1

**Introducción**

El objetivo del presente proyecto es crear un intérprete graficador que permita generar y guardar tablature de manera autónoma y digital. Este desarrollo permitirá analizar los sonidos de las notas emitidas por el bajo eléctrico para descomponerlos en frecuencias y poder determinar en qué posición o traste fue generada y en qué cuerda.

Para poder llevar a cabo el desarrollo del software se debe tener en cuenta que el mercado cuenta con múltiples tipos y marcas de bajos eléctricos que varían tanto en número de cuerdas como en la tonalidad que manejan, los bajos eléctricos que se usan más comúnmente son aquellos denominados estándar, ya que permiten el dinamismo que se necesita si se está empezando en el mundo musical. Estos serán el enfoque del proyecto ya que, al tener un mayor grado de aceptación, permitirá que el software tenga una mayor expansión de uso.

Además, integra una interfaz gráfica amigable con el usuario, permitiendo su uso intuitivo acertado. El bajo eléctrico estándar cuenta con cuatro cuerdas y afinación Sol, Re, La, Mi.

* 1. Objetivos
     1. Objetivo general

El objetivo general de este proyecto es el diseño e implementación de un software que realice la gráfica de la tablatura de notas generadas mediante un bajo eléctrico en tiempo real; el cuál se llevará a cabo mediante los diferentes objetivos específicos detallados a continuación.

* + 1. Objetivos específicos

Los objetivos específicos para lograr el objetivo general son los siguientes:

* Elaborar el plan de proyecto
* Levantar requerimientos.
* Definir los rangos de frecuencia de las notas del bajo eléctrico.
* Realizar el análisis del sistema.
* Realizar el diseño del sistema.
* Implementar el proyecto.
* Realizar las pruebas.
  1. Limitaciones del proyecto
     1. Timbre

Los bajos eléctricos se componen por varios elementos, los cuales definen su diseño y su timbre al tocar, uno de los componentes que alteran el sonido es el calibre de las cuerdas que se usan, ya que estas de entrada modifican la nota que producen dependiendo de su entorchado y el material del que estén echas, los bajos de cuatro cuerdas de serie vienen con las afinaciones estándar (Sol, Re, La, Mi) y con cuerdas de entorchado semiliso, permitiendo versatilidad.

Las pastillas, que funcionan captando las vibraciones de las cuerdas y transmitiéndolas al amplificador, también alteran el timbre, ya que dependiendo de la distribución de las mismas captaran en mayor o menor medida estas vibraciones, los bajos estándar se componen con al menos 2 de estas pastillas cubriendo las cuatro cuerdas del mismo.

El alma del bajo eléctrico, es una varilla delgada de acero que se encuentra entre el diapasón del bajo y el mástil, se emplea para modificar la curvatura de este mástil, que es donde se encuentran las cuerdas, modificar la curvatura ocasionara que el instrumento cambie su timbre, existiendo un estándar de medición que indica la curvatura optima del instrumento.

Existen a su vez diferentes maneras de tocar un bajo eléctrico, que alteran el timbre de la nota siendo la más popular el “fingerstyle” que consiste en pulsar las cuerdas con los dedos índice y corazón de la mano derecha, pero no es la única su variación más cercana es pulsar las cuerdas con ayuda de un “pick” o “púa”, y la variación más agresiva usada es el “slap”, que consiste en golpear directamente la cuerda con el pulgar, cada técnica genera distorsión adicional en las notas, por lo que las pastillas no captan de manera muy clara las vibraciones, ocasionando un timbre distante entre cada técnica.

Al existir tantos factores que podrían alterar el funcionamiento del software se ha decidido delimitar bastante el alcance del proyecto, definiendo entonces que se centrara en un bajo con cuatro cuerdas (Sol, Re, La, Mi), con entorche semiliso, que contengan mínimo 2 pastillas completas, que su alma esté debidamente acomodada y que la técnica utilizada sea “figerStyle”.

* 1. Estructura de la memoria

En el capítulo 2 se mencionan las investigaciones realizadas sobre implementaciones de software y herramientas utilizadas en el ámbito musical. En el capítulo 3 se dan a conocer el entorno de desarrollo, el modelo para desarrollar dicho software y se muestra una visión general del lenguaje de programación empleado. En el capítulo 4 se explica el desarrollo de la aplicación, conociendo las especificaciones y requisitos de la misma. En el capítulo 5 se expone el procedimiento de validación experimental y los resultados obtenidos. Finalmente se representan las conclusiones de este proyecto y los posibles trabajos futuros.

2

**Estado del arte**

En este capítulo se dan a conocer los trabajos y publicaciones previos relacionados con los objetivos de este proyecto. En la sección 2.1 se mencionan trabajos relacionados al manejo de sonidos, y en la sección 2.2 se muestran investigaciones sobre herramientas actuales que usan este mismo principio.

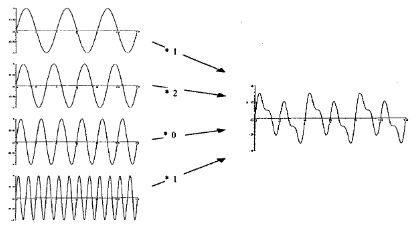
* 1. Sonido

El sonido audible consiste en ondas sonoras y ondas acústicas que se producen cuando las oscilaciones de la presión de aire, son convertidas en ondas mecánicas en el oído y percibidas por el cerebro, las cuales pueden viajar a través de cualquier medio sólido, liquido o gaseoso.

En la definición del sonido también debe considerarse tanto el fenómeno físico como el sicoacústico, ya que bajo la ausencia de un oyente puede existir un evento sonoro, pero no el evento auditivo.

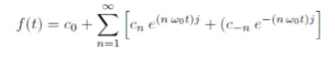
El sonido se compone de varios componentes como el tono, el cual es determinado por la frecuencia fundamental de las ondas sonoras medidas en ciclos, este es quien nos permite distinguir entre sonidos graves y agudos. La duración, que no es más que el tiempo de propagación de la onda. El timbre acompaña a la frecuencia y es el atributo que nos permite diferenciar entre dos sonidos con igual frecuencia

Los autores proponen la utilización de la transformada de Fourier para entender los sonidos y la música de una manera más matemática ya que esta expresa que toda función periódica puede expresarse como la suma infinita de funciones seno o coseno que son múltiplos enteros n de la frecuencia, hallando su frecuencia fundamental, y a su vez cada término de seno y coseno se le conocen como armónicos.

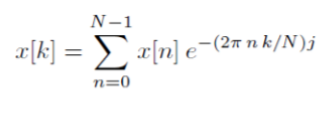


**Figura 2.1:** Una señal genérica se transforma por una sumatoria de señales sinusoidales

Logrando condensar su algoritmo se logra que realice los mismos cálculos, pero en un menor tiempo, dando lugar a la transformada rápida de Fourier. Estos infieren pues que la forma matemática más sencilla para la síntesis auditiva es utilizar las series de Fourier, pudiendo hallar la frecuencia del sonido basándose solo en su amplitud y su fase.



**Figura 2**.**2**: Transformada de Fourier



**Figura 2.3**: Transformada rápida de Fourier.

* 1. Herramientas actuales

En la actualidad el uso y tratamiento de señales tiene bastantes usos prácticos.

Algunos autores hacen mención de que el estudio de la síntesis de los sonidos de los instrumentos musicales es un tema motivador que interesa a los estudiantes de ciencias básicas, ingenierías y música ya que entrega una aproximación al tema desde diferentes perspectivas ya que el modelamiento de transmisión de ondas puede ser abordado desde la parte física y matemática a la vez, permitiendo versatilidad de aprendizaje sobre estos temas.

Algunos abordan este tema basándose en el funcionamiento de los afinadores actuales ya que el funcionamiento de estos dispositivos funciona de manera similar a lo planteado anteriormente, basados en que los sonidos de las cuerdas al aire de los instrumentos de cuerda tienen su propia frecuencia fundamental, la cual comparándola con registros anteriormente diseñados puede comunicarle al usuario cuando su instrumento esta afinado.

Usando este principio de tratamiento de señales y con algoritmos de recomposición y tratamiento de datos es como google implementa su búsqueda por voz implantado en nuestros teléfonos Smartphone, ya que utilizan la descomposición de los sonidos en ondas más manejables para compararlas con su inmensa base de datos, una red neuronal es la que se encarga de encontrar las similitudes para entender que es lo que se le está solicitando que busque, por lo que en ultimas somos los usuarios quienes no encargamos de entrenar a la red para que funcione más eficientemente.

En el campo de la robótica las utilizaciones de descomposición de sonidos en ondas manejables permiten que en la actualidad el androide llamado sophia pueda entender y dialogar con personas de su entorno, realizando tanto síntesis de lo que escucha como su proceso inverso, pudiendo así hacer uso de señales computacionales para comunicarse eficazmente.

3

**Análisis**

En este capítulo se presenta la fase de análisis, parte inicial de cualquier proyecto de software, donde se definen los requisitos y se enseña una vista global de la arquitectura que se desea implementar en el sistema. Ya que en el siguiente capítulo se tendrá en cuenta este apartado como base para el diseño de todos los aspectos de la aplicación. Por ello que la fase de análisis de se suma importancia para el futuro del software. Aquí es donde se sentarán las bases del proyecto.

El proceso a seguir se basa en primer lugar en una buena definición de requisitos y en elegir las metodologías más apropiadas para el desarrollo apropiado del proyecto.

La arquitectura del sistema tendrá el formato Modelo – Vista – Controlador constituyendo las funcionalidades de cada uno de estos componentes.

A continuación, se desglosan las funcionalidades y las características a modo de catálogo de requisitos, teniendo en cuenta tanto los funcionales como no funcionales

* 1. Catálogo de requisitos

El catálogo de requisitos es la especificación del comportamiento que se espera de cualquier proyecto de software, estudiando las necesidades de los usuarios se ha predefinido una serie de requisitos que se consideran indispensables para el proyecto. A continuación, se muestra una enumeración y breve descripción de los requisitos establecidos para el diseño y desarrollo de la aplicación.

* + 1. Requisitos funcionales

Los requisitos funcionales describen todas las interacciones que tendrán los usuarios con el software.

* + - 1. Uso de la aplicación

**RF1: Graficar**

1. El software deberá iniciar la operación únicamente cuando el usuario lo indique.
2. Una vez iniciado las operaciones el sistema empezara a escuchar las notas que el usuario genere.
3. A medida que va escuchando el sistema va guardado el registro en un archivo ‘Audio.wav’
4. El sistema carga el archivo ‘Audio.wav’ y aplica la transformada rápida de Fourier para hallar un magnitud y fase.
5. El sistema encuentra la frecuencia fundamental a partir de la magnitud y fase.
6. El sistema compara la frecuencia con los rangos definidos y guardara su posición en un vector de posiciones.
7. El sistema guarda las posiciones en un archivo ‘save.bin’
8. El sistema grafica en la ventana las notas una vez se acabe el tiempo.

**RF2: Re visualizar**

1. El sistema deberá volver a graficar la última tablatura generada
2. El sistema cargara el archivo ‘save.bin’
3. El sistema graficara en la ventana las notas una vez que haya cargado las posiciones.
   * + 1. Sistema

**RF3: Pantalla principal**

1. La pantalla principal de la aplicación constara de tres botones:
   1. **Graficar:** El sistema mostrará la ventana donde se graficará la tablatura.
   2. **Re visualizar:** El sistema mostrará la ventana con las últimas posiciones guardadas
   3. **Salir:** El sistema terminara las operaciones y se cerrara.
      1. Requisitos no funcionales

Requisitos complementarios que especifican criterios que juzgan operaciones del sistema en lugar de su comportamiento.

**RNF1: Portabilidad**

1. Disponibilidad de operación sobre cualquier portátil con micrófono y python.

**RNF2: Interfaz y usabilidad**

1. El software debe constar de una interfaz sencilla, atractiva e intuitiva.
2. La tablatura generada debe ser de fácil entendimiento.

**RNF3: Rendimiento**

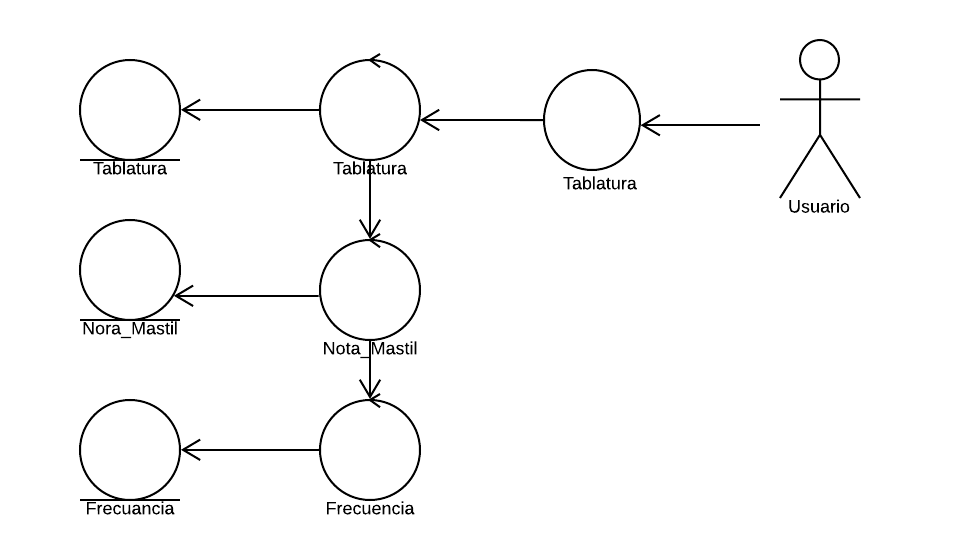
1. Se esperan tiempos de respuesta no superiores a 2 segundos en la petición de graficación una vez se acabe el tiempo de escucha.
2. Se esperan tiempos de respuesta no superiores a 2 segundos de carga para re visualizar la tablatura.
   1. Diseño Modelo- Vista- Controlador

El diseño Modelo – Vista – Controlador, o también llamado diseño MVC, está orientado a objetos dentro de la programación del software. Este permite separar la programación del sistema en tres campos. El Modelo contiene información del sistema, como pueden ser: las variables, las especificaciones, etc. El Controlador se encarga de como presentar lo que se encuentra en el modelo al usuario en una interfaz amigable e intuitiva. Finalmente, la Vista es la encargada de enseñar lo que le ordene el controlador.

* + 1. Comunicación entre campos

La comunicación entre los tres campos no es absoluta. El Controlador puede comunicarse con el Método sin ningún problema, y puede obtener toda la información que necesite cuando lo requiera, ya que el Controlador es el encargado de poner en pantalla el Modelo. También el Controlador puede comunicarse con la Vista para poder gestionar la interfaz de usuario. Mientras que la comunicación entre el Modelo y la Vista, idealmente no se debería contemplar, debido a que el Modelo es independiente de la interfaz de usuario. También se podría realizar un Modelo que se adaptara a la vista, pero esto afectaría a la reusabilidad del código y haría más complicada la búsqueda de errores durante la depuración.

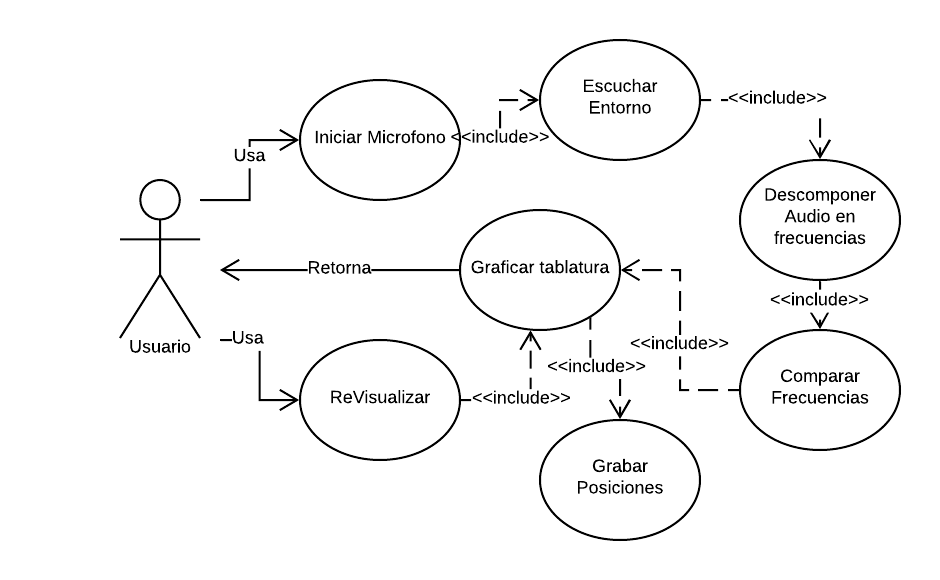
Además, la comunicación entre Vista y Controlador se puede realizar de tres maneras. Una forma es que la Vista a través de acciones pueda comunicarle al Controlador lo que el usuario está realizando en la interfaz. La segunda forma es que el Controlador tenga el control de lo que pase en la vista través de delegados. Por último, análogo a la anterior, el Controlador gestiona la información que quiera mostrar la vista a través de una fuente de datos. La comunicación entre el Modelo y el Controlador noes directa. Cuando el Modelo necesita actualizar información debe utilizar notificaciones, para notificarle al Controlador los nuevos cambios.



**Figura 3.1:** Modelo – Vista - Controlador

* 1. Diagrama de casos de uso

El diagrama de casos de uso representa la forma en como un usuario (actor) opera con el sistema en desarrollo, además de la forma, tipo y orden en como los elementos interactúan para cumplir con las funcionalidades del sistema. A continuación, el diagrama para el proyecto propuesto.



**Figura 3.2:** Diagrama de casos de uso.

* 1. Diagrama de clases
  2. Diagrama Entidad-Relación
  3. Diagrama de despliegue
  4. Diagramas de actividad

3.7.1 Diagrama de actividad Comparar

3.7.2 Diagrama de actividad Graficar

3.7.3 Diagrama de actividad GraficarL

3.7.4 Diagrama de actividad PassiveListen