DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

TESIS PREVIO A LA OBTENCIÓN DEL TITULO DE INGENIERÍA ELECTRONICA

TEMA: IMPLEMENTACIÓN DE UN SISTEMA DE CONVERSIÓN DE AUDIO A TEXTO EN TIEMPO REAL PARA PERSONAS CON DISCAPACIDAD AUDITIVA.

AUTORES: ORBE OROZCO LUIS IVÁN
ZURITA ULLAURI VÍCTOR JULIO

DIRECTOR: ING. SAENZ FABIAN
CODIRECTOR: ING ROMERO CARLOS

SANGOLQUÍ – ECUADOR

AÑO 2015

Certificado de tutoría

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICADO

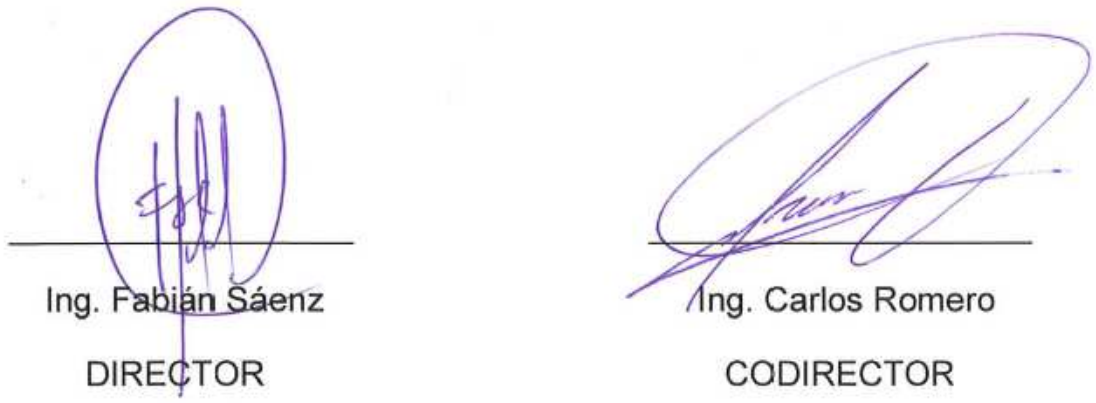
Ing. Fabián Sáenz
Ing. Carlos Romero

CERTIFICAN

El trabajo titulado “Implementación de un sistema de conversión de audio a texto en tiempo real para personas con discapacidad auditiva”, desarrollado por Luis Iván Orbe Orozco y Víctor Julio Zurita Ullauri, ha sido guiado y revisado periódicamente y cumple con todas las normas estatutarias establecidas por la Universidad de las Fuerzas Armadas (ESPE), en el Reglamento de Estudiantes.

Debido a que se trata de un trabajo de investigación recomendamos su publicación.

Sangolquí, 04 de Mayo del 2015



Ing. Fabián Sáenz
DIRECTOR

Ing. Carlos Romero
CODIRECTOR

Autoría de responsabilidad

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORÍA DE RESPONSABILIDAD

LUIS IVÁN ORBE OROZCO

VÍCTOR JULIO ZURITA ULLAURI

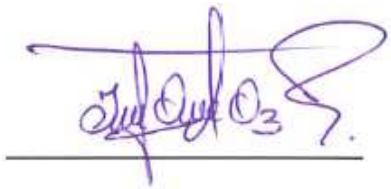
DECLARO QUE:

El proyecto de grado denominado "Implementación de un sistema de conversión de audio a texto en tiempo real para personas con discapacidad auditiva", ha sido desarrollado en base a una investigación exhaustiva, respetando derechos intelectuales de terceras personas, conforme las citas que constan al pie, de las páginas correspondientes, y cuyas fuentes se incorporan en la bibliografía.

Consecuentemente el diseño e implementación del presente trabajo es de nuestra total autoría.

En virtud de esta declaración nos responsabilizamos del contenido, veracidad y alcance del proyecto de grado en mención.

Sangolquí, 04 de Mayo, del 2015



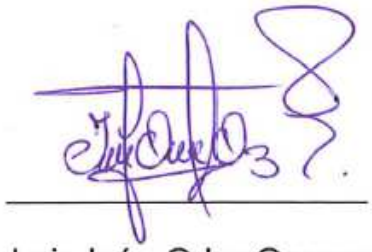
Luis Iván Orbe Orozco
172175052-7

Víctor Julio Zurita Ullauri
172118276-2

Autorización publicación biblioteca virtual

Nosotros, Orbe Orozco Luis Iván y Zurita Ullauri Víctor Julio

Autorizamos a la Universidad de las Fuerzas Armadas –ESPE- la publicación en la biblioteca virtual de la institución del proyecto de grado titulado: "***IMPLEMENTACION DE UN SISTEMA DE CONVERSION DE AUDIO A TEXTO EN TIEMPO REAL PARA PERSONAS CON DISCAPACIDAD AUDITIVA***." Cuyo contenido, ideas y criterios son de nuestra exclusiva responsabilidad y autoría.



Luis Iván Orbe Orozco
172175052-7

Víctor Julio Zurita Ullauri
172118276-2

Sangolquí, 04 de Mayo del 2015

DEDICATORIA

Dedicamos este logro a Dios y a nuestras respectivas familias, ya que con su apoyo y confianza en todo momento, hemos puesto todo el esfuerzo y dedicación para alcanzar esta tan anhelada meta.

AGRADECIMIENTO

A Dios por darnos la salud y la vida para culminar nuestra carrera con éxito, a nuestros respectivos padres, hermanos, compañeros y maestros que nos han permitido crecer como persona y como profesionales.

A nuestro director de tesis Ing. Fabián Sáenz y a nuestro codirector de tesis Ing. Carlos Romero que supieron guiarnos para alcanzar nuestro objetivo.

Y a todos aquellos amigos y personas que de una u otra manera han contribuido para que esto sea posible.

Luis Iván Orbe Orozco
Víctor Julio Zurita Ullauri

ÍNDICE DE GENERAL

Índice de contendido

Índice de figuras

Índice de Tablas

RESUMEN

La discriminación crea problemas a nivel social ya que limita el desarrollo personal de las personas con algún tipo de problemas, el proyecto busca facilitar la comunicación y superar las barreras de la discapacidad auditiva, ya que una de las principales necesidades de todo ser humano es el de poder comunicarse correctamente. En el Ecuador existe muy poca penetración de sistemas tecnológicos orientados a ayudar y atender las necesidades de las personas discapacitadas, pero gracias a que en los últimos años se ha podido observar un gran avance tecnológico en nuestro país, fue posible la implementación del presente proyecto de investigación en donde se vio la necesidad de ayudar a las personas con discapacidad auditiva, creando un sistema que beneficie a las mismas. En el presente proyecto se realizó el estudio y análisis de las técnicas y métodos empleados para la conversión de audio a texto, en donde se propuso utilizar las herramientas de CMU Sphinx, un software de reconocimiento de voz, del tipo open (source software libre). Originado en la Universidad de Carnegie Mellon (Pensilvania Estados Unidos), quienes ofrecen sus herramientas para el desarrollo de nuevos proyectos. La utilización del kit de herramientas Sphinx tales como el modelo de lenguaje, modelo acústico y modelo diccionario fueron adaptados y entrenados para crear nuestro propio sistema de reconocimiento de voz con idioma español ecuatoriano. El rendimiento del nuevo sistema propuesto, fue entrenado, medido, probado y presentado a través de una interfaz humano-máquina, pudiendo así lograr que un individuo común se comunique de manera adecuada con personas con discapacidad auditiva, haciendo que estos entiendan y comprendan exitosamente la información que se les ha transmitido. El sistema adiciona la característica de cargar un video de formato específico y generar los subtítulos del mismo.

PALABRAS CLAVE:

- CMU-SPHINX
- MODELO DE LENGUAJE
- MODELO ACÚSTICO
- MODELO DICCIONARIO
- ENTRENAMIENTO
- NUEVOS SISTEMA
- CÓDIGO ABIERTO

ABSTRACT

The discrimination creates socially problems, limits the personal development of the people with any problems. The purpose of the project was to facilitate communication and overcome the barriers of hearing impairment, because one of the main needs of every human being is the ability to communicate properly. In Ecuador there is little penetration of technological systems designed to assist and meet the needs of disabled people, but thanks that in recent years it has been observed technological breakthroughs in our country, was possible implementation of this research project, where was the need to help people with hearing impairment creating a system that benefits them. In this project the study and analysis was performed about of techniques and methods for converting speech to text, where it was proposed to use the tools of EMU Sphinx. This is a voice recognition software and it's type open source, originated in Carnegie Mellon University (Pennsylvania USA), they offer tools for developing new projects. The system is trained on Linux and then released like a new version generated library is used in Windows to make the graphical interface. The training was fed enough to be very robust and strong. Using the toolkit Sphinx, such as the language model, acoustic model and dictionary model, these were adapted and trained to create our own system speech recognition with Spanish language Ecuador. The performance of the proposed system was trained, measured, tested and presented in a human-machine interface, achieving a common person to communicate properly with the hearing impaired, was that these people successfully understand the information which has been transmitted to them. The system adds the feature to load a specific video format and generate the subtitle of it.

KEYWORDS:

- CMU SPHINX
- LANGUAGE MODEL
- ACOUSTIC MODE
- DICTIONARY MODEL
- TRAINING
- NEW SYSTEM
- OPEN SOURCE

CAPITULO I
PRESENTACIÓN

1.1. INTRODUCCIÓN.

El Procesamiento de señales tiene una larga e interesante historia, siendo esta una tecnología que posee un conjunto de disciplinas, entre las cuales se puede encuentran a las telecomunicaciones. Hoy en día, esta afirmación teórica se la puede encontrar en la implementación de la televisión digital, los sistema de información y el entretenimiento multimedia, por citar algunos ejemplos. A medida que los sistemas de comunicación se van convirtiendo en sistemas sin hilos, móviles y multifunción, la importancia de un procesamiento de señales, es cada vez más sofisticada y a la vez, se hace más relevante.

El Procesamiento de señales se refiere a la transformación, manipulación y representación de las señales y de la información e importancia que estas contienen. Es importante entender que este sistema debe funcionen en tiempo real, lo cual es el objetivo del presente trabajo, lo que significa que mediante un proceso de educamiento dentro del sistema, las muestras de audio a la salida, se calculan al mismo tiempo y a la misma velocidad, emparejadas con respecto del texto y obteniendo así una respuesta rápida y con retardos despreciables.

Otro tipo de problemas del tratamiento de señales al que se enfrenta es la interpretación de señales. Por ejemplo, en un sistema de reconocimiento de voz el objetivo es comprender la señal de entrada. Típicamente, un sistema como éste aplicará un procesado previo de filtrado, estimación de parámetros, etc. seguido por un sistema de reconocimiento de patrones que produzca una representación simbólica. Realmente, son muchas las aplicaciones que requieren esta especificación. El tratamiento de señales tanto en función del tiempo como de la frecuencia, es comúnmente utilizado en sistema de comunicaciones, radar, codificación y realce de voz y vídeo.

A finales de los años 90, según la O.M.S. (organización mundial de la salud), el 15 por ciento de la población mundial tiene alguna discapacidad física, psíquica o sensorial que dificulta su desarrollo personal e integración social, tal porcentaje equivale a 900 millones de personas. El Ecuador cuenta con 15´630.000 habitantes de los cuales el 5.64% poseen algún tipo de discapacidad. **fuente:** (de Castro & Morales, 2006)

Hasta 1950 la atención a las personas con capacidades especiales se realizaba bajo criterios de caridad y beneficencia, mas a partir de este año, la ayuda se fue tecnificando. Por los años 70 los organismos públicos ampliaron la cobertura de atención respecto de salud, educación y bienestar social y en la década de los 80 las Naciones Unidas decretaron el año del impedido que posteriormente se convirtió en la década del impedido donde se promovió a nivel mundial políticas para inserción social a las personas con capacidades especiales. Históricamente y mediante el transcurso de los años, se ha venido observado una evolución de los dispositivos tecnológicos, en donde cada vez más, el desarrollo de la tecnología le extiende la mano a la discapacidad, para que cualquier persona con problemas, pueda aumentar su confianza en sí misma**. fuente:** (Koon & Vega, 2000)

Finalmente el 29 de Julio de 1992 se expide la Ley 180 sobre Discapacidades la cual posteriormente llevó a la creación de CONADIS (Consejo Nacional de Discapacidades del Ecuador). Actualmente existen leyes para incluir a personas con capacidades especiales tanto en empresas públicas, como privadas y se observa en la televisión nacional, canales que ofrecen noticias y otros programas en lenguaje de signos.

En la actualidad, se han propuesto numerosos proyectos tecnológicos para solucionar este tipo de dificultades, sobre todo si hablamos de problemas auditivos, los cuales son la principal causa que dificulta el lenguaje y la comunicación. El problema que existe en este tipo de personas con discapacidad auditiva es que no tiene una herramienta que les ayude a tener una comunicación adecuada y entendible con las demás personas comunes, la mayoría de ellas solo pueden comunicarse entre sí mismas por medio del lenguajes de señas y signos. **fuente:** (Silvestre, 2010)

Experimentos tecnológicos realizados, nos demuestran que hoy en día se puede adaptar software para las personas con discapacidad con las tecnologías adecuadas y accesibles que permiten vivir con mayor calidad de vida. Así por ejemplo, Microsoft pone al alcance de las personas, opciones como; voz, lupa, teclado en pantalla, narrador y notificaciones visuales. Así mismo algunas otras empresas como; Apple, Samsung y Linux no pueden quedar atrás e introducen aplicaciones cada vez más modernas y novedosas para mejorar el estilo de vida de las personas con discapacidad. Actualmente no existen sistemas robustos y eficientes que ayuden a la correcta comunicación de las personas con discapacidad auditiva, sin embargo se están desarrollando proyectos los cuales tienen como propósito hacer posible la comunicación y mejorar la calidad de vida de las personas con discapacidades.

Las limitaciones de los sistemas actuales que sirven para ayudar a las personas con discapacidad, es su alto costo de obtención, ya que la mayoría de las personas discapacitadas no cuentan con un buen ingreso económico laboral para adquirir este tipo de sistemas, incluso existen personas discapacitadas que no tienen ingreso alguno debido a la discriminación existente. **fuente:** (Martin & Diego, 1998)

1.2. JUSTIFICACIÓN E IMPORTANCIA

Una de las principales necesidades de todo ser humano es el de poder comunicarse correctamente con las personas de su entorno, en el caso de las personas con discapacidad auditiva este problema es cotidiano. La gente discrimina a las personas con discapacidad auditiva ya que consideran que los retrasan o que son una carga que no les corresponde, debido a su ignorancia al no poder lidiar con una situación no tan común a lo que acostumbran, generando algunas veces inconscientemente discriminación. La discriminación crea problemas a nivel social ya que limita el desarrollo personal de la persona discriminada, el proyecto busca facilitar la comunicación y superar las barreras de la discapacidad auditiva.

En el Ecuador existe muy poca penetración de sistemas tecnológicos orientados a ayudar y atender las necesidades de las personas discapacitadas, ya que este tipo de personas han quedado, de cierta manera, discriminadas por la sociedad. Sin embargo gracias a que en los últimos años se ha podido observar un gran avance tecnológico en nuestro país, es posible la implementación del presente proyecto de investigación en donde se ve la necesidad de ayudar a las personas con discapacidad auditiva, creando un sistema que beneficie a las mismas.

El plan nacional del buen vivir tiene entre sus objetivos y políticas promover el acceso a la información y a las nuevas tecnologías de la comunicación para incorporar a la población a esta sociedad tecnológica. Gran parte de los objetivos referentes a este tema tienen de por medio la equidad e igualdad entre las personas en el Ecuador. **fuente:** (Plan Nacional del Buen Vivir, 2013)

Hoy en día existen algunas alternativas que ayudan a las personas con discapacidad auditiva, por ejemplo el estado y algunas entidades cuando dan información por medio televisivo, ponen traductores que por medio del lenguaje de señas y signos traducen toda la información para que las personas con discapacidad auditiva entiendan, pero la mayoría de veces no se da este tipo de traducciones sino muy rara vez aparecen, ya que no siempre existe la

preocupación de tener una igualdad entre las personas en donde todos podamos entender y tener una comunicación adecuada.

Lo importante de este proyecto es ayudar a las personas con discapacidad auditiva, para que estas tengan una adecuada y correcta comunicación no solo con las demás personas, sino también que puedan entender reproducciones multimedia como videos, transmisiones televisivas o videoconferencias por medio del subtitulado automático del sistema de conversión de audio a texto. Este nuevo sistema tiene una característica muy importante el cual es su bajo costo de adquisición, puesto que su desarrollo está limitado únicamente al conocimiento, lo cual lo hace un sistema totalmente accesible con todos los beneficios de tener una herramienta robusta que nos permitiría tener una comunicación fluida con palabras acertadas.

La ventaja de este proyecto es la capacidad de añadir nuevas palabras y educar al software para una respuesta confiable y rápida, permitiendo un sistema similar al de los diccionarios de los celulares, generando un sistema dinámico con respuestas acertadas que incluye palabras locales y modismos propios del lenguaje español. **fuente:** (Miranda de Lara, 2007)

El objetivo principal de este tema, es el estudio y la investigación para ver la situación actual de la tecnología y utilizarla para el apoyo a las personas con discapacidad auditiva, lo que también, hace que sea un paso más en el desarrollo del proyecto del plan nacional del buen vivir.

1.3. ALCANCE DEL PROYECTO.

El presente proyecto consistió en desarrollar un sistema de conversión de audio a texto en tiempo real mediante una interfaz humano-máquina, pudiendo así lograr que un individuo común se comunique de manera adecuada con personas con discapacidad auditiva, haciendo que estos entiendan y comprendan exitosamente la información que se les pretende transmitir. El sistema adiciono la característica de cargar un video de formato específico y generar los subtítulos del mismo.

En primera instancia, se determinó una plataforma que sirve como base para desarrollar el software, de tal manera que si buscamos controladores y librerías, dicha plataforma tenga la característica importante de adaptarlos para poder modificarlos, y así posteriormente, implementar dentro de este, un sistema más robusto a beneficio de las personas con discapacidad auditiva.

Dicho esto es importante mencionar que utilizaremos una herramienta de software adecuada, la cuales serán la puerta de conexión intermedia hacia el sistema de conversión audio a texto.

Posteriormente se hizo el desarrollo del software, el cual conllevo un proceso vista de mejor manera en la figura 1 y figura 2, se modificaron los controladores y librerías a nuestro beneficio para mejorar el sistema en el dominio del tiempo y de la frecuencia, haciéndolo más robusto tanto al ruido como a la lógica de interpretación de las palabras, introduciendo diccionarios para que el sistema obedezca de una manera ágil y fluida a las palabras de su locutor.

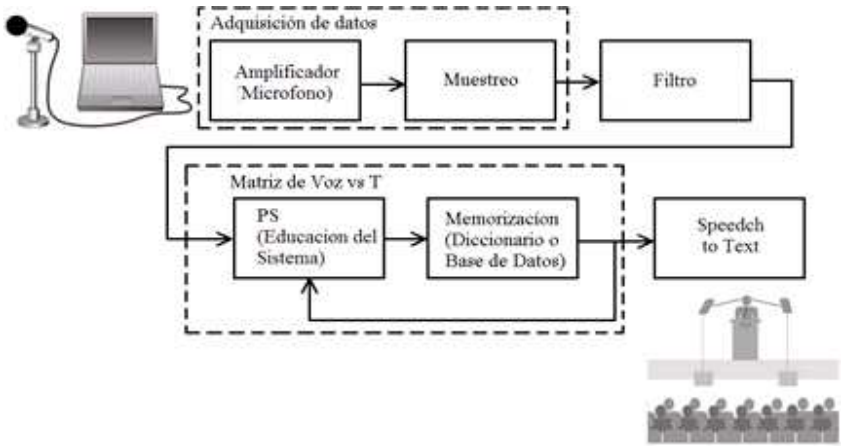


Figura 1. *Diagrama de bloques para conferencias*

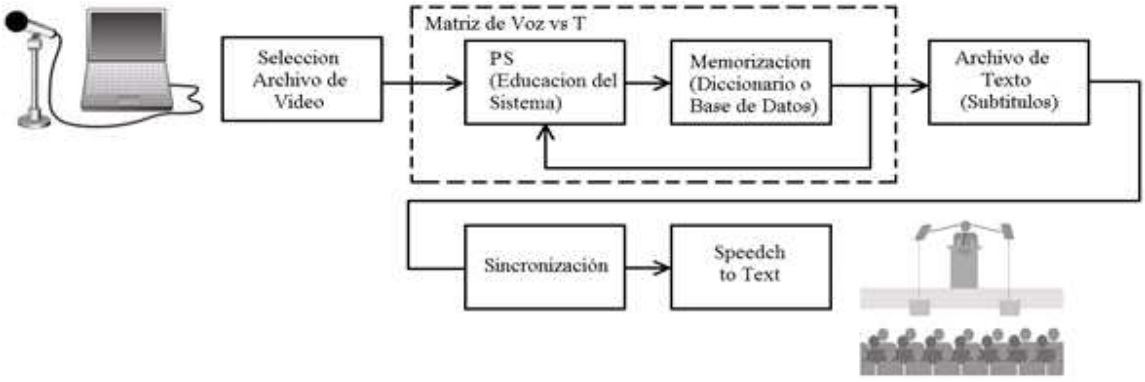


Figura 2. *Diagrama de bloques para reproducción de video*

Primer escenario.- En el caso de la figura 1, la primera operación que se desarrolló, es el bloque de la adquisición de datos, realizando un procesamiento de la señal de voz de entrada al sistema, con objeto de extraer la información acústica relevante para la tarea que debemos realizar, una vez hecho esto, se desarrolló un bloque de filtrado de la señal para reducir los efectos del ruido que acompaña la señal de voz ingresada. Al diseñar el filtro, se deberá considerar el hecho que la elección de las frecuencias de corte y paso, condicionarán al sistema en la discriminación de voces espectralmente parecidas, es decir que se obtuvieron netamente señales con información discriminando, tonalidades de voz, acentos, resonancias y teniendo en cuenta que la frecuencia de vos de una mujer, es diferente a la de un hombre.

Posteriormente se desarrolló un bloque de Matriz de Voz vs Tiempo la cual desarrolló la correcta segmentación de la señal y el entrenamiento del sistema a través de una retroalimentación. Dentro de este bloque se realizó el tratamiento de las señales de información como un proceso de las mismas, para la correspondiente manipulación de la información, como también de las librerías propias del software, para con esto ir desarrollando el sistema de conversión audio a texto y mediante instrucciones, ir modificarlo, mejorarlo y de esta manera lograr un sistema robusto con palabras acertadas, y una comunicación fluida y entendible

Este bloque, contiene a su vez a dos sub-bloques muy importantes los cuales son; el bloque de *procesamiento de señales*, y el bloque *memorización o base de datos*.

El primer sub-bloque a desarrollarse, fue el de ***procesamiento de señales***. Del cual depende en gran medida la exactitud del proceso de reconocimiento, ya que algunos sonidos que pueden captarse, correspondientes a ruido de fondo, podrían eventualmente confundirse con la voz; por ejemplo, el espectro de la respiración, o a su vez, podría confundir información con ruido de fondo; por ejemplo el espectro correspondientes a las palabras que inician con “ s ” o “ f ”. En el reconocimiento de señales de voz, se hizo necesario determinar con adecuada precisión los puntos de inicio y final de cada palabra en una matriz de vos vs tiempo, es decir, se debe diferenciar las partes de señal que llevan información de voz de aquellas que no llevan voz.

Una vez dicho esto se realizó el segundo sub-bloque en donde se creó una base de datos de modo diccionario, en el cual estarán almacenadas todas las palabras del sistema, pudiendo aumentar más palabras en caso de requerirlo.

Finalmente, se representó el proceso en una interface gráfica (maquina-humano), de una manera amigable y de fácil uso, cumpliendo así nuestros objetivos y sobre todo, contribuyendo con la sociedad con discapacidad auditiva.

Segundo Escenario.- En el caso de la figura 2, la primera operación que se desarrolló es cargar un archivo de video de un formato especifico al sistema, y se hizo el tratamiento de la señal de audio del video de la misma manera que en el bloque de Matriz de Voz vs Tiempo explicado anteriormente.

Una vez que se hizo el tratamiento de esta señal con objeto de extraer la información acústica relevante, el sistema nos generara un archivo de texto con los subtítulos del video.

Posteriormente se desarrollara un bloque de sincronización el cual fue el encargado de sincronizar los tiempos del audio del video con los subtítulos generados por el archivo de texto, haciendo un montaje de los subtítulos en el video.

Finalmente el presente proyecto concluyo con la implementación del sistema sobre un escenario de conferencia, o a su vez de reproducción de videos con un formato específico.

Así también se realizaron las respectivas pruebas de manera tal que se pueda medir su desempeño y rendimiento para así crear un sistema lo suficientemente ágil, confiable y robusto para incluirlo a las TICs para el beneficio de las personas discapacitadas**. fuente:** (Platero, 2006)

1.4. OBJETIVOS

1.4.1. GENERALES

- Diseñar, implementar y analizar el desempeño de un sistema de conversión de audio a texto en tiempo real para personas con discapacidad auditiva.

1.4.2. ESPECIFICOS

- Realizar un estudio de investigación sobre las diferentes técnicas de conversión audio a texto sobre una plataforma de software confiable para aplicaciones de video conferencias o reproducción de audio o video en tiempo real.
- Educar el sistema de conversión audio a texto mediante la utilización de diccionarios, de tal forma que este responda de una manera ágil y robusta ante las palabras su locutor.
- Diseñar un filtro para reducir los efectos del ruido que acompaña la señal de voz ingresada y haciendo una adecuada discriminación de las voces.
- Diseñar de matriz de voz vs tiempo para la correcta segmentación de la señal, evitando así obtener resultados erróneos en el análisis de las señales de voz.
- Diseñar una base de datos que sirva como diccionario para almacenar y agregar nuevas palabras.
- Diseñar un bloque de PDS (procesamiento digital de señales) para hacer un correcto tratamiento de la información de las señales de voz.
- Implementar y validar el sistema de conversión audio a texto sobre un escenario de videoconferencia o reproducción de video en formato específico con topología punto a punto sobre la plataforma determinada.

CAPÍTULO II
FUNDAMENTO TEÓRICO Y CONCEPTUAL

2.1. DISCAPACIDAD AUDITIVA EN EL ECUADOR

Las estadísticas en el año 1996 señalan que existen en promedio 213.000 personas con sordera en el Ecuador, estadísticas más actuales, establecidas por FUNCORAT (Fundación contra el ruido contaminante), señala que el 20% de la población ecuatoriana tiene problemas auditivos, esto significa que alrededor de 2.890.000 personas tienen algún grado de sordera.

Las personas sordas profundas, cuentan con el lenguaje de señas para su comunicación natural, y las personas con sordera parcial pueden acceder a ciertas ayudas técnicas con las cuales podrán comunicarse de una manera correcta y sin problemas. En ambos casos, las necesidades de comunicarse son diferentes y deben ser atendidas desde su especificidad. **fuente:** (Fundación Vivir la Sordera, 2010)

Situación de los niños sordos en Ecuador.- En el medio escolar a los niños con pérdida auditiva, no detectada, se les acusa de problemas conductuales y se cree que oír es cuestión de voluntad, a tal punto que la palabra “sordo” se usa como insulto, por lo tanto, la falta de detección de la sordera perjudica la autoestima, el aprendizaje y el desarrollo integral de las personas sordas.

En general, al igual que el resto, los niños con cierto grado de sordera, no tienen problemas intelectuales para adquirir los conocimientos del lenguaje, sin embargo, son las barreras audibles las que limitan su entorno, dificultando acceder a la educación, comunicarse libremente, informarse y participar. Una lengua plena para un buen entendimiento es la de viso-escrita que les permite entender la información transmitida, comunicarse plenamente y desarrollar su capacidad intelectual.

El 95% de niños con sordera en el ecuador, son descendientes de padres oyentes, la sordera al ser paulatina e invisible, puede convertirse en una experiencia traumática. Los padres con hijos sordos tratan de comunicarse con ellos desde el español en su forma oral y emplean métodos poco ortodoxos, tales como hablarles en tono de grito o en velocidades descendidas, no consiguen los resultados esperados. **fuente:** (Fundación Vivir la Sordera, 2010)

Actualmente en el Ecuador únicamente en 11 de sus 24 provincias, registran escuelas para niños con sordera. Sin embargo, se ha constatado que existen niños que crecen, sin educación, sufren maltratos y se encuentran aislados completamente.

Por otro lado, la mayoría de los profesores de las escuelas para niños con sordera profunda, tienen dificultades para comunicarse con sus alumnos porque no dominan la lengua de señas e incluso la desvalorizan. Solo cuando el niño sordo domina la lengua de señas, puede desarrollar su capacidad intelectual e incluso puede aprender una segunda lengua escrita. Actualmente, los adolescentes sordos terminan el bachillerato escribiendo y leyendo únicamente palabras sueltas y frases simples. **fuente:** (Fundación Vivir la Sordera, 2010)

Los adultos sordos en Ecuador.- Las personas con un grado de sordera crítica incluidas laboralmente, en muchos casos reciben constante maltrato por parte de sus compañeros, debido a la desinformación que obtienen a causa de su sordera, y así, la falta de consideración de sus necesidades específicas.

La inclusión tanto laboral como educativa demanda de un proceso que si no es llevado a cabo, resulta perjudicial para las personas con discapacidad porque la inclusión no significa ubicación física como objetos sino como sujetos.

En Ecuador la televisión pública y privada no ofrece acceso a la información a las personas sordas por la falta de subtítulos y/o lengua de señas. La producción audiovisual ecuatoriana carece de subtítulos en español excluyendo de este modo el acceso a la recreación de las personas sordas.

En las ventanillas de atención para personas con discapacidad, tanto en las empresas públicas como privadas, quienes atienden desconocen cómo relacionarse con una persona sorda, se limitan a gritar y enojarse. Las personas sordas tienen que enfrentarse con este mal trato todo el tiempo en los servicios de salud, educación, recreación, etc.

La discapacidad auditiva es la segunda de mayor incidencia en Ecuador con alta prevalencia a nivel nacional, por ello el Programa de Detección Temprana y Escolar de Discapacidad Auditiva invierte alrededor de 400.000 dólares en acciones para reducir y prevenir esta problemática en la población infantil.

Cada cirugía auditiva, está valorada en 16.000 dólares. Consiste en colocar en el cráneo, en la zona del oído, un implante que devuelve progresivamente la

capacidad de escuchar. Después de 120 días se coloca un aparato externo para completar el sistema osteointegrado con el que la persona recupera el 90% de su audición. La organización mundial de la salud sostiene que la producción actual de audífonos satisface menos del 10% de la necesidad mundial. **Fuente:** (Fundacion Vivir la Sordera, 2010)

2.2. LAS GRANDES APORTACIONES DE LAS TIC'S

Las Tecnologías de la Información y la Comunicación (TIC), como concepto general viene a referirse a la utilización de múltiples medios tecnológicos o informáticos para almacenar, procesar y difundir todo tipo de información, visual, digital o de otro tipo, con diferentes finalidades, como gestionar, organizar y sobre todo coordinador las diversas actividades laborales que a diario desempeñamos millones de personas.

Su incidencia e impacto conllevan a la adquisición, producción, almacenamiento, tratamiento, comunicación, registro y presentación de informaciones, en forma de voz, imágenes y datos contenidos en señales de naturaleza acústica, óptica o electromagnética. Las TICs incluyen la electrónica como tecnología base que soporta el desarrollo de las telecomunicaciones, la informática y el audiovisual. **fuente:** (Solar Perez, 2008)

Las TICs desde la ingeniería.- El conocimiento práctico es una característica distintiva de los ingenieros. Este impacto que ha generado las TIC en el mundo de la ingeniería ha ayudado a crear, planear, implantar, administrar, evaluar y supervisar sistemas de información, integrándolos a las organizaciones con el objeto de mejorar su competitividad.

El impacto de las TIC ha proporcionado a los ingenieros desarrollar, organizar, mantener y actualizar cualquier sistema de información, además de ayudar a las empresas a innovar tecnológicamente su forma de transmitir y controlar su información de manera óptima, confiable y precisa. Involucra tanto habilidades científicas y técnicas.

No obstante las TIC indudablemente han tenido un profundo impacto en todos los ámbitos, generando el desarrollo de diversas áreas del conocimiento de la ingeniería tales como: las telecomunicaciones, la electrónica, los sistemas informaticos, la Educación, Salud, Energía, Biotecnología, Transporte , entre otras. **Fuente:** (Jaramillo, 2010)

Las principales aportaciones de las TIC a las actividades humanas se concretan en una seria de funciones que nos facilitan la realización de nuestro trabajo.

Desde un plano de la ingeniería, como diseñadores podemos actuar dentro de tres mundos, cuya interacción queda ilustrada en la figura 3 y cuyas ventajas son:

- Facil acceso a una imensa fuente de información
- Proceso rápido y fiables de todo tipo de datos
- Canales de comunicación inmediata (on /off)
- Capacidad de almacenamiento
- Automatizacion de trabajos
- Interactividad
- Digitalizacion de toda información

Figura 3. *Las TIC en el plano de la ingeniería, nos permite actuar en tres mundos como diseñadores*

Impacto e incidencia de las TICs en la sociedad.- La tecnología es un fenómeno social, y como tal, está determinada por la cultura.

Cabe destacar que el uso de las TIC tiene un importante potencial para apoyar las acciones orientadas a contribuir en la satisfacción de las diferentes demandas sociales. Las TIC son incuestionables forman parte de la cultura tecnológica que nos rodea y con la que debemos convivir, además amplían nuestras capacidades físicas y mentales, y las posibilidades de cada vez un mayor desarrollo social.

Sus principales aportaciones a las actividades humanas se concretan en una serie de funciones que nos facilitan la realización de nuestros trabajos y a menudo también la comunicación con otras personas. Esto es precisamente lo que nos ofrecen las TIC y se lo puede observar de mejor manera, en la figura3. **Fuente:** (Jaramillo, 2010)

Figura 4.*Las TIC configuran la sociedad de la información, es una característica y factor de cambio de nuestra sociedad actual*

Las TIC en las discapacidades.- Existen minusvalías de varios tipos: visuales, auditivas, motoras, cognitivas, etc. Para estos tipos de personas, las TIC juegan un papel fundamental, no sólo facilitan su complicado día a día, sino que además se genera tecnología para facilitar el uso de las tecnologías.

Hoy en día ya existen dispositivos que, con bastante menos parafernalia y un menor sensacionalismo, resultan enormemente prácticos, tales como: teclados ampliados, ratones para mover con la boca, la cabeza o el mentón, pulsadores para sistemas de barrido (menús), marcadores por voz para teléfonos, comunicadores que mediante la pulsación de pictogramas, permiten reproducir mensajes pregrabados etc. La aparición y masificación de las interfaces táctiles en pantallas, teléfonos y tablets, ya han supuesto en sí mismas un enorme avance en la accesibilidad a los diversos dispositivos tecnológicos. Las interfaces vocales y las posibilidades que ofrecen, tanto para recibir instrucciones de forma oral como para la lectura de lo que aparece en pantalla con los conversores texto-voz, permiten el uso a colectivos con minusvalías motoras, auditivas e incluso visuales.

Es por este motivo por el que la Fundación “Vodafone” Realiza y promueve proyectos sociales y de innovación para la mejora de la calidad de vida de los discapacitados y a las personas mayores. Esta fundación, puso en marcha un plan de trabajo del que resulta una serie de conclusiones que pudieran ser tomadas en cuenta a la hora de facilitar la integración social y laboral a las personas con discapacidad. **fuente:** (Jaramillo, 2010)

En conclusión, las TIC nos proporcionan grandes ventajas tales como:

- Mayor comunicación.
- Mejor administración y distribución del Conocimiento.
- Mayores Fuentes de Conocimiento y Oportunidades de Investigación
- Aprendizaje Colectivo
- Desarrollo de Habilidades Adicionales
- Crecimiento como Persona
- Mejor Gestión Institucional y Servicio
- Trascender las barreras del tiempo y el espacio
- Facilitan la interacción e inclusión social.
- Mejora la motivación de las personas con discapacidad y su calidad de vida.
- Con la interrelación de tres sentidos, visual, táctil y auditiva, se logra la adquisición de conocimientos significativos.

2.3. NUEVAS TECNOLOGÍAS APLICADAS A LA DISCAPACIDAD

El objetivo principal de este tema es ver la situación actual de la tecnología en el apoyo a las personas con discapacidad.

El ordenador colabora como un instrumento que reduce las dificultades que se presentan en el desarrollo personal y eleva la autoestima del usuario en la búsqueda de dos objetivos:

- Ser un instrumento pedagógico (reeducación y refuerzo) y de rehabilitación para conseguir que una persona con deficiencia alcance un nivel físico mental y/o social óptimo.

- Equiparar oportunidades al estimular y facilitar la participación de las personas con discapacidad en todos los niveles de la vida social, cultural y económica.

Antes es importante saber diferenciar y por lo tanto se hacen las siguientes definiciones previas. Según la OMS (Organización Mundial de la Salud) se considera:

Deficiencia.- Pérdida o anormalidad de una estructura o función psicológica, fisiológica o anatómica.

Discapacidad.- Restricción o ausencia (causada por una deficiencia) de la capacidad de realizar una actividad en la forma o dentro del margen que se considera normal por el ser humano.

Minusvalía.- Situación desventajosa para una individuo determinado, consecuencia de una deficiencia o de una discapacidad, que le limita e impide desempeñar un rol que es normal en su caso (en función de la edad, sexo y factores sociales y culturales).

Podemos agrupar las tecnologías de ayuda al discapacitado en cinco grandes grupos:

- Sistemas Alternativos y Aumentativos de Acceso a la Información.
- Sistemas de Acceso.
- Sistemas Alternativos y Aumentativos de comunicación.
- Sistemas de Movilidad.
- Sistemas de Control de Entornos.

Fuente: (Sánchez Montoya, 1998)

2.3.1. SISTEMAS ALTERNATIVOS Y AUMENTATIVOS DE ACCESO A LA INFORMACIÓN.

Estos sistemas engloban las ayudas para personas con discapacidad visual y/o auditiva, modifican la señal, aumentándola o cambiando su modalidad para poder ser percibido por ellos

Cuando existe resto sensorial, la opción es el aumento de la señal que el contexto envía al sujeto. Los sistemas aumentativos se dirigen hacia la población con déficits visuales y auditivos.

Entre los primeros, disponemos de un numeroso arsenal de sistemas de aumento para mejorar la visión cuando es baja: Las lupas, tele-lupas o sistemas informáticos que amplían a información de la pantalla.

En el caso de las personas con déficits auditivos aparte de los crecientes avances con los implantes cocleares, también conocidos como los sistemas FM, que permiten a la persona hipoacúsica captar el mensaje de un emisor independientemente de la distancia.

Los sistemas alternativos son medios que permiten, a quienes presentan la imposibilidad de alcanzar la información mediante una determinada modalidad sensorial, cambiar la naturaleza de la misma de modo que pueda aprehenderse mediante una modalidad que la persona mantiene funcional. En ello se basa la sub-titulación de imágenes para personas con déficits auditivos. **Fuente:** (Koon & Vega, 2000)

Algunos ejemplos de estos sistemas son:

Tecnologías del Habla: El reconocimiento de voz y la conversión texto-voz

Sistemas multimedia interactivos: procesan, almacenan y transmiten de forma integrada imágenes, voz, texto y datos. Ofrecen la posibilidad de actuar sobre los contenidos de los mismos, surgiendo así la interactividad. Las personas con discapacidad se benefician de la existencia de servicios y aplicaciones multimedia que les permitirán, mediante las necesarias adaptaciones, perfeccionar el acceso multimodal en igualdad de condiciones.

Comunicaciones de avanzada: La conexión exclusiva a través de las computadoras va a dar paso a una amplia gama de dispositivos de acceso a una red de alta velocidad. Hay una acelerada tendencia a incluir la videotelefonía, teléfonos de texto, fax y otros.

Audífono: es un producto sanitario electrónico que amplifica y cambia el sonido para permitir una mejor comunicación. Los audífonos reciben el sonido a través de un micrófono, que luego convierte las ondas sonoras en señales eléctricas. El amplificador aumenta el volumen de las señales y luego envía el sonido al oído a través de un altavoz. **Fuente:** (Koon & Vega, 2000)

2.3.2. SISTEMAS DE ACCESO.

Son Interfaces adaptativos que permiten a las personas con discapacidad física o sensorial utilizar una computadora.

La accesibilidad o accesibilidad universal es el grado en el que todas las personas pueden utilizar un objeto, visitar un lugar o acceder a un servicio, independientemente de sus capacidades técnicas, cognitivas o físicas. Es indispensable e imprescindible, ya que se trata de una condición necesaria para la participación de todas las personas independientemente de las posibles limitaciones funcionales que puedan tener.

Para promover la accesibilidad se hace uso de ciertas facilidades que ayudan a salvar los obstáculos o barreras de accesibilidad del entorno, consiguiendo que estas personas, realicen la misma acción que pudiera llevar a cabo una persona sin ningún tipo de discapacidad. Estas facilidades son llamadas ayudas técnicas. Entre éstas se encuentran el alfabeto Braille, la lengua de señas, las sillas de ruedas, las señales auditivas de los semáforos, OCR (reconocimiento óptico de caracteres), Pizarras electrónicas, Pantallas táctiles, Bastones digitales, etc.

Considerando *"Convención sobre los Derechos de las Personas con Discapacidad"*, la accesibilidad es un derecho que implica la real posibilidad de una persona de ingresar, transitar y permanecer en un lugar, de manera segura, confortable y autónoma. Ello implica que las barreras de entorno físico deben ser suprimidas. **Fuente:** (Koon & Vega, 2000)

En informática, la accesibilidad incluye ayudas como las tipografías de alto contraste o gran tamaño, magnificadores de pantalla, lectores y revisores de pantalla, programas de reconocimiento de voz, teclados adaptados, y otros dispositivos apuntadores y de entrada de información.

"*El poder de la web reside en su universalidad. El acceso para todo el mundo, a pesar de la discapacidad, es un aspecto esencial*". Tim Berners-Lee (Director e inventor de la Red Mundial Web)

La accesibilidad aplicada al contenido de Internet se denomina accesibilidad web. En la Web, el W3C ha desarrollado directrices o pautas específicas para permitir y asegurar este tipo de accesibilidad.

Según **Egea 2007:21** podemos englobar en tres grandes líneas las principales dificultades que encuentran las personas con discapacidad en su relación con las tecnologías digitales:

- Posibilidad de manejo o acceso a los elementos físicos que nos proporcionan las tecnologías digitales.
- Posibilidad de efectuar una interacción con las interfaces presentes en cada medio.
- Posibilidad de acceder a los contenidos que nos presentan los terminales, que cada vez van siendo mayores y más complejos.

El Centro Nacional de Tecnologías de la Accesibilidad (CENTAC), es una organización cuyo objetivo es promover el desarrollo de las tecnologías de accesibilidad en todos los ámbitos posibles, con el fin último de facilitar la integración social, la igualdad en el acceso a las Tecnologías de la Sociedad de la Información, y en conclusión, de mejorar la vida de todas las personas con discapacidad, dependientes y la de sus familias. **Fuente:** (Koon & Vega, 2000)

2.3.3. SISTEMAS ALTERNATIVOS Y AUMENTATIVOS DE COMUNICACIÓN.

La comunicación aumentativa y alternativa incluye todas las modalidades de comunicación (aparte del habla) utilizadas para expresar pensamientos, necesidades, deseos e ideas. Todos utilizamos este tipo de comunicación cuando usamos gestos, expresiones faciales, símbolos, ilustraciones o escritura.

Las personas con graves disfunciones de habla o de lenguaje dependen de la comunicación aumentativa y alternativa para complementar el habla residual o como una alternativa al habla no funcional. Los instrumentos especiales de comunicación aumentativa, como los aparatos electrónicos y los tableros de comunicación con dibujos y símbolos, ayudan a las personas a expresarse y comunicarse. Esto puede mejorar la interacción social, el aprovechamiento escolar y los sentimientos de autoestima.

Las personas que utilizan los sistemas aumentativos y alternativos de comunicación no deben de dejar de hablar si son capaces de hacerlo. Estos instrumentos están encaminados a ayudarlos a comunicarse con mayor eficacia. Existen muchos tipos de sistemas aumentativos y alternativos de comunicación, y por lo general están clasificados en una de dos categorías: con ayuda o sin ayuda.

El mejor sistema de comunicación para una persona dada puede incluir una combinación de modalidades con ayuda y sin ayuda para adaptarse a diversas situaciones. **Fuente:** (Koon & Vega, 2000)

Los sistemas de comunicación sin ayuda.- no proporcionan salida de voz ni equipo electrónico. El interlocutor tiene que estar presente para que estos sistemas puedan funcionar (no pueden ser usados por teléfono ni para comunicarse con alguien que esté en otra habitación). Algunos ejemplos de este tipo de comunicación incluyen:

- gestos
- lenguaje corporal
- lenguaje por señales
- tableros de comunicación

Los tableros de comunicación pueden mostrar palabras, letras, números, ilustraciones o símbolos especiales.

Los sistemas de comunicación con ayuda.- son aparatos electrónicos que pueden contar o no con algún tipo de salida de voz. Los instrumentos que brindan salida de voz se denominan comunicadores con salida de voz. Estos aparatos pueden mostrar letras, palabras y frases, o una variedad de símbolos que permiten al usuario construir mensajes. Los mensajes pueden ser comunicados mediante voz electrónica o pueden aparecer impresos en una pantalla o en una cinta de papel. Muchos de estos sistemas pueden también conectarse a una computadora para obtener comunicación por escrito. Algunos de ellos pueden ser programados para producir distintos idiomas. **Fuente:** (Koon & Vega, 2000)

2.3.4. SISTEMAS DE MOVILIDAD.

Son aquellos relacionados a la movilidad personal y las barreras arquitectónicas. Ejemplos son: brazos o soportes articulados, conmutadores adosados a sillas de ruedas, emuladores de ratones, varillas, micro-robots, etc. Dos casos de búsqueda alternativas específicas son:

- Chip para parapléjicos.
- “Phantom” o dedo-robot para ciegos.

2.3.5. SISTEMAS DE CONTROL DE ENTORNOS.

Son aquellos que, con fines comunicativos que permiten la manipulación de dispositivos que ayudan a controlar un entorno. Se distinguen dos tipos:

Control Ambiental.- Interfaces que permiten a las personas con discapacidad motora, el poder controlar dispositivos de uso doméstico. Existe gran número de interfaces que permiten a las personas con discapacidad motora, el poder controlar dispositivos de uso doméstico. Ej. Las llamadas "casas inteligentes", cuyo software facilita:

- Conectar/desconectar timbres,
- Abrir/cerrar puertas,
- Comunicarse por teléfono,
- Control de luces/aire acondicionado/TV u otros dispositivos.

Realidad Virtual.- En este contexto, cabe esperar su desarrollo como una Tecnología Adaptativa mediante nuevos dispositivos de entrada y salida avanzados, tales como guantes sensitivos, dispositivos de seguimiento de movimientos oculares, posicionadores de 3D, etc. con alentadoras y crecientes posibilidades en el diseño de sistemas de asistencia a personas con discapacidad. **fuente:** (Koon & Vega, 2000)

2.4. PROYECTOS TECNOLÓGICOS DESARROLLADOS PARA DISCAPACITADOS

El desarrollo de proyectos es muy importante para ayudar a todas las áreas que requieren de mejoras a sus procesos o servicios, así mismo de las personas que requieren tecnologías que les ayuden a adaptarse a los cambios y facilitar su día a día.

Las personas con discapacidades físicas necesitan requieren y necesitas que constantemente se gestionen nuevos proyectos de desarrollo tecnológico para ayudarles a tener una mejor calidad de vida adaptada a los constantes cambios en las sociedades modernas.

Muchos países están incentivando la innovación y la creatividad con el objetivo principal de intentar proveer información objetiva y contextuada de la situación de la tecnología en apoyo a las personas con discapacidad y con ello estimular al desarrollo de nuevos proyectos tecnológicos que ofrezcan mejores servicios a sus usuarios.

EATCO, grupo de investigación de proyectos tecnológicos.- Las tecnologías de la información y el uso, cada vez mayor, de las redes telemáticas está generando nuevas formas de enseñar, de trabajar y, como no,

de comunicar, que han dado lugar a nuevos conceptos que implican nuevos estilos de vida, como son, la teleformación, el teletrabajo, la teleasistencia, los sistemas multimedia o la domótica. **Fuente:** (Koon & Vega, 2000)

Con la teleformación se pretende que cualquier persona, a través de los ordenadores y las redes telemáticas, independientemente de su edad, situación económica, o incapacidad física, pueda realizar cualquier tipo de curso o estudios desde su casa, de forma individual o compartida con otros familiares o amigos. Las nuevas tecnologías están modificando el sentido y las opciones del aprendizaje.

EATCO potencia el estímulo investigador en ámbitos innovadores. Proyecta la transferencia de tecnología y conocimiento al ámbito nacional e internacional, y con especial énfasis a su entorno más cercano, actualiza y profundiza sobre el desarrollo y difusión de nuevos sistemas educativos, nuevos métodos de información y conocimiento, y usos tecnológicos para mejorar la calidad de vida. **Fuente:** (Koon & Vega, 2000)

2.4.1. A NIVEL DE SOFTWARE

Windows.- para la accesibilidad conllevan un coste adicional. Micosoft pone al alcance de las personas con discapacidad en su menú Panel de control/accesibilidad las siguientes opciones:

- Voz
- Lupa
- Teclado en pantalla
- Narrador y notificaciones visuales

Apple.- pone a nuestro alcance varias opciones de accesibilidad para personas con discapacidad sin coste adicional.

- Ampliación de pantalla
- Interfaz simplificada que favorece la exploración y el aprendizaje
- Teclas para el ratón
- Teclas lentas
- Pulsación fácil
- Braille en espejo
- Lector de pantalla mediante gestos
- Subtítulo de películas

Linux.- nos ofrece varios proyectos dedicados a mejorar la accesibilidad de personas con discapacidad.

Además de otros grupos investigadores e innovadores que ponen a disposición diferentes software: **fuente:** (Koon & Vega, 2000)

TC-Soft.- dentro de las autoayudas es un teclado en pantalla que permite mediante su uso por el ratón o cualquier otro tipo de dispositivo adaptado a minusvalías simular las funciones que se pueden hacer con el teclado convencional. El objetivo de escribir este texto mediante el TC-Soft es mandarlo a cualquier aplicación Windows, ya sea este correo electrónico, chat, word etc.

Figura 5. *Software TC-Soft .Proyecto desarrollado por el grupo EATCO*

TComunica.- ésta dentro de los sistemas aumentativos de comunicación que utilizan símbolos pictográficos que representan diferentes categorías gramaticales y que llevan un dibujo representativo del concepto. El objetivo es el aprendizaje de los símbolos, posteriormente de frases y por último las utilice para comunicarse con otras personas.

Figura 6*Software TC-comunica .Proyecto desarrollado por el grupo EATCO*

2.4.2. A NIVEL DE HARDWARE

A nivel de hardware, estos se pueden dividir en dos grupos:

Básico.- fundamentales para que ordenador funcione (torre, monitor, teclado, ratón, etc.).

Complementario.- complementos (impresora, escáner, usb, webcam etc.).

Los siguientes son unos ejemplos de los diferentes tipos de tecnología de apoyo a nivel de software: botones, teclados especiales, y control remotos que le permite a una persona con discapacidad a controlar cosas en su ambiente:
Fuente: (Koon & Vega, 2000)

Ratón bucal.- permite mover el cursor del ordenador al desplazar el mando del joystick en la dirección deseada. Está especialmente indicado para ser utilizado con la barbilla, boca, muñeca o mano. En el extremo lleva incorporado un pulsador para hacer el clic del ratón.

Figura 7. *Hardware Mouse Bucal .Proyecto desarrollado por el grupo EATCO*

Emulador de ratón por pulsadores.- consiste en un dispositivo que tiene las mismas funciones que el ratón bucal, pero el joystick se ha sustituido por direcciones ortogonales y diagonales las cuales se realizan mediante ocho entradas digitales. Además de las cuatro para realizar el clic, doble clic, clic permanente y cambio de velocidad.

Figura 8.*Hardware Mouse por Joystick y pulsadores .Proyecto desarrollado por el grupo EATCO*

Emulador de ratón por reconocimiento de voz.- puede realizar todas las funciones de un ratón convencional pero controlado por la voz. El dispositivo tras un entrenamiento de las palabras que se van a emplear queda preparado para mover el cursor mediante las palabras reconocidas.

Figura 9.*Software TC-comunica .Proyecto desarrollado por el grupo EATCO*

Teclado por pedales.- es un dispositivo que funciona como un teclado programable de tres botones con el que puede usarse conjuntamente con un teclado regular.

Figura 10. *Hardware Teclado por Pedales.*

Teclados para una sola mano.- tienen una distribución especial de las teclas que permiten su acceso con una amplitud de movimiento reducida.

Figura 11. *Hardware Teclado para una sola mano.*

CAPÍTULO III
MATERIALES Y METODOS

3.1. MATERIALES

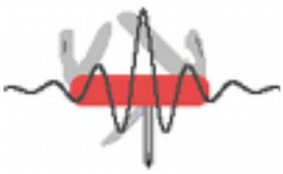
A lo largo del proyecto ha sido necesario emplear una serie de dispositivos físicos y aplicaciones informáticas con el fin de recopilar información, realizar tareas de análisis, diseño e implementación de código, llevar a cabo pruebas y documentar los distintos procesos, etc. A continuación se muestran los distintos medios utilizados de manera detallada y clasificados en función de su tipo.

Tabla 1.
Herramientas de Hardware

Herramienta	Característica
Ordenador Portátil  **Sony Vaio - Core i3**	**Implementación del nuevo sistema de conversión de audio a texto en tiempo real para personas con discapacidad auditiva**
 **Micrófono y Audifonos Ordenador**	**Grabación de audios**

Tabla 2.
Herramientas de Software

Herramienta	Característica
 **Linux Mint 16 Cinnamon DVD 64bit**	**Sistema Operativo: desarrollo del sistema con Sphinx**
 **Linux Ubuntu Desktop 11.04 DVD 64bits**	**Sistema Operativo: Grabación y depuración de audios (más de X horas de grabación)**
 **Microsoft Windows Professional 7**	**Sistema Operativo: Instalación de la Máquina virtual y desarrollo de la interface gráfica en Java**
 **CMU - Sphinx**	**Grupo de sistemas opensource de reconocimiento de voz: contiene las herramientas necesarias para la implementación del nuevo sistema.**
 **Oracle VM Virtual Box**	**Máquina Virtual: Contenedora de los Kernel Linux (Mind - Ubuntu)**

 **SoX - Sound eXchange**	**Procesamiento de Audio**
 **Editor y Grabador de Audio Audacity**	**Reproductor de Audio**
 **NetBeans Standard Package**	**Programación de Interface Grafica**
 **Java Platform (JDK)**	**Plataforma de Programación**
 **Microsoft Office Word 2010**	**Editor de Texto**
 **Prezi 2015 5.1.2**	**Editor de Diagramas**

3.1.1. SOFTWARE LINUX 170 250

Linux ha sido el sistema operativo elegido por gran cantidad de usuarios, empresas, corporaciones e incluso instituciones gubernamentales, debido a las enormes ventajas que presenta este software en comparación con sus mayores competidores actuales. **Fuente:** (Reyes, 2005)

Linux es un sistema operativo totalmente gratuito, es una ventana abierta por la que es posible huir hacia un mundo donde la verdadera informática puede ser disfrutada sin límites ni monopolios por los programadores, ofreciendo un sinfín de ventajas. Entre las cuales para nuestro caso en particular, se han considerado:

Velocidad y Seguridad.- Los problemas de seguridad, al igual que el tema de los virus, atacan con más frecuencia al sistema operativo Microsoft Windows, la plataforma Linux es más robusta en comparación, con lo cual hace más difícil que algún intruso pueda violar el sistema de seguridad.

La principal causa es que el SO Linux, viene con algunos “security holes” y es menos común que tenga problemas, sobre todo por su velocidad a la hora de resolver problemas.

Un caso práctico es el de “Phalax”, el primer troyano para Linux, un caso sonado que de verdad golpeo a Linux como una bofetada, pero el problema fue por un error humano, la mala administración, lo que causo que esto pasara a mayores. Normalmente en Linux, por su velocidad evolutiva, los parches de seguridad salen tan pronto como se alza la voz en cuanto a algún problema.

La línea de comandos.- Casi todas las funciones en Linux, se las puede realizar a través del terminal: instalaciones, configuraciones, solución de redes, montaje y desmontaje de discos etc. La poderosa línea de comandos puede hacer todo en un sistema operativo Linux, puede deshacer el mundo, o puede hacerlo más eficiente y veloz.

En la mayoría de las veces solo hace falta tres o cuatro líneas de comandos para solucionar cualquier cantidad de problema como; compatibilidad, librerías, etc. Otras veces la instalación entera se hace por comandos y mejor no hablar si la instalación no se completa correctamente.

Al instalar algunas librerías sólo se lo puede hacer por línea de comandos, e inclusive el software tampoco tiene interfaz gráfica para su instalación.

Servidores.- Al pensar que Windows tiene monopolizado el mercado de servidores, la dura realidad del siglo 21 es que Linux domina este mercado de manera abrumadora. Linux es mucho más eficiente a la hora de ser un servidor, puede, y va a ser un servidor de cualquier tipo, fácil de configurar, rápido y seguro.

Interoperabilidad.- La interoperabilidad se define como la capacidad que tiene un sistema, cuyas interfaces son totalmente conocidas, para funcionar con otros productos o sistemas existentes o futuros, sin restricción de acceso o de implementación.

Windows trabaja de manera excelentemente, incluso se podría decir que es bastante fácil. El problema suceder al momento de usar Windows con otros sistemas sin utilizar software de terceros, lo cual es algo prácticamente imposible.

Para Linux, esto no es un problema y por su parte, trabaja de maravilla en conjunto con otros sistemas como Mac, Windows, Solaris, Android y un gran etcétera. Por citar un ejemplo, Linux puede guardar o abrir documentos en cualquier formato.

Desarrollo.- El objetivo inicial de Linux, es propulsar el software de libre distribución junto con su código fuente para que pueda ser modificado por cualquier persona, dando rienda suelta a la creatividad. Este método también es aplicado a los programas que corren en el sistema, lo que hace que podamos encontrar muchísimos programas útiles totalmente gratuitos y con su código fuente.

Además Linux, tiene distribuciones que se actualizan cada 6 meses, y a veces lo hacen en grande, desde el núcleo hasta los íconos. Además Linux siempre se pregunta ¿qué quiere la comunidad? Todo está hecho con esa premisa.

Por lo general Microsoft Windows se actualiza muy lento y obliga a los usuarios y las compañías a evolucionar alrededor de él. Para la mayoría, Windows Vista fue fracaso, un paso para atrás para la compañía Microsoft, que tomó alrededor de 5 años en rectificarlo y mejorarlo. **Fuente:** (Reyes, 2005)

En resumen podemos decir que Linux sistema operativo totalmente libre de código abierto y por lo tanto se tiene la libertad de mejorarlo para adaptarlo a las necesidades. Más veloz, más fiable y seguro. Fuente:

En nuestro caso particular, y para cumplir con los objetivos del presente proyecto de tesis, trabajaremos con **"Sphinx"** desarrollado en la Universidad de Carnegie Mellon (Pensilvania Estados Unidos). Utiliza el SO Linux como plataforma base para dejar las herramientas necesarias para el reconocimiento de voz y posteriormente liberar nuevas versiones.

Sphinx es un Software de reconocimiento de voz, independiente, que proporciona las búsquedas en textos de manera rápida y relevante. Es del tipo open source (software libre) diseñado para desarrolladores que deja abierta a la creatividad para el desarrollo de nuevos proyectos. Ha sido diseñado para integrar información almacenada en bases de datos (SQL o XMLs) y obedecer de manera fácilmente y accesible por lenguajes de script (archivo de órdenes en texto plano). **Fuente:** (Reyes, 2005)

3.1.2. CMU SPHINX

Una de las tecnologías que más prometen en la revolución de la interfaz de usuario desde hace años es la del reconocimiento de voz. Los grandes de la informática han tratado de avanzar en un terreno que no obstante no acaba de cuajar entre el gran público, debido a que las soluciones no son lo eficientes que nos gustaría.

En Windows 7 existe un sistema integrado de reconocimiento de voz que funciona de forma bastante decente, mientras que aplicaciones como *´´Dragon Naturally Speaking´´ de Nuance* o *´´ViaVoice´´ de IBM* han sido las alternativas tradicionales. El desarrollo de *Nuance* es en la actualidad el más reputado, pero aun así sus prestaciones siguen sin convencer a la mayoría de los usuarios, que acaban manejando el PC con los tradicionales ratones y teclados.

En Slashdot (sitio de noticias orientado a la tecnología) se preguntaban recientemente si hay alguna alternativa Open Source en este segmento, y pues la hay, se llama **Sphinx** o **CMU Sphinx**, para ser precisos, y es un desarrollo de la Universidad de Carnegie-Mellon que se ofrece con licencia BSD (Berkeley Software Distribution) licencia de software libre permisiva otorgada principalmente para los sistemas. Esta licencia tiene menos restricciones en comparación con otras como la GPL estando muy cercana al dominio público.
Fuente: (Picajoso, 2010)

Sphinx abarca una serie de sistemas de software, inicio como sphinx 1, luego se produjeron las versiones 2, 3, 4 y Pocket Sphinx, todas tienen aplicaciones diferentes, aunque su función es la misma, el reconcimiendo del habla. **Fuente:** (CMU-Sphinx, 2014)

- Los decodificadores de voz vienen con modelos acústicos y aplicaciones de ejemplo.
- incluyen además el software para el entrenamiento de modelos acústicos (Retroalimentación).
- la compilación de un modelo de lenguaje (Cadenas de Marcoft).
- y un diccionario de pronunciación de dominio público (Ingles)
- análisis de WER (tasa de error) para mayor confiabilidad

3.1.2.1. VERSIONES SISTEMA DE SOFTWARE CMU-SPHINX

Sphinx.- es un sistema de habla continua y reconocimiento de habla, utiliza el Modelo oculto de Márkov (HMMs) y un lenguaje de modelado estadístico de n-gramas. Sphinx interpreta voz hablada en forma continua, reconocimiento de habla de vocabulario amplio.

Sphinx 2.- Es un sistema de reconocimiento de habla de alta resolución desarrollado originalmente en la *Universidad de Carnegie Mellon*, por el investigador *Xuedong Huang*, quien liberó su código como software libre con una licencia BSD.

Sphinx 2 se centra en el reconocimiento de voz en tiempo real, es adecuado para aplicaciones de lenguaje hablado e incorpora funciones tales como, *end-pointing, partial hypothesis generation, dynamic language model switching* entre otras, que son herramientas utilizadas en sistemas de diálogo y los sistemas de aprendizaje de idiomas. **Fuente:** (CMU-Sphinx, 2014)

Sphinx 2 utiliza una representación semi-continua para el modelado acústico, es decir que se utilizan un único conjunto de gaussianas para todos los modelos individuales como un vector de peso. Puede ser utilizado en los sistemas informáticos basados en PBX como por ejemplo, Asterisk.

Sphinx 3.- Es un cliente independiente que utiliza la representación frecuente HMMs (Hidden Markov Model) se ha utilizado principalmente para la alta precisión, no en tiempo real de reconocimiento.

Sin embargo la evolución reciente, en los algoritmos y hardware, han hecho que Sphinx 3 funcione "casi" en tiempo real, aunque todavía no es adecuado para las aplicaciones interactivas.

Sphinx 3 está en desarrollo y en colaboración con SphinxTrain (modelo acústico de entrenamiento) quien proporciona acceso a una serie de técnicas de modelado modernas como LDA / MLLT, MLLR y VTLN, para el entrenamiento de un modelo acústico que mejoran la precisión en el reconocimiento. **Fuente:** (CMU-Sphinx, 2014)

Sphinx 4.- Cliente multiplataforma escrito en Java, es una completa re-escritura de la máquina de Sphinx, con el objetivo de proporcionar un marco más flexible para la investigación en reconocimiento de voz, está escrito íntegramente en lenguaje de programación Java. *Sun Microsystems* (Compañía Informática de Sistemas Operativos) apoya el desarrollo de la Sphinx 4 y contribuyó experiencia en software de ingeniería para el proyecto.

Al estar escrito en java puede ser utilizado en una gran diversidad de sistemas operativos y hardware.

Los objetivos actuales de desarrollo incluyen:

- El desarrollo de un nuevo entrenador (modelo acústico)
- Implementación de adaptación de la persona que está hablando (por ejemplo, el método de entrenamiento MLLR(Máximum Likelihood Linear Regression))
- Mejora de la gestión de configuración
- La creación de una interfaz de usuario gráfica basada en el diseño gráfico de sistemas.

Pocket Sphinx.- es una versión de la Sphinx que se pueden utilizar en sistemas embebidos (por ejemplo, basado en un procesador ARM). PocketSphinx está siendo evaluado para desarrollar e incorporar características como la aritmética de coma fija y algoritmos eficientes para el cálculo de modelos mezclados. Puede ser utilizado en muchos equipos portátiles y también en teléfonos móviles. **Fuente:** (CMU-Sphinx, 2014)

3.1.2.2. HERRAMIENTAS CMU-SPHINX

En el presente proyecto escrito de tesis, vamos a describir las aplicaciones de CMU-Sphinx kit de herramientas.

Tales aplicaciones bien podrían ser aplicadas para el control de voz, de escritorios, varios dispositivos automotrices y/o casas inteligentes. Otras aplicaciones posibles son; la transcripción del habla, los subtítulos, traducción del habla, la búsqueda por voz y el aprendizaje de idiomas. Si se desea crear una de ellas, el conjunto de herramientas de CMUSphinx es su elección.

Esta vez, en el presente proyecto, CMU-Sphinx kit de herramientas permitió el desarrollo de la aplicación denominada “implementación de un sistema de conversión de audio a texto en tiempo real para personas con discapacidad auditiva.” Es muy importante recordar que CMU-Sphinx, está dirigido a los desarrolladores, para aplicar la tecnología de voz en sus aplicaciones, mas no para las prospecciones de reconocimiento de voz**. Fuente:** (CMU-Sphinx, 2014)

A continuación se presentan las herramientas utilizadas para el desarrollo de la implementación del presente proyecto de tesis: **Fuente:** (CMU-Sphinx, 2014)

Sphinxbase.- Es un conjunto de bibliotecas comunes utilizadas para varios proyectos de CMU Sphinx. Sphinxbase, es una biblioteca de apoyo requerido por Pocketsphinx, por lo tanto su instalación es imprescindible.

Pocketsphinx.- Es una biblioteca escrita en C para el reconocimiento de voz basada en SphinxBase. depende de otra llamada SphinxBase, por este motivo es que se necesita instalar las dos. Pocketsphinx es un reconocedor de voz que puede ser utilizado en sistemas embebidos.

Sphinxtrain.- Es una suite de herramientas que se utilizan para llevar a cabo la capacitación del modelo acústico. Estas herramientas son las encargadas del tratamiento y entrenamiento de los audios. Contiene recetas claves para su entrenamiento.

Sphinx4-5prealpha.- Es una versión completa de CMU-Sphinx escrita en Java. Se trata de un ajustable reconocedor, modificable, que proporciona alta precisión y alta velocidad en su desempeño.

Cmuclmtk.- Es una suite de herramientas que llevan a cabo el entrenamiento del modelo del lenguaje. Estas herramientas son las encargadas del tratamiento y ajuste de la gramática del lenguaje de oraciones para su respectivo entrenamiento.

Sox.- Es una herramienta de audio que no pertenece al kit de herramientas CMU-Sphinx, sin embargo su instalación resulta imprescindible. Sox es

una libre multiplataforma editor de audio digital , licenciado bajo la Licencia Pública , está escrito en la norma C , y tiene una interfaz de línea de comandos.

Los componentes principales del sistema CMU-Sphinx son el módulo **Sphinxtrain**, utilizado para entrenamiento de los modelos acústicos, y el módulo *Sphinx decoder*, utilizado para el reconocimiento de la voz. Que en nuestro caso particular será utilizado **Sphinx4**.

A su vez, **Cmuclmtk** no deja de ser muy importante en el desarrollo del proyecto, ya que es el encargado del entrenamiento del modelo de lenguaje. Este es el encargado del tratamiento de la gramática y sintaxis de las palabras dentro de las oraciones. **Fuente:** (CMU-Sphinx, 2014)

3.1.3. SOFTWARE JAVA

Surgió en 1991, cuando un grupo de ingenieros de Sun Microsystems trataron de diseñar un nuevo lenguaje de programación destinado a electrodomésticos. La reducida potencia de cálculo y memoria de los electrodomésticos llevó a desarrollar un lenguaje sencillo capaz de generar código de tamaño reducido.

Java fue diseñado para soportar múltiples arquitecturas, debido a que desarrollaron un código independiente del tipo de electrodoméstico, el cual se ejecutaba sobre una máquina virtual denominada Java Virtual Machine (JVM). Era la JVM quien interpretaba el código convirtiéndolo a código particular dependiendo de la CPU utilizada. Esto permitía lo que luego se ha convertido en el principal lema de lenguaje: "Write Once, Run Everywhere".

Las aplicaciones de Java son muy amplias, el lenguaje se utiliza en una gran variedad de dispositivos móviles, como teléfonos y pequeños electrodomésticos. Dentro del ámbito de Internet, Java permite desarrollar pequeñas aplicaciones (conocidas con el nombre de applets) que se incrustan en el código HTML de una página, para su directa ejecución desde un navegador. Su la instalación es liviana y sencilla. **Fuente:** (Aparicio Paniagua, 2012)

Esta vez Java tendrá un papel muy importante dentro del desarrollo del presente proyecto. Una vez liberada la nueva versión "*implementación de un sistema de conversión de audio a texto en tiempo real para personas con discapacidad auditiva*" con CMU-Sphinx kit de herramientas, Java será la encargada de compilar y correr dicha versión y reflejar mediante una interface gráfica el arduo trabajo que se ha llevado al cabo del desarrollo del proyecto

como tal. Mediante las utilizaciones de las herramientas de Java, será posible la interacción maquina-humano. **Fuente:** (Aparicio Paniagua, 2012)

Las principales características pueden resumirse en:

- Java es un lenguaje de programación de propósito general, concurrente y orientado a objetos.
- Su sintaxis es similar a C y C ++, pero omite muchas de las características que hacen que C y C ++ sea complejo, confuso e inseguro.
- Independiente de la plataforma de ejecución.
- Lenguaje compilado e interpretado.
- Lenguaje orientado a internet.
- JVM se implementa en código dependiente de la plataforma.
- Fuertemente tipado.
- Gestión de memoria por la JVM: para ello utiliza el operador new para la creación de objetos y la liberación de recursos es gestionada por el recolector de basura.

3.1.3.1. RAZONES PARA ESCOGER JAVA POR SOBRE OTROS LENGUAJES

Es orientado a objetos.- la programación orientada a objetos resulta muy conveniente para la mayoría de las aplicaciones. Entre las ventajas más evidentes que ofrece se encuentra un gran control sobre el código y una mejor organización, dado que basta con escribir una vez los métodos y las propiedades de un objeto, independientemente de la cantidad de veces que se utilicen.

Es muy flexible.- Java es un lenguaje especialmente preparado para la reutilización del código; permite a sus usuarios tomar un programa que hayan desarrollado tiempo atrás y actualizarlo con mucha facilidad, sea que necesiten agregar funciones o adaptarlo a un nuevo entorno.

Funciona en cualquier plataforma.- a diferencia de los programas que requieren de versiones específicas para cada sistema operativo (tales como Windows o Mac), las aplicaciones desarrolladas en Java funcionan en cualquier entorno, dado que no es el sistema quien las ejecuta, sino la máquina virtual (conocida como Java Virtual Machine o JVM).

Su uso no acarrea inversiones económicas.- programar en Java es absolutamente gratis; no es necesario adquirir ninguna licencia, sino simplemente descargar el kit de desarrollo (Java Development Kit o JDK) y dar riendas sueltas a la imaginación.

Es de fuente abierta.- Java ofrece el código de casi todas sus librerías nativas para que los desarrolladores puedan conocerlas y estudiarlas en profundidad, o bien ampliar su funcionalidad, beneficiándose a ellos mismos y a los demás.

Es un lenguaje expandible.- continuando con el punto anterior, cada programador tiene la libertad de revisar y mejorar el código nativo de Java, y su trabajo puede convertirse en la solución a los problemas de muchas personas en todo el mundo. Infinidad de desarrolladores han aprovechado esta virtud del lenguaje y continúan haciéndolo. **Fuente:** (Aparicio Paniagua, 2012)

3.1.3.2. MÁQUINA VIRTUAL DE JAVA (JVM)

La máquina virtual de Java (JVM) es la piedra angular de la plataforma Java. Es el componente de la tecnología responsable del hardware y de que sea independiente del sistema operativo, así como del pequeño tamaño de su código compilado, y de proteger a los usuarios de programas maliciosos.

La JVM es una máquina de computación abstracta. Al igual que una máquina de computación real, tiene un conjunto de instrucciones y manipula varias áreas de memoria diferentes en tiempo de ejecución. La máquina virtual de Java no entiende el lenguaje de programación Java, sólo de un formato binario especial, el formato de los ficheros ".class".

Un fichero de clase contiene instrucciones de la máquina virtual de Java y una tabla de símbolos, así como otra información complementaria. Por razones de seguridad, la máquina virtual Java impone fuertes restricciones sintácticas y estructurales en el código en un archivo ".class". **Fuente:** (Aparicio Paniagua, 2012)

3.1.3.3. KIT DE DESARROLLO Y ENTORNO DE EJECUCIÓN (JDK)

El Java Development Kit (JDK) es un entorno de desarrollo para aplicaciones Java. JDK no es un entorno integrado de desarrollo en el sentido de aglutinar todas las herramientas bajo la misma interfaz gráfica si no que se ejecutan en modo de comando. Los comandos principales son: **Fuente:** (Aparicio Paniagua, 2012)

- javac: compila el código y genera los archivos class.
- java: intérprete de la JVM. Ejecuta los bytecodes de los ficheros class.
- javadoc: genera documentación
- AppletViewer: visaliza páginas HTML.
- jdb: depura aplicaciones Java.

El Java Runtime Environment (JRE) es el software necesario para ejecutar cualquier aplicación desarrollada para la plataforma Java, y está incluido dentro del JDK. Puede también obtenerse como un paquete independiente, y puede considerarse como el entorno necesario para ejecutar una aplicación Java, mientras que un desarrollador debe además contar con otras facilidades que ofrece el JDK. **Fuente:** (Aparicio Paniagua, 2012)

3.1.3.4. DISTRIBUCIONES DE JAVA

Existen cuatro plataformas del lenguaje de programación Java:

- Java Platform, Standard Edition (Java SE)
- Java Platform, Enterprise Edition (Java EE)
- Java Platform, Micro Edition (Java ME)
- JavaFX

Cada plataforma Java proporciona una máquina virtual y una interfaz de programación de aplicaciones, esto permite que las aplicaciones escritas en una plataforma puedan ejecutarse en cualquier sistema compatible con todas las ventajas del lenguaje de programación Java: independencia de la plataforma, potencia, estabilidad, facilidad de desarrollo y seguridad. **Fuente:** (Aparicio Paniagua, 2012)

Java SE.- Es la base para las otras distribuciones Java. Su principal componente es el API, el cual proporciona la funcionalidad básica del lenguaje de programación Java. Define todo lo necesario, desde los tipos básicos hasta clases que se utilizan para la creación de redes, seguridad, acceso a base de datos, interfaz gráfica de usuario (GUI) desarrollo y análisis XML.

Además de la API, la plataforma Java SE dispone de una máquina virtual, herramientas de desarrollo, tecnologías de implementación, y otras bibliotecas de clases y herramientas usadas en aplicaciones de tecnología Java.

Java EE.- La plataforma Java EE está construida sobre la plataforma Java SE. La plataforma Java EE proporciona una API y un entorno de ejecución para

desarrollar y ejecutar aplicaciones de red a gran escala, a varios niveles y escalables.

Java ME.- La plataforma Java ME proporciona un API y una máquina virtual de pequeñas dimensiones para la ejecución de aplicaciones en dispositivos pequeños como los móviles. Este API es un subconjunto de la API de Java SE, junto con las bibliotecas especiales de clases útiles para el desarrollo de aplicación en dispositivos pequeños. Las aplicaciones JavaME a menudo son clientes de servicios de la plataforma Java EE.

Java FX.- JavaFX es una plataforma para la creación de aplicaciones dinámicas de Internet usando una interfaz de usuario ligera. Las aplicaciones de JavaFX utilizan gráficos de aceleración hardware y motores de comunicación para aprovechar los clientes de mayor rendimiento y además aporta un moderno look-and-feel y APIs de alto nivel para la conexión de fuentes de datos en red. Aplicaciones JavaFX pueden ser clientes de servicios de la plataforma Java EE.

3.1.3.5. JAVA ECLIPSE

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Eclipse es ahora desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios. **Fuente:** (Aparicio Paniagua, 2012)

Eclipse fue certificado como un EHR (electronic health record) completa a partir de julio de 2011, ahora está buscando la certificación de la Etapa 2 en el año 2015, y es utilizado por más de 7.000 prácticas de quiropráctica, Terapia Física, Médicos y de todo Estados Unidos. Prácticas van desde un solo practicante, prácticas multi-sitio multidisciplinarios con decenas de proveedores y más de 200 equipos en red que facturan millones en ingresos mensuales.

Eclipse es un programa informático compuesto por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar lo que se llama "Aplicaciones de Cliente Enriquecido", que trata de crear un archivo ejecutable independiente con una interfaz gráfica formada por varios controles para el usuario, opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. **Fuente:** (Aparicio Paniagua, 2012)

Arquitectura Eclipse.- La base para Eclipse es la Plataforma de cliente enriquecido (del Inglés Rich Client Platform RCP). Los siguientes componentes constituyen la plataforma de cliente enriquecido:

- Plataforma principal - inicio de Eclipse, ejecución de plugins
- OSGi - una plataforma para bundling estándar.
- El Standard Widget Toolkit (SWT) - Un widget toolkit portable.
- JFace - manejo de archivos, manejo de texto, editores de texto
- El Workbench de Eclipse - vistas, editores, perspectivas, asistentes

Eclipse es un entorno de desarrollo integrado (IDE) para Java y otros lenguajes de programación como C, C ++, PHP y Ruby, etc. Entorno de desarrollo proporcionado por Eclipse incluye las herramientas de desarrollo Eclipse Java (JDT) para Java, Eclipse CDT para C / C++, y Eclipse PDT para PHP, entre otros.

El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (plug-in) para proporcionar toda su funcionalidad al frente de la plataforma de *cliente enriquecido*, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software. Adicionalmente a permitirle a Eclipse extenderse usando otros lenguajes de programación como son C/C++ y Python, permite a Eclipse trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y Sistema de gestión de base de datos.

La arquitectura plugin permite escribir cualquier extensión deseada en el ambiente, como sería Gestión de la configuración. Se provee soporte para Java y CVS (“Concurrent Versioning System”, es una aplicación informática que implementa un sistema de control de versiones: mantiene el registro de todo el trabajo y los cambios en los ficheros) en el SDK de Eclipse. Y no tiene por qué ser usado únicamente con estos lenguajes, ya que soporta otros lenguajes de programación. **Fuente:** (Aparicio Paniagua, 2012)

La definición que da el proyecto Eclipse acerca de su software es: "una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular".

En cuanto a las aplicaciones clientes, Eclipse provee al programador con frameworks muy ricos para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software, aplicaciones web, etc. Por ejemplo, GEF (Graphic Editing Framework - Framework para la edición gráfica) es un plugin de Eclipse para el desarrollo de editores visuales que pueden ir desde procesadores de texto, hasta editores de diagramas UML, interfaces gráficas para el usuario (GUI), etc.

Dado que los editores realizados con GEF "viven" dentro de Eclipse, además de poder ser usados conjuntamente con otros plugins, hacen uso de su interfaz gráfica personalizable y profesional. **Fuente:** (Aparicio Paniagua, 2012)

El SDK de Eclipse incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código. Mediante diversos plugins estas herramientas están también disponibles para otros lenguajes como C/C++ (Eclipse CDT) y en la medida de lo posible para lenguajes de script no tipados como PHP o Javascript. El IDE también hace uso de un espacio de trabajo, en este caso un grupo de metadatos en un espacio para archivos plano, permitiendo modificaciones externas a los archivos en tanto se refresque el espacio de trabajo correspondiente. **Fuente:** (Aparicio Paniagua, 2012)

3.1.3.6. JAVA NETBEANS

NetBeans es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo el patrocinador principal de los proyectos. Actualmente Sun Microsystems es administrado por Oracle Corporation. **Fuente:** (Aparicio Paniagua, 2012)

Al día de hoy hay disponibles dos productos: el NetBeans IDE y NetBeans Platform.

NetBeans IDE es un entorno de desarrollo, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso. **Fuente:** (Aparicio Paniagua, 2012)

También está disponible NetBeans Platform, una base modular y extensible usada como estructura de integración para crear grandes aplicaciones de escritorio. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones.

Ambos productos son de código abierto y gratuitos para uso tanto comercial como no comercial. El código fuente está disponible para su reutilización de acuerdo con la Common Development and Distribution License (CDDL) y GNU General Public License (GPL) . **Fuente:** (Aparicio Paniagua, 2012)

Plataforma de NerBeans.- Framework esta simplificando el desarrollo de aplicaciones para escritorio Java Swing. El paquete de NetBeans IDE para Java SE contiene lo que se necesita para empezar a desarrollar plugins y aplicaciones basadas en la plataforma NetBeans; no se requiere un SDK adicional.

Las aplicaciones pueden instalar módulos dinámicamente. Algunas aplicaciones pueden incluir un modulo de actualización para permitir a los usuarios descargar Actualizaciones de firma digital y nuevas características directamente dentro de la aplicación en ejecución. Reinstalando una actualización o una nueva versión, no obligando a los usuarios a descargar toda la aplicación de nuevo. **Fuente:** (Aparicio Paniagua, 2012)

La plataforma ofrece servicios reusables comunes para las aplicaciones de escritorio, permitiendo a los desarrolladores centrarse en la lógica de sus aplicaciones. Algunas de las características de la aplicación son:

- Gestión de la interfaz de usuario (menús y barras de herramientas)
- Gestión de configuración de usuario
- Gestión de almacenamiento (guardar o cargar algún tipo de dato)
- Gestión de ventana
- Marco Asistente (soporta diálogos para a paso)
- Librería visual de Netbeans
- Herramientas de desarrollo integrado

3.1.4. OTROS SOFTWARE NECESARIO

Para el desarrollo del presente proyecto, se utilizaron además algunas herramientas de software extra. Su utilización fue de gran importancia para el desarrollo y su descripción no puede pasar por alto. No se puede olvidar que el desarrollo del nuevo sistema de conversión audio a texto, fue desarrollado en CMU-Sphinx el cual utiliza Linux como plataforma base, posteriormente es liberado como una nueva versión, y presentado en una interface máquina - humano libre y de multiplataforma modelado en Java.

3.1.4.1. SOX

SoX fue creado en julio de 1991 por Lance Norskog y enviado al grupo Usenet como “Aural eXchange: Sound sample translator”. Con la segunda versión (en noviembre de ese mismo año) fue rebautizado como Sound Exchange. Norskog siguió manteniendo y lanzando SoX via Usenet, FTP, y luego en la web hasta principios de 1995, cuando SoX estaba en su versión 11 (gamma). En mayo de 1996, Chris Bagwell comenzó a mantener y publicar las versiones actualizadas de SoX, comenzando con la versión sox-11gamma-cb. En septiembre de 2000, Bagwell registró el proyecto en SourceForge con el nombre de "sox". El registro fue anunciado el 4 de septiembre de 2000 y SoX 12.17 salió el 7 de septiembre de 2000.

A lo largo de su historia, SoX ha tenido muchos colaboradores, Guido van Rossum, más conocido como el creador del lenguaje de programación Python, fue un factor importante en los primeros días de SoX.

SoX es una utilidad que funciona por línea de comando y es multiplataforma (Windows, Linux, MacOS X). SoX lee y escribe archivos de audio en los formatos más populares y, opcionalmente, puede aplicar efectos en ellos. Puede combinar varias fuentes de entrada de audio, sintetizar y, en muchos sistemas, actuar como un reproductor de audio de propósito general o un grabador de audio multipista. También tiene capacidad para dividir la entrada en varios archivos de salida. **Fuente:** (Aparicio Paniagua, 2012)

Toda la funcionalidad de SoX está disponible utilizando el comando “sox”. Para simplificar la reproducción y grabación de audio, si SoX es invocado como reproductor, el archivo de salida se ajusta automáticamente para ser el dispositivo de sonido por defecto, y si se invoca como grabador, el dispositivo de sonido predeterminado se utiliza como fuente de entrada. Además, SoX

posee comandos para consultar información de la cabecera del archivo del archivo de audio.

SoX es una herramienta de procesamiento de audio mediante línea de comandos, especialmente adecuado para hacer rápida y sencilla la edición y el procesamiento batch. **Fuente:** (Aparicio Paniagua, 2012)

Las principales características de SoX son:

- Conversión de archivos de audio.
- Edición de archivos de audio: concatenación, borrado, repetición, volumen, normalización, etc.
- Cambio de los atributos de audio.
- Adición de efectos de audio.
- Multiplataforma.
- Posee multitud de opciones de manipulación de sonido.
- Lectura y escritura de una gran variedad de formatos de audio como: AU, WAV, AIFF, Ogg Vorbis, FLAC, RAW, etc.
- Grabación de audio.
- Reproducción de audio, incluso mediante URL.
- Procesamiento mediante retardos, cambios de fase, compresión, filtros de paso y paso alto, etc.
- Análisis estadísticos.
- Fusión de archivos.

3.1.4.2. AUDACITY

Fue creado en otoño de 1999 por Dominic Mazzoni y Roger Dannenberg en la universidad de Carnegie Mellon. Tras lo cual fue publicado en SourceForge.net como software libre en mayo de 2000. En mayo de 2008, Audacity fue incorporado a la lista de los 100 mejores productos del año según los lectores y editores de la revista.PC World. **Fuente:** (Aparicio Paniagua, 2012)

Audacity es una aplicación informática multiplataforma libre, que se puede usar para grabación y edición de audio, distribuido bajo la licencia GPL. Es el editor de audio más difundido en los sistemas GNU/Linux.

Al ser un software de grabación multipista, Audacity cumple los parámetros necesarios para poder ser utilizado en un estudio de grabación de presupuestos más bajos, también conocidos como "estudios caseros". No tiene las mismas

funciones ni potencia que softwares de uso comercial (Pro Tools, Cubase, FL Studio, etc.) pero es una herramienta bastante útil para personas que empiezan a experimentar el mundo de la grabación multipista. Cuenta con herramientas de edición de audio como copiar, cortar, pegar, junto con varios tipos de plugins y varios efectos básicos bastante útiles en una edición. **Fuente:** (Aparicio Paniagua, 2012)

Características

- Grabación de audio en tiempo real.
- Edición archivos de audio tipo Ogg Vorbis, MP3, WAV, AIFF, AU , LOF y WMP.
- Conversión entre formatos de tipo audio.
- Importación de archivos de formato MIDI, RAW y MP3.
- Edición de pistas múltiples.
- Agregar efectos al sonido (eco, inversión, tono, etc).
- Posibilidad de usar plug-ins para aumentar su funcionalidad.

3.2. METODOS

Un sistema de reconocimiento de voz es una herramienta computacional que realiza el procesamiento de la señal de voz emitida por el ser humano y reconoce la información contenida en ésta. Estos datos puede o bien convertirlo en texto o bien emitir órdenes para que actúen sobre un proceso. En un sistema de reconocimiento de voz intervienen diversas disciplinas, tales como: la fisiología, la acústica, el procesamiento de señales, la inteligencia artificial y la ciencia de la computación.

Cada palabra es única y tiene su propia “huella digital” y por lo tanta aparecen probabilísticamente según su mención. La figura 12 explica de manera gráfica, el reconocimiento de las palabras. La diferencia entre dictado y control radica principalmente:

Dictado → (Palabras infinitas)
Control → (Palabras finitas)

Figura 12.*Coeficientes espectrales para el reconocimiento de palabras.*

Si bien el proceso tiene como función obtener la secuencia de palabras asociada a la frase en lenguaje natural de entrada, las personas, al hablar, normalmente lo hacen de forma continua, es decir, sin pausas entre algunas palabras. A menudo, también existen problemas gramaticales, que incluyen elementos propios del habla espontánea como son las interjecciones, falsos comienzos, repeticiones, etc. Esto supone que el proceso de reconocimiento de habla, presente problemas de cobertura léxica y sintáctica. Por estas razones, la tarea del sistema de reconocimiento de habla es difícil, y además, es costosa, tanto en memoria como en cálculo.

El objetivo de esta sección es explicar la metodología de desarrollo seguida a lo largo del proyecto y la especificación de la funcionalidad a través de los casos de uso y los requisitos. Además, se comentará el diseño del sistema mediante su arquitectura. **fuente:** (CMU-Sphinx, 2014)

3.2.1. CONCEPTOS BÁSICOS

El habla es un fenómeno muy complejo, la gente rara vez entienden cómo ésta es producida y percibida. La percepción común es a menudo que; el discurso se construye con palabras, y cada palabra se compone de fonemas.

La realidad es, por desgracia muy diferente, el habla es un proceso dinámico y sin partes claramente diferenciadas. Siempre es útil obtener una edición de sonido, mirar en la grabación de voz y escuchar a la misma.

La figura 13, muestra un claro ejemplo de un discurso. Nótese como un discurso está compuesto por palabras, y que a su vez, cada palabra tiene su espectro en función del tiempo. **fuente:** (CMU-Sphinx, 2014)

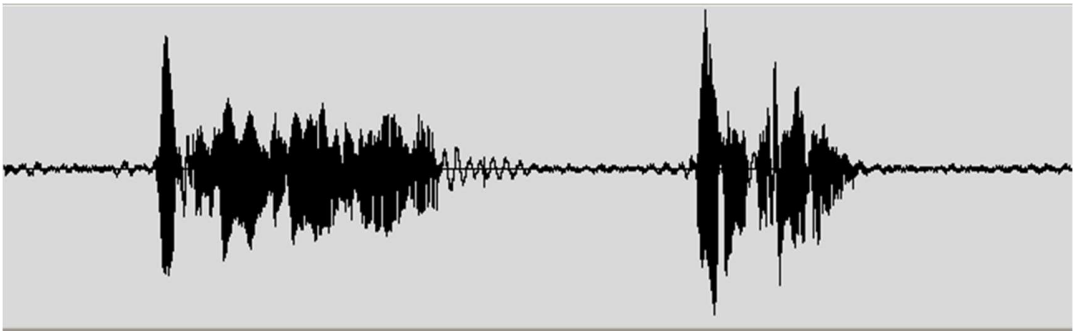


Figura 13. *Ejemplo de la grabación de un discurso en un editor de audio.*

Todas las descripciones modernas de un discurso, son hasta cierto grado de probabilidad. Esto significa que no hay límites entre las unidades, o entre las palabras.

Los discursos de traducción a texto, y otras aplicaciones semejantes de discurso, nunca son 100% correctas. Esta idea es bastante discutida entre los desarrolladores de software, que por lo general trabajan con sistemas deterministas y crean un montón de temas específicos sólo para buscar soluciones a la tecnología de voz. **fuente:** (CMU-Sphinx, 2014)

3.2.1.1. ESTRUCTURA DEL DISCURSO

En la práctica actual, la estructura de discurso se entiende como sigue:

El habla es una corriente continua de audio, donde se mezclan los estados más estables con los estados cambiantes de una forma dinámica. En esta secuencia de estados, se puede definir más o menos similares clases de sonidos, o fonemas. Se cree que todas las palabras que se construyen de fonemas, son entendidas, pero esto no es cierto. Las propiedades acústicas de una forma de onda correspondiente a un fonema, pueden variar en gran medida dependiendo de muchos factores como; contexto fonológico, volumen de la voz, sexo, edad, el estilo de discurso, acento, y así sucesivamente. **fuente:** (CMU-Sphinx, 2014)

Es de mucha importancia entender y tener en claro lo siguiente:

La existencia de la llamada “coarticulación” (influencia mutua entre los sonidos articulados del habla) hace que los fonemas suenen muy diferente de su representación canónica, y ya que las transiciones entre las palabras son más del modo informativas, los desarrolladores a menudo hablan de la

utilización de “difonos” (representa el sonido que va desde la mitad de un fonema hasta la mitad del fonema siguiente.), incluso a veces los desarrolladores también hablan de unidades sub-fonéticas (diferentes sub estados de un fonema).

Dentro de un fonema se pueden encontrar fácilmente tres o más regiones de diferente naturaleza. La primera parte del fonema depende de su fonema anterior, la parte media es estable, y la siguiente parte depende del fonema subsiguiente. Es por eso que se habla de tres estados en un fonema seleccionados para el reconocimiento de voz.

A veces, los fonemas son considerados en su contexto. Existen trifonos o incluso quinphones, pero hay que tener en cuenta la diferencia que existe entre; fonemas y difonos, porque se corresponden con el mismo rango en una forma de onda, y sólo son diferenciadas por su nombre. Esta herramienta a utilizar es preferiblemente denominada senone. La dependencia de un senone en contexto podría ser más compleja que contexto a la izquierda y la derecha. Y es sin duda, una función bastante compleja definida por un árbol de decisión.

Es importante saber que todos los fonemas se construyen de unidades parciales de palabra, o sub-palabras, como son las sílabas. A veces, las sílabas se definen también como "entidades de reducción-estable". A menudo cuando el habla se vuelve rápida, los fonemas cambian, pero las sílabas siguen siendo los mismos. Además, las sílabas están relacionadas con el contorno de entonación.

Hay otras maneras de construir sub-palabras, basadas en lenguajes de morfología-rica o basadas fonéticamente. Las sub-palabras se utilizan a menudo en el reconocimiento de voz de vocabulario abierto.

Las sup-palabras forman palabras, a su vez, las palabras son importantes en el reconocimiento de voz porque restringen combinaciones de fonemas significativamente. Por ejemplo: Si hay 40 fonemas y una palabra promedio tiene 7 fonemas, debe haber 40 ^ 7 palabras. Por suerte, incluso una persona muy educada rara vez utiliza más de 20.000 palabras en su práctica, lo que hace que el reconocimiento, sea de tal manera, más factible. **fuente:** (CMU-Sphinx, 2014)

Las palabras y otros sonidos no lingüísticos, que llamamos generalmente rellenos (respiración, um, eh, tos), forman expresiones, que son trozos de audio

separados entre pausas. No es necesario un juego de oraciones, son conceptos más semánticos.

3.2.1.2. PROCESO DE RECONOCIMIENTO

La forma más común para reconocer el habla es el siguiente y puede observarse de mejor forma con un ejemplo práctico en la figura 14

Se toma la forma de onda acústica, posteriormente se divide dicha onda en declaraciones por silencios, y finalmente se trata de reconocer lo que se ha dicho en cada enunciado. Para ello se debe tomar todas las posibles combinaciones de palabras escritas y tratar de coincidir con el audio. Se elige la mejor combinación coincidente. **fuente:** (CMU-Sphinx, 2014)

En primer lugar hay que tener muy en cuenta que esto se trata de un concepto de características. Dado que el número de parámetros es grande, este fue de alguna forma optimizado. La cantidad del habla calculada, generalmente es divida en marcos o ventanas. Para cada ventana, la longitud típicamente es de 10 milisegundos, donde extraemos 39 números de muestras que representan el habla. Esto es llamado **vector de características**. La manera de genera estos números es un tema de investigación, pero en caso simple Es un derivado del espectro.

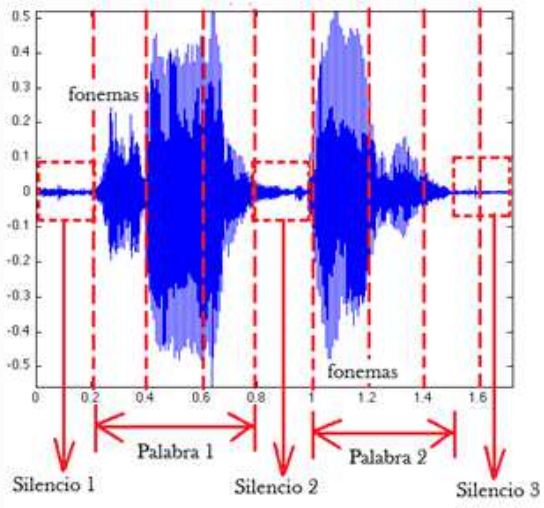


Figura 14. *Espectro de la vos en función de amplitud vs tiempo. La vos es dividida en ventanas y se logra reconoce dos palabras*

En segundo lugar, se trata de un concepto de modelo. El modelo describe un objeto matemático que reúne los atributos comunes de la palabra hablada. El modelo, utiliza una mezcla gaussiana de sus tres estados. Para decirlo más simple, es un vector de características más probable. En la práctica, para el modelo acústico, se utiliza los ya mencionados **senone.**

Desde el concepto de modelo, fue una ventaja bastante grande que éste se ajusta a la práctica, funcionando de manera excelente para resolver problemas internos, cómo es el modelo de adaptación a las nuevas condiciones.

El modelo de expresión se llama **"modelo oculto de Markov" o (HMM)**, es un modelo genérico que se puede describir como un canal de comunicación dentro de una caja negra. En este proceso el modelo HMM se describe como una secuencia de estados que cambian entre sí con cierta probabilidad. Este modelo está destinado a describir cualquier proceso secuencial como es el habla, porque ha demostrado ser muy práctico para su decodificación.

En tercer lugar, es un proceso de comparación de sí mismo. Se necesitaría un gran tiempo más que suficiente para comparar todos los vectores de características con todos los modelos, pero la búsqueda fue optimizada utilizando varios trucos, lo cual reduce el tiempo de procesamiento. En cualquiera de los puntos, se mantiene las mejores variantes más probables, y son extendidas al paso del tiempo, es decir que se realiza la mejor producción de variantes para el siguiente fotograma, en un tiempo óptimo considerable. **fuente:** (CMU-Sphinx, 2014)

3.2.1.3. OTROS CONCEPTOS UTILIZADOS

Base de datos de voz.- Bases de datos de voz se utilizó para entrenar, afinar y poner a prueba los sistemas de decodificación. Es un conjunto de grabaciones típicas de la base de datos. Se desarrolló el sistema de diálogos, que no es otra cosa que diálogos grabados de los usuarios. Para el sistema de dictado, únicamente lee las grabaciones e interpreta.

Bases de datos de texto.- Es el textos o muestras recogidas para la formación del modelo de lenguaje. Por lo general, las bases de datos de textos se recogen en forma de texto. El problema con la colección de muestras, es poner los documentos (PDF, páginas web, etc…) en forma de texto hablado. Es decir que fue necesario quitar las etiquetas, encabezados y expandir abreviaturas para ampliar el número a su forma hablada**.** **fuente:** (CMU-Sphinx, 2014)

3.2.1.4. OPTIMIZACIÓN SPHINX

Cuando se está desarrollando el reconocimiento de voz, el tema más complejo es hacer que la búsqueda sea precisa (considerar el mayor número de variantes como sea posible para que coincida) y lo suficientemente rápida para no procese durante mucho tiempo. Obviamente van a existir pequeñas fallas de precisión con el modelo, al tratar de coincidir con el discurso, debido a que los modelos no son perfectos. fuente: (CMU-Sphinx, 2014)

Por lo general, el sistema se ha probado en una base de datos de prueba, que está destinado a representar la tarea de destino correctamente.

Las siguientes características son ya optimizadas por Sphinx:

Tasa de errores.- Sphinx permite calificar la exactitud del reconocimiento y mejorarlo según las horas de grabación obtenidas, lo que hace que su tasa de error pueda bajar cada vez más. Esto permite mayor calidad y confiabilidad

Se tiene inicialmente el texto original y el texto de reconocimiento de longitud de N palabras. A partir de estos, la tasa de error de palabras es:

WER(%) = (I + D + S) / N → se mide generalmente en porcentaje .

Donde:

Las (I) se insertan
Las (D) se borraron
Las (S) son sustituidas.

Modelos.- Sphinx cuenta con las herramientas adheridas como cajas negras para crear el modelo de lenguaje ML y el modelo acústico MA. Además entre sus herramientas cuenta con el modelo de reconocimiento HMM.

Precisión.- Es casi lo mismo que la tasa de error de palabras, pero no cuenta con las denominadas inserciones.

Precisión = (N - D - S) / N

La utilización de precisión, es en realidad una mala idea para la mayoría de tareas que se van a realizar, ya que las inserciones también son importantes en los resultados finales. Pero para algunas otras tareas, la precisión es una medida razonable de la actuación del decodificador. **fuente:** (CMU-Sphinx, 2014)

Velocidad.- Sphinx permite dar una respuesta semejante al tiempo real, incluso su respuesta puede ser ajustada con parámetros de QoS y calidad del sonido.

Supongamos que el archivo de audio fue de 2 horas, y la decodificación tomó 6 horas, a continuación, la velocidad se cuenta como: 3xRT.

Curvas ROC.- Cuando hablamos de tareas de detección, hay falsas alarmas aciertos / fallos, es cuando entonces se utilizan las curvas ROC, es un gráfico que describe el número de falsas alarmas vs número de aciertos, y trata de encontrar el punto óptimo en el que el número de falsas alarmas es pequeño y el número de aciertos coincide con el 100%.

Hay otras propiedades de esta, menos importantes y que a menudo no son tomados en cuenta, pero pueden resultar ser importantes para muchas otras aplicaciones prácticas. Su principal tarea es la construcción de una medida y sistemáticamente aplicarlo durante el desarrollo del sistema. Su segunda tarea es recoger la base de datos de prueba y probar cómo realizar su solicitud.

3.2.2. DESCRIPCIÓN GENERAL

La idea principal es llegar a la explicación del todo, por lo que iremos uniendo conceptos como un rompecabezas, explicando piezas por pieza para llegar a una explicación entendible del total del proyecto, cuyo objetivo es el reconocimiento de vos en tiempo real. Una vez comprendido el estado del arte y teniendo en claro los conceptos del procesamiento para el reconocimiento de vos, a continuación aremos una breve descripción general de cómo funciona el sistema: **fuente:** (CMU-Sphinx, 2014)

El módulo Sphinxtrain se encargó de la obtención de los parámetros de los modelos de las unidades de sonido que se obtienen por medio de muestras de la señal de voz; a estas muestras se les denomina *bases de entrenamiento*.

El módulo de entrenamiento también requiere que se le indique qué unidades de sonido se desean entrenar y al menos la secuencia en la cual ocurre cada señal de voz dentro de la base de entrenamiento. Esta información se le proporciona al módulo Sphinxtrain a través de un archivo de transcripción, en el que se indican tanto la secuencia de palabras como los sonidos que no forman parte del discurso, como, pausas, disfluencias, etcétera, en el orden exacto como ocurren dentro de la señal de voz, seguidos por una etiqueta que puede ser utilizada para asociar esta secuencia con su correspondiente señal de voz.

Enseguida, el módulo de entrenamiento busca en un diccionario de lenguaje, que relaciona cada palabra con una secuencia de unidades de sonido, con el fin de obtener la secuencia de unidades de sonido que están asociadas con cada señal de voz. Asimismo se ayuda de un diccionario de relleno para establecer la correspondencia entre los sonidos que no forman parte del discurso.

Por su parte, el módulo Sphinx decoder, dadas las entradas adecuadas, realiza la tarea de reconocimiento. Las entradas necesarias son: los modelos acústicos entrenados, un archivo de índices del modelo, el modelo del lenguaje, un diccionario del lenguaje, un diccionario de relleno y un conjunto de señales acústicas que se desea reconocer. A los datos que se desea reconocer se les denomina conjunto de prueba.

Con los modelos acústicos entrenados, el módulo de entrenamiento generará archivos de índices del modelo. Un archivo de índices del modelo simplemente contiene identificadores para cada estado de cada modelo oculto de Markov, los cuales son usados por los módulos de entrenamiento y de reconocimiento para tener acceso al conjunto correcto de parámetros para aquellos estados de los modelos ocultos de Markov. Concualquier conjunto de modelos acústicos, se debe utilizar su archivo de índices de modelo para realizar la decodificación. **fuente:** (CMU-Sphinx, 2014)

3.2.3. MÉTODO CADENAS OCULTAS DE MARCOFF

Un modelo oculto de Márkov es un modelo estadístico en el que se asume que el sistema a modelar es un proceso de parámetros desconocidos. El objetivo es determinar los parámetros desconocidos u ocultos de dicha cadena a partir de los parámetros observables.

El proceso comienza en un estado inicial, específicamente diseñado para ello, y cada uno de los estados va a tener asociado un conjunto de probabilidades sobre un grupo de símbolos salientes. Por cada una de las ejecuciones, se va a elegir una transición hacia un estado nuevo, y se va a generar un símbolo de salida relacionado con dicho estado. La elección de las ejecuciones de cada transición y símbolo se va realizar en función de probabilidades, y por tanto va a ser una elección aleatoria. La figura 15 muestra claramente cómo este modelo funciona.

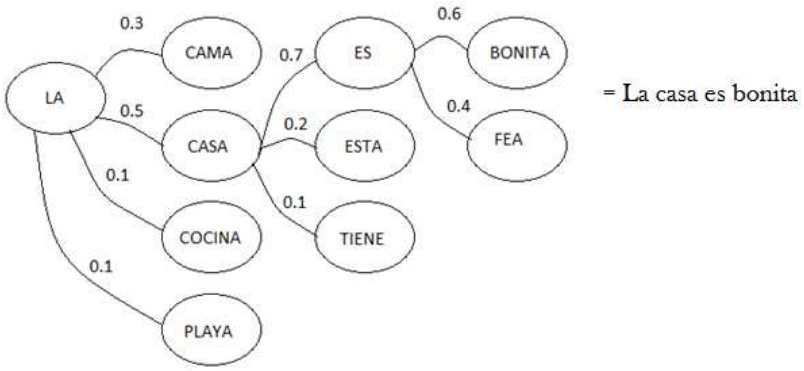


Figura 15.*Cadenas Ocultas de Marcoff. Elección aleatoria probabilística para el reconocimiento de palabras.*

Cada estado va a indicar cuáles son aquellos sonidos más probables para cada segmento del habla, mientras que las transiciones van a ser restricciones temporales para cada uno de esos sonidos, indicando cuáles son sus secuencias de apariciones. Esta técnica proporciona los mejores resultados hasta la fecha para el reconocimiento de habla aislada como continua.

La característica principal de los modelos de Márkov es que no se conoce el conjunto de estados por los que el proceso ha realizado el recorrido hasta llegar al conjunto de símbolos obtenidos en la salida, y este es el motivo fundamental por el que se le conoce como Modelo oculto de Márkov.

Las redes neuronales (NN), por su parte, son estructuras de procesamiento paralelo de información, formadas por numerosos nodos simples conectados entre sí mediante pesos y agrupados en diferentes capas, entre las que se deben distinguir la capa de entrada y la capa de salida. Debido a su naturaleza intrínsecamente no lineal, a su capacidad de clasificación, y sobre todo a la capacidad que tienen para aprender una determinada tarea a partir de pares observación-objetivo sin hacer suposición alguna sobre el modelo subyacente, se han convertido en una de las herramientas más atractivas para la solución del problema del reconocimiento de habla. Sin embargo, al principio el estudio de las redes neuronales fue abandonado, debido a que no se podía llevar a cabo su entrenamiento con algoritmos que fuesen eficientes. En la actualidad ha quedado perfectamente demostrado, que los modelos basados en las redes neuronales cuentan con una gran potencia desde el punto de vista computacional y se han conseguido resultados comparables a los obtenidos con otros métodos ya clásicos como los HMM. Sin embargo, las NN presentan diferentes problemas o inconvenientes como el desconocimiento a priori de la

estructura de capas y el número de nodos necesarios, un tiempo a veces muy elevado para su entrenamiento y la posibilidad de que se mantenga en los mínimos locales de las funciones de coste usadas durante el entrenamiento de la red. Esto implica que se haga necesario combinar dichos sistemas con técnicas basadas en programación dinámica y en modelos ocultos de Márkov. **fuente:** (Aparicio Paniagua, 2012)

El principal problema en los sistemas de reconocimiento de voz ha sido determinar cuáles son las posibles causas que hacen tan difícil llevar a cabo este proceso en condiciones generales. Algunas de las causas que se han conseguido averiguar son:

- La variación fonética de los hablantes. Ninguna persona habla igual que otra, es decir, los sonidos producidos por diferentes personas no suenan igual.
- Las ambigüedades acústicas. A veces no es posible mapear eventos acústicos a sus símbolos fonéticos correspondientes, no es posible realizar una buena decodificación ya que no se dispone de todas las características o fuentes de conocimiento que una persona utiliza durante el proceso de una conversación.
- Mala pronunciación de algunas palabras. A veces ciertas palabras de duración breve se omiten, como preposiciones o conjunciones, o se transforman en sonidos extraños.
- Rapidez al pronunciar las palabras. Con esto se consigue que no haya una clara transición entre las diferentes sílabas, llegando a la fusión u omisión de algunas de ellas.
- Coarticulación. Las características acústicas de los sonidos se ven afectadas por el contexto en el que se encuentran. La mayoría de estos efectos se traducen en alófonos que tiene cada fonema. Ello supone la necesidad de tener múltiples patrones que tengan en cuenta estas variaciones.
- Variaciones temporales. La duración de una palabra e incluso de los sonidos puede cambiar, generando la necesidad de realizar alineamientos dinámicos, que permitan tener en cuenta a estas posibles variaciones.
- Ruidos e interferencias. Las personas podemos hablar en ambientes no idóneos, con ruidos o en presencia de otros sonidos interferentes.

Para la tarea de reconocimiento de voz, en general, se asume que teniendo una entrada acústica “O”, se puede tratar como una secuencia de símbolos u observaciones individuales (por ejemplo, partiendo la señal cada 10 ms y representando cada segmento mediante valores en punto flotante de la energía y la frecuencia de ese segmento). Cada índice entonces representa un intervalo

de tiempo, y sucesivos “oi” representan segmentos consecutivos temporales de la entrada: **fuente:** (Pérez, 2007)

$$O = o_1, o_2, o_3, \ldots, o_t$$

Además, “O” es una secuencia de símbolos tomados de un alfabeto cualquiera “O”. De manera similar, se trata a una elocución como si estuviera compuesta simplemente por una cadena de palabras que pertenecen a un diccionario “W”. El mensaje se representa como sigue:

$$W = w_1, w_2, w_3, \ldots, w_t$$

La implementación probabilística de la intuición anterior, se expresa de la siguiente forma:

$$W = argmax P(W|O)$$

La función probabilística garantiza dar la elocución óptima “W”, pero dada una elocución “W” y una secuencia acústica “O” se necesita calcular P(W|O), lo que significa obtener la probabilidad, dadas todas las posibles secuencias de observación de acuerdo a un alfabeto “O”. Sin embargo, es posible calcular este valor a través del teorema de Bayes que nos dice:

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$

A P(x|y) se le conoce como probabilidad condicional y es la probabilidad de que un segundo evento (x) se presente, si un primer evento (y) ya ha sucedido.

De esta manera, la expresión anterior modificada con el teorema de Bayes queda como sigue:

$$W = argmax \frac{P(O|W)P(W)}{P(O)}$$

Sin embargo, P(O), la probabilidad de la secuencia de observación acústica aún resulta difícil de estimar porque no se sabe cuál es la secuencia de observación de todas las posibles dentro del dominio, pero se sabe que se está examinando la misma secuencia de observación O para cada elocución en

potencia y, por lo tanto, se debe tener la misma P(O) para todas las elocuciones posibles. Como el denominador es el mismo para cada elocución candidata W se puede ignorar y la expresión queda de la siguiente manera:

$$W = argmax \frac{P(O|W)P(W)}{P(O)} = argmax P(O|W)P(W)$$

Así, la elocución “W” más probable dada una secuencia de observación “O” se calcula mediante el producto de dos probabilidades P(O|W) y P(W). Para el caso del reconocimiento de voz P(O|W), la probabilidad de observación, se calcula a través del “modelo acústico” y P(W) , la probabilidad a priori, se calcula a través del “modelo del lenguaje” como sigue:

$$W = P(O|W)P(W)$$

3.2.4. MODELOS CMU-SPHINX KIT DE HERRAMIENTAS

De acuerdo con la estructura de discurso, se utilizaron tres modelos en el reconocimiento de voz. Esas tres entidades se combinan juntas en un motor para reconocer el habla.

Si se va a solicitar su motor para algún otro idioma, es necesario obtener este tipo para sustituirlo en su lugar. La gran ventaja es que para muchos idiomas existen modelos acústicos, diccionarios fonéticos e incluso grandes modelos de lenguaje vocabulario disponibles para su descarga. **fuente:** (CMU-Sphinx, 2014)

3.2.4.1. MODELO ACÚSTICO

El modelo acústico contiene propiedades acústicas para cada senone. Existen dos tipos de modelos:

Modelos independientes del contexto.- que contienen propiedades (vectores característicos más probables para cada fonema).

Modelos dependientes del contexto.- construidos a partir de senones con contexto definido.

El primer paso en cualquier sistema de reconocimiento automático de voz es extraer todas las características, es decir, identificar los componentes de la señal de audio que son buenos para identificar el contenido lingüístico y

descartando todas las otras cosas que lleva la información al igual que el ruido de fondo, emoción, etc.

El principal punto a entender sobre el discurso, fue que cada sonido generado por el ser humano depende; de la forma de la lengua, incluyendo dientes, acento, sexo, edad, cuerdas vocales, laringe, etc. Esto determina qué sonido se produce. Si pudiéramos determinar con exactitud la forma precisa del sonido, esto debería darnos una representación exacta del fonema que se está produciendo. **fuente:** (CMU-Sphinx, 2014)

La forma del **tracto vocal** de las palabras, se manifiesta en la envolvente del espectro de potencia de corto tiempo, y el trabajo de los coeficientes MFCC es representar con exactitud este diagrama.

En el procesamiento de audio, **Mel Frecuencia Cepstrum Coeficiente (MFCC)**, es una representación, de corto plazo, del espectro de potencia de un sonido, en una escala no lineal de la frecuencia Mel (frecuencia en escala musical no lineal). Cada palabra es representada como un vector y todas las palabras forman una matriz, de forma que al pronunciar una palabra en un altavoz o micrófono, se hace el producto punto entre la palabra mencionada y la matriz de palabras MFCC. Es decir que el producto resultante, estadísticamente es la correlación para saber que palabra fue la que se mencionó. **fuente:** (CMU-Sphinx, 2014)

CMU-Sphinx viene con varios modelos acústicos de alta calidad, por ejemplo, el modelo acústico estadounidense en inglés, el francés o chino. Estos modelos fueron optimizados cuidadosamente para lograr el mejor rendimiento reconocimiento y trabajar bien para casi todas las aplicaciones. La mayoría de las aplicaciones son de mando y control para utilizarlas directamente y también son útiles para aplicaciones de vocabulario grandes.

CMUSphinx proporciona la adaptación, que es una forma suficiente para los casos cuando se requiere más precisión. La adaptación se sabe que funciona bien cuando se está utilizando diferentes entornos de grabación como; (de cerca, de lejos, de micrófono o canal telefónico), o a su vez cuando se utiliza un acento ligeramente diferente. La adaptación, funciona bien si se necesita agregar soporte o apoyo a un nuevo lenguaje, simplemente mapeando el conjunto fonético acústico para apuntarlo al conjunto fonético con el diccionario. **fuente:** (CMU-Sphinx, 2014)

Fue posible entrenar un propio modelo y se muestra cómo hacerlo para el motor de reconocimiento de voz. Antes de comenzar con el entrenamiento, nos aseguraremos de estar familiarizados con los conceptos para preparar el modelo de lenguaje y de hecho es necesario entrenar el modelo y tener todos los recursos para hacerlo.

¿Cuánto se necesita para entrenar?

- **1 hora de la grabación.-** para el mando y control de un solo orador
- **5 horas de grabaciones de 200 oradores de mando y control.-** para el reconocimiento de muchos oradores
- **10 horas de grabaciones.-** para el dictado de un solo orador
- **50 horas de grabaciones de 200 oradores.-** para el dictado y reconocimiento de muchos oradores
- Además se necesita el conocimiento sobre la estructura fonética de la lengua (Español Ecuatoriano).
- Además se necesita de tiempo para entrenar el modelo y optimizar los parámetros más o menos en un mes.

3.2.4.2. DICCIONARIO FONÉTICO

El diccionario fonético, se define como la correlación existente entre la palabra propiamente pronunciada y como realmente se escribe dicha palabra (ortografía).

El diccionario fonético, contiene un archivo de mapeo desde palabras hasta fonemas, sólo dos o tres variantes de pronunciación se observan en él, por lo que se puede decir que ésta asignación no es muy eficaz, pero es lo suficientemente práctico la mayor parte del tiempo.

El diccionario no es la única variante de asignación de las palabras a los fonemas, se lo refuerza también crear una función compleja con un algoritmo de aprendizaje automático. **fuente:** (CMU-Sphinx, 2014)

Inicialmente los desarrolladores de CMU-Sphinx que ponen en nuestras manos sus herramientas para el reconocimiento de voz, dejan abierto un diccionario en inglés, francés o chino para su utilización óptima, sin embargo, en el desarrollo del presente proyecto, creamos nuestro propio modelo acústico

y de lenguaje, utilizando la base de datos de wikipwdia y añadiendo a el nuestro propio modelo de lenguaje español Ecuatoriano y así mismo, horas de grabación para el educamiento. **fuente:** (CMU-Sphinx, 2014)

3.2.4.3. MODELO DE LENGUAJE

El modelo de lenguaje se uso para restringir la búsqueda de la palabra. Este modelo define qué palabra podría seguir la secuencia, teniendo en consideración las palabras reconocidas previamente (recuerde que la igualación es un proceso secuencial HMM) y ayudo a limitar significativamente el proceso de correspondencia debido a la extracción de palabras que no son probables.

La mayoría de los modelos de lenguaje utilizan comúnmente modelos de estados **n-gram** que contienen estadísticas de las secuencias de palabras-modelos-lingüísticas, estos estados finitos definen secuencias de voz por la automatización de estados infinitos, a veces con ponderaciones. Para llegar a una buena tasa de precisión, el modelo de lenguaje debe ser un gran éxito en la restricción del espacio de búsqueda. Esto significa que debe ser muy bueno en la predicción de la siguiente palabra. **fuente:** (CMU-Sphinx, 2014)

Tabla 3.
Estados n-gram y predicción estadística para la secuencia de palabras.

DICCIONARIO	TEXTO	N-GRAMS
P1 P2 P3 P4 P5 P6 . . .	P1, P14, P11, P4, P100, P66, P71...	1-grams : la, un, se una, el, de, los, mi.. 2-grams : la casa, el perro, mi carro, la playa, los niños 3-grams : la isla bonita, el perro bravo, el carro nuevo

Un modelo de lenguaje generalmente restringe el vocabulario considerando las palabras que contiene. Eso fue un problema para el reconocimiento de nombres por ejemplo. Para hacer frente a esto, un modelo de lenguaje puede contener trozos más pequeños como sub-palabras o incluso silabas. Hay que tener en cuenta que la restricción de espacio de búsqueda en este caso suele ser peor, y que las correspondientes precisiones de reconocimiento son más bajos que con un modelo de lenguaje basado en palabras.

El kit de herramientas CMU-Sphinx incorpora un programa que genera un archivo de modelo de lenguaje, como es requerido por nuestro decodificador. El modelo de lenguaje contiene un formato de archivo **ARPA** que contiene las probabilidades del modelo idioma en formato ARPA-estándar.

Podríamos encontrar dos más tipos de gramáticas: estadístico (n-gramas) y autómatas (de estados finitos). Mas el propósito de este formato de archivo ARPA, es mostrarse de manera estándar gramaticalmente, de esta forma se podría utilizar una gramática directamente, sin la preocupación de cambiarla a un formato aceptable, por lo que se podría decir que Los archivos de formato ARPA son generalmente intercambiables. **fuente:** (CMU-Sphinx, 2014)

3.2.5. ACOPLAMIENTO DE LOS MODELOS

El acoplamiento de los modelos no fue otra cosa que la convivencia armoniosa que tienen los modelos de lenguaje, acústico y de diccionario para hacer funcionar el sistema de reconocimiento de vos, semejantes a las piezas que conforman el funcionamiento del sistema de un reloj. Como de manera gráfica, se muestra en la figura16. **fuente:** (CMU-Sphinx, 2014)

Para el entrenamiento del modelo acústico, resulto mucho más óptimo trabajar con archivos de audio a 16bits, 16MHz, Mono, esto necesita ser garantizado de cualquier manera por lo que el software SoX es el encargado precisamente de esto último. SoX forza al sistema a que todos los archivos de audio ingresados o grabados tuvieran estas características para el entrenamiento. Esto puede bien ser realizado por Audacity de igual manera

El reconocedor de palabras MFCC, por su parte, hace su desempeño sobre archivos de MS con formato de audio punto WAV (.wav). MFC, garantizó encontrar el número exacto de palabras dentro de un dictado corto, en donde cada palabra contiene su propia frecuencia que espera ser comparada posteriormente. **fuente:** (CMU-Sphinx, 2014)

En el entrenamiento del modelo de lenguaje, se incorporaron los archivos de texto plano con extensión TXT que son archivos que contienen; cuentos, noticias, deportes, documentales, historias etc. Cada palabra contenida en los archivo posee una frecuencia única asignada, que posteriormente será comparada con las palabras de los archivos WAV para el entrenamiento del sistema.

Los **senones**, tienen encargada la importante tarea de reconocer las frecuencias de las palabras de ambos modelos (acústico y de lenguaje), y emparejarlas, y de esta manera ir entrenando el sistema haciéndolo más robusto.

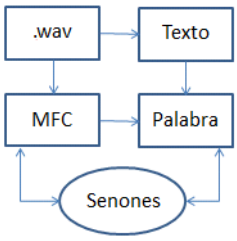


Figura 16. *Acoplamiento del sistema. Reconocimiento de voz.*

Para el sistema de reconocimiento de voz, todas las palabras que conforman las oraciones de los archivos "txt", ahora fueron almacenadas en un nuevo archivo con extensión ".dic", con el propósito de ordenarlas en orden alfabético, teniendo en consideración que hay que eliminar las palabras repetidas.

Acto seguido, se fueron creando los archivos ARPA que aparecen conteniendo las oraciones que se formaron en el reconocimiento a través de los n-grams, que a su vez utiliza el modelo HMM estadístico para formar oraciones.

El esquema de la figura 17, se muestra un pequeño resumen del acoplamiento del sistema de reconocimiento de voz. **fuente:** (CMU-Sphinx, 2014)

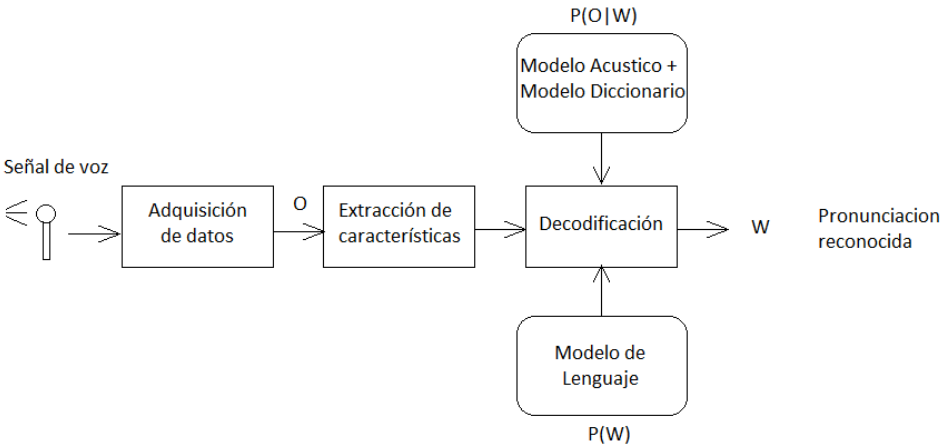


Figura 17. *Acoplamiento de Modelo. Captura el sonido y lo interpreta creando una matriz de palabras que aparecen probabilísticamente según una base de datos.*

3.2.6. SHELL SCRIPT

En informática un **Script**, es un archivo de órdenes de procesamiento por lotes. Es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano. Posee un conjunto de código abierto de instrucciones que deben ser interpretados línea a línea en tiempo real para su ejecución, estos deben ser convertidos a un archivo binario para poder ejecutarlo en sistema de línea de comandos UNIX con extensión SH.

Shell es el intérprete de comandos del sistema, una interfaz de texto de altas prestaciones que sirve fundamentalmente para tres cosas: administrar el sistema operativo, lanzar aplicaciones e interactuar con ellas, y como entorno de programación. Shell interpreta los Sprits linea a línea, por lo que se habla también de programacion en Sell, y por lo que se les suele conocer también como **Shells Scripts**. Pueden ir desde sencillas órdenes hasta complejas series de instrucciones para el arranque del propio sistema operativo. En general, tienen una sintaxis bastante clara y suponen un buen punto de partida para dar los primeros pasos en el mundo de la programación.

Aunque CMUSphinx intenta proporcionar un conjunto completo de herramientas para el desarrollo de aplicaciones de voz, a veces es necesario utilizar otros paquetes o lenguajes de programación. Se ha escogido esta vez continuar con los lenguajes de **Shell Script** que CMUSphinx apoya compatibilidad para trabajar. Siempre considerando las herramientas esenciales para el sistema de reconocimiento, como son sphinx4 o pocketsphinx.

GNU / Linux es la plataforma de desarrollo en la que se está trabajando. Se puede buscar ayudar con los problemas en Windows o Mac, y efectivamente encontrarlos, pero no hay garantías de un correcto funcionamiento. En Linux se puede solucionar muchas tareas ejecutando Scripts complejos, simplemente usando el lenguaje de programación de pitón. En Windows realizar estas mismas tareas podría ser muy problemático.

Para entender de mejor manera, desde un punto de vista lógico y didáctico el funcionamiento de cada modelo implementado en Shell Scripts, sería de importante ayuda realizar un diagrama de bloques o algoritmos para entender secuencia de cómo se maneja el sistema de funcionamiento de cada modelo.

3.2.7. ALGORITMOS

La tecnología de voz pone varias limitaciones en la forma en que es posible llevar a cabo la aplicación. Se necesita considerar todas las maneras posibles que nos ayuden a superar tales limitaciones. Para seguir, es necesario repensar y plantear de qué manera la aplicación se comportará e interactuara con el usuario.

La mayoría de los algoritmos son ampliamente cubiertos en la literatura científica, puesto a que, los nuevos métodos para resolver viejos problemas aumentan con más frecuencia.

Dictado Genérico.- Se necesita de un dominio para el reconocimiento de diálogos, lecturas, reuniones, correos de voz o transcripciones médicas. Tenga en cuenta que el lenguaje no es mucho más restringido que la lengua general propia. En realidad es trata de un pequeño vocabulario con secuencia especializada de términos.

Habrá un montón de nombres o palabras que no estén dentro, y eso constituye un enorme problema en la predicción de palabras, pero tampoco se encontrarán tema relacionados como por ejemplo con la física cuántica y eso es una gran ventaja.

El reconocedor utilizará las restricciones que nos ha facilitado con el modelo de lenguaje para mejorar la precisión del resultado. Mas se va a tener que construir un modelo de lenguaje para su dominio, pero eso no es tan complicado como parece. Así, si se logra cubrir las 60k palabras más comunes, la exactitud será lo mismo que cubrir 120k palabras.

Para crear otros idiomas con una rica morfología, la situación es diferente, pero también solucionable con el método de uso de las sub-palabras basada en la morfología / fonética. También, se tiene que construir un sistema de post-procesamiento, sistema de adaptación y el sistema de identificación del usuario.

Dicho y entendido lo anterior, lo que se hizo es dejar planteados los algoritmos para cada método, de forma que se entienda de una mejor manera, la secuencia lógica de que es lo que se quiere hacer y a donde se pretende llegar.

3.2.7.1. DIAGRAMA GRABACION DE AUDIO

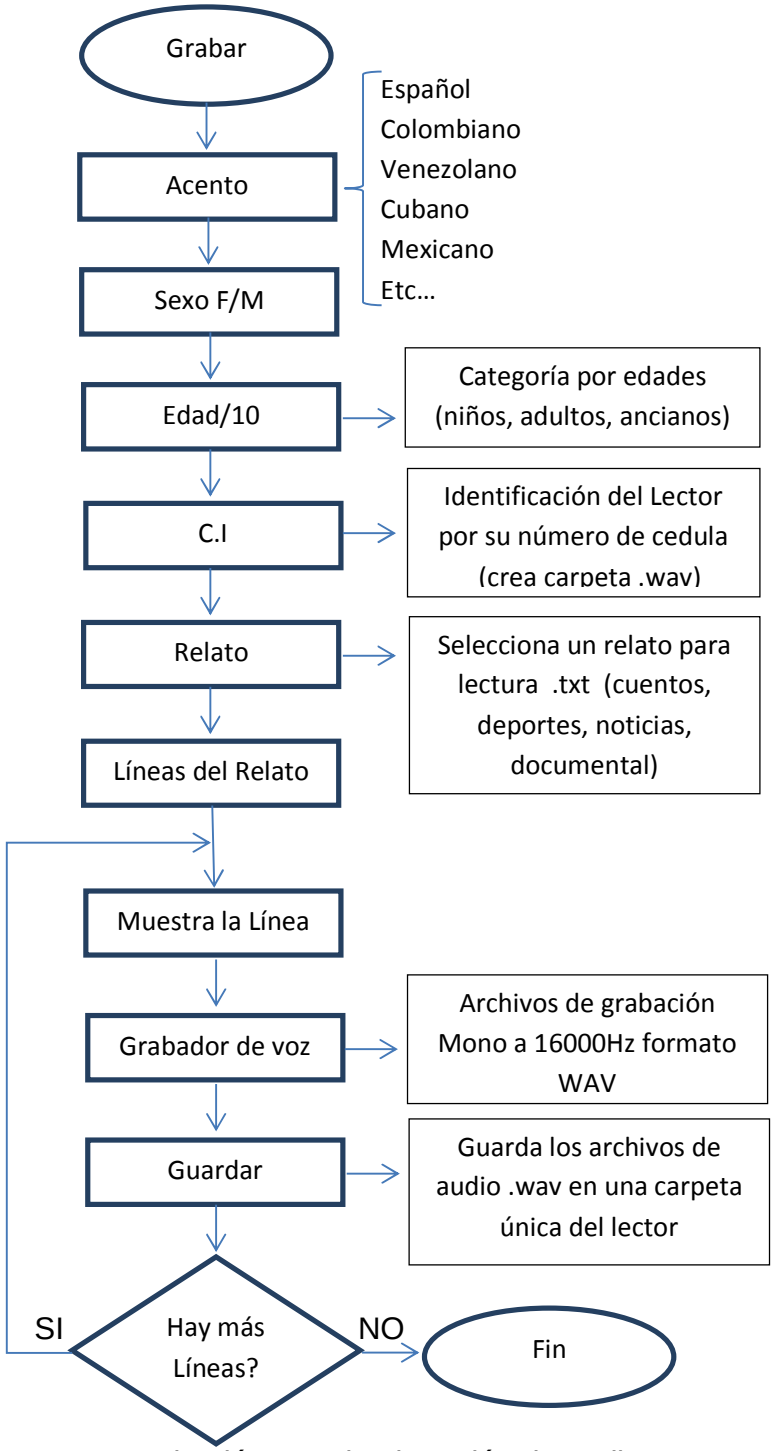


Figura 18. *Grabación para la obtención de audios para alimentar el modelo de lenguaje*

3.2.7.2. DIAGRAMA DEL MODELO DICCIONARIO

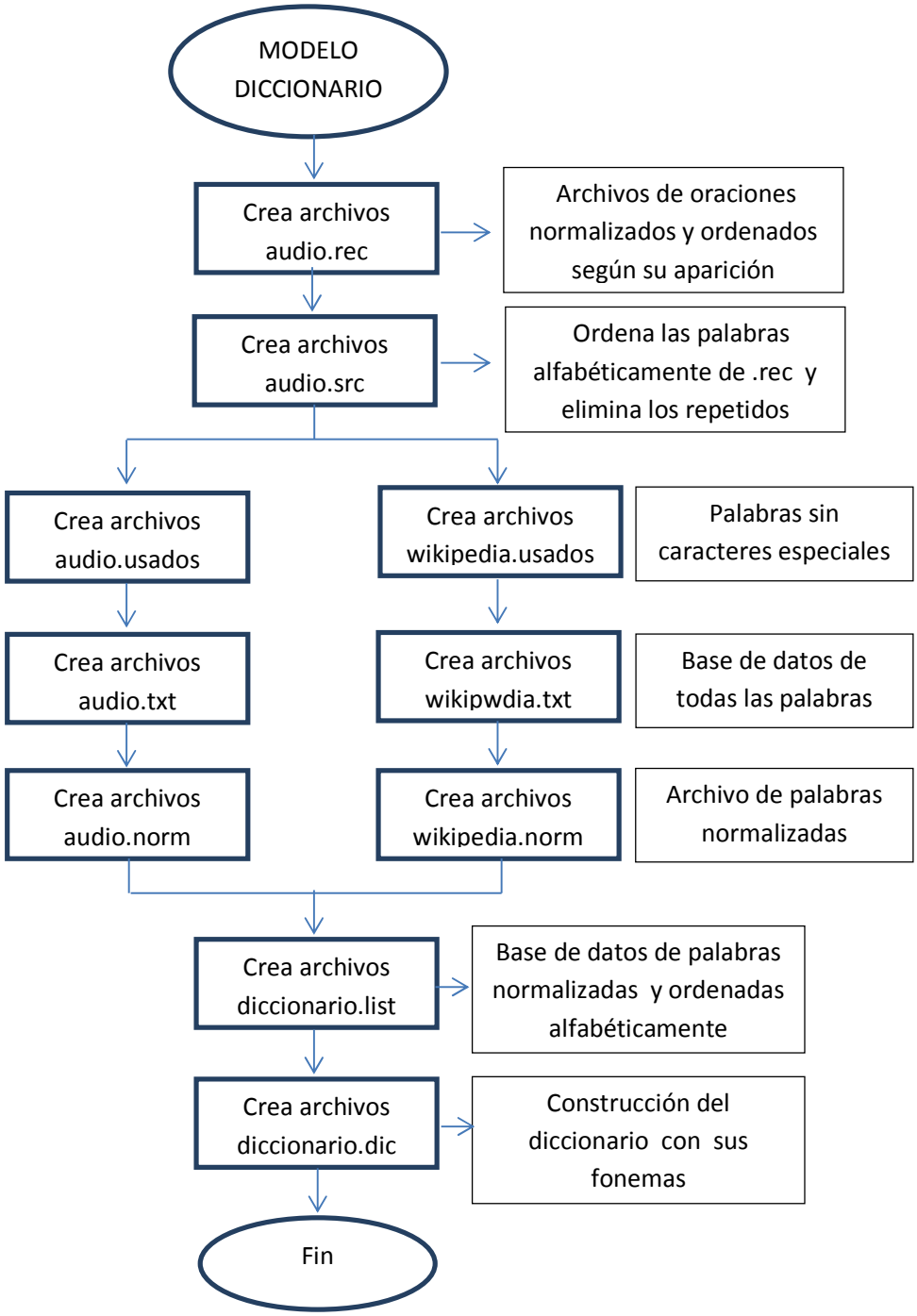


Figura 19 Modelo Diccionario. Crea archivos indispensables

3.2.7.3. DIAGRAMA DEL MODELO DE LENGUAJE

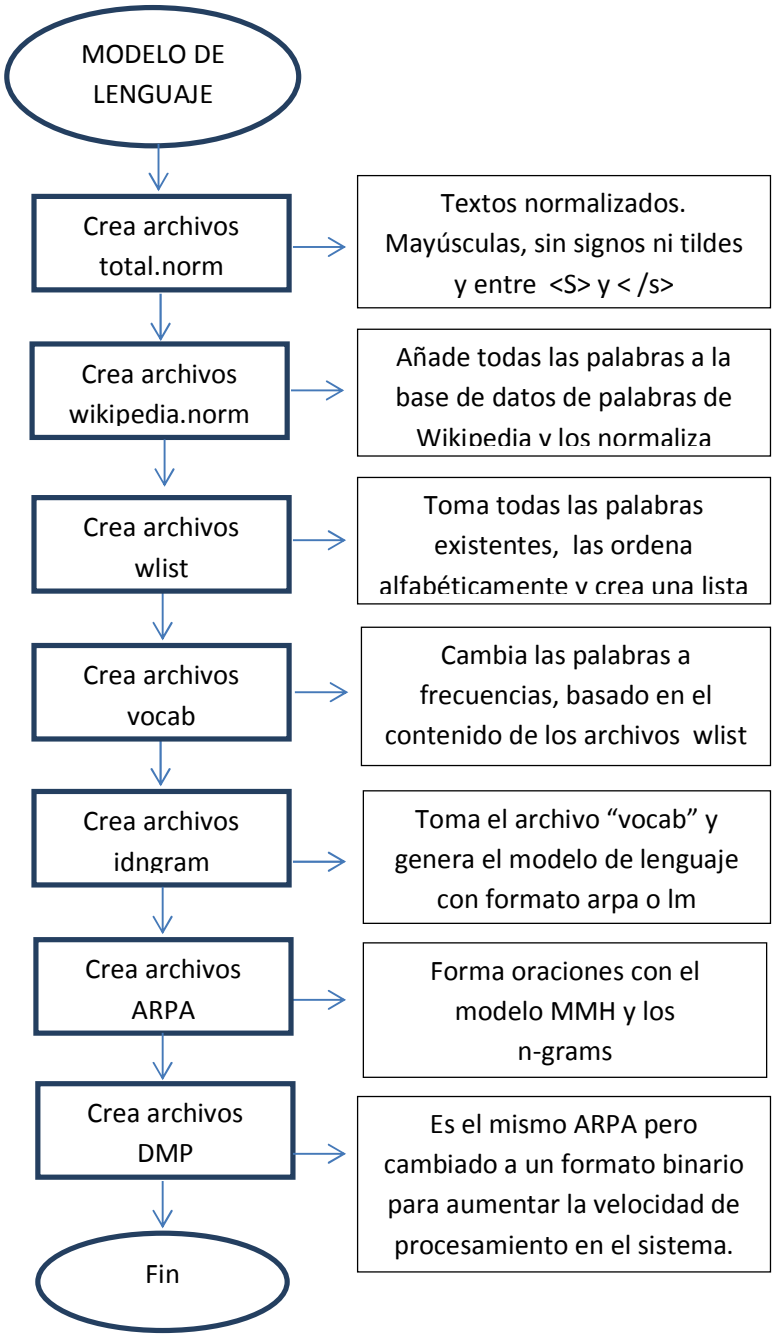


Figura 20. *Modelo de Lenguaje. Crea archivos planos indispensables*

3.2.7.4. DIAGRAMA DEL MODELO ACUSTICO

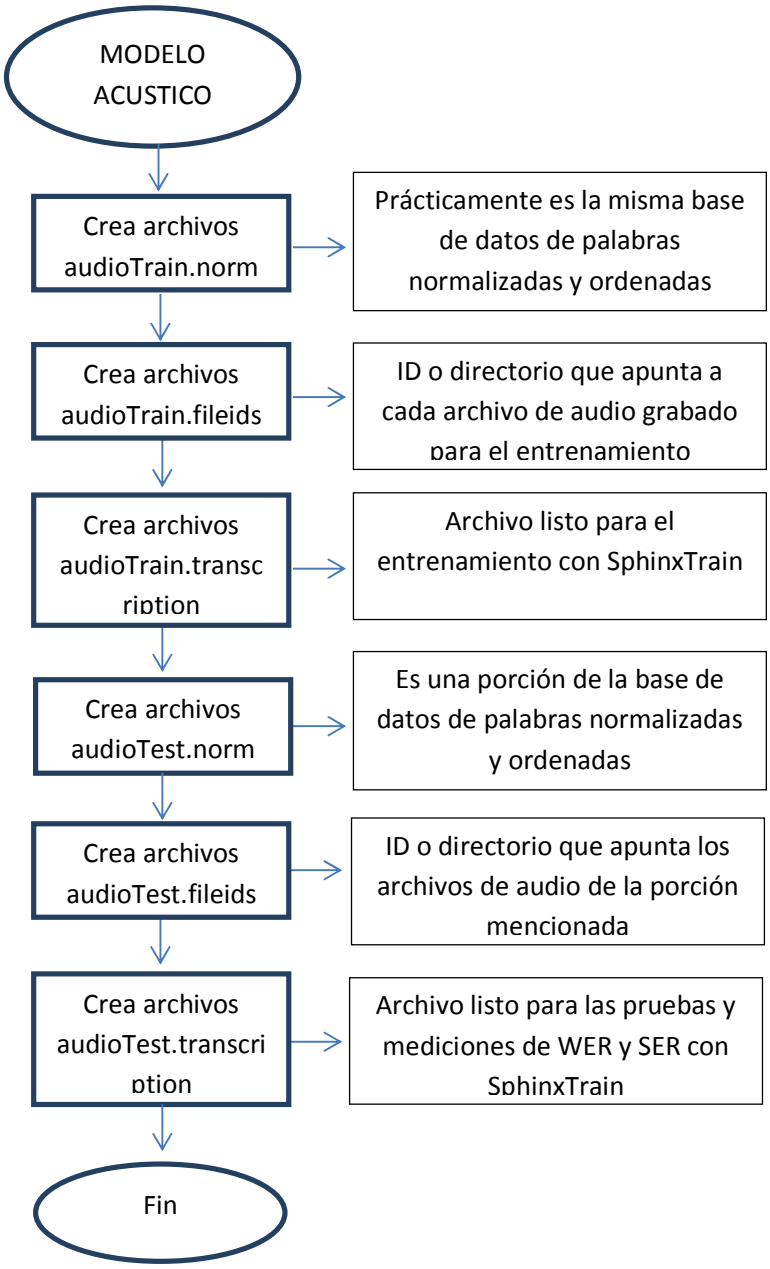


Figura 21 *Modelo Acústico. Biblioteca de audios con extensión WAV Crea audios, elimina el ruido y tractos vocales de la frecuencia de voz*

CAPÍTULO IV
DESARROLLO E IMPLEMENTACION DEL SISTEMA

4.1. DESARROLLO

Esta sección, tiene como propósito explicar el desarrollo y su funcionalidad a través de los requisitos del nuevo sistema. En esta parte se implementan los objetivos como piezas o elementos, diseñados y ordenados cronológicamente como un proceso de transformación para cumplir con las metas y que el sistema funcione correctamente cumpliendo así con todas sus finalidades planteadas.

El caso característico del sistema como un reconocedor de voz en tiempo real, permitió identificar y definir los requerimientos que la aplicación debe cumplir, así también como la interacción que debe existir entre el usuario y el sistema.

Durante la creación de este proyecto se ha seguido una metodología (Métrica III) de desarrollo de software. Esta es una técnica muy utilizada para mostrar la futura capacidad del sistema de una forma simple e intuitiva y cuyos procesos son los siguientes:

- Planificación de Sistemas de Información.
- Desarrollo de Sistemas de Información.
- Mantenimiento de Sistemas de Información.

Esta metodología ha sido concebida para abarcar el desarrollo completo de Sistemas de Información sea cual sea su complejidad y magnitud, por lo cual su estructura responde a desarrollos máximos. La ventaja de esta metodología es que pudo adaptarse y dimensionarse en cada momento de acuerdo a las características particulares del proyecto. Para el desarrollo de nuestro sistema, nos hemos centrado únicamente en los dos primeros procesos, porque se entiende que esta fase, como un proceso, se realizará en el futuro.

Antes de iniciar con el desarrollo de la aplicación de voz, se tuvo en cuenta varios puntos importantes, ya que se debe de considerar que se tengan a la mano todos los recursos necesarios para finalizar el propósito. Ellos definieron la forma en que se implementará la aplicación y se debe de tener muy en cuenta los siguientes: **fuente:** (CMU-Sphinx, 2014)

Cifras de precisión existentes.- En términos de precisión, los resultados podrían ser muy útiles para la interacción con los usuarios. Se puede asegurar que las cifras de precisión pueden aumentar considerablemente mediante el entrenamiento del sistema, pero esto conlleva un proceso que tomaría un buen tiempo y es poco probable que pueda hacerse rápidamente. Sin embargo la paciencia también juego un papel fundamental en el desarrollo.

Por ejemplo, para una tarea de reconocimiento de emisión de noticias, se recomienda realizarla con el 20 a 25% de precisión. Si esto no fuese suficiente para una aplicación, es posible y se considera una modificación, haciendo correcciones o adaptaciones preliminares hasta mejorar las cifras de precisión.

Recursos.- Fue importante tener muy en cuenta, la disponibilidad de los materiales de discurso para la capacitación, pruebas y la optimización del sistema.

El conjunto de pruebas que se hizo, es un tema crítico para cualquier aplicación de reconocimiento de voz, porque debe ser lo suficientemente representativa acústicamente en los términos de lenguaje. La buena noticia es que el conjunto de prueba, no necesito ser nada grande, inclusive basto con grabar 10 minutos para crear un buen conjunto, si se trata de mando y control. Para el aprendizaje y modelado se debe comprobar los recursos que estén presentes.

El creciente interés por la tecnología de voz, hace que las personas contribuyen con la creación de modelos para sus lenguas nativas. En general, se tuvo que recoger el material de audio para el idioma especificado, y lo que se hizo, en realidad no es para nada complicado. Audio libros, películas y noticias, documentales, proporcionaron suficientes material de grabaciones para construir un muy buen modelo acústico usando poco esfuerzo.

Para la construcción de un diccionario fonético, se puede utilizar el sintetizador **TTS** (para reconocimiento de textos escritos / sintetizador de voz) existente, que hoy en día cubren una gran cantidad de idiomas. También se podría iniciar un diccionario a mano y luego extenderla con herramientas de aprendizaje automático.

Para los modelos lingüísticos, se tuvo que encontrar un montón de textos para su dominio. Pudo utilizarse libros de texto, grabaciones ya transcritas o algunas otras fuentes. **fuente:** (CMU-Sphinx, 2014)

Tecnología.- Otra cuestión a considerar, es el conjunto de tecnologías particulares que se utilizaron para la construcción del nuevo sistema de reconocimiento de voz en tiempo real. La regla para elegir entre sphinx4 o pocketsphinx es el siguiente:

- Necesidad de velocidad o la portabilidad → uso pocketsphinx
- Necesita flexibilidad y facilidad de gestión → uso sphinx4

Aunque la gente suele preguntarse con frecuencia, si es mejor utilizar sphinx4 o pocketsphinx, esta pregunta no debe ser preocupante en absoluto, la precisión no es el argumento aquí. Tanto sphinx4 y pocketsphinx proporcionan una precisión aceptable, y aun así depende de muchos factores, no sólo del motor, el motor es sólo una parte del sistema y se debe incluir muchos más componentes.

Si hablamos de grandes decodificador de vocabulario, debe existir un marco de actualización, marco de adaptación y el marco de post-procesado. Todos ellos tienen que cooperar de alguna manera.

Si el sistema debe ser eficiente y razonablemente preciso o a su vez, se está ejecutando en un dispositivo incorporado, o como en nuestro caso particular, se está trabajando en el uso de reconocimiento de voz con un idioma extranjero (español Ecuatoriano), pocketsphinx fue la mejor elección aquí. Pocketsphinx fue nuestra herramienta de trabajo sobre Linux, para el desarrollo del sistema de reconocimiento de voz en tiempo real, que fue liberado como una nueva versión para integrarlo en una interface gráfica en Windows, para el uso de todos. **fuente:** (CMU-Sphinx, 2014)

La flexibilidad de sphinx4, permitió construir un sistema más rápido, es fácil de integrar en servidores de flash para proporcionar reconocimiento basado en web, además con sphinx4, resulta fácil manejar muchos casos haciendo decodificación a gran escala en un clúster. Sphinx4 fue en cambio nuestra herramienta de trabajo mediante la interface gráfica en Windows.

4.1.1. CONSTRUCCIÓN KIT DE HERRAMIENTAS SPHINX

CMU Sphinx kit de herramientas tiene una serie de paquetes para diferentes tareas y aplicaciones, que fueron instaladas en el siguiente orden y que a continuación se presentan cada una de ellas.

- **Sphinxbase.-** Biblioteca de ayuda requerida por Pocketsphinx y Sphinxtrain
- **Pocketsphinx.-** Biblioteca reconocedor escrito en C
- **Sphinxtrain.-** herramientas de capacitación modelo acústicas
- **CMUclmtk.-** Herramientas modelo idioma
- **Sphinx4.-** ajustable reconocedor, modificable escrito en Java

Se recomienda utilizar las últimas versiones disponibles. Para eso, Sphinx proporciona un link de descargas en donde se lo puede hacer de manera segura:

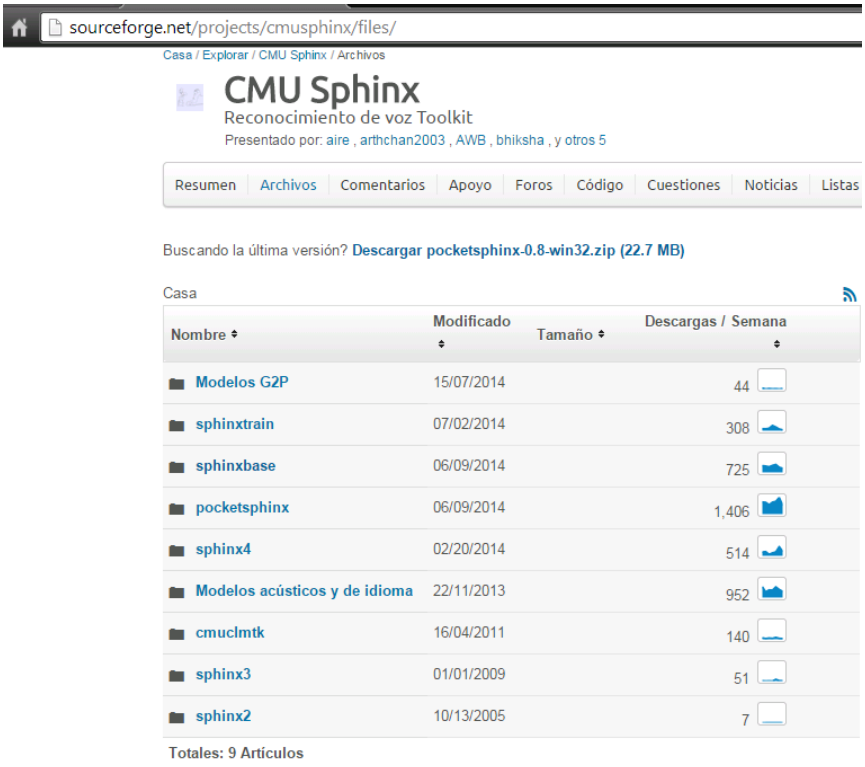


Figura 22. *Modelo Acústico. Link de descarga de los paquetes para la construcción del kit de herramientas Sphinx.*

En primer lugar, se descargaron los paquetes libres `pocketsphinx` y `sphinxbase`. Para hacerlo, CMUSphinx adiciona la página de descargas. Procúrese descargar todo en un mismo directorio.

En un entorno de tipo Unix, el primer paso que consideró a seguir es, la construcción e instalación de `SphinxBase`. Si este se ha descargado directamente desde el repositorio, se necesitó hacer lo siguiente por lo menos una vez para generar la configuración inicial de archivos:

```
% ./autogen.sh
```

Si se ha ejecutado `autogen.sh` al menos una vez, a continuación, solo faltaría compilar e instalar como sigue:

```
% ./configure
% make clean
% make
% sudo make install
```

El Sphinxbase se instaló en la carpeta `/urs/local/`. Pero no siempre todos los sistemas descargan sus bibliotecas dentro de esta carpeta automáticamente. Para cargarlos se necesitó hacer una configurar de ruta para buscar librerías compartidas, también se lo puede hacer trabajando sobre el directorio `/etc/ld.so.conf`, o a su vez, con la exportación de las variables de entorno:

A continuación, para las carpetas `pocketsphinx`, `Sphinxtrain` y `CMUclmtk` se realizó los mismos pasos anteriores.

```
% ./configure
% make clean
% make
% sudo make install
```

Es necesario que estas 4 herramientas Sphinx, deban estar en la misma carpeta de trabajo, al mismo tiempo acompañadas del ejecutable `script.sh` y también de la carpeta contenedora de los archivos, que es la carpeta madre para trabajar sobre el reconocimiento de voz y poder aplicar sobre ella el kit de herramientas Sphinx.

En nuestro caso este directorio fue el siguiente:

```
/home/tesis/Escritorio/tesis/.
```

La figura 23 muestra claramente lo que se explica y como está contenida la carpeta.

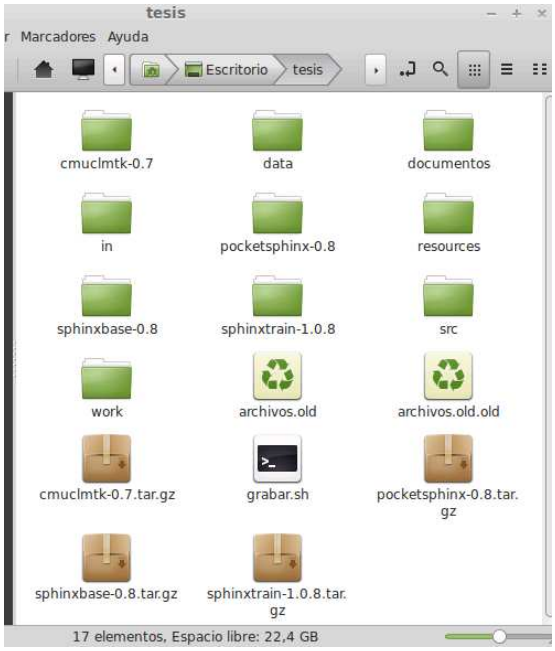


Figura 23 *Modelo Acústico. Link de descarga de los paquetes para la*

En la carpeta “data”, carpeta madre contenedora de los archivos de entrenamiento del sistema de reconocimiento de voz, se hallan tres subcarpetas muy importantes e imprescindibles: “etc” ”out” y ”wav”

Dentro de la carpeta “etc”.- se encuentran creados archivos de texto plano que contienen por ejemplo; archivo de fonemas, archivo de diccionario, archivos DMP, archivos ARPA y los demás archivos necesarios ya antes mencionados en el diagrama de la figura 1 en el capítulo tres. La figura 24, muestra cómo se crean los archivos de texto plano mencionados.

Dentro de la carpeta “out”.- está contenido como tal, los parámetros de modelo para su reproducción bajo Pocketsphinx, y así mismo una vez liberados como una nueva versión, se lo podrá observar en cambio en una interface gráfica hecha en java, bajo Sphinx4 adaptada para voz y video. En la figura 25 se puede ver cada modelo de parámetro mejorado.

Dentro de la carpeta “wav”.- por su parte, se encuentran todos los archivos de audios obtenidos, grabados y depurados con extensión .wav, separados por acento, sexo y edades. De hecho cada carpeta es única de un hablante en particular codificada con su número de cedula. Tal y como se muestra en la figura 26.

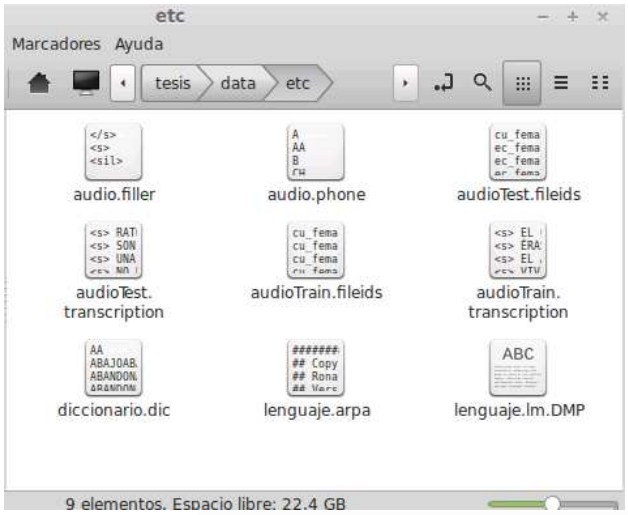


Figura 24. *Modelo de Lenguaje.*

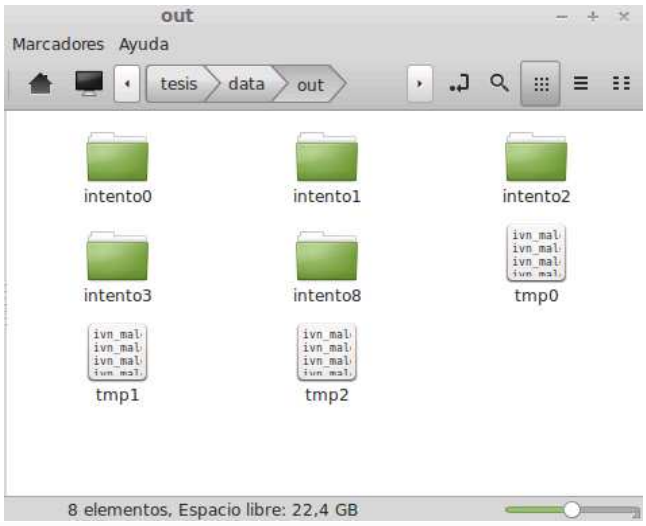


Figura 25*. Parámetros del modelo*

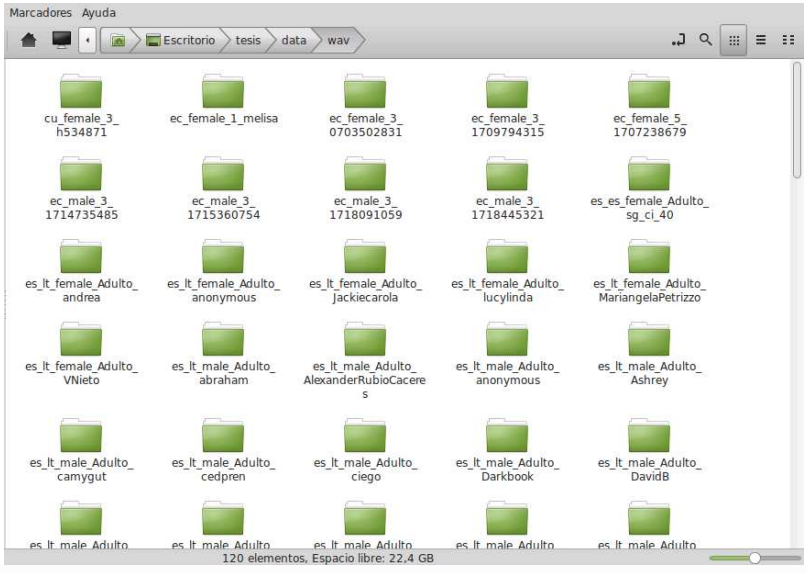


Figura 26. *Modelo Acústico. Biblioteca de audios con extensión WAV.*

4.1.2. CONSTRUCCIÓN MODELOS DE LENGUAJE

Hay dos tipos de modelos que describen el lenguaje; modelo de gramática y modelos de lenguaje estadístico.

El modelo de gramática.- describe tipos simples de lenguaje para mando y control, que por lo general se escriben a mano o son generados automáticamente con un código simple.

El modelo de lenguaje estadístico.-, existen muchas maneras de construirlo. Cuando el conjunto de datos es muy grande, hay mucho sentido en utilizar el modelado de lenguaje CMU kit de herramientas, pero cuando un modelo es pequeño, basta con utilizar un servicio web en línea rápida. Se recomienda que cuando se necesiten más opciones específicas, simplemente se utilice la caja de herramientas favoritas de CMU que construye los modelos ARPA. **fuente:** (CMU-Sphinx, 2014)

4.1.2.1. CONSTRUCCIÓN DE UN MODELO ESTADÍSTICO DE LENGUAJE

Antes que nada, se necesitó hacer una limpieza del texto, expandir abreviaturas, convertir números a palabras, limpiar los elementos que no sean palabra, limpiar signos de puntuación y caracteres especiales, etc. Pero lo más importante que hay que destacar en esta parte, es que el texto debe estar en mayúsculas para no tener problemas en la fonética de reconocimiento. A este texto se lo denomina "**texto normalizado**". **fuente:** (CMU-Sphinx, 2014)

Consultar sobre cómo crear un modelo de lenguaje utilizando los textos de Wikipedia resultó de gran ayuda.

Construcción modelo de formación ARPA y DMP.- El proceso para crear un modelo estadístico de lenguaje propio del idioma que se está desarrollando, fue el siguiente:

1. Preparar un texto de referencia que se utilice para generar el modelo de lenguaje. El modelo de lenguaje kit de herramientas espera que su entrada sea en forma de archivos de **texto normalizados**, con expresiones delimitadas por <s> y </ s>. El resultado fue el conjunto de oraciones que están delimitadas por los marcadores de inicio y final de cada oración: <s> y </ s>. A continuación un pequeño ejemplo de cómo debe lucir un texto normalizado para la posterior creación del modelo de lenguaje: **fuente:** (CMU-Sphinx, 2014)

```
<S> GENERALMENTE NUBLADO CON BROTES DISPERSOS DE LLUVIA Y
LLOVIZNA PERSISTENTE </ s>
<S> ALGUNOS INTERVALOS SECOS TAMBIEN CON BRUMOSO SOL
ESPECIALMENTE EN LA PARTE ORIENTAL DE LA MONTAÑA </ s>
<S> TEMPERATURAS MAS ALTAS DESDE LOS NUEVE HASTA LOS TRECE
GRADOS CELSIUS EN UNA LUZ MODERNA PRINCIPALMENTE AL ESTE CON
BRISA AL SURESTE </ s>
```

2. Generar el archivo de vocabulario. Esto fue, enlistar todas las palabras en un archivo plano usando simplemente:

```
% text2wfreq <weather.txt | wfreq2vocab> weather.tem.vocab

text2wfreq    <    $work/lenguaje.norm    |    wfreq2vocab    >
$work/lenguaje.vocab
```

3. Fue posible modificar el archivo de vocabulario para eliminar todas las palabras tales como; números, faltas de ortografía, nombres, etc. Si existen errores ortográficos, esta es una buena idea para luego fijarlos en la transcripción de entrada. .

4. Generar el modelo de lenguaje formato ARPA con los comandos:

```
% Text2idngram -vocab weather.vocab -idngram weather.idngram
<weather.closed.txt

text2idngram   -vocab   $work/lenguaje.vocab   -temp   $work   <
$work/lenguaje.norm > $work/lenguaje.idngram


% Idngram2lm -vocab_type 0 -idngram weather.idngram -vocab \
     weather.vocab -arpa weather.arpa

idngram2lm  -vocab_type  0  -idngram  $work/lenguaje.idngram  -
vocab $work/lenguaje.vocab -arpa $etc/lenguaje.arpa -buffer
104
```

5. Generar la forma binaria CMU (DMP). Para cargar rápidamente grandes modelos, es necesario convertirlos a un formato binario DMP, ya que al estar en su forma binaria, esto ahorrará mucho tiempo de inicialización en el decodificador. Sphinx4 requiere que se presente el modelo DMP en el componente `TrigramModel` y modelo ARPA al componente `SimpleNGramModel`.

```
sphinx_lm_convert -i weather.arpa -o weather.lm.DMP

sphinx_lm_convert -i $etc/lenguaje.arpa –o
$etc/lenguaje.lm.DMP
```

4.1.2.2. USANDO OTRO KITS DE HERRAMIENTAS DE MODELO DE LENGUAJE

También es posible utilizar otros conjuntos de herramientas diferentes que generen los mismos archivos de texto ARPA. Sin embargo, los archivos resultantes deben arreglarse con el fin de trabajar con los decodificadores Sphinx. Se puede utilizar por ejemplo `sphinx_lm_sort`, utilidad incluida con SphinxBase para ordenar un archivo de modelo de lenguaje formato ARPA para su uso con Sphinx:

A continuación, puede convertir el modelo a formato DMP y utilizarlo como de costumbre. Algunos juegos de herramientas, que por lo general son bastante cómodas de usar y que se pueden probar: **fuente:** (CMU-Sphinx, 2014)

- IRSLM.- Una herramienta para la estimación, la representación y cálculo de modelos de lenguaje estadísticos.
- MITLM.- es un conjunto de herramientas diseñadas para la estimación eficiente de los estadísticos n modelos de lenguaje -gram que implican la estimación de parámetros iterativo.
- SRILM.- es un conjunto de herramientas para la construcción y la aplicación de modelos estadísticos del lenguaje (LMS), principalmente para su uso en el reconocimiento de voz, marcado estadística y la segmentación, y la traducción automática.

4.1.3. ADAPTACIÓN MODELO ACÚSTICO

En esta parte, se describe cómo realizar algunas sencillas adaptaciones del modelo acústico para mejorar el reconocimiento de voz en su configuración.

Se Tuvo en cuenta que en la adaptación, no es necesario hacerla para un hablante en particular. Simplemente se trata de mejorar el ajuste entre los datos de adaptación y el modelo. Por ejemplo, se pudo adaptar la propia voz para realizar un buen dictado, pero también se pudo adaptar el ambiente grabación en particular, canal de transmisión de audio, o el acento de sus usuarios.

Se pudo utilizar el modelo entrenado con datos fonéticos y de emisión para producir un modelo acústico de fonemas al hacer la adaptación. El proceso de adaptación toma datos transcritos y mejora el modelo ya existente creado. Esto pudo dar lugar a un buen resultado. Por ejemplo, fue suficiente tener 5 minutos de discurso para mejorar significativamente la precisión del dictado por la adaptación al hablante particular. **fuente:** (CMU-Sphinx, 2014)

Los métodos de adaptación son un poco diferentes entre PocketSphinx y Sphinx4 debido a los diferentes tipos de modelos acústicos utilizados y la calibración del micrófono.

4.1.3.1. LA CREACIÓN DE UN CORPUS DE ADAPTACIÓN

La primera cosa que se realizó, fue crear un corpus de datos de adaptación. Esto consistió en crear una lista de frases, un diccionario que describe la pronunciación de todas las palabras en la lista, y una grabación hablada de cada frase. **fuente:** (CMU-Sphinx, 2014)

El conjunto real de las sentencias que se utilizo fue un tanto arbitraria, pero lo ideal es que debe tener una buena cobertura de las palabras o fonemas de uso más frecuente en el conjunto de oraciones o el tipo de texto que se desea reconocer. Es necesario tener un archivo de control, un archivo de transcripción, y un diccionario como sigue en un muy pequeño ejemplo para entender:

- **arctic5.txt.-** correspondientes a las oraciones plasmadas en un solo archivo de texto plano como tal, por ejemplo

```
Por vigésima vez, esa noche los dos hombres se dieron la
mano.
Me alegro mucho de volverte a ver.
Nunca vamos a olvidarnos, lo prometo.
Dios los bendiga, espero verlos siempre así hasta el día de
mi muerte
Uno siempre espera lo mejor de las cosas.
```

- **arctic5.fileids.-** correspondiente al número total de oraciones anteriores, denotadas de esta manera:

```
arctic_0001
```

```
arctic_0002
arctic_0003
arctic_0004
arctic_0005
```

- **arctic5.transcription.-** correspondiente a la transcripción de oraciones normalizadas, sin signos de puntuación ni tildes entre los indicadores <s> y </ s > tal y como sigue:

```
<S> POR VIGESIMA VEZ ESA NOCHE LOS DOS HOMBRES SE DIERON LA
MANO </ s> (arctic_0001)
<S> ME ALEGRO MUCHO DE VOLVERTE A VER </ s> (arctic_0002)
<S> NUNCA VAMOS A OLVIDARNOS LO PROMETO </ s> (arctic_0003)
<S> DIOS LOS BENDIGA ESPERO VERLOS SIEMPRE ASI HASTA EL DIA
DE MI MUERTE </ s> (arctic_0004)
<S> UNO SIEMPRE ESPERA LO MEJOR DE LAS COSAS </ s>
(arctic_0005)
```

- **arctic5.dic.-** correspondiente a la creación del diccionario de pronunciación de las palabras

```
A A
ALEGRO A L E G R O
DE D E
DIERON D I E R O N
DOS D O S
HOMBRES O M B R E S
LA L A
LOS L O S
MANO M A N O
ME M E
POR P O R
QUIERE K I E R E
VER B E R
VEZ B E S
VIGEZIMA B I G E S I M A
VOLVERTE B O L B E R T E
.
.
.
```

4.1.3.2. GRABACIÓN DE LOS DATOS DE LA ADAPTACIÓN

Antes que nada, fue muy necesario asegurarse de que se grabe en una tasa de muestreo 16 khz en mono, es decir en un solo canal.

En Linux, esto se pudo lograr fácilmente en una sola línea de comando. Antes se debe comprobar si están cargados los paquetes SoX, y si no, instalarlo inmediatamente usando este comando.

```
sudo apt-get install sox
```

Ahora, para grabar las oraciones y asegurarnos una tasa de muestreo de 16 khz en mono, wav s16bits, utilizamos el siguiente comando como sigue

```
for i in `seq 1 20`; do
       fn=`printf arctic_%04d $i`;
       read sent; echo $sent;
       rec -r 16000 -e signed-integer -b 16 -c 1 $fn.wav
2>/dev/null;
done < arctic20.txt
```

Esto hace eco de cada frase a la pantalla y empezó a grabar inmediatamente. Se dio un Control-C para pasar a la siguiente frase sucesivamente hasta terminar. Al final se vieron generados los siguientes archivos en el directorio actual: **fuente:** (CMU-Sphinx, 2014)

```
arctic_0001.wav
arctic_0002.wav
.....
arctic_0019.wav
arctic20.dic
arctic20.fileids
arctic20.transcription
arctic20.txt
```

Si se ha pulsa Control-C antes de que termine de hablar una sentencia, es probable que el registro pueda haberse truncado en la última palabra. Verifíquese que estas grabaciones suenen bien.

Para ello, bien pudieron reproducirse con :

```
for i in * .wav; no jugar $ i; hecho
```

4.1.3.3. USANDO EL MODELO DE LENGUAJE ESPAÑOL-ECUATORIANO

Como se adaptó una propiedad genérica del audio, a un canal, acento y lengua propia, entonces resulto necesario recoger muchas grabaciones de

forma manual. Por ejemplo, se pudo grabar y transcribir cientos de oraciones y utilizarlas para mejorar la precisión de reconocimiento por medio de la adaptación.

Para utilizar el modelo pocketsphinx, simplemente fue cuestión de poner los archivos de modelo a los recursos de la aplicación. Luego se apuntó a él con la opción –hmm. A continuación veamos cómo se escribe: **fuente:** (CMU-Sphinx, 2014)

```
% pocketsphinx_continuous -hmm <your_new_model_folder> -lm
<your_lm> -dict <your_dict>.

pocketsphinx_continuous                                    -hmm
/home/tesis/Escritorio/tesis/data/out/intento8/model_paramete
rs/data.cd_cont_400                                         -lm
/home/tesis/Escritorio/tesis/data/out/intento8/src/lenguaje.l
m.DMP                                                    -dict
/home/tesis/Escritorio/tesis/data/out/intento8/src/diccionari
o2.dic -bestpath
```

De apenas unas pocas frases, se debe obtener alrededor del 10% de mejora en relación WER.

4.1.4. CONSTRUCCIÓN DEL MODELO ACÚSTICO

Proyecto CMUSphinx viene con varios modelos acústicos de alta calidad. Hay modelos acústicos de inglés, también puede utilizar los modelos franceses o chinos mandarin entrenados en una enorme cantidad de datos acústicos.

CMUSphinx proporciona formas de adaptación suficientes para obtener más precisión. La adaptación se sabe que funciona bien cuando se está utilizando diferentes entornos de grabación, o cuando un acento, ligeramente diferente está presente, o incluso otro idioma. Por ejemplo, funciona bien si es necesario agregar rápidamente un nuevo texto de apoyo simplemente mapeando modelos fonéticos acústicos para apuntar fonéticamente con el diccionario. **fuente:** (CMU-Sphinx, 2014)

En este caso particular, se entrenó un propio modelo de lenguaje del idioma Español Ecuatoriano y se muestra cómo hacerlo para el motor de reconocimiento de voz con CMUSphinx.

Antes de comenzar con el entrenamiento, se debe asegurar de que se está familiarizado con los conceptos, se preparó el modelo de lenguaje (de hecho esto es necesario para entrenar el modelo) y obviamente los recursos para hacerlo.

Requerimientos para entrenar el modelo.- si se quieres crear un modelo acústico para una nueva lengua / dialecto, o simplemente se necesita de un modelo de especialización de aplicación vocabulario pequeña. Si se tiene un montón de datos para entrenar entonces: **fuente:** (CMU-Sphinx, 2014)

- Crear desde cero un modelo acústico de lenguaje propio (TIEMPO).
- Se necesita un modelo específico para la aplicación (VOCABULARIO)
- Se debe tener abundantes datos para entrenar .- **Por lo menos se debe recolectar 50 horas de grabaciones de al menos 200 oradores** para el reconocimiento de voz de cualquier persona
- Conocimiento sobre la estructura fonética de la lengua
- Entrenar el modelo y optimizar los parámetros (1 mes o más)

Se tuvo en cuenta que la cantidad de datos que figuran aquí, son necesarios para entrenar el modelo. Si se obtuviesen significativamente menos datos que los que se especifica, no se puede entrenar a un buen modelo.

4.1.4.1. PREPARACIÓN DE DATOS

El entrenador, aprende los parámetros del modelo de las unidades de sonido, utilizando un conjunto de muestras de señales de voz. Esto se conoce como una base de datos de entrenamiento. La base de datos contiene la información necesaria para extraer estadísticas del discurso en forma de modelo acústico.

El entrenador necesita que se le informe las unidades de sonido para aprender los parámetros, y la secuencia en que se produce cada señal de voz en su base de datos de entrenamiento. **fuente:** (CMU-Sphinx, 2014)

Esta información se proporciona al entrenador a través de un archivo llamado “archivo de transcripción”, en el que la secuencia de palabras y sonidos no vocales se escriben exactamente como se produjeron en una señal de voz,

seguido de una etiqueta que se utiliza para asociar esta secuencia con la señal de voz correspondiente.

Entonces, el entrenador mira dentro de un *diccionario* y mapea cada palabra, con el objetivo de asociar cada señal dicha, y deducir la secuencia de unidades de sonido que producen. **fuente:** (CMU-Sphinx, 2014)

Por lo tanto, además de las señales de voz, también se recibió una serie de transcripciones de la base de datos en un solo archivo, y adicional a eso, dos diccionarios. Uno en el que las palabras legítimas del lenguaje son mapeadas en secuencias de unidades de sonido (o sub-palabra), y el otro asignado a las correspondientes palabras en el que los sonidos no son de voz o unidades de sonido similares del habla. Nos referiremos a la primera como el idioma propiamente, y el segundo como el diccionario de relleno.

Después del entrenamiento, es obligatorio ejecutar el decodificador para comprobar los resultados del entrenamiento. El decodificador toma un modelo de pruebas de parte de la base de datos y referente a la transcripción y estima la calidad (WER) del modelo. Durante la fase de pruebas, se utiliza el modelo de lenguaje con la descripción del orden de palabras en la lengua que se está utilizando. **fuente:** (CMU-Sphinx, 2014)

En primer lugar, fue necesario diseñar una base de datos para el entrenamiento o descargar una ya existente, pero no olvidar de convertir a un formato adecuado para poder utilizarla.

Una base de datos debe ser una buena representación de lo que se va a reconocer. En nuestro caso, se alimenta la base de datos con todos los temas que estén a nuestro alcance, como por ejemplo, noticias, cuantos infantiles, deportes, reportajes, decoración de hogares, conversaciones telefónicas, anuncios, comida, etc. Utilizando vos de hombre, de mujer de diferentes edades y con diferentes tractos vocales, además de utilizar diversos escenarios de grabación (con ruido, sin ruido, de cerca, de lejos) para tratar de crear un diccionario lo suficiente mente robusto para el reconocimiento de voz. Sin embargo, si no se tiene las condiciones para obtener suficientes discurso de grabación, sin duda se pudo utilizar otra expresión, por ejemplo se pudo utilizar grabaciones y convertirlas a WAV. **fuente:** (CMU-Sphinx, 2014)

La Base de datos se construyó con suficientes hablantes de grabación, variedad de condiciones de grabación, variaciones acústicas suficientes y todas las posibles sentencias lingüísticas. El tamaño de la base de datos depende de

la complejidad de la tarea que desea manejar como se mencionó anteriormente, pero en este caso se considera una base de datos robusta.

Una base de datos debe tener las dos partes antes mencionadas (formación y parte de prueba). Por lo general, la parte de prueba es de aproximadamente 1/10 del tamaño total de datos, porque no es recomendable tener los datos de prueba a más de 4 horas de grabaciones.

Luego de diseñar los mensajes de bases de datos, hubo que post-procesarlos en una etapa llamada de depuración. La depuración es manual y consiste en escuchar cada grabación e ir corrigiendo los errores para asegurar que los resultados de audio, corresponde en realidad a las solicitudes. La estructura de archivos de la base de datos es: **fuente:** (CMU-Sphinx, 2014)

- **ETC**
  - **your_db.dic** → diccionario fonético
  - **your_db.phone** → archivo de fonemas
  - **your_db.lm.DMP** → Modelo de lenguaje
  - **your_db.filler** → Lista de rellenos
  - **your_db_train.fileids** → Lista de archivos para el entrenamiento
  - **your_db_train.transcription** → Transcripción para el entrenamiento
  - **your_db_test.fileids** → lista de archivos para las pruebas
  - **your_db_test.transcription** → Transcripción para las pruebas
- **WAV**
  - **speaker_1**
  - **file_1.wav** → Grabación de expresión verbal
  - **speaker_2**
  - **file_2.wav**

Vamos a ir a través de cada archivo describiendo su formato:

Archivos FILEIDS (your_db_train.fileids y your_db_test.fileids).- son archivo de texto plano, que contienen los directorios o IDs con los nombres de las grabaciones, uno por línea, por ejemplo.

```
female_ABIGAIL / TEXTO_DEPORTES4_arctic_0015
female_NELLY / TEXTO_DOCUMENTALES4_arctic_0023
male_IVAN / TEXTO_NOTICIAS3_arctic_0017
male_VICTOR / TEXTO_CUENTOS3_arctic_0017
```

Los archivos **fileids** contienen la ruta en un sistema de archivos relativa al directorio wav. Hay que tener en cuenta que el directorio de cada archivo de audio dentro de fileids, no debería tener extensiones, sólo sus nombres.

audioTrain.fileids.- Archivo denominado para el entrenamiento del modelo acústico. Contiene los directorios de toda la base de datos existente.

audioTest.Fileids.- Archivo denominado para las pruebas y mediciones de WER y SER. Contiene los directorios de una porción de la base de datos destinados a la decodificación.

Archivos TRANSCRIPTION (your_db_train.transcription y your_db_test.transcription).- son archivos de texto, con la transcripción para cada archivo de audio. Es decir, el espejo de lo que se dice en el audio, propiamente plasmado en un archivo de texto plano.

```
<S> hola mundo </ s> (file_1)
<S> buenos días </ s> (file_2)
<S> mucho gusto </ s> (file_3)
<S> buen provecho </ s> (file_4)
```

Es indispensable que cada línea de dialogo comience con <s> y termina con </ s> seguido de su respectivo id en paréntesis. Tenga en cuenta que el paréntesis sólo contiene el archivo, sin directorio **speaker_n**. Es fundamental la coincidencia exacta entre los archivos **fileids** y el archivo de **transcripción**, el número de líneas en tanto debe ser idéntico. La última parte del archivos id (Speaker1 / file_1) y el id del enunciado file_1 debe ser la misma en cada línea. A continuación se muestra un ejemplo de un archivo fileids creado para el archivo de la transcripción anterior.

```
speaker_1 / file_1
speaker_2 / file_2
speaker_3 / file_3
speaker_4 / file_4
```

Las Grabaciones de voz (archivos wav), Es muy importante que deban estar en formato WAV con frecuencia de muestreo específico - 16 kHz, 16 bits, mono para aplicación de escritorio. Los archivos de audio no deben ser muy largos, pero tampoco deben ser muy cortos. La longitud óptima no es inferior a 5 segundos y no más de 30 segundos.

Hay que tener mucho cuidado con la cantidad de silencio en el comienzo de la expresión y en el final de la expresión no debe exceder de 0,2 segundos. **fuente:** (CMU-Sphinx, 2014)

Es fundamental contar con los archivos de audio en un formato específico. Sphinxtrain admite algunas variedades de frecuencias de muestreo, pero por defecto está configurado para entrenar de mono 16bit 16kHz de formato WAV.

Diccionario fonético (your_db.dic).- Este archivo posee una línea por cada palabra. La palabra propia y después su transcripción fonética.

```
HOLA O L A
AMIGOS A M I G O S
```

Se pudo encontrar diccionarios fonéticos en Wikipedia o se puede utilizar un libro de fonética para crear el diccionario, obviamente propia del lenguaje español. Se debe tener mucho cuidado al no utilizar variantes de mayúsculas y minúsculas, como "e" y "E".

Sphinxtrain no admite caracteres especiales como '*' o '/' pero soporta la mayoría de otros como "+" o "-". Pero para estar seguros, es recomendable que se utilice únicamente un set fonético alfanumérico.

Se tiene que reemplazar caracteres especiales, como dos puntos o guiones o tildes, con algo alfanumérico. Por ejemplo, sustituya "á" con "aa" para que sea sólo alfanumérico. **fuente:** (CMU-Sphinx, 2014)

Hay una cosa muy importante aquí. Para una base de datos de vocabulario grande, se necesitó de una buena representación fonética, estos fonemas simplemente se describen en algunos libros. Si no se tiene un libro fonético, sólo puede utilizar la ortografía de la palabra y da muy buenos resultados:

```
UNO U N O
DOS D O S
```

Archivo de fonemas (your_db.phone).- debe tener un fonema en cada línea. El número de fonemas debe coincidir con los fonemas usados en el diccionario más el fonema especial SIL para representar el silencio.

El proceso de desarrollo léxico consistió en definir un conjunto de fonéticas y generar una lista de pronunciación de las palabras (diccionario) para el entrenamiento acústico y del modelos de lenguaje.

Pude llegar a suceder de que hay muchos errores de decodificación procedentes de las vocales y de algunos fonemas como por ejemplo S y Z. Dado que casi todas las sentencias fueron hablada en español mexicano, el fonema Z (un fonema castellano) no tiene sentido, por lo que para esta nueva versión se elimina y es remplazado solo por S. Por lo cual resultó imprescindible y de gran ayuda la investigación de una tabla de fonemas que se utilizó para reducir los errores que se puedan dar. **fuente:** (CMU-Sphinx, 2014)

A continuación se presenta una tabla de fonemas general del lenguaje español para adaptarlo a la creación de nuestro propio idioma Ecuatoriano español.

Tabla 4.
Fonemas para la construcción del reconocimiento de voz Español-Ecuatoriano.

Letra	Fonema	Letras	Ejemplo	Transcripción
A	/a/	A	AMO	A M O

AA	/a''/	Á	MÁS	M AA S
B	/B/	B,V,W	BIEN	B I E N
CH	/č/	CH,X	CHINO	CH I N O
D	/D/	D	DEDO	D E D O
E	/e/	E	PERA	P E R A
EA	/e''/	É	CAFÉ	C A F EA
F	/f/	F	FOCA	F O K A
G	/G/	G,W,H	HUESO	G U E S O
I	/i/	I	LISO	L I S O
IA	/i''/	Í	TENÍA	T E N I AA
J	/x/	G,J	JAMÓN	JAMOAN
K	/k/	C,K,Q	QUESO	KESO
L	/l/	L	LAGO	LAGO
M	/m/	M	MAMÁ	MAMAA
N	/n/	N	EN	EN
NY	/ñ/	Ñ	NIÑO	NINYO
O	/o/	O	OJO	OJO
OA	/o''/	Ó	CAMIÓN	CAMIOAN
P	/p/	P	PAVO	PABO
R	/r/	R	CARO	KARO
RR	/R/	R	RATÓN	RRATOAN
S	/s/	S,(C,Z)*	CASA	KASA
T	/t/	T	TOMA	TOMA
U	/u/	U,W	PUNTO	PUNTO
UA	/u''/	Ú	CANCÚN	KANKUAN
Y	/λ/	LL,Y	RAYO	RRAYO

Fuente: (Pérez, 2007)

Otro cambio importante en esta versión fue que hay cinco nuevos fonemas añadido para simbolizar los acentos del español:

```
AA
EA
IA
OA
UA
CH
NY
RR
```

Archivo de modelo Idioma LM.DMP (your_db.lm.DMP).- puede estar en formato ARPA o en formato DMP. Como se explicó con anterioridad en la formación de modelos ARPA. **fuente:** (CMU-Sphinx, 2014)

Diccionario de relleno (your_db.filler) contiene los fonemas de relleno (no cubiertos por el modelo de lenguaje, son sonidos no lingüísticos como por ejemplo el aliento, hmm o reír):

```
<S> SIL
</ s> SIL
<sil> SIL
```

4.1.4.2. RECOPILACIÓN DE LOS PAQUETES REQUERIDOS

Se requieren los siguientes paquetes para la formación:

- sphinxbase-0.8
- SphinxTrain-0.8
- pocketsphinx-0.8

Además, si se descarga los paquetes con una extensión **.gz**, necesariamente hubo que instalar **gunzip** o el equivalente a desempaquetar los archivos. Instalados los paquetes de Perl y Python en la carpeta de ruta ejecutable.

Es recomendable que si se entrena en Linux, de esta manera será capaz de utilizar todas las características de Sphinxtrain. Básicamente, fue necesario poner todo dentro de la carpeta raíz, descomprimir y descomprimir allí, y ejecutar configure, make clean, make y make install en cada carpeta del paquete, de hecho la carpeta base de datos, tuvo que estar dentro de esta carpeta como se mencionó anteriormente en "**la construcción del paquete de herramientas Sphinx**". En el momento en que termine esto, se obtuvo un directorio tutorial con el siguiente contenido. **fuente:** (CMU-Sphinx, 2014)

```
an4
an4_sphere.tar.gz
SphinxTrain
SphinxTrain.tar.gz
pocketsphinx
pocketsphinx.tar.gz
sphinxbase
sphinxbase.tar.gz
```

Después de instalar el software es posible que deba actualizar la configuración del sistema para que el sistema sea capaz de encontrar las librerías dinámicas. Por ejemplo

```
export PATH=/usr/local/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/lib
export PKG_CONFIG_PATH=/usr/local/lib/pkgconfig
```

Obviamente la carpeta puede ser una carpeta arbitraria pero recuerdese que se debió actualizar la configuración del entorno y de sus variables después de eso. **fuente:** (CMU-Sphinx, 2014)

Si se encuentra que los binarios fallan al cargar las librerías dinámicas, entonces sale un mensaje similar a este:

```
Failed to open libsphinx.so.0 no such file or directory.
```

Lo que significa que no se ha configurado el entorno correctamente.

4.1.4.3. CONFIGURACIÓN DEL ENTRENAMIENTO DE SCRIPTS

Para iniciar el entrenamiento, se cambió a la carpeta de base de datos y enseguida se ejecutó los siguientes comandos:

```
% sphinxtrain -t an4 setup

  sphinxtrain -t data setup
```

Obviamente, no se debe olvidar reemplazar an4 con el nombre de la tarea que se está trabajando.

Esto ayudo a copiar todos los archivos de configuración necesarios en la subcarpeta etc de la carpeta de base de datos, y preparar la base de datos para el entrenamiento: **fuente:** (CMU-Sphinx, 2014)

```
  etc
  feat
  logdir
  model_parameters
```

```
  model_architecture
  wav
```

Después del paso de configuración sólo dos carpetas de las anteriores serán las indispensables, las otras serán creadas durante el proceso de formación:

```
etc
wav
```

Después de eso, se tuvo que editar los archivos de configuración en la carpeta etc, hay muchas variables aquí, pero para empezar se cambió sólo algunos. En primer lugar encontrar el directorio `etc / sphinx_train.cfg`

Configuración del formato de audio de base de datos.-

```
# Audio waveform and feature file information
$CFG_WAVFILES_DIR = "$CFG_BASE_DIR/wav";
$CFG_WAVFILE_EXTENSION = 'wav';
$CFG_WAVFILE_TYPE = 'mswav'; # one of nist, mswav, raw
$CFG_FEATFILES_DIR = "$CFG_BASE_DIR/feat";
$CFG_FEATFILE_EXTENSION = 'mfc';
$CFG_VECTOR_LENGTH = 13;
```

Configure ruta de los archivos.-

```
# Variables used in main training of models
$CFG_DICTIONARY     = "$CFG_LIST_DIR/diccionario.dic";
$CFG_RAWPHONEFILE   = "$CFG_LIST_DIR/audio.phone";
$CFG_FILLERDICT     = "$CFG_LIST_DIR/audio.filler";
$CFG_LISTOFFILES    = "$CFG_LIST_DIR/audioTrain.fileids";
$CFG_TRANSCRIPTFILE                                          =
"$CFG_LIST_DIR/audioTrain.transcription";
$CFG_FEATPARAMS     = "$CFG_LIST_DIR/feat.params";
```

Si se está entrenando modelos continuos de vocabulario grande y se tiene más de 100 horas de datos, es mejor en su lugar poner 32. Puede ser de cualquier grado de 2, 4, 8, 16, 32, 64. Depende mucho de las horas de grabación que se hayan obtenido para el entrenamiento. **fuente:** (CMU-Sphinx, 2014)

```
# Number of (senones) to create in decision-tree clustering

$CFG_N_TIED_STATES = 1000;
```

Este valor es el número de senones para entrenar en un modelo. Entre más senoses tenga el modelo, hay más precisión en la discriminación de los sonidos. Por otro lado, si se tiene demasiados senones, el modelo no será lo suficientemente genérico como para reconocer el lenguaje. Eso significa que el WER será mayor en los datos. Es por eso que es importante que no se sobre entrene los modelos. **fuente:** (CMU-Sphinx, 2014)

En caso de que haya demasiados senones invisibles, las advertencias se generan en el registro de la norma en el escenario 50 a continuación:

```
ERROR: "gauden.c", line 1700: Variance (mgau= 948, feat= 0,
density=3,

component=38) is less then 0. Most probably the number of
senones is too

high for such a small training database. Use smaller
$CFG_N_TIED_STATES.
```

El número aproximado de senones y número de densidades se proporciona en la tabla siguiente:

Tabla 5.
Densidad de proporción para el entrenamiento (Se debe elegir el número de senones óptimo)

vocabulario	Horas en db	Senones	Densidades	Ejemplos
20	5	200	8	Reconocimiento de dígitos
100	20	2000	8	RM1 Mando y control
5000	30	4000	16	WSJ1 Pequeños dictados
20000	80	4000	32	WSJ1 Grandes dictados
60000	200	6000	16	HUB4 Filo de la noticia
60000	2000	12000	64	FISHER RICH Transcripcion telefonica

Fuente: (CMU-Sphinx, 2014)

Por supuesto, también hubo que entender que sólo los senones presentes en la transcripción podrían ser entrenados.

Esto significa que si la transcripción no es lo suficientemente genérico, por ejemplo si la misma palabra es hablada por 10.000 hablantes 10.000 veces, se tiene como resultado sólo un par de senones, sin importar cuántas horas de discurso se hayan grabado. En ese caso, sólo necesita miles de senones en el modelo. **fuente:** (CMU-Sphinx, 2014)

Podría ser que la diversidad pueda mejorar el modelo, pero se tuvo muy en cuenta que la recolección de voces artificiales, no ayudaron en la decodificación de la vida real. Con el fin de construir una mejor base de datos sólida, se necesitó reproducir un entorno real tanto como sea posible. Resultó mucho mejor recoger más discurso reales para tratar de optimizar el tamaño de la base de datos.

Es importante recordar, que los cálculos óptimos dependen mucho de su base de datos. Para el correcto entrenamiento de un modelo, fue necesario experimentar con diferentes valores y tratar de seleccionar el que de cómo resultado el mejor WER para desarrollo. Es posible también experimentar con varios senones, a veces también vale la pena experimentar con un set-fonético o incluso con un número determinado de iteraciones estimadas. **fuente:** (CMU-Sphinx, 2014)

Configurar parámetros de características de sonido

El valor predeterminado para los archivos de sonido utilizados fue de 16 mil muestras por segundo (16 KHz). Si este es el caso, el archivo `etc/feat.params` se genera automáticamente con los valores recomendados. Tasas de muestreo más baja también significó un cambio en los rangos de frecuencia de sonido utilizados, así mismo como el número de filtros que se utilizan para reconocer el habla, así que se debe tener mucho cuidado en utilizar los parámetros. **fuente:** (CMU-Sphinx, 2014)

Configure los parámetros de decodificación

Abrir `etc/sphinx_train.cfg`, asegurando de que está configurado correctamente lo siguiente:

```
$DEC_CFG_DICTIONARY                                          =
"$CFG_BASE_DIR/etc/diccionario.dic";
$DEC_CFG_FILLERDICT     = "$CFG_BASE_DIR/etc/audio.filler";
```

```
$DEC_CFG_LISTOFFILES                                    =
"$CFG_BASE_DIR/etc/audioTest.fileids";
$DEC_CFG_TRANSCRIPTFILE                                 =
"$CFG_BASE_DIR/etc/audioTest.transcription";
$DEC_CFG_RESULT_DIR     = "$CFG_BASE_DIR/result";

# This variables, used by the decoder, have to be user
defined, and
# may affect the decoder output

$DEC_CFG_LANGUAGEMODEL                                  =
"$CFG_BASE_DIR/etc/lenguaje.lm.DMP";
$DEC_CFG_LANGUAGEWEIGHT = "10";
$DEC_CFG_BEAMWIDTH = "1e-80";
$DEC_CFG_WORDBEAM = "1e-40";
```

4.1.4.4. ENTRENAMIENTO

En primer lugar, se fue al directorio de la base de datos:

```
cd data
```

Para entrenar, bastó con ejecutar los siguientes comandos:

```
sphinxtrain run
```

Entonces, este pasó por todas las etapas necesarias. Para mando y control, tomó apenas unos minutos para entrenar, pero para bases de datos suficientemente grandes, el entrenamiento tomó incluso decenas de horas.

Durante todas las etapas, la etapa más importante es la primera, la cual comprueba que todo está configurado correctamente y verifica que los datos de entrada son constantes. **fuente:** (CMU-Sphinx, 2014)

La salida típica durante la decodificación se ve así:

```
Baum welch starting for 2 Gaussian(s), iteration: 3 (1 of 1)
   0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
        Normalization for iteration: 3

 Current Overall Likelihood Per Frame = 30.6558644286942
 Convergence Ratio = 0.633864444461992
 Baum welch starting for 2 Gaussian(s), iteration: 4 (1 of 1)
        0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
```

```
        Normalization for iteration: 4
```

Este script procesa todos los pasos necesarios para entrenar el modelo, he indica que el modelo ha terminado de entrenar.

Scripts empleados, junto con el trabajo realizado tomaron algunos minutos a través de cada compilación.

Antes de ejecutar cualquier script, hubo que tener en cuenta el contenido del directorio actual. Después de ejecutar cada `slave*.pl` tomar nota del contenido de nuevo ya que se crearan varios nuevos directorios, estos directorios contienen archivos que se están generando en el trascurso de su entrenamiento. En este punto no se necesita saber sobre el contenido de tales directorios o archivos, aunque algunos de ellos es posible explorarlos si se lo desea. **fuente:** (CMU-Sphinx, 2014)

Uno de los archivos que aparecieron en el directorio actual es un archivo `.html,` llamado `data.html`, dependiendo de la base de datos que se esté utilizando. Este archivo contiene un informe de estado de los trabajos ya ejecutados. Verifica que el trabajo que se puso en marcha ha finalizado satisfactoriamente. Sólo entonces lanza el siguiente `slave*.pl` en la secuencia especificada. Se repitió este proceso hasta que se hayan ejecutado todos los `slave*.pl` en todos los directorios. **fuente:** (CMU-Sphinx, 2014)

El sistema no funciona directamente con señales acústicas. Las señales se transforman primero en una secuencia de vectores de característicos, que se utilizan en lugar de las señales acústicas reales.

El script `slave_feat.pl` calcula, para cada expresión del entrenamiento, una secuencia de vectores de 13 dimensiones (vectores de características) que consisten en los coeficientes Mel-frequency cepstral (MFCC), los cuales se almacenan automáticamente en un directorio llamado “feat”.

La lista de archivos de señales, contiene una lista con las rutas completas a los archivos de audio semejante. Dado que los datos están ubicados en el mismo directorio que se está trabajando, las rutas son relativas, no absolutas. **fuente:** (CMU-Sphinx, 2014)

Tenga en cuenta que el tipo de cálculo de vectores de las señales de voz para la formación y el reconocimiento, no se limita únicamente a MFCC, porque

se podría utilizar cualquier otra técnica de parametrización razonable en su lugar, y calcular las características distintas de MFCC. CMUSphinx puede utilizar características de cualquier tipo o dimensión. El formato de las características se describe como sigue:

Durante el entrenamiento y decodificación se suelen utilizar características estáticas como mel-cepstrum y características dinámicas como deltas mel-cepstrum. **fuente:** (CMU-Sphinx, 2014)

El cálculo se configuró con la opción -feat la cual va a leer el vector, calcular los deltas y combinarlos con un flujo de características del vector.

El archivo almacena los valores en formato binario y cada valor se almacena como un flotante de 4 bytes, que no es más que el número total de valores almacenados. Así, la cabecera que es básicamente el número total de valores distintos en la matriz característica (N x M). Dónde: **fuente:** (CMU-Sphinx, 2014)

N: número de frames
M: 13 vectores característicos

```
header (int32 length)
features of the frame 1 (13 floats or 13 * 4 bytes)
features of the frame 2 (13 floats or 13 * 4 bytes)
features of the frame ... (13 floats or 13 * 4 bytes)
features of the frame N (13 floats or 13 * 4 bytes)
```

Una vez que los trabajos iniciados desde el directorio `20.ci_hmm` se han ejecutado con éxito hasta el final, significa que se han entrenado los modelos independientes del contexto (IC) para las unidades de sub-palabra en el diccionario.

Cuando los trabajos iniciados desde el directorio `30.cd_hmm_untied` se ejecutaron por completo, significa que se han entrenado a todos los modelos dependientes del contexto de unidades de sub-palabras (trifonos) con estados desligados. Estos son los llamados modelos de CD-untied (Independientes del Contexto - desatados) y son necesarios para la construcción delo árbol de decisiones con el único final de atar estados para formar frases.

Los trabajos en `40.buildtrees` construirán el árbol de decisión para cada estado de cada unidad de sub-palabra. Los trabajos en `45.prunetree` podan los árboles de decisión y atar los estados.

Después de esto, los trabajos en `hmm_tied-50.cd` entrenaron los modelos finales para los trifonos en su corpus de entrenamiento. Estos son los llamados modelos de CD-tied (Dependientes del Contexto - atados). Los modelos de CD-tied son entrenados en muchas etapas. Comenzamos con 1 Gauss por HMMs, tras lo cual entrenamos 2 Gauss por HMMs y así sucesivamente hasta el número deseado de Gaussianas por Estado. Los trabajos en `hmm_tied-50.cd` entrenarán automáticamente todos estos modelos de CD-tied intermedios.

Al final de cualquier etapa puede utilizar los modelos de reconocimiento. Únicamente recuerdese que puede decodificar incluso mientras la formación está en curso, siempre y cuando se esté seguro de que ha cruzado la etapa que genera los modelos que desea decodificar. **fuente:** (CMU-Sphinx, 2014)

Transformación Entrenamiento Matrix (avanzado).- Algunas secuencias de comandos adicionales se pondrán en marcha si decide ejecutarlos. Matrices transformacionales pueden ayudar a la formación y el proceso de reconocimiento en algunas circunstancias.

Los siguientes pasos se ejecutarán en el archivo sphinx_train.cfg , si especifica `$ CFG_LDA_MLLT = 'yes';` Si se especifica `"no"`, por defecto, los pasos no hará nada.

```
# Calculate an LDA/MLLT transform?
$CFG_LDA_MLLT = 'no';
# Dimensionality of LDA/MLLT output
$CFG_LDA_DIMENSION = 29;
```

Los scripts de Perl, a su vez, configurar y ejecutar módulos de python. El producto final de estos pasos es un archivo `feature_transform` , en su directorio `model_parameters.`

Por último, un paso más a considerar es el ejecutará si se especifica la formación MMIE con `$CFG_MMIE = "yes"` o `"no"`

```
# MMIE training related variables
$CFG_MMIE = "no";
```

```
$CFG_MMIE_MAX_ITERATIONS = 5;
$CFG_LATTICE_DIR = "$CFG_BASE_DIR/lattice";
$CFG_MMIE_TYPE   = "rand"; # Valid values are "rand", "best"
or "ci"
$CFG_MMIE_CONSTE = "3.0";
$CFG_NUMLAT_DIR  = "$CFG_BASE_DIR/numlat";
$CFG_DENLAT_DIR  = "$CFG_BASE_DIR/denlat";
```

4.1.4.5. USANDO EL MODELO

Para ello, fue necesaria una etapa de decodificación de prueba. La decodificación es ahora una última etapa del proceso de entrenamiento. Puede reiniciar la decodificación con el siguiente comando:

```
sphinxtrain -s decode run
```

Este comando inició un proceso de decodificación utilizando el modelo acústico ya entrenado y el modelo de lenguaje que se configuró con anterioridad en el directorio `etc/sphinx_train.cfg`. **fuente:** (CMU-Sphinx, 2014)

```
MODULE: DECODE Decoding using models previously trained
        Decoding 130 segments starting at 0 (part 1 of 1)
        0%
```

Después del entrenamiento, el modelo acústico se encuentra en

```
% model_parameters/<your_db_name>.cd_cont_<number_of senones>

model_parameters/<your_db_name>.cd_cont_<number_of senones>
```

Sólo se necesita esa carpeta. El modelo debe tener los siguientes archivos:

```
mdef
feat.params
mixture_weights
means
noisedict
transition_matrices
variances
```

Dependiendo del tipo de modelo entrenado. Para utilizar el modelo con pocketsphinx, simplemente se apuntó a él con la opción –hmm como sigue:

```
% pocketsphinx_continuous -hmm <your_new_model_folder> -lm
your_lm -dict your_dict.

pocketsphinx_continuous –hmm
/tesis/data/out/intento8/model_parameters/data.cd_cont_400
-lm /tesis/data/out/intento8/src/lenguaje.lm.DMP -dict
/home/tesis/Escritorio/tesis/data/out/intento8/src/diccionari
o2.dic -bestpath no
```

4.1.4.6. SOLUCIÓN DE PROBLEMAS

Solucionar un problema, no es ciencia espacial, lo más probable es la razón de la falta. Cuidadosamente cuando se ejecute un error, es mejor leer los mensajes de la carpeta `logdir` que contiene los registros detallados de las acciones realizadas por cada uno. Además, los mensajes se copian en el archivo, `your_project_name.html`, que se pudieron revisar en un navegador si se lo desea. **fuente:** (CMU-Sphinx, 2014)

Hay muchos métodos que funcionen bien probadas para resolver problemas. Por ejemplo, tratar de reducir el conjunto de entrenamiento para ver en qué medio aparece el problema.

Si este fuera el caso, la propia página de CMU-SPHINX proporciona las soluciones y no hace falta más que consultar en dicha página que contiene la solución a errores comunes que se produjeron. **fuente:** (CMU-Sphinx, 2014)

4.1.5. CONSTRUCCIÓN DE UN DICCIONARIO

La construcción del diccionario para el modelo Español-Ecuatoriano, partió desde el archivo `audio.rec` que contiene los directorios actuales de los archivos de audio .wav para cada locutor con sus correspondientes oraciones para el entrenamiento.

```
src/audio.rec
```

Mediante la utilización del comando `sort y uniq` se ordenaron en columna alfabéticamente las palabras y se eliminan las palabras repetidas dando lugar al archivo:

```
work/audio.src
```

Del archivo `audio.src` de deriban los archivos :

```
audio.fileids
audio.usado
wikipedia.usado
audioTest.txt
audioTrain.txt
audioTest.fileids
audioTrain.fileids
```

El archivo `audio.usado` es una base de datos que contiene todas las palabras existentes del entrenamiento para el modelo de lenguaje Español-Ecuatoriano. Así mismo `wikipedia.usado` es la base de datos contenedora de las palabras extraídas de wikiedia para utilizarlas en la creación de nuestro modelo. Si a estos se les extrae todos los tipos existentes de caracteres especiales, entonces tenemos como resultado :

```
work/audio.txt
work/Wikipedia.txt
```

De la unión de estos dos archivos mediante comandos de Script Shell se genera `lenguaje.txt`

```
work/lenguaje.txt
```

La forma de **texto normalizado** de los archivos `audio.txt` y `wikipedia.txt` nos da como resultado los archivos:

```
work/audio.norm
work/Wikipedia.norm
```

Mediante la utilización de los comandos `text2wfreq` y `wfreq2vocab` como ya se los mencionaron con anterioridad, se genera el archivo `directorio.list` el cual contiene una base de datos de todas las palabras del entrenamiento **normalizadas** y ordenadas alfabéticamente. Esto se resume en que el archivo `.list` que se esta generando, es la unión de `audio.norm` y `wikipedia.norm`

```
work/diccionario.list
```

Finalmente y como último paso se genera el diccionario de palabras con su contenido fonético, el cual no es más que el mismo `diccionario.list` pero ahora cada palabra contiene su pronunciación fonética para el reconocimiento de voz.

```
work/diccionario.dic
```

Hay varias herramientas que ayudarán a ampliar un diccionario existente con nuevas palabras si se ha construido un nuevo diccionario a partir de cero. Como en nuestro caso se comenzó un nuevo idioma, es necesario caer en cuenta de las distintas reglas de vocabulario que existen, ya que resulta muy difícil de crearlas precisas para convertir texto a los sonidos. Sin embargo, la práctica demuestra que incluso la conversión ingeniosa podría producir un buen resultado para el reconocimiento de voz. **fuente:** (CMU-Sphinx, 2014)

Por ejemplo, muchos desarrolladores tuvieron éxito para crear ASR (Reconocedor de voz automático de primer nivel) con una simple síntesis basado en grafemas, donde cada letra es sólo asigna a sí mismo y no al fonema correspondiente.

Para la mayoría de las lenguas es necesario utilizar grafema especializado para fonema para hacer la conversión utilizando métodos de aprendizaje automático y pequeña base de datos existente.

Tenga en cuenta que si utiliza TTS’s a menudo es necesario hacer la conversión set-fonético. Un conjunto fonético TTS suelen ser más extensa de lo necesario. **fuente:** (CMU-Sphinx, 2014)

4.2. PRUEBAS E IMPLEMENTACIÓN

Finalmente resulta fundamental realizar las pruebas correspondientes para medir la calidad de la entrenada base de datos, con el fin de seleccionar los mejores parámetros, entender cómo funciona la aplicación y optimizar el rendimiento. La afinación de los parámetros de calidad tanto en Linux con PocketSphix como en Windows con Sphinx-4 resulta de gran importancia para que el sistema de reconocimiento de voz, responda de manera ágil y rápida, con palabras acertadas propias de la creación del modelo de lenguaje Español Ecuatoriano y de esta manera, cumplir con los objetivos propuestos.

4.2.1. PRUEBAS EN LINUX

Cuando el trabajo de recopilación, adaptación de los modelos, entrenamiento y reconocimiento se haya completado, el script calcula el “Word Error Rate” (WER) y “Sentence Error Rate” (SER). Cuanto más baja sea la tasas, el resultado es mucho mejor. Para tareas típicas de reconocimiento de 10 horas, la WER debe estar alrededor del 10%, pero para una tarea mucho más grande, podría llegar al 30%. Para el caso particular del presente proyecto, entrenado con 1150 senones, se calcularon las tasas como sigue:

```
SENTENCE ERROR:30.2% (92/130)   WORD ERROR RATE: 9.98%
```

Pueden encontrarse los detalles exactos de la decodificación, como por ejemplo; la alineación con su transcripción de referencia, la velocidad y el resultado para cada archivo, en la carpeta `result` que se creó después de la decodificación. Para ello, mira en el archivo `an4.align` :

```
Insertions: 0 Deletions: 0 Substitutions: 3
october twenty four nineteen seventy  (MMXG-CEN8-MMXG-B)
october twenty four nineteen seventy  (MMXG-CEN8-MMXG-B)
Words: 5 Correct: 5 Errors: 0 Percent correct = 100.00% Error
= 0.00% Accuracy = 100.00%
Insertions: 0 Deletions: 0 Substitutions: 0
TOTAL Words: 773 Correct: 587 Errors: 234
TOTAL Percent correct = 75.94% Error = 30.27% Accuracy =
69.73%
TOTAL Insertions: 48 Deletions: 15 Substitutions: 171
```

4.2.2. IMPLEMENTACION EN WINDOWS JAVA

Una vez finalizado el entrenamiento utilizando el kit de herramientas Sphinx, se creó una carpeta que contiene el corpus de datos con los modelos adaptados y entrenados. Entonces podemos decir, que ya se ha creado el nuevo modelo para el reconocimiento del idioma Español Ecuatoriano. El nuevo modelo de reconocimiento, fue liberado como una nueva versión para ser utilizada e implementada en Windows. De todo el proceso de creación del nuevo sistema de reconocimiento realizado en la plataforma de Linux, el resultado es una carpeta que contiene el corpus de adaptación y los parámetros del modelo. En Windows, esta carpeta es convertida en una librería con extensión “.jar” y será utilizada en java NetBeans para la implementación de una interface gráfica.

Dicha librería es ahora una herramienta muy poderosa, puesto que llevara a cabo el reconocimiento y predicción de voz a texto, actuando como decodificador para ambos escenarios. Como se puede ver en la figura 27, el modelo de 1150 senones, ahora es una librería que contiene a **etc** y **model_parameters**, resultado del entrenamiento con Sphinx.

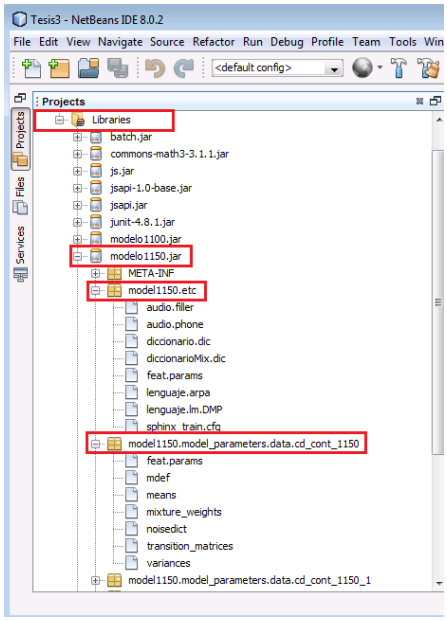


Figura 27. *Nueva versión liberada de reconocimiento de voz español ecuatoriano (model1150.jar)*

4.2.2.1. DENTRO DE ETC

Como muestra claramente la figura 27, dentro de “etc” se encuentran alojadas archivos muy importantes producto del entrenamiento con Sphinx. En modo de un pequeño repaso de Sphinx, describiremos cada archivo de la carpeta “etc”.

Hay que recordar que por cuestiones de asunto de ortografía, es mejor trabajar con textos normalizados, es decir que se debe cambiar todo el texto a mayúsculas, eliminar tanto acentos como signos, e indicar el inicio y fin de cada oración entre los indicadores. Analizamos la siguiente frase.

```
Hola amigos, que tengan buenos días
<s> HOLA AMIGOS QUE TENGAN BUENOS DIAS </s>
```

Posteriormente de cada oración de texto normalizado, se van sacando las palabras y ordenándolas alfabéticamente en columna.

```
AMIGOS
BUENOS
DIAS
HOLA
QUE
TENGAN
```

Dentro de “etc” siempre deben existir dos archivos importantes, **audio.files, audio.phone.**

audio.filer.- Este archivo contiene los relleno de la parte fonética, es decir que dentro de este archivo estarán las partes que no pertenecen al habla. Como por ejemplo el silencio, el ruido de fondo, la respiración, etc.

```
</s>    SIL
<s>     SIL
<sil>   SIL
```

audio.phone.- contiene la lista fonética relativa a la tabla 4, la cual el sistema termina de entender el sonido de cada fonema, una vez que se hayan obtenido y comparado los audios con el texto.

```
A
AA
B
CH
D
E
EA
F
G
I
IA
J
K
L
M
N
NY
O
OA
```

```
P
R
RR
S
T
U
UA
Y
SIL
```

diccionario.- De esta manera una vez que se tiene la tabla fonética definida, se pudo obtener una base de datos como sigue:

```
AMIGOS     A      M      I      G      O      S
BUENOS     B      U      E      N      O      S
DÍAS       D      IA     A      S
HOLA       O      L      A
QUE        K      E
TENGAN     T      E      N      G      A      N
```

Una vez que se ha creado el **diccionario**, se debe de tomar en consideración de que existen algunas palabras que escapan de las reglas del diccionario, pero que sin embargo hemos tratado de rescatar alguna de ellas, como por ejemplo:

```
GOOGLE     G      U      G      O      L
GOOGLE(2)  G      U      G      L      E
GOOGLE(3)  G      O      G      L      E
MIGUEL     M      I      G      E      L
FBI  E     F      E      B      E      I
ACK  A     S      E      K      A
```

Cuando se creó el modelo del lenguaje el sistema comienza a ir separando todo por palabras o n-gramas en donde existen los siguientes archivos indispensables que no pueden faltar dentro de la carpeta “etc” y estos son **lenguaje.arpa** y **lenguaje.DMP**

lenguaje.DMP.- Este archivo de extensión “DMP” es un archivo “arpa” en su forma binaria, la única diferencia es que al estar en un estado binario, el peso de este archivo es mucho menor que el de un archivo de extensión “arpa”, por lo tanto esto quiere decir que su velocidad de procesamiento es mucho más rápida y el tiempo de respuesta obviamente es mínimo.

lenguaje.arpa.- Este archivo contiene los denominados n-grams, con sus respectivas probabilidades de aparición de cada uno. Este archivo no solo trabaja probabilísticamente con una sola palabra, sino que trabaja con 1-grams, dos palabras 2-grams y hasta tres palabras 3-grams. Es importante hacer mención que al trabajar con Sphinx-4 sobre Windows, hay que forzar al sistema a trabajar con una codificación UTF-8 para no tener problemas con la gramática de las palabras.

```
1-GRAM
<s>
-2.3640 AMIGOS -1.0012
-4.3540 BUENOS -0.2218
-3.2580 DÍAS -1.5660
-0.4560 HOLA -1.3600
-1.7890 QUE -0.1230
-2.2580 TENGAN -2.2121
</s>

2-GRAM
<s>
-1.1129 HOLA -1.1235
-3.3695 HOLA AMIGOS -0.2541
-4.7752 AMIGOS QUE -1.2577
-4.3356 QUE TENGAN -1.2335
-5.4455 TENGAN BUENOS -0.2315
-4.6858 BUENOS DÍAS -1.3432
-2.3690 DÍAS -0.0012
</s>

3-GRAM
<s>
-4.2578 HOLA AMIGOS -1.0022
-4.2315 HOLA AMIGOS QUE -1.5632
-5.1235 AMIGOS QUE TENGAN -0.1237
-4.2544 QUE TENGAN BUENOS -1.2568
-4.2585 TENGAN BUENOS DÍAS -0.2357
-4.5656 BUENOS DÍAS -0.2654
</s>
```

feat.params.- Este archivo extrae y define los parámetros de afinamiento para el entrenamiento. Este archivo está enfocado más para la parte del entrenamiento que para el funcionamiento en sí. Lo cual nos lleva al **sphinx_train.cfg.**

```
-nfilt __CFG_NUM_FILT__
```

```
-lowerf __CFG_LO_FILT__
-upperf __CFG_HI_FILT__
-feat __CFG_FEATURE__
-svspec __CFG_SVSPEC__
-agc __CFG_AGC__
-cmn __CFG_CMN__
-varnorm __CFG_VARNORM__
```

sphinx_train.cfg.- Contiene las características impuestas con las que fue entrenado y afinado el sistema, por ejemplo define las frecuencias altas de corte, las frecuencias bajas de corte, el número de filtros que se utilizan, directorios de almacenamiento, numero de senoides, frecuencias de muestreo, diccionarios, modelos, etc.

```
# Feature extraction parameters
$CFG_WAVFILE_SRATE = 16000.0;
$CFG_NUM_FILT = 40; # For wideband speech it's 40, for
telephone 8khz reasonable value is 31
$CFG_LO_FILT = 133.3334; # For telephone 8kHz speech value is
200
$CFG_HI_FILT = 6855.4976; # For telephone 8kHz speech value
is 3500
```

Por ejemplo se pueden observar las frecuencias de muestreo de 16000Hz. Según el criterio de Nyquist, la frecuencia de muestreo debe ser dos veces la frecuencia de la señal más alta, es decir que `HI_FILT` debería ser de 8KHz, pero al tratarse de la utilización de filtros, se debe de tener un rango de guarda, por lo que se considera 6855Hz. Este número de frecuencias se justifica por estar orientados al reconocimiento de la voz.

4.2.2.2. DENTRO DE MODEL_PARAMETERS

Es verdad que de las librerías, se puede observar claramente que existen algunas carpetas contenidas dentro de model_parameter, pero solamente fueron colocadas allí por prevenir errores. La carpeta que nos va a servir es solamente la primera, denominada “model_parameter_data.cd_cont_1150” como muestra la figura 27. Dentro de esta carpeta se encuentras archivos binarios que representan los modelos del speech to text para el reconocimiento y entre los cuales están:

- feat.params
- mdef

- means
- mixture_weights
- noicedict
- transition_matrices
- variance

feat.params.- Dentro este, ya se encuentran los parámetros con los que se va a trabajar:

```
-nfilt 40
-lowerf 133.3334
-upperf 6855.4976
-feat 1s_c_d_dd
-agc none
-cmn current
-varnorm no
```

4.2.2.3. CONCEPTOS IMPORTANTES

Archivo de formato SRT.- El formato nativo para los subtítulos SRT, es de la forma más simple puesto que poseen muy poco formato. Cada aspecto de los subtítulos se guarda en un archivo de texto SRT basado en UTF-8. Ese formato de subtítulos está determinado por el reproductor de medios que utilices con ellos.

Los subtítulos dentro de un archivos SRT se organizan en bloques que representan cada cuadro en el que aparecen. Hay cuatro elementos en cada bloque:

- El orden de los subtítulos
- La indicación del tiempo y la duración
- El subtítulo en sí mismo
- Una línea en blanco para separarlo del siguiente subtítulo.

Por ejemplo, los dos primeros subtítulos en un vídeo podrían ser:

```
1
00:00:06,780 --> 00:00:09,259
¿Quién eres? ¿Me refiero a qué eres?
```

```
2
00:00:10,840 --> 00:00:13,590
He venido del futuro y estoy aquí para llevarte
```

Esto especifica que el primer subtítulo comienza casi siete segundos de un video, y tiene una duración de poco más de un segundo, el siguiente comienza casi inmediatamente después.

La marca de tiempo por encima de un subtítulo es importante, ya que le dice a su reproductor de medios cuándo mostrar el subtítulo y por cuánto tiempo. El momento es preciso, hasta la milésima de segundo. El formato básico pasa por la hora, el minuto, el segundo y la milésima de segundo, o hh:mm:ss, zzz. La flecha indica la duración y la segunda fecha denota cuándo se deben dejar de mostrar los subtítulos.

Reproductor VLC Media Player.- Es un reproductor portable y multiplataforma, con versiones para Linux, Mac, Microsoft Windows, Solaris, iOS y Android, entre otros. Entre sus características se tiene:

- Soporta un gran número de formatos de audio y vídeo sin necesidad de instalar códecs adicionales: MPEG-1, MPEG-2, MPEG-4, DivX, MP3, OGG, MOV, RAM, AVI, FLV, etc.
- Es una opción muy interesante frente a otros programas comerciales para reproducir CDs de música, películas en soporte DVD o VídeoCD, etc.
- Se puede utilizar como servidor de streaming en una red local o de banda ancha.

VLC Media Player es una aplicación local para reproducir archivos multimedia del disco duro. Este programa, que a simple vista parece no ser el mejor reproductor, pero que es una poderosa herramienta de reproducción, añade una característica muy importante de cargar archivos de formato “srt” para sub-titulación de videos.

4.2.2.4. IMPLEMENTACION MENÚ

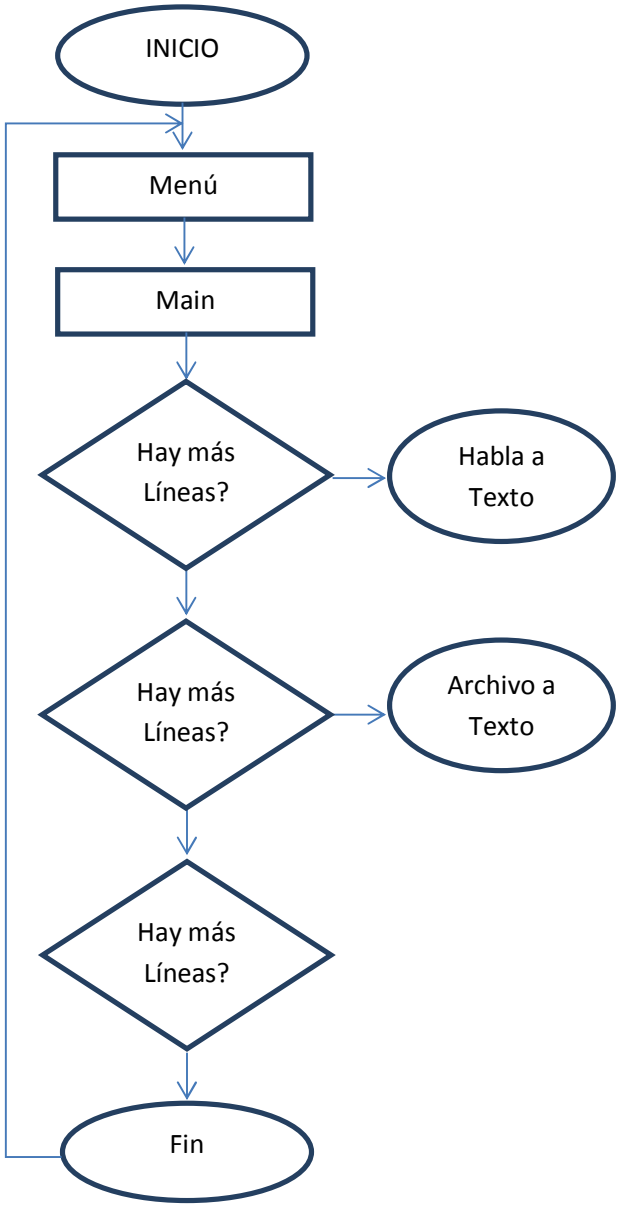


Figura 28. *Menú del Nuevo Sistema de Conversión. Elegir la fuente de conversión*

4.2.2.5. IMPLEMENTACION PRIMER ESCENARIO

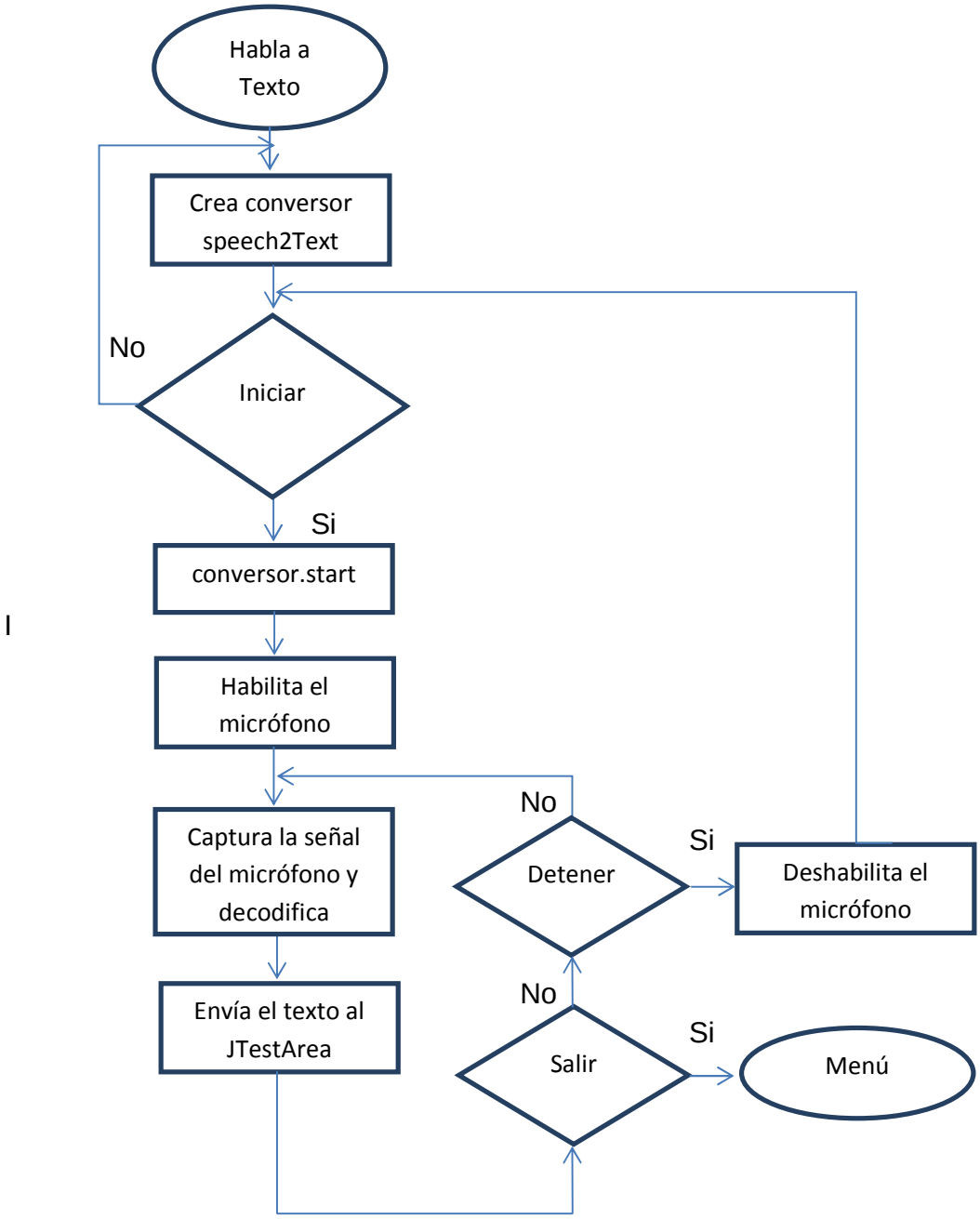


Figura 29.*Implementacion Primer Escenario de Conversión Speech to Text*

4.2.2.6. IMPLEMENTACION SEGUNDO ESCENARIO

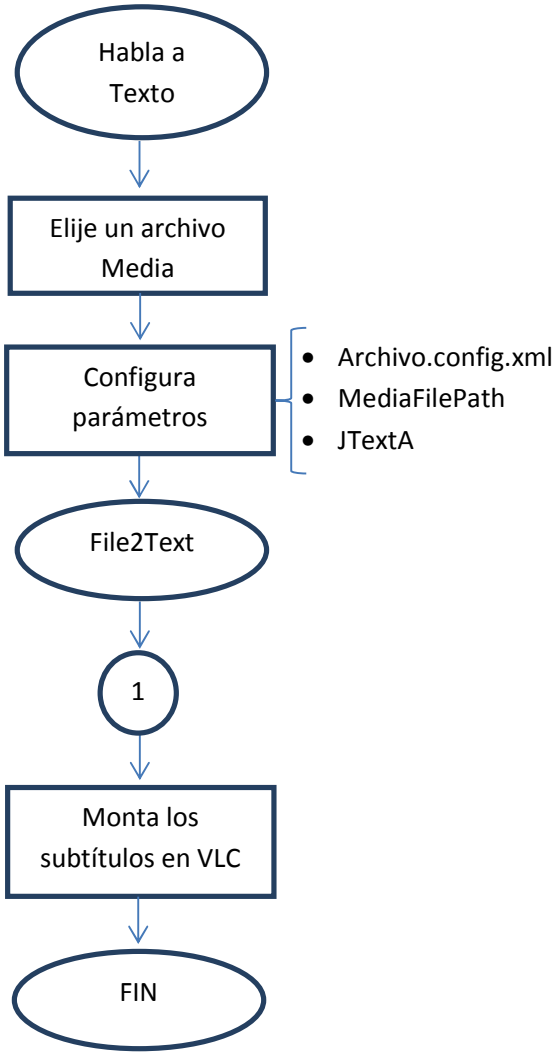


Figura 30. *Implementación Segundo Escenario de Conversión File to Text*

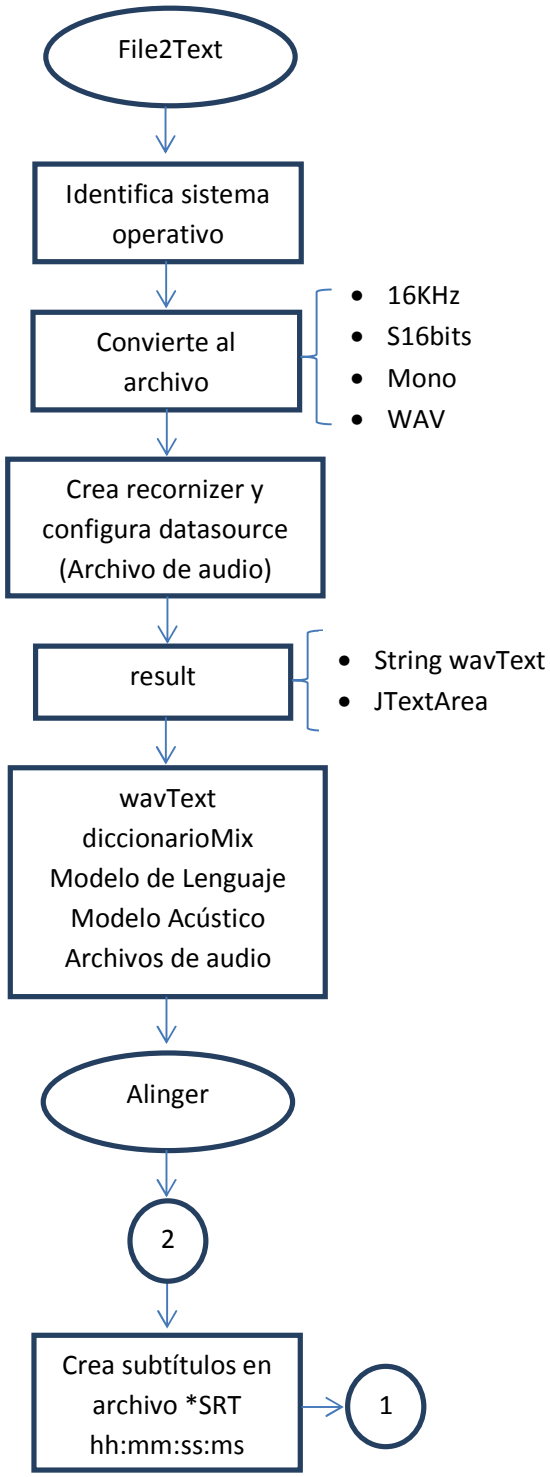


Figura 31. *Configuración del Traductor a Texto File2Text*

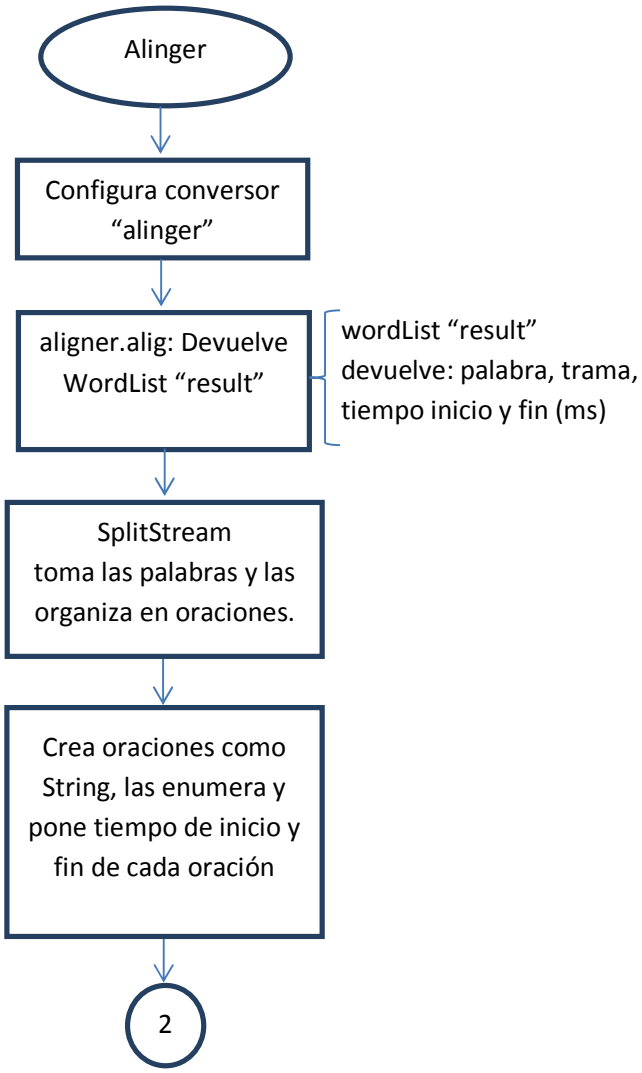


Figura 32. *Configuración del Conversor Alinger*

4.2.2.7. PRUEBAS EN WINDOWS

Esta vez fue el turno de utilizar Sphinx 4 para actuar y mostrar todo su potencial. Sphinx 4 se desarrolla enteramente en el lenguaje de programación Java^{MT} y es por lo tanto que se debe mencionar que es muy portátil para utilizarlo sobre cualquier sistema operativo. Ademas Sphinx 4 permite usar multithreading (capacidad de un programa o un sistema operativo para servir a más de un usuario a la vez y de manejar múltiples peticiones simultáneas sin la necesidad de tener múltiples copias de los programas que se ejecutan en el ordenador.) y permisos altamente flexible para la interfaz de usuario.

El sistema de reconocimiento de voz Sphinx4 es la última incorporación al repositorio de los sistemas de reconocimiento de voz. Sphinx 4 es diferente de los sistemas de CMUSphinx anteriores en términos de modularidad, flexibilidad y aspectos algorítmicos. Utiliza nuevas estrategias de búsqueda, es universal en su aceptación de las distintas clases de gramáticas y modelos de lenguaje, tipos de modelos acústicos y flujos de características.

La velocidad de respuesta con la utilización de Sphinx-4 obviamente supera las características de rendimiento ya observadas con PocketSphinx, de hecho con la utilización de Sphinx-4 se pudo observar resultados cercanos al tiempo real. Para lograr este objetivo fue necesaria la configuración de los parámetros para el afinamiento del sistema.

El afinamiento consiste en jugar con los parámetros de threshold el cual define un umbral de reconocimiento de voz y discriminación del ruido de fondo. Es decir que se escogió el mejor valor, el cual más se ajusta a una perfecta codificación de las palabras mencionadas y a un mejor tiempo de respuesta cercana al tiempo real.

Se tuvo en consideración que el locutor al mencionar un conjunto de oración por el micrófono, el ruido de fondo debe de quedar por debajo del umbral o threshold, mientras que las palabras mencionadas que superen el umbral, serán las decodificadas por Sphinx-4. La figura 33 puede describir que para esto, se tuvo en consideración dos escenarios muy importantes::

- **No dejar el umbral muy abajo.-** de otro modo el ruido de fondo también seria decodificado junto con las palabras mencionadas por el locutor.
- **No dejar el umbral muy arriba.-** porque de otro modo todo lo que se encuentre abajo del umbral sería considerado como ruido de fondo y no existiría decodificación.

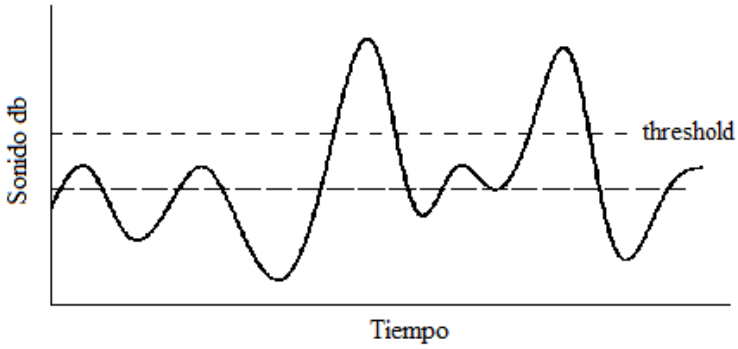


Figura 33. *Configuración del Umbral Threshold para la codificación de las palabras y el afinamiento del sistema*

En resumen, se puede decir que hubo que calibrar los parámetros de theshold ajustando sus valores para escoger el mejor resultado, de tal manera de ir afinando el sistema para que este responda de una manera fluida y rápida con palabras acertadas. El mejor parámetro de threshold fue escofido según las respuestas obtenidas en la tabla 6.

Tabla 6
Configuración de los parámetros de Theshold para la calibración de la decodificación.

	Threshold 1150				
	Acc	**Err**	**WER**	**Proc1**	**Proc2**
1	77.812	1189	22.98	7094.41	7094.41
2	77.773	1193	23.058	28942.76	7416.55
3	77.561	1203	23.251	11323.02	6310.28
4	77.542	1207	23.328	7115.27	7115.27
5	77.232	1223	23.637	6591.37	6591.37
6	77.116	1228	23.734	5527.4	5527.4
7	77.136	1225	23.676	4862.08	4862.08
8	77.039	1230	23.773	5433.53	5433.53
9	76.788	1243	24.024	5480.71	5480.71
10	76.644	1255	24.182	7160.75	7115.27
11	76.501	1263	23.929	5916.73	6591.37
12	76.398	1270	24.489	6227.42	5527.4

Obsérvese el mejor escogimiento de los parámetros de este filtro, por llamarlo así, el afinamiento del sistema se puede observar claramente en la tabla 6 cuyos datos hablan por sí solos.

Para la implementación del segundo escenario, por default el link de CMU-Sphinx, como un parámetro establecido de threshold recomienda utilizar un umbral de 13, sin embargo se fue probando el sistema cambiando este valor, y se llegó a la conclusión de que utilizar un threshold de 11 resulta mucho más aceptable en cuanto a las respuestas obtenidas del sistema.

Pero no solamente los parámetros de Threshold y Sonoides fueron tomados en cuanta para calibración y el afinamiento del sistema, varias tablas fueron creadas y analizadas para escoger el valor que mejor se ajuste a los resultados, posteriormente fueron probados y de esta manera se pudo obtener los resultados deseados para cumplir con los objetivos del presente trabajo. Asi por ejemplo se tienen la configuración de “WordInsertionProbability” como se observa en la tabla 7.

Tabla 7
Configuración del parámetro WordInsertionProbability

	Probabilidad de Inserción de Palabra 1150				
	Acc	**Err**	**WER**	**Proc1**	**Proc2**
1	77.503	1209	23.367	7028.32	7028.32
1.00E-010	69.115	1616	31.233	27962.24	6420.44
1.00E-020	56.146	2281	44.086	10581.15	5324.44
1.00E-030	39.409	3144	60.765	4135.61	4135.61
1.00E-040	24.797	3894	75.261	3267.76	3267.76
1.00E-050	12.737	4516	87.283	2339.47	2339.47
1.00E-060	5.276	4901	94.724	1469.06	1469.06
1.00E-070	1.546	5094	98.454	797.57	797.57
1.00E-080	0.367	5155	99.633	376.21	376.21
1.00E-090	0	5174	100	153.71	153.71
1.00E-100	0	5174	100	127.6	127.6

De esta manera se observa que la mejor opción es escoger un parámetro cercano a 1, puesto que tanto la WER como el grado de error son bajos y a pesar de no tener un buen tiempo de procesamiento, es el parámetro más aceptable en cuanto a precisión y acertación de las palabras.

Si no se está consiguiendo los resultados esperados, se siguió intentándolo y tratando de corregir los errores en el archivo de configuración y haciendo pequeños informes de rendimiento.

Si aún esto no satisface con los resultados, lo que se hizo fue abrir el archivo de configuración “xml” y empezar a jugar con los parámetros y seguir experimentando. Obviamente, se aseguro de hacer notas en cada configuración y sus salidas.

El archivo de configuración “xml” viene con sus valores de parámetros por defecto del entrenamiento con Sphinx utilizando PocketSphinx. Sin embargo, estos parámetros fueron debidamente ajustados de tal manera de ir calibrando el sistema para que este responda de una manera acertada y fluida con Sphinx4.

Entonces según las tablas contenidas dentro de los anexos, se definen en la tabla 8 los siguientes parámetros para la calibración y afinamiento del sistema para cada escenario.

Tabla 8.
Configuración de los parámetros para la calibración del Sistema

Calibración Parámetros		
Parámetros	**Esc S2T**	**Esc F2T**
wordInsertionProbability	1	0.7
languageWeight	7	6
Threshold	11	3
silenceInsertionProbability	0.1	0.1
unigramSmearWeight	1	1
growSkipInterval	0	0

4.2.2.8. ANALISIS DE RESULTADOS

Consumo en espacio de memoria del sistema.- Este es un sistema totalmente accesible de muy bajo costo, puesto que su implementación se realizó únicamente con la inversión del conocimiento. Su instalador tiene un espacio en disco que no supera las 100MB es como se observa en la figura 34.

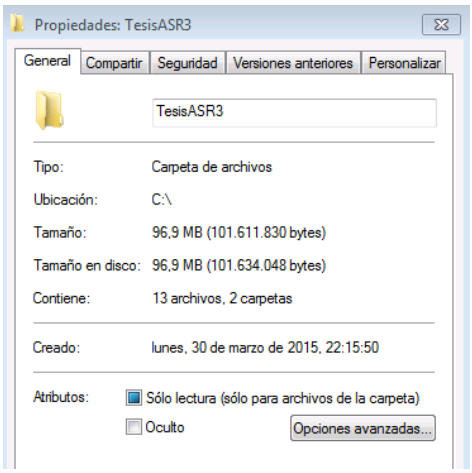


Figura 34. *Espacio de Memoria que ocupa el Nuevo Sistema de Reconocimiento*

Es muy importante mencionar que esta carpeta debe de quedar guardada bajo del directorio de C:/. Dentro de dicha carpeta se encontrara el ejecutable, que bien puede estar en el escritorio como acceso directo. Este ejecutable denominado pesa 1,92KB como muestra la figura 35.

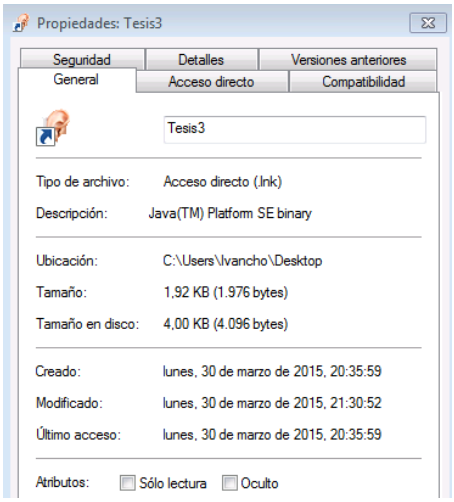


Figura 35. *Espacio de Memoria del Ejecutable*

Consumo de memoria RAM.- Primera mente en esta parte hay que mencionar que; el nuevo sistema de conversión de audio a texto en tiempo real, fue probado en dos diferentes computadoras para observar de mejor manera que sucede con el desempeño del mismo:

La Primera computadora tiene las siguientes características: Intel Corei3, Windows 7, Memoria instalada (RAM) 3GB, Sistema Operativo 64bits, java 7 32bits.

En donde el consumo de memoria RAM para el primer escenario oscila entre 25% a 30% como muestra la figura 36.

Para el segundo escenario, el consumo de memoria RAM esta entre los 35% a 40% como muestra la figura 37.

Pero en este primer caso, el consumo de memoria RAM fue asignado de forma manual con 1024MB en el proceso de instalación como se verá más adelante. El resultado permitió al procesador trabajar de manera óptima sin desbordar los parámetros en la decodificación, teniendo como resultado respuestas acertadas, fluidas y en tiempo real para el primer escenario. Para el segundo escenario muestra un grado de acertamiento del 94%.

Hay que destacar la participación de Java 7 sobre esta primera prueba. Java 7 fue quien permitió añadir manualmente el espacio de memoria, y al tratarse de un Java de 32bits, tuvo mejor respuesta en tiempo y en precisión que usando Java 8.

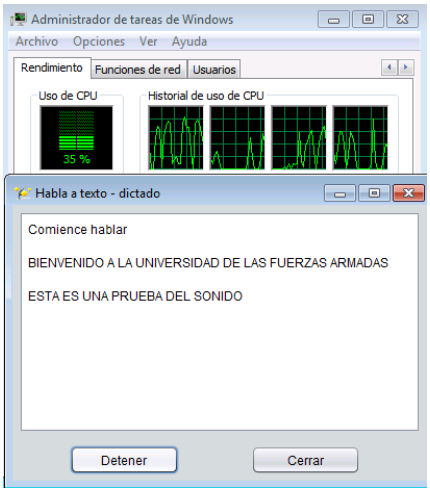


Figura 36*. Procesamiento Primer Escenario Corei3*

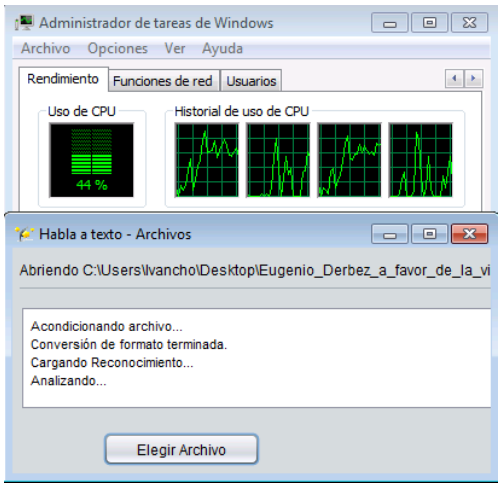


Figura 37. *Procesamiento Segundo Escenario Corei3*

La segunda computadora donde fue probada tiene las siguientes características: Intel Corei7, Windows 8, Memoria instalada (RAM) 8GB, Sistema Operativo 64bits, java 8 64bits.

En donde el consumo de memoria RAM para el primer escenario oscila entre 15% a 20% como muestra la figura 38.

Para el segundo escenario, el consumo de memoria RAM esta entre los 25% a 30% como muestra la figura 39.

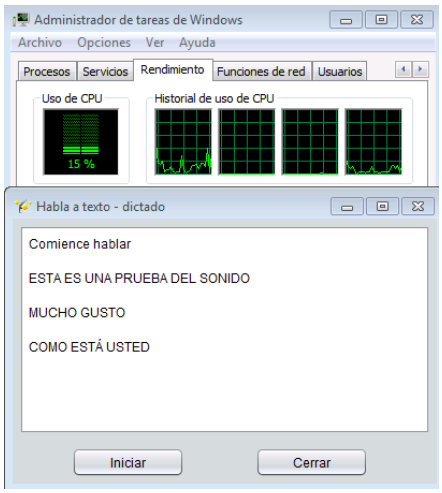


Figura 38. *Procesamiento Primer Escenario Corei7*

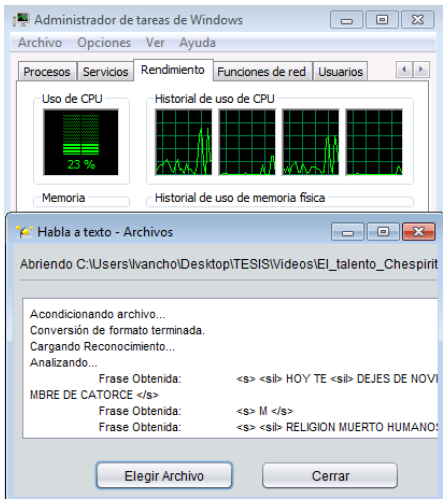


Figura 39. *Procesamiento Segundo Escenario Corei7*

Para este segundo caso, el consumo de memoria RAM fue asignado dinámicamente por Java 8. Como resultado, el procesador trabajo de manera óptima sin desbordar los parámetros en la decodificación, teniendo en consideración que en este caso el equipo tiene 8GB de memoria RAM. Como resultado se obtuvo respuestas acertadas, fluidas y en tiempo real para el primer escenario. Para el segundo escenario muestra un grado de acertamiento del 98%.

Hay que destacar la participación de Java 8 sobre esta segunda prueba. Java 8 permite añadir dinámicamente el espacio de memoria, y al tratarse de un Java de 64bits, interactuando en un procesador Corei5 de 64 bits, tuvo mejor respuesta en tiempo y en precisión que usando Java 7.

Velocidad de respuesta.- el presente sistema de reconocimiento al estar implementado sobre una plataforma independiente del sistema operativo como es java, se pudo observar que la velocidad de respuesta depende mucho tanto de las características del equipo, como del sistema operativo como tal sobre el cual se esté operando el sistema, por lo que no es una plataforma tan independiente, sino que depende del tiempo de procesamiento resultado que se puede medir por ejemplo como en la tabla 7.

Un tiempo estimado de respuesta para cada escenario fueron:

Para el primer escenario en las peores condiciones 1segundo y en el mejor de los casos 0.6 segundos

Para el segundo escenario en las peores condiciones un video de 10 minutos tarde en cargar 3 minutos y en las mejores condiciones 1 mnuto.

4.2.2.9. PROCESO DE INSTALACION DEL NUEVO SISTEMA DE RECONOCIMIENTO DE VOZ ESPAÑOL ECUATORIANO

Para la instalación del nuevo sistema de reconocimiento de voz en tiempo real, hay que seguir una serie de pasos para la correcta configuración:

1. Lo primero que se debe hacer es chequear las variables de entorno y verificar que efectivamente ésta vlc-player. Este paso es muy importante puesto que java llama a vlc como su reproductor de cabecera para el subtitulamiento, entonces se hace lo que sigue:

- Clic derecho en equipo y se selecciona la opción de “Propiedades”
- Se abrirá una ventana donde se debe buscar la opción “Configuración avanzada del sistema”
- Se escoge la opción “Variables de entorno”
- Por ultimo verifíquese que la ruta de vlc está sobre la variable Phat

Las figuras 40 y 41 muestran lo anteriormente descrito.

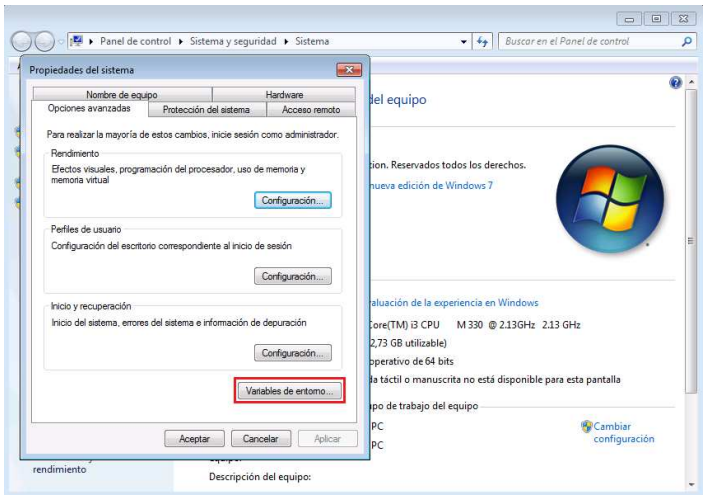


Figura 40. *Configuración avanzada del sistema - Variables de entorno*

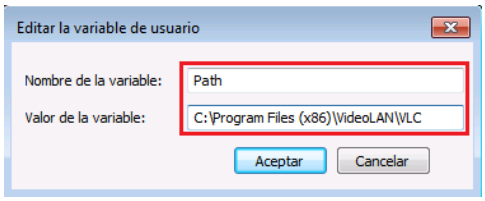


Figura 41. *Configuración Variables de entorno - Phat*

2. Al tratarse de un ejecutable de java, no es necesario tener a mano todo el proyecto en NetBeans, basta con copiar en "C:/" la carpeta "dist" que contendrá las librerías y el ejecutable solamente. Cambiamos el nombre a la carpeta "dist" por "TesisASR", de tal modo que el nuevo sistema de reconocimiento en tiempo real, queda bajo el directorio:

```
"C:/TesisASR/Tesis.jar"
```

3. Crear acceso directo del ejecutable y luego pegarlo en el escritorio sería la mejor opción para no perder de vista el icono y utilizar el nuevo sistema de reconocimiento cuando se lo desee. Una vez hecho esto, se debe de configurar las propiedades del sistema dando clic derecho sobre el icono y luego eligiendo "Propiedades"

Como muestra la figura 42, dentro de "Destino" se debe realizar algunas configuraciones:

- Direccionar javaw.exe para indicar la utilización de la máquina virtual de java jre
- Utilizar e del encoding de Windows UTF-8 para evitar retardos y errores
- Asignar un espacio de memoria RAM para optimizar recursos y dar mayor desempeño al sistema
- Direccionar el ejecutable dentro TesisASR en C:/

```
"C:\Program Files (x86)\Java\jre7\bin\javaw.exe" -
Dfile.encoding=UTF-8 -XX:MaxPermSize=512m -cp
"C:\TesisASR2\Tesis3.jar" project.frontend.Caratula
```

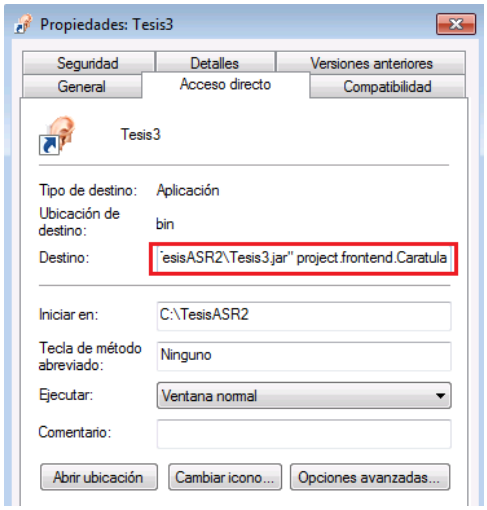


Figura 42. *Configuración de directorios, parámetros y asignación de memoria*

4. Entonces finalmente se verá implementado nuestro nuevo sistema de reconocimiento de voz en tiempo real con palabras propias del idioma español ecuatoriano, tal y como se muestra en la figura 43.

Figura 43. *Implementación del nuevo Sistema de Reconocimiento de Voz Español Ecuatoriano*

Enseguida que se dé clic sobre el icono del proyecto, se abre una pequeña ventana como se mira en la figura 44, la cual nos muestra que podemos elegir cualquiera de los dos escenarios propuestos, ya sea el de conversión de audio a texto o el de conversión de audio cargando un archivo de video.

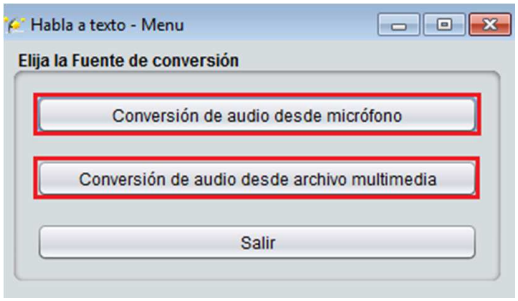


Figura 44. *Ventana con los dos escenarios propuestos a elegir*

Al escoger el primer escenario de “Conversión de audio desde micrófono” se abrirá una pequeña ventana en la cual es suficiente con dar clic en “iniciar” e inmediatamente comenzara a predecir en pantalla todo lo que se diga por el micrófono. Para muestra un botón, la figura 45 indica un claro ejemplo de cómo esta funciona el primer escenario de reconocimiento de voz a texto. Nótese que en la interface gráfica existe un scroll para mostrar un historial de todo lo dicho.

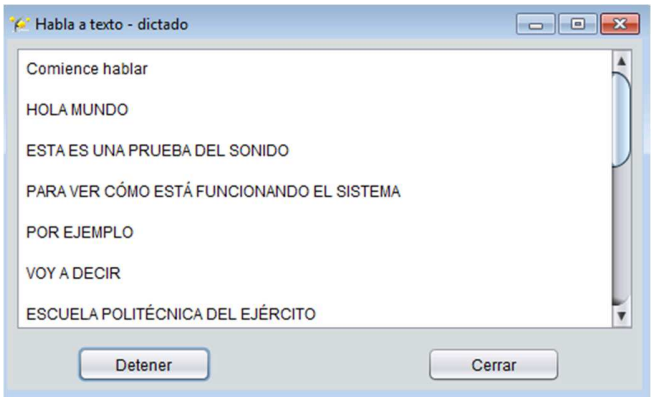


Figura 45. *Presentación Primer Escenario Speech to Text*

Al elegir la opción del segundo escenario de “Conversión de audio desde archivo multimedia” primero aparecerá una pantalla que nos permitirá elegir el archivo de video que se desee cargar para el sub-titulamiento del mismo.

Cuando se haya elegido un video sucederá lo siguiente

- Acondiciona los archivos
- Extrae únicamente el sonido del archivo de video
- Convierte el audio a un formato específico para el reconocimiento
- Extrae las frases del audio del video
- Coloca las frases en un archivo SRT
- Sincroniza el tiempo de video con las frases y las monta sobre VLC-Player

Las figuras 46 y 47 nos muestran como esto sucede:

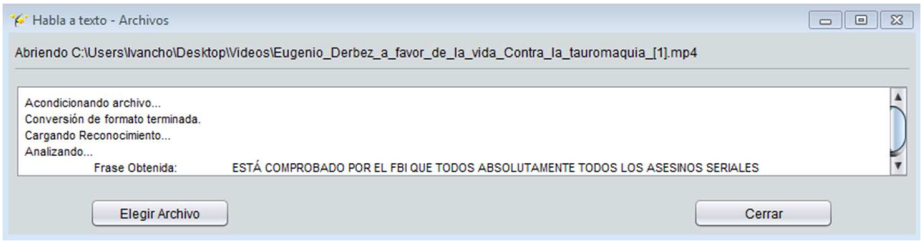


Figura 46. *Segundo escenario - Elección de video a subtitular*

Figura 47. *Presentación Segundo Escenario Speech to Text*

CAPÍTULO V
CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

- Fue posible diseñar un sistema de reconocimiento de habla en tiempo real y adaptarlo para su uso por parte de personas con discapacidad auditiva.

- Se constató la existencia de varias soluciones de reconocimiento de habla, tanto comerciales como gratuitas y con aplicaciones tanto de dictado como de mando para el pc.

- El sistema usa un diccionario de pronunciación, un modelo de lenguaje basado en 3-gramas, creado tanto con oraciones-frases descargadas de wikipedia así como de las usadas para el modelo acústico, y un modelo acústico creado a partir de las grabaciones.

- Se logró adecuar el sistema tanto para filtrar las bandas de frecuencia importantes para una señal de voz de conversación, como para sus niveles de amplitud (threshold) mediante un archivo de configuración xml.

- El sistema está creado sobre el core de Sphinx-4, que es un framework que permite crear aplicaciones de reconocimiento de habla y que facilita las labores de PDS, segmentación de palabras e integración de la información del lenguaje en tres líneas de funcionamiento: diccionario, probabilidad de aparición y secuencia de palabras y analisis espectral de palabras.

- Para hacer el sistema más robusto se debe grabar diferentes frecuencias de voces, tales como hombres, mujeres, niños y ancianos, y también con diferentes tipos de acentos y tonalidades.

- Si grabamos las mismas oraciones utilizando diferentes locutores, el sistema se volverá más independiente del interlocutor. Si alimentamos al sistema con más oraciones, este tendrá más versatilidad de vocabulario.

- El nuevo sistema reconocedor de voz que se obtuvo en este trabajo es propio del idioma español con acento ecuatoriano para habla continua, es independiente del hablante, es decir que el número de usuarios es ilimitado.

- Este sistema de reconocimiento de voz, no toma en cuenta el contexto cultural, no conoce el significado de lo que el hablante dice y sólo se limitan a interpretar las palabras desde un punto de vista espectral (acústico) y gramatical - probabilístico.

- En estos casos, también hay que tener en cuenta el tipo de micrófono, la calidad del canal de transmisión, el eco, el ruido, la distancia a la que la persona sostiene el micrófono, etc.

- El sistema es dependiente del equipo (computador) donde se ejecuta, lo que se notó en la etapa de pruebas de manera empírica. Sin embargo no se realizó una cuantificación entre varios sistemas.

- Los interfaces vocales aportan a los entornos virtuales un nuevo medio para interactuar con el entorno. El reconocimiento de voz es un campo de investigación muy amplio y complejo, con muchos problemas por resolver y muchas cosas que mejorar.

- La cantidad de senoides obtenidos para el modelo es directamente dependiente de la cantidad de oraciones – frases (utterances) con que se alimenta el sistema, e indirectamente de parámetros externos como ruido ambiental. A mas senoides, más general es el modelo.

5.2. RECOMENDACIONES

- Se recomienda utilizar una codificación UTF-8 en el sistema operativo de Windows, previo a la utilización del sistema, para la correcta codificación de las palabras

- Se recomienda ajustar los parámetros de funcionamiento (xml) para cada modelo creado.

- El tema del presente trabajo, se lo deja con las puertas abiertas para próximos temas de investigación, por ejemplo para implementar sistemas de reconocimiento de voz y video en Smart Phones. la domótica, seguridad, etc.

Bibliografía

Aparicio Paniagua, V. (2012). *Desarrollo de una plataforma para la grabación y análisis de somniloquias: una aproximación usando técnicas de análisis de voz.* Madrid: Universidad Carlos III.

CMU-Sphinx. (2014). Obtenido de http://cmusphinx.sourceforge.net/wiki/

de Castro, C., & Morales, C. (2006). *Tecnología para el apoyo a personas con discapacidades.* Chile: Interface hombre máquina.

Fundacion Vivir la Sordera. (2010). Obtenido de http://smart-track.info/vivir-sordera/SorderaenelEcuadorHoy/Situaci%C3%B3n/Adultossordos.aspx

Fundación Vivir la Sordera. (2010). *Situación de los niños sordos en Ecuador.* Obtenido de http://www.vivirlasordera.org.ec/vivir-sordera/SorderaenelEcuadorHoy/Situaci%C3%B3n/Ni%C3%B1ossordos.aspx

Fundación Vivir la Sordera. (2010). *Sordeta en Ecuador.* Obtenido de http://smart-track.info/vivir-sordera/SorderaenelEcuadorHoy.aspx

Jaramillo, P. (2010). *Nuevas Tecnologías, Derechos y Transformaciones Sociales.* Obtenido de http://www.monografias.com/trabajos82/impacto-e-incidencias-tic/impacto-e-incidencias-tic2.shtml

Koon, R., & Vega, E. (2000). *El impacto Tecnológico en las personas con discapacidad.* Córdoba: Articulo del Congreso CIIEE.

Martin, E., & Diego, C. (1998). *TICs Aplicadas a las discapacidades motoras.* Barcelona: Software para discapacitados.

Miranda de Lara, R. (2007). *Discapacidad y Accesibilidad.* Madrid: Fundación Orange 7Ed.

Pérez, P. (2007). *Construcción de un reconocedor de voz utilizando Sphinx y el Corpus DIMEx100.* Mexico DF: Universidad Nacional Autonoma de Mexico.

Picajoso, P. (2010). *reconocimiento de voz en Linux*. Obtenido de http://www.muylinux.com/2010/07/22/sphinx-reconocimiento-de-voz-en-linux

Plan Nacional del Buen Vivir. (2013). Obtenido de http://www.buenvivir.gob.ec/objetivos-nacionales-para-el-buen-vivir

Platero, C. (2006). *Introducción al Procesamiento digital de señales.* Madrid: Universidad Politecnica de Madrid.

Reyes, J. (2005). *Unix Linux*. Obtenido de http://www.maginvent.org/articles/linuxmm/Ventajas_Linux.html

Sánchez Montoya, R. (1998). *Ordenador y discapacidad.* Madrid: CEPE.

Silvestre, T. (2010). *TICs y el Impacto Tecnológico en las personas con discapacidad.* Tecnología de la informacion y la comunicación para la inclusión.

Solar Perez, V. (2008). *El uso de las TIC (Tecnologías de la Información y la Comunicación) como herramienta didáctica en la escuela.* Obtenido de www.eumed.net/rev/cccss/02/vsp.htm

ACTA DE ENTREGA

El proyecto fue entregado al Departamento de Eléctrica y Electrónica y reposa en la Universidad de las Fuerzas Armadas – ESPE desde:

Sangolquí, 04 de Mayo de 2015

ELABORADO POR:

Luis Iván Orbe Orozco
172175052-7

Víctor Julio Zurita Ullauri
172118276-2

AUTORIDAD:

Ing. Paul Bernal

DIRECTOR DE LA CARRERA DE ELÉCTRICA Y ELECTRÓNICA

TELECOMUNICACIONES