UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

título del TFG
Documentación Técnica

Presentado por nombre alumno
en Universidad de Burgos — 27 de junio de 2019
Tutor: nombre tutor

Índice general

Índice de figuras

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En este apartado se desarrollará la planificación temporal del proyecto, así como también un estudio de viabilidad.

A.2. Planificación temporal

Para la planificación temporal del proyecto, se sigue una metodología *Scrum*. Se realizarán diferentes *sprints*, en los que se marcaban objetivos. Idealmente, los *sprints* duran dos semanas. Debido a carga de trabajo externa a este proyecto, esta duración se ha visto afectada y se han realizado *sprints* de más y menos duración. . Durante cada uno de ellos, se irán creando y realizando las diferentes tareas correspondientes a los objetivos fijados en cada *sprints*. Al finalizar cada *sprint*, se realizan reuniones con los tutores para revisar los objetivos cumplidos y marcar unos nuevos.

Se ha utilizado Github para el seguimiento de la aplicación, ayudados por el tablero Kanban de Zen-hub. Esta herramienta nos ha permitido llevar un seguimiento más detallado de la planificación en cada momento. El repositorio del proyecto, y las issues realizadas se encuentra en el repositorio del proyecto[1].

A su vez, cabe destacar que los primeros *sprints* no se muestran de manera correcta. Esto es debido al fallo de no cerrar los *sprints* en github, por lo que los gráficos *Burndown* están alterados.

[1] https://github.com/AdrianArnaiz/TFG-Neurodegenerative-Disease-Detection

Sprint 1

Idealmente, el comienzo de este sprint comenzaba el 15 de Diciembre. Debido a cargas de trabajo externas al proyecto, al final se comenzó el 14 de febrero. Duración del sprint: 14 de febrero 2019 al 28 de febrero 2019. Ver gráfico *Burndown* en la figura A.9.

Figura A.1: Burndown chart del sprint 1

Tareas realizadas:

- Instalación del cliente Github: GitTortoise.
- Crear la estructura de directorios del repositorio.
- Instalación de LaTeX y todos sus componentes.
- Lectura en profundidad de dos artículos de Giovanni Dimauro: [2] y [1] .

En este sprint se realizó la fase inicial del proyecto, que comprendía desde la instalación y creación de elementos principales, hasta la iniciación de la investigación.

Sprint 2

Duración del sprint: 28 de febrero de 2019 al 13 de marzo de 2019. Ver gráfico *Burndown* en la figura A.2.

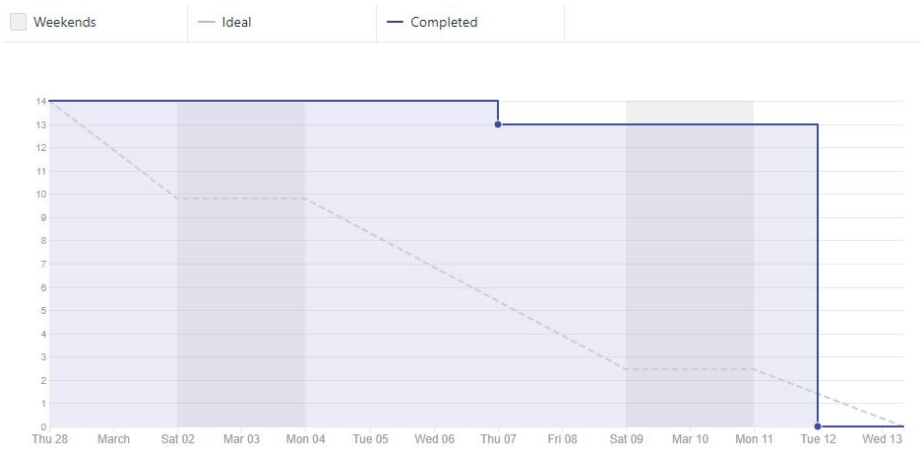

Figura A.2: Burndown chart del sprint 2

Tareas realizadas:

- Realizar la taxonomía de las características extraídas de cada audio, qué características de cada tipo...
- Identificar las herramientas de extracción de características de audio.
- Realizar taxonomía de correspondencia entre las herramientas y las características concretas a extraer.
- Realizar un arreglo en la preservación de privacidad de los audios del conjunto de datos.

En este sprint, seguimos explorando el estado del arte ahora de manera más precisa: empezar a ver en profundidad el proceso de extracción de características y medidas de los audios, explorando en diferentes artículos que características extraer y buscando las herramientas necesarias para su extracción.

Sprint 3

Duración del sprint: 13 de marzo 2019 al 29 de marzo 2019. Ver gráfico *Burndown* en la figura A.3.

Figura A.3: Burndown chart del sprint 3

Tareas realizadas:

- Búsqueda de audios, peticiones y realización de taxonomía de audios encontrados.
- Lectura en profundidad de los artículos más importantes del estado del arte, i.e. [4].
- Finalización de exploración del estado del arte: realizar taxonomía y resúmenes.

En este sprint se llevó a cabo la finalización de estado del arte y su resumen de aspectos y artículos más importantes. Un aspecto importante fue la búsqueda de audios, trabajo bastante arduo dentro del proyecto debido a que son muy difíciles de encontrar. Empezamos el proceso de extracción de características descrito en [4]: instalación y familiarización de librerías, segmentación de audios, etc...

Sprint 4

Duración del sprint: 29 de marzo 2019 al 11 de abril 2019. Ver gráfico *Burndown* en la figura A.4.

Figura A.4: Burndown chart del sprint 4

Tareas realizadas:

- Instalar herramientas de manejo de audio como Disvoice.
- Explorar el funcionamiento de estas herramientas.
- Preprocesar los audios: eliminar silencios inicial y final.
- Extracción de diferentes características de los audios con Disvoice.
  - Extracción medidas prosódicas de audios completos.
  - Extraer medidas de fonación de *voiced frames* de vocales y frase.
  - Extraer medidas de articulación (cepstrales y de energía de transiciones).

Se realizó el proceso de extracción de características. Se recorre toda la estructura de audios para crear las correspondientes matrices de características (en numpy) de cada tipo de audio y así dejar preparados los datos para entrenar modelos. Hay una matriz de características por cada tipo de audio y cada tipo de características. Para ello se realizó desde la instalación y comprensión del funcionamiento de las herramientas de manejo de audio, hasta extraer las medidas completas con la herramienta Disvoice.

Sprint 5

Duración del sprint: 15 de abril 2019 al 2 de mayo 2019. Ver gráfico *Burndown* en la figura A.5.

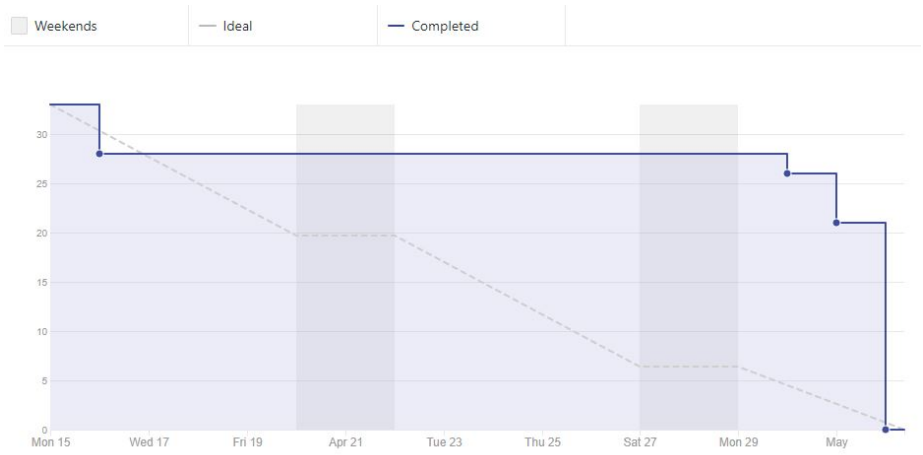

Figura A.5: Burndown chart del sprint 5

Tareas realizadas:

- Documentación de avances hasta el momento.
- Realización de los **primeros experimentos** con clasificadores, incluyendo la limpieza de características necesaria para ello.
- Realización proyección de características.
- Investigación, documentación y aprendizaje sobre *Deep Learning*.

En este sprint, se documentó el trabajo realizado hasta el momento. También se realizó el estudio con clasificadores para los conjuntos de características extraídos con Disvoice. Probamos proyecciones de datos de estas características para graficarlos. En este sprint se comenzó la investigación y aprendizaje sobre Deep Learning, para su aplicación a posteriores sprints.

Sprint 6

Duración del sprint: 2 de mayo 2019 al 17 de mayo 2019. Ver gráfico *Burndown* en la figura A.6.

Figura A.6: Burndown chart del sprint 6

Tareas realizadas:

- Búsqueda de librerías de *Deep Learning* para audios.
- Modificación de las características Disvoice.
- División del conjunto de datos por sexo.
- Realización de **segundos experimentos** con características Disvoice modificadas.

En este sprint realizamos la modificación de las características de Disvoice (adición de edad y sexo, división por sexos...) y realizamos los experimentos con estas nuevas características. También, siguiendo la línea del anterior sprint, profundizamos más en el *Deep Learning*, Buscando bibliotecas para la extracción de características de nuestros audios.

Sprint 7

Duración del sprint: 17 de mayo 2019 a 14 de junio 2019. Ver gráfico *Burndown* en la figura A.7.

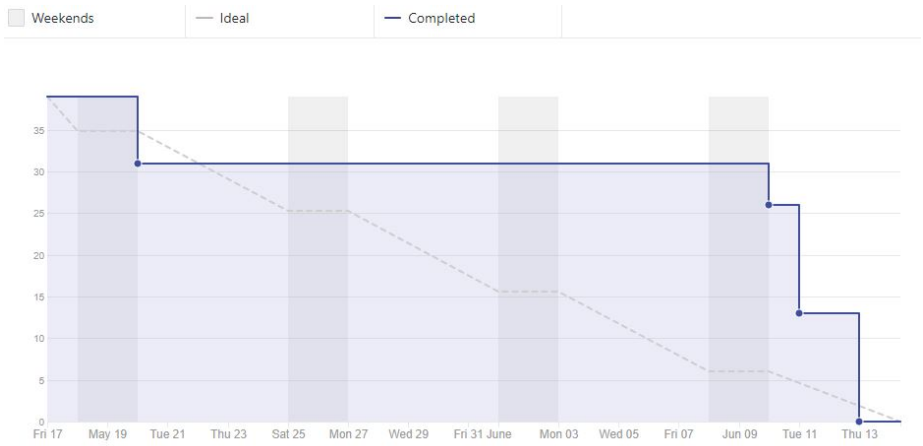

Figura A.7: Burndown chart del sprint 7

Tareas realizadas:

- Arreglo de fallo encontrado.
- Repetición de primeros y segundos experimentos: incluye repetición de análisis y resumen de resultados.
- Instalación biblioteca VGGish (y VGGish2Keras).
- Realizar extracción de características con VGGish (VGGish2Keras).
- Realización **terceros experimentos** con características VGGish.
- Análisis de viabilidad comercial: *lean canvas*.

En la reunión previa a este sprint, se revisó el código y encontramos un fallo de **data snooping**. No hacíamos validación cruzada anidada (*nested CV*). Debido a esto, tuvimos que arreglar el error y repetir los experimentos realizados hasta el momento (primeros y segundos con características Disvoice). Tras finalizar, tuvimos que volver a analizar y resumir todos los resultados de los experimentos, aunque estos habían variado menos de lo esperado. También, siguiendo con la línea de anteriores sprints, realizamos la instalación de la biblioteca VGGish de *Deep Learning*, extrajimos las características de los audios con esta biblioteca y realizamos los experimentos con estas nuevas características. Para finalizar, y por pedido de la OTRI (Ofinia de transmisión de información de la Universidad de Burgos), se

realizó un análisis de viabilidad comercial, concretamente un *lean canvas* de nuestro proyecto.

Cabe destacar que la duración de este sprint es bastante mayor de lo normal. Se debe a dos aspectos. El primero de ellos es que la carga de trabajo de este sprint era sustancialmente superior a los demás. Se debe a que en fallo encontrado requería de una repetición de gran parte del trabajo y además no podíamos frenarnos en los experimentos con *Deep Learning*. Por otra parte, coincidión con las dos últimas semanas del curso, las cuales estaban repletas de trabajos finales y exámenes de convocatoria, que hacían que el tiempo dedicado al proyecto tuviera que ser menor.

Sprint 8

Duración del sprint: 14 de junio 2019 a 24 de junio 2019. Ver gráfico *Burndown* en la figura A.8.

Figura A.8: Burndown chart del sprint 8

Tareas realizadas:

- Aprendizaje del patrón de diseño mediador-fachada.
- Realización de la aplicación de escritorio.

En este sprint se realizó la aplicación demostrativa de escritorio. Para ello, se sugirió hacer mediante el patrón de diseño mediador-fachada. Por ello antes de realizar la aplicación se tuvo que comprender ese patrón. Una

vez comprendido se realizó la aplicación. Como ya hemos comentado en la memoria principal, esta aplicación, aunque es la parte más vistosa, personalmente no la considero la parte más importante del proyecto. Únicamente es una aplicación que muestra una posible aplicación del resultado de este investigación, y que con las líneas de investigación futuras pudiera llegar a un producto funcional y estable.

Sprint 9

Duración del sprint: 24 de junio 2019 a 1 de junio 2019. Ver gráfico *Burndown* en la figura A.9.

Figura A.9: Burndown chart del sprint 1

Tareas realizadas:

- Revisión de la memoria hasta el momento.
- Finalización de la memoria.
- Búsqueda de documentación de pruebas unitarias en Minería de Datos.
- Finalización de los anexos.

En este último sprint se hicieron las últimas tareas del proyecto. Se terminó toda la documentación, lo que comprendía desde revisar lo hecho anteriormente, realizar lo que faltaba por terminar y buscar información sobre pruebas unitarias en el campo de la minería de datos.

A.3. Estudio de viabilidad

En este apartado se abordará la viabilidad económica, comercial y legal de este proyecto.

Viabilidad económica

En este apartado, simularemos el coste del proyecto que tendría en una empresa, o en una venta al público.

Coste personal

Este proyecto cuenta con dos **profesores contratados** durante 6 meses para este proyecto. Por cada profesor se asignan 0,5 créditos, por lo que el coste[2] por tutor consideramos que es el siguiente:

- Sueldo base mensual ayudante doctor: 1815,61 euros. 21787,32 euros anuales. En total imparte 24 créditos por lo que el coste anual por crédito es de 907,5 euros. El coste total es el siguiente:

$$907,5 \cdot 0,5 créditos = 453,75 euros \tag{A.1}$$

- Sueldo base mensual doctor permanente: 2325,24 euros. 27902,88 euros anuales. En total imparte 24 créditos por lo que el coste anual por crédito es de 1162.62 euros. El coste total es el siguiente:

$$1162,62 \cdot 0,5 créditos = 581,31 euros \tag{A.2}$$

También cuanta con un desarrollador del proyecto. Supondremos un salario bruto de 2000 euros para el desarrollador. Habrá que tener en cuenta los gastos de seguridad social[3].

- Cotización por parte de la empresa: 23.6 %
- Cotización por parte del empleado: 4.7 %
- **Total: 28.3 %**

[2] http://www.ubu.es/sites/default/files/portal_page/files/pdi_laboral_2017_2.pdf

[3] http://www.seg-social.es/wps/portal/wss/internet/Trabajadores/CotizacionRecaudacionTrabajadores/

Por lo tanto el coste del desarrollador se detalla en A.1

Concepto	**Coste**
Salario mensual neto	1217.25
Retención IRPF (15 %)	216.75
Seguridad social (28.3 %)	566
Salario mensual bruto	2000
Coste total (6 meses)	12000

Tabla A.1: Costes de personal

Coste informático

El coste informático es mínimo. Se ha utilizado un único portátil personal. Su coste fue de 800 euros y se supone una amortización de 5 años. Para ello se contabilizará únicamente el cose amortizado.

$$(800/(12 \cdot 5)) \cdot 6 = 80 euros \tag{A.3}$$

Ingresos

Para este proyecto se ha contado con la concesión de la beca prototipos de la Oficina de Transmisión de Información de la Universidad de Burgos. La cuantía de la beca ha sido de 800 euros, por lo que estos serán los ingresos del proyecto.

Coste Total

El coste total se detalla en la tabla A.2

Concepto	**Coste**
Coste desarrollador	12000
Costes Tutores(15 %)	1035.06
Hardware	80
Beca prototipos	-800
Coste total (6 meses)	12315.6

Tabla A.2: Coste Total

Viabilidad comercial

Para analizar la viabilidad comercial, realizaremos un *lean canvas*, A.10, la cual es una herramienta de análisis de negocio desarrollada por Alex Osterwalder [5].

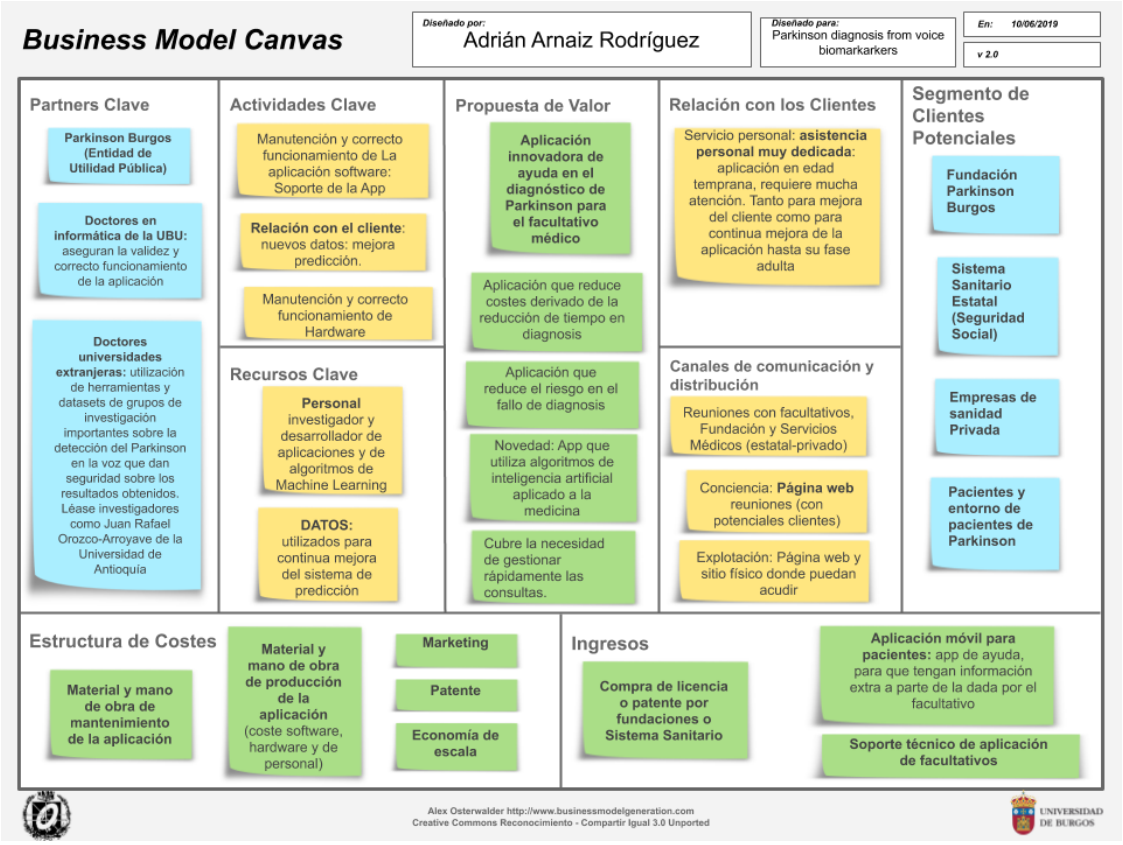

Figura A.10: Lean Canvas del proyecto

Viabilidad legal

Todo el código utilizado es de dominio público. Hemos utilizado multitud de librerías del lenguaje Python, las cuales son todas de dominio público. Las herramientas más concretas que hemos utilizado han sido las siguientes.

- **Disvoice**: alojado en repositorio público[4] de Github. **Tiene la licencia pública MIT**.

[4] https://github.com/jcvasquezc/DisVoice

- **VGGish**: alojado en repositorio público de Github[5].Pertenece al proyecto de Tensorflow. **Tiene la licencia pública Apache license 2.0**.

Todas las demás librerías utilizadas son propias de Python. También mencionar que no utilizamos el repositorio Disvoice original, sino que le hemos modificado para la ejecución en Windows. Esto ha sido posible ya que la licencia MIT permite tanto el uso comercial como la modificación el mismo.

[5] https://github.com/tensorflow/models/tree/master/research/audioset

Apéndice B

Especificación de Requisitos

B.1. Introducción

En esta sección se abordarán los diferentes objetivos y requisitos del proyecto. Se presentarán tanto los requisitos globales del proyecto, como los requisitos funcionales y casos de uso de la aplicación.

B.2. Objetivos generales

- Investigar y condensar el estado del arte de la investigación sobre la detección del Parkison a través de la voz resumiendo los artículos científicos más importantes, identificando las tecnologías, herramientas y procesos actuales utilizadas en ellos, realizando taxonomías, etc.

- Recopilación de bases de datos adecuadas para la investigación, tanto para uso en este proyecto como para su uso en proyectos posteriores, cerciorando que son *datasets* de audios correctos para las tareas necesitadas (i.e. audios etiquetados).

- Realización de un estudio comparativo en cuanto a la utilización de diferentes modelos de clasificación y conjuntos de características extraídas de los audios. Se compararán resultados, tanto entre los diferentes experimentos que nosotros realicemos, como entre nuestros mejores experimentos y resultados científicos de anteriores experimentos (publicados en artículos científicos).

- Aportación de una nueva perspectiva a este campo de investigación: extracción de las características de los audios mediante *Deep Learning*.
- Finalmente, utilizar todo lo descrito anteriormente para realizar una aplicación la cual permita la monitorización de la diagnosis de la enfermedad del Parkinson a través de la voz. La aplicación será capaz de discernir, mediante un clasificador, si la persona de un audio subido a la aplicación web tiene Parkinson o no. Esta aplicación será un ejemplo de como poder plasmar los resultados de esta investigación en una herramienta funcional.

B.3. Catalogo de requisitos

En esta sección se enumerarán los requisitos funcionales de nuestra aplicación y los actores que la realizan.

Actores:

Usuario El usuario será quien utiliza la aplicación. Idealmente, será el facultativo médico en su consulta.

Científico de datos Este actor será quien realiza la investigación, quien realiza los experimentos con los clasificadores.

- **RF.1** Crear una herramienta que nos permita detectar con que probabilidad tiene Parkinson la persona de un audio dado.
  - **RF.1.1** El usuario podrá elegir el audio que quiere predecir.
  - **RF.1.2** El usuario podrá añadir la edad y el sexo del paciente.
  - **RF.1.3** El usuario podrá ver la probabilidad en porcentaje de PArkinson de la persona del audio.
- **RF.2** Crear una herramienta que permita ver las gráficas de amplitud de onda y el espectrograma de frecuencias de un audio.
  - **RF.2.1** El usuario podrá elegir el audio del que quiere ver las gráficas.
  - **RF.2.2** El usuario podrá ver las gráficas en la pantalla.
  - **RF.2.3** El usuario podrá navegar en las gráficas (zoom, moverse).
  - **RF.2.4** El usuario podrá guardar las gráficas.

- **RF.3** Crear una herramienta que permita reproducir un audio.
  - **RF.3.1** El usuario podrá elegir el audio que quiere reproducir.
  - **RF.3.2** El usuario podrá reproducir un audio dando al play.
  - **RF.3.3** El usuario podrá pausar el audio dando al pause.
  - **RF.3.4** El usuario podrá rebobinar el audio dando a rebobinar.
  - **RF.3.5** El usuario podrá cambiar el volumen deslizando una pestaña.
  - **RF.3.6** El usuario podrá enmudecer el audio pulsando al mute.
- **RF.4** Crear una herramienta que permita cargar y borrar audios al entorno para trabajar con ellos.
  - **RF.4.1** El usuario podrá elegir los audios de cualquier ruta que quiere cargar en la aplicación.
  - **RF.4.2** El usuario podrá elegir los audios cargados en la aplicación que quiere borrar.
- **RF.5** Crear modelos de clasificación para los audios con Python.
  - **RF.5.1** El científico de datos podrá cargar datos de los audios.
  - **RF.5.2** El científico de datos podrá elegir el tipo de modelo que queremos entrenar.
  - **RF.5.3** El científico de datos podrá elegir la validación cruzada que queremos realizar.
  - **RF.5.4** El científico de datos podrá particionar los datos de manera estratificada.
  - **RF.5.5** El científico de datos podrá entrenar el modelo con la anterior configuración.
- **RF.6** Evaluar clasificadores.
  - **RF.6.1** El científico de datos podrá obtener medidas de rendimiento de los clasificadores.
  - **RF.6.2** El científico de datos podrá resumir medidas de rendimiento de todos los clasificadores.
  - **RF.6.3** El científico de datos podrá mostrar medidas de rendimiento en tablas o gráficas para ayudar a la decisión.

B.4. Especificación de requisitos

Diagrama de casos de uso

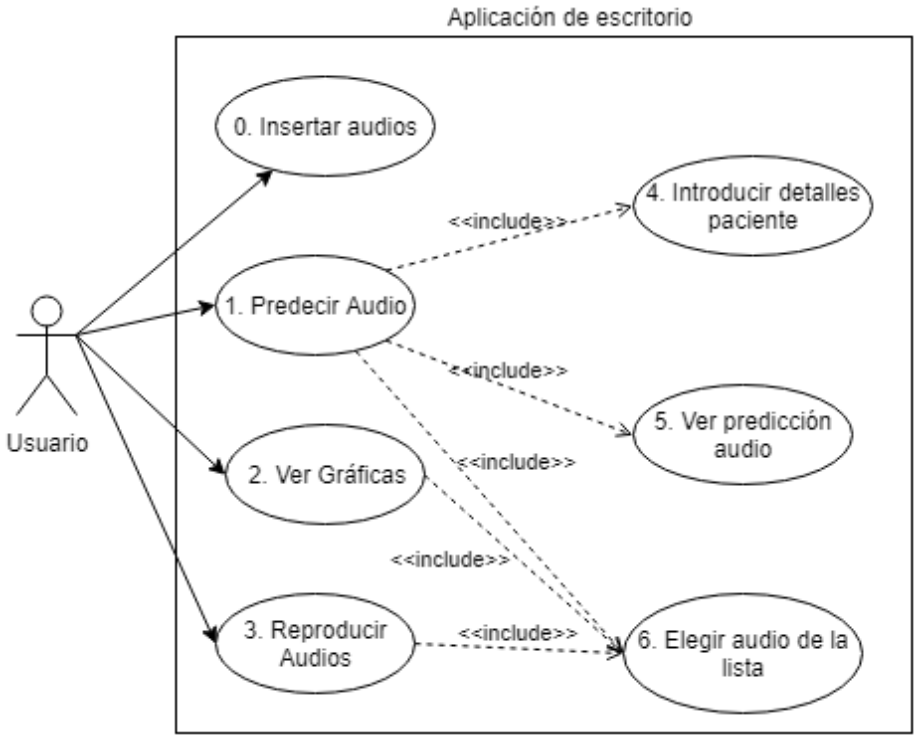

Figura B.1: Diagrama de casos de uso de la aplicación.

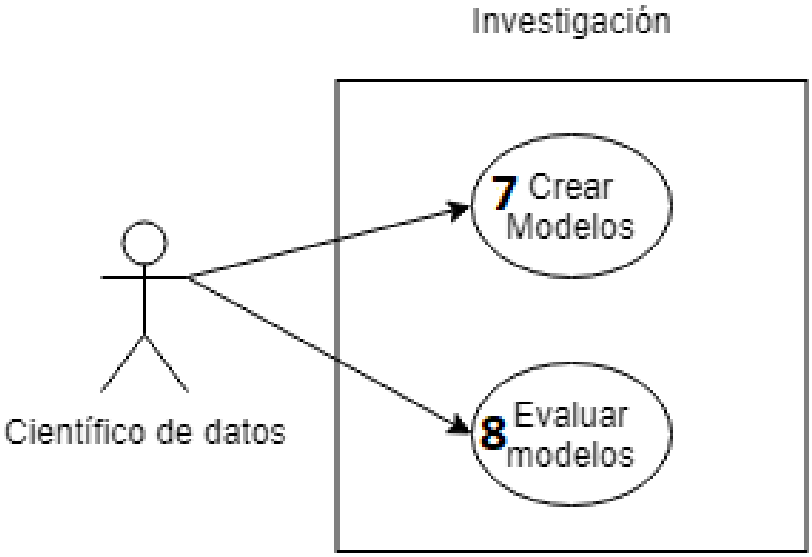

Figura B.2: Diagrama de casos de uso de la investigación.

Especificación de los casos de uso

Caso de uso 0: Insertar audios.		
Descripción	Permite al usuario cargar audios de su equipo a la aplicación.	
Requisitos	RF-4	
	RF-4.1	
	RF-4.2	
Precondiciones	Ninguna	
Secuencia normal	Paso	Acción
	1	El usuario pincha el botón 'Añadir'.
	2	Se despliega un menú de búsqueda de archivos.
	3	Se eligen los audios y se da a 'Abrir'
	4	Se cargan las imágenes dentro de la aplicación.
Postcondiciones	Los audios aparecen en el recuadro de audios cargados en la aplicación, listos para reproducir, predecir o mostrar gráficas.	
Excepciones	Audio no encontrado	
Importancia	Alta	
Urgencia	Alta	

Tabla B.1: Caso de uso 0: Insertar audios.

Caso de uso 1: Predecir Audio.		
Descripción	Permite al usuario obtener la predicción de parkinson de un audio cargado en la aplicación.	
Requisitos	RF-1	
	RF-1.1	
	RF-1.2	
	RF-1.3	
Precondiciones	Tener un audio cargado	
Secuencia normal	Paso	Acción
	1	El usuario elige un audio de la lista.
	2	El usuario introduce edad y sexo del paciente.
	3	El usuario pincha el botón predecir.
	4	Aparece la predicción en pantalla.
Postcondiciones	Aparece la predicción del audio en porcentaje en la pantalla.	
Excepciones	Audio no seleccionado. Edad no introducida. Edad no está en formato válido.	
Importancia	Alta	
Urgencia	Alta	

Tabla B.2: Caso de uso 1: Predecir Audio.

Caso de uso 2: Ver gráficas.		
Descripción	Permite al usuario obtener las gráficas de un audio.	
Requisitos	RF-2	
	RF-2.1	
	RF-2.2	
	RF-2.3	
	RF-2.4	
Precondiciones	Tener un audio cargado y con los detalles del paciente introducidos.	
Secuencia normal	Paso	Acción
	1	El usuario elige un audio de la lista.
	2	El usuario pincha el botón predecir.
	3	Aparecen las gráficas en pantalla.
	4	El usuario puede navegar o guardar las gráficas.
Postcondiciones	Aparece las gráficas de amplitud y espectrograma de frecuencias en la pantalla.	
Excepciones	Audio no seleccionado.	
Importancia	Alta	
Urgencia	Alta	

Tabla B.3: Caso de uso 2: Ver Gráficas.

Caso de uso 3: Reproducir audios.		
Descripción	Permite al usuario reproducir un audio.	
Requisitos	RF-3	
	RF-3.1	
	RF-3.2	
	RF-3.3	
	RF-3.4	
	RF-3.5	
	RF-3.6	
Precondiciones	Tener un audio cargado	
Secuencia normal	Paso	Acción
	1	El usuario elige un audio de la lista.
	2	El usuario pincha el botón play.
	3	El sonido comienza a sonar.
	4	El usuario puede pinchar el botón pause, volumen o rebobinar.
	5	El audio se para, cambia de volumen o vuelve a empezar.
Postcondiciones	Ninguna	
Excepciones	Audio no seleccionado.	
Importancia	Alta	
Urgencia	Alta	

Tabla B.4: Caso de uso 3: Reproducir audios.

Caso de uso 4: Introducir detalles del paciente		
Descripción	Permite al usuario introducir la edad y el sexo del paciente para su posterior predicción.	
Requisitos	RF-1	
	RF-1.2	
Precondiciones	Tener un audio cargado y seleccionado	
Secuencia normal	Paso	Acción
	1	El usuario pincha en hombre o mujer.
	2	El usuario introduce la edad (entero mayor que 0).
	3	El usuario puede dar a predecir.
Postcondiciones	La predicción se realiza de manera satisfactoria ya que los datos del paciente del audio se han leído correctamente.	
Excepciones	Edad no insertada. Edad en formato erróneo.	
Importancia	Alta	
Urgencia	Alta	

Tabla B.5: Caso de uso 4: Introducir detalles del paciente

Caso de uso 5: Ver predicción del audio		
Descripción	Permite al usuario ver el porcentaje de parkinson con el color indicado.	
Requisitos	RF-1	
	RF-1.3	
Precondiciones	Tener un audio cargado, seleccionado y con los datos del paciente introducidos	
Secuencia normal	Paso	Acción
	1	El usuario pincha en predicción.
	2	Se clasifica el audio.
	3	Se muestra la predicción del audio: porcentaje y si tiene parkinson o no.
Postcondiciones	Se muestra el texto del resultado de la predicción.	
Excepciones	Ninguna.	
Importancia	Alta	
Urgencia	Alta	

Tabla B.6: Caso de uso 5: Ver predicción del audio

Caso de uso 6: Elegir audio de la lista		
Descripción	Permite al usuario elegir un audio de los cargados para su procesamiento.	
Requisitos	RF-1.1	
	RF-2.1	
	RF-3.1	
	RF-4.2	
Precondiciones	Tener un audio cargado	
Secuencia normal	Paso	Acción
	1	El usuario elige pinchando un audio de la lista.
	2	El audio se remarca en azul.
Postcondiciones	Es posible la reproducción o predicción del audio seleccionado.	
Excepciones	Audio no seleccionado.	
Importancia	Alta	
Urgencia	Alta	

Tabla B.7: Caso de uso 6: Elegir audio de la lista

Caso de uso 7: Crear modelos		
Descripción	El científico de datos crea los modelos de predicción.	
Requisitos	RF-5	
	RF-5.1	
	RF-5.2	
	RF-5.3	
	RF-5.4	
	RF-5.5	
Precondiciones	Tener las características de los audios preextraídas	
Secuencia normal	Paso	Acción
	1	El científico de datos elige un modelo de python a entrenar.
	2	El científico de datos elige los datos con los que entrenar.
	3	El científico de datos elige el tipo de validación cruzada.
	4	El científico de datos elige como particionar los datos.
	5	El modelo se entrena.
Postcondiciones	Tenemos un modelo ya entrenado.	
Excepciones	Ninguna	
Importancia	Alta	
Urgencia	Alta	

Tabla B.8: Caso de uso 7: Crear modelos

Caso de uso 8: Evaluar modelos		
Descripción	Permite al científico de datos realizar una comparativa de resultados de los modelos.	
Requisitos	RF-6	
	RF-6.1	
	RF-6.2	
	RF-6.3	
Precondiciones	Haber entrenado los modelos	
Secuencia normal	Paso	Acción
	1	El científico de datos extrae las medidas de rendimiento de varios modelos.
	2	El científico de resume las medidas de rendimiento en gráficas o tablas.
	3	El científico de datos visualiza las medidas de rendimiento en gráficas o tablas.
Postcondiciones	Vemos los resultados de la evaluación de los modelos.	
Excepciones	Ninguna.	
Importancia	Alta	
Urgencia	Alta	

Tabla B.9: Caso de uso 8: Evaluar modelos

Apéndice C

Especificación de diseño

C.1. Introducción

En éste apéndice, se describirán cómo están implementados los datos de esta aplicación, cuales son los procedimientos más relevantes de la aplicación, y cómo se organizan los proyectos en paquetes.

C.2. Diseño de datos

En este sección, explicaremos como están organizados tanto el conjunto de audios obtenido, cómo son y como se organizan las características extraídas de los audios, y el diagrama de clases de la aplicación.

Conjunto de audios

El conjunto de audios es el descrito en [3]. También lo hemos descrito en la sección 5.3 de la memoria principal. Estos audios no se encuentran en el repositorio, ya que son de carácter privado. En [3] se describe como: "(...)incluye grabaciones de voz de 50 personas con PD y 50 controles sanos, 25 hombres y 25 mujeres en cada grupo. Todos los participantes son hablantes nativos de español colombiano. La edad de los hombres con PD varía de 33 a 77 años (media 62.2 $\pm$ 11.2), la edad de las mujeres con PD varía de 44 a 75 años (media 60.1 $\pm$ 7.8). Para el caso de los controles sanos, la edad de los hombres varía de 31 a 86 (media 61.2 $\pm$ 11.3) y la edad de las mujeres de 43 a 76 años (media 60.7 $\pm$ 7.7). Por lo tanto, la base de datos está bien equilibrada en términos de edad y género. Las grabaciones fueron capturadas en condiciones de control de ruido, en una cabina de prueba de sonido que

se construyó en la Clínica Noel, en Medellín, Colombia. Los registros se muestrearon a 44100 Hz con 16 bits de resolución, utilizando un micrófono omnidireccional dinámico (Shure, SM 63L) que se usa comúnmente para aplicaciones profesionales. Las grabaciones se capturaron utilizando una tarjeta de audio profesional de hasta 24 bits y de tal forma que admite hasta 96 Kbps de frecuencia de muestreo (M-Audio, Fast Track C400). Estos audios están en formato wav."(p.343). Tendremos 100 audios de una frase y 100 audios de cada una de las 5 palabras. En caso de las vocales, tenemos 300 audios para cada vocal, en vez de 100, ya que hay 3 audios para cada vocal de cada persona. Aunque en el total hacen un número de audios de 2100 audios, como hemos comentado en la memoria, no se puede entrenar con todos a la vez. Tenemos que entrenar los modelos con las características extraídas de cada grupo de 100 o 300 audios (si son vocales) por separado. Esto hace que entrenemos los modelos con bases de datos pequeñas y corran riesgo de sobreajustarse.

Destacar también, que tenemos un archivo tipo excel donde se indica: edad,sexo, UPDRS, HY y tiempo desde la detección de la enfermedad del paciente. Este archivo se encuentra en `doc/masRecursos/PCGITA_metadata.xlsx`.

Características extraídas

Como se ha comentado el la memoria hemos extraído diferentes características de los audios. Todas, se encuentran el el directorio o subdirectorios de `src/CaracterísticasExtraídas`. También, todas ellas están guardadas de manera serializada en formato tipo *numpy* con la herramienta *pickle*. Las características extraídas son las siguientes

- **Características Disvoice (primera fase):** se encuentran en `src/CaracterísticasExtraídas` y se extraen en el *notebook* `Extracción de características.ipynb` del directorio `src`. Están descritas más en detalle en la memoria en la sección 5.5.2 (Aspectos Relevantes - Primera Fase: atributos Disvoice - Modelado del discurso). A grandes rasgos, de los audios de la frase sacamos medidas de fonación, articulación y prosodia; de los audios de cada palabra sacamos medidas de fonación y articulación, y de los audios de cada vocal sacamos únicamente medidas de fonación. Hay un total de **18 conjuntos** diferentes de características. El tamaño de las matrices de características es el siguiente
    - Fonación: 100×30 y 300×30 (para vocales). Se sacan 29 medidas más la clase.

  - Articulación: 100×489. Se sacan 488 medidas más la clase.
  - Prosodia: 100×39. Se sacan 28 medidas más la clase.

- **Características MODIFICADAS Disvoice (segunda fase):** se encuentran en `src/ CaracterísticasExtraídas/EdadYSexo` y en `src/CaracterísticasExtraídas/DivisionSexo` y se extraen en `src/ Extracción de características MODIFICADAS Disvoice.ipynb`. Están descritas más en detalle en la memoria en la sección 5.6.1 (Aspectos Relevantes - Segunda Fase - Modelado del discurso). A grandes rasgos, primero se ha añadido los atributos edad y sexo a las anteriores características, por lo que tendremos otros 18 conjuntos de datos nuevos nuevos, cuya medida de las matrices es igual que la primera fase, pero teniendo 2 columnas más (2 características más). Por otro lado, se han dividido entre hombres y mujeres, por lo que tendremos otros 35 conjuntos de datos más, cada uno con la mitad de instancias que el original (tamaño: tendrán la mitad de filas que los de la fase 1, y tendrán 1 columna más: la edad). Por último, tenemos los conjuntos de características exactos a los de Orozco, solo los MFCC. Estos también están divididos por sexo. De estos últimos tenemos 24 conjuntos, cuyo tamaño es 50×35 (34 medidas más la clase). Por lo hque forman un total de **78 conjuntos** de características diferentes.

- **Características VGGish (tercera fase):** se encuentran en:

  - `src/CaracterísticasExtraídas/vggish/embeddings`
  - `src/CaracterísticasExtraídas/vggish/espectros`.

  Están descritas más en detalle en la memoria en la sección 5.7.1 (Aspectos Relevantes - Tercera Fase - Modelado del discurso), y en los *notebooks*. Se extraen para los tipos de audio de las 5 vocales y la frase los *embeddings*, que genera unas matrices de características de tamaño 100×256 para la frase y 300×256 para cada vocal. Esto es debido a que *VGGish* nos saca 128 *embeddings* para cada fragmento de 25ms de un audio, y nosotros, hemos decidido que para formar un único array de cada audio, hacer la media y desviación de los *embeddings*. Por otro lado, sacamos también los espectros de frecuencia del audio, con la herramienta del preprocesado de VGGish. Los detalles se encuentran en la memoria, en el apartado recientemente indicado. El tamaño de las matrices de datos de los espectros es de 100×128 para la frase y 300×128 para cada vocal. Al igual que antes, es debido a que VGGish nos da 64 espectros de cada fragmento de 25ms, y hemos hecho la media y la desviación para tener un único array.

Diagrama de clases de la aplicación

En este apartado se comentara las clases realizadas por nosotros en el proyecto. Es decir, las clases que se han creado para ayudar a los experimentos y realizar la aplicación. Señalamos que, únicamente se analizaran las realizadas por nosotros, todas la estructura de *VGGish* no será analizada. De echo, por ejemplo, *VGGish2Keras* es un conjunto de módulos con funciones, que no contienen ninguna clase.

Clases de la etapa de experimentación

Cabe destacar que, en los experimentos con los clasificadores, se ha utilizado los *notebooks* de *Jupyter* para su realización. Por ello, en la etapa de realización de experimentos únicamente se realizaron 3 clases, sin ninguna relación entre ellas, las cuales únicamente se usaban en los *notebooks* tanto de extracción de características como en los experimentos con clasificadores.

De estas clases no se hará diagrama, ya que son clases independientes, y se explicarán en el apartado Manual del programador, ver capítulo D.3. De estas 3 clases, 2 se alojan en el directorio `src` y son: `extractorCcas.py` y experimenter.py. Las otra está en el directorio `src/vggish` y se corresponde con el nombre `extractor_ccas_vggish.py`. También tenemos diferentes modulos de carga de datos, estos no son clases, siguen la estructura tipo los cargadores de datos de *Scikit-Learn* (i.e load_iris[1]), funciones dentro de módulos que devuelven objetos de tipo *Bunch*

Clases de la aplicación

. Para la aplicación se han creado las siguientes clases, alojadas en `src/demo`:

- `VentanaInicio` dentro de `main.py`: Clase que contiene toda la estructura gráfica de la ventada de la aplicación. Como suele pasar con los objetos de *tkinter*, no tiene funciones, únicamente se inicializan todos los elementos gráficos (*frames*, botones...), las variables necesarias para cada uno, y los mediadores. En esta clase, todos los elementos gráficos se guardarán como atributos (botones, *frames*...) para un acceso más fácil a ellos y, por tanto, una modificación más sencilla.
- `MediadorVentana` dentro de `MediadorVentana.py`: Clase que tiene como único atributo la ventana anterior. Contiene todos los métodos

[1] https://github.com/scikit-learn/scikit-learn/blob/7813f7efb/sklearn/datasets/base.py#L326

necesarios para la reproducción de los audios y muestra de detalles de la carga y reproducción de los mismos. **Encapsula la comunicación de los objetos gráficos de `VentanaInicio`**, en lo relativo a reproducción y carga de archivos. Debido a que comunica elementos, y los modifica, la mayoría de funciones no devuelve nada.

- `MediadorPrediccion` dentro de `MediadorPrediccion.py`: Tiene dos atributos, la ventana de la aplicación y un objeto tipo *FachadaPrediccion*. Las funciones de este mediador llaman a los métodos para predecir y mostrar gráficos de la fachada y se encarga de cambiar los elementos de la ventana. **Encapsula la comunicación de los objetos gráficos de `VentanaInicio`**, en lo relativo a la predicción y muestra de gráficos.

- `FachadaPrediccion` dentro del fichero `FachadaPrediccion.py` del paquete predicción: Tiene dos atributos, el modelo *keras VGGish* para la extracción de características de los audios y el modelo *MLP* con 10 neuronas para la predicción. Se encarga de interactuar con los módulos de *VGGish2Keras* para devolver los resultados de la predicción al mediador.

- Módulos propios de *VGGish* (*VGGish2Keras*): `mel_features.py`, `vggish_keras.py`, `vggish_params.py`. No son clases, únicamente contienen métodos.

El diagrama de clases se puede ver en la Figura C.1.

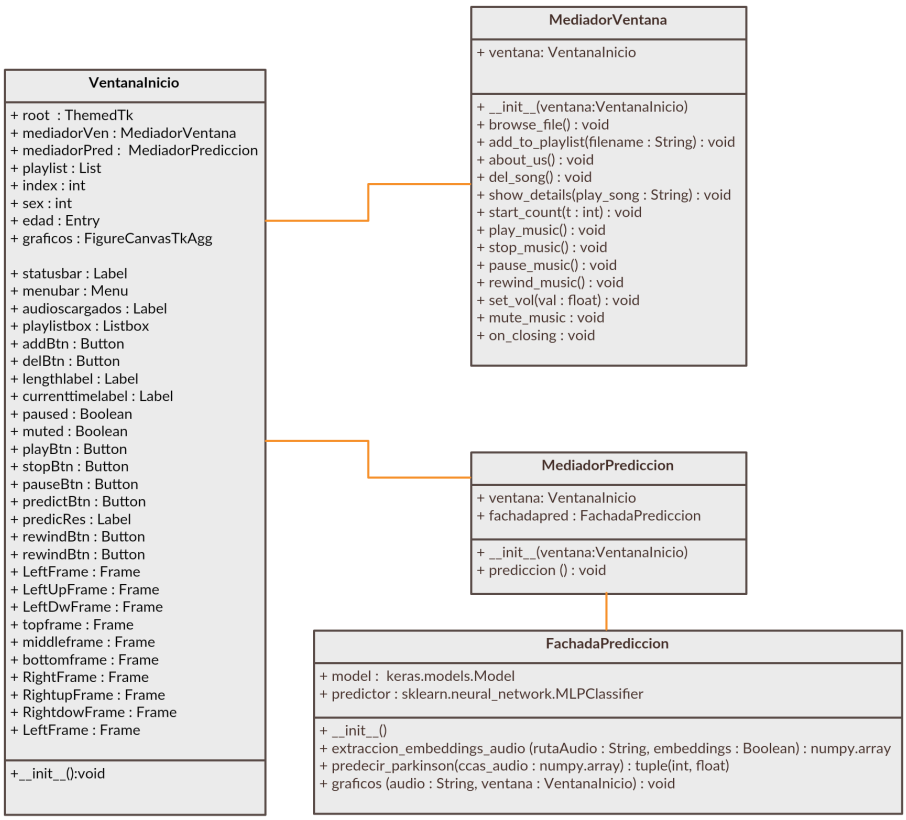

Figura C.1: Diagrama de clases de la aplicación

C.3. Diseño procedimental

En esta sección se mostrarán los diagramas de secuencia respectivos a las 3 tareas principales de la aplicación: cargar audios (ver Figura C.2), reproducir audios y predicción de audios-muestra de gráficos (se ejecutan a la vez, cuando predices un audio, automáticamente se muestra sus gráficas).

Diagramas de secuencia

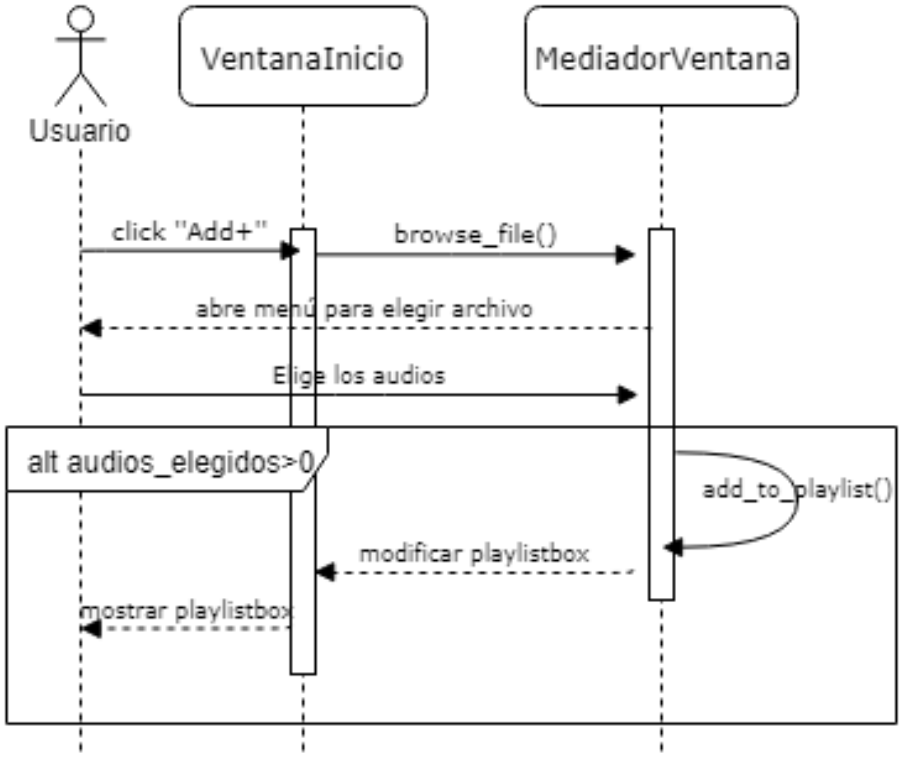

Figura C.2: Diagrama de secuencia para cargar un audio

C.4. Diseño arquitectónico

Patrón Mediador

Patrón Fachada

Paquetes

Apéndice D

Documentación técnica de programación

D.1. Introducción

D.2. Estructura de directorios

D.3. Manual del programador

D.4. Compilación, instalación y ejecución del proyecto

D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

E.1. Introducción

E.2. Requisitos de usuarios

E.3. Instalación

E.4. Manual del usuario

Bibliografía

[1] Giovanni Dimauro, Vincenzo Di Nicola, Vitoantonio Bevilacqua, Danilo Caivano, and Francesco Girardi. Assessment of speech intelligibility in parkinson's disease using a speech-to-text system. *IEEE Access*, pages 22199–22208, 2017.

[2] Giovanni Dimauro, Vincenzo Di Nicola, Vitoantonio Bevilacqua, Francesco Girardi, and Vito Napoletano. Voxtester, software for digital evaluation of speech changes in parkinson disease. *Proc. IEEE Int. Symp. Med. Meas. Appl. (MeMeA)*, pages 1–6, 2016.

[3] J. R. Orozco-Arroyave, J. D. Arias-Londoño, J. F. Vargas-Bonilla, M. González-Rátiva, and E. Nöth. New spanish speech corpus database for the analysis of people suffering from parkinson's disease. *Proceedings of the 9th Language Resources and Evaluation Conference (LREC)*, pages 342–347, 2014.

[4] J. R. Orozco-Arroyave, F. Hönig, J. D. Arias-Londoño, J. F. Vargas-Bonilla, K. Daqrouq, S. Skodda, J. Rusz, and E. Nöth. Automatic detection of parkinson's disease in running speech spoken in three different languages. *The Journal of the Acoustical Society of America*, 139:481–500, Enero 2016.

[5] Alex Osterwalder. Business model design and innovation, 2007.