UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE CUENCA

CARRERA: INGENIERÍA ELECTRÓNICA

ROBOT AUTÓNOMO DE SOPORTE Y ASISTENCIA PARA NIÑOS CON DISCAPACIDADES O DESORDENES DE LA COMUNICACIÓN

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TITULO DE:

INGENIERO ELECTRÓNICO

Autores:

Ochoa Guaraca Mario Esteban

Carpio Moreta Marco Antonio

Tutor:

Ing. Vladimir Espartaco Robles Bykbaev

Cuenca, Marzo de 2016

Resumen

Según los últimos informes de la Organización Mundial de la Salud y el Banco Mundial, en la actualidad existe una falta de personal, herramientas tecnológicas, estructuras adecuadas y programas orientados a proporcionar servicios para el cuidado de la salud y la rehabilitación para niños con desórdenes de la comunicación. Esta situación es aún más compleja en los países en vías desarrollo (África o América Latina), donde un fonoaudiólogo debe llevar a cabo varias actividades como el seguimiento de los pacientes, la preparación de informes, la prestación de servicios de asesoramiento, la planificación de las terapias, y otros muchos más. Por estos motivos, en este trabajo se presenta un nuevo enfoque que se basa en un asistente robótico para brindar soporte a la terapia de lenguaje.

La estructura que describe de forma detallada el proceso de diseño y construcción del robot se organiza de la siguiente forma: se empieza con una breve reseña acerca de la realidad actual en cuanto a las discapacidades en el Ecuador y los trabajos enfocados al uso de agentes robóticos en la terapia. En la segunda parte se presenta el marco metodológico en donde se detalla cada etapa de la construcción del prototipo robótico, teniendo como las principales a las siguientes: el diseño y construcción de la estructura física y de la placa electrónica, el desarrollo de los módulos de interacción y de la aplicación enfocada a la terapia de lenguaje, las pruebas tanto de laboratorio como de campo con sus respectivos resultados y análisis. Como parte final se presentan las conclusiones que se obtuvieron al desarrollar el presente trabajo y algunas sugerencias sobre las posibles implementaciones que se puedan dar en trabajos futuros.

Gracias a las nuevas tecnologías existentes, se ha logrado obtener un prototipo con funciones innovadoras y a un relativo bajo costo, en cuanto al cerebro del robot se ha usado un «Raspberry Pi 2 Model B» que cuenta con 1GB de memoria RAM y un procesador de 4 núcleos, lo que permite implementar funciones como: visión artificial empleando la «Raspicam» (una cámara propia de Raspberry) e interacción humano-robot mediante una pantalla de 7 pulgadas con interfaz táctil. Por otra parte, se ha utilizado la tecnología de impresión 3D para crear la estructura física del robot, por lo que el trabajo se vuelca directamente en el diseño del mismo, incluso para la alimentación del robot se ha usado bancos de baterías una tecnología emergente.

El asistente robótico implementa varios ejercicios, actividades y juegos que se pueden realizar durante una sesión de terapia, y es capaz de registrar el proceso de interacción con cada paciente. Algunas de sus características más relevantes son los siguientes: posee diferentes trajes que permiten representar animales o personas, implementa procesos basados en visión artificial, reconocimiento automático del habla, conversión de texto a voz, y es capaz de proporcionar varios servicios que incluyen control a distancia y un modo de funcionamiento autónomo. Nuestro enfoque ha sido diseñado con el apoyo de un equipo de expertos en el área de educación especial y fonoaudiología, y ha obtenido resultados prometedores en pruebas controladas de laboratorio.

Cesión de derechos de los autores

Nosotros Mario Esteban Ochoa Guaraca y Marco Antonio Carpio Moreta, manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de titulación intitulado: "*Robot autónomo de soporte y asistencia para niños con discapacidades o desórdenes de la comunicación*", mismo que ha sido desarrollado para optar por el titulo de: *Ingeniero Electrónico con mención en Sistemas Industriales*, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en nuestra condición de autores nos reservamos los derechos morales de la obra antes citada. En concordancia, suscribimos este documento en el momento que hacemos entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Politécnica Salesiana.

Cuenca, 15 de marzo de 2016

Mario Esteban Ochoa Guaraca	Marco Antonio Carpio Moreta
CI: 010430664-2	CI: 010448925-7

Declaratoria de coautoría del docente tutor

Yo, Vladimir Espartaco Robles Bykbaev declaro que bajo mi dirección y asesoría fue desarrollado el trabajo de titulación "*Robot autónomo de soporte y asistencia para niños con discapacidades o desórdenes de la comunicación*" realizado por Mario Esteban Ochoa Guaraca y Marco Antonio Carpio Moreta, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana para ser considerados como trabajo final de titulación.

Cuenca, 15 de marzo de 2016

Ing. Vladimir Espartaco Robles Bykbaev

CI: 0300991817

Tutor del trabajo de titulación

Agradecimientos

Queremos expresar nuestro más sincero agradecimiento a nuestro tutor de proyecto técnico Ing. Vladimir Robles y a todos quienes durante este tiempo han formado parte del Grupo de Investigación en Inteligencia Artificial y Tecnologías de Asistencia (GIIATA) de la Universidad Politécnica Salesiana - Sede Cuenca, por brindarnos de manera desinteresada toda el soporte necesario para desarrollar de mejor manera el presente trabajo

Además queremos hacer un extensivo agradecimiento al Dr. Hernán Tenorio, Director del «Instituto de Parálisis Cerebral del Azuay» (IPCA), a la Mst. Gabriela Ortíz, Fonoaudióloga del IPCA, al Lic. Juan Medina, Fonoaudiólogo del IPCA, a la Lic. Gloria Sánchez Directora de la «Unidad Educativa Especial del Azuay» (UNEDA) y al Licenciado Marcelo Tapia, Terapista de Lenguaje de la UNEDA, por sus valiosas sugerencias en cuanto al diseño y funcionalidades del robot y sobre todos por abrirnos las puertas de sus respectivos centros y brindarnos la oportunidad de palpar una realidad diferente a la nuestra, permitiéndonos ver el verdadero enfoque de la ingeniería: el aporte a la sociedad.

Mario Esteban - Marco Antonio

Dedicatoria

A mis padres por brindarme su apoyo incondicional en cada momento de mi vida y ser el pilar fundamental por el cual he llegado a esta importante etapa de mi formación académica, a mis hermanos por tomarme como referente en su etapa formativa, siendo así la motivación para ser un buen ejemplo para ellos, a todos mis familiares y amigos que me han brindado gratas experiencias a los largo de mi vida universitaria haciendo muy llevaderos los días de estudio, y a mis profesores que han logrado de manera sabia impartir sus valiosos conocimientos permitiéndome aplicarlos al beneficio de la sociedad.

Mario Esteban

A Dios y la Virgen por ayudarme y protegerme cada día de mi vida, a mis padres por darme la oportunidad de convertirme en un profesional, tener paciencia e incentivarme para siempre buscar ser algo más en la vida, a mis hermanos por ser la motivación para continuar mis estudios, dándoles un ejemplo de lo que pueden llegar a conseguir si se lo proponen, a mis abuelos y tíos que siempre me apoyan y me aconsejan seguir adelante en mis saberes para ser mejor de lo que fue el día anterior.

Marco Antonio

Índice

Índice de figuras

1. Introducción

El avance de la ciencia ha permitido que las tecnologías relacionadas con la inteligencia artificial tengan un gran impacto en el campo de la educación, pero a pesar de esto, aún existen pocos sistemas enfocados en brindar un soporte para las terapias aplicadas en los niños con diferentes discapacidades, principalmente de la comunicación.

La realidad actual según el Consejo Nacional de Igualdad de Discapacidades (2016) establece que en el Ecuador existen $408,021$ personas con discapacidad, de las cuales:

- $51,875$ tienen discapacidad auditiva
- $193,520$ tienen discapacidad física
- $91,651$ tienen discapacidad intelectual
- $5,614$ tienen discapacidad de lenguaje
- $7,690$ tienen discapacidad psicológica
- $9,675$ tienen discapacidad psicosocial
- $47,996$ tienen discapacidad visual

Como se puede apreciar, $5,614$ personas presentan algún tipo de discapacidad de lenguaje, pero se debe tener en cuenta que las discapacidades auditivas e intelectuales pueden tener algún desorden de lenguaje asociado, por lo que se tiene $149,140$ personas con algún desorden de la comunicación asociado, por otro lado, el $8{,}7\,\%$ del total de personas con discapacidad son niños de hasta 12 años de edad[2]. Por lo tanto, se ha analizado que es de vital importancia el desarrollo de herramientas que permitan brindar un soporte para mejorar la interacción social de estos grupos poblacionales y especialmente en las etapas tempranas del desarrollo, enfocándonos específicamente en personas que sufren trastornos relacionados con el lenguaje. Para ello, se plantea el diseño, construcción e implementación de un sistema inteligente autónomo que brinde soporte en las terapias entorno a las discapacidades de lenguaje, siendo así una herramienta que permita a los terapeutas la aplicación de terapias de una manera innovadora y llamativa para los pacientes.

2. Objetivos

2.1. General

Implementar un sistema robótico inteligente autónomo que brinde soporte durante el proceso de terapia de lenguaje.

2.2. Específicos

- Diseñar y construir la estructura física que sustentará el robot prototipo.
- Diseñar e implementar una placa electrónica que posibilite establecer el proceso de comunicación entre el software de control y el hardware.
- Diseñar y programar las funciones básicas de control del robot, tales como:
  - Detección de rostros empleando técnicas de visión por computador.
  - Control de motores a fin de lograr 3 grados de libertad.
  - Diseñar y desarrollar un módulo de control remoto basado en interfaces web.
  - Diseñar e implementar un módulo de conversión de texto a voz.
  - Implementar un módulo de búsqueda web a fin interactuar con el mismo a través de comandos de voz.
- Diseñar y desarrollar una aplicación que contenga ejercicios enfocados al soporte de la terapia de lenguaje.

3. Estado del arte

Estudios recientes han demostrado que los niños con algún tipo de discapacidad son más receptivos respecto a agentes robóticos que frente a sus pares. Un ejemplo de ello se puede encontrar en [12], donde se demuestra que niños con autismo pueden desarrollar habilidades comunicativas al interactuar con robots humanoides. Por otra parte, en [19] se ha demostrado igualmente que al interactuar con el robot humanoide KASPAR han mejorado sus habilidades sociales y de atención, en [1] se hace un avance importante sobre el mismo robot, dotándolo de sensores táctiles en la piel, como parte del proyecto ROBOSKIN. Dichos sensores pueden generar una retroalimentación de diferentes tipos de interacciones, desde amable hasta brusca, debido a la detección de variaciones de fuerza en cada sensor. En tal virtud, está probado el impacto positivo que tiene el incorporar agentes robóticos tanto en las actividades diarias del aprendizaje como en las diferentes terapias relacionadas. El uso de robots en la interacción con niños con desorden del espectro autista ha sido ampliamente explotado; uno de los robots comerciales más usados para este propósito es NAO, un robot humanoide con 11 grados de libertad en sus extremidades inferiores y 14 grados de libertad en sus extremidades superiores, cuenta con sensores táctiles, 2 acelerómetros, 3 giroscopios y un sistema de visión artificial compuesto por dos cámaras en la parte frontal de la cabeza [7]. Debido a los múltiples grados de libertad y a su variedad de sensores, este robot puede desenvolverse prácticamente de manera autónoma. En [16] se presenta un experimento piloto acerca del impacto del robot NAO en la interacción con niños autistas, teniendo resultados positivos en los primeros acercamientos, además de citar valiosas recomendaciones relacionadas con el uso de robot con niños con discapacidades. Por otro lado en [5] se muestra el uso de PABI (*Penguin for Autism Behavioral Interventions*) un robot inspirado en un pingüino, el cual cuenta con 8 grados de libertad, incluyendo movimiento de ojos, pico, cabeza y alas, lo que le permite expresar emociones, además, posee un procesador interno, conexión Wifi y Bluetooth, y una cámara USB en cada ojo, lo que le permite realizar procesos programados, conectarte a un PC y mantener el contacto visual con el paciente. En las pruebas piloto realizadas con PABI se ha visto que los niños establecen un lazo afectivo con él, demostrando así la efectividad del uso de asistentes robóticos en las terapias.

4. Marco metodológico

4.1. Diseño y construcción de la estructura física

Una de las primeras etapas en el desarrollo del prototipo tiene que ver con su parte estructural, ya que es la encargada de soportar todos los elementos electrónicos y mecánicos necesarios para darle al robot las funcionalidades requeridas.

Dado que la población objetivo del proyecto son niños con desórdenes de la comunicación, fue necesario contar con la asesoría de expertos en la terpia infanto-juvenil, y por ello se desarrolló este proceso de forma conjunto con 2 centros de educación especial en la cuidad de Cuenca, donde se recogieron sugerencias acerca de la parte estética del prototipo. Entre estas sugerencias se encuentra el diseño antropomórfico de la estructura, debido a que los niños son más receptivos frente a figuras concretas y que se encuentren dentro de su entorno de desarrollo, ya que en las primeras etapas del proyecto el diseño estaba planteado en forma de pirámide o en forma de una gota de agua. Otra importante sugerencia fue la de diseñar solamente de la parte interna a manera de un esqueleto, con el fin poder utilizar diferentes trajes sobre la misma estructura. Tomando como base estas sugerencias, se procedió al diseño mediante un software de modelado 3D, obteniendo 7 piezas, 2 piernas, 2 brazos, la cabeza y el tronco divido en 2 partes con el objetivo de poder trabajar sobre los elementos electrónicos y mecánicos internos del prototipo. Todas las piezas forman finalmente la estructura completa del robot, como se puede apreciar en la figura 1.

Figura 1: Diseño 3D del asistente robótico.

Posteriormente, en cuanto a la construcción de la estructura, se hizo uso de la tecnología de impresión 3D, para esta etapa fue necesario contratar el servicio de impresión, dado que no era factible llevarla a cabo en la universidad. A continuación se procedió al ensamblaje de todos los componentes teniendo finalmente el prototipo armado como se muestra en la figura 2.

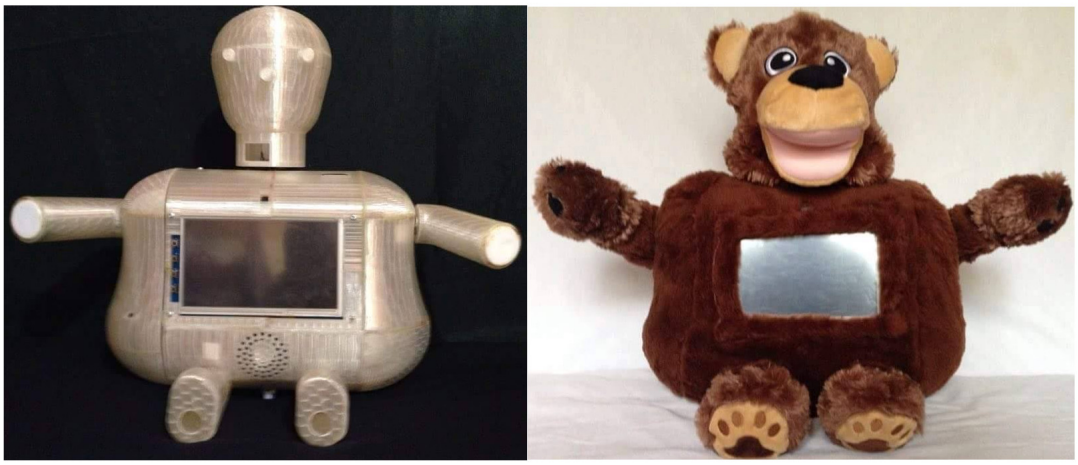

Figura 2: Estructura armada sin traje (izquierda) y con el traje de perro (derecha).

4.2. Diseño y construcción de la placa electrónica

La placa electrónica será la encargada de comunicar la parte de hardware y de software, además de brindar la alimentación a cada parte del sistema, por tanto, es necesario tener claras las funciones específicas que va a realizar, para esto, el diseño de la misma se basa en el diagrama de bloques que se presenta en la figura 3.

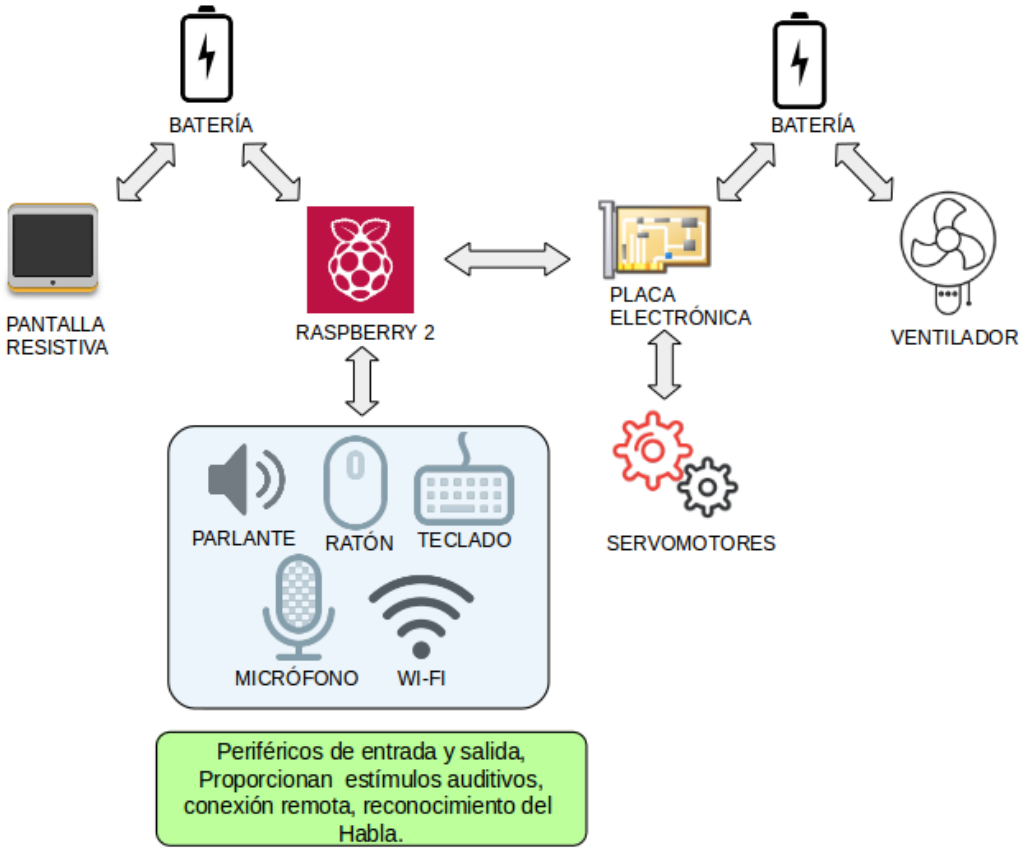

Figura 3: Diagrama de bloques del sistema electrónico .

Cabe recalcar que la placa electrónica será la encargada de manejar tanto la parte de control con un voltaje de $3{,}3V$, como la parte de potencia con un voltaje de $5V$, por lo que es necesario aislar cada uno de

los circuitos dentro de la misma.

El nivel de tensión de $5V$ alimenta a los USB, al driver de motores, los servomotores, parlante y pantalla, mientras que el segundo nivel de tensión de $3{,}3V$ proporcionado por los pines del Raspberry Pi será el encargado de controlar el sentido de giro, activación y desactivación de los motores mediante el uso del driver de motores DC.

En cuanto a la corriente, la limitante es la batería o fuente externa que alimenta al circuito impreso, por lo que se realizó una prueba de consumo de corriente en el arranque de los dispositivos como Raspberry Pi y la pantalla táctil, llegando a consumir aproximadamente $2A$ solamente al cargar el sistema operativo, ya que el consumo disminuye cuando el sistema se estabiliza.

4.2.1. Diseño del PCB

La primera etapa de cualquier diseño de PCB[1] es generar un esquema del circuito, es decir, generar las conexiones apropiadas para cada elemento involucrado en la placa electrónica. Para el desarrollo tanto esquemático como del trazado de las pistas se utiliza el software de diseño electrónico *«Altium Designer»,* obteniendo las diferentes secciones de la placa en forma esquemática como se muestra en las figuras 4, 5 y 6.

En la figura 4 se puede observar las conexiones que se realizarán desde los puertos GPIO[2] del Raspberry hacia los diferentes periféricos. En el caso de los servomotores y la pantalla táctil se puede apreciar que la comunicación entre la película táctil y el Raspberry se realiza mediante el protocolo SPI[3], es por ello que se usa los pines MOSI[4], MISO[5], CLK[6] y SPI_INT[7] del Raspberry, mientras que el *display* específicamente se comunica mediante HDMI directamente hacia el puerto correspondiente en el Raspberry. Además, se puede ver que los pines S1, S2 y S3 son los encargados de enviar la señal PWM[8] hacia los servomotores, y los pines 2 y 3 de los mismos están respectivamente conectados a $5V$ y GND.

[1](Printed Circuit Board): Tarjeta de circuito impreso
[2](General Purpose Input-Output): Entradas-Salidas de propósito general
[3](Serial Peripheral Interface): Interfaz de periféricos seriales
[4](Master Output Slave Input): Salida de datos del Master(Maestro) y entrada de datos al Slave(Esclavo).
[5](Master Input Slave Output): Salida de datos del Slave(Esclavo) y entrada al Master(Maestro).
[6](Clock): Es el pulso que marca la sincronización. Con cada pulso de este reloj, se lee o se envía un bit.
[7]Interrupción del protocolo de comunicación SPI
[8](Pulse Width Modulation): Modulación por ancho de pulso

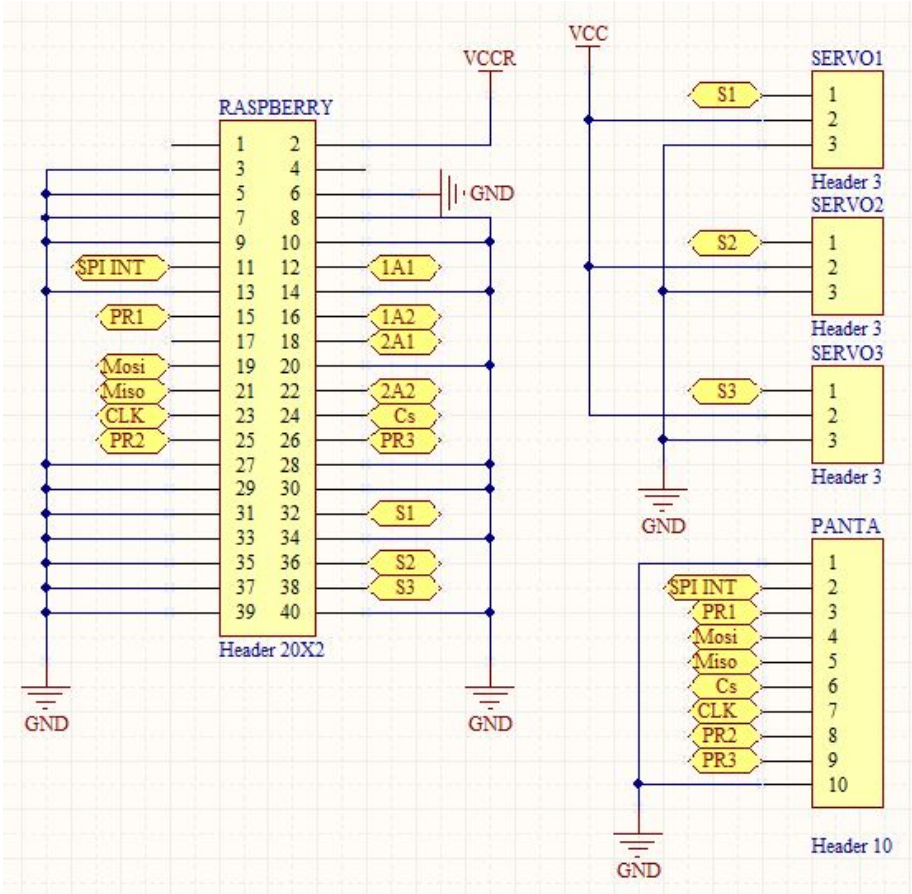

Figura 4: Conexiones para Raspberry pi, servomotores y pantalla.

En la figura 5 se puede notar que todos los voltajes involucrados en la alimentación son de $5V$ incluyendo el voltaje que se tiene en los puertos USB, mas no se debe olvidar que los voltajes del control los cuales se obtienen del Raspberry son de $3{,}3V$, deben ser manejado de manera aislada, en la figura 6 se detalla el uso de estos niveles de tensión.

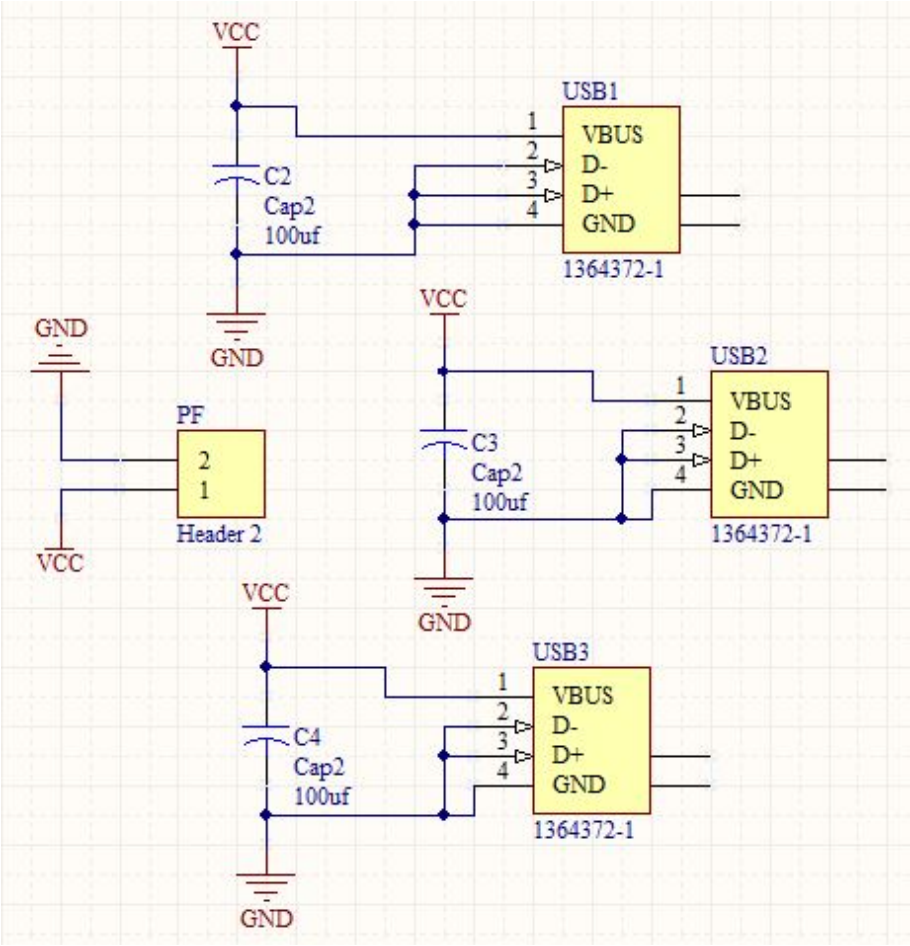

Figura 5: Entradas y salidas de alimentación de voltaje.

Como se puede ver en la figura 6, los pines 2, 7, 10, 15 y 16 del driver *«L298»* son los únicos que manejan el voltaje lógico de $3{,}3V$, por lo que se los maneja de manera aislada llamándolos a estos $VCCR$ a diferencia de VCC que representa el voltaje de potencia de $5V$. De igual forma, se aprecia también que los pines 1 y 9, encargados de la activación de los motores, están conectados mediante una resistencia hacia VCC, por lo que se puede saber que los motores estarán siempre activados. Finalmente, se puede notar el puente de diodos conectados entre el driver y el conector de los motores con el fin de evitar el retorno de corrientes hacia la placa electrónica, principalmente cuando los motores se mueven manualmente, dado que estos se comportan como generadores e introducen corriente no deseada sobre el circuito.

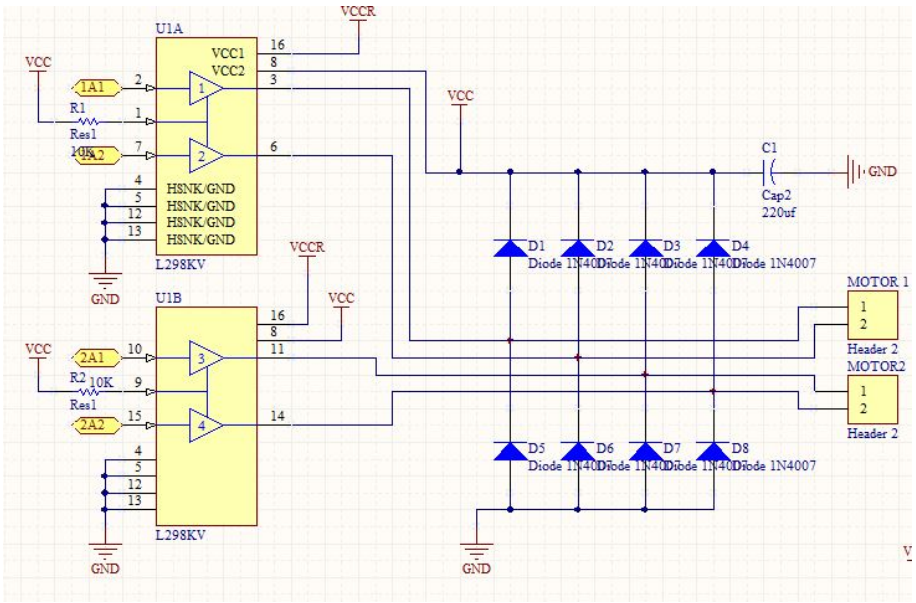

Figura 6: Driver para motores DC.

Una vez realizado el circuito esquemático, se procede a ubicar los componentes dentro de la placa según las necesidades y la ubicación que tendrá la placa en el sistema en general para a continuación realizar el trazado de las pistas.

En esta parte de proceso es necesario tener en cuenta algunas reglas de ruteo, como por ejemplo: El grosor de la pista debe estar de acuerdo a la corriente que pasará por dicha pista, se debe evitar ángulos de 90 grados ya que evita el libre flujo de corriente a lo largo de la placa, separar la parte de control de la parte de potencia, se debe tener un mínimo de separación entre pista de 0.4mm, evitar la aglomeración de pista sobre alguna sección de la placa ya que podría generar un efecto capacitivo no deseado sobre la placa electrónica, entre otros.

Basado en las reglas mencionadas anteriormente se obtiene el diseño que se muestra en la figura 7, donde se puede observar que la placa es de doble cara, dado que permite de mejor manera aislar los circuito de control y de potencia, además de cumplir con los requisitos citados en el párrafo anterior.

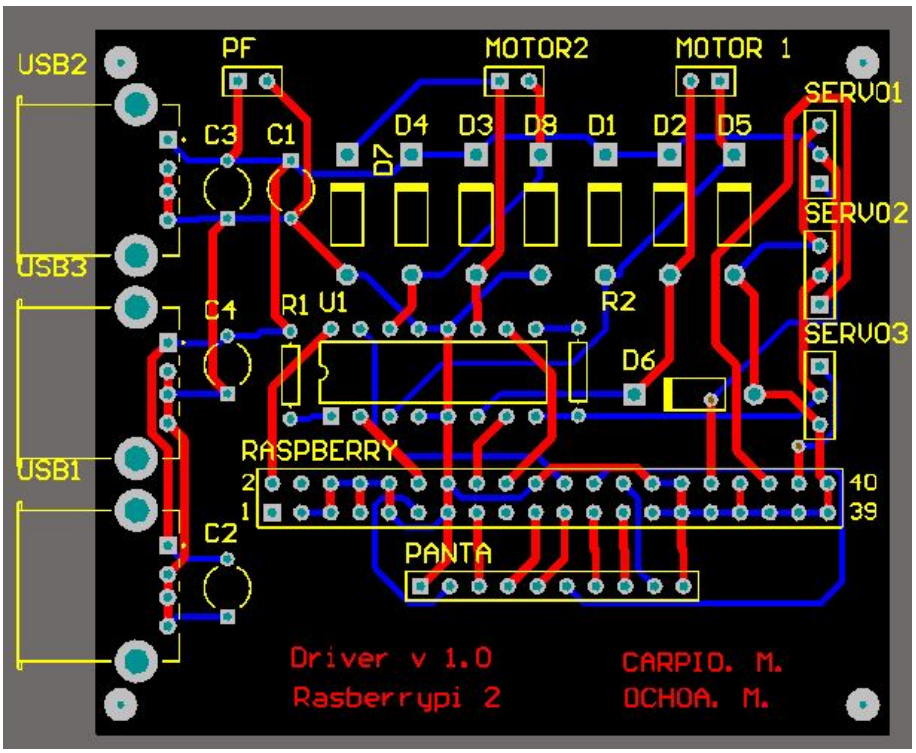

Figura 7: Diseño de trazado de pistas de la placa electrónica

4.3. Desarrollo de los módulos de interacción

Previo al desarrollo de cada uno de los módulos de software, es necesario definir el lenguaje de programación a utilizar (dado que esto será determinante en el desempeño del sistema). En virtud de que se utilizará un Raspberry Pi, el cual es un sistema embebido con capacidades limitadas, se debe procurar que los procesos sean, en la medida de lo posible, muy eficientes.

Basado en la premisa expuesta en el párrafo anterior, el lenguaje a utilizar es Python, ya que Raspberry esta desarrollado para explotar las ventajas que ofrece este lenguaje de programación, si bien es cierto, cuando se trabaja con sistemas en donde se debe cuidar el rendimiento la mejor opción es usar lenguajes como C o C++, se ha notado una marcada diferencia en la cantidad de librerías desarrolladas para Python sobre las librerías desarrolladas para C o C++. Además, dichas librerías tienen una comunidad que continuamente las esta optimizando y depurando por lo que se las puede usar de manera confiable, por lo tanto se puede explotar de mejor manera las ventajas que ofrece el modulo Raspberry Pi.

Una vez definido el lenguaje a utilizar, es necesario obtener las librerías suficientes para desarrollar los diferentes módulos, los cuales se especificarán en cada una de las secciones siguientes.

4.3.1. Desarrollo del módulo de detección de rostros

El desarrollo del módulo de detección de rostros consta de 4 etapas principales que se puede observar en la figura 8.

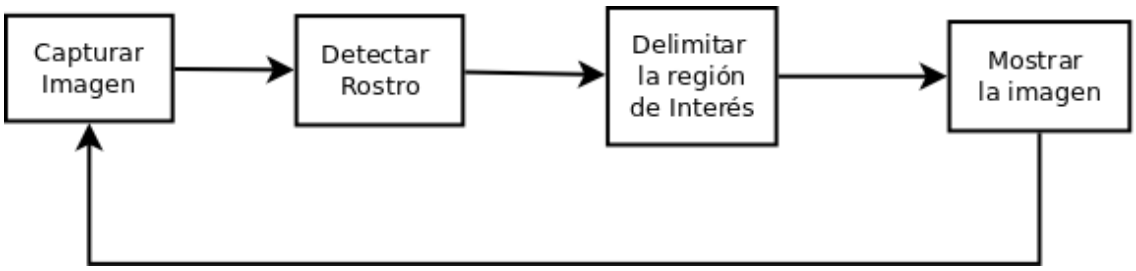

Figura 8: Diagrama de flujo del proceso de detección facial.

En cuanto a la captura de la imagen es necesario utilizar ciertas librerías que permiten adquirir los datos desde la cámara propia para Raspberry (RasPicam) que utiliza un protocolo de comunicación serial CSI (*Camera Serial Interface*) mediante un conector de cable plano S5, el cual va anclado a la tarjeta Raspberry.

Existe una librería para Python llamada «*picamera*», la cual es desarrollada por *Dave Jones* y permite utilizar de manera sencilla la cámara RasPicam, mediante esta librería se puede obtener imágenes y videos a partir de este periférico[10].

Por otro lado, la librería que se utiliza para el procesamiento de la imágenes es OpenCV, la cual esta enfocada en tareas de visión por computadora y es muy útil para trabajar con las imágenes obtenidas desde la cámara[8].

OpenCV está concebida para trabajar sobre una PC, por lo que sus librerías incluyen métodos para la obtención de imágenes a través de una cámara web estándar y guarda sus capturas en forma de matriz, mientras que al usar las RasPicam el formato utilizado es diferente, para lo cual es necesario un complemento de la librería *«picamera»* llamado «*picamera.array*», este complemento permite almacenar la captura en el mismo formato que maneja OpenCV [8, 10].

Para detectar un rostro en una imagen se usa *clasificadores en cascada basados en características Haar*, este método fue propuesto por *Paul Viola* y *Michael Jones* en su investigación *«Rapid Object Detection using a Boosted Cascade of Simple Features»* publicada en 2001, el cual aprovecha las características zonas claras y oscuras de una imagen para detectar un objeto dentro de la misma, como por ejemplo en el caso de detección de rostros la zona de los ojos es mas oscura que la zona de la nariz o los pómulos, entre otras. Además, el método usado en la investigación de *Viola y Jones* está optimizado ya que en lugar de comprobar cada una de estas características se haga de manera secuencial, es decir, comprueba una cantidad determinada de características y si la imagen pasa esta primera etapa se comprueba otro grupo de características, de allí su nombre *clasificadores en cascada*[18].

Nuevamente, OpenCV permite implementar los clasificadores en cascada de manera sencilla mediante sus librerías, en este caso se usa *«cascade_classifier»* el cual recibe como uno de sus argumentos un archivo en formato XML (*eXtensible Markup Language*) que representa los datos del clasificador entrenado, este archivo puede ser creado mediante un entrenador que está implementado en OpenCV o se puede utilizar los clasificadores pre-entrenados para detección de rostros, ojos, sonrisa, etc, que ofrece la misma librería[8].

Una vez aplicado el clasificador éste devuelve una matriz con los valores de coordenadas en donde se encuentra un rostro, como se puede ver en la figura 9, en el caso que exista mas de un rostro el clasificador devuelve en las siguientes filas de la matriz la coordenadas correspondientes.

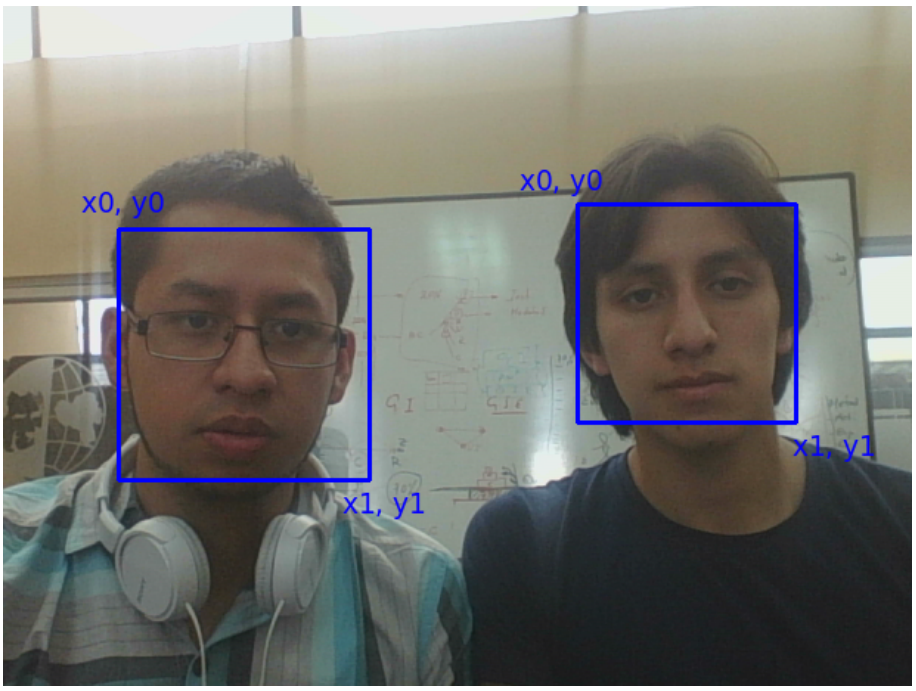

Figura 9: Detección facial.

Contando ya con la posición del rostro se puede dibujar un rectángulo tomando como referencia las coordenadas correspondientes a dicha posición. Para realizar esta tarea se utiliza un método llamado «*cv2.rectangle*» propio de *OpenCV*, véase la figura 9.

Finalmente, se muestra la imagen con el comando *«cv2.imshow»*, teniendo en cuenta que debe estar dentro de un bucle (en éste caso *«while»*) para que permita ver la imagen de manera continua.

Después de todo el desarrollo, es necesario reunir todas estas funcionalidades dentro de una o varias clases con el fin de utilizarlas dentro de otros programas de manera sencilla, por lo que se genera un archivo llamado *«CamaraRaspberry.py»* donde se encuentran clases y métodos con las mencionadas funcionalidades. En la figura 10 se representa en diagrama con notación UML dichas clases y métodos.

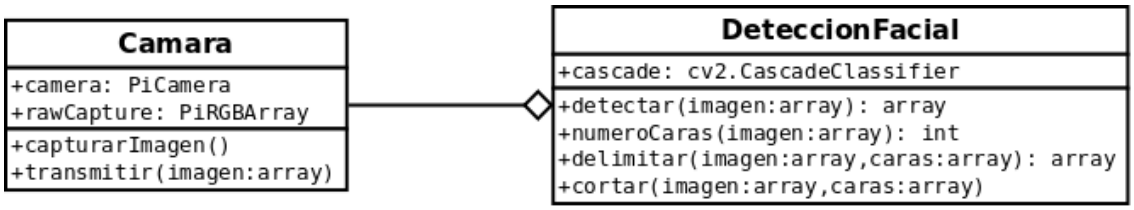

Figura 10: Diagrama de clase de Detección Facial

4.3.2. Desarrollo del módulo de control de motores

El desarrollo del módulo de control de motores es uno de los más importantes, ya que permite la interacción entre el robot y su entorno, que en este caso específico se utiliza para simular emoción, aprobación, desaprobación, entre otras emociones necesarias para llevar a cabo la terapia de lenguaje.

Los motores a utilizar son de tipo servomotor, dado que ofrecen ventajas como el control sobre el ángulo de giro necesario para mover los brazos y la cabeza del robot, y el torque que posee gracias a sus juegos de engranajes, aspecto que permite mover todo el robot mediante sus ruedas en la parte inferior. En la figura 11 se puede ver la ubicación de los motores dentro del robot.

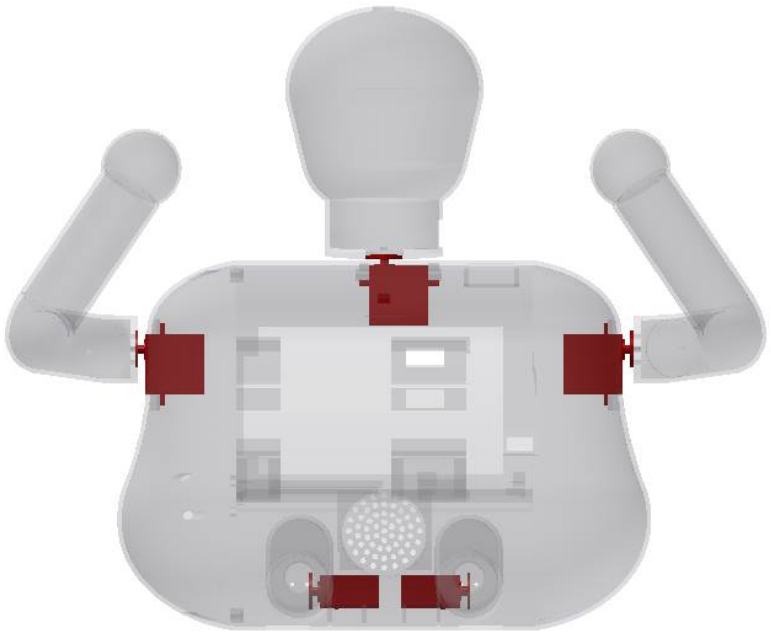

Figura 11: Ubicación de los servomotores dentro del robot.

Aprovechando que el Raspberry Pi posee varias entradas y salidas digitales, se puede utilizarlas para enviar las señales de control hacia la tarjeta electrónica encargada de gestionar dichas señales, entonces el desarrollo se centra en la generación de estas señales, proceso que se especifica en los siguientes párrafos.

Existen algunas librerías encargadas de gestionar las entradas y salidas digitales del Raspberry Pi, entre las más utilizadas se encuentran *«RPi.GPIO» y «pigpio»*. Para este caso se utiliza la librería *«pigpio»* desarrollada por el usuario *joan2937* del repositorio digital *GitHub*[9], ya que presenta la gran ventaja de que sus señales, principalmente las señales de PWM, son generadas mediante hardware, por lo que no se ven afectadas por los procesos que se ejecutan internamente en el Raspberry Pi. Esto se diferencia de la libreria *«RPi.GPIO»* desarrollada por *Ben Croston* [3] que genera sus señales mediante software, por lo que al ejecutar varios procesos estas señales se ven distorsionadas y los motores se mueven de manera aleatoria e incluso se pierde torque dado que deja de generar el voltaje necesario para que se muevan a plena potencia.

La frecuencia de PWM utilizada es de 100Hz, y el ciclo útil necesario para mover en motor entre 0 y 270 grados con dicha frecuencia es de 15 % hasta 65 %, todos estos datos se puede obtener de la hoja de datos correspondiente a cada motor[4].

De igual forma que en el caso anterior, es necesario reunir todas las funcionalidades desarrolladas dentro de una o varias clases, por consiguiente para este módulo se tiene el archivo «motores.py» con una clase que posee el mismo nombre y los diferentes métodos que permiten manejar cada motor de manera independiente y combinada, en la figura 12 se representa en diagrama con notación UML dicha clase y métodos.

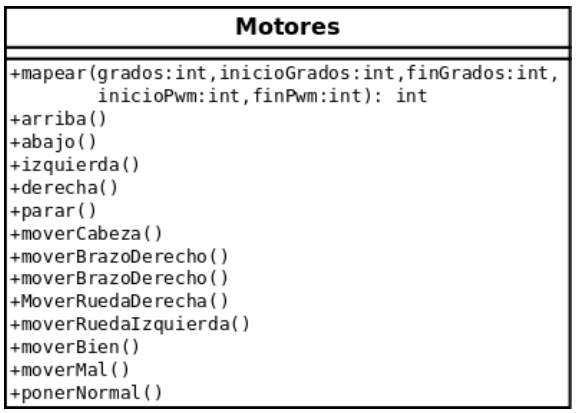

Figura 12: Diagrama de clases de Motores

4.3.3. Desarrollo del módulo de conversión de texto a voz

Para el desarrollo de este módulo, se exploraron varias opciones de conversión de texto a voz, entre ellas *«Festival», «MBrola», «Espeak» y «spd-say»*, siendo *«Festival»* la opción elegida dada su casi exclusiva compatibilidad con el español gracias al proyecto desarrollado por la *Junta de Andalucía* donde se han creado las voces masculina y femenina compatible con *«Festival»*[17, 11].

La implementación en este caso es sencilla, ya que solo es necesario pasar como parámetro el texto que se desea convertir a voz hacia la función *«SayText»* de *«Festival»*, sin embargo, es necesario crear una clase con el fin de usarlo dentro de otras aplicaciones, y sobre todo para que sea compatible con Python.

4.3.4. Desarrollo del módulo de búsqueda web mediante comandos de voz

Al igual que en el caso anterior, existen algunas alternativas para la conversión de voz a texto, y después de analizar las diferentes opciones, la elegida fue *«SpeechRecognition»,* la cual usa diferentes librerías como: *Google Speech Recognition, Wit.ai, IBM Speech to Text, y AT&T Speech to Text*. De entre estas opciones la más robusta es *Google Speech Recognition*, ya que cuenta con una gran precisión en cuanto al reconocimiento de voz, además *«SpeechRecognition»* cuenta con muchas funcionalidades, como por ejemplo: acoplar el micrófono al ruido del ambiente, establecer el umbral en decibelios en donde es detectable la voz, establecer el tiempo de espera antes de procesar la voz obtenida, etc.[20].

Dadas las múltiples funcionalidades que presenta dicha librería, no es necesario crear un clase especial, ya que simplemente instanciando los objetos correspondientes está lista para su uso.

Cabe recalcar que una de las principales desventajas de usar *Google Speech Recognition* es que es necesario una conexión a internet para que el reconocimiento funcione, por lo que limita las funcionalidades del robot en lugares sin acceso a internet.

Este modulo será utilizado específicamente en el modo autónomo de la aplicación que se explicará en la siguiente sección.

4.3.5. Desarrollo del módulo de control mediante interfaz web

Para el desarrollo de este módulo se utiliza un framework llamado «Flask», el cual está enfocado exclusivamente a Python, por lo que permite ejecutar *scripts* de dicho lenguaje mediante una interfaz web, además permite generar un servidor propio dentro del sistema operativo, y como consecuencia de ello posibilita desplegar páginas web sobre este servidor[15].

Entonces, según lo expuesto anteriormente, «Flask» será la interfaz entre la página web, que estará desplegada sobre el propio servidor, y las funcionalidades de detección facial, control de motores y conversión de texto a voz descritas en las secciones anteriores.

La aplicación web consta de 4 ventanas, las cuales están desarrolladas en HTML y CSS, en la figura 13, 14, 15, 16 se puede ver la ventana de «Inicio», «Movimiento», «Cámara» y «Hablar» respectivamente.

En la figura 13 se aprecia la página de inicio en donde se muestra simplemente el nombre del robot y una pequeña descripción del objetivo del mismo, en la parte superior se encuentran los enlaces hacia las ventanas con las diferentes funcionalidades.

Figura 13: Pagina principal

En la figura 14 se puede ver 4 botones, los cuales permiten al usuario mover las ruedas del robot de manera remota, por tanto el nombre de cada botón corresponde al movimiento que se puede hacer, siendo estos: «Arriba», «Abajo», «Izquierda», «Derecha»

Figura 14: Pagina de Movimiento

En la figura 15 se tiene la funcionalidad de acceder a la cámara de robot, permitiendo al usuario observar

de manera remota lo que se encuentra al rededor del mismo.

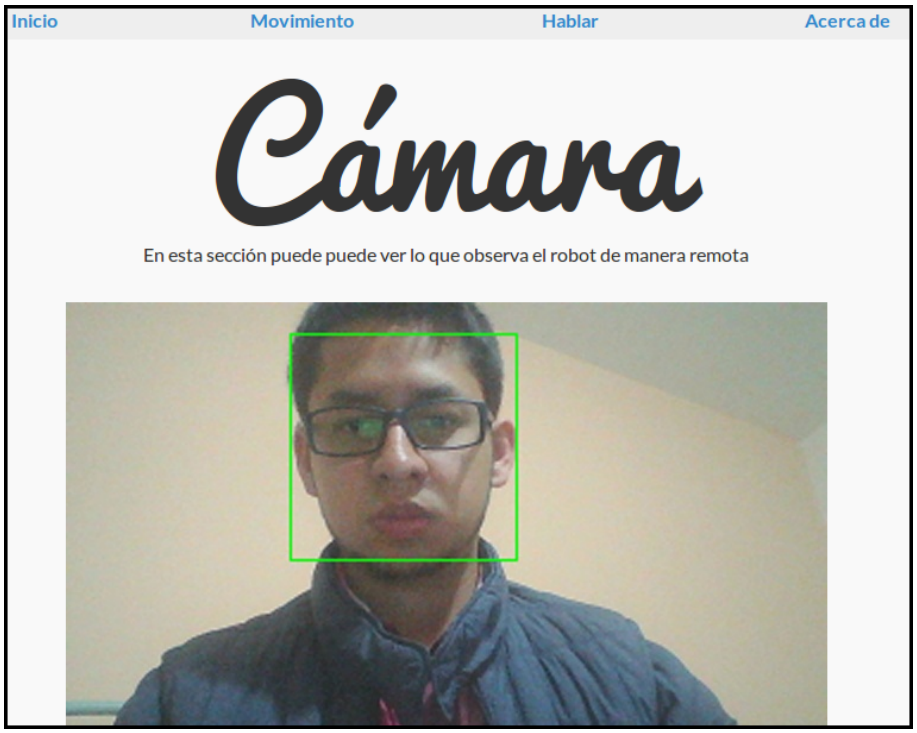

Figura 15: Pagina de Cámara

Finalmente, en la figura 16 se tiene la página con la funcionalidad de hablar, en este caso es necesario introducir una palabra o frase en la caja de texto y pulsar en el botón «Hablar», esta frase será enviada la librería de conversión de texto a voz y el robot pronunciará la frase que el usuario desea, igualmente de forma remota.

Figura 16: Pagina de Hablar

En un inicio se planteó tener todas las funcionalidades dentro de una sola página, sin embargo, se presentó el problema de que la corriente excedía el estado seguro del banco de baterías, por lo que se apagaba y se bloqueaba como medida de seguridad, es por ello que se decidió realizar cada función en una página diferente.

La ventaja de realizar el control remoto mediante una aplicación web es que se puede acceder al robot

desde cualquier dispositivo que cuente con una conexión wireless y un navegador web, entre estos se encuentran: Laptops con Windows, Linux o Mac, Tablets, teléfonos inteligentes Android o IOs, por lo que no se limita a un solo tipo de dispositivo.

4.4. Diseño y desarrollo de una aplicación enfocada al soporte en la terapia de lenguaje

En esta sección se describe el proceso de desarrollo de la aplicación para soporte en la terapia de lenguaje, con el fin de darle una funcionalidad concreta al robot y a todos los módulos desarrollados en la secciones anteriores.

Dado que la aplicación debe servir como soporte en la terapia de lenguaje, se ha implementado una base de datos en donde se almacenará todo lo relacionado a los pacientes, desde datos personales hasta su desempeño en cada ejercicio, todo ello con el objetivo de que el terapista pueda consultar dichos registros y reforzar los conceptos en terapias específicas dependiendo del paciente y sus necesidades particulares.

4.4.1. Usuarios y roles

Dado que la base de datos maneja información sensible, es necesario implementar un sistema de usuarios y roles. Para ello se han creado 3 roles principales: Terapista, Administrador y SuperUsuario, los cuales cuentan con permisos específicos según el rol.

En la figura 17 se puede observar en el diagrama de casos de uso del *«Administrador»* el cual tiene permisos para modificar y registrar tanto pacientes como usuarios, con una excepción en éste último dado a que no es posible que un administrador pueda crear usuario de tipo *«SuperUsuario»* o incluso de un *«Administrador»*, por tanto está limitado a crear usuarios de tipo *«Terapista»* solamente, además, se puede ver que el usuario *«Administrador»* puede modificar los registros generados por los ejercicios realizados durante la terapia.

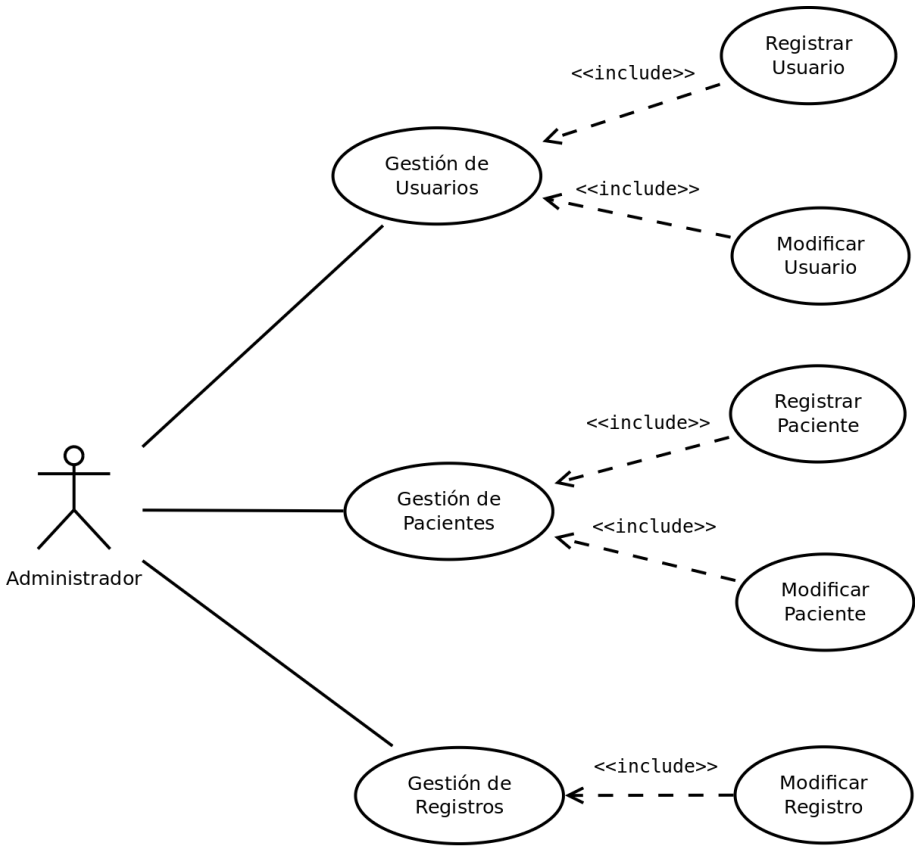

Figura 17: Diagrama Caso-Uso del Administrador

En la figura 18 se tiene el diagrama de caso-uso del «Terapista» donde se puede observar que el permiso más importante es el de realizar la terapia después de haber seleccionado un paciente, privilegio con el cual no cuenta ningún otro usuario, además se puede ver que también puede registrar un nuevo paciente y en cuanto a los registros puede ver en detalles cada uno de estos con el fin de generar una plan de terapia basado en los mismos.

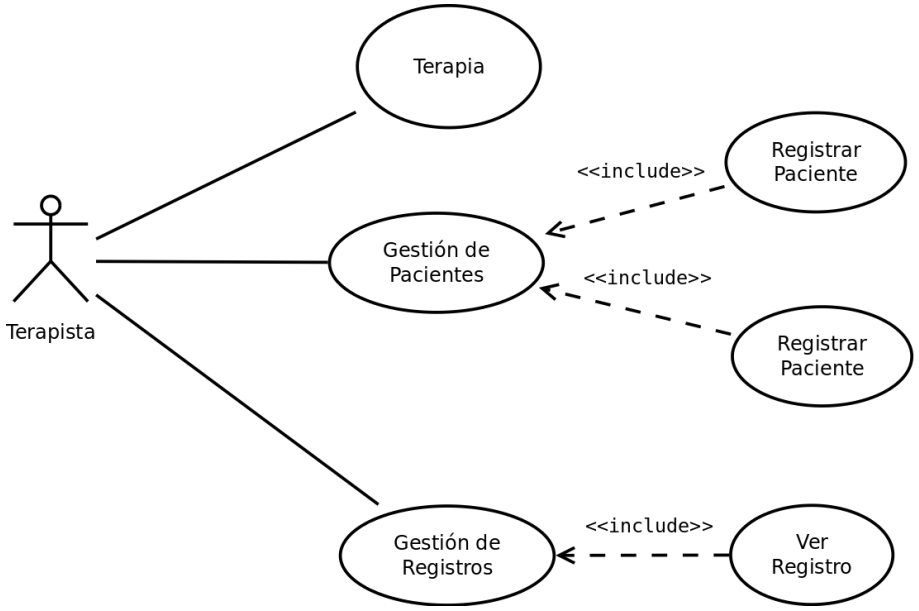

Figura 18: Diagrama Caso-Uso del Terapista

El último tipo de usuario es el «SuperUsuario», cuyo diagrama de casos de uso se tiene en la figura

19, donde se puede observar que hereda los permisos del administrador, así como incluye permisos propios enfocados directamente al mantenimiento y detección de fallos, agregar nuevas actividades y realizar consultas directas a la base de datos.

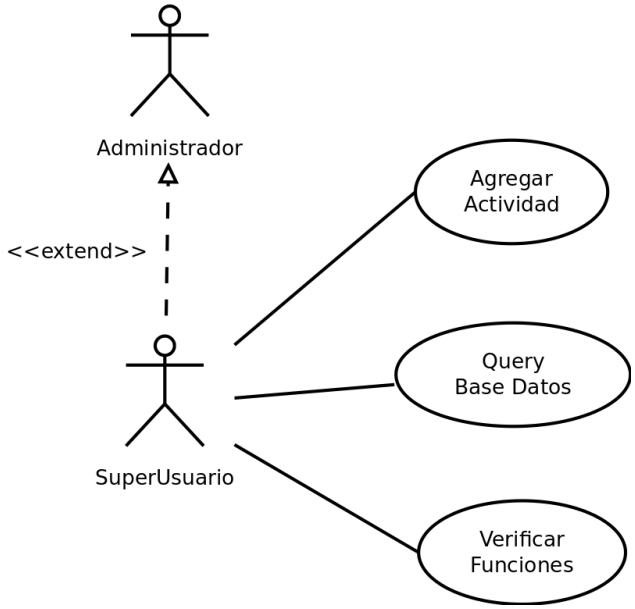

Figura 19: Diagrama Caso-Uso del SuperUsuario

4.4.2. Esquema General

Partiendo de los diagramas de casos de uso de las figuras 17, 18 y 19 se obtiene el esquema general de la aplicación como se puede ver en la figura 37 en la sección de anexos, sustentada por la base de datos, cuyo esquema se muestra en la figura 20.

En cuanto a las base de datos se tienen 8 tablas, las cuales serán las encargadas de almacenar los datos necesarios para el funcionamiento de todo el sistema, en la tabla de *«Usuario»* se tiene la toda la información personal y profesional del usuario así como el rol del mismo dentro de la aplicación, en la tabla de *«Login»* se tiene el nombre de usuario (*UserName*) y la contraseña (*Password*) necesarios para autenticarse en la aplicación, así como la clave foránea *(ForeignKey*) necesaria para relacionar un Usuario con una cuenta de autenticación. En la tabla *«Paciente»* se tiene al igual que en la tabla *«Usuario»,* todos los datos personales del paciente. Por otro lado, ya dentro de la terapia se emplea la tabla *«Sesion»* que maneja las sesiones realizadas, la cual contiene las claves foráneas que relacionan a un terapista y a un paciente con una sesión. Paralelamente se tiene la tabla *«Ejercicio»,* la que contiene campos que describen los ejercicios a realizarse, además, dado que dentro de una sesión se puede realizar más de un ejercicio que puede ser usado en más de una sesión, se debe tener una tabla intermedia llamada *«Ejercicio_Sesion»* que represente la relación de mucho a muchos presente entre la tabla *«Ejercicio»* y la tabla *«Sesion»*. De igual forma, dentro del desarrollo del ejercicio se tiene la tabla *«Diccionario»* que contiene todas las palabras con las que se va a trabajar en la terapia, dicha tabla tiene campos como: Palabra, Imagen, Silabas, Sonidos, Dificultad y Categoría con el fin de brindar la mayor cantidad de contenido multimedia y abarcar la mayoría de características de una palabra para que puedan ser usados en ejercicios posteriores. Finalmente, es necesario registrar las palabras en las que el paciente comete errores, por lo que se tiene la tabla «Falla» que contiene el registro de las palabras que falló y la palabra con la cual la confundió, ya que como se mencionó anteriormente, estos datos serán muy importantes para el terapista con el fin de desarrollar planes de terapia basados en dichos resultados

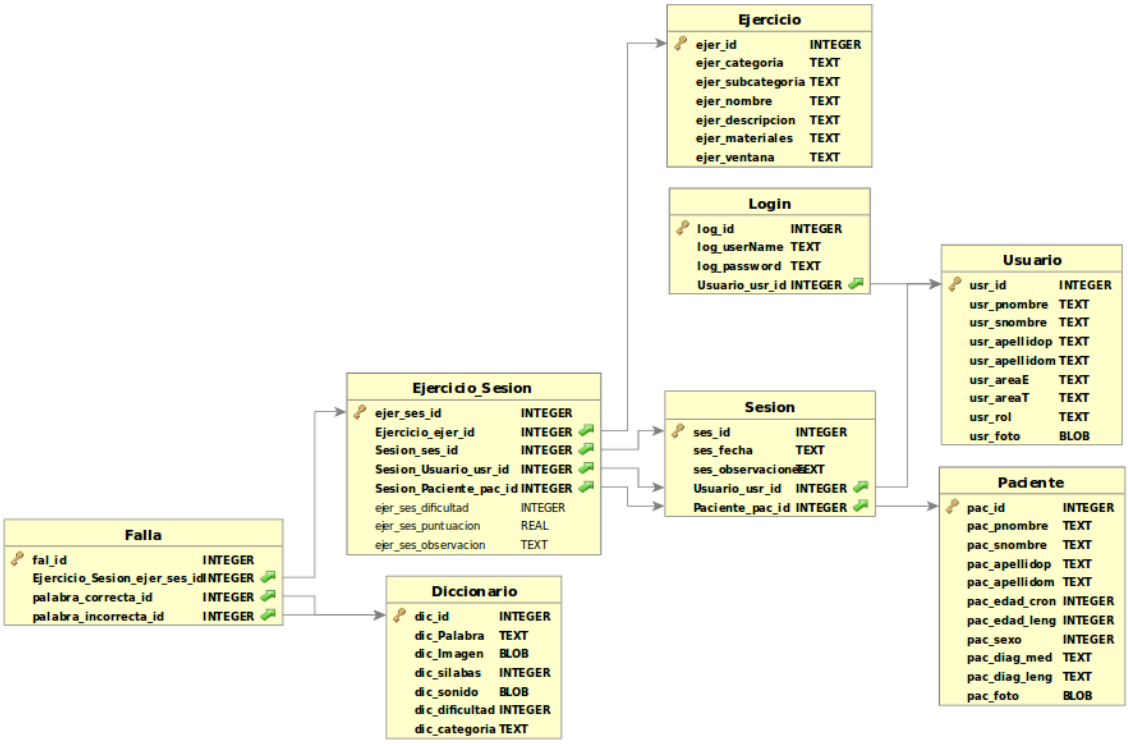

Figura 20: Diagrama entidad-relación de la base de datos.

4.4.3. Interfaz gráfica

Teniendo claro el esquema general de la aplicación, se procede a desarrollar las interfaces gráficas mediante una librería para Python llamada «PyQt» la cual permite generar *Widgets* o ventanas con diferentes elementos como cajas de texto, listas, botones, etiquetas, listas desplegables, etc[14].

La libreria PyQt cuenta además con un complemento llamado «QtDesigner» el cual permite diseñar las ventanas de manera gráfica, simplemente arrastrando y ubicando los elementos necesarios dentro del área gráfica (panel, ventana, contenedor, etc.)[13]. Posteriormente se generará un archivo con extensión XML que se convertirá a código Python mediante el comando «`pyuic4`». Los detalles de este proceso se pueden ver en [6], donde se encuentra un tutorial bastante ilustrativo.

La primera ventana en desplegarse al iniciar la aplicación es la ventana de *«Login»*, la cual se muestra en la figura 21. En ella, se puede observar que cuenta con una lista desplegable donde es factible seleccionar el usuario, y en la parte derecha se encuentra un teclado numérico con el fin de ingresar una contraseña de 4 dígitos, además, se puede ver que tiene 2 botones en la parte superior: el primer botón (derecha) sirve para acceder a las configuraciones de WiFi mientras que el segundo botón (izquierda) sirve para apagar el Raspberry y consecuentemente el robot en general.

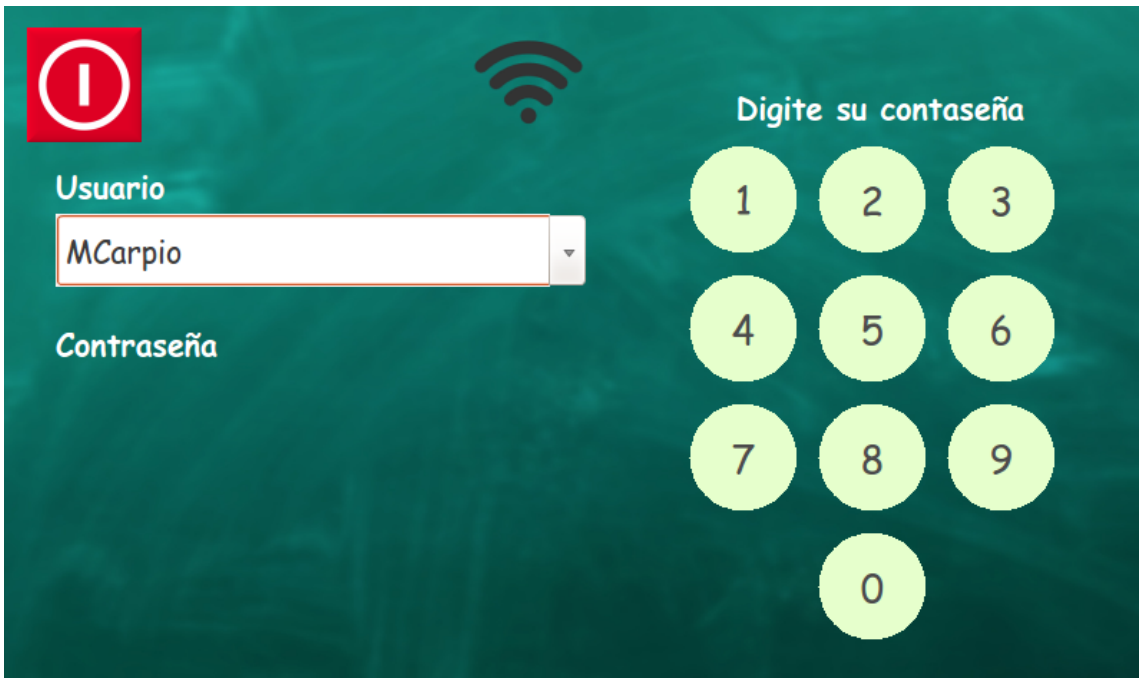

Figura 21: Ventana de Login

En la figura 22 se puede ver el diseño en *«QtDesigner»* de la ventana de Login presentada en la figura 21, como se aprecia, *«QtDesigner»* permite ubicar de manera sencilla los elementos pero presenta una clara desventaja en cuanto a la estética, ya que no permite cambiar los fondos, los colores, las formas de los botones, entre otras características, por lo que para obtener la ventana que se muestra en la figura 21 es necesario agregar estas características mediante lineas de código.

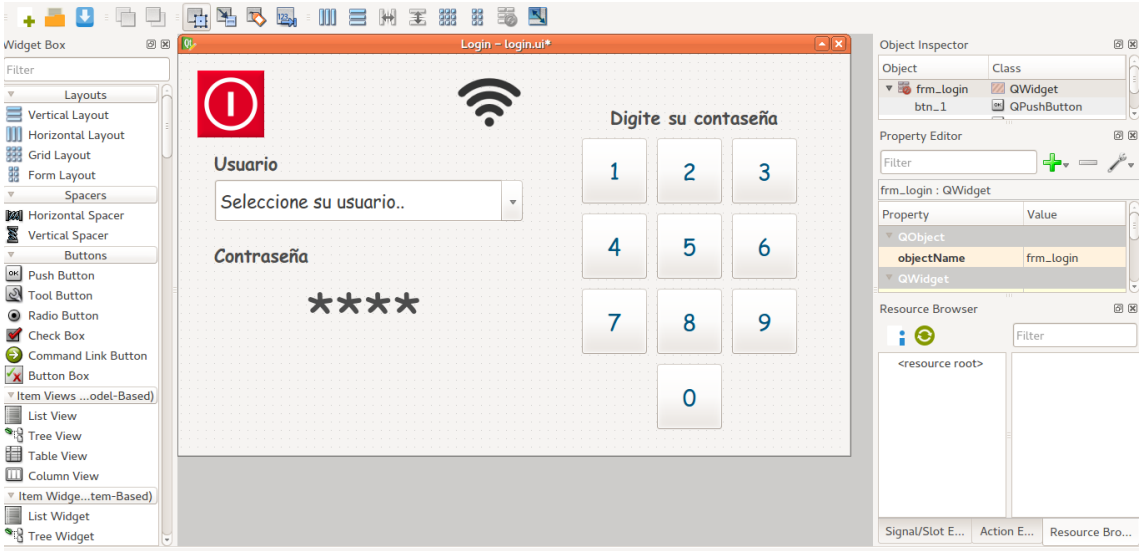

Figura 22: Diseño de ventana de Login

El proceso es el mismo para todas las ventanas de la interfaz gráfica de la aplicación, por lo que no es necesario detallar los pasos seguidos para crear cada una de ellas.

En los siguientes apartados se describen las funciones más importantes de la aplicación, y la manera de usarlas.

4.4.4. Manejo de pacientes

El manejo de pacientes hace referencia al registro, modificación y selección de los pacientes dentro del sistema, dichos registros estarán almacenados en la base de datos conteniendo la información personal y contendrán las siguientes variables: edad cronológica, edad de lenguaje y los diagnósticos tanto médico como de lenguaje. Como se aprecia en la figura 23, dichos datos son importantes para los terapistas, ya que basados en esta información se crean los planes iniciales de terapia.

Figura 23: Ventana de registro de paciente

El manejo de los pacientes está restringido según el rol de usuario, por lo que en el caso del *«Terapista»* tiene permisos para registrar un nuevo usuario o para seleccionar un usuario existente con el fin de realizar la terapia, por otro lado, el *«Administrador»* tiene permisos para registrar y modificar los datos del paciente, y por último el *«SuperUsuario»* cuenta con los mismos permisos que el *«Administrador»* (ver diagrama de caso-uso de la figura 19).

4.4.5. Manejo de usuarios

Similar al manejo de pacientes, el manejo de usuarios hace referencia al registro y modificación de los usuarios de la aplicación, la información contenida para los usuarios serán los datos personales, el área de especialización, el área de trabajo, el nombre de usuario y la contraseña (ver figura 24). Estos datos también estarán almacenados en la base de datos como en el caso anterior.

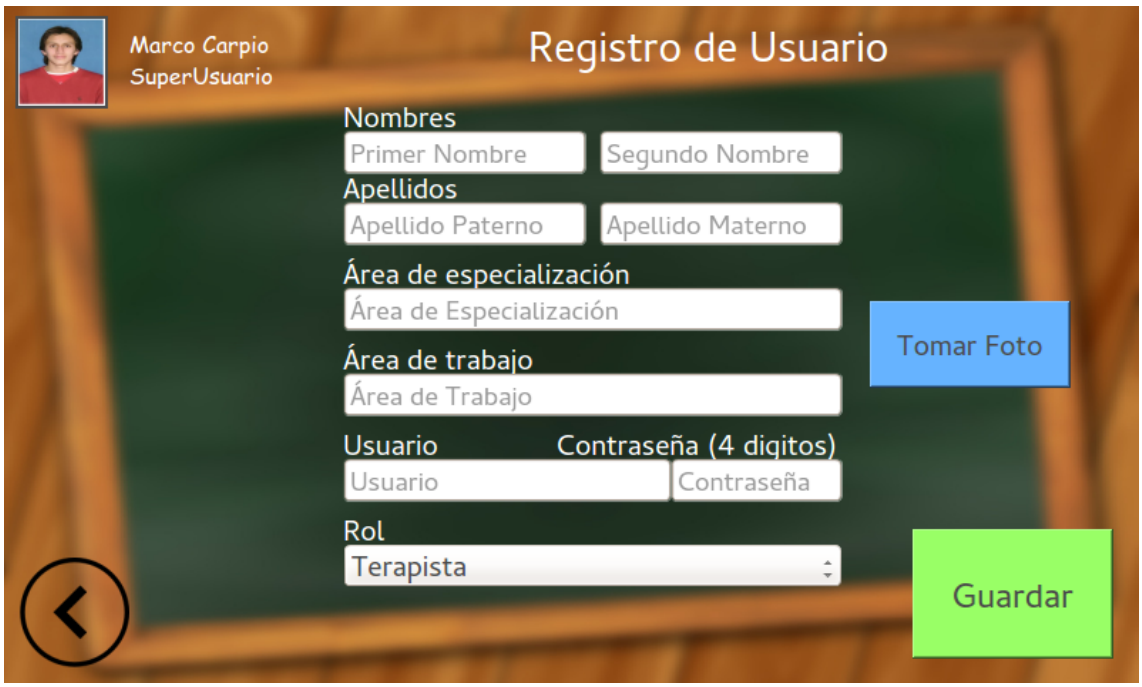

Figura 24: Ventana de registro de paciente

El manejo de los usuarios del igual forma está restringido según el rol de usuario, por lo que un *«Terapista»* no podrá crear un nuevo usuario, un *«Administrador»*, podrá crear y modificar usuarios pero solamente de tipo *«Terapista»*, mientras que el *«SuperUsuario»* podrá crear y modificar usuarios de tipo *«Terapista»* , *«Administrador»* e incluso otro *«SuperUsuario»*.

Se debe tener en cuenta que ni en pacientes ni en usuarios, se tiene la opción de eliminar con el fin de mantener la integridad de la base de datos, ya que eliminar un registro implica cambiar todos los índices de la base de datos.

4.4.6. Terapia

Para el desarrollo de esta parte de la aplicación se consultó a expertos en el área de lenguaje con el fin de establecer las actividades que se realizan dentro de la terapia, teniendo como resultado el esquema que se muestra en la figura 25 donde se puede observar que la terapia de lenguaje en general tiene 4 grandes categorías: Lenguaje, Pragmática, Articulación y Oral-Motora, aterrizando específicamente sobre la parte de lenguaje se tiene 3 subcategorías: Fonológica, Morfosintáctica y Semántica. En este caso, se ha desarrollando un ejercicio para cada una de estas subcategorías, teniendo: *«Duración acústica»*, *«Reconocimiento de Categorías»* y *«Discriminación Semántica»* correspondientes a Fonológica, Morfosintáctica y Semántica respectivamente.

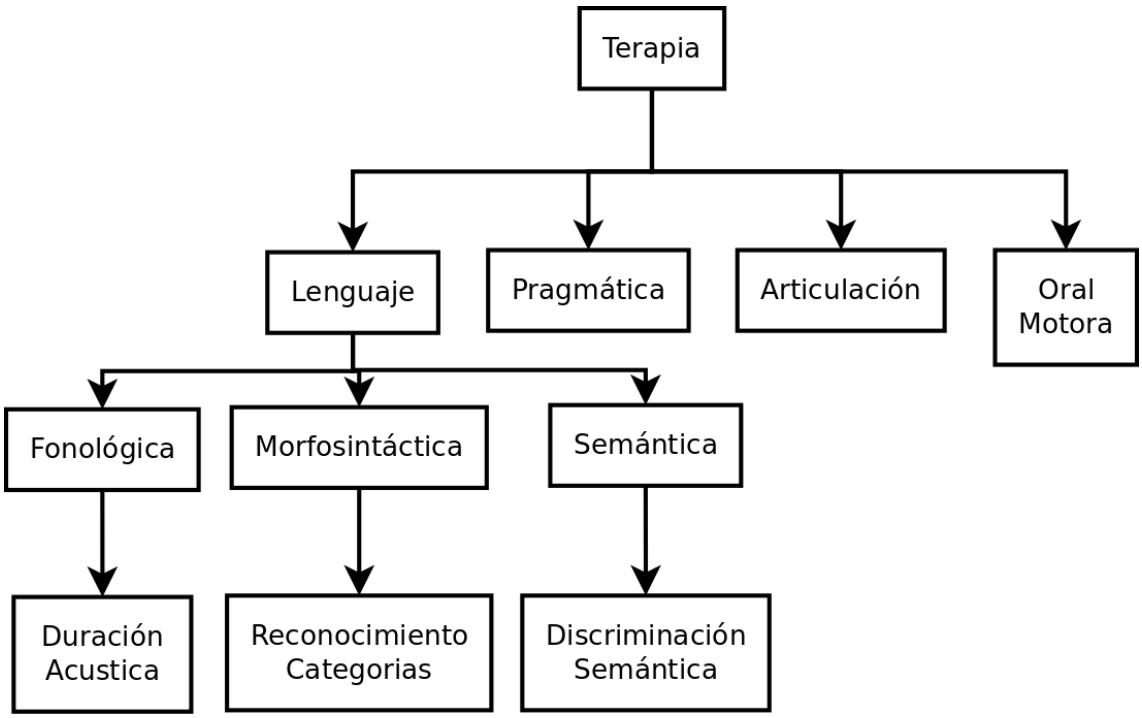

Figura 25: Esquema de actividades de terapia

El ejercicio de *«Duración acústica»* tiene como objetivo que el paciente pueda diferenciar de entre dos palabras dadas cuál es la mas larga. Para ello, se cuenta con un estimulo visual y auditivo, en el caso de que la palabra esté asociada a un sonido específico, como por ejemplo la palabra «perro», está asociada a un ladrido, se reproduce tanto el ladrido como la palabra. Para acceder a este estímulo se debe pulsar el botón que se encuentra debajo de la imagen correspondiente, como se muestra en la figura 26. Una vez que el paciente haya recibido los estímulos necesarios, el paciente debe pulsar la imagen que corresponda a la palabra más larga, en el caso que el paciente este imposibilitado de pulsar puede ser ayudado por el terapista, a continuación se evalúa si la respuesta fue correcta o no. En caso de ser correcta se aumenta 2 puntos en la calificación final y se cambia de palabras, caso contrario se resta 1 punto de la misma y se procede a dar una nueva oportunidad para seleccionar la palabra correcta, si después de 4 oportunidades no se ha logrado pulsar la palabra correcta, automáticamente se cambia de palabras, la actividad termina después de realizar 5 veces el ejercicio con palabras elegidas de manera aleatoria.

Figura 26: Ventana de Duración Acústica

Para todos los ejercicios la dinámica es la misma, es decir, elegir la palabra correcta, cambiando el criterio en cada caso.

En el caso de *«Reconocimiento de Categorías»* se presenta dos palabras de diferentes categorías, con sus respectivas imágenes y sonidos, siendo el objetivo que el paciente pueda elegir la categoría que se muestra en el enunciado, en el caso de la figura 27, se tiene que elegir cuál de las dos palabras corresponde a un «Vehículo», siendo la palabra de la izquierda la correcta.

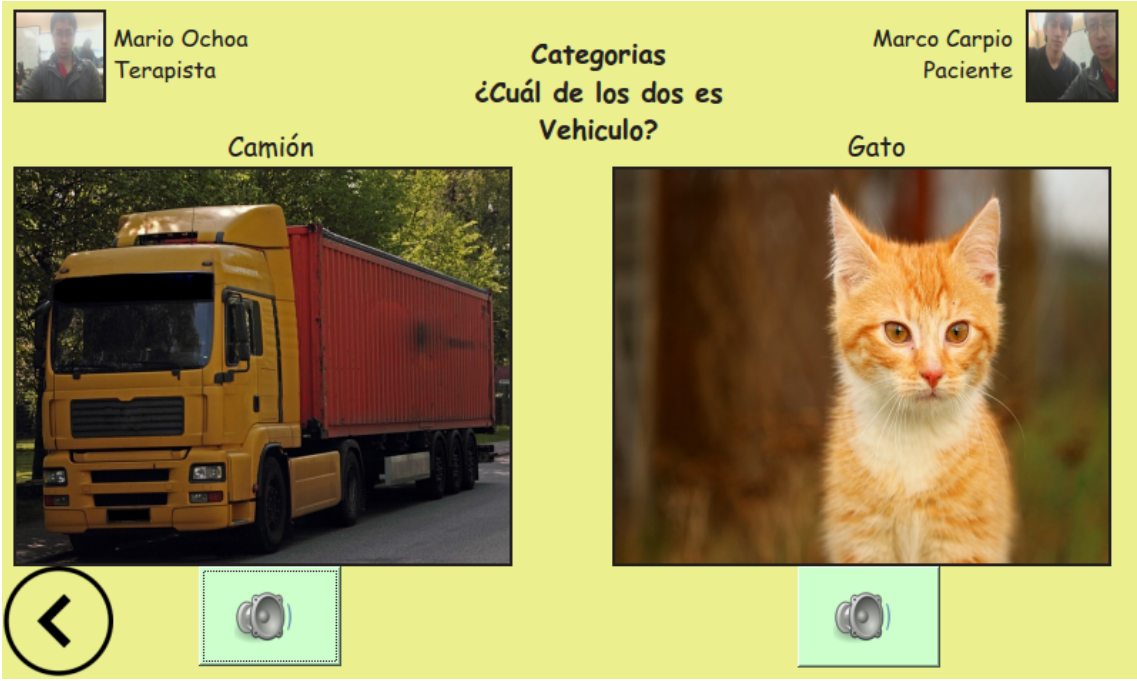

Figura 27: Ventana de Reconocimiento de Categorías

Finalmente, en el caso de *«Discriminación Semántica»* se presenta tres palabras de diferentes campos semánticos, de igual forma con sus respectivas imágenes y sonidos, el objetivo en este caso es que el

paciente pueda elegir la palabra que corresponde al enunciado. Por ejemplo, en la figura 28 se debe escoger la imagen que corresponda a «Gallina», siendo la imagen de la izquierda la correcta, se debe notar que este caso se omite el estimulo textual, con el fin de evaluar la discriminación en lugar de la comparación que podría producirse al mantener este estímulo.

Figura 28: Ventana de Discriminación Semántica

Si bien es cierto, no se ha abarcado todas las actividades que se pueden realizar en la terapia de lenguaje dado que son muchas alternativas, dependiendo de los antecedentes del paciente, se tiene un esquema general sobre el cual se puede agregar actividades en trabajos futuros.

4.4.7. Registros

Todas la actividades realizadas se almacenan el los registros con el fin de que el terapista pueda acceder a ellos en cualquier momento y plantear nuevos planes de terapia basado en los mismos. Como se puede ver en la figura 29, el usuario puede filtrar según la fecha de la sesión, y según criterios como: Nombre de Paciente, Apellido de Paciente, Nombre de Terapista, Apellido de Terapista y Nombre de Ejercicio.

Figura 29: Ventana de registros de las diferentes sesiones realizadas con los pacientes.

La información que se almacena en los registros corresponde a cada ejercicio realizado, por lo que el terapista puede ver en detalle la puntuación, observaciones del ejercicio, observaciones de la sesión, y las palabras que ha fallado como se puede observar en la figura 30.

Figura 30: Ventana registro de las sesiones, donde se ha seleccionado una sesión específica (detalle del registro).

4.4.8. Modo Autónomo y Control remoto

Estos módulos específicamente son los únicos en los que no se tiene interacción directa con la pantalla táctil, por tanto se tiene simplemente una imagen de fondo acorde con el traje que ocupa el robot, como se

puede ver en la figura 31.

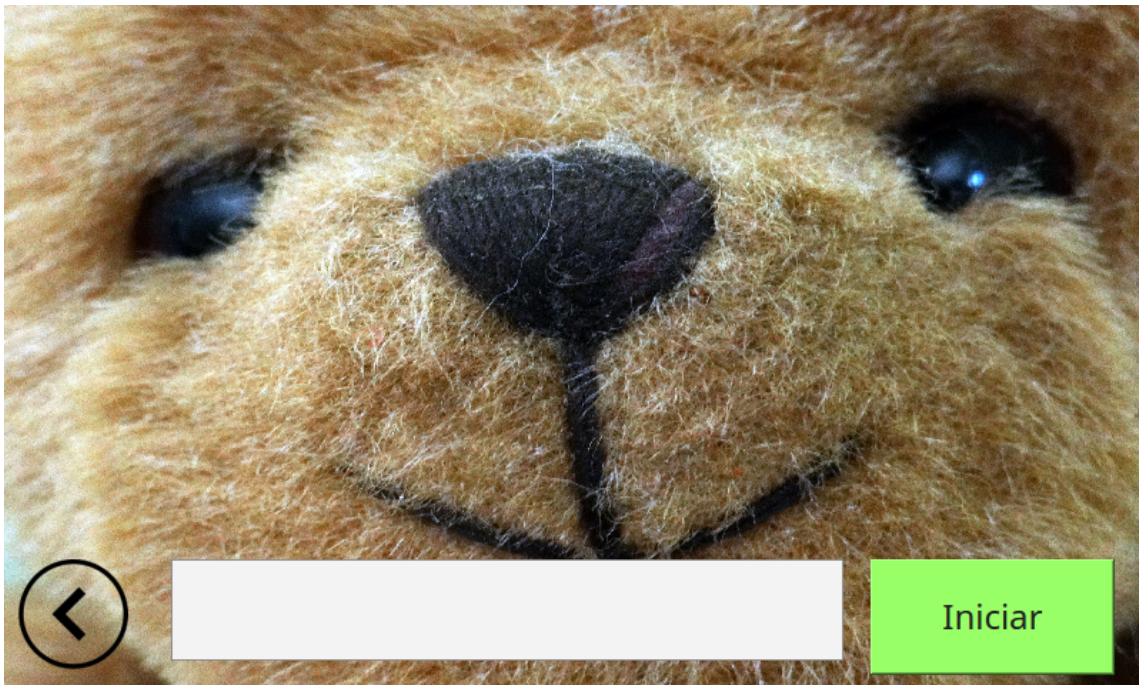

Figura 31: Ventana de Modo Autónomo y Control Remoto

Se debe tener en cuenta que tanto el Modo Autónomo como el Control Remoto necesitan tener una conexión a internet, por lo que previo a iniciar cualquiera de estos dos modos, el sistema hace una comprobación de sus estados de conexión, en caso de no contar con acceso a internet, no será posible iniciar estos modos.

Al iniciar el modo autónomo, el robot queda a la escucha de los comandos de voz programados, en este caso. Para inicar, el robot queda a la espera de un «Hola» y en el caso de escuchar esta palabra, saludará y quedará en escucha activa, es decir, cualquier palabra pronunciada será procesada como un comando. La funcionalidad dada en este caso es la de obtener conceptos de palabras a manera de un diccionario, por tanto, si el usuario dice «Buscar», el robot responderá «Qué palabra deseas buscar?», en este momento el usuario debe pronunciar la palabra de la cual desea obtener su significado o concepto, finalmente el robot confirma la palabra a buscar y en el caso de que se encuentre dicho concepto, procede a leerlo y a continuación queda en escucha pasiva nuevamente; si el usuario pronuncia «Adiós», automáticamente termina el modo autónomo, en el diagrama de flujo de la figura 32 se muestra en detalle este proceso.

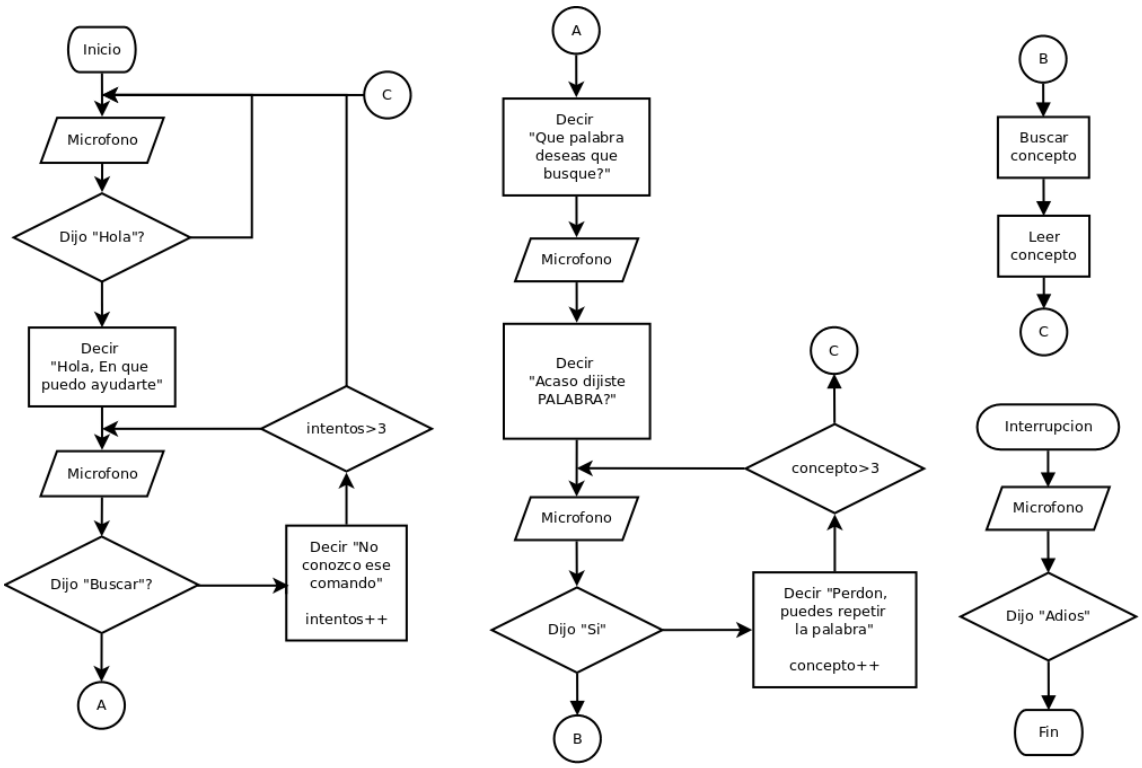

Figura 32: Diagrama de flujo del Modo Autónomo

Por otro lado, en cuanto al control remoto, cuando se pulsa en iniciar se activa el servidor dentro del Raspberry y se muestra un mensaje con la dirección IP a la cual se puede conectar para acceder a las funciones de control remoto, funciones que ya se mencionaron en el apartado de «Desarrollo del módulo de control mediante interfaz web».

4.4.9. Mantenimiento

Como se mencionó anteriormente las funciones de mantenimiento le pertenecen exclusivamente al «SuperUsuario», entre estas funciones se encuentran las siguientes: Revisar el estado de la cámara, parlante, micrófono y motores, prender y apagar el servidor y finalmente, puede desplegar una ventana con la línea de comando del sistema operativo con el fin de poder modificar con acceso total cualquier parte del programa, es por ello que este rol lo tienen solamente los miembros que estén encargados de dar mantenimiento al robot y tengan conocimiento de la estructura de la aplicación, en el gráfico de la figura 33, se puede ver el esquema de las funciones del «SuperUsuario».

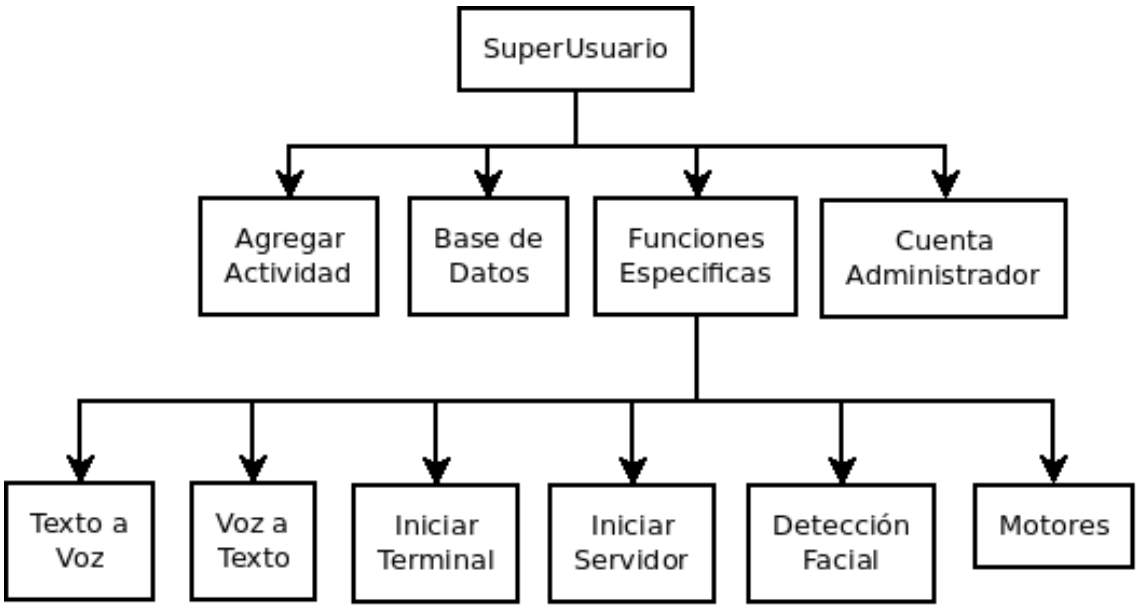

Figura 33: Funciones del SuperUsuario

5. Experimentación y resultados

En los experimentos realizados se prueban las funcionalidades especificas del robot, como tiempo de ejecución, consumo del procesador y el consumo de memoria, en cada una de las etapas de la aplicación ya que estos datos pueden ser de gran utilidad para la optimización de procesos para las siguientes versiones del prototipo.

En la figura 34 se muestra una gráfica acerca de tiempo de ejecución, donde se puede ver que el encendido y apagado del robot son los procesos con más duración, siguiendo con el proceso de «Ingreso a terapia» teniendo un tiempo de 7,3 segundos, dado a que el sistema carga todas las imágenes, sonidos y palabras de la base de datos y se elige 10 al azar para realizar la terapia. Este proceso puede ser optimizado evitando cargar todas las palabras con su respectivo contenido multimedia, haciendo para ello un preprocesamiento de los requerimientos y cargando solamente palabras específicas, lo cual reduciría a 3,23 segundos como su puede ver en el proceso de «Ventana-Terapia». Otros procesos a tener en cuenta son los de «Hablar-Control» y «Cámara-Control», teniendo tiempos de ejecución de 1,87 y 2,9 segundos respectivamente, que sigue siendo un tiempo considerable, y es producto de los procesos adyacentes que conllevan estas acciones. Por ejemplo, la recepción y procesamiento de las peticiones de las páginas web hacia el servidor, y sobre todo el encendido de los periféricos tanto el parlante como la cámara consumen tiempos y recursos importantes. En cuanto a los otros procesos se puede ver que el tiempo del bastante bajo, por lo que no sería necesario una intervención urgente.

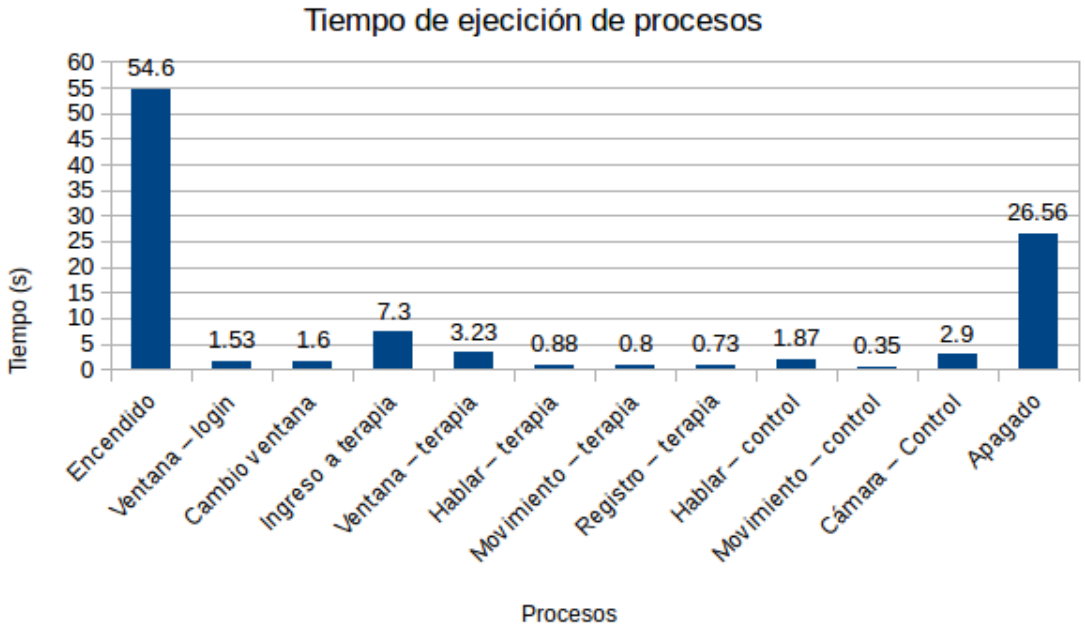

Figura 34: Tiempo de ejecución de procesos

En cuanto al uso del CPU se ha seguido en secuencia el proceso para realizar una terapia, revisar los registros y utilizar el control remoto, teniendo los resultados que se muestran en la figura 35, se puede observar que los procesos que más uso hacen de CPU son el «Iniciar el control remoto» con un consumo del 75,2 %, seguido por el uso de la «Cámara» con un consumo de 48,5 %. Por ello, optimizar estos procesos implica profundizar en campos de manejo de periféricos, lo cual está fuera de los alcances de este trabajo. Por otra parte se puede ver un 43,9 % en el proceso de «Ingreso a terapia», las razones de este consumo son las mismas que provocan un tiempo de ejecución de 7,3 segundos explicadas en el apartado anterior; en promedio se tiene un consumo de 29,25 %, lo cual se encuentra dentro de los margenes normales de cualquier aplicación que usa el entorno gráfico del Raspberry.

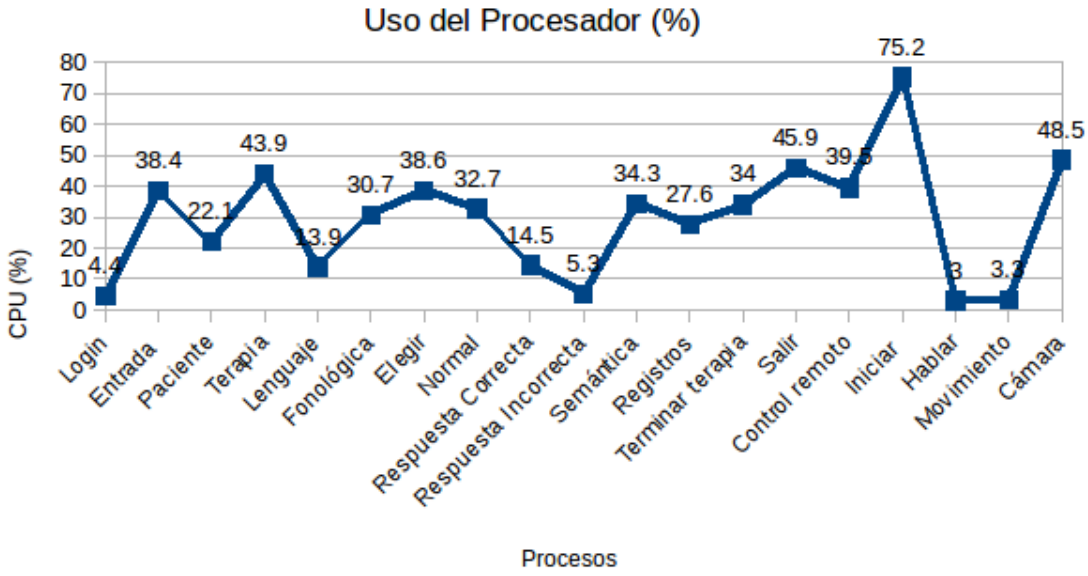

Figura 35: Uso de procesador

La última prueba realizada está relacionada con el uso de la memoria del Raspberry, se ha seguido la misma secuencia del caso anterior, teniendo los resultados que se presentan en la figura 36 donde se puede observar que el consumo de memoria aumenta a medida que se avanza en los procesos, con excepción del «Ingreso a terapia» y el uso de la «Cámara» en el control remoto contrario a al caso anterior, por lo que contrastando las figuras 35 y 36 se puede ver que en estas operaciones específicas todo el consumo se vuelca hacia el procesador, por lo que la memoria disminuye su consumo. En promedio el consumo es de 11,76 %, que de igual forma está dentro de los márgenes normales para este tipo de aplicaciones.

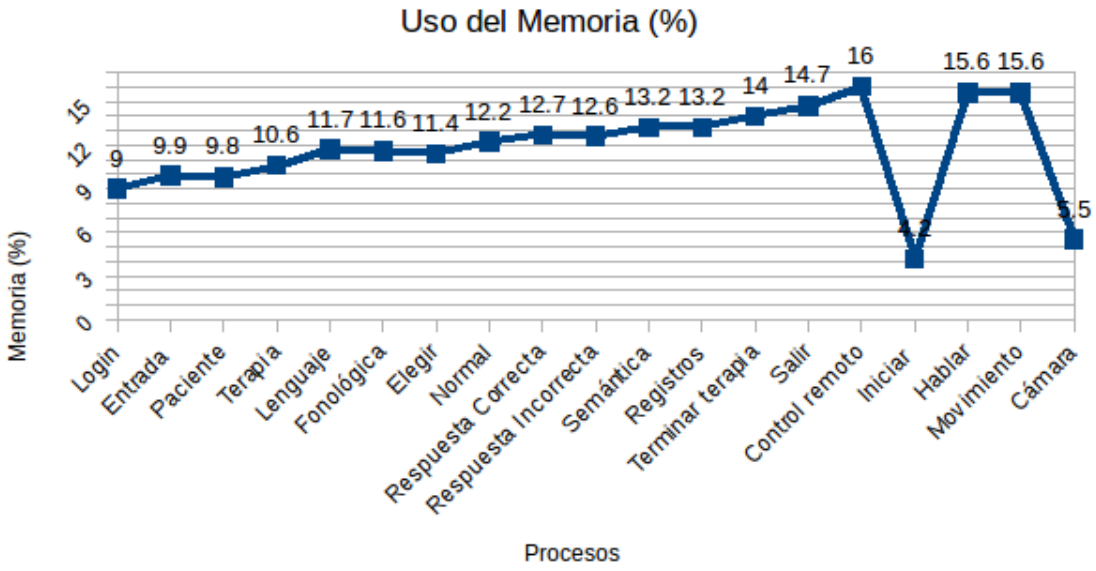

Figura 36: Consumo de Memoria

6. Conclusiones

El desarrollar un prototipo que cubra necesidades especificas, como en nuestro caso la terapia de lenguaje, conlleva realizar una lluvia de ideas sobre las funcionalidades que tendrá el producto final, y por ello es muy importante contar con la asesoría de expertos en el área y sobre todo con el usuario final, con el fin de que las funcionalidades del producto estén acordes con las necesidades del usuario.

Específicamente en el ámbito de la construcción del robot, es importante saber usar la tecnología que está a nuestro alcance, por ello se ha visto que la impresión 3D es una de las tecnologías más explotadas en los últimos años, ya que permite construir cualquier prototipo sin importar las complejidad del mismo, centrando el trabajo en el diseño del mismo. A pesar de tener un muy buen acabado, fue necesario revestir el prototipo, con el fin de que se vea estéticamente y sea amigable, ya que el robot está enfocado para trabajar con niños.

Por otro lado, en cuanto a la circuitería electrónica, se pudo observar que es de vital importancia aislar los circuitos de control frente a los circuitos de potencia, ya que un excesivo consumo de corriente por parte de cualquier elemento puede ocasionar que colapse el sistema de alimentación, es por ello que nos vimos en la necesidad de utilizar 2 baterías, aspecto que resultó favorable tanto en tiempo de uso como en la seguridad frente a interferencias de corrientes, ya que los bancos de baterías cuentan con sus sistemas de seguridad que evitan el excesivo consumo apagando y bloqueando inmediatamente todo el sistema.

Por otro lado, en lo referido al software se pudo notar un buen desempeño del sistema al usar en todos sus módulos lenguaje Python, ya que Raspberry está desarrollado para explotar las prestaciones de este lenguaje de programación, sin embargo, se realizaron pruebas de ciertos módulos del sistema en lenguaje C++ y se pudo observar un consumo de recursos menor, por lo que se puede concluir que necesariamente para sistemas embebidos lo mejor es usar un lenguaje de programación de bajo nivel.

En uso de librerías ya desarrolladas representa una gran ventaja para el programador, ya que se puede confiar plenamente en las funcionalidades de dichas librerías, una de las más importantes para el desarrollo de este proyecto fue «OpenCV», la cual nos permitió desarrollar las funcionalidades de visión artificial. Una ventaja importante es que OpenCV tiene compatibilidad multiplataforma, incluyendo Raspbian, el sistema operativo de Raspberry.

A pesar de las grandes ventajas que presenta «OpenCV», existen factores que tecnológicamente aún no se pueden controlar en cuanto a visión artificial, uno de los más importantes es la iluminación, ya que un ligero cambio de iluminación puede provocar que el sistema, en este caso, deje de detectar los rostros.

En cuanto al desarrollo de la conversión de texto a voz, «Festival» es una de las mejores herramientas que se tiene, ya que es una de las pocas que cuenta con un módulo compatible con el español.

Por parte de la aplicación final, es necesario tener un esquema definido antes de realizar la aplicación con el fin de delimitar el trabajo y no alejarse de los objetivos, además es necesario documentar de manera detallada todas las partes que conforman la aplicación, con la finalidad de que sea posible dar mantenimiento en caso de ser necesario.

7. Recomendaciones

A continuación se citan algunas recomendaciones importantes que en su momento fueron muy útiles para el desarrollo de este trabajo y que de seguro serán de gran utilidad para posteriores trabajos similares.

Empezando con la parte de hardware se recomienda verificar que se de un soporte por parte del fabricante en cuanto a los controladores, ya que en el caso de la película táctil de la pantalla no se tenía los controladores necesarios para el correcto funcionamiento, por lo que se procedió a obtener los datos directamente del protocolo de comunicación SPI usado por el controlador de la pantalla táctil y utilizar esos datos para modificar la ubicación del puntero del mouse del sistema operativo.

Es importante usar banco de baterías confiables, con la finalidad de asegurar la estabilidad del sistema, ya que en nuestro caso, el fabricante establece una corriente de salida por cada puerto de $1A$ y de $2A$, pero cuando se realizó pruebas de consumo, la corriente máxima de salida fue de 700mA.

Se recomienda en caso de usar algún dispositivo que cuente con un sistema Operativo como en el caso de Raspberry y su sistema operativo Raspbian, realizar el desarrollo de las aplicaciones en una computador con un sistema operativo compatible, por ejemplo Ubuntu, ya que esto permite desarrollar y probar los módulos en una PC antes de pasarlos a Raspberry, lo que ahorra mucho tiempo de desarrollo.

En caso de usar Python como lenguaje de programación se recomienda usar «sqlite» como gestor de base de datos, ya que cuenta con el soporte necesario para realizar todas las operaciones necesarias para bases de datos relacionales, sin embargo, se puede usar cualquier otro gestión y usar un framework compatible con su lenguaje de programación.

En cuanto al manejo de los puertos de entrada y salida digitales del Raspberry, se puede recomendar el uso de las librerías que generan las señales mediante hardware ya que aquellas librerías que generan las señales por medio del software pueden verse afectadas por los procesos adyacentes ejecutándose en Raspberry.

8. Trabajo futuro

Como se mencionó anteriormente, se realizaron tanto pruebas de laboratorio como pruebas de campo, donde se recogieron algunas recomendaciones e incluso cambios importantes que se debe dar al prototipo, la cuales se citan a continuación con el fin de que en posteriores intervenciones sobre este trabajo de investigación se puedan llevar a cabo.

Una de las más importantes y más urgentes modificaciones se de debe dar al prototipo es usar una pantalla de táctil de tipo capacitivo, ya que al realizar las pruebas con la pantalla táctil resistiva hubo casos en los cuales los niños con discapacidades motoras e incluso aquellos sin dicha discapacidad se les hacia difícil pulsar en zonas pequeñas, debido a que las pantallas táctiles resistivas están concebidas para usar un puntero en lugar de los dedos.

En cuanto al diseño de la estructura se puede realizar un modelo más compacto, mejorando así la estética del robot e incluso ahorrando material y tiempo en la etapa de construcción.

Como se mencionó en las conclusiones, en sistemas embebidos la opción más eficiente es usar lenguajes de bajo nivel, por lo que en un trabajo futuro se puede migrar el sistema al lenguaje de programación C o C++ o alguna de sus partes más críticas.

En cuanto a la placa se puede usar un gestor de baterías, con la finalidad de manejar de mejor manera la carga de las baterías, además de poder enviar los datos de consumo y otros datos técnicos hacia el Raspberry y poder generar planes de ahorro de energía o apagados automáticos y seguros en caso de que la batería este demasiado baja.

Se puede notar que existe una dependencia de una conexión a internet en los módulos de Control Remoto y Reconocimiento de voz, por lo que se puede atacar estos puntos de la siguiente manera: en cuanto al control remoto solamente es necesario una conexión a una red LAN más no es necesario una conexión directa a internet, por lo que se puede plantear el uso de un punto de acceso o «Hotspot» creado internamente en el Raspberry y por otro lado, en cuanto al reconocimiento de voz, se puede realizar investigaciones e implementar los Modelos Ocultos de Markov para realizar un reconocimiento si necesidad de internet, aunque sería necesario realizar un análisis de consumo de recursos en el Raspberry para ver si es conveniente aplicar esta técnica estadística para dicho objetivo.

9. Bibliografía

Referencias

[1] F. Amirabdollahian, B. Robins, K. Dautenhahn, and Z. Ji, "Investigating tactile event recognition in child-robot interaction for use in autism therapy," in *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, August 2011, pp. 5347 – 5351. 3

[2] CONSEJO NACIONAL DE IGUALDAD DE DISCAPACIDADES, "Estadística Personas con discapacidad: Registro nacional de discapacidades," 2016, consulta: Febrero-2016. [Online]. Available: 'http://www.consejodiscapacidades.gob.ec/wp-content/uploads/downloads/2016/02/estadisticas{_}discapacidad.pdf' 1

[3] Croston Ben, "raspberry-gpio-python," consulta: Febrero-2016. [Online]. Available: https://sourceforge.net/projects/raspberry-gpio-python/ 4.3.2

[4] DFRobot, "Control Specification," 2015, consulta: Febrero-2016. [Online]. Available: http://www.dfrobot.com/image/data/SER0038/SER0038{_}ControlSpecification.png 4.3.2

[5] L. Dickstein-Fischer and G. S. Fischer, "Combining psychological and engineering approaches to utilizing social robots with children with Autism," in *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE*, August 2014, pp. 792 – 795. 3

[6] elementzonline, "Getting started with PyCharm and Qt 4 Designer," 2015, consulta: Febrero-2016. [Online]. Available: https://elementztechblog.wordpress.com/2015/04/14/getting-started-with-pycharm-and-qt-4-designer/ 4.4.3

[7] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier, "Mechatronic design of NAO humanoid," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, May 2009, pp. 769 – 774. 3

[8] Itseez, "OpenCV," 2016, consulta: Febrero-2016. [Online]. Available: http://opencv.org/ 4.3.1

[9] joan2937, "pigpio," consulta: Febrero-2016. [Online]. Available: https://github.com/joan2937/pigpio 4.3.2

[10] D. Jones, "Picamera," 2014, consulta: Febrero-2016. [Online]. Available: https://picamera.readthedocs.org/en/release-1.10/ 4.3.1

[11] La forja de guadalinex, "Proyecto Hispavoces," consulta: Febrero-2016. [Online]. Available: http://forja.guadalinex.org/frs/?group{_}id=21 4.3.3

[12] H. Lee and E. Hyun, "The intelligent robot contents for children with speech-language disorder," *Journal of Educational Technology & Society*, vol. 18, no. 3, pp. 100 – 113, 2015. 3

[13] QtCompany, "Qt Designer Manual," 2016, consulta: Febrero-2016. [Online]. Available: http://doc.qt.io/qt-4.8/designer-manual.html 4.4.3

[14] Riverbank, "PyQt," 2015, consulta: Febrero-2016. [Online]. Available: https://www.riverbankcomputing.com/software/pyqt/intro 4.4.3

[15] A. Ronacher, "Flask," 2014, consulta: Febrero-2016. [Online]. Available: http://flask.pocoo.org/ 4.3.5

[16] S. Shamsuddin, H. Yussof, L. Ismail, F. A. Hanapiah, S. Mohamed, H. A. Piah, and N. I. Zahari, "Initial response of autistic children in human-robot interaction therapy with humanoid robot NAO," in *Signal Processing and its Applications (CSPA), 2012 IEEE 8th International Colloquium on*, March 2012, pp. 188 – 193. 3

[17] The Centre for Speech Technology Research - The University of Edinburgh, "The Festival Speech Synthesis System," consulta: Febrero-2016. [Online]. Available: http://www.cstr.ed.ac.uk/projects/festival/ 4.3.3

[18] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. I – 511. 4.3.1

[19] J. Wainer, B. Robins, F. Amirabdollahian, and K. Dautenhahn, "Using the Humanoid Robot KASPAR to Autonomously Play Triadic Games and Facilitate Collaborative Play Among Children With Autism," *IEEE Transactions on Autonomous Mental Development*, vol. 6, no. 3, pp. 183 – 199, Sept 2014. 3

[20] A. Zhang, "Speech Recognition (version 3.2)." 2016, consulta: Febrero-2016. [Online]. Available: https://pypi.python.org/pypi/SpeechRecognition/ 4.3.4

10. Anexos

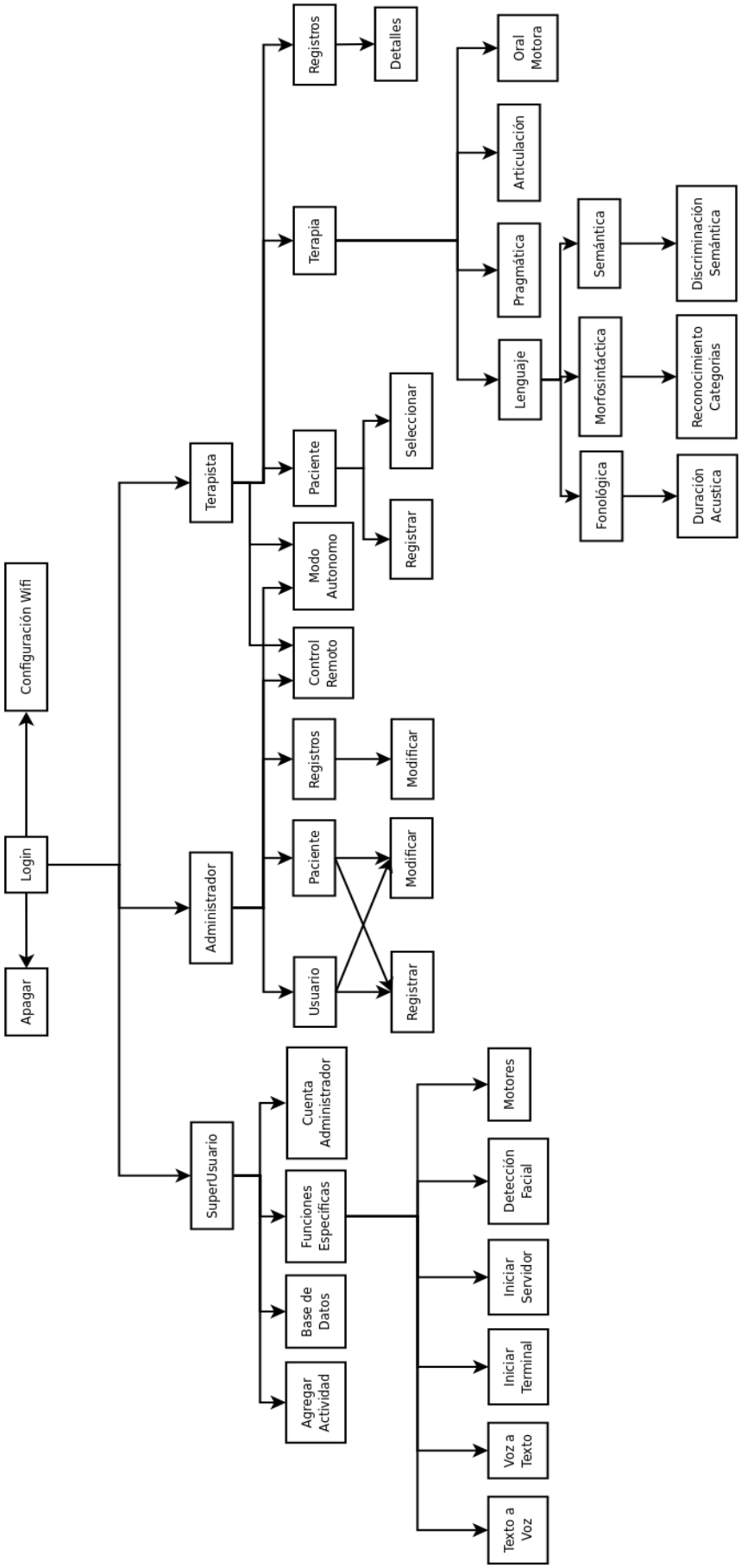

Figura 37: Diagrama Ventanas