadas")
162     worksheet1.write(0, 3,"Revoluciones")
163     worksheet1.write(0, 4,"Rumbo de aguja" )
164     worksheet1.write(0, 5,"Rumbo verdadero")
165     worksheet1.write(0, 6,"Latitud")
166     worksheet1.write(0, 7,"Longitud")
167     worksheet1.write(0, 8,"Acaecimientos")
168     worksheet1.write(0, 9,"Acaecimientos")
169
170     worksheet2 = workbook.add_worksheet("meteo")
171     worksheet2.write(0, 0,"Hora" )
172     worksheet2.write(0, 1,"Dirección del viento")
173     worksheet2.write(0, 2,"Velocidad del viento")
174     worksheet2.write(0, 3,"Visibilidad")
175     worksheet2.write(0, 4,"Barómetro" )
176     worksheet2.write(0, 5,"Bola seca")
177     worksheet2.write(0, 6,"Bola húmeda")
178     worksheet2.write(0, 7,"Cantidad de nubes")
179     worksheet2.write(0, 8,"Clase de nubes")
180     worksheet2.write(0, 9,"Estado de la mar")
181     worksheet2.write(0, 10,"Estado de la mar")
182
183
184     worksheet3 = workbook.add_worksheet("máquinas")
185     worksheet3.write(0, 0,"Hora" )
```

Figura A2-5: Versión del código con reconocedor de Vosk 5/39

```
186     ronda=datetime.datetime.now().hour
187     if ronda>=23 or ronda<3:
188         worksheet3.write(1,0,"0400")
189     if ronda>=3 and ronda<7:
190         worksheet3.write(1,0,"0800")
191     if ronda>=7 and ronda<11:
192         worksheet3.write(1,0,"1200")
193     if ronda>=11 and ronda<15:
194         worksheet3.write(1,0,"1600")
195     if ronda>=15 and ronda<19:
196         worksheet3.write(1,0,"2000")
197     if ronda>=19 and ronda<23:
198         worksheet3.write(1,0,"0000")
199     worksheet3.write(0, 1,"Revoluciones Br")
200     worksheet3.write(0, 2,"Revoluciones Er")
201     worksheet3.write(0, 3,"Gasolina consumida")
202     worksheet3.write(0, 4,"Gasolina restante")
203     worksheet3.write(0, 5,"Aceite consumido")
204     worksheet3.write(0, 6,"Aceite restante")
205     worksheet3.write(0, 7,"Agua consumida")
206     worksheet3.write(0, 8,"Agua producida")
207     worksheet3.write(0, 9,"Agua restante")
208     worksheet3.write(0, 10,"Agua restante")
209
210     worksheet4 = workbook.add_worksheet("Ronda mayor")
211     worksheet4.write(0, 0,"Pañol")
212     worksheet4.write(0, 1,"Temperatura máxima")
213     worksheet4.write(0, 2,"Temperatura mínima")
214     worksheet4.write(0, 3,"Humedad relativa")
215     worksheet4.write(0, 4,"Humedad relativa")
216
217
218     cell_format = workbook.add_format({'bold': True, 'center_across':True,'border':2})
219     cell_format1 = workbook.add_format({'border':1})
220     worksheet1.set_row(0,None, cell_format)
221     worksheet2.set_row(0,None, cell_format)
222     worksheet3.set_row(0,None, cell_format)
```

Figura A2-6: Versión del código con reconocedor de Vosk 6/39

```

223 worksheet4.set_row(0,None, cell_format)
224 worksheet1.set_column(8, 8, 100,cell_format1)
225 worksheet1.set_column(1, 7, 20,cell_format1)
226 worksheet2.set_column(1, 9, 20,cell_format1)
227 worksheet3.set_column(1, 9, 20,cell_format1)
228 worksheet4.set_column(0, 3, 18,cell_format1)
229 worksheet1.set_column(0,0,None,cell_format1)
230 worksheet2.set_column(0,0,None,cell_format1)
231 worksheet3.set_column(0,0,None,cell_format1)
232
233 def ArrancarConfirmacion():
234     p2 = pyaudio.PyAudio()
235     stream2 = p2.open(format=pyaudio.paInt16, channels=1, rate=16000, input=True, frames_per_buffer=8000)
236     stream2.start_stream()
237     return stream2
238
239 def PararConfirmacionYContinuar():
240     stream2.stop_stream()
241     stream2.close()
242
243 mt = 1
244 mn = 1
245 rev = 1
246 ra = 1
247 rv = 1
248 lat = 1
249 long = 1
250 aca = 1
251 dv = 1
252 vv = 1
253 v = 1
254 bar = 1
255 bs = 1
256 bh = 1
257 cn = 1
258 cn2 = 1
259 m = 1

```

Figura A2-7: Versión del código con reconocedor de Vosk 7/39

```

260 revbr = 1
261 rever = 1
262 gc = 1
263 gr = 1
264 ac = 1
265 ar = 1
266 agc = 1
267 agp = 1
268 agr = 1
269 comp = 1
270 ma = 1
271 mi = 1
272 h = 1
273 t1 = 1
274 t2 = 1
275
276
277 model = Model("model")
278 rec = KaldiRecognizer(model,16000)
279
280
281 p = pyaudio.PyAudio()
282 stream = p.open(format=pyaudio.paInt16,channels=1,rate=16000,input=True,frames_per_buffer=8000)
283 stream.start_stream()
284
285 salir=False
286 print("Creando Cuaderno de Bitácora")
287 print("Puede empezar a hablar ahora")
288
289 if __name__ == '__main__':
290     while not salir:
291         data = stream.read(4000)
292
293         if len(data)==0:
294             break
295         if rec.AcceptWaveform(data):
296             result = rec.Result()

```

Figura A2-8: Versión del código con reconocedor de Vosk 8/39

```
297 result = json.loads(result)
298 result = result['text']
299 print (result)
300
301 if 'millas totales' in result:
302     SpeakText(f"Usted ha dicho {result}")
303     SpeakText("¿Desea confirmar?")
304     print ("Esperando confirmación")
305     stream.stop_stream()
306     stream2=ArrancarConfirmacion()
307     while True:
308         data2=stream2.read(4000)
309         if len(data2)==0:
310             break
311         if rec.AcceptWaveform(data2):
312             result2=json.loads(rec.Result())
313             result2=result2['text']
314             print (result2)
315             if 'afirmativo' in result2:
316                 stream.start_stream()
317                 try:
318                     statement1 = result.replace ("millas totales ", "")
319                     worksheet1.write(mt,1,f"{to_number(statement1)} NM")
320                     now = datetime.datetime.now()
321                     tiempo= "{}:{}".format(now.hour,now.minute)
322                     worksheet1.write(t1,0,tiempo)
323                     mt=mt+1
324                     t1=t1+1
325                     PararConfirmacionYContinuar()
326                     break
327                 except:
328                     print("Número no reconocido")
329                     break
330             if 'negativo' in result2:
331                 SpeakText("Repita la frase")
332                 stream.start_stream()
333                 PararConfirmacionYContinuar()
```

Figura A2-9: Versión del código con reconocedor de Vosk 9/39

```
334 break
335
336 if 'millas recorridas' in result:
337     SpeakText(f"Usted ha dicho {result}")
338     SpeakText("¿Desea confirmar?")
339     print ("Esperando confirmación")
340     stream.stop_stream()
341     stream2=ArrancarConfirmacion()
342     while True:
343         data2=stream2.read(4000)
344         if len(data2)==0:
345             break
346         if rec.AcceptWaveform(data2):
347             result2=json.loads(rec.Result())
348             result2=result2['text']
349             print (result2)
350             if 'afirmativo' in result2:
351                 stream.start_stream()
352                 try:
353                     statement1 = result.replace("millas recorridas", "")
354                     worksheet1.write(mn,2,f"{to_number(statement1)} NM")
355                     mn=mn+1
356                     PararConfirmacionYContinuar()
357                     break
358                 except:
359                     print("Número no reconocido")
360                     break
361             if 'negativo' in result2:
362                 SpeakText("Repita la frase")
363                 stream.start_stream()
364                 PararConfirmacionYContinuar()
365                 break
366
367 if 'revoluciones del eje' in result:
368     SpeakText(f"Usted ha dicho {result}")
369     SpeakText("¿Desea confirmar?")
370     print ("Esperando confirmación")
```

Figura A2-10: Versión del código con reconocedor de Vosk 10/39

```

371 stream.stop_stream()
372 stream2=ArrancarConfirmacion()
373 while True:
374     data2=stream2.read(4000)
375     if len(data2)==0:
376         break
377     if rec.AcceptWaveform(data2):
378         result2=json.loads(rec.Result())
379         result2=result2['text']
380         print (result2)
381         if 'afirmativo' in result2:
382             stream.start_stream()
383             try:
384                 statement1 = result.replace("revoluciones del eje","")
385                 worksheet1.write(rev,3,f"{to_number(statement1)} rpm")
386                 rev=rev+1
387                 PararConfirmacionYContinuar()
388                 break
389             except:
390                 print("Número no reconocido")
391                 break
392         if 'negativo' in result2:
393             SpeakText("Repita la frase")
394             stream.start_stream()
395             PararConfirmacionYContinuar()
396             break
397
398 if 'rumbo de aguja' in result:
399     SpeakText(f"Usted ha dicho {result}")
400     SpeakText("¿Desea confirmar?")
401     print ("Esperando confirmación")
402     stream.stop_stream()
403     stream2=ArrancarConfirmacion()
404     while True:
405         data2=stream2.read(4000)
406         if len(data2)==0:
407             break

```

Figura A2-11: Versión del código con reconocedor de Vosk 11/39

```

408 if rec.AcceptWaveform(data2):
409     result2=json.loads(rec.Result())
410     result2=result2['text']
411     print (result2)
412     if 'afirmativo' in result2:
413         stream.start_stream()
414         try:
415             statement1 = result.replace("rumbo de aguja","")
416             worksheet1.write(ra,4,to_number(statement1))
417             ra=ra+1
418             PararConfirmacionYContinuar()
419             break
420         except:
421             print("Número no reconocido")
422             break
423     if 'negativo' in result2:
424         SpeakText("Repita la frase")
425         stream.start_stream()
426         PararConfirmacionYContinuar()
427         break
428
429 if 'rumbo verdadero' in result:
430     SpeakText(f"Usted ha dicho {result}")
431     SpeakText("¿Desea confirmar?")
432     print ("Esperando confirmación")
433     stream.stop_stream()
434     stream2=ArrancarConfirmacion()
435     while True:
436         data2=stream2.read(4000)
437         if len(data2)==0:
438             break
439         if rec.AcceptWaveform(data2):
440             result2=json.loads(rec.Result())
441             result2=result2['text']
442             print (result2)
443             if 'afirmativo' in result2:
444                 stream.start_stream()

```

Figura A2-12: Versión del código con reconocedor de Vosk 12/39

```
445         try:
446             statement1 = result.replace("rumbo verdadero","")
447             worksheet1.write(rv,5,to_number(statement1))
448             rv=rv+1
449             PararConfirmacionYContinuar()
450             break
451         except:
452             print("Número no reconocido")
453             break
454     if 'negativo' in result2:
455         SpeakText("Repita la frase")
456         stream.start_stream()
457         PararConfirmacionYContinuar()
458         break
459
460     if 'latitud' in result:
461         SpeakText(f"Usted ha dicho {result}")
462         SpeakText("¿Desea confirmar?")
463         print ("Esperando confirmación")
464         stream.stop_stream()
465         stream2=ArrancarConfirmacion()
466         while True:
467             data2=stream2.read(4000)
468             if len(data2)==0:
469                 break
470             if rec.AcceptWaveform(data2):
471                 result2=json.loads(rec.Result())
472                 result2=result2['text']
473                 print (result2)
474                 if 'afirmativo' in result2:
475                     stream.start_stream()
476                     try:
477                         statement=frase(result)
478                         worksheet1.write(lat,6,statement)
479                         lat=lat+1
480                         PararConfirmacionYContinuar()
481                         break
```

Figura A2-13: Versión del código con reconocedor de Vosk 13/39

```
482         except:
483             print("Número no reconocido")
484             break
485     if 'negativo' in result2:
486         SpeakText("Repita la frase")
487         stream.start_stream()
488         PararConfirmacionYContinuar()
489         break
490
491     if 'longitud' in result:
492         SpeakText(f"Usted ha dicho {result}")
493         SpeakText("¿Desea confirmar?")
494         print ("Esperando confirmación")
495         stream.stop_stream()
496         stream2=ArrancarConfirmacion()
497         while True:
498             data2=stream2.read(4000)
499             if len(data2)==0:
500                 break
501             if rec.AcceptWaveform(data2):
502                 result2=json.loads(rec.Result())
503                 result2=result2['text']
504                 print (result2)
505                 if 'afirmativo' in result2:
506                     stream.start_stream()
507                     try:
508                         statement=frase(result)
509                         worksheet1.write(long,7,statement)
510                         long=long+1
511                         PararConfirmacionYContinuar()
512                         break
513                     except:
514                         print("Número no reconocido")
515                         break
516                 if 'negativo' in result2:
517                     SpeakText("Repita la frase")
518                     stream.start_stream()
```

Figura A2-14: Versión del código con reconocedor de Vosk 14/39

```

519         PararConfirmacionYContinuar()
520         break
521
522     if 'suceso' in result:
523         SpeakText(f"Usted ha dicho {result}")
524         SpeakText("¿Desea confirmar?")
525         print ("Esperando confirmación")
526         stream.stop_stream()
527         stream2=ArrancarConfirmacion()
528         while True:
529             data2=stream2.read(4000)
530             if len(data2)==0:
531                 break
532             if rec.AcceptWaveform(data2):
533                 result2=json.loads(rec.Result())
534                 result2=result2['text']
535                 print (result2)
536                 if 'afirmativo' in result2:
537                     stream.start_stream()
538                     statement1 = result.replace("suceso", "")
539                     worksheet1.write(aca,8,statement1)
540                     aca=aca+1
541                     PararConfirmacionYContinuar()
542                     break
543                 if 'negativo' in result2:
544                     SpeakText("Repita la frase")
545                     stream.start_stream()
546                     PararConfirmacionYContinuar()
547                     break
548
549     if 'dirección del viento' in result:
550         SpeakText(f"Usted ha dicho {result}")
551         SpeakText("¿Desea confirmar?")
552         print ("Esperando confirmación")
553         stream.stop_stream()
554         stream2=ArrancarConfirmacion()
555         while True:

```

Figura A2-15: Versión del código con reconocedor de Vosk 15/39

```

556         data2=stream2.read(4000)
557         if len(data2)==0:
558             break
559         if rec.AcceptWaveform(data2):
560             result2=json.loads(rec.Result())
561             result2=result2['text']
562             print (result2)
563             if 'afirmativo' in result2:
564                 stream.start_stream()
565                 try:
566                     statement1 = result.replace("dirección del viento", "")
567                     worksheet2.write(dv,1,to_number(statement1))
568                     now=datetime.datetime.now()
569                     tiempo="{:}:{:}".format(now.hour,now.minute)
570                     worksheet2.write(t2,0,tiempo)
571                     dv=dv+1
572                     t2=t2+1
573                     PararConfirmacionYContinuar()
574                     break
575                 except:
576                     print("Número no reconocido")
577                     break
578             if 'negativo' in result2:
579                 SpeakText("Repita la frase")
580                 stream.start_stream()
581                 PararConfirmacionYContinuar()
582                 break
583
584     if 'velocidad del viento' in result:
585         SpeakText(f"Usted ha dicho {result}")
586         SpeakText("¿Desea confirmar?")
587         print ("Esperando confirmación")
588         stream.stop_stream()
589         stream2=ArrancarConfirmacion()
590         while True:
591             data2=stream2.read(4000)
592             if len(data2)==0:

```

Figura A2-16: Versión del código con reconocedor de Vosk 16/39

```

593         break
594     if rec.AcceptWaveform(data2):
595         result2=json.loads(rec.Result())
596         result2=result2['text']
597         print (result2)
598         if 'afirmativo' in result2:
599             stream.start_stream()
600             try:
601                 statement1 = result.replace("velocidad del viento","")
602                 worksheet2.write(vv,2,f"{to_number(statement1)} kts")
603                 vv=vv+1
604                 PararConfirmacionYContinuar()
605                 break
606             except:
607                 print("Número no reconocido")
608                 break
609         if 'negativo' in result2:
610             SpeakText("Repita la frase")
611             stream.start_stream()
612             PararConfirmacionYContinuar()
613             break
614
615     if 'visibilidad' in result:
616         SpeakText(f"Usted ha dicho {result}")
617         SpeakText("¿Desea confirmar?")
618         print ("Esperando confirmación")
619         stream.stop_stream()
620         stream2=ArrancarConfirmacion()
621         while True:
622             data2=stream2.read(4000)
623             if len(data2)==0:
624                 break
625             if rec.AcceptWaveform(data2):
626                 result2=json.loads(rec.Result())
627                 result2=result2['text']
628                 print (result2)
629                 if 'afirmativo' in result2:

```

Figura A2-17: Versión del código con reconocedor de Vosk 17/39

```

630         stream.start_stream()
631         try:
632             statement1=result.replace("visibilidad","")
633             worksheet2.write(v,3,to_number(statement1))
634             v=v+1
635             PararConfirmacionYContinuar()
636             break
637         except:
638             print("Número no reconocido")
639             break
640         if 'negativo' in result2:
641             SpeakText("Repita la frase")
642             stream.start_stream()
643             PararConfirmacionYContinuar()
644             break
645
646     if 'presión' in result:
647         SpeakText(f"Usted ha dicho {result}")
648         SpeakText("¿Desea confirmar?")
649         print ("Esperando confirmación")
650         stream.stop_stream()
651         stream2=ArrancarConfirmacion()
652         while True:
653             data2=stream2.read(4000)
654             if len(data2)==0:
655                 break
656             if rec.AcceptWaveform(data2):
657                 result2=json.loads(rec.Result())
658                 result2=result2['text']
659                 print (result2)
660                 if 'afirmativo' in result2:
661                     stream.start_stream()
662                     try:
663                         statement1 = result.replace("presión","")
664                         worksheet2.write(bar,4,f"{to_number(statement1)} hPa")
665                         bar=bar+1
666                         PararConfirmacionYContinuar()

```

Figura A2-18: Versión del código con reconocedor de Vosk 18/39

```

667         break
668     except:
669         print("Número no reconocido")
670         break
671     if 'negativo' in result2:
672         SpeakText("Repita la frase")
673         stream.start_stream()
674         PararConfirmacionYContinuar()
675         break
676
677     if 'bola seca' in result:
678         SpeakText(f"Usted ha dicho {result}")
679         SpeakText("¿Desea confirmar?")
680         print ("Esperando confirmación")
681         stream.stop_stream()
682         stream2=ArrancarConfirmacion()
683         while True:
684             data2=stream2.read(4000)
685             if len(data2)==0:
686                 break
687             if rec.AcceptWaveform(data2):
688                 result2=json.loads(rec.Result())
689                 result2=result2['text']
690                 print (result2)
691                 if 'afirmativo' in result2:
692                     stream.start_stream()
693                     try:
694                         statement1 = result.replace("bola seca","")
695                         worksheet2.write(bs,5,to_number(statement1))
696                         bs=bs+1
697                         PararConfirmacionYContinuar()
698                         break
699                     except:
700                         print("Número no reconocido")
701                         break
702                 if 'negativo' in result2:
703                     SpeakText("Repita la frase")

```

Figura A2-19: Versión del código con reconocedor de Vosk 19/39

```

704         stream.start_stream()
705         PararConfirmacionYContinuar()
706         break
707
708     if 'bola húmeda' in result:
709         SpeakText(f"Usted ha dicho {result}")
710         SpeakText("¿Desea confirmar?")
711         print ("Esperando confirmación")
712         stream.stop_stream()
713         stream2=ArrancarConfirmacion()
714         while True:
715             data2=stream2.read(4000)
716             if len(data2)==0:
717                 break
718             if rec.AcceptWaveform(data2):
719                 result2=json.loads(rec.Result())
720                 result2=result2['text']
721                 print (result2)
722                 if 'afirmativo' in result2:
723                     stream.start_stream()
724                     try:
725                         statement1=result.replace("bola húmeda","")
726                         worksheet2.write(bh,6,to_number(statement1))
727                         bh=bh+1
728                         PararConfirmacionYContinuar()
729                         break
730                     except:
731                         print("Número no reconocido")
732                         break
733                 if 'negativo' in result2:
734                     SpeakText("Repita la frase")
735                     stream.start_stream()
736                     PararConfirmacionYContinuar()
737                     break
738
739     if 'cantidad de nubes' in result:
740         SpeakText(f"Usted ha dicho {result}")

```

Figura A2-20: Versión del código con reconocedor de Vosk 20/39

```

741 SpeakText("¿Desea confirmar?")
742 print ("Esperando confirmación")
743 stream.stop_stream()
744 stream2=ArrancarConfirmacion()
745 while True:
746     data2=stream2.read(4000)
747     if len(data2)==0:
748         break
749     if rec.AcceptWaveform(data2):
750         result2=json.loads(rec.Result())
751         result2=result2['text']
752         print (result2)
753         if 'afirmativo' in result2:
754             stream.start_stream()
755             try:
756                 statement1 = result.replace("cantidad de nubes","")
757                 worksheet2.write(cn,7,f"{to_number(statement1)} octas")
758                 cn=cn+1
759                 PararConfirmacionYContinuar()
760                 break
761             except:
762                 print("Número no reconocido")
763                 break
764         if 'negativo' in result2:
765             SpeakText("Repita la frase")
766             stream.start_stream()
767             PararConfirmacionYContinuar()
768             break
769
770 if 'clase de nubes' in result:
771     SpeakText(f"Usted ha dicho {result}")
772     SpeakText("¿Desea confirmar?")
773     print ("Esperando confirmación")
774     stream.stop_stream()
775     stream2=ArrancarConfirmacion()
776     while True:
777         data2=stream2.read(4000)

```

Figura A2-21: Versión del código con reconocedor de Vosk 21/39

```

778         if len(data2)==0:
779             break
780         if rec.AcceptWaveform(data2):
781             result2=json.loads(rec.Result())
782             result2=result2['text']
783             print (result2)
784             if 'afirmativo' in result2:
785                 stream.start_stream()
786                 statement1=result.replace("clase de nubes","")
787                 worksheet2.write(cn2,8,statement1)
788                 cn2=cn2+1
789                 PararConfirmacionYContinuar()
790                 break
791             if 'negativo' in result2:
792                 SpeakText("Repita la frase")
793                 stream.start_stream()
794                 PararConfirmacionYContinuar()
795                 break
796
797 if 'estado de la mar' in result:
798     SpeakText(f"Usted ha dicho {result}")
799     SpeakText("¿Desea confirmar?")
800     print ("Esperando confirmación")
801     stream.stop_stream()
802     stream2=ArrancarConfirmacion()
803     while True:
804         data2=stream2.read(4000)
805         if len(data2)==0:
806             break
807         if rec.AcceptWaveform(data2):
808             result2=json.loads(rec.Result())
809             result2=result2['text']
810             print (result2)
811             if 'afirmativo' in result2:
812                 stream.start_stream()
813                 try:
814                     statement=conversor(result)

```

Figura A2-22: Versión del código con reconocedor de Vosk 22/39

```

815         worksheet2.write(m,9,statement)
816         m=m+1
817         PararConfirmacionYContinuar()
818         break
819     except:
820         print("Número no reconocido")
821         break
822     if 'negativo' in result2:
823         SpeakText("Repita la frase")
824         stream.start_stream()
825         PararConfirmacionYContinuar()
826         break
827
828     if 'revoluciones babor' in result:
829         SpeakText(f"Usted ha dicho {result}")
830         SpeakText("¿Desea confirmar?")
831         print ("Esperando confirmación")
832         stream.stop_stream()
833         stream2=ArrancarConfirmacion()
834         while True:
835             data2=stream2.read(4000)
836             if len(data2)==0:
837                 break
838             if rec.AcceptWaveform(data2):
839                 result2=json.loads(rec.Result())
840                 result2=result2['text']
841                 print (result2)
842                 if 'afirmativo' in result2:
843                     stream.start_stream()
844                     try:
845                         statement1=result.replace("revoluciones babor","")
846                         worksheet3.write(revbr,1,f"{to_number(statement1)} rpm")
847                         revbr=revbr+1
848                         PararConfirmacionYContinuar()
849                         break
850                     except:
851                         print("Número no reconocido")

```

Figura A2-23: Versión del código con reconocedor de Vosk 23/39

```

852         break
853     if 'negativo' in result2:
854         SpeakText("Repita la frase")
855         stream.start_stream()
856         PararConfirmacionYContinuar()
857         break
858
859     if 'revoluciones estribor' in result:
860         SpeakText(f"Usted ha dicho {result}")
861         SpeakText("¿Desea confirmar?")
862         print ("Esperando confirmación")
863         stream.stop_stream()
864         stream2=ArrancarConfirmacion()
865         while True:
866             data2=stream2.read(4000)
867             if len(data2)==0:
868                 break
869             if rec.AcceptWaveform(data2):
870                 result2=json.loads(rec.Result())
871                 result2=result2['text']
872                 print (result2)
873                 if 'afirmativo' in result2:
874                     stream.start_stream()
875                     try:
876                         statement1=result.replace("revoluciones estribor","")
877                         worksheet3.write(rever,2,f"{to_number(statement1)} rpm")
878                         rever=rever+1
879                         PararConfirmacionYContinuar()
880                         break
881                     except:
882                         print("Número no reconocido")
883                         break
884                 if 'negativo' in result2:
885                     SpeakText("Repita la frase")
886                     stream.start_stream()
887                     PararConfirmacionYContinuar()
888                     break

```

Figura A2-24: Versión del código con reconocedor de Vosk 24/39

```
889
890     if 'gasolina consumida' in result:
891         SpeakText(f"Usted ha dicho {result}")
892         SpeakText("¿Desea confirmar?")
893         print ("Esperando confirmación")
894         stream.stop_stream()
895         stream2=ArrancarConfirmacion()
896         while True:
897             data2=stream2.read(4000)
898             if len(data2)==0:
899                 break
900             if rec.AcceptWaveform(data2):
901                 result2=json.loads(rec.Result())
902                 result2=result2['text']
903                 print (result2)
904                 if 'afirmativo' in result2:
905                     stream.start_stream()
906                     try:
907                         statement1=result.replace("gasolina consumida","")
908                         worksheet3.write(gc,3,f"{to_number(statement1)} L")
909                         gc=gc+1
910                         PararConfirmacionYContinuar()
911                         break
912                     except:
913                         print("Número no reconocido")
914                         break
915                 if 'negativo' in result2:
916                     SpeakText("Repita la frase")
917                     stream.start_stream()
918                     PararConfirmacionYContinuar()
919                     break
920
921     if 'gasolina restante' in result:
922         SpeakText(f"Usted ha dicho {result}")
923         SpeakText("¿Desea confirmar?")
924         print ("Esperando confirmación")
925         stream.stop_stream()
```

Figura A2-25: Versión del código con reconocedor de Vosk 25/39

```
926     stream2=ArrancarConfirmacion()
927     while True:
928         data2=stream2.read(4000)
929         if len(data2)==0:
930             break
931         if rec.AcceptWaveform(data2):
932             result2=json.loads(rec.Result())
933             result2=result2['text']
934             print (result2)
935             if 'afirmativo' in result2:
936                 stream.start_stream()
937                 try:
938                     statement1=result.replace("gasolina restante","")
939                     worksheet3.write(gr,4,f"{to_number(statement1)} L")
940                     gr=gr+1
941                     PararConfirmacionYContinuar()
942                     break
943                 except:
944                     print("Número no reconocido")
945                     break
946             if 'negativo' in result2:
947                 SpeakText("Repita la frase")
948                 stream.start_stream()
949                 PararConfirmacionYContinuar()
950                 break
951
952     if 'aceite consumido' in result:
953         SpeakText(f"Usted ha dicho {result}")
954         SpeakText("¿Desea confirmar?")
955         print ("Esperando confirmación")
956         stream.stop_stream()
957         stream2=ArrancarConfirmacion()
958         while True:
959             data2=stream2.read(4000)
960             if len(data2)==0:
961                 break
962             if rec.AcceptWaveform(data2):
```

Figura A2-26: Versión del código con reconocedor de Vosk 26/39

```

963 result2=json.loads(rec.Result())
964 result2=result2['text']
965 print (result2)
966 if 'afirmativo' in result2:
967     stream.start_stream()
968     try:
969         statement1=result.replace("aceite consumido","")
970         worksheet3.write(ac,5,f"{to_number(statement1)} L")
971         ac=ac+1
972         PararConfirmacionYContinuar()
973         break
974     except:
975         print("Número no reconocido")
976         break
977 if 'negativo' in result2:
978     SpeakText("Repita la frase")
979     stream.start_stream()
980     PararConfirmacionYContinuar()
981     break
982
983 if 'aceite restante' in result:
984     SpeakText(f"Usted ha dicho {result}")
985     SpeakText("¿Desea confirmar?")
986     print ("Esperando confirmación")
987     stream.stop_stream()
988     stream2=ArrancarConfirmacion()
989     while True:
990         data2=stream2.read(4000)
991         if len(data2)==0:
992             break
993         if rec.AcceptWaveform(data2):
994             result2=json.loads(rec.Result())
995             result2=result2['text']
996             print (result2)
997             if 'afirmativo' in result2:
998                 stream.start_stream()
999                 try:

```

Figura A2-27: Versión del código con reconocedor de Vosk 27/39

```

1000 statement1=result.replace("aceite restante","")
1001 worksheet3.write(ar,5,f"{to_number(statement1)} L")
1002 ar=ar+1
1003 PararConfirmacionYContinuar()
1004 break
1005 except:
1006     print("Número no reconocido")
1007     break
1008 if 'negativo' in result2:
1009     SpeakText("Repita la frase")
1010     stream.start_stream()
1011     PararConfirmacionYContinuar()
1012     break
1013
1014 if 'agua consumida' in result:
1015     SpeakText(f"Usted ha dicho {result}")
1016     SpeakText("¿Desea confirmar?")
1017     print ("Esperando confirmación")
1018     stream.stop_stream()
1019     stream2=ArrancarConfirmacion()
1020     while True:
1021         data2=stream2.read(4000)
1022         if len(data2)==0:
1023             break
1024         if rec.AcceptWaveform(data2):
1025             result2=json.loads(rec.Result())
1026             result2=result2['text']
1027             print (result2)
1028             if 'afirmativo' in result2:
1029                 stream.start_stream()
1030                 try:
1031                     statement1=result.replace("agua consumida","")
1032                     worksheet3.write(agc,7,f"{to_number(statement1)} L")
1033                     agc=agc+1
1034                     PararConfirmacionYContinuar()
1035                     break
1036                 except:

```

Figura A2-28: Versión del código con reconocedor de Vosk 28/39

```
1037         print("Número no reconocido")
1038         break
1039     if 'negativo' in result2:
1040         SpeakText("Repita la frase")
1041         stream.start_stream()
1042         PararConfirmacionYContinuar()
1043         break
1044
1045     if 'agua producida' in result:
1046         SpeakText(f"Usted ha dicho {result}")
1047         SpeakText("¿Desea confirmar?")
1048         print("Esperando confirmación")
1049         stream.stop_stream()
1050         stream2=ArrancarConfirmacion()
1051         while True:
1052             data2=stream2.read(4000)
1053             if len(data2)==0:
1054                 break
1055             if rec.AcceptWaveform(data2):
1056                 result2=json.loads(rec.Result())
1057                 result2=result2['text']
1058                 print(result2)
1059                 if 'afirmativo' in result2:
1060                     stream.start_stream()
1061                     try:
1062                         statement1=result.replace("agua producida","")
1063                         worksheet3.write(agr,7,f"{to_number(statement1)} L")
1064                         agr=agr+1
1065                         PararConfirmacionYContinuar()
1066                         break
1067                     except:
1068                         print("Número no reconocido")
1069                         break
1070                 if 'negativo' in result2:
1071                     SpeakText("Repita la frase")
1072                     stream.start_stream()
1073                     PararConfirmacionYContinuar()
```

Figura A2-29: Versión del código con reconocedor de Vosk 29/39

```
1074         break
1075
1076     if 'agua restante' in result:
1077         SpeakText(f"Usted ha dicho {result}")
1078         SpeakText("¿Desea confirmar?")
1079         print("Esperando confirmación")
1080         stream.stop_stream()
1081         stream2=ArrancarConfirmacion()
1082         while True:
1083             data2=stream2.read(4000)
1084             if len(data2)==0:
1085                 break
1086             if rec.AcceptWaveform(data2):
1087                 result2=json.loads(rec.Result())
1088                 result2=result2['text']
1089                 print(result2)
1090                 if 'afirmativo' in result2:
1091                     stream.start_stream()
1092                     try:
1093                         statement1=result.replace("agua restante","")
1094                         worksheet3.write(agr,7,f"{to_number(statement1)} L")
1095                         agr=agr+1
1096                         PararConfirmacionYContinuar()
1097                         break
1098                     except:
1099                         print("Número no reconocido")
1100                         break
1101                 if 'negativo' in result2:
1102                     SpeakText("Repita la frase")
1103                     stream.start_stream()
1104                     PararConfirmacionYContinuar()
1105                     break
1106
1107     if 'compartimento' in result:
1108         SpeakText(f"Usted ha dicho {result}")
1109         SpeakText("¿Desea confirmar?")
1110         print("Esperando confirmación")
```

Figura A2-30: Versión del código con reconocedor de Vosk 30/39

```

1111 stream.stop_stream()
1112 stream2=ArrancarConfirmacion()
1113 while True:
1114     data2=stream2.read(4000)
1115     if len(data2)==0:
1116         break
1117     if rec.AcceptWaveform(data2):
1118         result2=json.loads(rec.Result())
1119         result2=result2['text']
1120         print (result2)
1121         if 'afirmativo' in result2:
1122             stream.start_stream()
1123             statement1=result.replace("compartimento","pañol")
1124             worksheet4.write(comp,0,statement1)
1125             comp=comp+1
1126             PararConfirmacionYContinuar()
1127             break
1128         if 'negativo' in result2:
1129             SpeakText("Repita la frase")
1130             stream.start_stream()
1131             PararConfirmacionYContinuar()
1132             break
1133
1134     if 'temperatura máxima' in result:
1135         SpeakText(f"Usted ha dicho {result}")
1136         SpeakText("¿Desea confirmar?")
1137         print ("Esperando confirmación")
1138         stream.stop_stream()
1139         stream2=ArrancarConfirmacion()
1140         while True:
1141             data2=stream2.read(4000)
1142             if len(data2)==0:
1143                 break
1144             if rec.AcceptWaveform(data2):
1145                 result2=json.loads(rec.Result())
1146                 result2=result2['text']
1147                 print (result2)

```

Figura A2-31: Versión del código con reconocedor de Vosk 31/39

```

1148     if 'afirmativo' in result2:
1149         stream.start_stream()
1150         try:
1151             statement1 = result.replace("temperatura máxima","")
1152             worksheet4.write(ma,1,f"{to_number(statement1)}°C")
1153             ma=ma+1
1154             PararConfirmacionYContinuar()
1155             break
1156         except:
1157             print("Número no reconocido")
1158             break
1159     if 'negativo' in result2:
1160         SpeakText("Repita la frase")
1161         stream.start_stream()
1162         PararConfirmacionYContinuar()
1163         break
1164
1165     if 'temperatura mínima' in result:
1166         SpeakText(f"Usted ha dicho {result}")
1167         SpeakText("¿Desea confirmar?")
1168         print ("Esperando confirmación")
1169         stream.stop_stream()
1170         stream2=ArrancarConfirmacion()
1171         while True:
1172             data2=stream2.read(4000)
1173             if len(data2)==0:
1174                 break
1175             if rec.AcceptWaveform(data2):
1176                 result2=json.loads(rec.Result())
1177                 result2=result2['text']
1178                 print (result2)
1179                 if 'afirmativo' in result2:
1180                     stream.start_stream()
1181                     try:
1182                         statement1 = result.replace("temperatura mínima","")
1183                         worksheet4.write(mi,2,f"{to_number(statement1)}°C")
1184                         mi=mi+1

```

Figura A2-32: Versión del código con reconocedor de Vosk 32/39

```
1185         PararConfirmacionYContinuar()
1186         break
1187     except:
1188         print("Número no reconocido")
1189         break
1190     if 'negativo' in result2:
1191         SpeakText("Repita la frase")
1192         stream.start_stream()
1193         PararConfirmacionYContinuar()
1194         break
1195
1196     if 'humedad' in result:
1197         SpeakText(f"Usted ha dicho {result}")
1198         SpeakText("¿Desea confirmar?")
1199         print("Esperando confirmación")
1200         stream.stop_stream()
1201         stream2=ArrancarConfirmacion()
1202         while True:
1203             data2=stream2.read(4000)
1204             if len(data2)==0:
1205                 break
1206             if rec.AcceptWaveform(data2):
1207                 result2=json.loads(rec.Result())
1208                 result2=result2['text']
1209                 print(result2)
1210                 if 'afirmativo' in result2:
1211                     stream.start_stream()
1212                     try:
1213                         statement1=result.replace("humedad","")
1214                         worksheet4.write(h,3,f"to_number(statement1){%}")
1215                         h=h+1
1216                         PararConfirmacionYContinuar()
1217                         break
1218                     except:
1219                         print("Número no reconocido")
1220                         break
1221                 if 'negativo' in result2:
```

Figura A2-33: Versión del código con reconocedor de Vosk 33/39

```
1222         SpeakText("Repita la frase")
1223         stream.start_stream()
1224         PararConfirmacionYContinuar()
1225         break
1226
1227     if 'crónica' in result:
1228         SpeakText(f"Usted ha dicho {result}")
1229         SpeakText("¿Desea confirmar?")
1230         print("Esperando confirmación")
1231         stream.stop_stream()
1232         stream2=ArrancarConfirmacion()
1233         while True:
1234             data2=stream2.read(4000)
1235             if len(data2)==0:
1236                 break
1237             if rec.AcceptWaveform(data2):
1238                 result2=json.loads(rec.Result())
1239                 result2=result2['text']
1240                 print(result2)
1241                 if 'afirmativo' in result2:
1242                     stream.start_stream()
1243                     with open (ficherotxt, mode='a') as file:
1244                         result=result.replace("crónica","")
1245                         result=datoscronica(result)
1246                         result=frasecronica(result)
1247                         file.write(f"{result}.")
1248                         file.write("\n")
1249                     PararConfirmacionYContinuar()
1250                     break
1251                 if 'negativo' in result2:
1252                     SpeakText("Repita la frase")
1253                     stream.start_stream()
1254                     PararConfirmacionYContinuar()
1255                     break
1256
1257     if 'guardar' in result:
1258         SpeakText("¿Quiere guardar el Cuaderno de Bitácora?")
```

Figura A2-34: Versión del código con reconocedor de Vosk 34/39

```

1259     print ("Esperando confirmación")
1260     stream.stop_stream()
1261     stream2=ArrancarConfirmacion()
1262     while True:
1263         data2=stream2.read(4000)
1264         if len(data2)==0:
1265             break
1266         if rec.AcceptWaveform(data2):
1267             result2=json.loads(rec.Result())
1268             result2=result2['text']
1269             print (result2)
1270
1271             if 'negativo' in result2:
1272                 SpeakText("Repita la frase")
1273                 stream.start_stream()
1274                 PararConfirmacionYContinuar()
1275                 break
1276
1277             if 'afirmativo' in result2:
1278                 stream.start_stream()
1279                 PararConfirmacionYContinuar()
1280                 workbook.close()
1281                 salir=True
1282                 break
1283
1284     if 'consultar' in result:
1285         consulta=False
1286         SpeakText ("Indique sobre qué columna desea saber en qué fila se encuentra")
1287         SpeakText ("Si por el contrario le gustaría saber el estado de una hoja diga el nombre de la hoja")
1288         SpeakText("Si quiere cancelar diga cancelar")
1289         print("esperando")
1290
1291     while not consulta:
1292         stream.stop_stream()
1293         stream2=ArrancarConfirmacion()
1294         while True:
1295             data2=stream2.read(8000)

```

Figura A2-35: Versión del código con reconocedor de Vosk 35/39

```

1296         if len(data2)==0:
1297             break
1298         if rec.AcceptWaveform(data2):
1299             result2=json.loads(rec.Result())
1300             result2=result2['text']
1301             print (result2)
1302             if 'cancelar' in result2:
1303                 consulta=True
1304                 SpeakText("Opción cancelada")
1305                 stream.start_stream()
1306                 PararConfirmacionYContinuar()
1307                 break
1308             if 'millas totales' in result2:
1309                 SpeakText(f"La columna de millas totales se encuentra en la fila {mt}")
1310             if 'millas recorridas' in result2:
1311                 SpeakText(f"La columna de millas navegadas se encuentra en la fila {mn}")
1312             if 'revoluciones del eje' in result2:
1313                 SpeakText(f"La columna de revoluciones del eje se encuentra en la fila {rev}")
1314             if 'rumbo de aguja' in result2:
1315                 SpeakText(f"La columna de rumbo de aguja se encuentra en la fila {ra}")
1316             if 'rumbo verdadero' in result2:
1317                 SpeakText(f"La columna de rumbo verdadero se encuentra en la fila {rv}")
1318             if 'latitud' in result2:
1319                 SpeakText(f"La columna de latitud se encuentra en la fila {lat}")
1320             if 'longitud' in result2:
1321                 SpeakText(f"La columna de longitud se encuentra en la fila {long}")
1322             if 'acaecimiento' in result2:
1323                 SpeakText(f"La columna de acaecimiento se encuentra en la fila {aca}")
1324             if 'dirección del viento' in result2:
1325                 SpeakText(f"La columna de dirección del viento se encuentra en la fila {dv}")
1326             if 'velocidad del viento' in result2:
1327                 SpeakText(f"La columna de velocidad del viento se encuentra en la fila {vv}")
1328             if 'visibilidad' in result2:
1329                 SpeakText(f"La columna de visibilidad se encuentra en la fila {v}")
1330             if 'barómetro' in result2:
1331                 SpeakText(f"La columna de barómetro se encuentra en la fila {bar}")
1332             if 'bola seca' in result2:

```

Figura A2-36: Versión del código con reconocedor de Vosk 36/39

```

1333     SpeakText(f"La columna de bola seca se encuentra en la fila {bs}")
1334     if 'bola húmeda' in result2:
1335         SpeakText(f"La columna de bola húmeda se encuentra en la fila {bh}")
1336     if 'cantidad de nubes' in result2:
1337         SpeakText(f"La columna de cantidad de nubes se encuentra en la fila {cn}")
1338     if 'clase de nubes' in result2:
1339         SpeakText(f"La columna de clase de nubes se encuentra en la fila {cn2}")
1340     if 'estado de la mar' in result2:
1341         SpeakText(f"La columna de estado de la mar se encuentra en la fila {m}")
1342     if 'motor babor' in result2:
1343         SpeakText(f"La columna de motor babor se encuentra en la fila {revbr}")
1344     if 'motor estribor' in result2:
1345         SpeakText(f"La columna de motor estribor se encuentra en la fila {rever}")
1346     if 'gasolina consumida' in result2:
1347         SpeakText(f"La columna de gasolina consumida se encuentra en la fila {gc}")
1348     if 'gasolina restante' in result2:
1349         SpeakText(f"La columna de gasolina restante se encuentra en la fila {gr}")
1350     if 'aceite consumido' in result2:
1351         SpeakText(f"La columna de aceite consumido se encuentra en la fila {ac}")
1352     if 'aceite restante' in result2:
1353         SpeakText(f"La columna de aceite restante se encuentra en la fila {ar}")
1354     if 'agua consumida' in result2:
1355         SpeakText(f"La columna de agua consumida se encuentra en la fila {agc}")
1356     if 'agua producida' in result2:
1357         SpeakText(f"La columna de agua producida se encuentra en la fila {agp}")
1358     if 'agua restante' in result2:
1359         SpeakText(f"La columna de agua restante se encuentra en la fila {agr}")
1360     if 'compartimento' in result2:
1361         SpeakText(f"La columna de compartimento se encuentra en la fila {comp}")
1362     if 'temperatura máxima' in result2:
1363         SpeakText(f"La columna de temperatura máxima se encuentra en la fila {ma}")
1364     if 'temperatura mínima' in result2:
1365         SpeakText(f"La columna de temperatura mínima se encuentra en la fila {mi}")
1366     if 'humedad' in result2:
1367         SpeakText(f"La columna de humedad se encuentra en la fila {h}")
1368     if 'hoja de crónica' in result2:
1369         if mt==mn==rev==ra==rv==lat==long==aca:

```

Figura A2-37: Versión del código con reconocedor de Vosk 39/40

```

1370     SpeakText("Toda la hoja se encuentra en la misma fila")
1371     else:
1372         SpeakText("Una columna no se encuentra en la misma fila")
1373     if 'hoja de meteorología' in result2:
1374         if dv==vv==v==bar==bh==bs==cn==cn2==m:
1375             SpeakText("Toda la hoja se encuentra en la misma fila")
1376         else:
1377             SpeakText("Una columna no se encuentra en la misma fila")
1378     if 'hoja de máquinas' in result2:
1379         if revbr==rever==gc==gr==ac==ac==agc==agp==agr:
1380             SpeakText("Toda la hoja se encuentra en la misma fila")
1381         else:
1382             SpeakText("Una columna no se encuentra en la misma fila")
1383     if 'hoja de ronda mayor' in result2:
1384         if comp==ma==mi==h:
1385             SpeakText("Toda la hoja se encuentra en la misma fila")
1386         else:
1387             SpeakText("Una columna no se encuentra en la misma fila")
1388
1389     if 'la actitud' in result:
1390         SpeakText(f"Usted ha dicho {result}")
1391         SpeakText("¿Desea confirmar?")
1392         print ("Esperando confirmación")
1393         stream.stop_stream()
1394         stream2=ArrancarConfirmacion()
1395         while True:
1396             data2=stream2.read(4000)
1397             if len(data2)==0:
1398                 break
1399             if rec.AcceptWaveform(data2):
1400                 result2=json.loads(rec.Result())
1401                 result2=result2['text']
1402                 print (result2)
1403                 if 'afirmativo' in result2:
1404                     stream.start_stream()
1405                     try:
1406                         Latitud()

```

Figura A2-38: Versión del código con reconocedor de Vosk 38/39

```
1407         lat=lat+1
1408         PararConfirmacionYContinuar()
1409         break
1410     except:
1411         print("Número no reconocido")
1412         break
1413     if 'negativo' in result2:
1414         SpeakText("Repita la frase")
1415         stream.start_stream()
1416         PararConfirmacionYContinuar()
1417         break
```

Figura A2-39: Versión del código con reconocedor de Vosk 39/39