UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

Desarrollo e incorporación de capacidad para envío-recepción de SMS y redes sociales a un sistema aumentativo y alternativo de comunicación

Tesis

QUE PARA OBTENER EL TÍTULO DE

INGENIERO ELÉCTRICO ELECTRÓNICO

PRESENTA:
Tirado Torres Ricardo Aldair

DIRECTOR DE TESIS:
Díaz Rangel Ismael

Ciudad Nezahualcóyotl, Estado de México, 2022

Agradecimientos

A mi padre, Antonio Tirado Durán, y a mi madre, América Ivonne Torres Main, por la vida y por enseñarme a vivirla, además del infinito amor y soporte que me han dado.

A mis abuelos, Dulce y Ricardo, por la crianza y amor que me dieron desde que era un infante.

Al Dr. Ismael Díaz Rangel, por todo el apoyo ofrecido durante la última etapa de la carrera.

A mi amigo Daniel, por siempre estar a mi lado durante la vida universitaria.

Por último, pero no por eso menos importante a todos mis familiares y amigos.

Investigación realizada gracias al Programa UNAM-PAPIIT IT103220.

Contenido

Capítulo 1. Introducción

La comunicación es la habilidad que facilita la transmisión de todo tipo de ideas, necesidades, emociones y sentimientos entre dos o más individuos, por lo que resulta indispensable que las personas interactúen con el mundo mediante un sistema de signos organizado y empleado por una comunidad, como lo son las palabras, ya sea lenguaje oral o escrito, los gestos, que se refiere a la comunicación no verbal como el lenguaje corporal, la mímica o los pictogramas.

Ahora bien, las personas se han planteado problemas e identificando necesidades a las que se les debe dar una solución pertinente; es por ello que la tecnología surge para facilitar y suplir a las necesidades básicas del ser humano. El desarrollo tecnológico siempre ha estado presente a lo largo de la historia de la humanidad, no obstante, en los últimos años, los avances se enfocan a resolver alguna problemática con soluciones cada vez más innovadoras.

En sociedad actual, se busca que todos los individuos tengan las mismas oportunidades sin importar sus condiciones físicas, mentales, sociales, etcétera; a lo cual se le denomina de manera general como inclusión. Sin embargo, en algunos casos las limitantes de algunas personas son muy específicas y afectan a una cantidad de gente no muy elevada, en comparación al número de habitantes en el mundo, debido a ello el interés de los grandes corporativos tecnológicos para crear soluciones de su ámbito no tiene interés comercial, por lo que no suelen hacer algo al respecto, y las pequeñas compañías que desarrollan sistemas tecnológicos para lograr una sociedad más incluyente, suelen ofrecer productos con precios elevados, dejándolos fuera del alcance de muchos de los posibles usuarios.

Para las personas que han perdido la capacidad del habla, existen soluciones que de manera genérica se denominan Sistemas Aumentativos y Alternativos de Comunicación (SAAC).

En la Facultad de Estudios Superiores Aragón se trabaja en un SAAC que busca ayudar a personas que además de tener limitada su capacidad de hablar, también tengan limitaciones en su motricidad fina; el sistema trabaja sobre una microcomputadora con un monitor. Actualmente es un prototipo funcional que permite generar oraciones mediante una interfaz a base de un alfabeto y otra con pictogramas.

Este trabajo consiste en incorporar al SACC mencionado, la capacidad de envío y recepción de SMS (Servicio de mensajes breves, Short Message Service, en inglés), así como de comunicación mediante WhatsApp y Telegram.

1.1 Objetivo general

Desarrollar e integrar a un sistema aumentativo y alternativo de comunicación un módulo que adicione capacidad para envío y recepción de mensajes vía telefonía y redes sociales.

1.2 Objetivos particulares

- Desarrollar un módulo para envío-recepción de mensaje por telefonía vía GSM, considerando un costo asequible sin comprometer la fiabilidad.
- Incorporar una agenda para agregar y eliminar contactos.
- Enviar y recibir mensajes en las redes sociales WhatsApp y Telegram.
- Permitir la consulta del historial de mensajes GSM, WhatsApp y Telegram.
- Posibilitar la elección de un contacto particular de la agenda para envío y visualización de mensajes.
- Diseñar las interfaces gráficas de las nuevas funcionalidades para su integración homogénea al SAAC existente.

1.3 Actividades

- Investigar qué son los SAAC.
- Revisar las condiciones de las personas que requieren el uso de un SAAC.
- Indagar sobre la telefonía celular en México, analizando los precios y beneficios de cada una de las compañías.
- Revisar las opciones de módulos para la recepción y trasmisión de mensajes sobre redes de telefonía.
- Estudiar la compatibilidad del dispositivo de comunicación con las diferentes de compañías de telefonía celular.
- Aprender codificación y creación de interfaces gráficas con Python.
- Llevar a cabo pruebas con el dispositivo de comunicación para el envío y recepción de mensajes vía SMS.
- Crear un programa en Python para enviar y recibir mensajes SMS.
- Incorporar la opción de agregar/eliminar contactos.
- Diseñar una interfaz gráfica para la gestión de contactos.
- Investigar y adecuar métodos para revisar el histórico de mensajes.
- Crear una interfaz gráfica para la selección y visualización de un mensaje histórico.
- Revisar información acerca del envío de mensajes utilizando aplicaciones como WhatsApp y Telegram empleando Python.
- Efectuar pruebas para enviar y recibir mensajes a través de WhatsApp y Telegram.
- Crear una interfaz gráfica para la selección y visualización de un mensaje histórico.
- Incorporar una interfaz completa que incluya todas las funcionalidades de las opciones de mensajería.
- Integrar la interfaz con todas sus características al sistema aumentativo y alternativo de comunicación.

1.4 Justificación

Una meta de toda sociedad es la de construir un mundo donde todos sus habitantes tengan las mismas oportunidades sin importar su discapacidad, etnia, edad, religión, identidad de género, etcétera. Hay casos donde las personas padecen condiciones o enfermedades que además de disminuir sus capacidades, sufren severamente de manera física y/o mental; pero no solo ellos, ya que también sus personas cercanas pueden ver drásticamente mermada su calidad de vida; por ejemplo, en algún momento, las personas con Esclerosis Lateral Amiotrófica (ELA), pierden la capacidad de hablar y escribir, aunque cognitivamente se encuentren en plenitud; es decir, viven atrapados en un cuerpo disfuncional, lo cual provoca una gran angustia a quienes deben cuidar de ellos, entre otras cosas por lo limitada que resulta la comunicación.

Como el caso mencionado hay más, y trabajar en propuestas que tengan como propósito mejorar la calidad de vida de sectores de la sociedad que no son el foco de interés de las grandes corporaciones, es algo que siempre valdrá la pena el esfuerzo, y más en ambientes académicos donde se busca fomentar, además del conocimiento, principios que hagan de los estudiantes, mejores individuos para la sociedad.

1.5 Antecedentes

En este apartado primero se hablará del SAAC sobre el cuál se parte en este trabajo para sumarle capacidades, y después se mencionarán antecedentes históricos de diversas propuestas de SAAC.

En 2016 fue cuando, en grupo IDEA de la Facultad de Estudios Superiores Aragón, el alumno Giovanni Belli junto con profesores identificaron una problemática y comenzaron a definir el desarrollo de un SAAC, el objetivo fue crear un sistema considerado de alta tecnología con la meta de que tuviese un costo que pudiese ser accesible para los posibles pacientes, ya que, al investigar, se encontró que los SAAC de alta tecnología tienen costo elevados. Además, el sistema debía ser útil para quienes además de tener limitada capacidad del habla, tuviesen también limitantes en su motricidad fina. La propuesta fue desarrollada sobre una microcomputadora de muy bajo costo (Raspberry pi), como se muestra en la ilustración 1.1, y con un sistema operativo y herramientas de desarrollo de uso libre, dando como resultado un SAAC con una interfaz alfabética. Para usarlo, un cursor se mueve de manera automática sobre el alfabeto de fila en fila, y al momento de pasar por fila donde se encuentra la letra requerida, el paciente debe presionar un botón, por lo que el cursor se mueve en columnas, y al pasar por la letra deseada, nuevamente presiona el botón, con ello es posible formar palabras y oraciones, además del mensaje escrito (que se muestra en un campo dentro de la interfaz), también es posible que se escuche de manera verbal(ilustración 1.2). Este sistema fue presentado como trabajo de titulación [1].

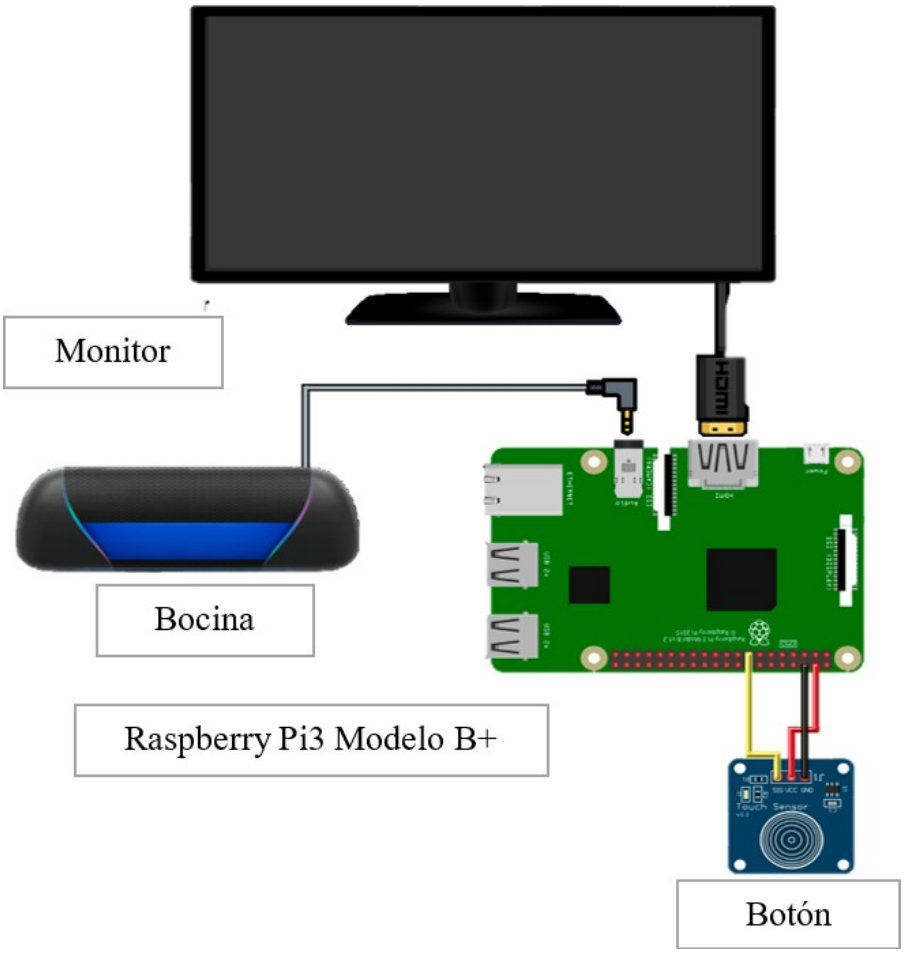


Ilustración 1.1 Elementos del SAAC.

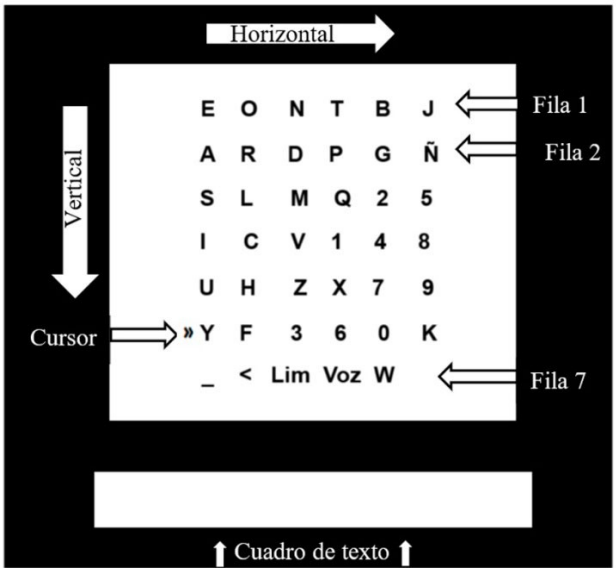


Ilustración 1.2 SAAC con interfaz alfabética.

Posteriormente se incorporó un nuevo tipo de interfaz a base de pictogramas (ilustración 1.3), preservando la opción del alfabeto, el cual permite escribir cualquier frase, y la nueva opción pictográfica, está limitada a poco más de 400 oraciones predefinidas, pero tiene la ventaja de que la cantidad de "clics" es reducida, por lo que se puede escribir de manera más rápida, aunque cabe mencionar que a la interfaz alfabética también se le incorporó un predictor de texto, lo cual agiliza la escritura. Las nuevas funcionalidades también fueron presentadas como un trabajo de titulación [2].

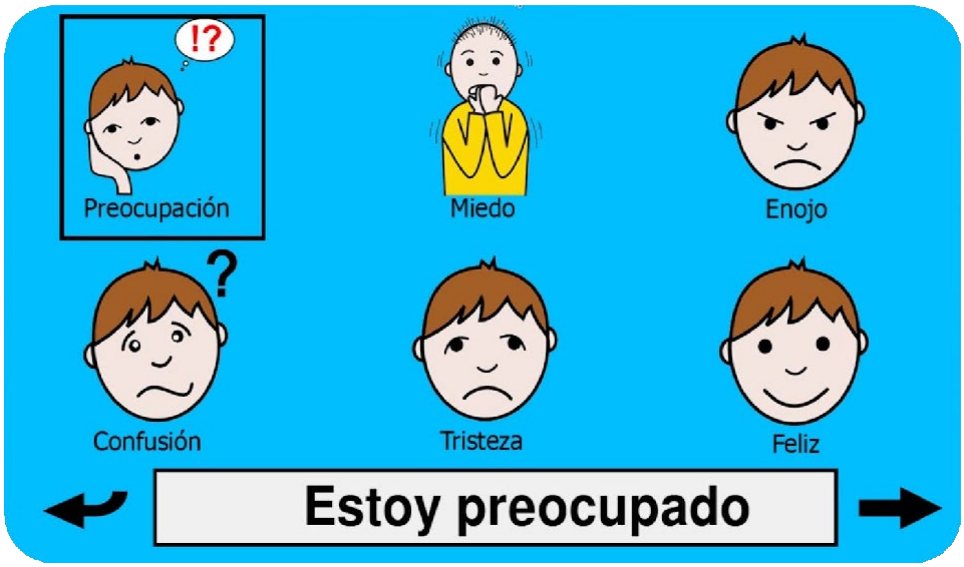


Ilustración 1.3 SAAC con interfaz pictográfica.

La interacción con esta interfaz es semejante a la alfabética; es decir, un cursor se desplaza automáticamente por los pictogramas, que inicialmente representan categorías, y al seleccionar uno, se presenta una nueva plantilla de pictogramas, al ir realizando elecciones, en la parte inferior de la interfaz, se puede observar la conformación de oraciones, eventualmente se puede crear una nueva oración o hacer que el SAAC la verbalice.

Ahora se hablará de antecedentes históricos de los SAAC.

"Los sistemas aumentativos y alternativos de comunicación (SAAC) tienen un pasaje histórico bastante breve, puesto que antes de 1975, había descripciones de personas que no realizaban una comunicación verbal, sin embargo, empleaban tableros de letras simples, tableros con imágenes y máquinas de escribir para manifestar sus mensajes.

En una edad temprana, los SAAC se limitaron a un número reducido de personas con afasia, un trastorno causado por lesiones en las regiones del cerebro que controlan el lenguaje, que tenían la capacidad de señalar símbolos de forma independiente. No fue hasta el año de 1975, cuando el Congreso de los Estados Unidos promulgó la Ley de educación de todos los niños discapacitados (Ley pública 94-142 o PL 94-142), que demanda que todos los niños discapacitados entre 5 y 21 años de edad reciban educación pública gratuita. Antes de esto, no

había una ley obligatoria de educación especial, por lo que las personas que no tenían la capacidad de hablar y que anteriormente habían sido excluidas de la educación general, comenzaron a recibir la debida atención de educadores del habla y el lenguaje" [3].

Un factor que influyó en sobremanera al campo de los SAAC, fue el desarrollo de las microcomputadoras, ya que, antes de este desarrollo tecnológico, los dispositivos aumentativos y alternativos de comunicación (AAC) eran sistemas eléctricos rústicos. Muchos de estos primeros sistemas consistían en adaptaciones a máquinas de escribir eléctricas o eran simplemente dispositivos de escaneo electrónico, sin alguna salida hablada o impresa.

Uno de los primeros dispositivos comerciales de comunicación aumentativa fue Canon Communicator, diseñado por la corporación Canon, específicamente para personas con impedimentos del habla. El dispositivo funcionaba con baterías, se sujetaba a la muñeca del usuario y tenía el tamaño de un teléfono celular. Contaba con un teclado con las letras en orden alfabético, además de poseer un sistema que memorizaba y almacenaba frases y palabras de la última persona que lo utilizó. De igual forma, se podía realizar impresiones a través de una pequeña impresora de tiras. La desventaja más clara de este sistema era que los usuarios contaran con suficiente ortografía y con habilidades motores finas, para presionar los paneles.

Un sistema electrónico de comunicación aumentativa y alternativa que se adaptó a una gran variedad de usuarios fue el AutoCom. En su momento, el AutoCom se consideraba un sofisticado dispositivo AAC portátil con 128 áreas de símbolos que podían programarse y personalizarse para el usuario. Tenía un tamaño de 20 por 24 pulgadas y pesaba 17 libras. AutoCom proporcionó al usuario una salida impresa en una pequeña impresora de tiras de papel y una pantalla LED de 32 caracteres. Aunque AutoCom proporcionó opciones para usar símbolos de imágenes y múltiples niveles de programación, su uso todavía estaba limitado a las personas que podían acceder al dispositivo tocando directamente los símbolos con un puntero especial. El habla sintetizada no era una opción en la mayoría de estos primeros sistemas.

En 1978, se desarrollaron el HandiVoice 110 y 120 por la compañía Phonic Ear. Estos dos dispositivos fueron los primeros SAAC portátiles disponibles comercialmente con salida de voz sintetizada. Los dispositivos HandiVoice sensibilizaron a los profesionales y usuarios sobre el potencial comunicativo de los SAAC portátiles con salida de voz. Con los continuos avances en la tecnología de microcomputadoras, los dispositivos AAC se han vuelto más sofisticados y más adaptables para una amplia variedad de usuarios.

1.6 Descripción del capitulado

Capítulo 2.- En este apartado se describe el marco teórico donde se menciona información más a fondo de los SAAC, las compañías de telefonía de México, sus tarifas e infraestructura, los módulos GSM, los comandos AT, las redes sociales

WhatsApp y Telegram en su versión de navegador web y los lenguajes de programación Python, HTML5 y JavaScript.

Capítulo 3.- En este apartado, se narra el funcionamiento del proyecto realizado, utilizando un mapa de navegación para facilitar el entendimiento del sistema, así como la visualización de los pictogramas empleados en cada sección del SAAC. De igual forma, se usan diagramas de flujo para explicar el trabajo que realizan las funciones programadas en el software.

Capítulo 4.- En este apartado, se muestran las pruebas realizadas durante el desarrollo del proyecto junto con los resultados obtenidos de estas, los cuales ayudaron a mejorar la versión final del proyecto. Se incluyen funciones programadas de manera externa al SAAC, así como circuitos electrónicos que comunican al computador con el módulo GSM.

Conclusiones y trabajo futuro.- En esta sección se dan las conclusiones del desarrollo de la propuesta presentada, así como la mención de las características que se tienen consideradas incluir en las siguientes versiones del SAAC de Grupo IDEA.

Se concluye el trabajo con la presentación de referencias y un anexo con las rutinas de programación adicionadas al SAAC original.

Capítulo 2. Marco Teórico

2.1 Los SAAC

De acuerdo con la American Speech-Language-Hearing Association (ASHA), la comunicación aumentativa y alternativa es un sistema de toma de decisiones que considera métodos de comunicación individuales y determina su efectividad en personas con diversos trastornos del habla en un principio temporal o permanente. El programa AAC consta de los cuatro elementos fundamentales que incluyen símbolos, estrategias, técnicas y ayuda. Están disponibles varios tipos de los símbolos mencionados anteriormente, como gráficos, auditivos, gesticulares o táctiles. Los símbolos pueden ser creados por el remitente sin ninguna ayuda, por ejemplo: gestos rituales y expresión facial o aplicando ayudas, como objetos e imágenes. Las estrategias incluyen procedimientos que deberían aumentar la velocidad de transmisión o el tiempo necesario para buscar mensajes apropiados. Con referencia a las técnicas, se utilizan universalmente diversos métodos de transmisión de mensajes. Dos métodos básicos, la selección indirecta o de selección y la directa requieren una variedad de medios para permitir que las personas se comuniquen.

El escaneo requiere que una persona participe auditiva, visual o táctilmente, mientras retiene el pensamiento o mensaje que le gustaría comunicar. El término "ayuda" se refiere a un dispositivo electrónico y no electrónico que se utiliza para transmitir o recibir mensajes. La frase ayuda en este sentido puede diferir de los dispositivos simples (por ejemplo, un teléfono o una foto adjunta a una hoja de papel o un solo mensaje almacenado en un dispositivo).

2.1.1 Condiciones de salud de pacientes que requieren un SAAC

El objetivo de los Sistemas Alternativos y Aumentativos de Comunicación es, precisamente, la enseñanza de un conjunto de estructuras de códigos no vocales, que permitirán establecer o ampliar las posibilidades de comunicación de personas con déficit o incapacidad para hablar, fomentando así su autoestima y autonomía personal y social.

Quienes van a utilizar los SAAC son aquellas personas que presentan una dificultad importante en su capacidad de comunicación de forma puntual o permanente. Se podrían clasificar en cuatro grupos:

- Personas con discapacidad motora a consecuencia de parálisis cerebral, traumatismos craneoencefálicos o patologías neuromusculares progresivas, como las distrofias musculares.
- Personas con discapacidad intelectual, cognitiva o psíquica, como trastornos de desarrollo y trastorno espectro autista (TEA).
- Personas con discapacidad sensorial, como sordera, ceguera o sordo-ceguera.

- Personas sometidas a operaciones o con lesiones que impliquen limitaciones en los órganos encargados del habla. [4]

2.1.2 Clasificación de los SAAC

El hecho de que los Sistemas o Estrategias se llamen Alternativos o Aumentativos, no designa otra cosa que la finalidad con la cual se van a utilizar, pero los sistemas pueden ser los mismos. Los sistemas de comunicación pueden clasificarse según precisen o no de algún soporte técnico para su expresión. En el primer caso se hablará de sistemas sin ayuda y, en el segundo caso, de sistemas con ayuda.

Los Sistemas de Comunicación sin ayuda posibilitan las diferentes formas de intercambiar la información usando el cuerpo, en vez de algún tipo de ayuda o herramienta. Por ejemplo, las personas que son sordas y usan gestos o lengua de signos para comunicarse, lo hacen con sus manos, las expresiones de su cara y a menudo con la manera de estar de pie o de situar su cuerpo. El aprendizaje procedimental de estos sistemas es principalmente: la Dactilología, la Lengua de Signos, el Bimodal y la Palabra Complementada.

Los Sistemas de Comunicación con ayuda posibilitan el uso de asistencias o herramientas para la comunicación. Estas ayudas permiten preguntar, hablar sobre sentimientos, y contar o que se cuenten las cosas que han pasado durante el día. Las herramientas que se usan más frecuentemente son: papel o cartulina, una carpeta o un libro. Éstos tienen dibujos, letras o palabras escritas en ellos con los cuales se pueden elaborar los tableros o plantillas de comunicación. También hay dispositivos electrónicos que pueden decir o imprimir los mensajes que una persona selecciona o crea. Algunos son muy simples y otros muy sofisticados llamados de alta tecnología.

Dentro de los Sistemas de Comunicación con ayuda destacan varios sistemas tales como:

- Sistema BLISS, que son símbolos gráfico-visuales que representan significados. Estos símbolos se combinan de diversas maneras formando así nuevos significados, con lo que se crea un sistema complejo capaz de expresar conceptos diferentes.
- Sistema SPC, que es un sistema de comunicación no oral basado en símbolos pictográficos con gran sencillez y transparencia de los pictogramas que se usan.
- Sistema Minspeak, que es un sistema de comunicación aumentativa diseñado para una serie de comunicadores con voz. Se basa en la misma idea que los jeroglíficos: un mismo dibujo puede tener uno u otro significado dependiendo qué rasgo se destaque de él o con cuál otro se combine. [5]

2.2 Telefonía móvil

La telefonía móvil es conocida también como Servicio Móvil o Telefonía Celular y es un servicio de conexión a la red telefónica pública mediante una red

inalámbrica, en la cual los usuarios tienen la posibilidad de originar y recibir llamadas telefónicas. Además, con el servicio móvil también pueden enviar o recibir mensajes de texto (SMS) y tener acceso a Internet (transferencia de datos).

La telefonía móvil tiene 2 modalidades para su servicio: Prepago, que funciona por medio de recargas periódicas y Pospago, en la que se fijan montos y productos incluidos en el servicio. La telefonía móvil convierte todo el tráfico que se utiliza diariamente para la comunicación (voz, datos, texto, mensajes multimedia etc.) en señales de radiofrecuencia (RF), las cuales viajan a través del aire (espectro radioeléctrico) hasta llegar a su destino.

La telefonía celular es un medio de comunicación personal con características de inmediatez, efectividad, interactividad, confidencialidad y seguridad que han revolucionado la vida social, la tecnología y la economía a nivel mundial. Los teléfonos móviles (celulares) tienen la capacidad de ser portátiles debido a que cuentan con una antena y una fuente de poder de baja potencia que permiten el envío de señales dentro de un rango reducido.

2.2.1 Sistema Global de Comunicaciones Móviles

Las siglas GSM se corresponden al nombre en inglés del Sistema Global de Comunicaciones Móviles. Se trata de un estándar muy utilizado desde principios de siglo y también se conoce como 2G debido a que supuso un salto de las comunicaciones analógicas a las digitales.

La banda de frecuencia en la que opera el GSM difiere según el territorio. En Europa se utiliza el espectro radioeléctrico de 900 y 1800 MHZ, mientras que en Estados Unidos la banda es la de 1900. Esto hace que no todos los móviles GSM puedan funcionar en todo el mundo, a no ser que su tecnología esté preparada para conectarse a todas las bandas.

Los teléfonos que se utilizan se denominan estaciones móviles. Para que esta estación sea operativa se necesita una tarjeta SIM, que contiene información sobre el terminal y su usuario. Información referente al operador de red, tipo de contrato y otros detalles también están grabados en la tarjeta. Cada estación móvil tiene un identificador único, el IMEI. Las tarjetas también tienen su propio identificador internacional, con lo que se puede transferir a otro equipo sin perder la información.

La SIM indica a la estación base o torre de repetición quién es el usuario que se comunica a través de ella. Esta conexión se realiza a través de ondas de radio. Las estaciones base están unidas en red a través de un controlador. Este se encarga de gestionar todos los recursos para que la comunicación sea lo mejor posible. A su vez todos los controladores se conectan a un centro conmutador mediante cable. Los conmutadores son controlados por el operador de telefonía, donde se recopilan todos los datos y se verifican las identidades de cada SIM.

2.2.2 SIM

La tarjeta SIM es un chip en el que se guarda información imprescindible para que el servicio de telefonía funcione. También incluye los datos de acceso para desbloquear un terminal, sin los cuales no sirve el teléfono.

Concretamente en la SIM se graba el número de teléfono que recibe el abonado, un código internacional que no se pueda duplicar, qué operador presta el servicio (que tiene un código propio) y una clave de seguridad. Además de contraseñas de acceso, el tradicional PIN de 4 dígitos para acceder a la red y el PUK con el que se puede desbloquear un terminal si es necesario. [6]

La tecnología (2G) presta los servicios de llamadas de voz; intercambio de mensajes de texto de hasta 140 caracteres; funciones de llamada como llamada en espera y llamadas en conferencia; baja transferencia de datos móviles para aplicaciones de mensajería como WhatsApp y correo electrónico, entre otras.

La tecnología (3G) presta todos los servicios incluidos en las redes 2G con mejora en la velocidad de transferencia de datos móviles, además de la introducción de nuevos servicios de localización, multimedia e internet, entre sus principales aplicaciones se encuentran: Sistemas de posicionamiento Global (GPS), navegación fluida en Internet, video conferencias, transacciones financieras, video a demanda, etc.

2.2.2 Telefonía móvil en México

Actualmente en México existen 314 MHz asignados para la provisión de servicios de banda ancha móvil en México, esta cantidad espectral está distribuida entre los operadores: Telcel, AT&T, Movistar y Servicios de Acceso Inalámbricos (SAI).

Las bandas de frecuencias del espectro radioeléctrico utilizadas para el servicio móvil en México son la banda de 800 MHz (814-849 MHz / 859-894 MHz); banda PCS (1850-1910 MHz / 1930-1990 MHz) y banda AWS (1710-1780 MHz / 2110-2180 MHz).

Las generaciones de tecnología de telefonía móvil que se utilizan actualmente en México son GSM (2G), GPRS (2.5G), EDGE (2.75G), UMTS (3G), HSDPA (3.5G), HSUPA (3.75G), HSPA+ (3.9G) y LTE (4G).

2.2.3 Proveedores de servicio

En el mes de enero de 2015 entró en vigor una nueva reforma en el sector de telecomunicaciones que vendría a cambiar radicalmente la oferta en la industria de la telefonía móvil.

En términos generales la Reforma de Telecomunicaciones es una ley que ayudó a regular la competencia entre grandes y pequeñas compañías en el mercado ofreciendo como consecuencia mejores ofertas para el consumidor final.

De esta forma, podemos decir que el sector de las telecomunicaciones en México está dividido entre los operadores que tienen infraestructura propia y permisos con

el fin de la explotación del espectro de radiofrecuencia y operadores más pequeños que arriendan el equipamiento y redes para ofrecer sus servicios.

2.3.4 Operadores con red propia en México:

- Telcel
- Movistar
- AT&T
- Telecomunicaciones Indígenas Comunitarias

2.3.5 Operadores Móviles con red propia

Telcel: es una compañía de telecomunicaciones de origen 100% mexicano fundada en el año de 1989. Cuenta con la mayor cobertura en territorio nacional, ofrece servicios de telefonía celular e Internet de alta velocidad con las tecnologías 3G y 4G LTE.

Movistar: la compañía de origen español fue la primera en hacerle frente a Telcel en el año 2000, tras la compra de los operadores Cedetel, Bajacel, Norcel, Movitel y Pegaso PCS. Actualmente cuenta con una cobertura considerable en gran parte del territorio mexicano y brinda servicios de telefonía celular, Internet en casa e Internet de alta velocidad con la tecnología 3G y 4G LTE de Movistar.

AT&T: la compañía de origen estadounidense inició sus actividades en México en el año 2015, tras haber adquirido las compañías de telefonía móvil Iusacell y Nextel. Su enfoque principal son los servicios de telefonía celular de alta velocidad con la tecnología 4G LTE de AT&T, pero también ofrece telefonía celular e Internet en casa.

Telecomunicaciones Indígenas Comunitarias: es la primera red propia de uso social indígena y opera desde el año 2016. Posee dos títulos de concesión:

Para uso de bandas del espectro radioeléctrico con el fin de uso social indígena, sin fines de lucro.

Con el fin de prestar cualquier servicio de telecomunicaciones y radiodifusión de uso indígena (solo en algunas partes del país)".

2.3.6 Tarifas

Al momento de la escritura de este trabajo se consultaron las tarifas del operador Telcel para el territorio de México, mismas que se presentan en la tabla 2.1.

Tabla 2.1 Tarifas de Telcel para México.

Plan	Incluye	Precio
Amigo sin límite 20	☑ 100MB Datos ☑ Vigencia 1 día	**$20**
Amigo sin límite 30	☑ 130MB Datos ☑ Vigencia 3 días	**$30**
Amigo sin límite 50	☑ 400MB Datos ☑ Vigencia 7 días	**$50**
Amigo sin límite 80	☑ 500MB Datos ☑ Vigencia 13 días	**$80**
Amigo sin límite 100	☑ 1.3GB Datos ☑ Vigencia 15 días	**$100**

Todos los planes Telcel incluyen:

- Mensajes SMS y minutos ilimitados para realizar llamadas a números fijos y móviles en todo México, Estados Unidos y Canadá.
- Redes sociales incluidas (Facebook, Messenger, Twitter y WhatsApp - sin incluir acceso a ligas externas, llamadas de voz o video llamadas).
- Datos de alta velocidad a nivel nacional con la red 4G.
- Las redes incluidas en prepago aplican en recargas de $50 pesos o más.

Como se observa en la Tabla 2.2, todos los planes de AT&T incluyen:

- Mensajes SMS y minutos ilimitados para realizar llamadas a números fijos y móviles en todo México, Estados Unidos y Canadá.
- Redes sociales incluidas (Facebook, Messenger, Twitter y WhatsApp - sin incluir acceso a ligas externas, llamadas de voz o video llamadas).
- Datos de alta velocidad a nivel nacional con la red 4G
- Las redes incluidas en prepago aplican en recargas de $50 pesos o más.

Tabla 2.2 Tarifas de ATT para México.

Más $30	☑ 300MB Datos ☑ Vigencia 3 días	**$30**
Más $50	☑ 500MB Datos ☑ Vigencia 5 días	**$50**
Más $100	☑ 1.5GB Datos ☑ Vigencia 14 días	**$100**
Más $150	☑ 2.3GB Datos ☑ Vigencia 25 días	**$150**
Más $200	☑ 3GB Datos ☑ Vigencia 30 días	**$200**

Todos los planes pospago de Movistar incluyen

- Mensajes SMS y minutos ilimitados para realizar llamadas a números fijos y móviles en todo México, Estados Unidos, Canadá y Puerto Rico.
- Redes sociales incluidas (Facebook, Messenger, Twitter y WhatsApp - sin incluir acceso a ligas externas, llamadas de voz o video llamadas).
- GB de almacenamiento en Movistar Cloud.
- Datos de alta velocidad a nivel nacional con la red 4G.

Esto se puede visualizar de mejor manera en la Tabla 2.3.

Tabla 2.3 Tarifas de Movistar para México. [7]

Plan	Incluye	Precio
Recarga $30	☑ 300MB Datos ☑ Vigencia 3 días	**$30**
Paquete Portabilidad $50	☑ 2.4GB Datos ☑ Vigencia 23 días	**$50**
Recarga $50	☑ 500MB Datos ☑ Vigencia 7 días	**$50**
Paquete Portabilidad $100	☑ 6GB Datos ☑ Vigencia 30 días	**$100**
Recarga $100	☑ 1.4GB Datos ☑ Vigencia 15 días	**$100**

2.4 Raspberry Pi 4 Modelo B

La Raspberry Pi es una computadora de bajo costo y con un tamaño compacto, del porte de una tarjeta de crédito, puede ser conectada a un monitor de computador o un TV, y usarse con un mouse y teclado estándar. Es un pequeño computador que correo un sistema operativo Linux capaz de permitirle a las personas de todas las edades explorar la computación y aprender a programar lenguajes como Scratch y Python. Es capaz de hacer la mayoría de las tareas típicas de un computador de escritorio, desde navegar en internet, reproducir videos en alta resolución, manipular documentos de ofimática, hasta reproducir juegos.

Además, la Raspberry Pi tiene la habilidad de interactuar con el mundo exterior, puede ser usada en una amplia variedad de proyectos digitales, desde reproductores de música y video, detectores de padres, estaciones meteorológicas hasta cajas de aves con cámaras infrarrojas. [8]

La Raspberry Pi 4 Modelo B es uno de los últimos productos de la gama de computadoras Raspberry Pi. Ofrece incrementos revolucionarios en la velocidad

del procesador, el rendimiento multimedia, la memoria y la conectividad en comparación con la Raspberry Pi 3 Modelo B + de la generación anterior, al tiempo que conserva la compatibilidad con versiones anteriores y un consumo de energía similar. Para el usuario final, Raspberry Pi 4 Modelo B proporciona un rendimiento de escritorio comparable al de los sistemas de PC x86 de nivel de entrada. La Raspberry Pi 4 Modelo B permanecerá en producción hasta al menos enero de 2026.

2.4.1 Especificaciones técnicas generales

- Procesador SoC Broadcom BCM2711, Cortex-A72 de cuatro núcleos (ARM v8) de 64 bits a 1,5 GHz.
- Memoria SDRAM LPDDR4-3200 de 1 GB, 2 GB, 4 GB u 8 GB (según el modelo).
- Conectividad 2,4 GHz y 5,0 GHz IEEE 802.11ac inalámbrico, Bluetooth 5.0, BLE.
- Gigabit Ethernet.
- 2 puertos USB 3.0; 2 puertos USB 2.0.
- Cabecera GPIO de 40 pines estándar de Raspberry Pi (totalmente compatible con versiones anteriores de placas anteriores).
- 2 × puertos micro-HDMI (compatible con hasta 4kp60).
- Puerto de pantalla MIPI DSI de 2 carriles.
- Puerto de audio estéreo y video compuesto de 4 polos.
- Ranura para tarjeta micro-SD para cargar sistema operativo y almacenamiento de datos.
- 5V CC mediante conector USB-C (mínimo 3A*).
- 5V CC a través del cabezal GPIO (mínimo 3A*).
- Alimentación a través de Ethernet (PoE) habilitada (requiere PoE HAT separado).
- Temperatura de funcionamiento: 0 – 50 grados C ambiente.

*Se puede utilizar una fuente de alimentación de 2.5 A de buena calidad si los periféricos USB posteriores consumen menos de 500 mA en total. [9]

2.4.2 Condiciones de uso

- La Raspberry Pi 4 solo debe conectarse a una fuente de alimentación externa de 5 V / 3 A CC o 5.1 V / 3 A CC como mínimo. Cualquier fuente de alimentación externa utilizada con la Raspberry Pi 4 Modelo B deberá cumplir con las regulaciones y estándares relevantes aplicables en el país de uso previsto.

- Debe utilizarse en un ambiente bien ventilado y, si se usa dentro de una caja, la caja no debe cubrirse.
- Debe colocarse sobre una superficie estable, plana y no conductora en uso y no debe ser contactado por elementos conductores.
- La conexión de dispositivos incompatibles a la conexión GPIO puede afectar el cumplimiento y resultar en daños a la unidad.
- Todos los periféricos utilizados con la microcomputadora deben cumplir con las normas pertinentes del país de uso y estar marcados en consecuencia para garantizar que se cumplan los requisitos de seguridad y rendimiento. Estos artículos incluyen, entre otros, teclados, monitores y ratones cuando se utilizan junto con la Raspberry Pi.
- Cuando se conectan periféricos que no incluyen el cable o conector, el cable o conector debe ofrecer un aislamiento y un funcionamiento adecuados para que se cumplan los requisitos de seguridad y rendimiento relevantes.

2.4.3 Instrucciones de seguridad

Para evitar un mal funcionamiento o daños, se debe tener en cuenta lo siguiente:

- No se debe exponer al agua, la humedad ni colocarse sobre una superficie conductora mientras esté en funcionamiento.
- No se debe exponer al calor de ninguna fuente; Raspberry Pi 4 Modelo B está diseñado para un funcionamiento confiable a temperatura ambiente normal.
- Tener cuidado durante la manipulación para evitar daños mecánicos o eléctricos en la placa de circuito impreso y los conectores.
- Evitar manipular la placa de circuito impreso mientras esté encendida y solo manejarla por los bordes para minimizar el riesgo de daños por descarga electrostática. [10]

2.5 Módulos GSM/LTE

Un módulo GSM/GPRS es un chip o circuito que se utiliza para establecer comunicación entre un dispositivo móvil o una computadora y un sistema GSM o GPRS. El módem (modulador-demodulador) es una parte crítica aquí.

Pueden presentar todas las funcionalidades de un teléfono móvil a través de una computadora, como hacer y recibir llamadas, SMS, MMS, etc. Estos se emplean principalmente para servicios de SMS y MMS basados en computadora.

Estos módulos consisten en un módulo GSM o módem GPRS alimentado por un circuito de alimentación e interfaces de comunicación (como RS-232, USB 2.0 y otros) para computadora. Un módem GSM puede ser un dispositivo de módem dedicado con una conexión serial, USB o Bluetooth, o puede ser un teléfono móvil que proporcione capacidades de módem GSM.

Un módulo GSM o módulo GPRS es similar al módem, pero hay una diferencia: un módem GSM/GPRS es un equipo externo, mientras que el módulo GSM/GPRS es un módulo que se puede integrar dentro de un equipo. Es una pieza de hardware incrustada. Un móvil GSM, por otro lado, es un sistema completo en sí mismo con procesadores integrados que se dedican a proporcionar una interfaz entre el usuario y la red móvil.

Los módems inalámbricos generan, transmiten o decodifican datos de una red celular para establecer comunicación.

Un módem GSM/GPRS es una clase de módem inalámbrico, diseñado para la comunicación a través de la red GSM y GPRS. Requiere una tarjeta SIM (*Subscriber Identity Module*) al igual que los teléfonos móviles para activar la comunicación con la red. Además, cuentan con un número IMEI (International Mobile Equipment Identity) similar al de los teléfonos móviles para su identificación.

2.5.1 Comandos AT

El modem necesita comandos AT, para interactuar con el procesador o controlador, los cuales se comunican a través de comunicación serial. Estos comandos son enviados por el controlador/procesador. El modem devuelve un resultado después de recibir un comando. El procesador/controlador/computadora puede enviar diferentes comandos AT admitidos por el MODEM para interactuar con la red celular GSM y GPRS.

Sus funciones incluyen:

- Leer, escribir y borrar mensajes SMS.
- Enviar mensajes SMS.
- Controlar la intensidad de la señal.
- Controlar el estado de carga y el nivel de carga de la batería.
- Leer, escribir y buscar entradas en la guía telefónica.

Un teléfono móvil y un módulo de identidad del suscriptor (SIM) juntos forman una estación móvil. Es el equipo del usuario el que se comunica con la red móvil. Un teléfono móvil se compone de terminación móvil, equipo terminal y adaptador de terminal.

Mobile Termination está interconectado con la red móvil GSM y está controlado por un procesador de banda base. Maneja el acceso a SIM, codificación y decodificación de voz, señalización y otras tareas relacionadas con la red. El Equipo terminal es un procesador de aplicaciones que se ocupa de las operaciones de manejo relacionadas con el teclado, la pantalla, la memoria del teléfono y otros servicios de hardware y software integrados en el teléfono. El adaptador de terminal establece la comunicación entre el equipo terminal y la terminación móvil mediante comandos AT. La comunicación con la red en un móvil GSM/GPRS se realiza mediante el procesador de banda base.

2.5.2 Aplicaciones del módulo GSM o módulo GPRS

Se conocen como comandos AT porque cada línea de comando comienza con "AT" o "at". Los comandos AT son instrucciones que se utilizan para controlar un módem. AT es la abreviatura de ATtention.

Los módems y teléfonos móviles GSM/GPRS admiten un conjunto de comandos AT que es específico de la tecnología GSM, que incluye comandos relacionados con SMS como AT+CMGS (Enviar mensaje SMS), AT+CMSS (Enviar mensaje SMS desde almacenamiento), AT+CMGL (Lista de mensajes SMS) y AT+CMGR (Leer mensajes SMS).

Tenga en cuenta que el "AT" inicial es el prefijo que informa al módem sobre el inicio de una línea de comando. No es parte del nombre del comando AT. Por ejemplo, D es el nombre del comando AT real en ATD y +CMGS es el nombre del comando AT real en AT+CMGS. Sin embargo, algunos libros y sitios web los usan indistintamente como el nombre de un comando AT.

2.5.3 Módulo SIM800L

El módulo SIM800L es un dispositivo quad-band GSM/GPRS, trabaja en las frecuencias GSM850MHz, EGSM900MHz, DCS1800MHz y PCS1900MHz. Este módulo de telefonía celular permite añadir voz, texto, datos y SMS.

Por sí solo, este módulo no puede hacer nada. Se requiere un microcontrolador para controlarlo por ejemplo un Arduino, pero cualquier microcontrolador 3 - 5V con una UART puede enviar y recibir comandos a través de los pines RX/TX. También necesita un chip SIM 2G.

2.5.4 Características técnicas

- Voltaje de Operación: 3.4V ~ 4.4V DC.
- Nivel Lógico de 3V a 5V.
- Consumo de corriente (max): 500 mA.
- Consumo de corriente (sleep): 0.7 mA.
- Interfaz: Serial UART.
- Quad-band 850/900/1800/1900MHz – se conectan a cualquier red mundial GSM con cualquier SIM 2G.
- Trabaja solo con tecnología 2G (en Perú Movistar, Claro y Entel).
- Hacer y recibir llamadas de voz usando un auricular o un altavoz de 8Ω externo + micrófono electret.
- Enviar y recibir mensajes SMS.
- Enviar y recibir datos GPRS (TCP/IP, HTTP, etc.).
- Receptor FM.
- Controlado por Comandos AT (3GPP TS 27.007, 27.005 y SIMCOM enhanced AT Commands).
- Interfaz de comandos AT con detección “automática” de velocidad de transmisión.
- Soporta A-GPS.

- Velocidad máxima de transmisión 85.6 Kbps.
- Protocolo TCP/IP en chip.
- Codificación: CS-1, CS-2, CS-3 y CS-4.
- Soporta USSD.
- Soporta Reloj en tiempo real (RTC).
- Velocidades de transmisión serial desde 1200bps hasta 115200bps.
- Tamaño de la SIM: Micro SIM. [11]

2.5.5 Módulo GA6-B

El módulo GA6 es una versión mini de la placa de desarrollo de núcleo serial GSM / GPRS basada en el chip GPRS A6. Este chip es compatible con la red GSM/GPRS, disponible para transmisión remota de datos de mensajes GPRS y SMS. Con la ayuda del chip A6, GPRS nunca se desactiva hasta que su aplicación está activa y en línea.

Tiene más relación precio-rendimiento que los módulos SIM800, SIM900. Este módulo pequeño y de bajo consumo de energía puede comunicarse con los microcontroladores y placas como Arduino a través de la interfaz UART, con la capacidad de recepción de comandos, incluidos los estándares GSM 07.07, GSM 07.05.

Se puede utilizar para proyectos IoT, aplicaciones M2M, automatización industrial, proyectos BMS, automatización del hogar, transporte público, seguimiento personal, detección de entorno eléctrico, POS inalámbrico, medición inteligente y otras aplicaciones M2M. [12]

Características:

- Frecuencia de trabajo: red cuatribanda, 850/900/1800/1900MHZ.
- Voltaje de trabajo: 3.5~4.2 VDC (Adaptado a 5V).
- Corriente de trabajo: máximo de 2A.
- Corriente de sueño: 5mA.
- Porta tarjetas Micro SIM incorporado, puede instalar tarjetas Micro SIM
- Interfaz de comunicación: puerto serie TTL.
- Tasa de baudios: 115200 bps y también se puede configurar mediante un comando AT.
- Voltaje lógico de interfaz: 3,3 V.
- Realice y responda llamadas telefónicas con un auricular y un micrófono electret.
- Enviar y recibir mensajes SMS.
- Enviar y recibir datos GPRS (TCP/IP, HTTP, etc.). [12]

2.6 Python

Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90 cuyo nombre está inspirado en el grupo de cómicos

ingleses “Monty Python”. Es un lenguaje similar a Perl, pero con una sintaxis muy limpia y que favorece un código legible.

Se trata de un lenguaje interpretado o de script, con tipado dinámico, fuertemente tipado, multiplataforma y orientado a objetos.

2.5.1 Lenguaje interpretado o de script

Un lenguaje interpretado o de script es aquel que se ejecuta utilizando un programa intermedio llamado intérprete, en lugar de compilar el código a lenguaje máquina que pueda comprender y ejecutar directamente una computadora (lenguajes compilados).

La ventaja de los lenguajes compilados es que su ejecución es más rápida. Sin embargo, los lenguajes interpretados son más flexibles y más portables.

Python tiene, no obstante, muchas de las características de los lenguajes compilados, por lo que se podría decir que es semi-interpretado. En Python, como en Java y muchos otros lenguajes, el código fuente se traduce a un pseudocódigo máquina intermedio llamado bytecode la primera vez que se ejecuta, generando archivos .pyc o .pyo (bytecode optimizado), que son los que se ejecutarán en sucesivas ocasiones.

2.5.2 Tipado dinámico

La característica de tipado dinámico se refiere a que no es necesario declarar el tipo de dato que va a contener una determinada variable, sino que su tipo se determinará en tiempo de ejecución según el tipo del valor al que se asigne, y el tipo de esta variable puede cambiar si se le asigna un valor de otro tipo.

No se permite tratar a una variable como si fuera de un tipo distinto al que tiene, es necesario convertir de forma explícita dicha variable al nuevo tipo previamente. Por ejemplo, si tenemos una variable que contiene un texto (variable de tipo cadena o string) no podremos tratarla como un número (sumar la cadena “9” y el número 8). En otros lenguajes el tipo de la variable cambiaría para adaptarse al comportamiento esperado, aunque esto es más propenso a errores.

2.5.3 Multiplataforma

El intérprete de Python está disponible en multitud de plataformas (UNIX, Solaris, Linux, DOS, Windows, OS/2, Mac OS, etc.) por lo que si no utilizamos librerías específicas de cada plataforma nuestro programa podrá correr en todos estos sistemas sin grandes cambios.

2.5.4 Orientado a objetos

La orientación a objetos es un paradigma de programación en el que los conceptos del mundo real relevantes para nuestro problema se trasladan a clases y objetos en nuestro programa. La ejecución del programa consiste en una serie de interacciones entre los objetos. Python también permite la programación imperativa, programación funcional y programación orientada a aspectos. [13]

2.7 HTML

HTML surge como una propuesta para crear la estructura básica de páginas web, organizar su contenido y compartir información. El lenguaje y la web misma nacieron principalmente con la intención de comunicar información por medio de texto.

El limitado objetivo de HTML motivó a varias compañías a desarrollar nuevos lenguajes y programas para agregar características a la web nunca antes implementadas. Estos desarrollos iniciales crecieron hasta convertirse en populares y poderosos accesorios.

De las opciones propuestas, Java y Flash fueron las más exitosas; ambas fueron masivamente adoptadas y ampliamente consideradas como el futuro de Internet. Sin embargo, tan pronto como el número de usuarios se incrementó e Internet pasó de ser una forma de conectar amantes de los ordenadores a un campo estratégico para los negocios y la interacción social, limitaciones presentes en estas dos tecnologías probaron ser una sentencia de muerte.

El mayor inconveniente de Java y Flash puede describirse como una falta de integración. Ambos fueron concebidos desde el principio como complementos (plugins), algo que se inserta dentro de una estructura pero que comparte con la misma solo espacio en la pantalla. No existía comunicación e integración alguna entre aplicaciones y documentos.

La falta de integración resultó ser crítica y preparó el camino para la evolución de un lenguaje que comparte espacio en el documento con HTML y no está afectado por las limitaciones de los plugins. JavaScript, un lenguaje interpretado incluido en navegadores, claramente era la manera de mejorar la experiencia de los usuarios y proveer funcionalidad para la web. Sin embargo, después de algunos años de intentos fallidos para promoverlo y algunos malos usos, el mercado nunca lo adoptó plenamente y pronto su popularidad declinó.

En ese momento, JavaScript no era capaz de reemplazar la funcionalidad de Flash o Java. A pesar de ser evidente que ambos limitaban el alcance de las aplicaciones y aislaban el contenido web, populares funciones como la reproducción de video se estaban convirtiendo en una parte esencial de la web y solo eran efectivamente ofrecidas a través de estas tecnologías.

A pesar del suceso inicial, el uso de Java comenzó a declinar. La naturaleza compleja del lenguaje, su evolución lenta y la falta de integración disminuyeron su importancia hasta el punto en el que hoy día no es más usado en aplicaciones web de importancia. Sin Java, el mercado volcó su atención a Flash. Pero el hecho de que Flash comparte las mismas características básicas que su competidor en la web lo hace también susceptible de correr el mismo destino.

JavaScript era claramente el lenguaje que permitía a los desarrolladores innovar y hacer cosas que nadie había podido hacer antes en la web. En los últimos años, programadores y diseñadores web alrededor del mundo surgieron con los más increíbles trucos para superar las limitaciones de esta tecnología y sus iniciales deficiencias en portabilidad. Gracias a estas nuevas implementaciones, JavaScript,

HTML y CSS se convirtieron pronto en la más perfecta combinación para la necesaria evolución de la web. HTML5 es, de hecho, una mejora de esta combinación, el pegamento que une todo.

HTML5 propone estándares para cada aspecto de la web y también un propósito claro para cada una de las tecnologías involucradas. A partir de ahora, HTML provee los elementos estructurales, CSS se encuentra concentrado en cómo volver esa estructura utilizable y atractiva a la vista, y JavaScript tiene todo el poder necesario para proveer dinamismo y construir aplicaciones web completamente funcionales.

Google Chrome ya implementa muchas de las características de HTML5 y además es una buena plataforma para pruebas. Por otro lado, Firefox es uno de los mejores navegadores para desarrolladores y también provee total soporte para HTML5. [14]

2.8 WhatsApp Web

WhatsApp Web es una manera de utilizar WhatsApp a través de un navegador, pudiendo escribir mensajes, leerlos o enviar archivos. Prácticamente se puede hacer lo mismo que en la versión móvil, pero desde el PC. Esto puede ser muy útil, por ejemplo, si se está utilizando WhatsApp mientras se está frente a un ordenador, evitando la necesidad de emplear un móvil.

Al hacer una comparativa de WhatsApp Web frente a WhatsApp, se observará que se tienen disponibles prácticamente todas las opciones, desde enviar notas de voz hasta enviar fotografías. Faltan algunas cosas como el poder realizar llamadas de voz o de vídeo o compartir la ubicación, pero por lo demás, es prácticamente igual que utilizar la versión móvil. Gracias a que WhatsApp está trabajando en su función dispositivo, se puede utilizar WhatsApp Web, aunque no se tenga el teléfono encendido al lado

Utilizar WhatsApp Web es relativamente sencillo, y solo se tiene que realizar un sencillo proceso para vincular la web con la cuenta a través del móvil. Para eso, se entra en web.whatsapp.com, que es la página principal del servicio. Allí aparecerá un código QR de acceso que se va a tener que escanear con la versión móvil de WhatsApp.

A partir del momento en que se inicia sesión en WhatsApp Web, se verán exactamente las mismas conversaciones que se tengan en el móvil, y se podrá interactuar con los contactos sin ningún problema. Todo lo que se escriba en la web se reflejará en el móvil, pues ambos servicios estarán sincronizados. Cabe señalar que el navegador puede almacenar el cache de la página, por lo que, una vez escaneado el código, en próximas ocasiones que se utilice WhatsApp Web, no será necesario iniciar sesión, pues ya estará guardada. [16]

2.9 Telegram Web

Telegram Web se trata de la aplicación de Telegram trasladada al navegador web, conservando las características en escritura, lectura y envío de mensajes.

Telegram Web es una de las opciones más cómodas para usar la plataforma cuando se está trabajando en el ordenador y tiene muchas ventajas: no necesita que se instale nada, se pueden utilizar las funciones básicas y se tiene la posibilidad de comunicarse sin necesidad de emplear el móvil.

Para acceder, se debe entrar en la página web, introducir el país y escribir el número de teléfono. Al hacerlo, se recibirá en el móvil un mensaje de Telegram facilitando el código de inicio de sesión. Una vez introducido, se podrá comenzar a utilizar la aplicación. Para próximas sesiones no será necesario utilizar un código de verificación, puesto que el navegador web almacenará la información de la sesión en la memoria cache. [17]

Capítulo 3. Desarrollo experimental

En este apartado se describe como ha sido desarrollada la propuesta de implementación de un módulo para dotar de comunicación GSM sobre un SAAC existente que fue mencionado en el capítulo introductorio, así como también la incorporación de la capacidad de comunicaciones vía WhatsApp y Telegram. La estructura general consiste presentar un diagrama de bloques, la implementación del módulo GSM, los mapas de navegación del sistema, en los que se muestran las trayectorias de las opciones para usuarios, así como la presentación de las plantillas añadidas y sus funciones; también se presentan los diagramas de flujo más importantes que representan a las rutinas de programación incorporadas.

3.1 Diagrama de bloques

Antes de comenzar con la descripción de la implementación de las funcionalidades mencionadas, en la Ilustración 3.1 se muestra un diagrama de bloques general que representa al sistema aumentativo y alternativo de comunicación en su totalidad, en el que ya se incluye un bloque para la comunicación de mensajes del tipo SMS (módulo GSM).

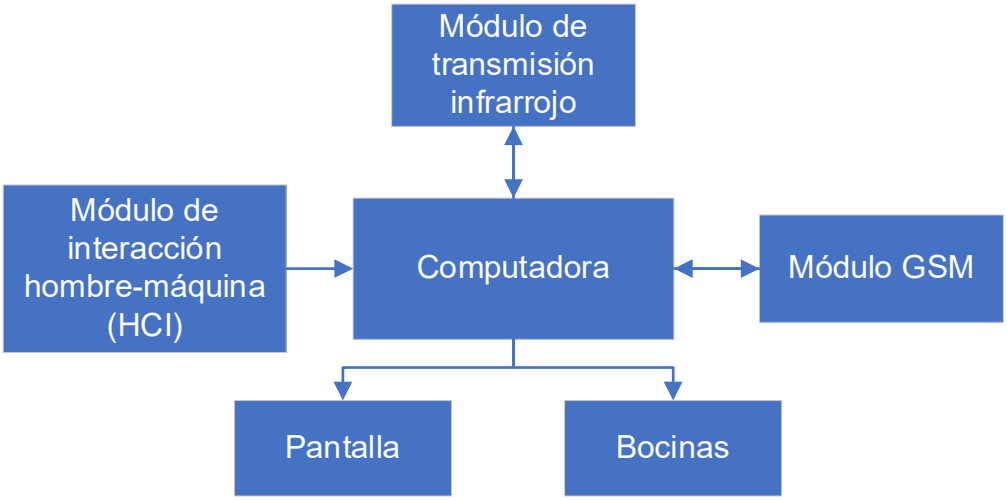


Ilustración 3.1 Diagrama de bloques del SAAC.

Módulo de interacción hombre-máquina (HCI, Human Computer Interface): Entiéndase como el dispositivo con el que el usuario puede manipular o interactuar el SAAC de una manera ergonómica y que existen diversas opciones ajustables a las condiciones y limitantes de los pacientes, que, pero para el sistema consiste solo en un emisor de pulsos eléctricos lógicos (niveles de voltaje altos y bajos).

Computadora: Se refiere a la máquina electrónica que recibe información por parte de módulos y periféricos para tratarla y enviarla a unidades de salidas, esto con la ayuda de operaciones lógicas y matemáticas controladas por programas informáticos. En nuestro caso, se utilizó una microcomputadora Raspberry Pi 4 modelo B, así como el lenguaje de programación de Python, con el que se obtiene la información proporcionada del módulo HCI para procesarla y mostrar una salida

ya sea en la pantalla o también en las bocinas. Con respecto al módulo de transmisión infrarrojo y el módulo GSM, existe una comunicación bidireccional por lo que se reciben datos de estos módulos y también se les transmite información.

Módulo de transmisión infrarrojo: Se entiende como el módulo que emplea LED emisores de luz infrarroja para efectuar una comunicación de sistemas entre al menos dos puntos. En el SAAC, se utilizó este módulo para permitir el control de aparatos como pantallas, ventiladores, equipos de audio, etcétera.

Módulo GSM: Es un dispositivo que se utiliza para establecer comunicación entre la computadora y el sistema GSM, por lo que se habilita de manera básica el envío y recepción de mensajes del tipo SMS. En el proyecto se usó el GA6-B, que se comunica con la Raspberry empleando el protocolo UART.

Pantalla: Llámese al dispositivo de salida que se encarga de mostrar información al usuario, ya sea en forma de imágenes o de textos por medio de un adaptador de video. Para el proyecto se empleó una pantalla de resolución HD (1280 x 720 píxeles).

Bocinas: Se entiende como el dispositivo de salida encargado de transformar la energía eléctrica en energía mecánica, que a su vez se transforma en energía acústica, para transmitir información a un usuario.

3.2 Módulo para comunicación del SAAC vía GSM

En esta sección se describirá como fue desarrollado el módulo GSM, que incorpora elementos de software y hardware. El software corresponde a la incorporación de plantillas sobre la interfaz del SAAC; y el hardware, a la inclusión de una tarjeta para la gestión de comunicación utilizando la red de telefonía celular. Para empezar, se muestra en la ilustración 3.2 el diagrama a bloques de la incorporación mencionada.

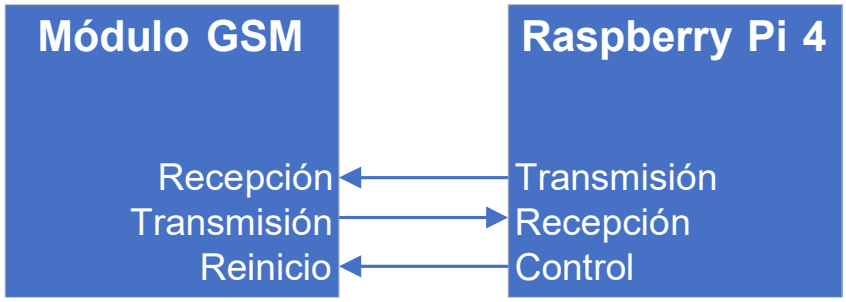


Ilustración 3.2 Diagrama a bloques de la incorporación del módulo GSM al SAAC.

El módulo GSM empleado para el envío y recepción de mensajes tipo SMS fue el GA6-B, que, debido a su sencillez y costo asequible, resulto ser la mejor opción para el proyecto. Este dispositivo cuenta con 16 pines, como se muestra en la Ilustración 3.3, de los cuales se emplearon únicamente los pines de 5V, RST, GND, URX y UTX. Por su parte, la tarjeta SIM que se empleó fue la proporcionada por Telcel, debido a que la compañía mencionada tiene la mayor cobertura de redes en todo el país, por lo que el sistema podría funcionar en cualquier parte de la República Mexicana.

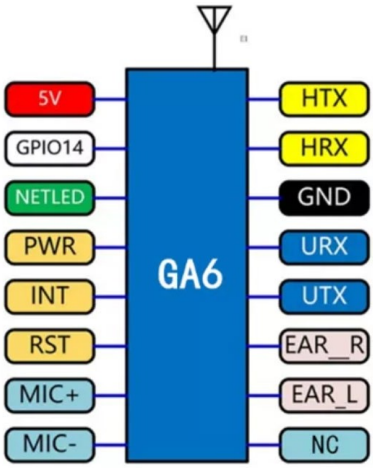


Ilustración 3.3 Disposición de pines del módulo GA6-B.

En el caso de la computadora, se optó por la Raspberry Pi 4 modelo B, ya que se necesitaba velocidad del sistema para comunicarse y controlar el GA6-B, así como para envió y recepción de mensajes en WhatsApp Web y Telegram Web. Para el control del reinicio del módulo GSM se utilizó el pin 20 de la Raspberry como salida lógica y para la comunicación serial se utilizaron los pines del UART0 (ver Ilustración 3.4).

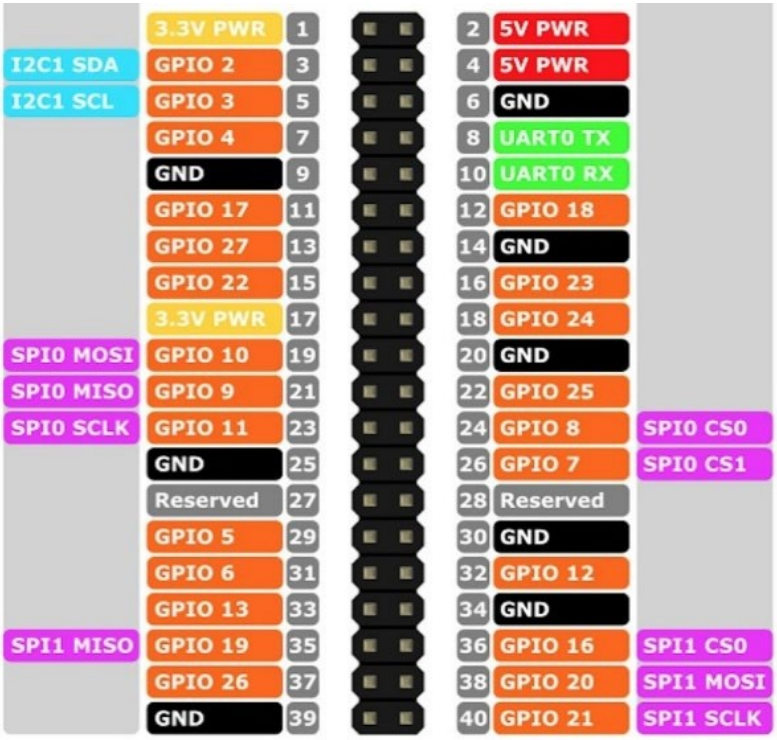


Ilustración 3.4 Disposición de pines del Raspberry Pi 4 B.

Como el voltaje de operación del UART del Raspberry Pi 4 (3.3 V) es compatible con el voltaje de operación del UART del GA6-B (2.8 V) y este último soporta voltajes de hasta 5 V, se conectaron los pines de ambos dispositivos de manera

directa; sin embargo, para el caso de la alimentación y del RST fue necesario incluir una circuitería para ajustar los niveles de voltaje a 3.3 V (ver Ilustración 3.).

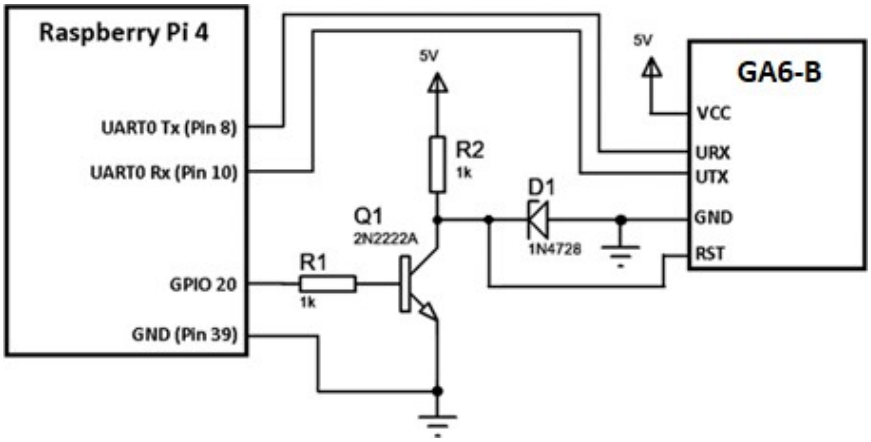


Ilustración 3.5 Diagrama de conexión del Raspberry Pi 4B y el GA6-B.

Primeramente, se suministró una tensión de 5 V al GA6-B a través de una fuente de alimentación externa, ya que se necesitaba proporcionar 2 A al momento de enviar SMS, y los pines de alimentación de la Raspberry no pueden entregar ese nivel de corriente.

Finalmente, el pin 39 del Raspberry se conectó al GND del módulo GSM para tener el mismo retorno común.

Construcción de placa electrónica

En esta sección se describirá como fue fabricada la placa electrónica que incorpora todos los componentes del circuito para la comunicación del módulo GSM con el SAAC.

Una vez que se tiene el circuito electrónico para la comunicación y control del GA6-B, se procedió a realizar su diseño de la placa de circuito impreso o PCB (printed circuit board, por sus siglas en inglés), usando la aplicación de Proteus. Como se observa en la Ilustración 3.6, se hizo el diseño en una placa rectangular de tamaño 67.5 x 35 mm, dejando en las 4 esquinas un espacio para la incorporación de los tornillos de sujeción.

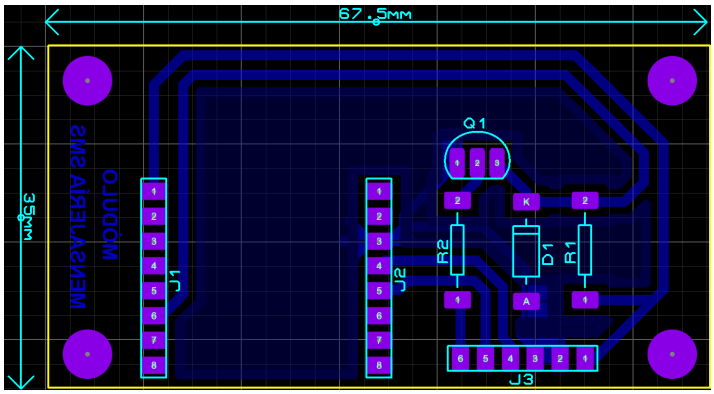


Ilustración 3.6 Diseño PCB del circuito de control del GA6-B.

3.3 Mapas de navegación

Para comenzar, se presenta una versión parcial del mapa de navegación del sistema (ilustración 3.7), que muestra parte de los bloques ya existentes en el SACC, así como las nuevas trayectorias desarrolladas en este trabajo.

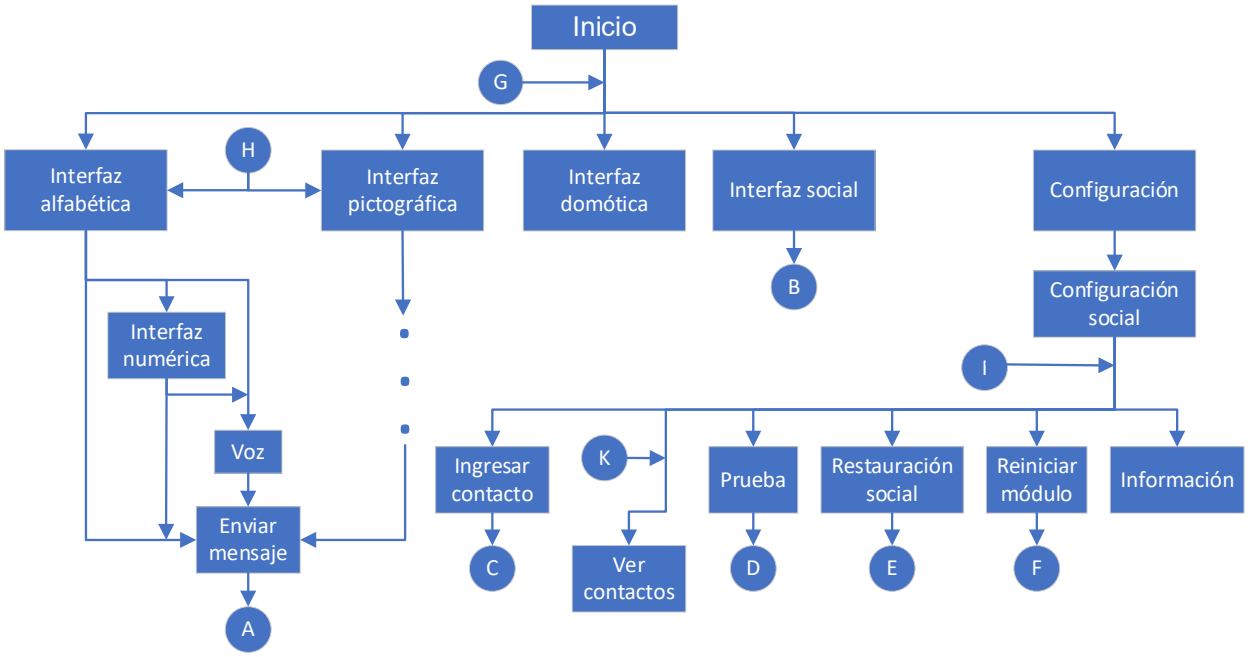


Ilustración 3.7 Mapa de navegación (sección inicial).

El primer elemento denominado “Inicio” se refiere a la visualización de la plantilla de selección de interfaz (ver ilustración 3.8), en la que se tienen cinco rutas principales, que son la interfaz alfabética, pictográfica, domótica, social y la configuración. Exceptuando la sección de domótica, se agregaron características de mensajería SMS y redes sociales en las demás rutas.

Ilustración 3.8 Selección de interfaz.

Cabe señalar que cada plantilla o elemento del mapa de navegación tiene un botón de regreso, el cual dirige al usuario a un bloque o plantilla anterior. Por el lado de las redes sociales se utilizó el navegador de Chromium, por lo que la

interfaz en esta trayectoria, en lugar de mostrar plantillas, genera un espacio para presentar una pestaña del navegador, y un espacio donde se colocaron elementos para la interacción con algunas opciones para el envío y revisión de mensajes, lo cual se describirá más adelante con mayor detalle.

3.3.1 Interfaz alfabética

Con el fin de tener un orden, se explicarán los elementos de navegación de la interfaz alfabética (ilustración 3.9) en un orden de izquierda a derecha. Esta plantilla únicamente tiene la modificación (con respecto a su homónimo original) de que tiene un símbolo entre el apartado “Voz” y “Nueva frase”, y que se denominará como “Enviar”. El motivo por el que no se escribió el nombre y se caracterizó con un pictograma fue a causa de que los elementos horizontales quedarían muy próximos y el cursor llegaría a encimarse con opciones no deseadas.

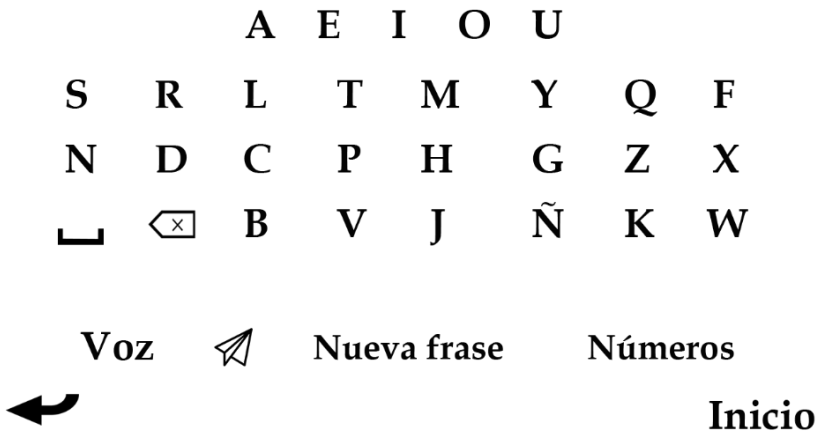


Ilustración 3.9 Interfaz alfabética

Al acceder a la parte de “Voz”, el sistema nos dirigirá a “Repetir frase” (ver Ilustración 3.10) en donde, de igual manera, se incorporó el botón “Enviar”. De esta forma, si el operador del SAAC redactaba una frase y quería escuchar como sonaba, no era necesario regresar a la interfaz alfabética para enviar el mensaje.

Ilustración 3.10 Repetir frase

Por su lado, al seleccionar “Números”, el sistema guiará hacia la interfaz numérica, divisada en la Ilustración 3.11, en donde se tiene el añadido del símbolo para enviar el mensaje desde ese elemento y si se quiere escuchar el mensaje, se puede emplear la opción “Voz”, que de igual forma nos llevará a la plantilla “Repetir frase”.

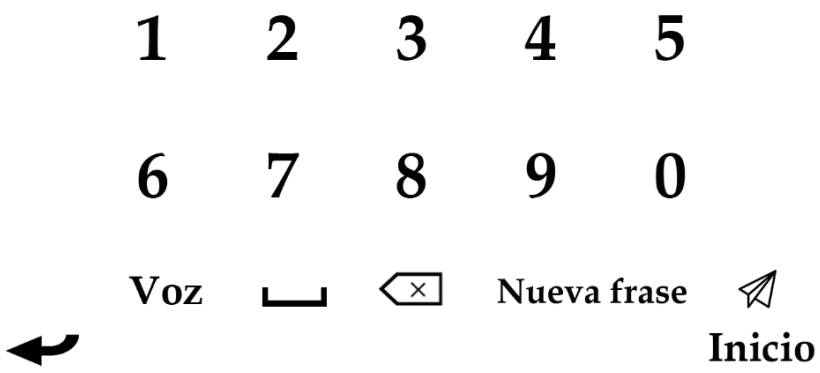


Ilustración 3.11 Interfaz numérica

Si se analiza una vez más el mapa de navegación, se intuye que el pictograma “Enviar” ya sea en la interfaz alfabética y numérica o en el apartado “Repetir frase”, continúan en la sección A (ver Ilustración 3.8).

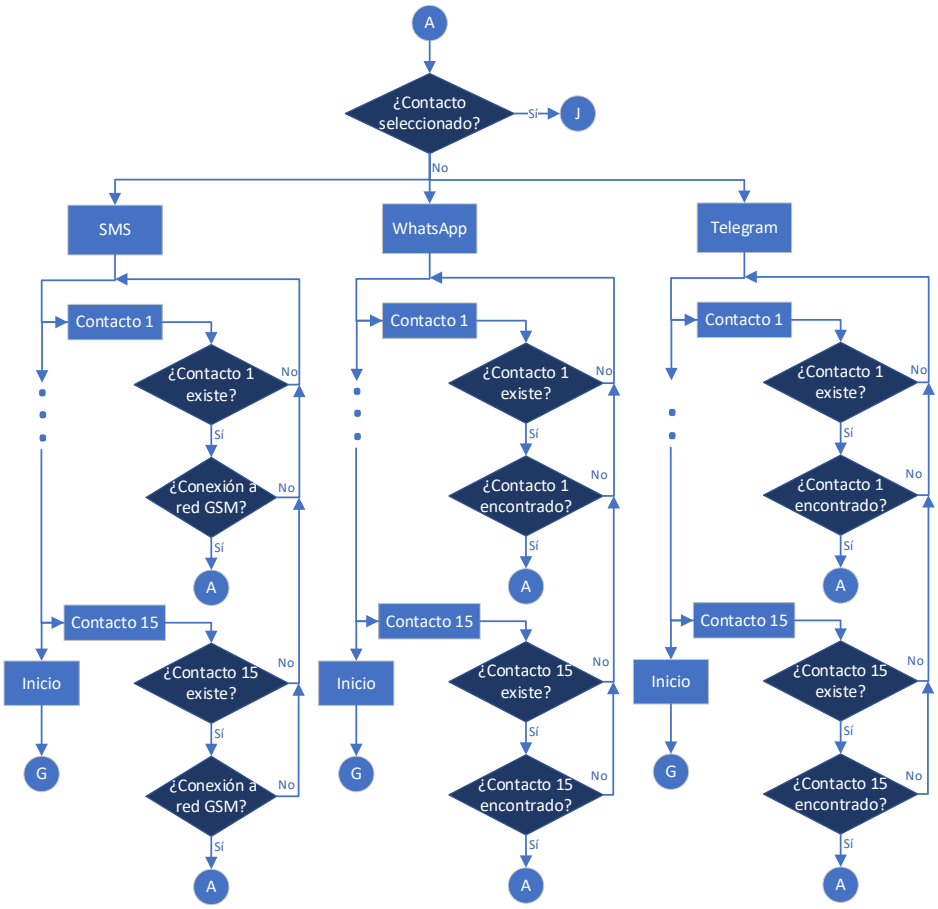


Ilustración 3.12 Mapa de navegación (sección A)

En esta continuación del mapa, se tiene al elemento “Selección de mensajería”, y que, como se observa en la Ilustración 3.13, no es nada más que una plantilla con las 3 opciones para mandar un mensaje: a través de la red GSM (SMS), por medio de WhatsApp Web o por Telegram Web.

Ilustración 3.13 Selección de mensajería

Sin importar cual opción se elija, el siguiente apartado en aparecer será el que se ve en la Ilustración 3.14, referente a la selección de contactos. De acuerdo a la sección A del mapa de navegación, el sistema tiene la capacidad de almacenar hasta 15 contactos, y cada uno de ellos tiene la misma interacción: se evalúa si la casilla elegida por el cursor es un contacto existente o si se trata de un espacio vacío.

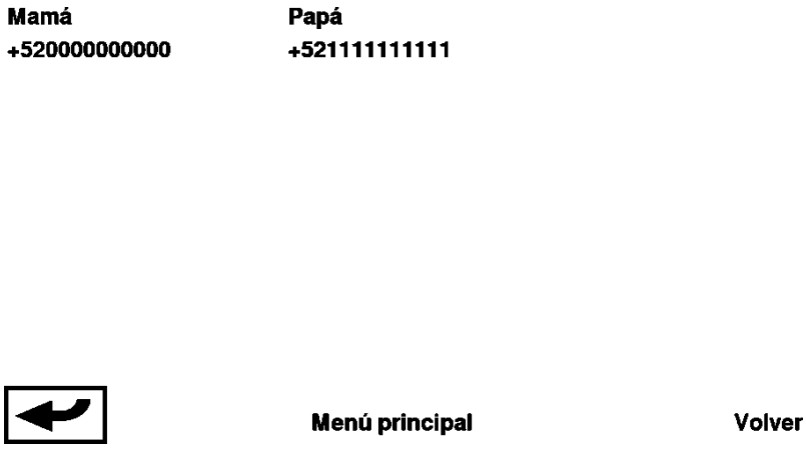


Ilustración 3.14 Selección de contactos

Para la ruta de SMS, si el contacto existe, El mensaje se enviará por medio de la red GSM se retornará al cabo de 3 segundos al apartado de “Selección de mensajería”, dándole al usuario la opción de enviar el mismo mensaje por otro medio. Sin embargo, si se trata de un espacio vacío, se le indicará al usuario para que seleccione un contacto que exista. Se puede dar lugar a un tercer caso, en el que se seleccione un contacto, pero no se tenga conexión con algún operador telefónico o que el módulo GSM se encuentre desconectado, así como en el

segundo caso, el SAAC le indicará al usuario sobre el fallo en la conexión con un operador telefónico.

Del lado de la ruta de WhatsApp y Telegram, se mantiene la notificación para una selección inexistente, de lo contrario, en lugar de un pictograma, mostrará la interfaz de WhatsApp Web y Telegram Web, en donde se enviará el mensaje automáticamente. Una vez realizado el envío, el sistema retornará a la “Selección de mensajería”.

3.3.2 Interfaz pictográfica

Regresando al mapa de navegación de la Ilustración 3.7, se tiene como siguiente opción a la interfaz pictográfica, la cual contiene pictogramas que se usan para la generación de frases. En este caso las plantillas del SAAC original no fueron modificadas, y solo hasta que se tiene una oración se presenta la alternativa para usar alguna opción de envío de mensajes.

3.3.2 Interfaz social

Excluyendo a la ruta de la interfaz domótica, (puesto a que no tiene ninguna relación con la mensajería) se proseguirá con la opción de la interfaz social, que esta descrita en el mapa de navegación de la sección B (ver Ilustración 3.15), cuyo apartado inicia con la misma platilla de la Ilustración 3.13. Cuando se haya elegido la interfaz de mensajería (SMS, WhatsApp o Telegram), el SAAC presentará la plantilla de “Selección de contactos” que se ha revisado en la sección A.

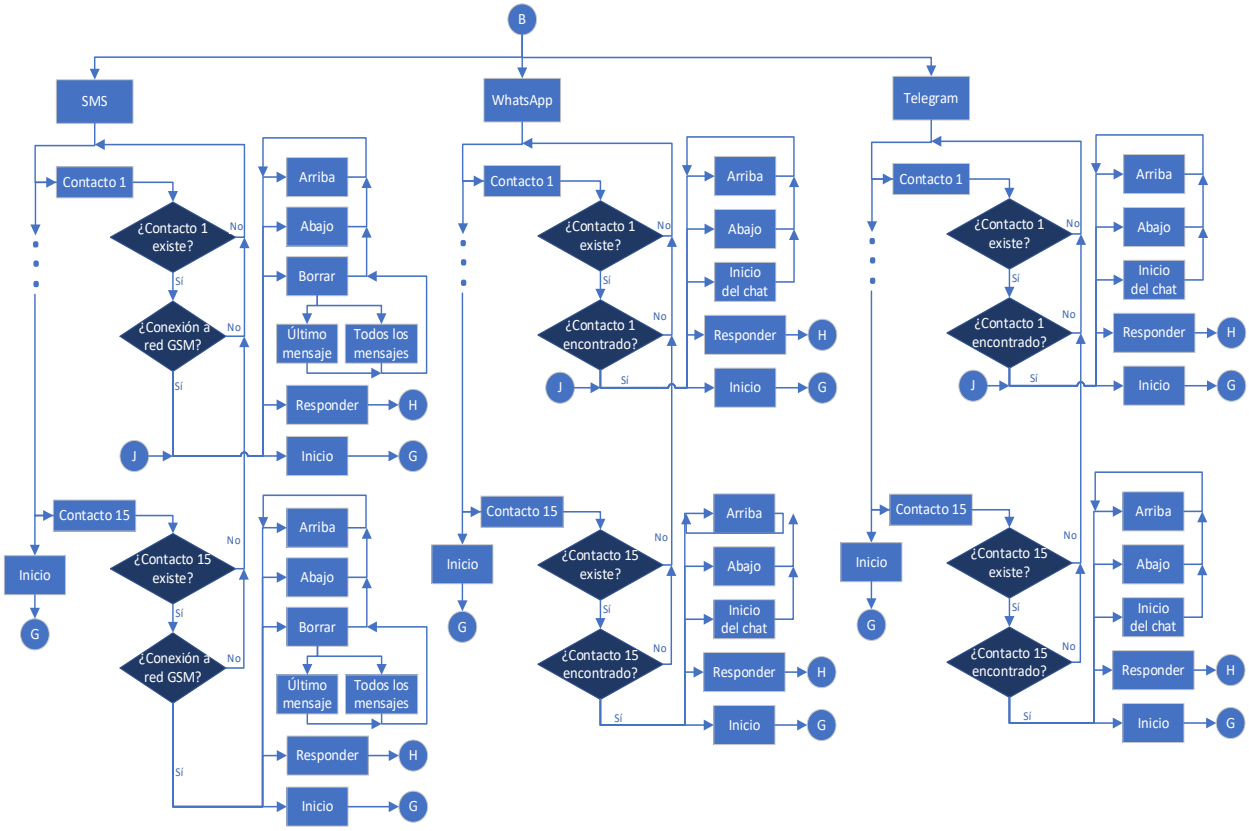


Ilustración 3.15 Mapa de navegación (sección B).

Para la ramificación de la mensajería por SMS, se tienen dos tomas de decisiones, independientemente del número de contacto, se evaluará si la selección es correcta (existe un contacto agregado en la casilla elegida) y si se tiene conexión con el operador telefónico. En caso de una negativa en cualquiera de las evaluaciones, el sistema regresará al usuario a la “Selección de contactos”, con la correspondiente notificación. Sin embargo, si se realiza la elección correcta y se tiene buena señal, se expondrá el chat del contacto escogido, como se observa en la Ilustración 3.16.

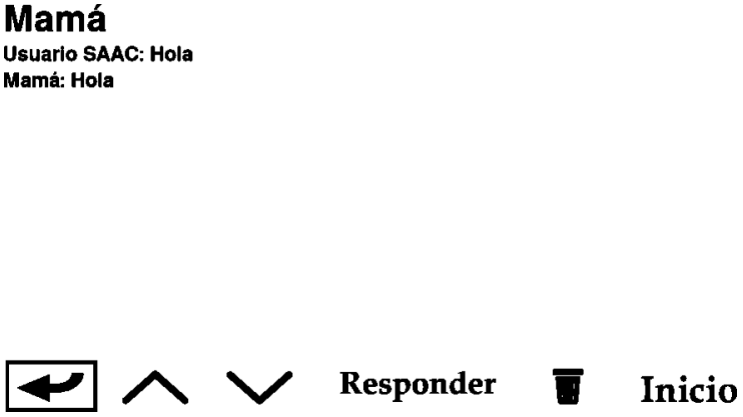


Ilustración 3.16 Chat de SMS

Para la plantilla de los chats de mensajería SMS, se poseen elementos importantes: primeramente, se cuenta con el botón de regreso cuya función se explicó en los inicios de este capítulo, también se tiene una flecha que apunta hacia arriba y otra hacia abajo y que sirven para navegar a través de conversaciones de mayor extensión. La opción “Responder” ayuda al usuario a escribir un mensaje dirigido hacia el contacto del chat actual, por lo que no necesita retroceder hasta la ruta de la interfaz alfabética o pictográfica para redactar el texto. El icono entre “Responder” e “Inicio”, se trata de la función de borrar, que al ser elegida presentará la opción de eliminar el último mensaje escrito en la conversación o, en dado caso, todos los mensajes del chat (ver Ilustración 3.17). Finalmente, el botón de “Inicio” se ocupa como una salida rápida, ya que se dirige al globo G, ubicado en la “Selección de interfaz”; esto evita el uso repetitivo del botón de regresar. Exceptuando a “Inicio” y “Responder” todas las demás herramientas se mantienen en la plantilla del chat del contacto.

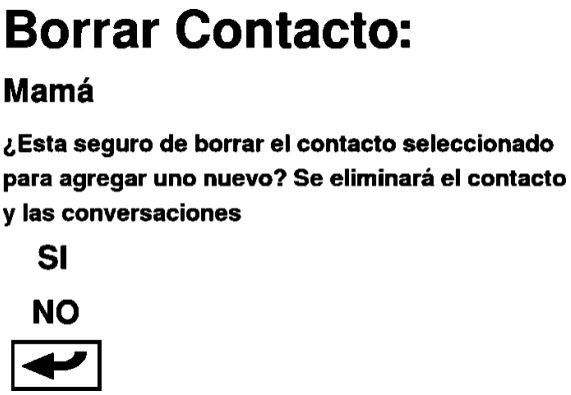


Ilustración 3.17 Borrado de mensaje en el chat de SMS.

Abordando a la opción de “Responder”, esta se dirige al globo H, que únicamente presenta la alternativa de utilizar la interfaz alfabética o pictográfica, para la redacción de un mensaje a enviar (ver Ilustración 3.18). Llegado a la sección A del mapa de navegación, se presenta una toma de decisión, en la que se analiza si ya se tiene un contacto y un tipo de mensajería seleccionada. A diferencia de la interfaz alfabética y pictográfica, en la interfaz social se obtienen estos datos desde un inicio, por lo que el sistema se redirige al globo J, enviando el mensaje y regresando al chat del contacto elegido.

Ilustración 3.18 Interfaz de respuesta.

Para la ramificación de la mensajería por WhatsApp y Telegram, se observa que tiene la misma estructura que el apartado visto de SMS, solo que en lugar de consultar la conexión con la red GSM, se examina si el nombre del contacto se ubica en WhatsApp o Telegram. En el caso de los controles, la plantilla usada es la de la Ilustración 3.19 y presenta algunas diferencias, una de ellas es que, en lugar del botón de borrado de mensaje, se tiene un botón con el que se accede hasta el final de la conversación. Esto resulta útil ya que se tiene una navegación más dinámica. Por el lado del botón “Responder”, se conserva la funcionalidad vista en SMS, aunque se cambió el nombre de la herramienta por un icono, para mantener un correcto espaciado.

Ilustración 3.19 Controles para WhatsApp Web y Telegram Web.

3.3.4 Configuración

Por último, en la interfaz de configuración se tiene un nuevo apartado, denominado "Configuración social" (ver Ilustración 3.20) y que interactúa principalmente con la mensajería SMS y los contactos añadidos al SAAC. A su vez, este apartado tiene 6 opciones diferentes, como se ve en la Ilustración 3.21.

Ilustración 3.20 Interfaz de configuración.

Ilustración 3.21 Configuración social.

La primera opción es "Ingresar contacto", cuyo mapa de navegación se muestra en la Ilustración 3.22.

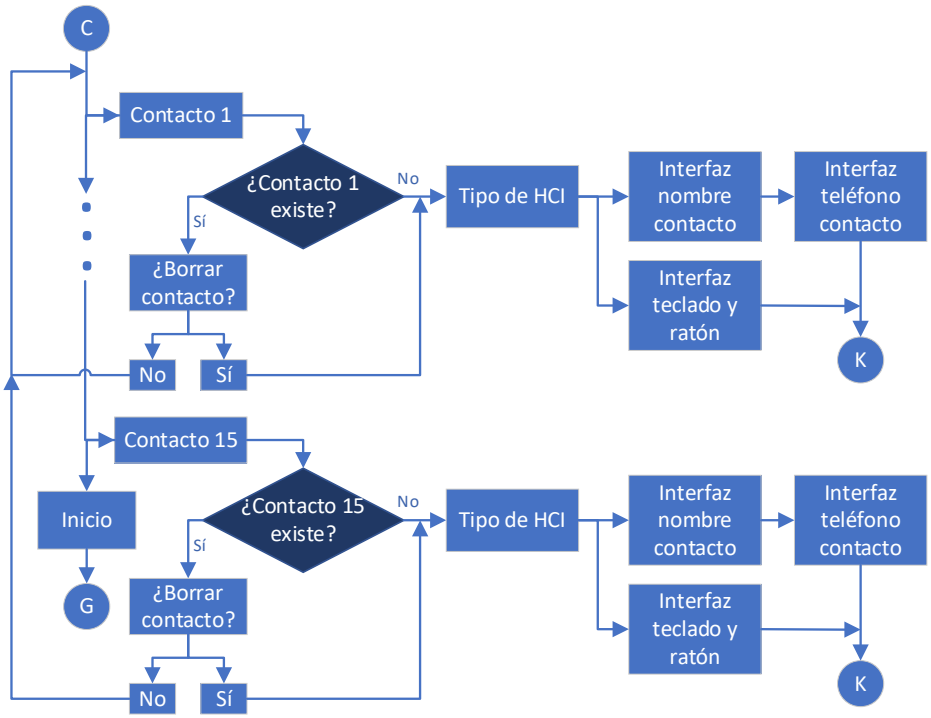


Ilustración 3.22 Mapa de navegación (sección C).

Lo primero que se observa en el mapa, es que dependiendo del contacto que se elija, se tomará una decisión: Si existe, se ejecutará la interfaz de la Ilustración 3.23, para que el usuario decida si se eliminará. Si la casilla escogida se encuentra vacía o si el usuario descarta a la selección, el SAAC mostrará la plantilla de la Ilustración 3.24, con el objetivo de que se decida si el contacto se ingresará empleando teclado y ratón o un botón pulsador (o algún otro dispositivo HCI adaptado al SAAC).

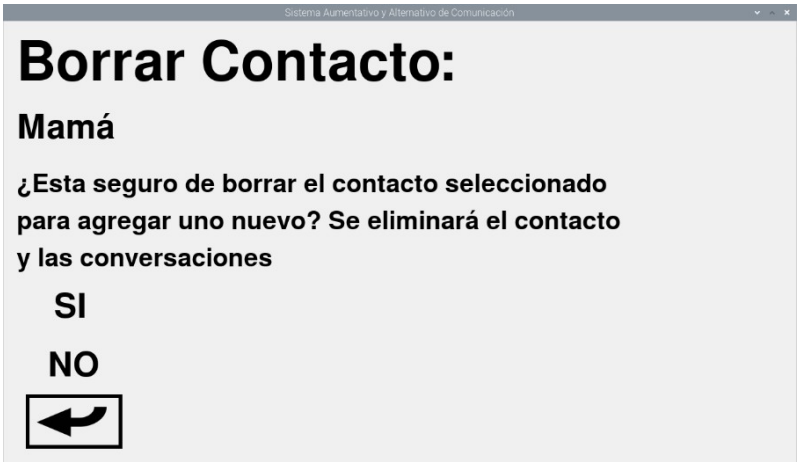


Ilustración 3.23 Borrado de contacto.

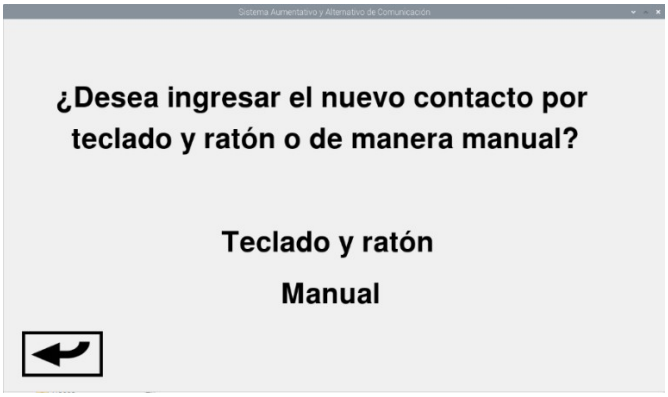


Ilustración 3.24 Selección del dispositivo HCI.

Para la primera ruta, se usa una interfaz parecida a la alfabética (ver Ilustración 3.25), pero solo con 2 controles: “Nueva frase”, que funciona para limpiar el nombre completo y “Guardar”, que confirma el nombre con el que se almacenará el contacto. Una vez que se tiene el nombre, se ingresa el número con una plantilla parecida a la interfaz numérica (ver Ilustración 3.26), pero con herramientas reducidas. Una vez que se guarda el teléfono, se notifica al usuario sobre los datos del contacto guardado.

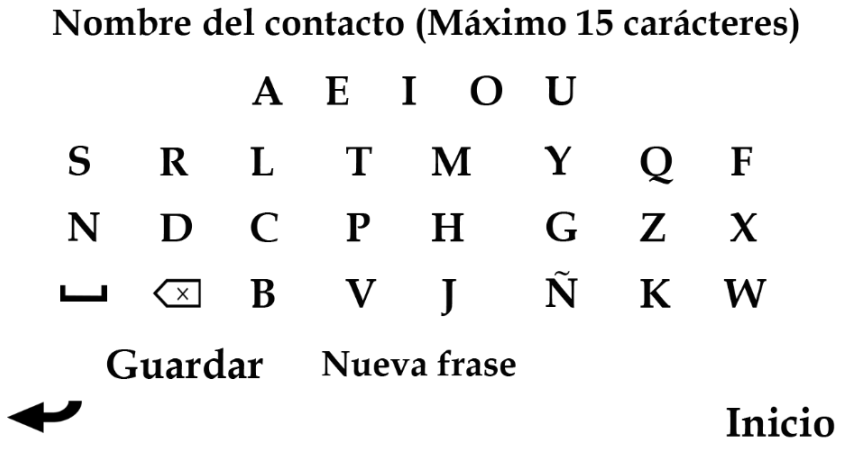


Ilustración 3.25 Interfaz para el nombre del contacto.

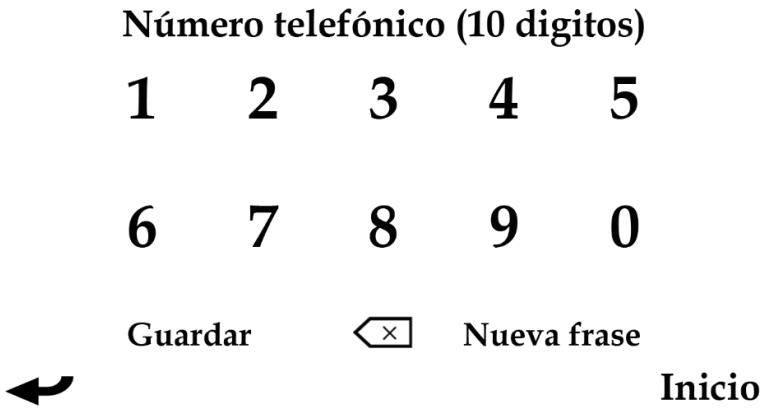


Ilustración 3.26 Interfaz para el teléfono del contacto.

Para la segunda ruta, se emplea el ratón para seleccionar el recuadro en donde se escribirán los datos, ya sea el nombre o el número telefónico (ver Ilustración 3.25). Si el sistema detecta alguna discrepancia, por ejemplo, la falta de dígitos en el teléfono o la ausencia de un nombre, arrojará un mensaje de error. Para guardar el contacto, únicamente se da clic en el botón “Confirmar”.

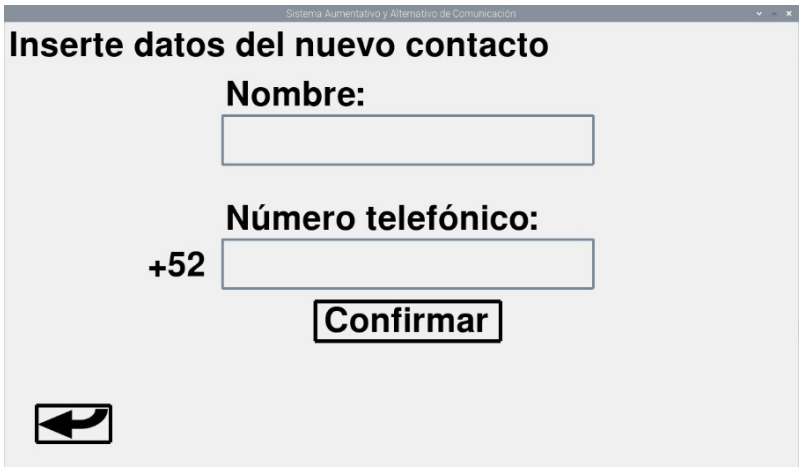


Ilustración 3.27 Interfaz para el teclado y ratón.

Al momento de guardar un contacto nuevo, sin importar la interfaz que se haya elegido, el sistema se dirigirá al globo K, que corresponde a la plantilla de “Ver Contactos”, mostrada en la Ilustración 3.28.

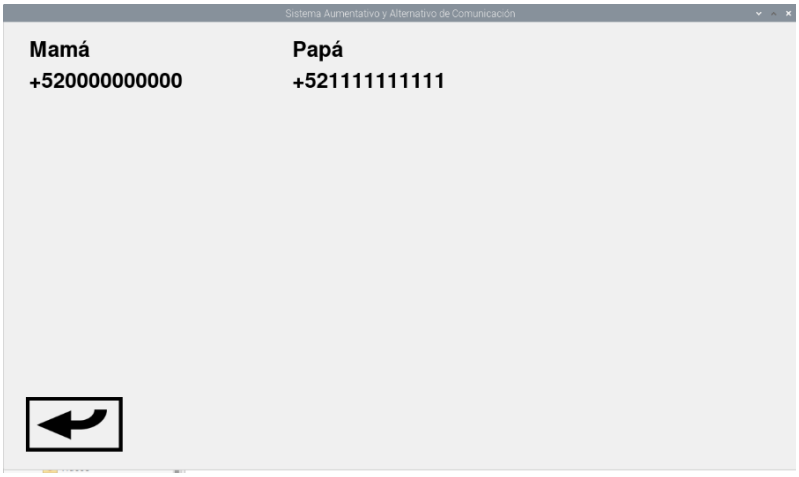


Ilustración 3.28 Ver contactos.

La ruta de “Prueba” es una función que ayuda a facilitar la comunicación entre el usuario y algún contacto deseado, puesto que la opción mencionada tiene como propósito enviar un mensaje del tipo SMS a un contacto a escoger. El texto a enviar es “Mensaje de prueba SMS” y como se señala en el mapa de navegación de la Ilustración 3.29, la única acción a realizar es la elección del contacto, obviando el hecho de que el contacto debe existir y la señal del módulo GSM debe ser optima.

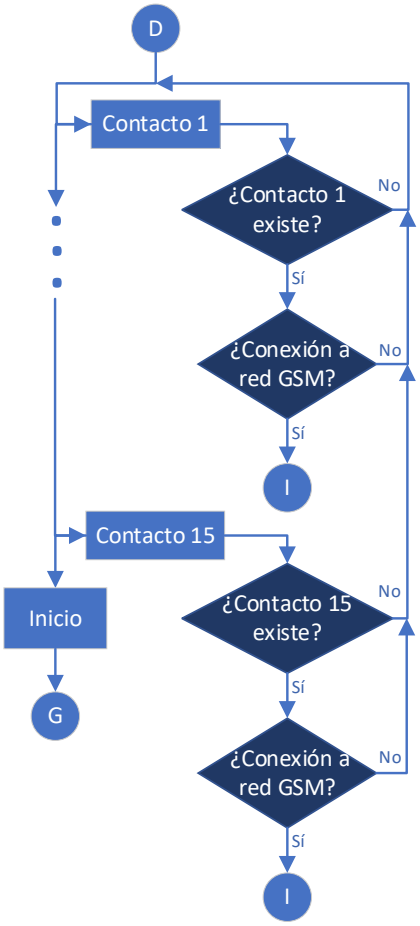


Ilustración 3.29 Mapa de navegación (sección D).

Para el apartado de “Restaurar” se concibe el mapa de la Ilustración 3.30, el cual describe las 3 opciones, no obstante, si se decide restaurar o no el aspecto social, el sistema se dirigirá al globo I que corresponde a la configuración social. Para la tercera elección, se presentará la plantilla de la Ilustración 3.31, que describe de manera breve, el funcionamiento de la herramienta de restauración.

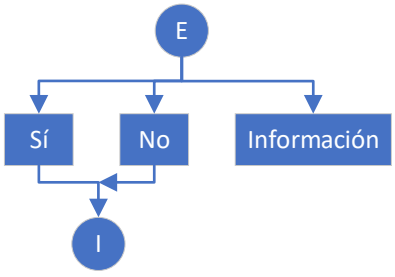


Ilustración 3.30 Mapa de navegación (sección E).

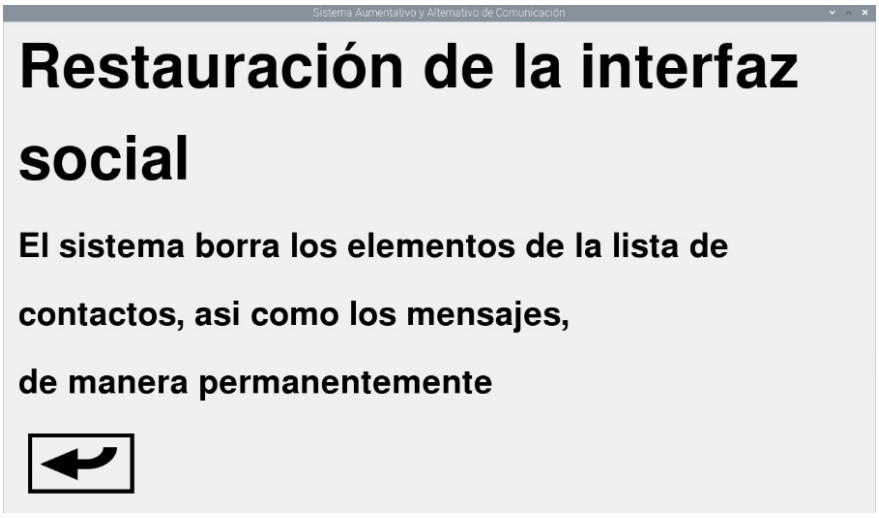


Ilustración 3.31 Información de "Restaurar".

Para el apartado de "Reiniciar módulo" se maneja una estructura parecida a la de "Restaurar" (ver Ilustración 3.32), a pesar de ello, radica la diferencia de que en esta opción se emplea únicamente cuando el módulo GA6-B no pueda conectarse a un operador telefónico y requiera reiniciarse su sistema para conectarse a la red GSM. Como se explica en la sección de información, obtenida de la Ilustración 3.33, aunque el módulo no se halle en condiciones de trabajar, el SAAC mantendrá su funcionamiento, salvó la mensajería SMS.

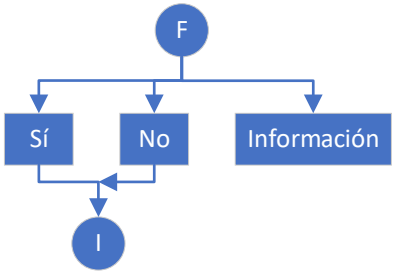


Ilustración 3.32 Mapa de navegación (sección F)

Ilustración 3.33 Información de "Reiniciar módulo".

Para concluir con la configuración social, en la Ilustración 3.34, se exhibe la información referente a la mensajería SMS, como lo son el número telefónica de la SIM que maneja el GA6-B, la compañía celular, el estado de la conexión y se explica el significado del símbolo “Enviar”, visto anteriormente en otras plantillas. Este apartado no tiene un mapa de navegación como tal, porque el sistema solo recaba la información obtenida por el dispositivo GSM y la exhibe en la interfaz del SAAC.

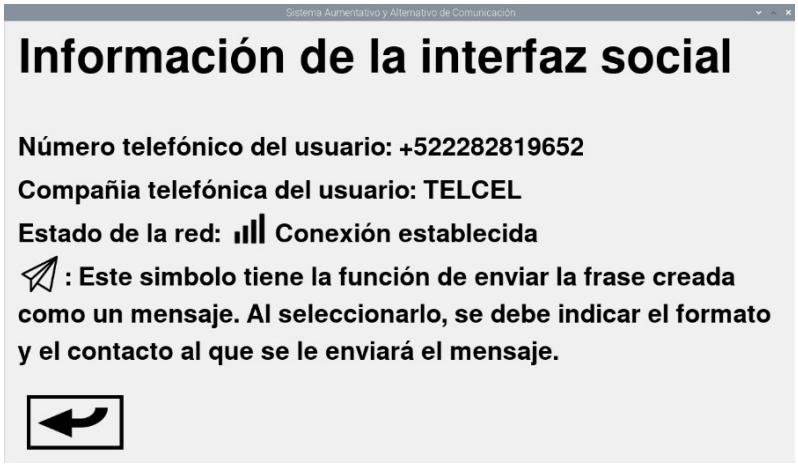


Ilustración 3.34 Información de la configuración social.

3.4 Programación

En este apartado se presentarán los diagramas de flujo de las funciones más importantes incorporadas al SAAC en este trabajo, la codificación se incluye en el Anexo de este trabajo.

3.4.1 Función “Selección de contactos”

Ahora se describirá la función “Selección de contactos”, que resulta de vital importancia puesto que dicha característica no solo está presente en la opción de configuración social, sino que abarca a las opciones para envió de mensajes (SMS, WhatsApp y Telegram) desde la interfaz alfabética, pictográfica y social.

Cuando el SAAC se encuentra en alguna sección que pasa por la opción de “Selección de contactos”, si el usuario la elige, se ejecuta el diagrama de flujo de la ilustración 3.35, lo que ocurre es que se presenta al usuario la plantilla de nombre “PictoRegreso”, y también se presentan los contactos registrados, dando como resultado lo mostrado en la ilustración 3.36. Y lo siguiente es que el sistema hace que un cursor se mueva de manera circular y de fila en fila, esperando la selección del usuario, cuando una fila es seleccionada, ahora el sistema de manera automática de desplaza de columna en columna, esperando que usuario seleccione un contacto, cuando ello ocurre, se guarda la selección que será usada en donde se requiera, que puede ser por citar algo, puede ser el envío de un SMS.

Además de los contactos, el sistema pasa por opciones para regresar a la selección de interfaz y también a la plantilla anterior (desde donde fue invocada la función de selección de contacto).

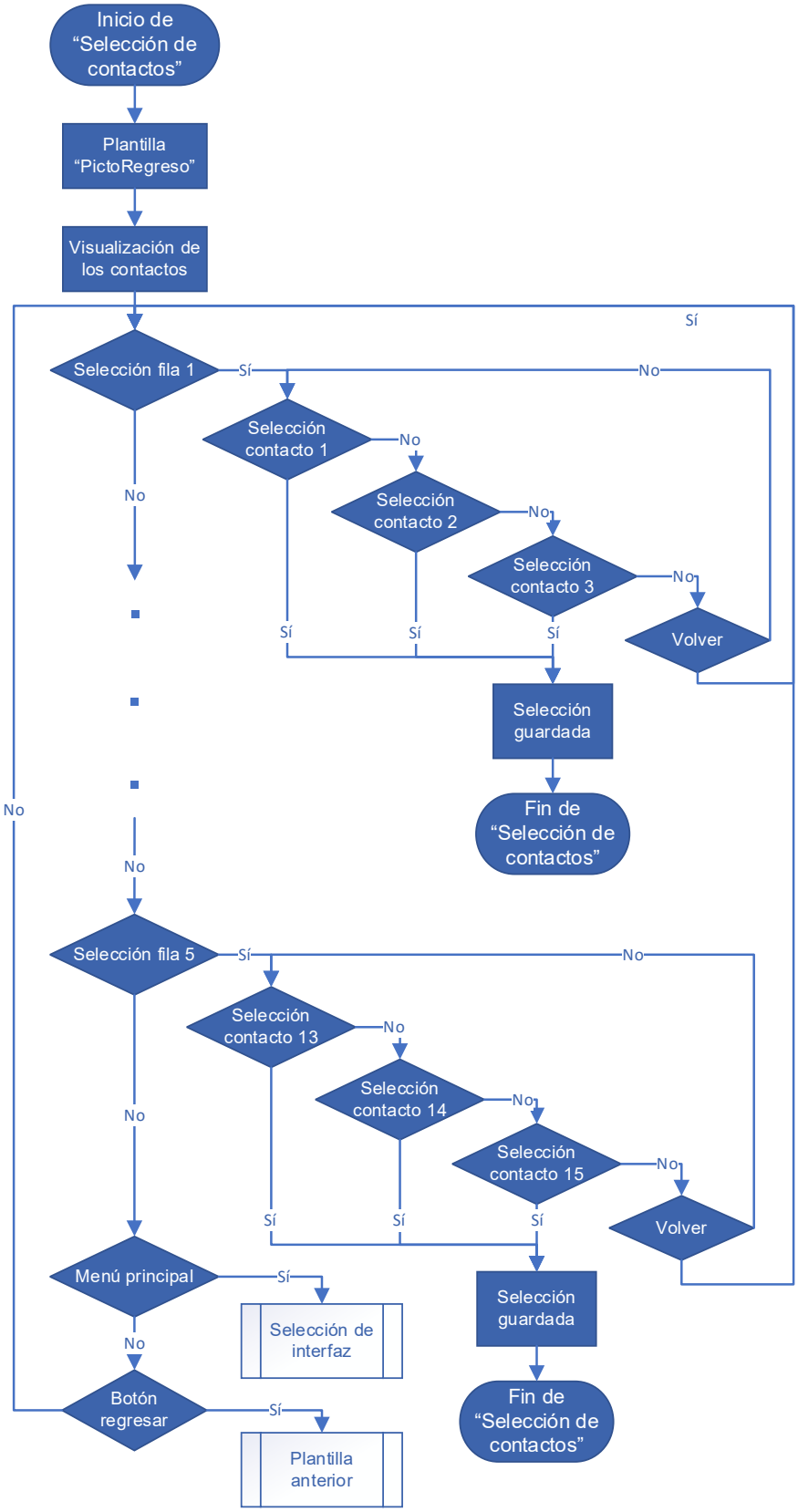


Ilustración 3.35 Diagrama de flujo de "Selección de contactos".

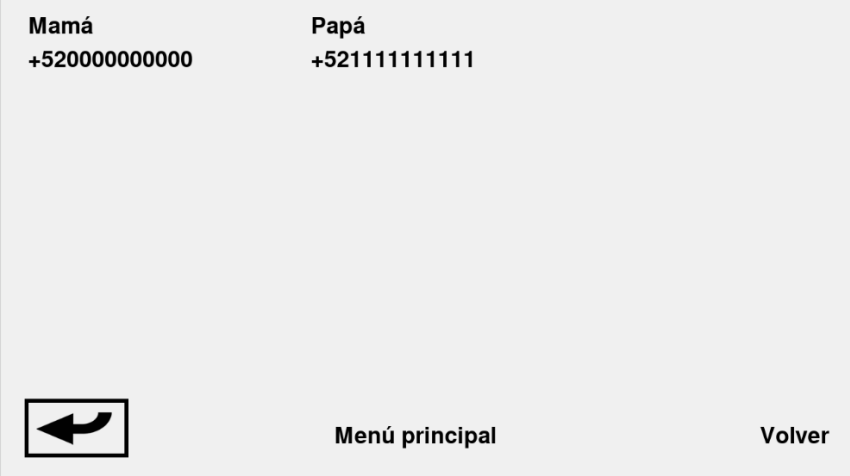


Ilustración 3.36 Selección de contactos.

3.4.2 Función “Ingresar contacto”

En esta sección se describirá la función “Ingresar contacto”, cuyo propósito radica en borrar y/o añadir un nuevo contacto en la lista de contactos. La función se puede ver representada en el diagrama de flujo de la Ilustración 3.37.

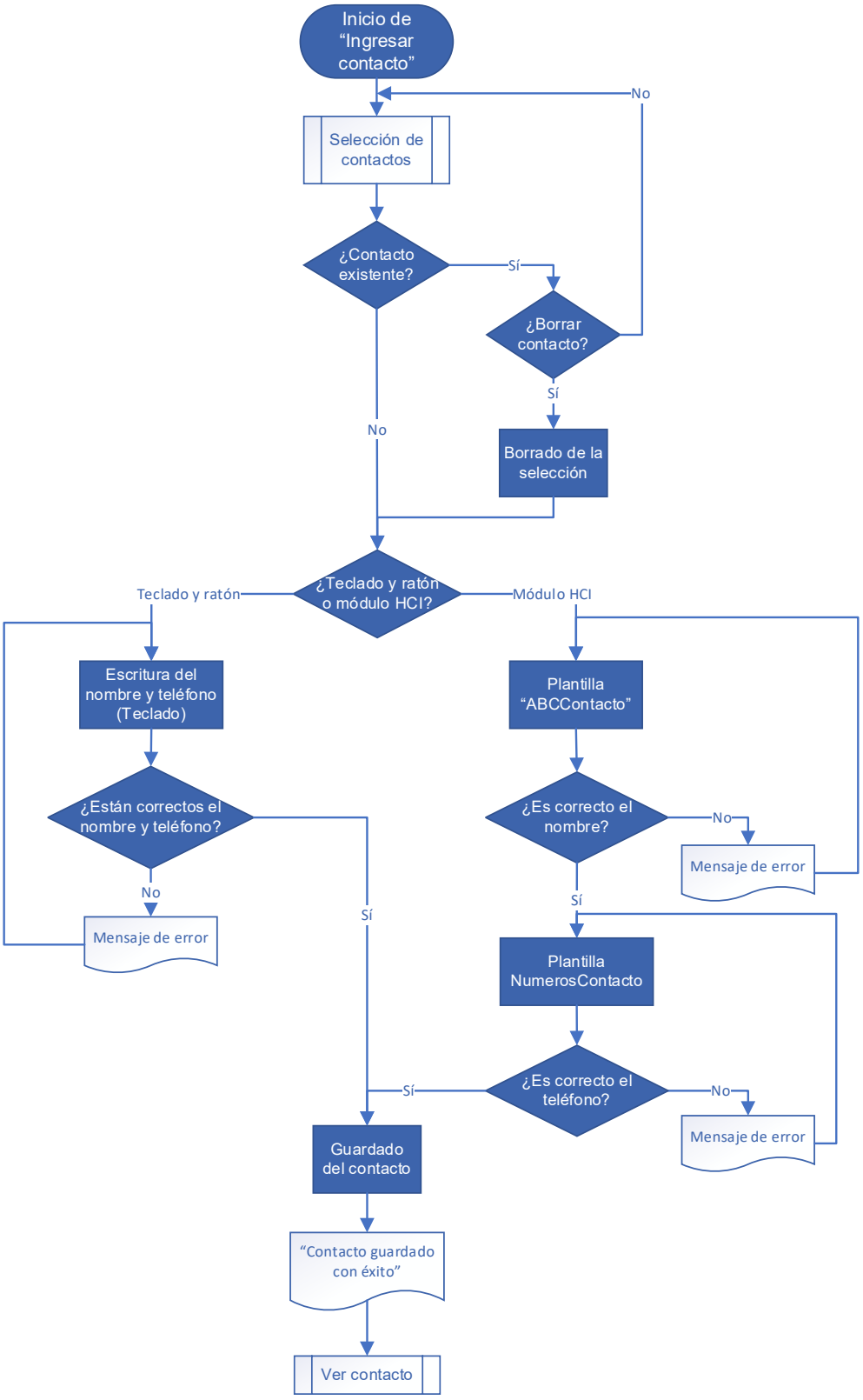


Ilustración 3.37 Diagrama de flujo de "Ingresar contacto".

Este diagrama comienza llamando a la función de “Selección de contactos”, vista anteriormente. Lo que se va a retornar de ese subproceso es el contacto elegido, el cual se va a evaluar en una toma de decisiones, para asegurar que la casilla seleccionada tiene un contacto registrado. En caso de que, si se tiene un contacto guardado, el sistema procederá a preguntar al usuario si desea eliminar la selección para en su lugar, agregar un nuevo contacto, como se ve en la Ilustración 3.38.

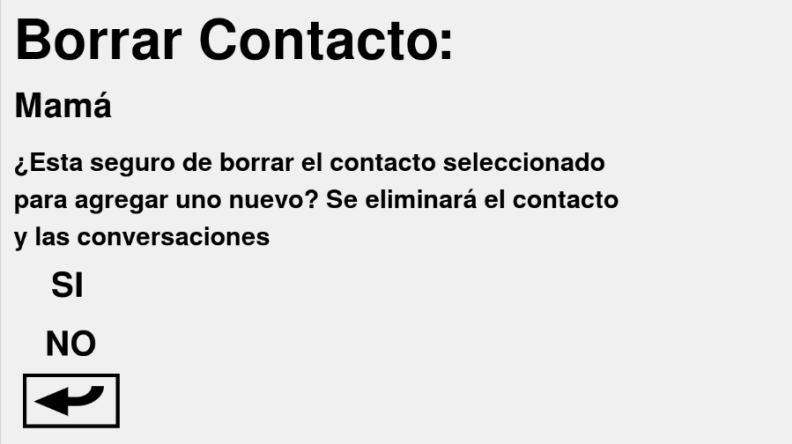


Ilustración 3.38 Borrado de un contacto.

Si la opción es “NO”, se retornará a la función de Selección de contactos para seleccionar otra casilla; sin embargo, si la opción es “SI”, se borrará el contacto y se continuará con otra toma de decisión, que trata acerca del uso de un dispositivo HCI para ingresar el contacto a ingresar. La primera opción es a través de un teclado y ratón, mientras que la segunda es el módulo HCI con el que se interactúa con el SAAC (botón pulsador).

Para la ruta de teclado y ratón, se desplegará la plantilla de la Ilustración 3.39, en donde se observan dos cuadros de texto, en el primero se añadirá el nombre y en el segundo el número telefónico a 10 dígitos. Una vez que se da clic al botón de “Confirmar”, el sistema evaluará si los datos ingresados son correctos. Si no es así, se mantendrá en la plantilla de teclado y ratón, si la información es correcta, se ejecutará el guardado del contacto, se imprimirá un letrero con la leyenda “Contacto guardado con éxito” y se redirigirá a la función “Ver contactos”. El caso contrario se puede dar cuándo no se introduce información, o en el caso de los números, no se introducen los diez dígitos respectivos.

Para la ruta de módulo HCI se desplegará una plantilla denominada “ABCContacto”, en la que se podrá generar el nombre el nombre con el método ya mencionado de barrido sobre un alfabeto. Si el dato resulta erróneo, se imprimirá un letrero con el mensaje de error correspondiente, y se volverá a ejecutar la plantilla anterior. En caso contrario, ahora se mostrará la plantilla “NumerosContacto” con el objetivo de añadir el teléfono. De igual manera, si el número es incorrecto, se visualizará el letrero de error, pero si el dato es correcto, se ejecutará el guardado del contacto, se imprimirá un letrero con la leyenda “Contacto guardado con éxito” y se redirigirá a la función “Ver contactos”.

3.4.3 Función “Ver contactos”

En esta sección se describirá la función “Ver contactos”, que se encarga únicamente de mostrar la agenda de contactos. La función se resume en el diagrama de flujo presentado en la Ilustración 3.13.

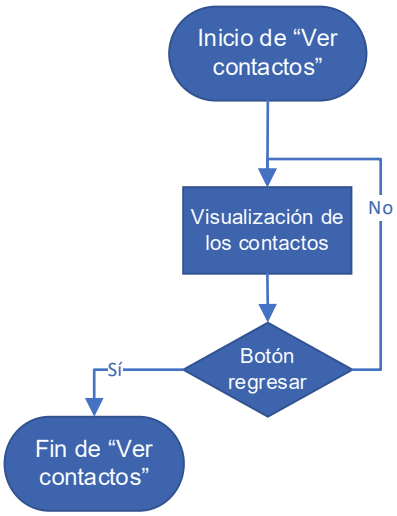


Ilustración 3.39 Diagrama de flujo de “Ver contactos”

De manera simplificada, esta función se encarga de mostrar al usuario la agenda de contactos, pero con la ventaja de solo disponer con el botón de regreso (ver Ilustración 3.40), por lo que esta función ayuda a realizar consultas sobre la información de los contactos, pero si alterar la lista.

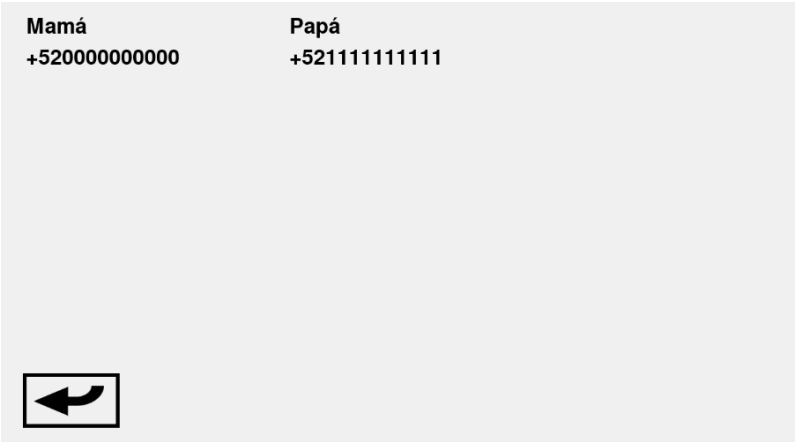


Ilustración 3.40 Visualización de la agenda de contactos

3.4.4 Función “Prueba”

En esta sección se describirá la función “Prueba”, cuyo propósito es la de realizar un envío de un mensaje SMS a un contacto de la agenda, sin la necesidad de escribir el mensaje propiamente. En la Ilustración 3.41 se observa el diagrama de flujo correspondiente.

Ilustración 3.41 Diagrama de flujo de "Prueba"

En esta función aparece un nuevo subproceso, el cual es "Enviar mensaje SMS", que, de manera simplificada, realiza la comunicación con el módulo GA6-B para comunicarse con la red GSM y enviar un mensaje hacia el contacto previamente elegido en "Selección de contactos". En caso de que la casilla seleccionada este vacía, se producirá un mensaje de error dentro del subproceso.

3.4.5 Función "Restaurar"

En esta sección se describirá la función "Restaurar", que se encarga de eliminar toda la información de la lista de contactos del sistema. La función se resume en el diagrama de flujo presentado en la Ilustración 3.42.

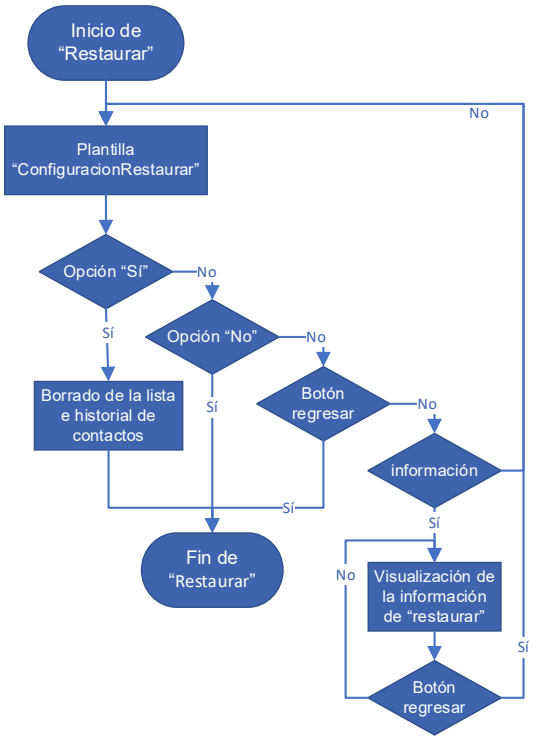


Ilustración 3.42 Diagrama de flujo de "Restaurar"

Esta función parte presentando la plantilla “ConfiguraciónRestaurar” y cuyo propósito tiene el de mostrar 4 opciones, tal como el diagrama. Para la opción “Sí”, se eliminarán tanto los datos de la lista como del historial de mensajes (de este último solo será del apartado de GSM) y se terminará la función. Para la opción “No” como para el botón de regreso, se finalizará la función. Y en el caso de información, se visualizará la información sobre las consecuencias de restaurar el sistema social. Este último, solo tiene la opción de regresar a la plantilla “ConfiguraciónRestaurar” o mantenerse en la información.

3.4.6 Función “Reiniciar módulo”

En esta sección se describirá la función “Reiniciar módulo”, que se encarga de reiniciar el dispositivo GSM por medio de un pin de la Raspberry. Esta herramienta se deberá utilizar cuando la señal de la red sea mala. La función se resume en el diagrama de flujo presentado en la Ilustración 3.43.

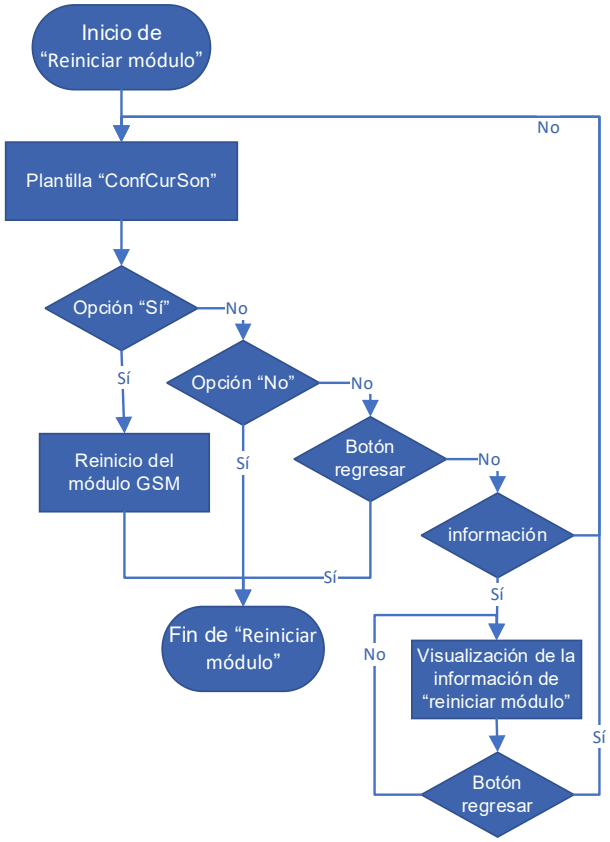


Ilustración 3.43 Diagrama de flujo de “Reiniciar módulo”

Esta función parte presentando la plantilla “ConfiguraciónRestaurar” y cuyo propósito tiene el de mostrar 4 opciones, tal como el diagrama. Para la opción “Sí”, se eliminarán tanto los datos de la lista como del historial de mensajes (de este último solo será del apartado de GSM) y se terminará la función. Para la opción “No” como para el botón de regreso, se finalizará la función. Y en el caso de información, se visualizará la información sobre las consecuencias de restaurar el sistema social. Este último, solo tiene la opción de regresar a la plantilla “ConfiguraciónRestaurar” o mantenerse en la información.

3.4.7 Función “Enviar mensaje SMS”

En esta sección se describirá la función “Enviar mensaje SMS”, que se encarga de verificar que la casilla seleccionada en la función “Selección de contactos” tenga un contacto registrado, para que de esta forma se envíe el SMS al elemento escogido. La función se resume en el diagrama de flujo presentado en la Ilustración 3.44.

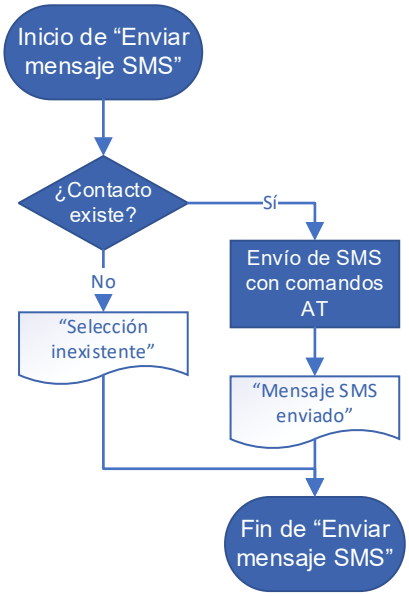


Ilustración 3.44 Diagrama de flujo de “Enviar mensaje SMS”

Al iniciar esta función se toma una decisión con respecto a la casilla seleccionada, puesto que, si la elección no tiene ningún contacto guardado, se mostrará el mensaje “elección inexistente” y se finalizará la función. En caso contrario, el sistema utilizará los comandos AT, para comunicarse con el módulo y de esta manera, se envíe el mensaje generado para el contacto escogido. Una vez se ha enviado el texto, se mostrará la notificación “Mensaje SMS enviado” y se terminará ejecutar la función.

3.4.8 Función “Enviar mensaje WhatsApp”

En esta sección se describirá la función “Enviar mensaje WhatsApp”, que se encarga de verificar que la casilla seleccionada en la función “Selección de contactos” tenga un contacto registrado, para que de esta forma se envíe el mensaje por WhatsApp Web al elemento escogido. La función se resume en el diagrama de flujo presentado en la Ilustración 3.45.

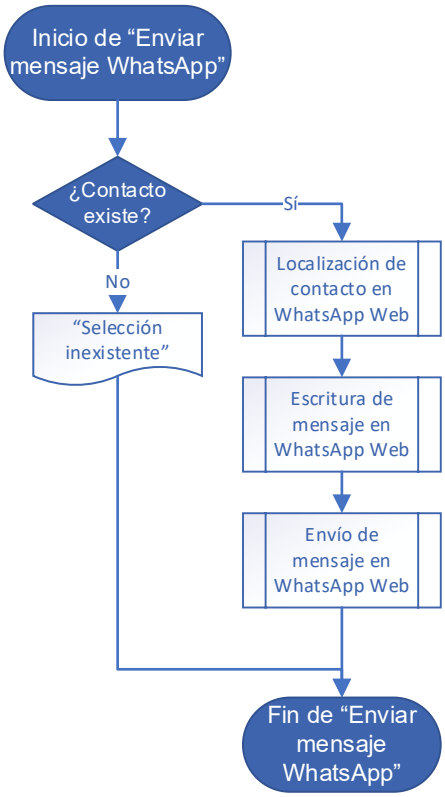


Ilustración 3.45 Diagrama de flujo de “Enviar mensaje WhatsApp”

Así como la función “Enviar mensaje SMS”, esta función evalúa si el contacto existe o no. Si la selección es correcta, se ejecutarán 3 subprocesos, que interactúan con el código HTML de WhatsApp Web para que, por medio de funciones de JavaScript, se localice y seleccione el contacto elegido, se escriba el mensaje en el cuadro de texto y se presione el botón de enviar. Una vez se ha enviado el texto, se terminará ejecutar la función.

3.4.9 Función “Enviar mensaje Telegram”

En esta sección se describirá la función “Enviar mensaje Telegram”, que se encarga de verificar que la casilla seleccionada en la función “Selección de contactos” tenga un contacto registrado, para que de esta forma se envíe el mensaje por Telegram Web al elemento escogido. La función se resume en el diagrama de flujo presentado en la Ilustración 3.46.

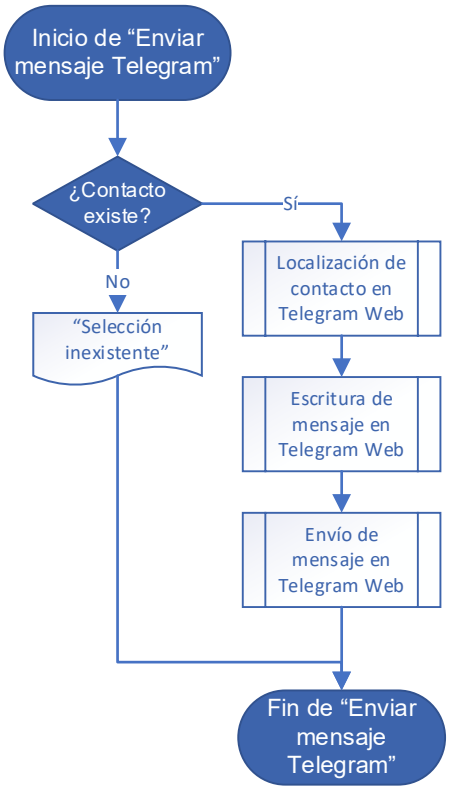


Ilustración 3.46 Diagrama de flujo de “Enviar mensaje Telegram”

Así como la función “Enviar mensaje SMS”, esta función evalúa si el contacto existe o no. Si la selección es correcta, se ejecutarán 3 subprocesos, que interactúan con el código HTML de Telegram Web para que, por medio de funciones de JavaScript, se localice y seleccione el contacto elegido, se escriba el mensaje en el cuadro de texto y se presione el botón de enviar. Una vez se ha enviado el texto, se terminará ejecutar la función.

Capítulo 4. Pruebas y resultados

En esta sección se detallarán las pruebas realizadas en el sistema, con el fin de mostrar las funcionalidades añadidas al SAAC con el que se trabajó. La estructura general consiste en describir la prueba realizada junto, así como presentar las ilustraciones que ejemplifiquen el resultado obtenido. Este capítulo se dividirá a su vez en 5 apartados: pruebas de la configuración inicial, pruebas de SMS, pruebas de WhatsApp, pruebas de Telegram y pruebas de la configuración social.

4.1 Pruebas de la configuración inicial

En este apartado se describirán las pruebas realizadas al momento de iniciar el SAAC. En estas configuraciones iniciales, el usuario no tendrá ninguna intervención puesto que el sistema actuará de forma automática y de acuerdo a la situación que se presente.

La primera característica que se experimentó fue la detección del módulo GSM, ya que el SAAC al momento de iniciar, revisará si el módulo tiene una correcta comunicación con la Raspberry. En la prueba, el GA6-B esta desacoplado, por lo que se mostró el mensaje de la Ilustración 4.1, lo que indica que el dispositivo de mensajería SMS está conectado al sistema de manera incorrecta o también, que no se pudo establecer conexión con el operador de la red telefónica. Si no se presenta ningún inconveniente, no se mostrará nada en especial.

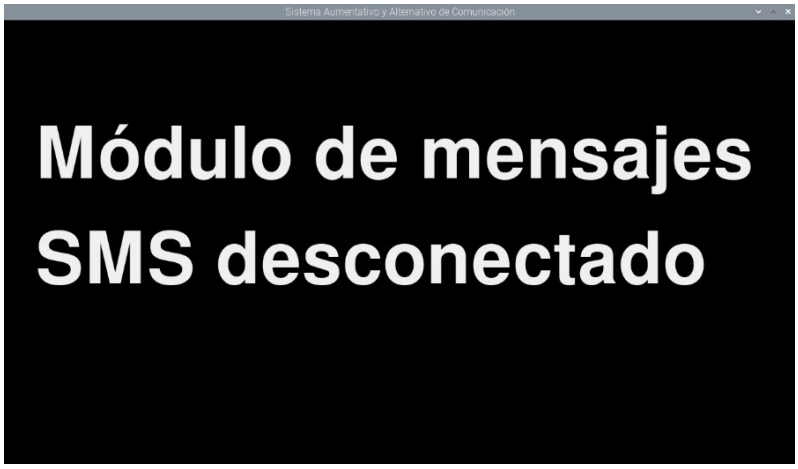


Ilustración 4.1 Notificación de módulo GSM desconectado.

Una vez que se comprueba el estado del dispositivo GSM, el SAAC de manera automática abrirá en el navegador web Chromium, WhatsApp y Telegram, cada uno en su respectiva pestaña (ver Ilustración 4.2 y 4.3).

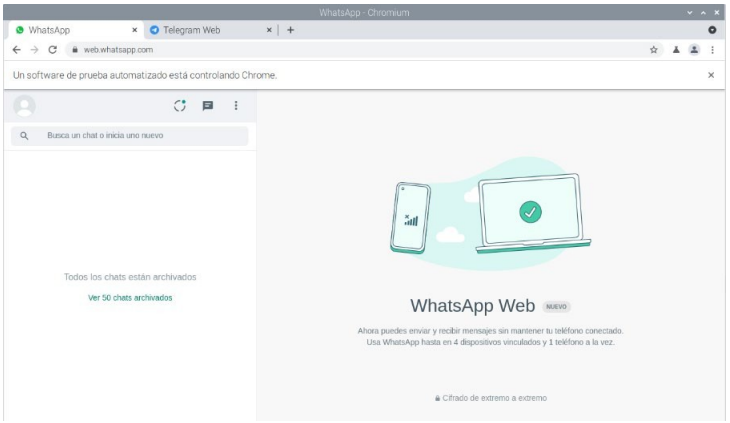


Ilustración 4.2 Pestaña de WhatsApp Web en Chromium.

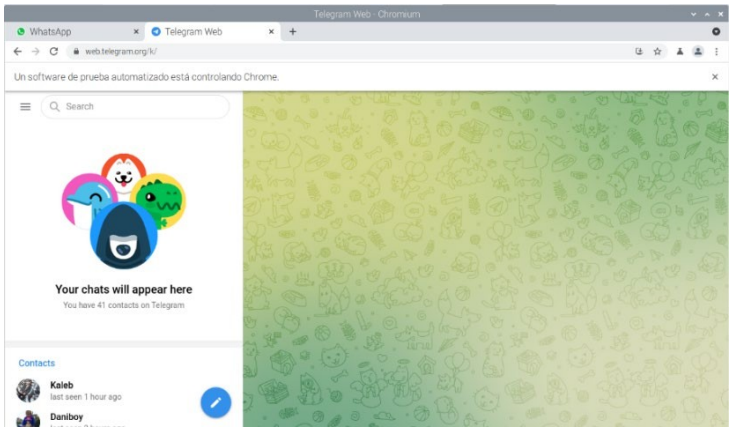


Ilustración 4.3 Pestaña de Telegram Web en Chromium.

Con las redes sociales abiertas en la web, se minimizarán las pestañas y la Raspberry interaccionará con el GA6-B para buscar los nuevos SMS. Si no hay mensajes, se iniciará el apartado de selección de interfaz, sin embargo, si hay mensajes nuevos no leídos, se notificará sobre ello, por medio de un letrero, que se visualizará en el siguiente apartado.

En un caso especial, en el que se hace la recarga de saldo, se mostrará el letrero de la Ilustración 4.4, en donde se indica el montó recargado.

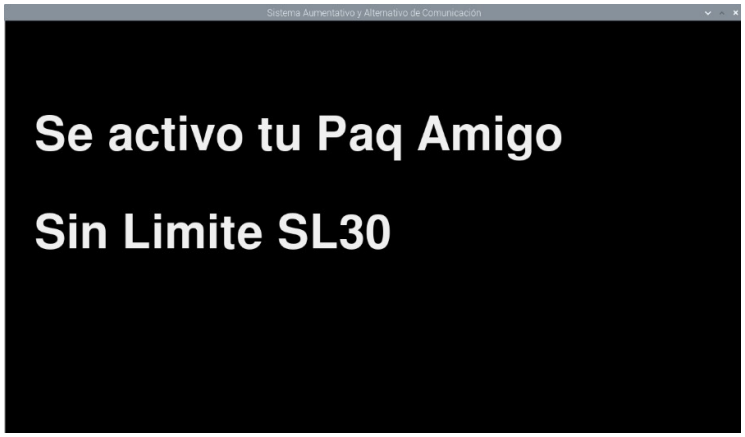


Ilustración 4.4 Mensaje de recarga de saldo.

El sistema está adaptado para checar cada cierto tiempo la señal en la red. En caso de tener buena señal se verá en la esquina superior izquierda un símbolo (Ilustración 4.5), no obstante, al desconectar el dispositivo, se vio un cambio en el símbolo (Ilustración 4.6), lo que indica tanto la perdida de señal como la desconexión del sistema.

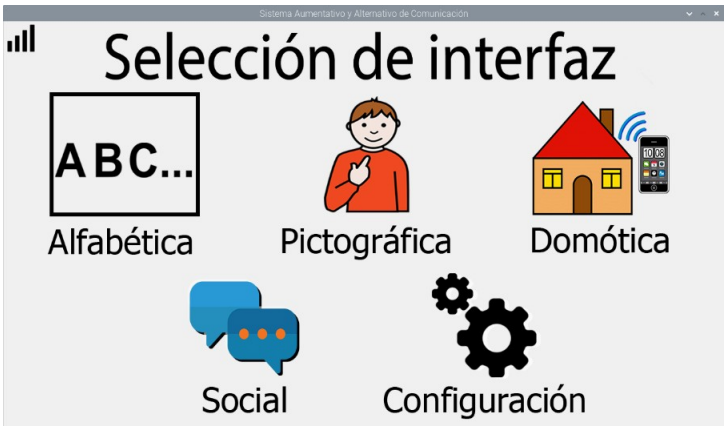


Ilustración 4.5 Icono de buena señal.

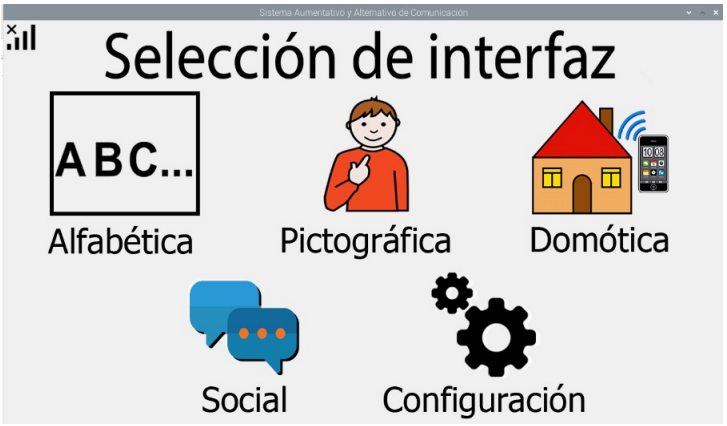


Ilustración 4.6 Icono de mala señal.

4.2 Pruebas de SMS

En este apartado se describirán las pruebas realizadas para el envío de mensajes SMS por medio de la interfaz alfabética y pictográfica, así como la visualización de los historiales de mensajes, junto con sus herramientas para la interacción con el chat.

Para la primera prueba se utilizó un teléfono celular, cuyo número telefónico estuviera registrado en la lista de contactos del SAAC, esto debido a que el sistema rechazará los mensajes entrantes de contactos no registrados. Desde el teléfono móvil se envió un SMS por lo que al cabo de unos segundos el sistema mostró la notificación de la Ilustración 4.7. Contemplando que el usuario no estuviera presente al momento de la notificación, en la selección de interfaz, abajo del tipo de señal, se cuenta con un símbolo que indica la recepción de SMS recibidos (ver Ilustración 4.8).

Ilustración 4.7 Notificación de nuevos SMS.

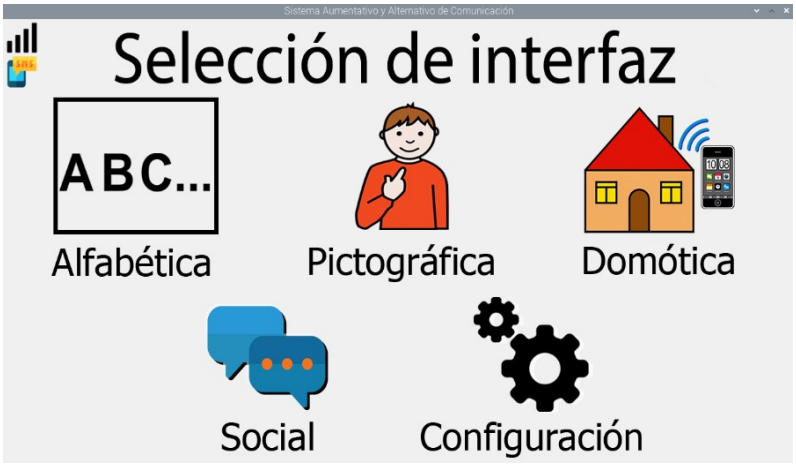


Ilustración 4.8 Selección de interfaz con el icono de nuevos SMS.

Accediendo a la interfaz social, en el apartado de mensajería SMS, se obtuvo el resultado de la Ilustración 4.9, en donde se observa que el contacto denominado “Mamá”, tiene un símbolo que indica la recepción de un nuevo SMS, el cual no ha sido leído.

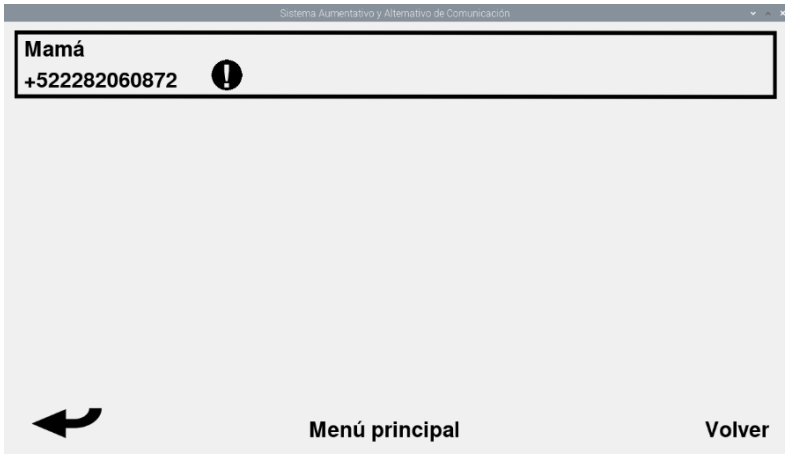


Ilustración 4.9 Notificación de nuevo SMS por parte del contacto “Mamá”.

Dentro de la interfaz del chat “Mamá” se visualiza el mensaje enviado (ver Ilustración 4.10), por lo que se comprueba la correcta recepción del mensaje en el SAAC. Ahora bien, para demostrar el envío de un SMS y recepción de este en el móvil, se utilizó la herramienta “Responder” en donde se generó la frase “Hola” ´por medio de la interfaz pictográfica. Al momento de enviar el respectivo mensaje, apareció el letrero de la Ilustración 4.11, además de que, al retornar al chat, se visualizó el mensaje en el historial de mensajes (ver Ilustración 4.12). El resultado obtenido se aprecia en la Ilustración 4.13.

Ilustración 4.10 Historial de chat antes de responder.

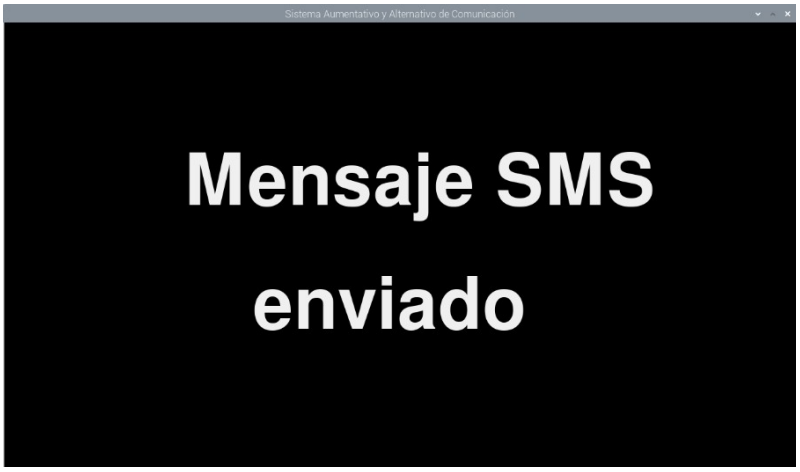


Ilustración 4.11 Notificación de envío del SMS.

Ilustración 4.12 Historial de chat después de responder.

Ilustración 4.13 Recepción del SMS en el teléfono celular.

Puesto que se comprobó que en la interfaz pictográfica el funcionamiento era el adecuado, la siguiente prueba a ejecutar fue el envío de un mensaje empleando la interfaz alfabética, para ello, se utilizó la frase “Hola 2” de la Ilustración 4.14, con el objetivo de diferenciar el texto enviado por ambas interfaces.

Ilustración 4.14 Envío del mensaje “Hola 2”.

El resultado se ve reflejado tanto en la Ilustración 4.15 como en la Ilustración 4.16, que son el chat de SMS del SAAC y del teléfono celular.

Ilustración 4.15 Perspectiva del chat desde el SAAC.

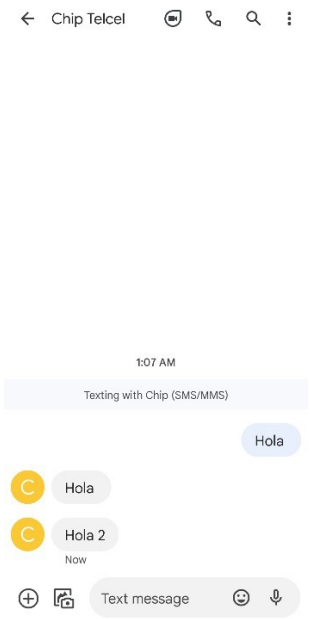


Ilustración 4.16 Perspectiva del chat desde el teléfono celular.

Con respecto a los controles del chat de SMS, se enviaron 20 mensajes al contacto “Mamá”, cada uno de ellos enumerado como se observa en la Ilustración 4.17, a causa de que se necesitaba usar un historial de mensajes grande para comprobar la funcionalidad de los botones para subir y bajar el chat, así como la eliminación del último mensaje o de todo el historial.

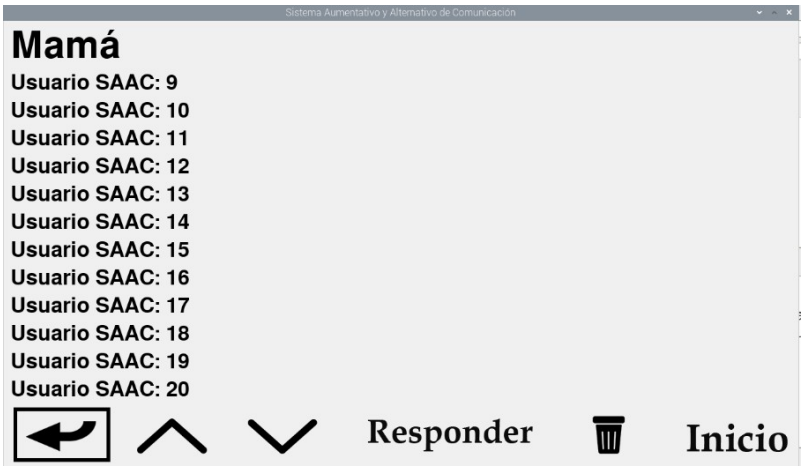


Ilustración 4.17 Historial de mensajes.

En un inicio se visualizan los mensajes desde el 9 hasta el 20, sin embargo, al presionar el botón de desplazamiento hacia arriba, el chat traslada los mensajes resultando ahora la visualización desde el mensaje 8 hasta el 19 (ver Ilustración 4.18).

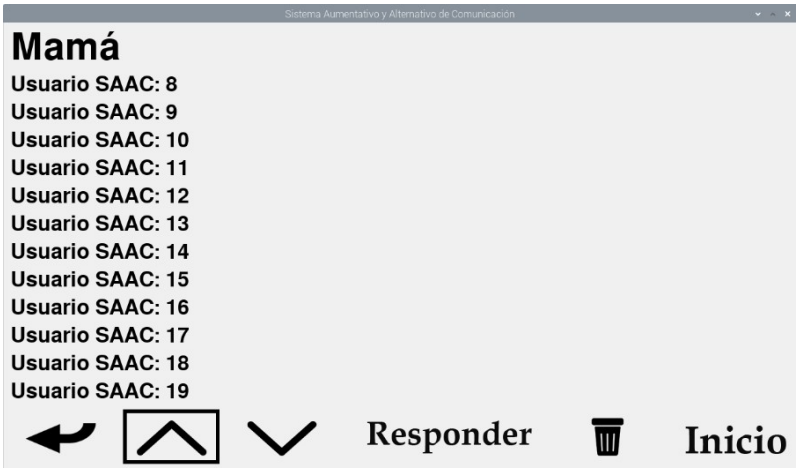


Ilustración 4.18 Botón de desplazamiento hacia arriba.

Los mensajes que se observan son desde el 8 hasta el 19, no obstante, al presionar el botón de desplazamiento hacia abajo, el historial de mensajes pasa hacia los mensajes desde 9 hasta el 20 (ver Ilustración 4.19).

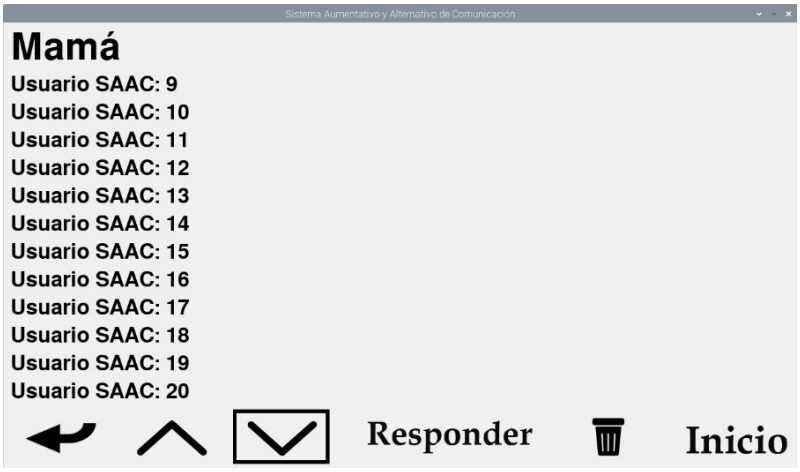


Ilustración 4.19 Botón de desplazamiento hacia abajo.

Ahora bien, al emplear la herramienta de borrar, se eligió eliminar el último mensaje (ver Ilustración 4.20), derivando en que el mensaje 20 ya no exista y se refleje como que ahora el mensaje 19 sea el más reciente (ver Ilustración 21).

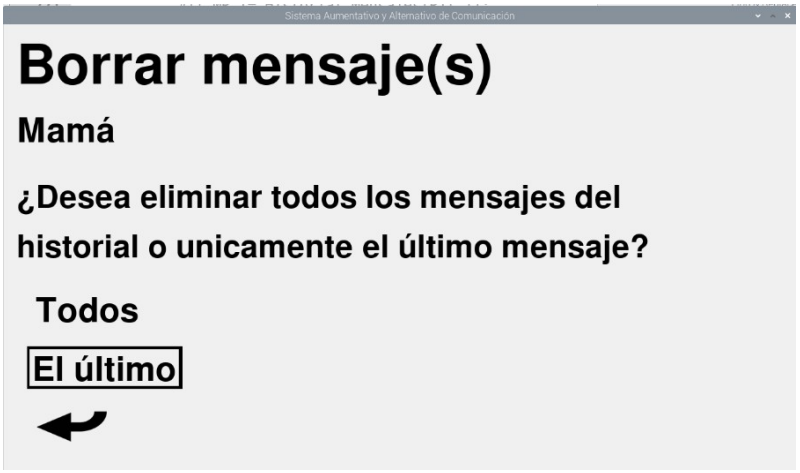


Ilustración 4.20 Eliminación del último mensaje.

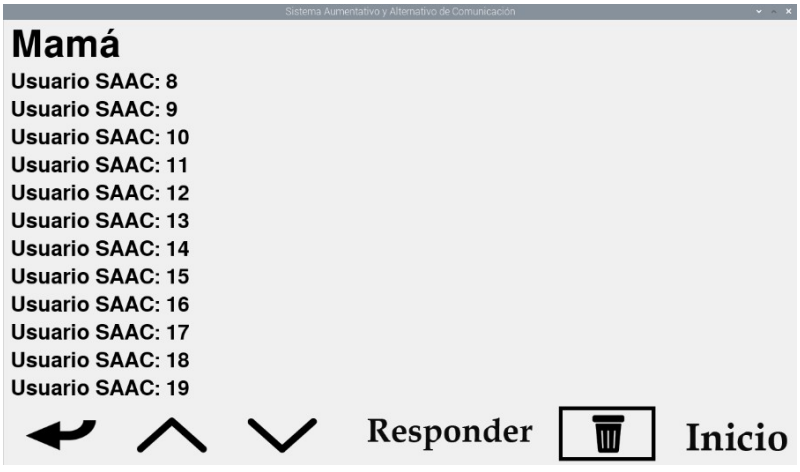


Ilustración 4.21 Resultado de la eliminación del último mensaje.

Sin embargo, si se decide eliminar todos los mensajes (ver Ilustración 4.22), el resultado obtenido será el de la Ilustración 4.23, en la que se observa que el chat se haya vacío, sin ningún mensaje por parte del usuario como del contacto “Mamá”.

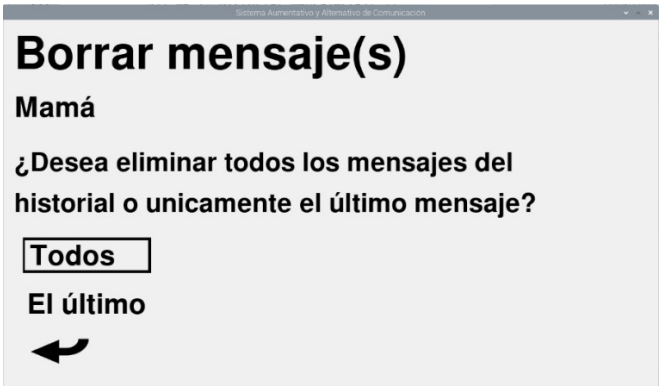


Ilustración 4.22 Eliminación de todos los mensajes.

Ilustración 4.23 Resultado de la eliminación del todos los mensajes.

Finalmente, en la Ilustración 4.24 y 4.25 Comprobó el sistema de notificaciones ante errores posible, como lo son, la selección de una casilla vacía al momento de seleccionar un contacto y la notificación que indica el mal funcionamiento del dispositivo GSM ante el envío de mensajes.

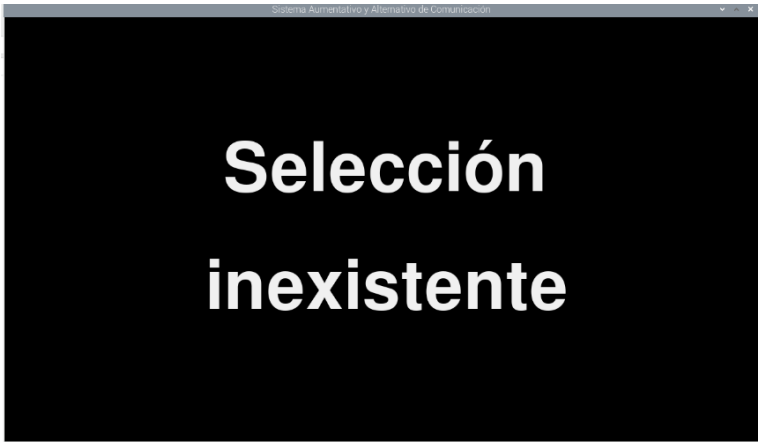


Ilustración 4.24 Notificación de una selección invalida.

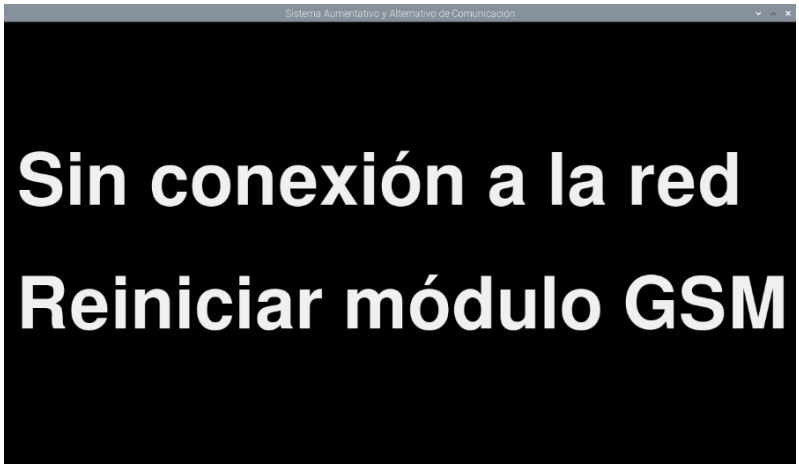


Ilustración 4.25 Notificación ante la mala señal del dispositivo GSM.

4.3 Pruebas de WhatsApp

En este apartado se describirán las pruebas realizadas para el envío de mensajes por WhatsApp por medio de la interfaz alfabética y pictográfica, así como la visualización de los historiales de mensajes en la interfaz de WhatsApp Web, junto con sus herramientas para la interacción con el chat.

Como primera prueba, se realizó el envío del mensaje “Hola” hacia el contacto “Mamá”. Como se observa en las Ilustraciones 4.26 y 4.27, el sistema minimiza la ventana del SAAC y maximiza el navegador de Chromium en la pestaña de WhatsApp Web. Una vez localizado el contacto y abierto su chat, se escribe automáticamente el mensaje en el cuadro de texto y se envía la frase utilizando el botón de enviar, localizado a la derecha del cuadro.

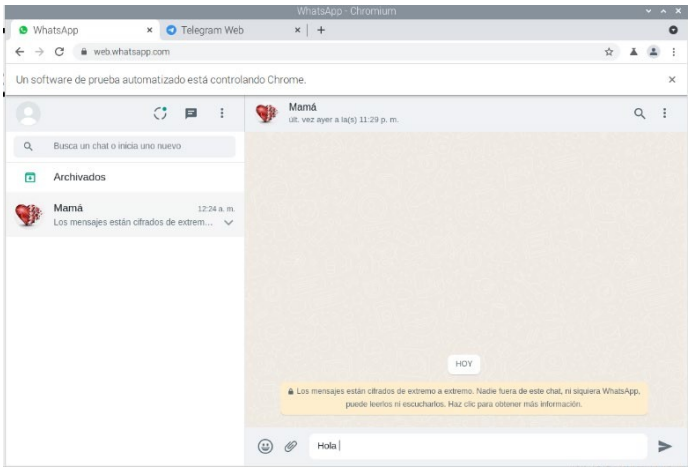


Ilustración 4.26 Escritura del mensaje “Hola” en el cuadro de texto de WhatsApp Web.

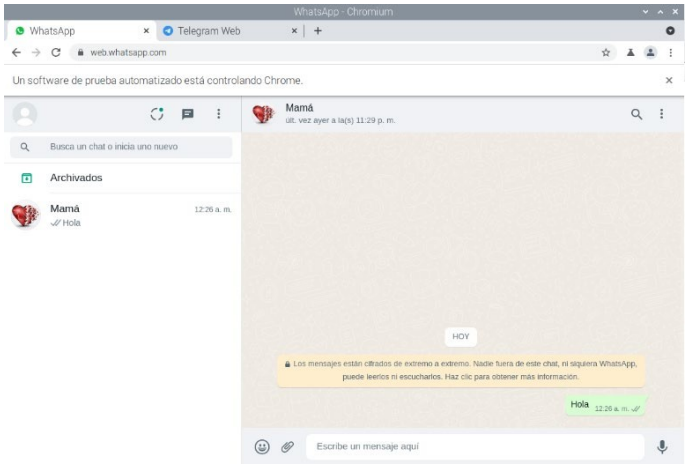


Ilustración 4.27 Envío de mensaje "Hola" en WhatsApp Web.

Como siguiente prueba, se empleó la interfaz social para corroborar la funcionalidad de las herramientas para el chat de WhatsApp Web. En la Ilustración 4.28 se observa que, así como en el chat SMS, se enviaron 20 mensajes enumerados desde el 1 hasta el 20.

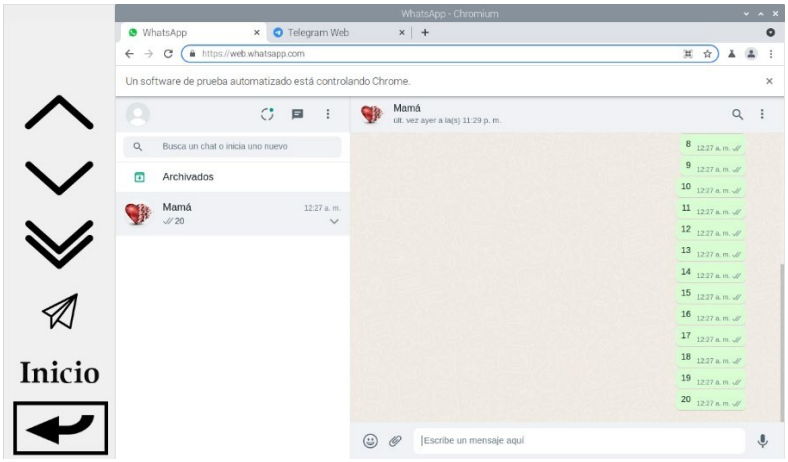


Ilustración 4.28 Historial de mensajes de WhatsApp Web.

En primer lugar, se eligió el botón de desplazamiento hacia arriba, el cual realizó su función, deslizando el historial de mensajes, como se observa en la Ilustración 4.29.

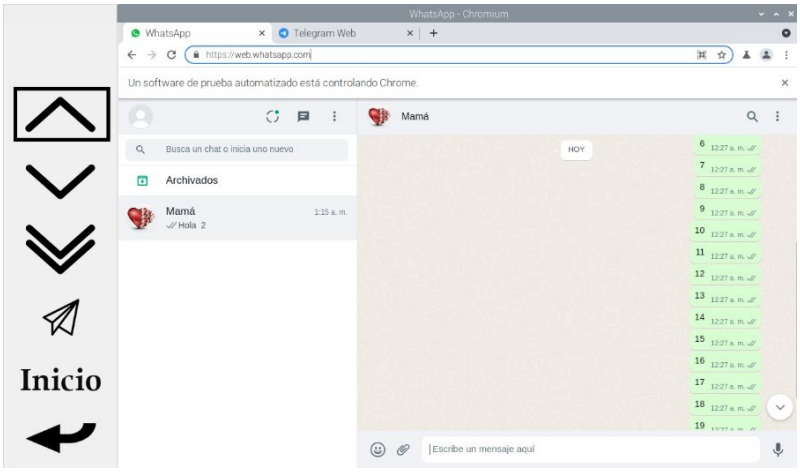


Ilustración 4.29 Botón de desplazamiento hacia arriba.

Después, se utilizó el botón de deslizamiento hacia abajo para regresar al estado inicial del chat, como se ve en la Ilustración 4.30.

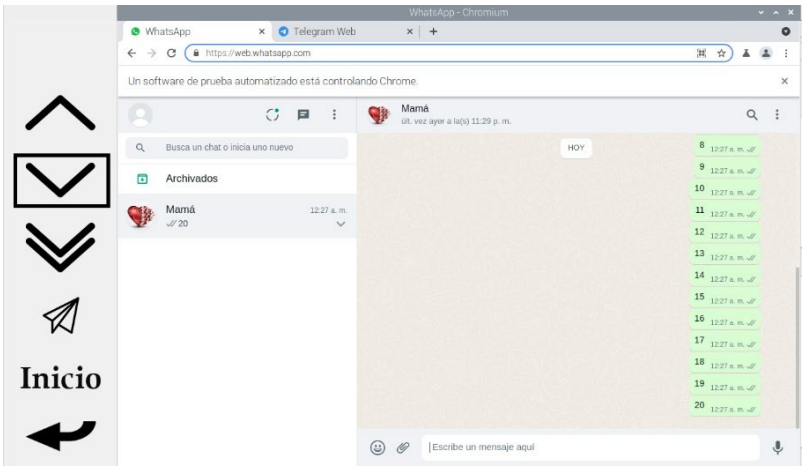


Ilustración 4.30 Botón de desplazamiento hacia abajo.

Con respecto al botón del final de la conversación, se probó su funcionalidad volviendo a subir en el historial de mensajes y eligiendo la herramienta mencionada, por lo que la interfaz de WhatsApp regreso hasta el final de la conversación (ver Ilustración 4.31 y 4,32).

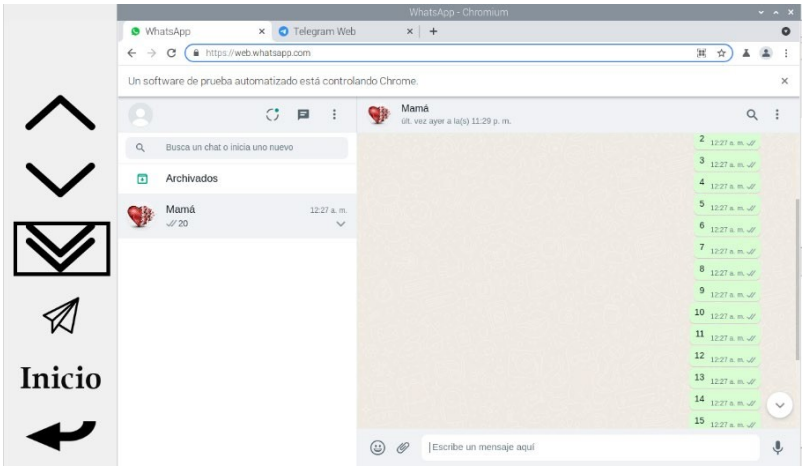


Ilustración 4.31 Antes de presionar el botón de final de la conversación.

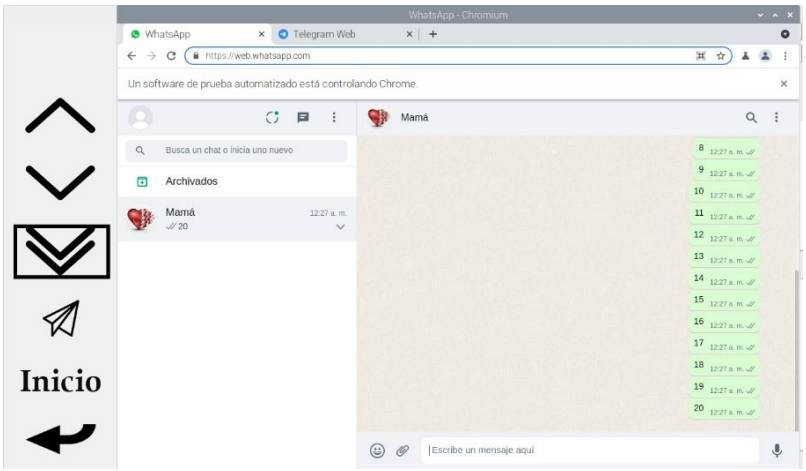


Ilustración 4.32 Antes de presionar el botón de final de la conversación.

La última prueba realizada fue la que implica la opción de “Responder”. Para ello se redactó el mensaje “Hola 2” desde la interfaz alfabética. El resultado obtenido fue el que se observa en la Ilustración 4.33 y 4.34.

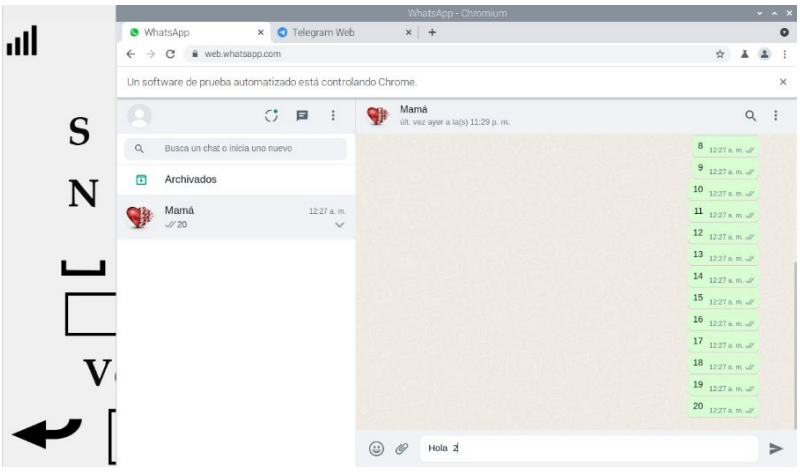


Ilustración 4.33 Escritura del mensaje “Hola 2” en el cuadro de texto de WhatsApp Web.

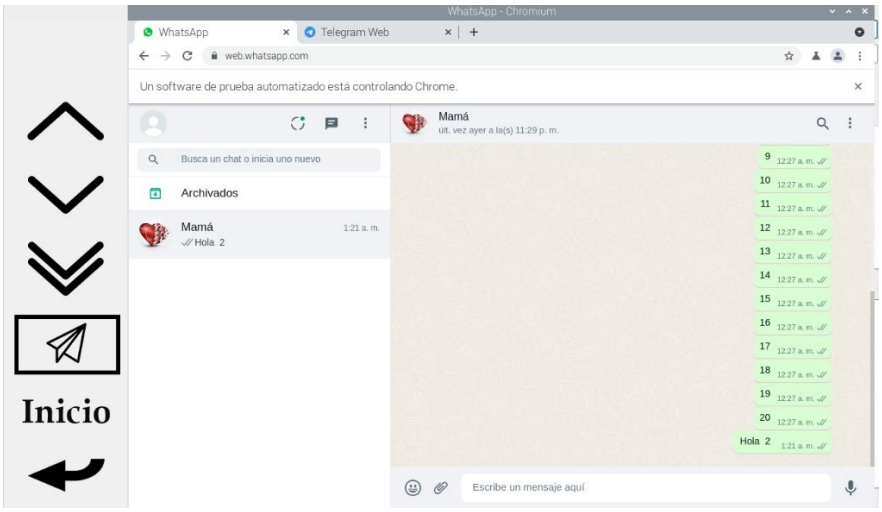


Ilustración 4.34 Envío de mensaje "Hola 2" en WhatsApp Web.

4.4 Pruebas de Telegram

En este apartado se describirán las pruebas realizadas para el envío de mensajes por Telegram por medio de la interfaz alfabética y pictográfica, así como la visualización de los historiales de mensajes en la interfaz de Telegram Web, junto con sus herramientas para la interacción con el chat.

Como primera prueba, se realizó el envío del mensaje "Hola" hacia el contacto "Mamá". Como se observa en las Ilustraciones 4.35 y 4.36, el sistema minimiza la ventana del SAAC y maximiza el navegador de Chromium en la pestaña de Telegram Web. Una vez localizado el contacto y abierto su chat, se escribe automáticamente el mensaje en el cuadro de texto y se envía la frase utilizando el botón de enviar, localizado a la derecha del cuadro.

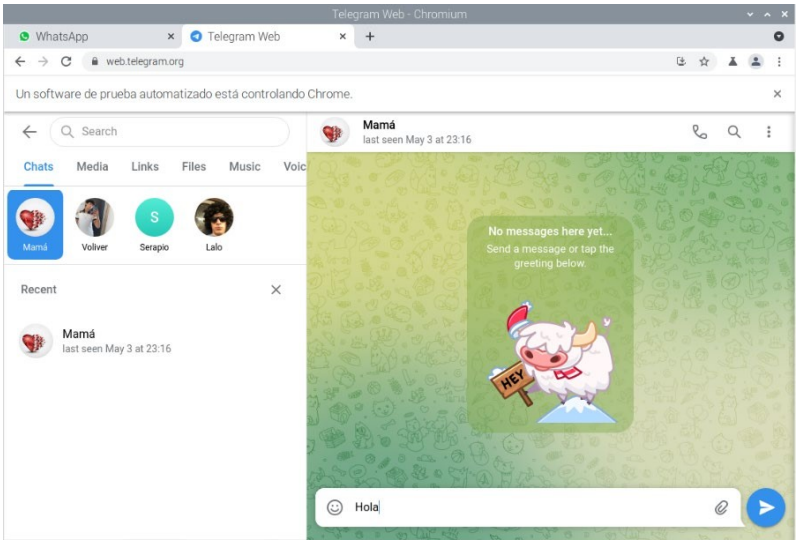


Ilustración 4.35 Escritura del mensaje "Hola" en el cuadro de texto de Telegram Web.

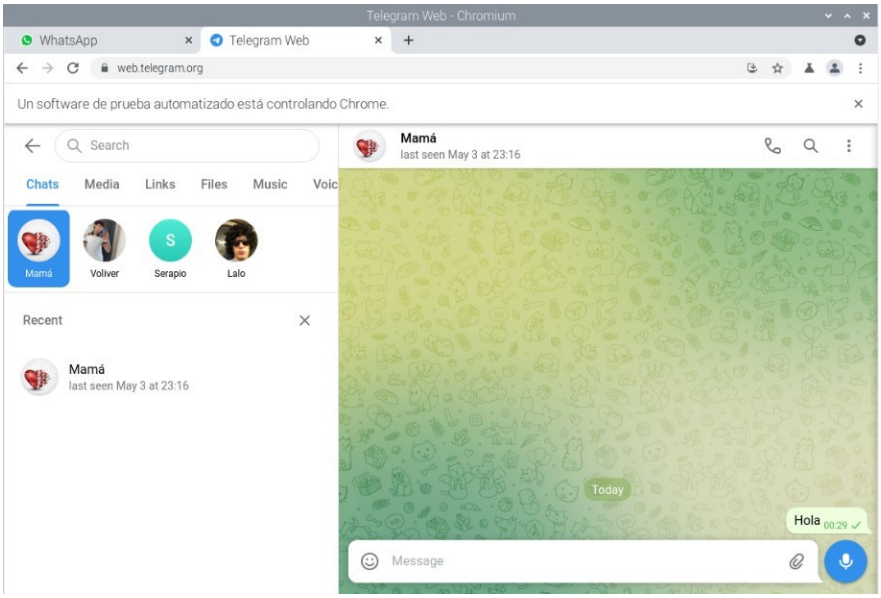


Ilustración 4.36 Envío de mensaje "Hola" en Telegram Web.

Como siguiente prueba, se empleó la interfaz social para corroborar la funcionalidad de las herramientas para el chat de Telegram Web y para ello se enviaron 20 mensajes enumerados desde el 1 hasta el 20. Se eligió el botón de desplazamiento hacia arriba, el cual realizó su función, deslizando el historial de mensajes, como se observa en la Ilustración 4.37.

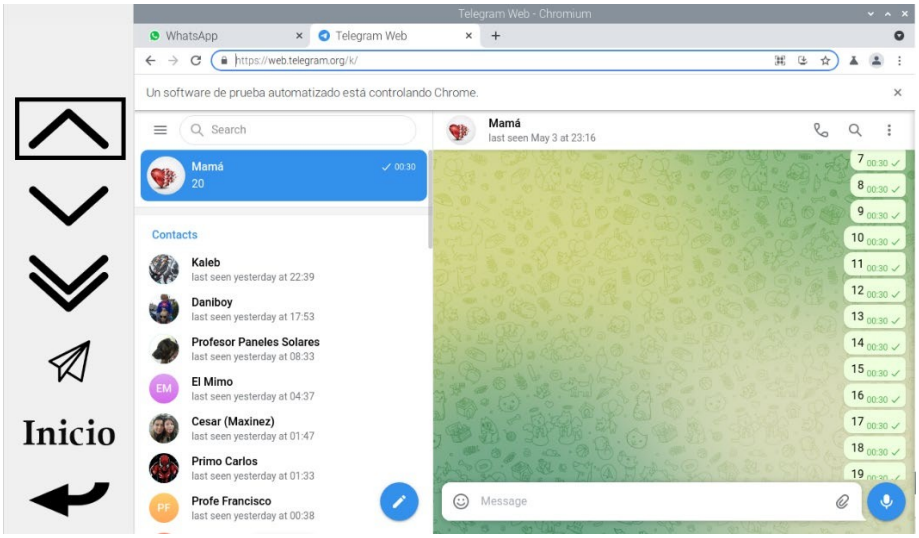


Ilustración 4.37 Botón de desplazamiento hacia arriba.

Después, se utilizó el botón de deslizamiento hacia abajo para regresar al estado inicial del chat, como se ve en la Ilustración 4.38.

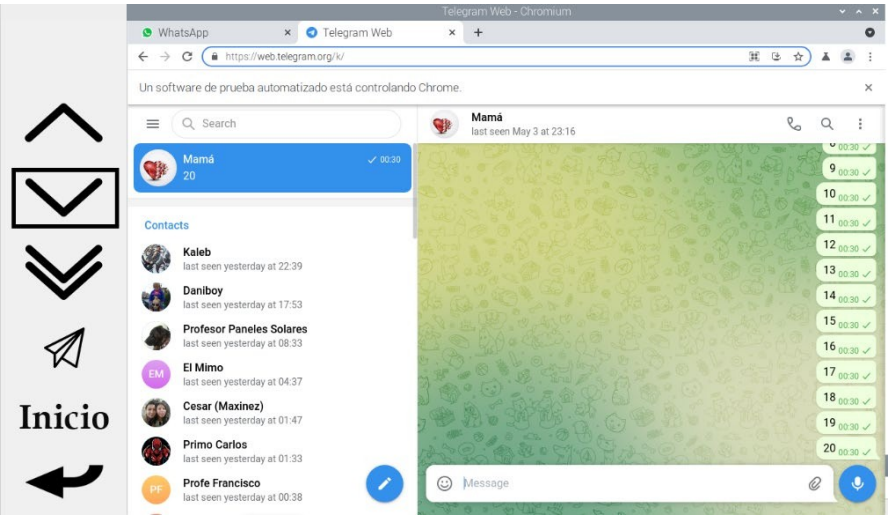


Ilustración 4.38 Botón de desplazamiento hacia abajo.

Con respecto al botón del final de la conversación, se probó su funcionalidad volviendo a subir en el historial de mensajes y eligiendo la herramienta mencionada, por lo que la interfaz de Telegram regreso hasta el final de la conversación (ver Ilustración 4.39 y 4,40).

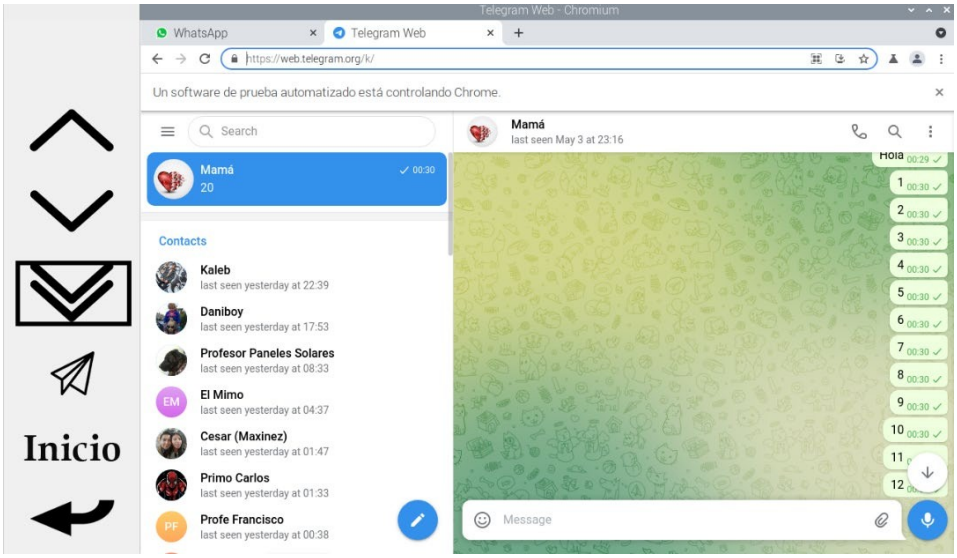


Ilustración 4.39 Antes de presionar el botón de final de la conversación.

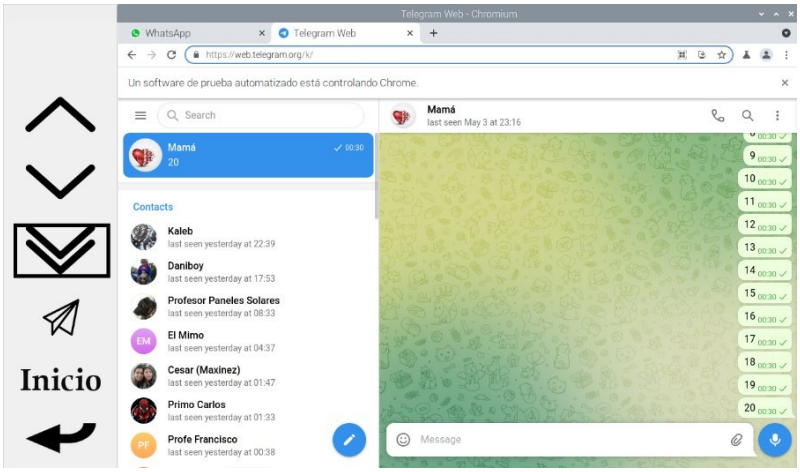


Ilustración 4.40 Antes de presionar el botón de final de la conversación.

La última prueba realizada fue la que implica la opción de “Responder”. Para ello se redactó el mensaje “Hola 2” desde la interfaz alfabética. El resultado obtenido fue el que se observa en la Ilustración 4.41 y 4.42.

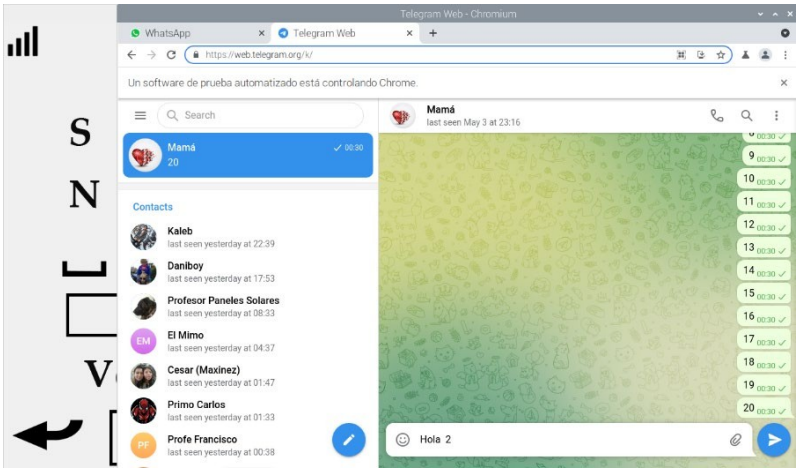


Ilustración 4.41 Escritura del mensaje “Hola 2” en el cuadro de texto de Telegram Web.

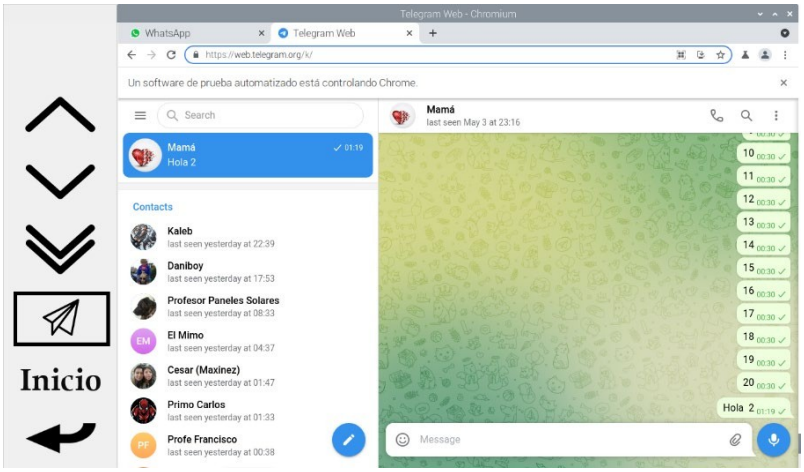


Ilustración 4.42 Envío de mensaje “Hola 2” en Telegram Web.

4.5 Pruebas de la configuración social

En este apartado se describirán las pruebas realizadas en el apartado de la configuración social, en el que se incluyen las funcionalidades de ingresar y/o borrar contactos, mensajes de prueba SMS, restauración del sistema social y reinicio del módulo encargado de los SMS.

La primera opción en la que se experimentó fue la de “Prueba”, que simplemente, al elegir el contacto, envía un mensaje predeterminado al elemento, mostrando la notificación de la Ilustración 4.43. El resultado se refleja en la Ilustración 4.44.

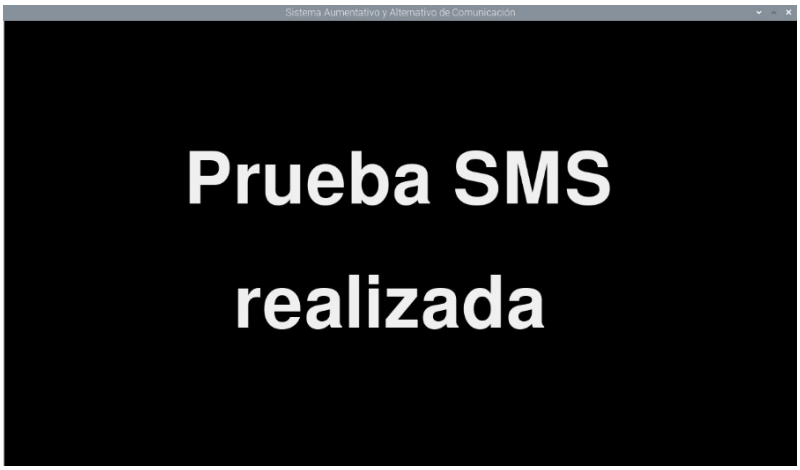


Ilustración 4.43 Notificación de prueba SMS realizada.

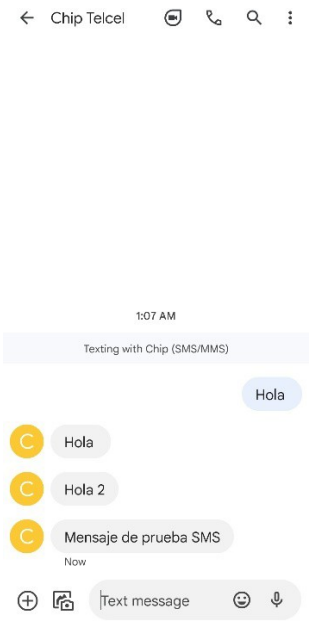


Ilustración 4.44 Recepción de la prueba en el teléfono celular.

La siguiente herramienta fue la de borrar contacto, por lo que se accedió a la opción “Ingresar contacto”, en donde se escogió el contacto “Mamá”. Una vez que se corroborará la eliminación de la selección (ver Ilustración 4.45), se ingresaron los datos del nuevo contacto en la interfaz de teclado y ratón.

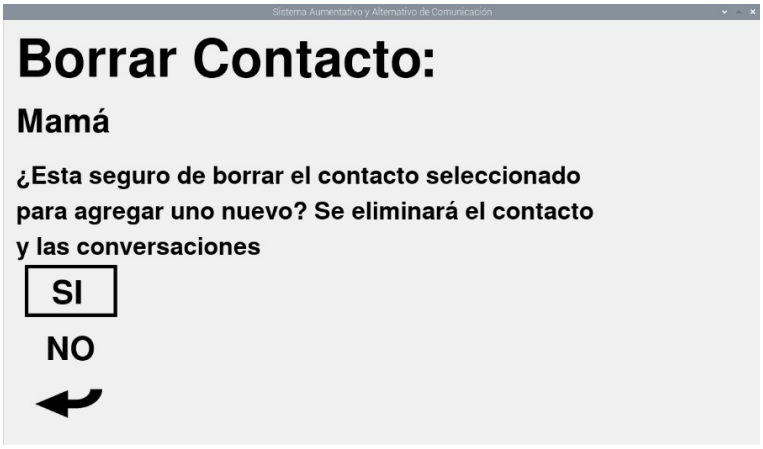


Ilustración 4.45 Borrado del contacto "Mamá".

Se introdujeron 8 dígitos en el teléfono para comprobar el mensaje de error de la Ilustración 4.46 y después se ingresaron los 10 números para guardar el contacto con el botón "Confirmar", como se ve en la Ilustración 4.47.

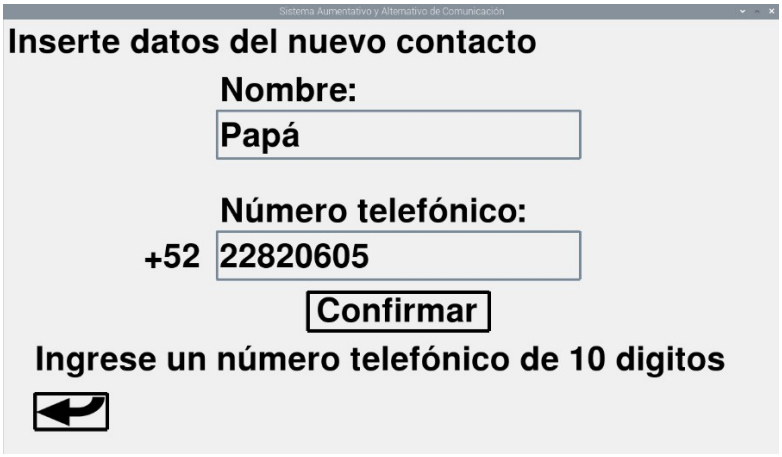


Ilustración 4.46 Notificación de error en el ingreso de datos del contacto desde la interfaz de teclado y ratón.

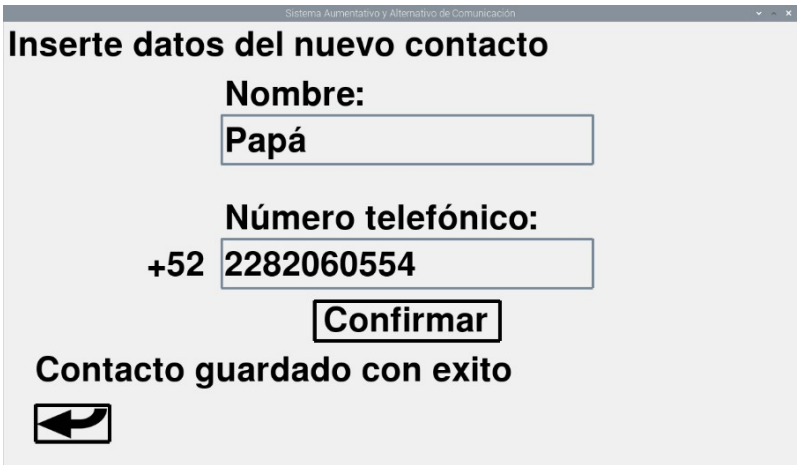


Ilustración 4.47 Notificación de guardo correcto del contacto desde la interfaz de teclado y ratón.

Una vez que se guarda el nuevo contacto, el sistema abrirá la opción “Ver contactos” en la Ilustración 4.48. Cabe resaltar que el contacto “Mamá” ha sido eliminado para ser reemplazado por el contacto “Papá”.

Ilustración 4.48 Interfaz de visualización de contactos (1 elemento).

Por el lado del interfaz del módulo HCI, se agregó el contacto denominado abuelo, ingresando el nombre en la Ilustración 4.49 y el número telefónico en la Ilustración 4.50. Cabe destacar que, si no se ingresan los 10 dígitos en la plantilla numérica, se mostrará el mensaje de error de la Ilustración 4.51.

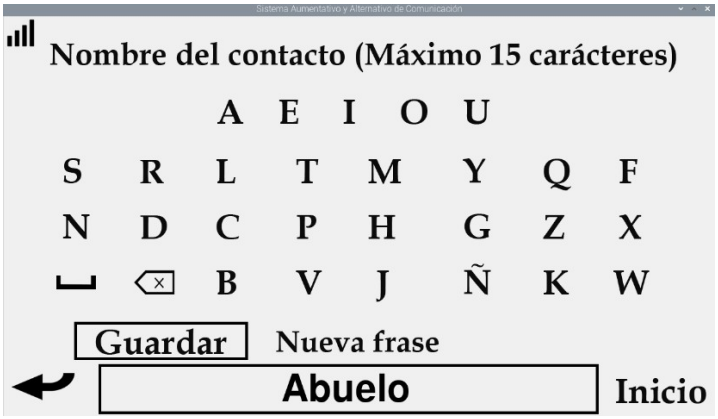


Ilustración 4.49 Nombre del contacto desde la interfaz del módulo HCI.

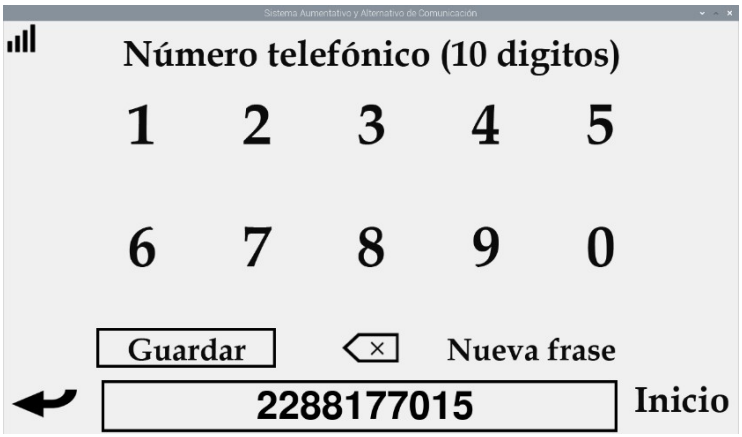


Ilustración 4.50 Número telefónico del contacto desde la interfaz del módulo HCI.

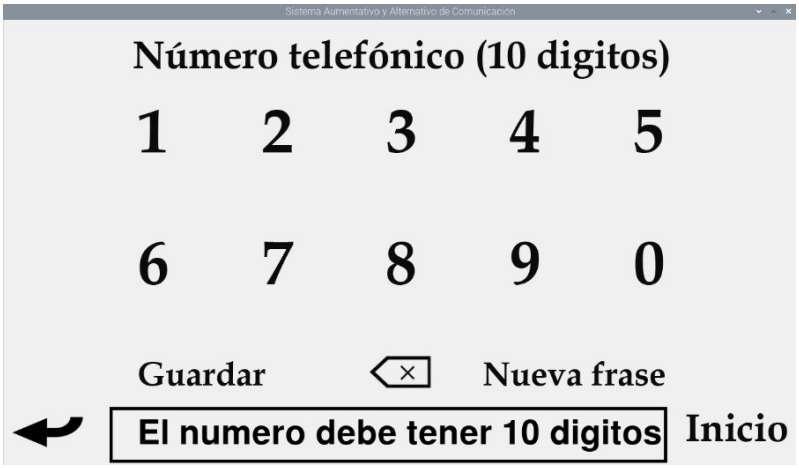


Ilustración 4.51 Notificación de error en el número telefónico desde la interfaz del módulo HCI.

Con el nuevo contacto guardado, el sistema abrirá la opción “Ver contactos” en la Ilustración 4.52, en donde se observa el nuevo elemento agregado a la agenda de contactos.

Ilustración 4.52 Interfaz de visualización de contactos (2 elementos).

La última prueba se realizó con la restauración del sistema social, cuyo objetivo es el de eliminar los elementos de la agenda y, por tanto, los historiales de chat de SMS. De esta forma, al aceptar la opción de restauración se mostrará la notificación dela Ilustración 4.53 y en el apartado de visualización de contactos se tendrá la lista vacía de la Ilustración 4.54.

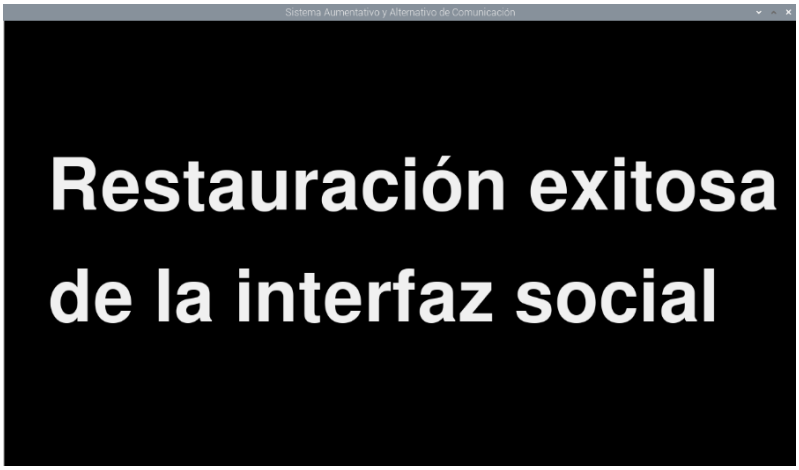


Ilustración 4.53 Notificación de la restauración de la interfaz social.

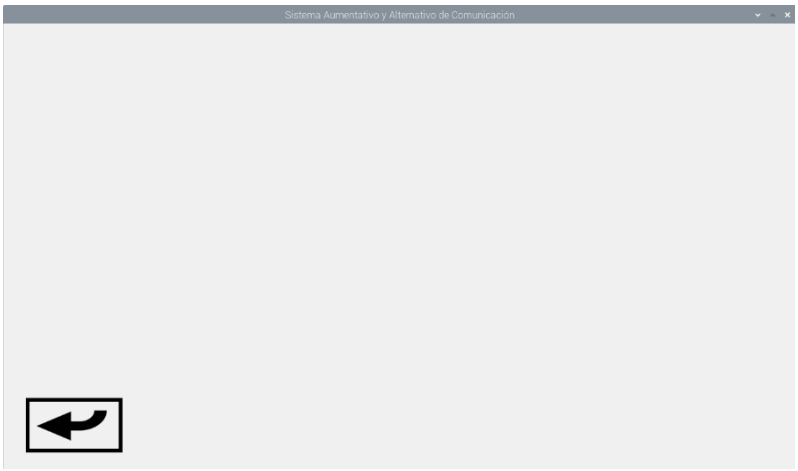


Ilustración 4.54 Visualización de contactos después de la restauración del sistema social.

Conclusiones

Se investigaron diversos dispositivos para el envío de mensajes SMS, y uno de los más populares es el SIM800L, es de bajo costo, pero la tecnología que usa (2G) en corto plazo quedará obsoleta, por lo que se decidió trabajar con el módulo GA6-B que cuesta solo un poco más que el modelo mencionado, pero soporta redes 4G por lo que su vigencia será de algunos años. Las pruebas realizadas fueron con el operador Telcel, se eligió debido a su amplia cobertura, y las pruebas fueron exitosas, cabe mencionar que el módulo puede trabajar también con las redes de otros operadores.

Se generó una lista en la cual se podían almacenar un máximo de 15 contactos, puesto que se necesitaba acceder a estos de manera rápida, por lo que una mayor cantidad de elementos supondría que para acceder al último se necesitaría pasar por los anteriores, haciendo tediosa la selección de contactos. A través de una sola opción se pueden eliminar elementos, así como agregarlos a la lista.

La recepción y envío de mensajes en las redes sociales abordadas en este documento fue posible gracias a su versión web, puesto que utilizando el navegador de Chromium, se logró manipular el entorno de WhatsApp Web y de Telegram Web. Con la ayuda del módulo de Selenium en Python y las funciones de JavaScript, se logró redactar mensajes en el cuadro de texto y enviarlos al contacto correspondiente.

Para la consulta del historial de mensajes en el apartado de SMS, se utilizó un lugar en la memoria en el que se almacenaban tanto los mensajes que enviaba el usuario, así como los recibidos por parte de los contactos, manteniendo un orden cronológico. En el caso de las redes sociales, se utilizó el propio entorno de WhatsApp Web y Telegram Web para visualizar el histórico de mensajería, ya que no era necesario almacenar la información si esta podía consultarse desde la aplicación web.

Con la función de selección de contactos, vista en el capítulo 3, se consiguió distribuir de manera sencilla los contactos almacenados, así como su elección para el envío de mensajes y la visualización de los historiales de chat de WhatsApp, Telegram o SMS.

Las plantillas que se crearon y modificaron cumplen no solo con la función que les fue asignada, sino que se integran de manera uniforme con las plantillas originales del SAAC, esto provoca que el sistema se mantenga en armonía, evitando la confusión del usuario ante interfaces complicadas.

Finalmente se concluye que al SAAC existente se le proporcionó la capacidad de enviar y recibir mensajes por medio de la red GSM y de las redes sociales de WhatsApp y Telegram, además de incorporar características adicionales como lo son el manejo de una lista de contactos, visualización de chats, restauración del sistema, entre otros.

Trabajo a futuro

Las características añadidas al SAAC le proporcionan una capacidad de comunicación básica, lo cual se puede mejorar en un futuro, agregando características como lo son el envío de emojis, stickers u otras fuentes que complementen la comunicación del usuario del sistema con su entorno. La visualización de videos o imágenes se puede realizar mientras que el navegador este configurado para descargar dichos elementos de manera automática. También se espera que más adelante, el usuario pueda acceder a carpetas dentro de la computadora para así seleccionar y enviar imágenes y videos anteriormente guardados.

La agenda de contactos de mantuvo con un límite de 15 elementos con el objetivo de facilitar el acceso hacia cada uno de ellos, no obstante, en un futuro se podría incrementar el almacenamiento, siempre y cuando no se afecte la accesibilidad de los contactos.

La detección de mensajes nuevos de WhatsApp y Telegram fue una característica que no se pudo completar puesto que el sistema de notificaciones del navegador utilizado, utiliza enlaces externos, las cuales no pueden ser consultados desde el código fuente de la página. Se espera que más adelante se desarrolle una aplicación que detecte la entrada de nuevos mensajes, así como lo está en la mensajería SMS.

El SAAC debe ser continuamente revisado, a causa de que las características de las páginas de WhatsApp Web y Telegram Web, son actualizadas de manera seguida por lo que el sistema de mensajería puede terminar inservible si no se va actualizando junto con las redes sociales. En cambio, el apartado de GSM se mantendrá en funcionamiento hasta que la tecnología en redes 4G sea obsoleta.

Referencias

[1] G. Belli Gómez, *Desarrollo de un sistema computacional para la comunicación verbal, que brinda apoyo a personas mudas con escasa movilidad mediante una interfaz visual y síntesis de voz,* Ciudad de México: Tesis UNAM, 2018.

[2] O. A. Delgadillo Martínez, *Sistema computacional aumentativo y alternativo de comunicación como herramienta de apoyo a personas con discapacidad motora y del habla,* Ecatepec de Morelos Estado de México: Tesis UAEM, 2021.

[3] G. C. Vanderheiden, «A Journey through early augmentative communication and computer access,» *Rehabilitation Research and Development,* vol. 39, nº 6, pp. 39-53, 2002.

[4] L. U. e. Internet, «UNIR,» 25 Mayo 2020. [En línea]. Available: https://www.unir.net/educacion/revista/los-sistemas-aumentativos-y-alternativos-de-comunicacion-saac/#:~:text=Quienes%20van%20a%20utilizar%20los,de%20forma%20puntual%20o%20permanente.. [Último acceso: 13 Abril 2022].

[5] P. Montero González, «Sistemas alternativos y aumentativos de comunicación (SAAC) y accesibilidad,» *Puertas a la lectura,* nº Extraordinario 4, pp. 129-136, 2003.

[6] U. I. d. Valencia, «Universidadviu,» 10 Octubre 2018. [En línea]. Available: https://www.universidadviu.com/es/actualidad/nuestros-expertos/que-es-gsm-y-como-funciona. [Último acceso: 17 Febrero 2022].

[7] W. MX, «WhistleOut,» 15 Enero 2022. [En línea]. Available: https://www.whistleout.com.mx/CellPhones/Guides/la-mejor-compania-de-celular-mexico. [Último acceso: 13 Abril 2022].

[8] R. PI, «Raspberry PI,» 9 Febrero 2022. [En línea]. Available: https://raspberrypi.cl/que-es-raspberry/. [Último acceso: 9 Febrero 2022].

[9] R. Pi, «Raspberry Pi,» 12 Febrero 2022. [En línea]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/. [Último acceso: 12 Febrero 2022].

[10] M. +, «Manuales +,» 8 Mayo 2021. [En línea]. Available: https://manuals.plus/es/raspberry-pi/raspberry-pi-4-model-b-manual#axzz7Syr0tIGj. [Último acceso: 4 Marzo 2022].

[11] Eneka, «Eneka,» 15 Abril 2022. [En línea]. Available: https://www.eneka.com.uy/robotica/modulos-comunicacion/m%C3%B3dulo-gsm-gprs-sim900-7477-detail.html. [Último acceso: 15 Abril 2022].

[12] Ai ThinkerTechnology Co.Ltd, *A6/A7/A6C User Manual,* Ai ThinkerTechnology Co.Ltd, 2016.

[13] R. González Duque, Python para todos, Creative Commons.

[14] J. D. Gauchat, El gran libro de HTML5, CSS3 y Javascript, Barcelona: Marcombo, 2012.

[15] Instituto Federal de Telecomunicaciones, «ift,» 10 febrero 2022. [En línea]. Available: http://comparador.ift.org.mx/. [Último acceso: 10 febrero 2022].

Fuentes de imágenes

Ilustración 1,2 y 3 [1].

Ilustración 3.3: https://www.b4x.com/android/forum/threads/guuooo-tech-iot-ga6-b-any-info-welcome-gsm-gprs-module.110349/ Recuperado el 16 el febrero del 2022.

Ilustración 3.4: https://www.programoergosum.es/tutoriales/introduccion-a-pines-gpio-en-raspbian/ Recuperado el 17 de febrero del 2022.

Ilustración 3.9 (Icono enviar): https://www.flaticon.es/icono-gratis/enviar_2983788?related_id=2983788&origin=search Recuperado el 4 de diciembre del 2022.

Ilustración 3.13 (Icono SMS): https://es.vexels.com/png-svg/vista-previa/135150/icono-de-sombra-de-mensajeria Recuperado el 24 de noviembre del 2022.

Ilustración 3.13 (Icono WhatsApp): https://www.stickpng.com/es/img/iconos-logotipos-emojis/companias-technologicas/logo-whatsapp Recuperado el 17 de noviembre del 2022.

Ilustración 3.13 (Icono Telegram): https://www.stickpng.com/es/img/iconos-logotipos-emojis/companias-technologicas/logo-telegram Recuperado el 28 de noviembre del 2022.

Ilustración 3.16 (Icono borrar): https://www.flaticon.es/icono-premium/compartimiento_3494533?term=borrar&page=1&position=7&page=1&position=7&related_id=3494533&origin=search Recuperado el 3 de diciembre del 2022.

Ilustración 3.18 (Icono buena señal): https://www.flaticon.com/premium-icon/signal-status_4234636?related_id=4234636 Recuperado el 13 de diciembre del 2022.

Ilustración 3.19 (Icono botón arriba): https://www.flaticon.es/icono-premium/punta-de-flecha-hacia-arriba_3838683?term=arriba&page=1&position=6&page=1&position=6&related_id=3838683&origin=search Recuperado el 14 de noviembre del 2022.

Ilustración 3.19 (Icono botón abajo): https://www.flaticon.es/icono-gratis/flecha-hacia-abajo-para-navegar_32195?term=abajo&page=1&position=1&page=1&position=1&related_id=32195&origin=search Recuperado el 14 de noviembre del 2022.

Ilustración 3.19 (Icono botón final): https://www.flaticon.es/icono-premium/abajo_3838648?related_id=3838648 Recuperado el 14 de noviembre del 2022.

Ilustración 3.20 (configuración social): https://icon-icons.com/es/icono/mensajes/40676 Recuperado el 5 de diciembre del 2022.

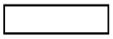
Ilustración 3.21 (Icono reiniciar módulo): https://www.freepik.es/iconos-gratis/boton-encendido_14550727.htm Recuperado el 26 de noviembre del 2022.

Ilustración 4.6 (Icono mala señal): https://www.flaticon.com/premium-icon/no-signal_4234712?related_id=4234712&origin=search Recuperado el 9 de diciembre del 2022.

Ilustración 4.8 (Notificación SMS nuevo): https://www.flaticon.es/icono-premium/sms_1865026?related_id=1865026&origin=search Recuperado el 3 de diciembre del 2022.

Anexo

A: Cursores añadidos al SAAC

 Cursor – Botón de enviar (interfaz alfabética y numérica)	 Cursor – Botón de guardar
 Cursor – Selección de contacto	 Cursor – Selección de fila de contactos
 Cursor – Botón de enviar (repetir frase)	 Cursor - selección de interfaz

B: Funciones agregadas a la programación del SAAC

```
1    import serial
2    from selenium import webdriver
3    from selenium.webdriver.common.keys import Keys
4    from selenium.webdriver.chrome.options import Options
5    from selenium.webdriver.support.ui import WebDriverWait
6    from selenium.webdriver.support import expected_conditions as EC
7    from selenium.webdriver.common.by import By
8    from selenium.common.exceptions import TimeoutException,
NoSuchElementException, StaleElementReferenceException,
```

```
ElementNotInteractableException
9
10  options = webdriver.ChromeOptions()
11  options.add_argument("user-data-dir=/home/pi/.config/chromium/SAAC")
12  driver = webdriver.Chrome(executable_path='/usr/bin/chromedriver',
options=options)
13
14  A6 = serial.Serial("/dev/serial0", 115200, timeout=1)
15  A6.close()
16  GPIO.setmode(GPIO.BCM)
17  GPIO.setwarnings(False)
18  GPIO.setup(21, GPIO.IN,GPIO.PUD_DOWN)
19  GPIO.setup(20, GPIO.OUT)
20
21  ######################################################################
22
23    def GuardarContactoBD(self):
24          global NombreContacto
25          global NumeroContacto
26          sql1="INSERT INTO listacontactos (numero,nombre)"
27          sql2="VALUES ('" + NumeroContacto + "','" + NombreContacto +
"');"
28          sql=sql1+sql2
29          try:
30              self.cursor.execute(sql)
31              self.connection.commit()
32          except Exception as e:
33              raise
34
35      def RestaurarSocialBD(self):
36          global Lista_Contactos
37          global Historial_Mensajes
38          global Mensajes_Leidos
39
40          for NumTel in Lista_Contactos.values():
41              sql = "DELETE FROM `listacontactos` WHERE
`listacontactos`.`Numero` = '" + NumTel + "';"
42              try:
43                  self.cursor.execute(sql)
44                  self.connection.commit()
45              except Exception as e:
46                  raise
47
48          Historial_Mensajes.clear()
49          Lista_Contactos.clear()
50          Mensajes_Leidos.clear()
51
52      def EliminarContactoBD(self,EliminarContacto):
53          global Lista_Contactos
54          global Historial_Mensajes
55          EliminarTelefono = Lista_Contactos[EliminarContacto]
56          sql1="DELETE FROM listacontactos WHERE "
57          sql2="Numero = '" + EliminarTelefono + "' AND Nombre = '" +
EliminarContacto + "';"
58          sql=sql1+sql2
59          Lista_Contactos.pop(EliminarContacto)
60          Historial_Mensajes.pop(EliminarContacto)
```

```
61          Mensajes_Leidos.pop(EliminarContacto)
62          try:
63              self.cursor.execute(sql)
64              self.connection.commit()
65          except Exception as e:
66              raise
67
68      def ConsultaListaContactos(self):
69          global Lista_Contactos
70          global Historial_Mensajes
71          global Mensajes_Leidos
72          database = DataBase()
73          sql="SELECT * FROM listacontactos"
74          try:
75              self.cursor.execute(sql)
76              TuplaContactos = self.cursor.fetchall()
77              i = len(TuplaContactos)
78              j = 0
79              while j < i:
80                  Telefono = TuplaContactos[j][0]
81                  Contacto = TuplaContactos[j][1]
82                  Lista_Contactos[Contacto] = Telefono
83                  Historial_Mensajes[Contacto] = []
84                  Mensajes_Leidos[Contacto] = True
85                  j = j + 1
86          except Exception as e:
87              raise
88
89      def GuardarMensajeBD(self, Numero, Mensaje):
90          sql1="INSERT INTO historialmensajes (numero,mensaje)"
91          sql2="VALUES ('" + Numero + "','" + Mensaje + "');"
92          sql=sql1+sql2
93          try:
94              self.cursor.execute(sql)
95              self.connection.commit()
96          except Exception as e:
97              raise
98
99      def ConsultaHistorialMensajes(self):
100         global Lista_Contactos
101         global Historial_Mensajes
102         sql="SELECT * FROM historialmensajes"
103         try:
104             self.cursor.execute(sql)
105             TuplaContactos = self.cursor.fetchall()
106             i = len(TuplaContactos)
107             j = 0
108             while j < i:
109                 ID = TuplaContactos[j][0]
110                 Telefono = TuplaContactos[j][1]
111                 Mensaje = TuplaContactos[j][2]
112                 for key in Lista_Contactos:
113                     if Telefono == Lista_Contactos[key]:
114                         Contacto = key
115                 Historial_Mensajes[Contacto].append(Mensaje)
116                 j = j + 1
117         except Exception as e:
```

```
118             raise
119
120     def EliminarUltimoMensaje(self, Contacto):
121         global Historial_Mensajes
122         global Lista_Contactos
123         Numero = Lista_Contactos[Contacto]
124         TotalMensaje = len(Historial_Mensajes[Contacto])
125         Mensaje = Historial_Mensajes[Contacto][TotalMensaje-1]
126         sql1 = "SELECT * FROM `historialmensajes` WHERE (Numero = "
127         sql2 = "'" + Numero + "' AND Mensaje = '" + Mensaje + "');"
128         sql = sql1 + sql2
129         try:
130             self.cursor.execute(sql)
131             TuplaContactos = self.cursor.fetchall()
132             i = len(TuplaContactos)
133             ID = TuplaContactos[i-1][0]
134         except Exception as e:
135             raise
136         sql3 = "DELETE FROM `historialmensajes` WHERE ID = " +
str(ID) + ";"
137         try:
138             self.cursor.execute(sql3)
139             self.connection.commit()
140         except Exception as e:
141             raise
142         Historial_Mensajes[Contacto].pop()
143
144     def EliminarHistorialMensajes(self, Contacto):
145         global Historial_Mensajes
146         global Lista_Contactos
147         Numero = Lista_Contactos[Contacto]
148         sql1 = "DELETE FROM `historialmensajes` WHERE Numero = "
149         sql2 = "'" + Numero + "';"
150         sql = sql1 + sql2
151         Historial_Mensajes[Contacto].clear()
152         try:
153             self.cursor.execute(sql)
154             self.connection.commit()
155         except Exception as e:
156             raise
157
158 ######################################################################
159
160
AbecedarioContacto=pygame.image.load(Ubicacion+"ABCContacto.png").convert
_alpha()
161
ConfSocial=pygame.image.load(Ubicacion+"ConfSocial.png").convert_alpha()
162
CursorABCEnviar=pygame.image.load(Ubicacion+"CursorABCEnviar.png").conver
t_alpha()
163
CursorColumnaContactos=pygame.image.load(Ubicacion+"CursorColumnaContacto
s.png").convert_alpha()
164
CursorFilaContactos=pygame.image.load(Ubicacion+"CursorFilaContactos.png"
).convert_alpha()
```

```
165
CursorNumGuardar=pygame.image.load(Ubicacion+"CursorNumGuardar.png").conv
ert_alpha()
166
CursorRepEnviar=pygame.image.load(Ubicacion+"CursorRepEnviar.png").conver
t_alpha()
167
NumerosContacto=pygame.image.load(Ubicacion+"NumerosContacto.png").conver
t_alpha()
168
OpcionesSocial=pygame.image.load(Ubicacion+"OpcionesSocial.png").convert_
alpha()
169
CursorSelInt=pygame.image.load(Ubicacion+"CursorSelInt.png").convert_alph
a()
170
SimboloEnviar=pygame.image.load(Ubicacion+"SimboloEnviar.png").convert_al
pha()
171
ChatSMS=pygame.image.load(Ubicacion+"ChatSMS.png").convert_alpha()
172
ChatSocial=pygame.image.load(Ubicacion+"ChatSocial.png").convert_alpha()
173
BuenaSeñal=pygame.image.load(Ubicacion+"BuenaSeñal.png").convert_alpha()
174
MalaSeñal=pygame.image.load(Ubicacion+"MalaSeñal.png").convert_alpha()
175
NuevoSMS=pygame.image.load(Ubicacion+"NuevoSMS.png").convert_alpha()
176
177 ######################################################################
178
179     def SesionEnWhatsApp():
180         try:
181             chat_pane = WebDriverWait(driver, 20).until(
182                 EC.presence_of_element_located((By.ID, 'pane-side')))
183             return True
184         except TimeoutException:
185             return False
186
187     def SesionEnTelegram():
188         try:
189             chat_pane = WebDriverWait(driver, 20).until(
190                 EC.presence_of_element_located((By.ID, 'column-
left')))
191             return True
192         except TimeoutException:
193             return False
194
195     def CargarWhatsAppTelegram():
196         driver.set_window_size(1280, 720)
197         driver.get('https://web.whatsapp.com/')
198         WhatsAppLog = SesionEnWhatsApp()
199         if WhatsAppLog == False:
200             print("Sesión no iniciada en WhatsApp")
201
driver.execute_script('window.open("https://web.telegram.org/k/");')
202         driver.implicitly_wait(20)
```

```
203         TelegramLog = SesionEnTelegram()
204         if TelegramLog == False:
205             print("Sesión no iniciada en Telegram")
206         driver.minimize_window()
207 
208     def ComunicacionModulo():
209         global ModuloA6
210         ModuloA6 = False
211         A6.open()
212         A6.write(b'AT\r')
213         for x in range(10):
214             respuesta = A6.readline()
215             AT_de = respuesta.decode("iso-8859-1")
216             if AT_de == 'OK\r\n':
217                 ModuloA6 = True
218                 break
219         if ModuloA6 != True:
220             fuente = pygame.font.Font(None, 170)
221             mensaje = fuente.render("Módulo de mensajes", 1,
(Blanco))
222             mensaje2 = fuente.render("SMS desconectado",1, (Blanco))
223             screen.fill(Negro)
224             screen.blit(mensaje, (50, 150))
225             screen.blit(mensaje2, (50, 320))
226             pygame.display.flip()
227             time.sleep(2)
228         A6.close()
229 
230     def EstadoConexion():
231         if ModuloA6 == False:
232             EstadoRedGSM = "Sin conexión"
233         else:
234             A6.open()
235             mantener = True
236             A6.write(b'AT+CREG?\r')
237             while mantener:
238                 resultado = A6.readline()
239                 resultado_dec = resultado.decode("iso-8859-1")
240                 if resultado_dec == '+CREG: 1,1\r\n':
241                     EstadoRedGSM = "Conexión establecida"
242                     mantener = False
243                 if resultado_dec == '+CREG: 1,0\r\n' or resultado_dec
== '+CREG: 1,2\r\n' or resultado_dec == '+CREG: 1,3\r\n':
244                     EstadoRedGSM = "Sin conexión"
245                     mantener = False
246             A6.close()
247         return EstadoRedGSM
248 
249     def MensajeEntrante(R, MR):
250         global Lista_Contactos
251         global Historial_Mensajes
252         global NombreContacto
253         global NumeroContacto
254         Desconocido = True
255         for key in Lista_Contactos:
256             if R == Lista_Contactos[key]:
257                 R = key
```

```
258                     Desconocido = False
259             if not Desconocido:
260                 MR = R + ": " + MR
261                 if MR != Historial_Mensajes[R][-1]:
262                     Tel = Lista_Contactos[R]
263                     database = DataBase()
264                     database.GuardarMensajeBD(Tel, MR)
265                     database.CerrarBD()
266                     Historial_Mensajes[R].append(MR)
267                     Mensajes_Leidos[R] = False
268                 else:
269                     Desconocido = True
270             return Desconocido
271 
272         def RevisarMensajesNoLeidos():
273             if ModuloA6 == False:
274                 pass
275             else:
276                 MensajesNuevos = False
277                 A6.close()
278                 A6.open()
279                 Loop = True
280                 Desconocido = True
281                 A6.write(b'AT+CMGL="REC UNREAD"\r')
282                 while Loop:
283                     Lectura = A6.readline()
284                     Lectura_Dec = Lectura.decode("iso-8859-1")
285                     if Lectura_Dec.startswith('+CMGL:'):
286                         MensajesNuevos = True
287                         Remitente = "+52" + Lectura_Dec.split('"')[3]
288                         Lectura = A6.readline()
289                         Lectura_Dec = Lectura.decode("iso-8859-1")
290                         Lectura_Dec = Lectura_Dec.replace('\x00','')
291                         Lectura_Dec = Lectura_Dec.replace('}','ñ')
292                         Lectura_Dec = Lectura_Dec.replace(']','Ñ')
293                         Lectura_Dec = Lectura_Dec.replace('','é')
294                         MensajeRemitente = Lectura_Dec.replace('\r\n','')
295                         if Remitente == '+52Telcel':
296                             f = pygame.font.Font(None, 120)
297                             MensajeSaldo = MensajeRemitente.split('
con')[0]
298                             m1 = f.render(MensajeSaldo[:22], 1, (Blanco))
299                             m2 = f.render(MensajeSaldo[23:],1, (Blanco))
300                             screen.fill(Negro)
301                             screen.blit(m1, (50, 150))
302                             screen.blit(m2, (50, 320))
303                             pygame.display.flip()
304                             time.sleep(2)
305                         else:
306                             D = MensajeEntrante(Remitente,
MensajeRemitente)
307                             if D == False:
308                                 Desconocido = False
309                     elif Lectura_Dec == 'OK\r\n':
310                         A6.write(b'AT+CMGD=1,3\r')
311                         Loop = False
312                     elif Lectura_Dec.startswith("Call Ready") or
```

```
Lectura_Dec.startswith("SMS Ready") or
Lectura_Dec.startswith("AST_POWERON") or Lectura_Dec.startswith('+CMS
ERROR:') or Lectura_Dec.startswith('+CME ERROR:'):
313                     Loop = False
314                 else:
315                     continue
316             A6.close()
317             if MensajesNuevos:
318                 if not Desconocido:
319                     fuente = pygame.font.Font(None, 170)
320                     mensaje = fuente.render("Nuevos mensajes", 1,
(Blanco))
321                     mensaje2 = fuente.render("de SMS no leidos",1,
(Blanco))
322                     screen.fill(Negro)
323                     screen.blit(mensaje, (50, 150))
324                     screen.blit(mensaje2, (50, 320))
325                     pygame.display.flip()
326                     time.sleep(2)
327                 else:
328                     pass
329             else:
330                 pass
331
332 ###################################################################
333
334     def InformacionRestauracionSocial():
335         TituloFuente = pygame.font.Font(None, 130)
336         Negrita = pygame.font.Font(None, 72)
337         Fuente = pygame.font.Font(None, 72)
338         screen.fill(ColorDeFondo)
339         screen.blit(PictoRegreso, (0, 0))
340         screen.blit(CuadritoPeque, (0, 0))
341         TituloInformacion = TituloFuente.render("Restauración de la
interfaz", 1, (Negro))
342         TituloInformacionB = TituloFuente.render("social", 1,
(Negro))
343         PrimerParametro = Negrita.render("El sistema borra los
elementos de la lista de" ,1, (Negro))
344         PrimerParametroB = Negrita.render("contactos, así como los
mensajes," ,1, (Negro))
345         PrimerParametroC = Negrita.render("de manera permanentemente"
,1, (Negro))
346         screen.blit(TituloInformacion, (20, 15))
347         screen.blit(TituloInformacionB, (20, 150))
348         screen.blit(PrimerParametro, (20, 300))
349         screen.blit(PrimerParametroB, (20, 400))
350         screen.blit(PrimerParametroC, (20, 500))
351         pygame.display.flip()
352
353 ###################################################################
354
355     def CargarParametrosSocial():
356         global NumeroUsuario
357         global CompañiaUsuario
358         if ModuloA6 == False:
359             NumeroUsuario = "Sin datos"
```

```
360             CompañiaUsuario = "Sin datos"
361             pass
362         else:
363             A6.open()
364             A6.write(b'AT+COPS=?\r')
365             evaluar = True
366             while evaluar:
367                 resultado = A6.readline()
368                 AT_dec = resultado.decode("iso-8859-1")
369                 if AT_dec.startswith("+COPS:"):
370                     SinConexion = False
371                     evaluar = False
372                 elif AT_dec.startswith("Call Ready") or
AT_dec.startswith("SMS Ready") or AT_dec.startswith("AST_POWERON"):
373                     SinConexion = True
374                     evaluar = False
375             if SinConexion:
376                 A6.close()
377             else:
378                 AT_COPS = AT_dec.split('"')
379                 CompañiaUsuario = AT_COPS[1]
380                 time.sleep(1)
381                 A6.write(b'AT+CNUM\r')
382                 evaluar = True
383                 while evaluar:
384                     resultado = A6.readline()
385                     AT_dec = resultado.decode("iso-8859-1")
386                     if AT_dec.startswith("+CNUM:"):
387                         AT_CNUM = AT_dec.split('"')
388                         NumeroUsuario = AT_CNUM[3]
389                         evaluar = False
390                 A6.write(b'AT+CMGF=1\r')
391                 A6.write(b'AT+CSDH=1\r')
392                 A6.close()
393
394 ######################################################################
395
396     global formato
397     global Historial_Mensajes
398     global Lista_Contactos
399     global Mensajes_Leidos
400     global CuadriculaContactos
401     global NombreContacto
402     global NumeroContacto
403     global NumeroUsuario
404     global CompañiaUsuario
405     global ContactoElegido
406     global UltimaSeleccion
407     formato=""
408     CuadriculaContactos=""
409     Historial_Mensajes = {}
410     Lista_Contactos = {}
411     Mensajes_Leidos = {}
412     NombreContacto = ""
413     NumeroContacto = ""
414     NumeroUsuario = ""
415     CompañiaUsuario = ""
```

```
416     ContactoElegido = ""
417     UltimaSeleccion = ""
418     ModuloA6 = False
419 
420 ########################################################################
421 
422     def seleccion(plantilla,Cuadro,posx,posy,GSM):
423         global VelocidadCursor
424         global PosiciondeTitulo
425         global Mensajes_Leidos
426         screen.fill(ColorDeFondo)
427         screen.blit(plantilla, (0, 0))
428         if GSM == "Conexión establecida":
429             screen.blit(BuenaSeñal, (5,5))
430         else:
431             screen.blit(MalaSeñal, (5,5))
432         for key in Mensajes_Leidos:
433             if Mensajes_Leidos.get(key) == True:
434                 continue
435             else:
436                 screen.blit(NuevoSMS, (5,60))
437                 break
438         pygame.display.flip()
439         screen.blit(Cuadro, (posx, posy))
440         pygame.display.flip()
441 
442 ########################################################################
443 
444     def cambio(plantilla,posx,posy,text,GSM):
445         global frase
446         global TamLet
447         global PosYCuaTex
448         global PosXCuaTex
449         global Mensajes_Leidos
450         frase=text
451         ProcesamientoDeTextoPrincipal()
452         screen.fill(ColorDeFondo)
453         if GSM == "Conexión establecida":
454             screen.blit(BuenaSeñal, (5,5))
455         else:
456             screen.blit(MalaSeñal, (5,5))
457         for key in Mensajes_Leidos:
458             if Mensajes_Leidos.get(key) == True:
459                 continue
460             else:
461                 screen.blit(NuevoSMS, (5,60))
462                 break
463         screen.blit(plantilla, (0, 0))
464         pygame.draw.rect(screen,Negro,(200,595,915,100))
465         pygame.draw.rect(screen,Blanco,(205,600,905,90))
466         fuente = pygame.font.Font(None, TamLet)
467         mensaje = fuente.render(frase, 1, (0, 0, 0))
468         screen.blit(mensaje, (PosXCuaTex+50, PosYCuaTex+275))
469         pygame.display.flip()
470         screen.blit(Cuadrito, (posx, posy))
471         pygame.display.flip()
472 
```

```
473     def regreso(plantilla,posX,texto,GSM):
474         global Mensajes_Leidos
475         screen.fill(ColorDeFondo)
476         if GSM == "Conexión establecida":
477             screen.blit(BuenaSeñal, (5,5))
478         else:
479             screen.blit(MalaSeñal, (5,5))
480         for key in Mensajes_Leidos:
481             if Mensajes_Leidos.get(key) == True:
482                 continue
483             else:
484                 screen.blit(NuevoSMS, (5,60))
485                 break
486         screen.blit(plantilla, (0, 0))
487         screen.blit(CuadritoPeque, (0, 0))
488         pygame.draw.rect(screen,Negro,(200,595,915,100))
489         pygame.draw.rect(screen,Blanco,(205,600,905,90))
490         fuente = pygame.font.Font(None, 100)
491         mensaje = fuente.render(texto, 1, (0, 0, 0))
492         screen.blit(mensaje, (posX, 603))
493         pygame.display.flip()
494 
495     def siguiente(plantilla,GSM):
496         global Mensajes_Leidos
497         screen.fill(ColorDeFondo)
498         if GSM == "Conexión establecida":
499             screen.blit(BuenaSeñal, (5,5))
500         else:
501             screen.blit(MalaSeñal, (5,5))
502         for key in Mensajes_Leidos:
503             if Mensajes_Leidos.get(key) == True:
504                 continue
505             else:
506                 screen.blit(NuevoSMS, (5,60))
507                 break
508         screen.blit(plantilla, (0, 0))
509         screen.blit(CuadritoPeque, (1085,0))
510         pygame.draw.rect(screen,Negro,(200,595,915,100))
511         pygame.draw.rect(screen,Blanco,(205,600,905,90))
512         fuente = pygame.font.Font(None, 100)
513         mensaje = fuente.render("Siguiente", 1, (0, 0, 0))
514         screen.blit(mensaje, (530, 603))
515         pygame.display.flip()
516 
517 ######################################################################
518 
519     def previsualizacionNumerica(pre,Cuadrito,posx,posy,GSM):
520         global TamLet
521         global PosYCuaTex
522         global PosXCuaTex
523         global Mensajes_Leidos
524         if pre =="Borrar todo":
525            TamLet=100
526            PosYCuaTex=327
527            PosXCuaTex=450
528         elif pre =="Regresar":
529            TamLet=100
```

```
530             PosYCuaTex=327
531             PosXCuaTex=500
532         else:
533             ProcesamientoDeTextoPrincipal()
534         screen.fill(ColorDeFondo)
535         screen.blit(Numeros, (0, 0))
536         if GSM == "Conexión establecida":
537             screen.blit(BuenaSeñal, (5,5))
538         else:
539             screen.blit(MalaSeñal, (5,5))
540         for key in Mensajes_Leidos:
541             if Mensajes_Leidos.get(key) == True:
542                 continue
543             else:
544                 screen.blit(NuevoSMS, (5,60))
545                 break
546         pygame.draw.rect(screen,Negro,(170,625,900,90))
547         pygame.draw.rect(screen,Blanco,(175,630,890,80))
548         fuente = pygame.font.Font(None, TamLet)
549         pre=pre.capitalize()
550         mensaje = fuente.render(pre, 1, (0, 0, 0))
551         screen.blit(mensaje, (PosXCuaTex, PosYCuaTex+300))
552         screen.blit(Cuadrito, (posx, posy))
553         pygame.display.flip()
554
555     def previsualizacion(pre,Cuadrito,posx,posy,GSM):
556         global Prediccion1
557         global Prediccion2
558         global Prediccion3
559         global TamLet
560         global PosYCuaTex
561         global PosXCuaTex
562         global Mensajes_Leidos
563         screen.fill(ColorDeFondo)
564         screen.blit(Abecedario, (0, 0))
565         if GSM == "Conexión establecida":
566             screen.blit(BuenaSeñal, (5,5))
567         else:
568             screen.blit(MalaSeñal, (5,5))
569         for key in Mensajes_Leidos:
570             if Mensajes_Leidos.get(key) == True:
571                 continue
572             else:
573                 screen.blit(NuevoSMS, (5,60))
574                 break
575         ProcesamientoDeTextoPredictor(Prediccion1,243)
576         pygame.draw.rect(screen,Negro,(100,437,300,70))
577         pygame.draw.rect(screen,Blanco,(105,442,290,60))
578         fuente = pygame.font.Font(None, TamLet)
579         Prediccion1=Prediccion1.capitalize()
580         mensaje = fuente.render(Prediccion1, 1, (0, 0, 0))
581         screen.blit(mensaje, (PosXCuaTex, PosYCuaTex))
582         Prediccion1=Prediccion1
583         ProcesamientoDeTextoPredictor(Prediccion2,603)
584         pygame.draw.rect(screen,Negro,(460,437,300,70))
585         pygame.draw.rect(screen,Blanco,(465,442,290,60))
586         fuente = pygame.font.Font(None, TamLet)
```

```
587             Prediccion2=Prediccion2.capitalize()
588             mensaje = fuente.render(Prediccion2, 1, (0, 0, 0))
589             screen.blit(mensaje, (PosXCuaTex, PosYCuaTex))
590             Prediccion2=Prediccion2
591             ProcesamientoDeTextoPredictor(Prediccion3,1003)
592             pygame.draw.rect(screen,Negro,(860,437,300,70))
593             pygame.draw.rect(screen,Blanco,(865,442,290,60))
594             fuente = pygame.font.Font(None, TamLet)
595             Prediccion3=Prediccion3.capitalize()
596             mensaje = fuente.render(Prediccion3, 1, (0, 0, 0))
597             screen.blit(mensaje, (PosXCuaTex, PosYCuaTex))
598             Prediccion3=Prediccion3
599             if pre =="Borrar todo":
600               TamLet=100
601               PosYCuaTex=327
602               PosXCuaTex=450
603             elif pre =="Selección de interfaz":
604                TamLet=100
605                PosYCuaTex=327
606                PosXCuaTex=260
607             else:
608               ProcesamientoDeTextoPrincipal()
609             pygame.draw.rect(screen,Negro,(170,625,900,90))
610             pygame.draw.rect(screen,Blanco,(175,630,890,80))
611             fuente = pygame.font.Font(None, TamLet)
612             pre=pre.capitalize()
613             mensaje = fuente.render(pre, 1, (0, 0, 0))
614             screen.blit(mensaje, (PosXCuaTex, PosYCuaTex + 300))
615             screen.blit(Cuadrito, (posx, posy))
616             pygame.display.flip()
617
618         def previsualizacionNC(pre,Cuadrito,posx,posy,GSM):
619             global TamLet
620             global PosYCuaTex
621             global PosXCuaTex
622             global Mensajes_Leidos
623             screen.fill(ColorDeFondo)
624             screen.blit(AbecedarioContacto, (0, 0))
625             if GSM == "Conexión establecida":
626                 screen.blit(BuenaSeñal, (5,5))
627             else:
628                 screen.blit(MalaSeñal, (5,5))
629             if pre =="Borrar todo":
630                 TamLet=100
631                 PosYCuaTex=327
632                 PosXCuaTex=450
633             elif pre =="Selección de contactos":
634                 TamLet=100
635                 PosYCuaTex=327
636                 PosXCuaTex=240
637             else:
638                 ProcesamientoDeTextoPrincipal()
639             pygame.draw.rect(screen,Negro,(170,625,900,90))
640             pygame.draw.rect(screen,Blanco,(175,630,890,80))
641             fuente = pygame.font.Font(None, TamLet)
642             pre=pre.capitalize()
643             mensaje = fuente.render(pre, 1, (0, 0, 0))
```

```
644             screen.blit(mensaje, (PosXCuaTex, PosYCuaTex + 300))
645             screen.blit(Cuadrito, (posx, posy))
646             pygame.display.flip()
647 
648         def previsualizacionNumericaNC(pre,Cuadrito,posx,posy,GSM):
649             global TamLet
650             global PosYCuaTex
651             global PosXCuaTex
652             global Mensajes_Leidos
653             screen.fill(ColorDeFondo)
654             screen.blit(NumerosContacto, (0, 0))
655             if GSM == "Conexión establecida":
656                 screen.blit(BuenaSeñal, (5,5))
657             else:
658                 screen.blit(MalaSeñal, (5,5))
659             if pre =="Borrar todo":
660                TamLet=100
661                PosYCuaTex=327
662                PosXCuaTex=450
663             elif pre =="Nombre del contacto":
664                TamLet=100
665                PosYCuaTex=327
666                PosXCuaTex=275
667             else:
668                ProcesamientoDeTextoPrincipal()
669             pygame.draw.rect(screen,Negro,(170,625,900,90))
670             pygame.draw.rect(screen,Blanco,(175,630,890,80))
671             fuente = pygame.font.Font(None, TamLet)
672             mensaje = fuente.render(pre, 1, (0, 0, 0))
673             screen.blit(mensaje, (PosXCuaTex, PosYCuaTex+300))
674             screen.blit(Cuadrito, (posx, posy))
675             pygame.display.flip()
676 
677 ######################################################################
678 
679         def Selecciondeinterfaz():
680             CargarParametros()
681             while True:
682                 RevisarMensajesNoLeidos()
683                 EstadoRedGSM = EstadoConexion()
684                 global PlantillaPictografica
685                 global PlantillaNumerica
686                 global PlantillaAlfabetica
687                 global PlantillaSocial
688                 global PlantillaConfiguracion
689                 global frase
690 
seleccion(SeleccionInterfaz,CursorSelInt,0,5,EstadoRedGSM)
691                 x=0
692                 while x <= VelocidadCursor :
693                     if GPIO.input(21) == GPIO.HIGH:
694                        PlantillaNumerica="Desactivada"
695                        PlantillaPictografica="Desactivada"
696                        PlantillaSocial="Desactivada"
697                        PlantillaConfiguraction="Desactivada"
698                        PlantillaAlfabetica="Activada"
699                        sonidoSeleccion()
```

```
700                     frase=""
701                     interfazAlfabetica()
702                 time.sleep(0.01)
703                 x=x+10
704
seleccion(SeleccionInterfaz,CursorSelInt,427,5,EstadoRedGSM)
705             x=0
706             while x <= VelocidadCursor:
707                 if GPIO.input(21) == GPIO.HIGH:
708                     sonidoSeleccion()
709                     Idioma="-ves+"
710                     PlantillaNumerica="Desactivada"
711                     PlantillaAlfabetica="Desactivada"
712                     PlantillaSocial="Desactivada"
713                     PlantillaConfiguraction="Desactivada"
714                     PlantillaPictografica="Activada"
715                     SelecciondeinterfazPictografica()
716                 time.sleep(0.01)
717                 x=x+10
718
seleccion(SeleccionInterfaz,CursorSelInt,850,5,EstadoRedGSM)
719             x=0
720             while x <= VelocidadCursor:
721                 if GPIO.input(21) == GPIO.HIGH:
722                     sonidoSeleccion()
723                     PlantillaNumerica="Desactivada"
724                     PlantillaAlfabetica="Desactivada"
725                     PlantillaAlfabetica="Desactivada"
726                     PlantillaDomotica="Activada"
727                     frase=""
728                     Idioma="-ves+"
729                     interfazDomotica()
730                 time.sleep(0.01)
731                 x=x+10
732
seleccion(SeleccionInterfaz,CursorSelInt,215,303,EstadoRedGSM)
733             x=0
734             while x <= VelocidadCursor:
735                 if GPIO.input(21) == GPIO.HIGH:
736                     sonidoSeleccion()
737                     PlantillaNumerica="Desactivada"
738                     PlantillaAlfabetica="Desactivada"
739                     PlantillaPictografica="Desactivada"
740                     PlantillaConfiguraction="Desactivada"
741                     PlantillaSocial="Activada"
742                     frase=""
743                     Idioma="-ves+"
744                     interfazSocial()
745                 time.sleep(0.01)
746                 x=x+10
747
seleccion(SeleccionInterfaz,CursorSelInt,630,303,EstadoRedGSM)
748             x=0
749             while x <= VelocidadCursor:
750                 if GPIO.input(21) == GPIO.HIGH:
751                     sonidoSeleccion()
752                     PlantillaNumerica="Desactivada"
```

```
753                     PlantillaAlfabetica="Desactivada"
754                     PlantillaSocial="Desactivada"
755                     PlantillaPictografica="Desactivada"
756                     PlantillaConfiguraction="Activada"
757                     frase=""
758                     Idioma="-ves+"
759                     interfazConfiguracion()
760                 time.sleep(0.01)
761                 x=x+10
762
763 ########################################################################
764
765     def interfazConfiguracion():
766         while True:
767             RevisarMensajesNoLeidos()
768             EstadoRedGSM = EstadoConexion()
769             seleccion(Configuraciones,Cuadrito,0,0,EstadoRedGSM)
770             TiempoEspera(interfazConfiguracionSintetizador)
771             seleccion(Configuraciones,Cuadrito,428,0,EstadoRedGSM)
772             TiempoEspera(interfazConfiguracionColor)
773             seleccion(Configuraciones,Cuadrito,853,0,EstadoRedGSM)
774             TiempoEspera(interfazConfiguracionCursor)
775             seleccion(Configuraciones,Cuadrito,0,290,EstadoRedGSM)
776             TiempoEspera(ConfiguracionRestaurar)
777             seleccion(Configuraciones,Cuadrito,428,290,EstadoRedGSM)
778             TiempoEspera(ConfiguracionDomotica)
779             seleccion(Configuraciones,Cuadrito,853,290,EstadoRedGSM)
780             TiempoEspera(interfazConfiguracionSocial)
781             seleccion(Configuraciones,CuadritoPeque,0,0,EstadoRedGSM)
782             TiempoEspera(Selecciondeinterfaz)
783
784 ########################################################################
785
786     def interfazConfiguracionSocial():
787         global CuadriculaContactos
788         CuadriculaContactos = ""
789         while True:
790             RevisarMensajesNoLeidos()
791             EstadoRedGSM = EstadoConexion()
792             seleccion(ConfSocial,Cuadrito,0,0,EstadoRedGSM)
793             x=0
794             while x <= VelocidadCursor:
795                 if GPIO.input(21) == GPIO.HIGH:
796                     sonidoSeleccion()
797                     CuadriculaContactos = "Agregar"
798                     SeleccionContactos()
799                 time.sleep(0.01)
800                 x=x+10
801             seleccion(ConfSocial,Cuadrito,428,0,EstadoRedGSM)
802             x=0
803             while x <= VelocidadCursor:
804                 if GPIO.input(21) == GPIO.HIGH:
805                     sonidoSeleccion()
806                     VerContactos()
807                 time.sleep(0.01)
808                 x=x+10
809             seleccion(ConfSocial,Cuadrito,853,0,EstadoRedGSM)
```

```
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    CuadriculaContactos = "Prueba"
                    SeleccionContactos()
                time.sleep(0.01)
                x=x+10
            seleccion(ConfSocial,Cuadrito,0,290,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    interfazRestaurarSocial()
                time.sleep(0.01)
                x=x+10
            seleccion(ConfSocial,Cuadrito,428,290,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    OpcionesReiniciarModulo()
                time.sleep(0.01)
                x=x+10
            seleccion(ConfSocial,Cuadrito,853,290,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    interfazSocialInformacion()
                time.sleep(0.01)
                x=x+10
            seleccion(ConfSocial,CuadritoPeque,0,0,EstadoRedGSM)
            TiempoEspera(interfazConfiguracion)

    def interfazAgregarContacto(ContactoP):
        global Lista_Contactos
        global Historial_Mensajes
        global frase
        i = 0
        k = 0
        for key in Lista_Contactos:
            if ContactoP == i:
                ContactoAEliminar = key
            i = i + 1
        while True:
            if  ContactoP >= len(Lista_Contactos):
                SeleccionAgregarContacto()
            else:
                TituloFuente = pygame.font.Font(None, 130)
                TituloFuenteB = pygame.font.Font(None, 80)
                Fuente = pygame.font.Font(None, 60)
                screen.fill(ColorDeFondo)
                screen.blit(PictoRegreso, (0, 0))
                TituloInformacionA = TituloFuente.render("Borrar
Contacto:", 1, (Negro))
                TituloInformacionB =
```

```
TituloFuenteB.render(ContactoAEliminar, 1, (Negro))
866                 PrimerParametroA = Fuente.render("¿Está seguro de
borrar el contacto seleccionado" ,1, (Negro))
867                 PrimerParametroB = Fuente.render("para agregar uno
nuevo? Se eliminará el contacto" ,1, (Negro))
868                 PrimerParametroC = Fuente.render("y las
conversaciones" ,1, (Negro))
869                 PrimerParametroD = TituloFuenteB.render("SI" ,1,
(Negro))
870                 PrimerParametroE = TituloFuenteB.render("NO" ,1,
(Negro))
871                 screen.blit(TituloInformacionA, (20, 15))
872                 screen.blit(TituloInformacionB, (20, 140))
873                 screen.blit(PrimerParametroA, (20, 240))
874                 screen.blit(PrimerParametroB, (20, 300))
875                 screen.blit(PrimerParametroC, (20, 360))
876                 screen.blit(PrimerParametroD, (80, 430))
877                 screen.blit(PrimerParametroE, (70, 525))
878                 screen.blit(CuadritoPeque, (0, -190+(95*k)))
879                 pygame.display.flip()
880                 x=0
881                 while x <= VelocidadCursor:
882                     if GPIO.input(21) == GPIO.HIGH:
883                        sonidoSeleccion()
884                        if k == 0:
885                           database = DataBase()
886
database.EliminarContactoBD(ContactoAEliminar)
887                           database.CerrarBD()
888                           SeleccionAgregarContacto()
889                        else:
890                           ContactoAEliminar = ""
891                           SeleccionContactos()
892                     time.sleep(0.01)
893                     x=x+10
894                 k = k + 1
895                 if k >= 3:
896                     k = 0
897
898     def SeleccionAgregarContacto():
899         k = 0
900         Fuente = pygame.font.Font(None, 80)
901         while True:
902             screen.fill(ColorDeFondo)
903             screen.blit(PictoRegreso, (0, 0))
904             PrimerParametroA = Fuente.render("¿Desea ingresar el
nuevo contacto por" ,1, (Negro))
905             PrimerParametroB = Fuente.render("teclado y ratón o de
manera manual?" ,1, (Negro))
906             PrimerParametroC = Fuente.render("Teclado y ratón" ,1,
(Negro))
907             PrimerParametroD = Fuente.render("Manual" ,1, (Negro))
908             screen.blit(PrimerParametroA, (100, 120))
909             screen.blit(PrimerParametroB, (130, 200))
910             screen.blit(PrimerParametroC, (420, 400))
911             screen.blit(PrimerParametroD, (535, 500))
912             if k == 0:
```

```
913                 screen.blit(CuaFila1, (0, 375))
914             elif k == 1:
915                 screen.blit(CuaPredictor, (390, 50))
916             else:
917                 screen.blit(CuadritoPeque, (0, 0))
918             pygame.display.flip()
919             x=0
920             while x <= VelocidadCursor:
921                 if GPIO.input(21) == GPIO.HIGH:
922                     sonidoSeleccion()
923                     if k == 0:
924                         IngresarDatosNuevoContacto()
925                     elif k == 1:
926                         AgregarNombre()
927                     else:
928                         SeleccionContactos()
929                 time.sleep(0.01)
930                 x=x+10
931             k = k + 1
932             if k >= 3:
933                 k = 0
934 
935     def IngresarDatosNuevoContacto():
936         global Lista_Contactos
937         global Historial_Mensajes
938         global NombreContacto
939         global NumeroContacto
940         Contacto = ''
941         Telefono = ''
942         Texto = pygame.font.Font(None, 80)
943         Captura_Nombre = pygame.Rect(350, 150, 600, 80)
944         Captura_Telefono = pygame.Rect(350, 350, 600, 80)
945         Boton_Regresar = pygame.Rect(50, 618, 120, 60)
946         Confirmar_Datos = pygame.Rect(500, 450, 300, 65)
947         Color_Estado1 = Gris
948         Estado_Color1 = False
949         Color_Estado2 = Gris
950         Estado_Color2 = False
951         i = False
952         j = False
953         k = False
954         while True:
955             if GPIO.input(21) == GPIO.HIGH:
956                 time.sleep(0.5)
957                 SeleccionContactos()
958             for event in pygame.event.get():
959                 if event.type == pygame.MOUSEBUTTONDOWN:
960                     if Boton_Regresar.collidepoint(event.pos):
961                         SeleccionContactos()
962                     if Captura_Nombre.collidepoint(event.pos):
963                         Estado_Color1 = not Estado_Color1
964                     else:
965                         Estado_Color1 = False
966                     if Captura_Telefono.collidepoint(event.pos):
967                         Estado_Color2 = not Estado_Color2
968                     else:
969                         Estado_Color2 = False
```

```
970                         Color_Estado1 = Negro if Estado_Color1 else Gris
971                         Color_Estado2 = Negro if Estado_Color2 else Gris
972                         if Confirmar_Datos.collidepoint(event.pos):
973                             if len(Telefono) == 10:
974                                 if Contacto:
975                                     NombreContacto =
Contacto.capitalize()
976                                     NumeroContacto = "+52" + Telefono
977                                     database = DataBase()
978                                     database.GuardarContactoBD()
979                                     database.CerrarBD()
980                                     Lista_Contactos[NombreContacto] =
NumeroContacto
981                                     Historial_Mensajes[NombreContacto] =
[]
982                                     Mensajes_Leidos[NombreContacto] =
True
983                                     NombreContacto = ""
984                                     NumeroContacto = ""
985                                     CuadriculaContactos = ""
986                                     i = True
987                                 else:
988                                     k = True
989                             else:
990                                 j = True
991                 if event.type == pygame.KEYDOWN:
992                 if Estado_Color1:
993                         if event.key == pygame.K_BACKSPACE:
994                             Contacto = Contacto[:-1]
995                         elif event.key == pygame.K_SPACE:
996                             Contacto += " "
997                         else:
998                             letra = event.unicode
999                             if letra.isalnum() and len(Contacto) <
18:
1000                                 Contacto += letra
1001                     elif Estado_Color2:
1002                         if event.key == pygame.K_BACKSPACE:
1003                             Telefono = Telefono[:-1]
1004                         else:
1005                             numero = event.unicode
1006                             if numero.isdecimal() and len(Telefono) <
10:
1007                                 Telefono += numero
1008             screen.fill(ColorDeFondo)
1009             screen.blit(PictoRegreso, (0, 0))
1010             Etiqueta_Instruccion = Texto.render("Inserte datos del
nuevo contacto", 1, (Negro))
1011             Etiqueta_NombreContacto = Texto.render("Nombre:", 1,
(Negro))
1012             Etiqueta_TelefonoContacto = Texto.render("Número
telefónico:", 1, (Negro))
1013             Etiqueta_Lada = Texto.render("+52", 1, (Negro))
1014             Etiqueta_Confirmar = Texto.render("Confirmar", 1,
(Negro))
1015             screen.blit(Etiqueta_Instruccion, (5, 5))
1016             screen.blit(Etiqueta_NombreContacto, (Captura_Nombre.x+5,
```

```
Captura_Nombre.y-65))
1017            screen.blit(Etiqueta_TelefonoContacto,
(Captura_Telefono.x+5, Captura_Telefono.y-65))
1018            screen.blit(Etiqueta_Lada, (Captura_Telefono.x-120,
Captura_Telefono.y+10))
1019            screen.blit(Etiqueta_Confirmar, (Confirmar_Datos.x+15,
Confirmar_Datos.y))
1020            pygame.draw.rect(screen, Color_Estado1, Captura_Nombre,
4)
1021            pygame.draw.rect(screen, Color_Estado2, Captura_Telefono,
4)
1022            pygame.draw.rect(screen, Negro, Boton_Regresar, 6)
1023            pygame.draw.rect(screen, Negro, Confirmar_Datos, 6)
1024            NombreContacto = Texto.render(Contacto, 1, (Negro))
1025            screen.blit(NombreContacto, (Captura_Nombre.x+5,
Captura_Nombre.y+10))
1026            TelefonoContacto = Texto.render(Telefono, 1, (Negro))
1027            screen.blit(TelefonoContacto, (Captura_Telefono.x+5,
Captura_Telefono.y+10))
1028            if i:
1029                Etiqueta_Guardado = Texto.render("Contacto guardado
con éxito", 1, (Negro))
1030                screen.blit(Etiqueta_Guardado, (50, 530))
if j:
1031                Etiqueta_Error = Texto.render("Ingrese un número
telefónico de 10 dígitos", 1, (Negro))
1032                screen.blit(Etiqueta_Error, (50, 530))            if
k:
1033                Etiqueta_Error = Texto.render("Ingrese un nombre de
contacto", 1, (Negro))
1034                screen.blit(Etiqueta_Error, (50, 530))
1035            pygame.display.flip()
1036            if i or j or k:
1037                time.sleep(2)
1038                j = False
1039                k = False
1040                if i:
1041                    i = False
1042                    VerContactos()
1043
1044    def AgregarNombre():
1045        while True:
1046            global frase
1047            EstadoRedGSM = EstadoConexion()
1048            previsualizacionNC(frase,CuaFila1,0,115,EstadoRedGSM)
1049            TiempoEspera(interfazFila1Nombre)
1050            previsualizacionNC(frase,CuaFila2,0,123,EstadoRedGSM)
1051            TiempoEspera(interfazFila2Nombre)
1052            previsualizacionNC(frase,CuaFila2,0,215,EstadoRedGSM)
1053            TiempoEspera(interfazFila3Nombre)
1054            previsualizacionNC(frase,CuaFila2,0,315,EstadoRedGSM)
1055            TiempoEspera(interfazFila4Nombre)
1056            previsualizacionNC(frase,CuaFila2,0,417,EstadoRedGSM)
1057            TiempoEspera(interfazFila5Nombre)
1058            previsualizacionNC("Selección de
contactos",CuaRegresar,0,5,EstadoRedGSM)
1059            x=0
```

```
1060            while x <= VelocidadCursor:
1061                if GPIO.input(21) == GPIO.HIGH:
1062                   sonidoSeleccion()
1063                   frase=""
1064                   SeleccionContactos()
1065                time.sleep(0.01)
1066                x=x+10
1067
1068    def interfazFila1Nombre():
1069        while True:
1070            global frase
1071            EstadoRedGSM = EstadoConexion()
1072            previsualizacionNC(frase +
"a",Cu Letra,280,4,EstadoRedGSM)
1073            x=0
1074            while x <= VelocidadCursor:
1075                if GPIO.input(21) == GPIO.HIGH:
1076                   sonidoSeleccion()
1077                   frase=frase + "a"
1078                   ChecarNombre()
1079                   AgregarNombre()
1080                time.sleep(0.01)
1081                x=x+10
1082            previsualizacionNC(frase +
"e",CuaLetra,385,4,EstadoRedGSM)
1083            x=0
1084            while x <= VelocidadCursor:
1085                if GPIO.input(21) == GPIO.HIGH:
1086                   sonidoSeleccion()
1087                   frase=frase + "e"
1088                   ChecarNombre()
1089                   AgregarNombre()
1090                time.sleep(0.01)
1091                x=x+10
1092            previsualizacionNC(frase +
"i",CuaLetra,495,4,EstadoRedGSM)
1093            x=0
1094            while x <= VelocidadCursor:
1095                if GPIO.input(21) == GPIO.HIGH:
1096                   sonidoSeleccion()
1097                   frase=frase + "i"
1098                   ChecarNombre()
1099                   AgregarNombre()
1100                time.sleep(0.01)
1101                x=x+10
1102            previsualizacionNC(frase +
"o",Cu Letra,610,4,EstadoRedGSM)
1103            x=0
1104            while x <= VelocidadCursor:
1105                if GPIO.input(21) == GPIO.HIGH:
1106                   sonidoSeleccion()
1107                   frase=frase + "o"
1108                   ChecarNombre()
1109                   AgregarNombre()
1110                time.sleep(0.01)
1111                x=x+10
1112            previsualizacionNC(frase +
```

```
"u",CuaLetra,720,4,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    frase=frase + "u"
                    ChecarNombre()
                    AgregarNombre()
                time.sleep(0.01)
                x=x+10
            previsualizacionNC(frase,CuaInicio,0,15,EstadoRedGSM)
            TiempoEspera(AgregarNombre)

    def interfazFila2Nombre():
        while True:
            global frase
            EstadoRedGSM = EstadoConexion()
            previsualizacionNC(frase +
"s",CuaLetra,0,115,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    frase=frase + "s"
                    ChecarNombre()
                    AgregarNombre()
                time.sleep(0.01)
                x=x+10
            previsualizacionNC(frase +
"r",CuaLetra,134,115,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    frase=frase + "r"
                    ChecarNombre()
                    AgregarNombre()
                time.sleep(0.01)
                x=x+10
            previsualizacionNC(frase +
"l",CuaLetra,272,115,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    frase=frase + "l"
                    ChecarNombre()
                    AgregarNombre()
                time.sleep(0.01)
                x=x+10
            previsualizacionNC(frase +
"t",Cu Letra,415,115,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    frase=frase + "t"
```

```
1165                    ChecarNombre()
1166                    AgregarNombre()
1167                time.sleep(0.01)
1168                x=x+10
1169            previsualizacionNC(frase +
"m",Cu Letra,554,115,EstadoRedGSM)
1170            x=0
1171            while x <= VelocidadCursor:
1172                if GPIO.input(21) == GPIO.HIGH:
1173                    sonidoSeleccion()
1174                    frase=frase + "m"
1175                    ChecarNombre()
1176                    AgregarNombre()
1177                time.sleep(0.01)
1178                x=x+10
1179            previsualizacionNC(frase +
"y",CuaLetra,720,115,EstadoRedGSM)
1180            x=0
1181            while x <= VelocidadCursor:
1182                if GPIO.input(21) == GPIO.HIGH:
1183                    sonidoSeleccion()
1184                    frase=frase + "y"
1185                    ChecarNombre()
1186                    AgregarNombre()
1187                time.sleep(0.01)
1188                x=x+10
1189            previsualizacionNC(frase +
"q",CuaLetra,870,115,EstadoRedGSM)
1190            x=0
1191            while x <= VelocidadCursor:
1192                if GPIO.input(21) == GPIO.HIGH:
1193                    sonidoSeleccion()
1194                    frase=frase + "q"
1195                    ChecarNombre()
1196                    AgregarNombre()
1197                time.sleep(0.01)
1198                x=x+10
1199            previsualizacionNC(frase +
"f",CuaLetra,998,115,EstadoRedGSM)
1200            x=0
1201            while x <= VelocidadCursor:
1202                if GPIO.input(21) == GPIO.HIGH:
1203                    sonidoSeleccion()
1204                    frase=frase + "f"
1205                    ChecarNombre()
1206                    AgregarNombre()
1207                time.sleep(0.01)
1208                x=x+10
1209            previsualizacionNC(frase,CuaInicio,0,15,EstadoRedGSM)
1210            TiempoEspera(AgregarNombre)
1211
1212    def interfazFila3Nombre():
1213        while True:
1214            global frase
1215            EstadoRedGSM = EstadoConexion()
1216            previsualizacionNC(frase +
"n",CuaLetra,0,210,EstadoRedGSM)
```

```
1217            x=0
1218            while x <= VelocidadCursor:
1219                if GPIO.input(21) == GPIO.HIGH:
1220                   sonidoSeleccion()
1221                   frase=frase + "n"
1222                   ChecarNombre()
1223                   AgregarNombre()
1224                time.sleep(0.01)
1225                x=x+10
1226            previsualizacionNC(frase +
"d",CuaLet a,134,210,EstadoRedGSM)
1227            x=0
1228            while x <= VelocidadCursor:
1229                if GPIO.input(21) == GPIO.HIGH:
1230                   sonidoSeleccion()
1231                   frase=frase + "d"
1232                   ChecarNombre()
1233                   AgregarNombre()
1234                time.sleep(0.01)
1235                x=x+10
1236            previsualizacionNC(frase +
"c",CuaLetra,272,210,EstadoRedGSM)
1237            x=0
1238            while x <= VelocidadCursor:
1239                if GPIO.input(21) == GPIO.HIGH:
1240                   sonidoSeleccion()
1241                   frase=frase + "c"
1242                   ChecarNombre()
1243                   AgregarNombre()
1244                time.sleep(0.01)
1245                x=x+10
1246            previsualizacionNC(frase +
"p",CuaLetra,415,210,EstadoRedGSM)
1247            x=0
1248            while x <= VelocidadCursor:
1249                if GPIO.input(21) == GPIO.HIGH:
1250                   sonidoSeleccion()
1251                   frase=frase + "p"
1252                   ChecarNombre()
1253                   AgregarNombre()
1254                time.sleep(0.01)
1255                x=x+10
1256            previsualizacionNC(frase +
"h",CuaLetra,554,210,EstadoRedGSM)
1257            x=0
1258            while x <= VelocidadCursor:
1259                if GPIO.input(21) == GPIO.HIGH:
1260                   sonidoSeleccion()
1261                   frase=frase + "h"
1262                   ChecarNombre()
1263                   AgregarNombre()
1264                time.sleep(0.01)
1265                x=x+10
1266            previsualizacionNC(frase +
"g",CuaLetra,724,210,EstadoRedGSM)
1267            x=0
1268            while x <= VelocidadCursor:
```

```
1269                if GPIO.input(21) == GPIO.HIGH:
1270                    sonidoSeleccion()
1271                    frase=frase + "g"
1272                    ChecarNombre()
1273                    AgregarNombre()
1274                time.sleep(0.01)
1275                x=x+10
1276            previsualizacionNC(frase +
"z",Cu Letra,870,210,EstadoRedGSM)
1277            x=0
1278            while x <= VelocidadCursor:
1279                if GPIO.input(21) == GPIO.HIGH:
1280                    sonidoSeleccion()
1281                    frase=frase + "z"
1282                    ChecarNombre()
1283                    AgregarNombre()
1284                time.sleep(0.01)
1285                x=x+10
1286            previsualizacionNC(frase +
"x",CuaLetra,998,210,EstadoRedGSM)
1287            x=0
1288            while x <= VelocidadCursor:
1289                if GPIO.input(21) == GPIO.HIGH:
1290                    sonidoSeleccion()
1291                    frase=frase + "x"
1292                    ChecarNombre()
1293                    AgregarNombre()
1294                time.sleep(0.01)
1295                x=x+10
1296            previsualizacionNC(frase,CuaInicio,0,15,EstadoRedGSM)
1297            TiempoEspera(AgregarNombre)
1298
1299    def interfazFila4Nombre():
1300        while True:
1301            global frase
1302            EstadoRedGSM = EstadoConexion()
1303            previsualizacionNC(frase,CuaLetra,0,310,EstadoRedGSM)
1304            x=0
1305            while x <= VelocidadCursor:
1306                if GPIO.input(21) == GPIO.HIGH:
1307                    sonidoSeleccion()
1308                    frase=frase + " "
1309                    ChecarNombre()
1310                    AgregarNombre()
1311                time.sleep(0.01)
1312                x=x+10
1313            preborrar(frase)
1314
previsualizacionNC(prefrase,CuaLetra,137,310,EstadoRedGSM)
1315            x=0
1316            while x <= VelocidadCursor:
1317                if GPIO.input(21) == GPIO.HIGH:
1318                    sonidoSeleccion()
1319                    borrar(frase)
1320                    AgregarNombre()
1321                time.sleep(0.01)
1322                x=x+10
```

```
1323            previsualizacionNC(frase +
"b",CuaLetra,272,310,EstadoRedGSM)
1324            x=0
1325            while x <= VelocidadCursor:
1326                if GPIO.input(21) == GPIO.HIGH:
1327                   sonidoSeleccion()
1328                   frase=frase + "b"
1329                   ChecarNombre()
1330                   AgregarNombre()
1331                time.sleep(0.01)
1332                x=x+10
1333            previsualizacionNC(frase +
"v",CuaLetra,418,310,EstadoRedGSM)
1334            x=0
1335            while x <= VelocidadCursor:
1336                if GPIO.input(21) == GPIO.HIGH:
1337                   sonidoSeleccion()
1338                   frase=frase + "v"
1339                   ChecarNombre()
1340                   AgregarNombre()
1341                time.sleep(0.01)
1342                x=x+10
1343            previsualizacionNC(frase +
"j",CuaLetra,554,310,EstadoRedGSM)
1344            x=0
1345            while x <= VelocidadCursor:
1346                if GPIO.input(21) == GPIO.HIGH:
1347                   sonidoSeleccion()
1348                   frase=frase + "j"
1349                   ChecarNombre()
1350                   AgregarNombre()
1351                time.sleep(0.01)
1352                x=x+10
1353            previsualizacionNC(frase +
"ñ",CuaLetra,720,310,EstadoRedGSM)
1354            x=0
1355            while x <= VelocidadCursor:
1356                if GPIO.input(21) == GPIO.HIGH:
1357                   sonidoSeleccion()
1358                   frase=frase + "ñ"
1359                   ChecarNombre()
1360                   AgregarNombre()
1361                time.sleep(0.01)
1362                x=x+10
1363            previsualizacionNC(frase +
"k",CuaLetra,870,310,EstadoRedGSM)
1364            x=0
1365            while x <= VelocidadCursor:
1366                if GPIO.input(21) == GPIO.HIGH:
1367                   sonidoSeleccion()
1368                   frase=frase + "k"
1369                   ChecarNombre()
1370                   AgregarNombre()
1371                time.sleep(0.01)
1372                x=x+10
1373            previsualizacionNC(frase +
"w",CuaLetra,1000,310,EstadoRedGSM)
```

```
1374            x=0
1375            while x <= VelocidadCursor:
1376                if GPIO.input(21) == GPIO.HIGH:
1377                   sonidoSeleccion()
1378                   frase=frase + "w"
1379                   ChecarNombre()
1380                   AgregarNombre()
1381                time.sleep(0.01)
1382                x=x+10
1383            previsualizacionNC(frase,CuaInicio,0,15,EstadoRedGSM)
1384            TiempoEspera(AgregarNombre)
1385
1386    def interfazFila5Nombre():
1387        while True:
1388            global frase
1389            global NombreContacto
1390            EstadoRedGSM = EstadoConexion()
1391            previsualizacionNC(frase,CuaNuevaFrase,-
665,15 ,EstadoRedGSM)
1392            x=0
1393            while x <= VelocidadCursor:
1394                if GPIO.input(21) == GPIO.HIGH:
1395                   if len(frase) == 0:
1396                       frase = "NingunCaracterPuesto"
1397                   ChecarNombre()
1398                   NombreContacto = frase
1399                   frase = ""
1400                   AgregarNumero()
1401                time.sleep(0.01)
1402                x=x+10
1403            previsualizacionNC("Borrar todo",CuaNuevaFrase,-
313,15 ,EstadoRedGSM)
1404            x=0
1405            while x <= VelocidadCursor:
1406                if GPIO.input(21) == GPIO.HIGH:
1407                   sonidoSeleccion()
1408                   frase=""
1409                   AgregarNombre()
1410                time.sleep(0.01)
1411                x=x+10
1412            previsualizacionNC(frase,CuaInicio,0,15,EstadoRedGSM)
1413            TiempoEspera(AgregarNombre)
1414
1415    def ChecarNombre():
1416        global frase
1417        if len(frase) > 15:
1418            if frase == "NingunCaracterPuesto":
1419                m = "Escribe el nombre del contacto"
1420                frase = ""
1421            else:
1422                m = "Máximo de caracteres usados"
1423                borrar(frase)
1424            screen.fill(ColorDeFondo)
1425            screen.blit(AbecedarioContacto, (0, 0))
1426            pygame.draw.rect(screen,Negro,(170,625,900,90))
1427            pygame.draw.rect(screen,Blanco,(175,630,890,80))
1428            fuente = pygame.font.Font(None, 80)
```

```
1429                mensaje = fuente.render(m, 1, (0, 0, 0))
1430                screen.blit(mensaje, (215, 635))
1431                pygame.display.flip()
1432                pygame.time.delay(3000)
1433                AgregarNombre()
1434            else:
1435                None
1436
1437        def AgregarNumero():
1438            pygame.time.delay(250)
1439            while True:
1440                global frase
1441                global NombreContacto
1442                EstadoRedGSM = EstadoConexion()
1443
previsualizacionNumericaNC(frase,CuaFilaNumeros,0,0,EstadoRedGSM)
1444                TiempoEspera(interfazFila1NumTel)
1445
previsualizacionNumericaNC(frase,CuaFilaNumeros,0,210,EstadoRedGSM)
1446                TiempoEspera(interfazFila2NumTel)
1447
previsualizacionNumericaNC(frase,CuaFila2,60,410,EstadoRedGSM)
1448                TiempoEspera(interfazFila3NumTel)
1449                previsualizacionNumericaNC("Nombre del
contacto",CuaRegresar,0,5,EstadoRedGSM)
1450                x=0
1451                while x <= VelocidadCursor:
1452                    if GPIO.input(21) == GPIO.HIGH:
1453                        sonidoSeleccion()
1454                        frase = ""
1455                        NombreContacto = ""
1456                        AgregarNombre()
1457                    time.sleep(0.01)
1458                    x=x+10
1459
1460        def interfazFila1NumTel():
1461            while True:
1462                global frase
1463                EstadoRedGSM = EstadoConexion()
1464                previsualizacionNumericaNC(frase +
"1",CuaLetra,112,13,EstadoRedGSM)
1465                x=0
1466                while x <= VelocidadCursor:
1467                    if GPIO.input(21) == GPIO.HIGH:
1468                        sonidoSeleccion()
1469                        frase=frase + "1"
1470                        ChecarNumero()
1471                        AgregarNumero()
1472                    time.sleep(0.01)
1473                    x=x+10
1474                previsualizacionNumericaNC(frase +
"2",Cu Letra,312,13,EstadoRedGSM)
1475                x=0
1476                while x <= VelocidadCursor:
1477                    if GPIO.input(21) == GPIO.HIGH:
1478                        sonidoSeleccion()
1479                        frase=frase + "2"
```

```
1480                    ChecarNumero()
1481                    AgregarNumero()
1482                time.sleep(0.01)
1483                x=x+10
1484            previsualizacionNumericaNC(frase +
"3",Cu Letra,512,13,EstadoRedGSM)
1485            x=0
1486            while x <= VelocidadCursor:
1487                if GPIO.input(21) == GPIO.HIGH:
1488                    sonidoSeleccion()
1489                    frase=frase + "3"
1490                    ChecarNumero()
1491                    AgregarNumero()
1492                time.sleep(0.01)
1493                x=x+10
1494            previsualizacionNumericaNC(frase +
"4",CuaLetra,712,13,EstadoRedGSM)
1495            x=0
1496            while x <= VelocidadCursor:
1497                if GPIO.input(21) == GPIO.HIGH:
1498                    sonidoSeleccion()
1499                    frase=frase + "4"
1500                    ChecarNumero()
1501                    AgregarNumero()
1502                time.sleep(0.01)
1503                x=x+10
1504            previsualizacionNumericaNC(frase +
"5",CuaLetra,912,13,EstadoRedGSM)
1505            x=0
1506            while x <= VelocidadCursor:
1507                if GPIO.input(21) == GPIO.HIGH:
1508                    sonidoSeleccion()
1509                    frase=frase + "5"
1510                    ChecarNumero()
1511                    AgregarNumero()
1512                time.sleep(0.01)
1513                x=x+10
1514
previsualizacionNumericaNC(frase,CuaInicio,0,0,EstadoRedGSM)
1515            TiempoEspera(AgregarNumero)
1516
1517    def interfazFila2NumTel():
1518        while True:
1519            global frase
1520            EstadoRedGSM = EstadoConexion()
1521            previsualizacionNumericaNC(frase +
"6",Cu Letra,112,227,EstadoRedGSM)
1522            x=0
1523            while x <= VelocidadCursor:
1524                if GPIO.input(21) == GPIO.HIGH:
1525                    sonidoSeleccion()
1526                    frase=frase + "6"
1527                    ChecarNumero()
1528                    AgregarNumero()
1529                time.sleep(0.01)
1530                x=x+10
1531            previsualizacionNumericaNC(frase +
```

```
"7",CuaLetra,312,227,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    frase=frase + "7"
                    ChecarNumero()
                    AgregarNumero()
                time.sleep(0.01)
                x=x+10
            previsualizacionNumericaNC(frase +
"8",CuaLetra,512,227,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    frase=frase + "8"
                    ChecarNumero()
                    AgregarNumero()
                time.sleep(0.01)
                x=x+10
            previsualizacionNumericaNC(frase +
"9",CuaLetra,712,227,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    frase=frase + "9"
                    ChecarNumero()
                    AgregarNumero()
                time.sleep(0.01)
                x=x+10
            previsualizacionNumericaNC(frase +
"0",CuaLetra,912,227,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    frase=frase + "0"
                    ChecarNumero()
                    AgregarNumero()
                time.sleep(0.01)
                x=x+10

previsualizacionNumericaNC(frase,CuaInicio,0,0,EstadoRedGSM)
            TiempoEspera(AgregarNumero)

    def interfazFila3NumTel():
        while True:
            global frase
            global NumeroContacto
            EstadoRedGSM = EstadoConexion()
            previsualizacionNumericaNC(frase,CuaNuevaFrase,-
630,0,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
```

```
1583                    if len(frase) == 0:
1584                        frase = "NingunCaracterPuesto"
1585                    if len(frase) < 10:
1586                        frase = frase + "XXXXXXXXXX"
1587                    ChecarNumero()
1588                    NumeroContacto = frase
1589                    frase = ""
1590                    ContactoAgregado()
1591                time.sleep(0.01)
1592                x=x+10
1593            preborrar(frase)
1594
previsualizacionNumericaNC(prefrase,CuaBorrar,5,0,EstadoRedGSM)
1595            x=0
1596            while x <= VelocidadCursor:
1597                if GPIO.input(21) == GPIO.HIGH:
1598                    sonidoSeleccion()
1599                    borrar(frase)
1600                    AgregarNumero()
1601                time.sleep(0.01)
1602                x=x+10
1603            previsualizacionNumericaNC("Borrar todo",CuaNuevaFrase,-
25,0,EstadoRedGSM)
1604            x=0
1605            while x <= VelocidadCursor:
1606                if GPIO.input(21) == GPIO.HIGH:
1607                    sonidoSeleccion()
1608                    frase = ""
1609                    AgregarNumero()
1610                time.sleep(0.01)
1611                x=x+10
1612
previsualizacionNumericaNC(frase,CuaInicio,0,0,EstadoRedGSM)
1613            x=0
1614            while x <= VelocidadCursor:
1615                if GPIO.input(21) == GPIO.HIGH:
1616                    sonidoSeleccion()
1617                    AgregarNumero()
1618                time.sleep(0.01)
1619                x=x+10
1620
1621    def ChecarNumero():
1622        global frase
1623        if len(frase) > 10:
1624            if frase == "NingunCaracterPuesto":
1625                m = "Escribe el teléfono del contacto"
1626                frase = ""
1627            else:
1628                if frase.find("XXXXXXXXXX") != -1:
1629                    m = "El numero debe tener 10 dígitos"
1630                    frase = frase[:len(frase)-10]
1631                else:
1632                    m = "Solo se permiten 10 dígitos"
1633                    borrar(frase)
1634            screen.fill(ColorDeFondo)
1635            screen.blit(NumerosContacto, (0, 0))
1636            pygame.draw.rect(screen,Negro,(170,625,900,90))
```

```
1637            pygame.draw.rect(screen,Blanco,(175,630,890,80))
1638            fuente = pygame.font.Font(None, 80)
1639            mensaje = fuente.render(m, 1, (0, 0, 0))
1640            screen.blit(mensaje, (215, 635))
1641            pygame.display.flip()
1642            pygame.time.delay(3000)
1643            AgregarNumero()
1644        else:
1645            None
1646
1647    def ContactoAgregado():
1648        global NombreContacto
1649        global NumeroContacto
1650        global Lista_Contactos
1651        global Historial_Mensajes
1652        fuente = pygame.font.Font(None, 170)
1653        fuente2 = pygame.font.Font(None, 85)
1654        NombreContacto = NombreContacto.capitalize()
1655        NumeroContacto = "+52" + NumeroContacto
1656        database = DataBase()
1657        database.GuardarContactoBD()
1658        database.CerrarBD()
1659        Lista_Contactos[NombreContacto] = NumeroContacto
1660        Historial_Mensajes[NombreContacto] = []
1661        Mensajes_Leidos[NombreContacto] = True
1662        Contacto = NombreContacto + ": " + NumeroContacto
1663        mensaje = fuente.render("Contacto guardado", 1, (Blanco))
1664        mensaje2 = fuente.render("correctamente",1, (Blanco))
1665        mensaje3 = fuente2.render(Contacto,1, (Blanco))
1666        screen.fill(Negro)
1667        screen.blit(mensaje, (70, 150))
1668        screen.blit(mensaje2, (70, 320))
1669        screen.blit(mensaje3, (70, 500))
1670        pygame.display.flip()
1671        pygame.time.delay(5000)
1672        NombreContacto = ""
1673        NumeroContacto = ""
1674        CuadriculaContactos = ""
1675        VerContactos()
1676
1677    def VerContactos():
1678        global Lista_Contactos
1679        while True:
1680            Fuente = pygame.font.Font(None, 50)
1681            screen.fill(ColorDeFondo)
1682            screen.blit(PictoRegreso, (0, 0))
1683            screen.blit(CuadritoPeque, (0, 0))
1684            i = 0
1685            j = 0
1686            for key in Lista_Contactos:
1687                if i > 2:
1688                    j = j + 1
1689                    i = 0
1690                N = key
1691                T = Lista_Contactos[key]
1692                Nombre = Fuente.render(N,1,(Negro))
1693                Telefono = Fuente.render(T,1,(Negro))
```

```
1694                 screen.blit(Nombre, (40+(425*i), 25+(115*j)))
1695                 screen.blit(Telefono, (40+(425*i), 75+(115*j)))
1696                 i = i + 1
1697             pygame.display.flip()
1698             TiempoEspera(interfazConfiguracionSocial)
1699
1700     def interfazPruebaSMS(ContactoP):
1701         global Lista_Contactos
1702         global Historial_Mensajes
1703         global CuadriculaContactos
1704         EstadoRedGSM = EstadoConexion()
1705         Fuente = pygame.font.Font(None, 170)
1706         Fuente2 = pygame.font.Font(None, 170)
1707         i = 0
1708         if EstadoRedGSM == "Sin conexión":
1709             screen.fill(Negro)
1710             PrimerParametroA = Fuente.render("Sin conexión a la red"
,1, (Blanco))
1711             PrimerParametroB = Fuente2.render("Reiniciar modulo GSM"
,1, (Blanco))
1712             screen.blit(PrimerParametroA, (20, 190))
1713             screen.blit(PrimerParametroB, (20, 390))
1714             pygame.display.flip()
1715             time.sleep(3)
1716             CuadriculaContactos = ""
1717             interfazConfiguracionSocial()
1718         else:
1719             for key in Lista_Contactos:
1720                 if ContactoP == i:
1721                     ContactoPrueba = key
1722                 i = i + 1
1723             while True:
1724                 if  ContactoP >= len(Lista_Contactos):
1725                     screen.fill(Negro)
1726                     PrimerParametroA = Fuente.render("Selección" ,1,
(Blanco))
1727                     PrimerParametroB = Fuente.render("inexistente"
,1, (Blanco))
1728                     screen.blit(PrimerParametroA, (370, 190))
1729                     screen.blit(PrimerParametroB, (340, 390))
1730                     pygame.display.flip()
1731                     time.sleep(4)
1732                     SeleccionContactos()
1733                 else:
1734                     NumeroPrueba = Lista_Contactos[ContactoPrueba]
1735                     NumeroAEnviar = NumeroPrueba.encode("iso-8859-1")
1736                     Mensaje = 'Mensaje de prueba SMS\x1A\r\n'
1737                     MensajeAEnviar = Mensaje.encode("iso-8859-1")
1738                     A6.open()
1739                     A6.write(b'AT+CMGS=')
1740                     A6.write(NumeroAEnviar)
1741                     A6.write(b'\r\n')
1742                     A6.write(MensajeAEnviar)
1743                     MensajeSinCtrlZ = Mensaje.split('\x1A\r\n')
1744                     MensajeAGuardar = MensajeSinCtrlZ[0]
1745                     MensajeAGuardar = "Usuario SAAC" + ": " +
MensajeAGuardar
```

```
1746                    database = DataBase()
1747                    database.GuardarMensajeBD(NumeroPrueba,
MensajeAGuardar)
1748                    database.CerrarBD()
1749
Historial_Mensajes[ContactoPrueba].append(MensajeAGuardar)
1750                    screen.fill(Negro)
1751                    SegundoParametroA = Fuente.render("Prueba SMS"
,1, (Blanco))
1752                    SegundoParametroB = Fuente.render("realizada" ,1,
(Blanco))
1753                    screen.blit(SegundoParametroA, (290, 190))
1754                    screen.blit(SegundoParametroB, (370, 390))
1755                    pygame.display.flip()
1756                    time.sleep(3)
1757                    CuadriculaContactos = ""
1758                    A6.write(b'AT\r')
1759                    A6.close()
1760                    interfazConfiguracionSocial()
1761
1762    def interfazRestaurarSocial():
1763        while True:
1764            RevisarMensajesNoLeidos()
1765            EstadoRedGSM = EstadoConexion()
1766            seleccion(ConfRestaurar,Cuadrito,0,0,EstadoRedGSM)
1767            x=0
1768            while x <= VelocidadCursor:
1769                if GPIO.input(21) == GPIO.HIGH:
1770                    sonidoSeleccion()
1771                    database = DataBase()
1772                    database.RestaurarSocialBD()
1773                    database.CerrarBD()
1774                    fuente = pygame.font.Font(None, 170)
1775                    mensaje = fuente.render("Restauración exitosa", 1,
(Blanco))
1776                    mensaje2 = fuente.render("de la interfaz
social",1, (Blanco))
1777                    screen.fill(Negro)
1778                    screen.blit(mensaje, (70, 200))
1779                    screen.blit(mensaje2, (70, 380))
1780                    pygame.display.flip()
1781                    pygame.time.delay(5000)
1782                    interfazConfiguracionSocial()
1783                time.sleep(0.01)
1784                x=x+10
1785            seleccion(ConfRestaurar,Cuadrito,428,0,EstadoRedGSM)
1786            TiempoEspera(interfazConfiguracionSocial)
1787            seleccion(ConfRestaurar,Cuadrito,853,0,EstadoRedGSM)
1788            TiempoEspera(RestaurarSocialInformacion)
1789            seleccion(ConfRestaurar,CuadritoPeque,0,0,EstadoRedGSM)
1790            TiempoEspera(interfazConfiguracionSocial)
1791
1792    def RestaurarSocialInformacion():
1793        while True:
1794            InformacionRestauracionSocial()
1795            TiempoEspera(interfazRestaurarSocial)
1796
```

```
1797    def OpcionesReiniciarModulo():
1798        while True:
1799            EstadoRedGSM = EstadoConexion()
1800            seleccion(ConfRestaurar,Cuadrito,0,0,EstadoRedGSM)
1801            x=0
1802            while x <= VelocidadCursor:
1803                if GPIO.input(21) == GPIO.HIGH:
1804                    sonidoSeleccion()
1805                    fuente = pygame.font.Font(None, 170)
1806                    mensaje = fuente.render("Reinicio del módulo", 1,
(Blanco))
1807                    mensaje2 = fuente.render("GMS completado",1,
(Blanco))
1808                    screen.fill(Negro)
1809                    screen.blit(mensaje, (70, 200))
1810                    screen.blit(mensaje2, (70, 380))
1811                    pygame.display.flip()
1812                    GPIO.output(20, True)
1813                    time.sleep(2)
1814                    GPIO.output(20, False)
1815                    time.sleep(2)
1816                    interfazConfiguracionSocial()
1817                time.sleep(0.01)
1818                x=x+10
1819            seleccion(ConfRestaurar,Cuadrito,428,0,EstadoRedGSM)
1820            TiempoEspera(interfazConfiguracionSocial)
1821            seleccion(ConfRestaurar,Cuadrito,853,0,EstadoRedGSM)
1822            TiempoEspera(ReiniciarModuloInformacion)
1823            seleccion(ConfRestaurar,CuadritoPeque,0,0,EstadoRedGSM)
1824            TiempoEspera(interfazConfiguracionSocial)
1825
1826    def ReiniciarModuloInformacion():
1827        while True:
1828            InformacionReiniciarModulo()
1829            TiempoEspera(interfazRestaurarSocial)
1830
1831    def InformacionReiniciarModulo():
1832        TituloFuente = pygame.font.Font(None, 130)
1833        Negrita = pygame.font.Font(None, 75)
1834        screen.fill(ColorDeFondo)
1835        screen.blit(PictoRegreso, (0, 0))
1836        screen.blit(CuadritoPeque, (0, 0))
1837        TituloInformacion = TituloFuente.render("Reinicio del módulo
GSM", 1, (Negro))
1838        PrimerParametro = Negrita.render("Se reiniciará el módulo
encargado del envío" ,1, (Negro))
1839        PrimerParametroB = Negrita.render("y recepción de mensajes
SMS. Realizar" ,1, (Negro))
1840        PrimerParametroC = Negrita.render("cuando el estado de la red
sea: 'Sin conexión'." ,1, (Negro))
1841        PrimerParametroD = Negrita.render("Esta acción solo afectará
a los mensajes SMS," ,1, (Negro))
1842        PrimerParametroE = Negrita.render("el sistema seguirá
funcionando correctamente." ,1, (Negro))
1843        screen.blit(TituloInformacion, (20, 15))
1844        screen.blit(PrimerParametro, (20, 150))
1845        screen.blit(PrimerParametroB, (20, 230))
```

```
1846            screen.blit(PrimerParametroC, (20, 310))
1847            screen.blit(PrimerParametroD, (20, 390))
1848            screen.blit(PrimerParametroE, (20, 470))
1849            pygame.display.flip()
1850
1851        def interfazSocialInformacion():
1852            global NumeroUsuario
1853            global CompañiaUsuario
1854            EstadoRedGSM = EstadoConexion()
1855            while True:
1856                TituloFuente = pygame.font.Font(None, 110)
1857                Fuente = pygame.font.Font(None, 60)
1858                screen.fill(ColorDeFondo)
1859                screen.blit(PictoRegreso, (0, 0))
1860                screen.blit(CuadritoPeque, (0, 0))
1861                screen.blit(SimboloEnviar, (-1120, -160))
1862                if EstadoRedGSM == "Sin conexión":
1863                    screen.blit(MalaSeñal, (370, 310))
1864                else:
1865                    screen.blit(BuenaSeñal, (370, 310))
1866                TituloInformacion = TituloFuente.render("Información de
la interfaz social", 1, (Negro))
1867                PrimerParametroA = Fuente.render("Número telefónico del
usuario: "+NumeroUsuario,1, (Negro))
1868                PrimerParametroB = Fuente.render("Compañía telefónica del
usuario: "+CompañiaUsuario,1, (Negro))
1869                PrimerParametroC = Fuente.render("Estado de la red:
"+EstadoRedGSM,1, (Negro))
1870                PrimerParametroD = Fuente.render("       : Este símbolo
tiene la función de enviar la frase creada" ,1, (Negro))
1871                PrimerParametroE = Fuente.render("como un mensaje. Al
seleccionarlo, se debe indicar el formato " ,1, (Negro))
1872                PrimerParametroF = Fuente.render("y el contacto al que se
le enviará el mensaje." ,1, (Negro))
1873                screen.blit(TituloInformacion, (20, 15))
1874                screen.blit(PrimerParametroA, (20, 180))
1875                screen.blit(PrimerParametroB, (20, 250))
1876                screen.blit(PrimerParametroC, (20, 320))
1877                screen.blit(PrimerParametroD, (20, 390))
1878                screen.blit(PrimerParametroE, (20, 450))
1879                screen.blit(PrimerParametroF, (20, 510))
1880                pygame.display.flip()
1881                TiempoEspera(interfazConfiguracionSocial)
1882
1883        def EnviarMensajeSMS(ContactoP, Mensaje):
1884            global Lista_Contactos
1885            global Historial_Mensajes
1886            global CuadriculaContactos
1887            global formato
1888            EstadoRedGSM = EstadoConexion()
1889            Fuente = pygame.font.Font(None, 170)
1890            i = 0
1891            if isinstance(ContactoP, int) == True:
1892                for key in Lista_Contactos:
1893                    if ContactoP == i:
1894                        Contacto = key
1895                    i = i + 1
```

```
1896            if  ContactoP >= len(Lista_Contactos):
1897                    screen.fill(Negro)
1898                    PrimerParametroA = Fuente.render("Selección" ,1,
(Blanco))
1899                    PrimerParametroB = Fuente.render("inexistente"
,1, (Blanco))
1900                    screen.blit(PrimerParametroA, (370, 190))
1901                    screen.blit(PrimerParametroB, (340, 390))
1902                    pygame.display.flip()
1903                    time.sleep(4)
1904                    SeleccionContactos()
1905        else:
1906            Contacto = ContactoP
1907        if EstadoRedGSM == "Sin conexión":
1908            screen.fill(Negro)
1909            PrimerParametroA = Fuente.render("Sin conexión a la red"
,1, (Blanco))
1910            PrimerParametroB = Fuente.render("Reiniciar módulo GSM"
,1, (Blanco))
1911            screen.blit(PrimerParametroA, (20, 190))
1912            screen.blit(PrimerParametroB, (20, 390))
1913            pygame.display.flip()
1914            time.sleep(3)
1915            CuadriculaContactos = ""
1916            formato = ""
1917            if PlantillaSocial == "Activada":
1918                VerChatSMS(Contacto)
1919            else:
1920                interfazSocial()
1921        else:
1922            NumeroContacto = Lista_Contactos[Contacto]
1923            NumeroAEnviar = NumeroContacto.encode("iso-8859-1")
1924            Mensaje = Mensaje + '\x1A\r\n'
1925            MensajeAEnviar = Mensaje.encode("iso-8859-1")
1926            A6.open()
1927            A6.write(b'AT+CMGS=')
1928            A6.write(NumeroAEnviar)
1929            A6.write(b'\r\n')
1930            A6.write(MensajeAEnviar)
1931            MensajeSinCtrlZ = Mensaje.split('\x1A\r\n')
1932            MensajeAGuardar = MensajeSinCtrlZ[0]
1933            MensajeAGuardar = "Usuario SAAC" + ": " + MensajeAGuardar
1934            database = DataBase()
1935            database.GuardarMensajeBD(NumeroContacto,
MensajeAGuardar)
1936            database.CerrarBD()
1937            Historial_Mensajes[Contacto].append(MensajeAGuardar)
1938            screen.fill(Negro)
1939            SegundoParametroA = Fuente.render("Mensaje SMS" ,1,
(Blanco))
1940            SegundoParametroB = Fuente.render("enviado" ,1, (Blanco))
1941            screen.blit(SegundoParametroA, (290, 190))
1942            screen.blit(SegundoParametroB, (400, 390))
1943            pygame.display.flip()
1944            time.sleep(3)
1945            A6.write(b'AT\r')
1946            A6.close()
```

```
1947            formato = ""
1948            if PlantillaSocial == "Activada":
1949                VerChatSMS(Contacto)
1950            else:
1951                interfazSocial()
1952
1953    def VerChatSMS(ContactoP):
1954        global Lista_Contactos
1955        global Historial_Mensajes
1956        global ContactoElegido
1957        global formato
1958        global PlantillaSocial
1959        Fuente = pygame.font.Font(None, 170)
1960        Titulo = pygame.font.Font(None, 90)
1961        Texto = pygame.font.Font(None, 50)
1962        i = 0
1963        k = 0
1964        l = 0
1965        if isinstance(ContactoP, int) == True:
1966            for key in Lista_Contactos:
1967                if ContactoP == i:
1968                    ContactoElegido = key
1969                i = i + 1
1970            if ContactoP >= len(Lista_Contactos):
1971                    screen.fill(Negro)
1972                    PrimerParametroA = Fuente.render("Selección" ,1,
(Blanco))
1973                    PrimerParametroB = Fuente.render("inexistente"
,1, (Blanco))
1974                    screen.blit(PrimerParametroA, (370, 190))
1975                    screen.blit(PrimerParametroB, (340, 390))
1976                    pygame.display.flip()
1977                    time.sleep(4)
1978                    ContactoElegido = ""
1979                    SeleccionContactos()
1980        else:
1981            ContactoElegido = ContactoP
1982        j = len(Historial_Mensajes[ContactoElegido]) - 12
1983        if j < 0:
1984            j = 0
1985        Mensajes_Leidos[ContactoElegido] = True
1986        while True:
1987            RevisarMensajesNoLeidos()
1988            if Mensajes_Leidos[ContactoElegido] == False:
1989                Mensajes_Leidos[ContactoElegido] = True
1990            screen.fill(ColorDeFondo)
1991            screen.blit(ChatSMS, (0, 0))
1992            c = Titulo.render(ContactoElegido, True, Negro)
1993            screen.blit(c, (10, 5))
1994            if len(Historial_Mensajes[ContactoElegido]) < 12:
1995                for i in range(0,
len(Historial_Mensajes[ContactoElegido])):
1996                    Historial =
Texto.render(Historial_Mensajes[ContactoElegido][i], True, Negro)
1997                    screen.blit(Historial, (10, 80 + (i*45)))
1998            elif len(Historial_Mensajes[ContactoElegido]) >= 12:
1999                for i in range(0 + j, 12 + j):
```

```
                    Historial =
Texto.render(Historial_Mensajes[ContactoElegido][i], True, Negro)
                    screen.blit(Historial, (10, 80 + ((i-j)*45)))
            if k == 0:
                screen.blit(CuadritoPeque, (-20, 20))
            elif k == 1:
                screen.blit(CuadritoPeque, (157, 20))
            elif k == 2:
                screen.blit(CuadritoPeque, (334, 20))
            elif k == 3:
                screen.blit(CuaPredictor, (475, 190))
            elif k == 4:
                screen.blit(CuadritoPeque, (860, 20))
            elif k == 5:
                screen.blit(CuaInicio, (0, 15))
            pygame.display.flip()
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                    sonidoSeleccion()
                    if k == 1:
                        if j > 0:
                            j = j - 1
                    elif k == 2:
                        if j <
(len(Historial_Mensajes[ContactoElegido]) - 12):
                            j = j + 1
                    elif k == 3:
                        SeleccionInterfazRes()
                    elif k == 4:
                        if len(Historial_Mensajes[ContactoElegido])
!= 0:
                            BorrarChatSMS(ContactoElegido)
                    elif k == 5:
                        ContactoElegido = ""
                        formato = ""
                        PlantillaSocial = "Desactivada"
                        Selecciondeinterfaz()
                    else:
                        ContactoElegido = ""
                        SeleccionContactos()
                time.sleep(0.01)
                x=x+10
            k = k + 1
            if k >= 6:
                k = 0

    def BorrarChatSMS(Contacto):
        k = 0
        while True:
            TituloFuente = pygame.font.Font(None, 130)
            TituloFuenteB = pygame.font.Font(None, 80)
            Fuente = pygame.font.Font(None, 75)
            screen.fill(ColorDeFondo)
            screen.blit(PictoRegreso, (0, 0))
            TituloInformacionA = TituloFuente.render("Borrar
mensaje(s)", 1, (Negro))
```

```
2053            TituloInformacionB = TituloFuenteB.render(Contacto, 1,
(Negro))
2054            PrimerParametroA = Fuente.render("¿Desea eliminar todos
los mensajes del" ,1, (Negro))
2055            PrimerParametroB = Fuente.render("historial o únicamente
el último mensaje?" ,1, (Negro))
2056            PrimerParametroD = TituloFuenteB.render("Todos" ,1,
(Negro))
2057            PrimerParametroE = TituloFuenteB.render("El último" ,1,
(Negro))
2058            screen.blit(TituloInformacionA, (20, 15))
2059            screen.blit(TituloInformacionB, (20, 140))
2060            screen.blit(PrimerParametroA, (20, 250))
2061            screen.blit(PrimerParametroB, (20, 330))
2062            screen.blit(PrimerParametroD, (50, 430))
2063            screen.blit(PrimerParametroE, (45, 525))
2064            if k == 0:
2065                screen.blit(CuaNumero, (-400, -108))
2066            elif k == 1:
2067                screen.blit(CuaNumero, (-400, -15))
2068            elif k == 2:
2069                screen.blit(CuadritoPeque, (0, 0))
2070            pygame.display.flip()
2071            x=0
2072            while x <= VelocidadCursor:
2073                if GPIO.input(21) == GPIO.HIGH:
2074                    sonidoSeleccion()
2075                    if k == 0:
2076                        database = DataBase()
2077                        database.EliminarHistorialMensajes(Contacto)
2078                        database.CerrarBD()
2079                        time.sleep(1)
2080                        VerChatSMS(Contacto)
2081                    elif k == 1:
2082                        database = DataBase()
2083                        database.EliminarUltimoMensaje(Contacto)
2084                        database.CerrarBD()
2085                        time.sleep(1)
2086                        VerChatSMS(Contacto)
2087                    elif k == 2:
2088                        VerChatSMS(Contacto)
2089                time.sleep(0.01)
2090                x=x+10
2091            k = k + 1
2092            if k >= 3:
2093                k = 0
2094
2095    def SeleccionInterfazRes():
2096        global ContactoElegido
2097        global Mensajes_Leidos
2098        global formato
2099        while True:
2100            RevisarMensajesNoLeidos()
2101            EstadoRedGSM = EstadoConexion()
2102            screen.fill(ColorDeFondo)
2103            screen.blit(SeleccionInterfaz, (0, 0))
2104            screen.blit(CursorSelInt, (0, 5))
```

```
2105            screen.blit(PictoRegreso, (0, 0))
2106            pygame.draw.rect(screen,ColorDeFondo,(0,0,1200,115))
2107            pygame.draw.rect(screen,ColorDeFondo,(900,115,400,400))
2108            pygame.draw.rect(screen,ColorDeFondo,(200,440,1000,500))
2109            if EstadoRedGSM == "Conexión establecida":
2110                screen.blit(BuenaSeñal, (5,5))
2111            else:
2112                screen.blit(MalaSeñal, (5,5))
2113            for key in Mensajes_Leidos:
2114                if Mensajes_Leidos.get(key) == True:
2115                    continue
2116                else:
2117                    screen.blit(NuevoSMS, (5,60))
2118                    break
2119            pygame.display.flip()
2120            x=0
2121            while x <= VelocidadCursor:
2122                if GPIO.input(21) == GPIO.HIGH:
2123                   sonidoSeleccion()
2124                   frase=""
2125                   interfazAlfabetica()
2126                time.sleep(0.01)
2127                x=x+10
2128            screen.fill(ColorDeFondo)
2129            screen.blit(SeleccionInterfaz, (0, 0))
2130            screen.blit(CursorSelInt, (427, 5))
2131            screen.blit(PictoRegreso, (0, 0))
2132            pygame.draw.rect(screen,ColorDeFondo,(0,0,1200,115))
2133            pygame.draw.rect(screen,ColorDeFondo,(900,115,400,400))
2134            pygame.draw.rect(screen,ColorDeFondo,(200,440,1000,500))
2135            if EstadoRedGSM == "Conexión establecida":
2136                screen.blit(BuenaSeñal, (5,5))
2137            else:
2138                screen.blit(MalaSeñal, (5,5))
2139            for key in Mensajes_Leidos:
2140                if Mensajes_Leidos.get(key) == True:
2141                    continue
2142                else:
2143                    screen.blit(NuevoSMS, (5,60))
2144                    break
2145            pygame.display.flip()
2146            x=0
2147            while x <= VelocidadCursor:
2148                if GPIO.input(21) == GPIO.HIGH:
2149                   sonidoSeleccion()
2150                   Idioma="-ves+"
2151                   SeleccionderinterfazPictografica()
2152                time.sleep(0.01)
2153                x=x+10
2154            screen.fill(ColorDeFondo)
2155            screen.blit(SeleccionInterfaz, (0, 0))
2156            screen.blit(CuadritoPeque, (0, 0))
2157            screen.blit(PictoRegreso, (0, 0))
2158            pygame.draw.rect(screen,ColorDeFondo,(0,0,1200,115))
2159            pygame.draw.rect(screen,ColorDeFondo,(900,115,400,400))
2160            pygame.draw.rect(screen,ColorDeFondo,(200,440,1000,500))
2161            if EstadoRedGSM == "Conexión establecida":
```

```
2162                 screen.blit(BuenaSeñal, (5,5))
2163             else:
2164                 screen.blit(MalaSeñal, (5,5))
2165             for key in Mensajes_Leidos:
2166                 if Mensajes_Leidos.get(key) == True:
2167                     continue
2168                 else:
2169                     screen.blit(NuevoSMS, (5,60))
2170                     break
2171             pygame.display.flip()
2172             x=0
2173             while x <= VelocidadCursor:
2174                 if GPIO.input(21) == GPIO.HIGH:
2175                    sonidoSeleccion()
2176                    if formato == "SMS":
2177                        VerChatSMS(ContactoElegido)
2178                    elif formato == "WHATSAPP":
2179                        driver.set_window_size(1100, 720)
2180
driver.switch_to.window(driver.window_handles[0])
2181                        VerChatWhatsApp()
2182                    elif formato == "TELEGRAM":
2183                        driver.set_window_size(1100, 720)
2184
driver.switch_to.window(driver.window_handles[1])
2185                        VerChatTelegram()
2186                 time.sleep(0.01)
2187                 x=x+10
2188
2189     def SeleccionarChatWhatsApp(ContactoP):
2190         global Lista_Contactos
2191         global CuadriculaContactos
2192         global formato
2193         Fuente = pygame.font.Font(None, 170)
2194         i = 0
2195         if isinstance(ContactoP, int) == True:
2196             for key in Lista_Contactos:
2197                 if ContactoP == i:
2198                     Contacto = key
2199                 i = i + 1
2200             if  ContactoP >= len(Lista_Contactos):
2201                     driver.minimize_window()
2202                     screen.fill(Negro)
2203                     PrimerParametroA = Fuente.render("Selección" ,1,
(Blanco))
2204                     PrimerParametroB = Fuente.render("inexistente"
,1, (Blanco))
2205                     screen.blit(PrimerParametroA, (370, 190))
2206                     screen.blit(PrimerParametroB, (340, 390))
2207                     pygame.display.flip()
2208                     time.sleep(3)
2209                     SeleccionContactos()
2210         else:
2211             Contacto = ContactoP
2212         elements = driver.find_elements(By.TAG_NAME, 'span')
2213         for element in elements:
2214             if element.text == Contacto:
```

```
2215                 element.click()
2216                 return
2217         driver.minimize_window()
2218         screen.fill(Negro)
2219         PrimerParametroC = Fuente.render("Chat no" ,1, (Blanco))
2220         PrimerParametroD = Fuente.render("encontrado" ,1, (Blanco))
2221         screen.blit(PrimerParametroC, (400, 190))
2222         screen.blit(PrimerParametroD, (360, 390))
2223         pygame.display.flip()
2224         time.sleep(3)
2225         SeleccionContactos()
2226 
2227     def EnviarMensajeWhatsApp():
2228         global frase
2229         element = driver.find_element(By.XPATH,
'//*[@id="main"]/footer/div[1]/div/span[2]/div/div[2]/div[2]/button/span'
)
2230         element.click()
2231         time.sleep(1)
2232         if PlantillaSocial == "Activada":
2233             frase = ""
2234             VerChatWhatsApp()
2235         else:
2236             driver.minimize_window()
2237             interfazSocial()
2238 
2239     def IncluirMensajeWhatsApp(Mensaje):
2240         ChatBox = driver.find_element(By.XPATH,
'//*[@id="main"]/footer/div[1]/div/span[2]/div/div[2]/div[1]/div/div[2]')
2241         ChatBox.send_keys(Mensaje)
2242         time.sleep(1)
2243         EnviarMensajeWhatsApp()
2244 
2245     def VerChatWhatsApp():
2246         global formato
2247         global PlantillaSocial
2248         k = 0
2249         while True:
2250             screen.fill(ColorDeFondo)
2251             screen.blit(ChatSocial, (0, 0))
2252             if k == 0:
2253                 screen.blit(CuadritoPeque, (-20, 20))
2254             elif k == 1:
2255                 screen.blit(CuadritoPeque, (-20, -500))
2256             elif k == 2:
2257                 screen.blit(CuadritoPeque, (-20, -390))
2258             elif k == 3:
2259                 screen.blit(CuadritoPeque, (-20, -280))
2260             elif k == 4:
2261                 screen.blit(CuadritoPeque, (-20, -170))
2262             elif k == 5:
2263                 screen.blit(CuadritoPeque, (-20, -70))
2264             pygame.display.flip()
2265             x=0
2266             while x <= VelocidadCursor:
2267                 if GPIO.input(21) == GPIO.HIGH:
2268                     sonidoSeleccion()
```

```
2269                    if k == 1:
2270
driver.execute_script("document.querySelector('._33LGR').scrollBy(0,-
50)")
2271                    elif k == 2:
2272
driver.execute_script("document.querySelector('._33LGR').scrollBy(0,50)")
2273                    elif k == 3:
2274                        try:
2275
driver.execute_script("document.querySelector('._3K42l').click()")
2276                        except:
2277                            None
2278                    elif k == 4:
2279                        driver.minimize_window()
2280                        SeleccionInterfazRes()
2281                    elif k == 5:
2282                        formato = ""
2283                        PlantillaSocial = "Desactivada"
2284                        driver.minimize_window()
2285                        Selecciondeinterfaz()
2286                    else:
2287                        driver.minimize_window()
2288                        SeleccionContactos()
2289                time.sleep(0.01)
2290                x=x+10
2291            k = k + 1
2292            if k >= 6:
2293                k = 0
2294
2295    def SeleccionarChatTelegram(ContactoP):
2296        global Lista_Contactos
2297        global CuadriculaContactos
2298        global formato
2299        global UltimaSeleccion
2300        Fuente = pygame.font.Font(None, 170)
2301        i = 0
2302        if isinstance(ContactoP, int) == True:
2303            for key in Lista_Contactos:
2304                if ContactoP == i:
2305                    Contacto = key
2306                i = i + 1
2307            if  ContactoP >= len(Lista_Contactos):
2308                    driver.minimize_window()
2309                    screen.fill(Negro)
2310                    PrimerParametroA = Fuente.render("Selección" ,1,
(Blanco))
2311                    PrimerParametroB = Fuente.render("inexistente"
,1, (Blanco))
2312                    screen.blit(PrimerParametroA, (370, 190))
2313                    screen.blit(PrimerParametroB, (340, 390))
2314                    pygame.display.flip()
2315                    time.sleep(3)
2316                    SeleccionContactos()
2317        else:
2318            Contacto = ContactoP
2319        if Contacto == UltimaSeleccion:
```

```
2320            return
2321        ChatPane = driver.find_element(By.XPATH, '//*[@id="folders-
container"]/div')
2322        elements = ChatPane.find_elements(By.TAG_NAME, 'span')
2323        time.sleep(1)
2324        for element in elements:
2325            if element.text == Contacto:
2326                ChatSeleccionado =
element.find_element(By.CLASS_NAME, 'peer-title')
2327                Chat_ID = ChatSeleccionado.get_attribute('data-peer-
id')
2328                Chat_Click = driver.find_element(By.XPATH,
'//li[@data-peer-id="' + Chat_ID + '"]')
2329                Chat_Click.click()
2330                ChatBox = driver.find_element(By.XPATH,
'//*[@id="column-
center"]/div/div/div[4]/div/div[1]/div/div[8]/div[1]/div[1]')
2331                ChatBox.send_keys(".")
2332                ChatBox.send_keys(Keys.BACK_SPACE)
2333                UltimaSeleccion = Contacto
2334                return
2335        driver.minimize_window()
2336        screen.fill(Negro)
2337        PrimerParametroC = Fuente.render("Chat no" ,1, (Blanco))
2338        PrimerParametroD = Fuente.render("encontrado" ,1, (Blanco))
2339        screen.blit(PrimerParametroC, (400, 190))
2340        screen.blit(PrimerParametroD, (360, 390))
2341        pygame.display.flip()
2342        time.sleep(3)
2343        SeleccionContactos()
2344
2345    def EnviarMensajeTelegram():
2346        global PlantillaSocial
2347        global frase
2348        element = driver.find_element(By.XPATH, '//*[@id="column-
center"]/div/div/div[4]/div/div[5]/button')
2349        element.click()
2350        time.sleep(1)
2351        if PlantillaSocial == "Activada":
2352            frase = ""
2353            VerChatTelegram()
2354        else:
2355            driver.minimize_window()
2356            interfazSocial()
2357
2358    def IncluirMensajeTelegram(Mensaje):
2359        ChatBox = driver.find_element(By.XPATH, '//*[@id="column-
center"]/div/div/div[4]/div/div[1]/div/div[8]/div[2]/div[1]')
2360        ChatBox.click()
2361        ChatBox.send_keys(Mensaje)
2362        time.sleep(1)
2363        EnviarMensajeTelegram()
2364
2365    def VerChatTelegram():
2366        global formato
2367        global PlantillaSocial
2368        k = 0
```

```
2369        while True:
2370            screen.fill(ColorDeFondo)
2371            screen.blit(ChatSocial, (0, 0))
2372            if k == 0:
2373                screen.blit(CuadritoPeque, (-20, 20))
2374            elif k == 1:
2375                screen.blit(CuadritoPeque, (-20, -500))
2376            elif k == 2:
2377                screen.blit(CuadritoPeque, (-20, -390))
2378            elif k == 3:
2379                screen.blit(CuadritoPeque, (-20, -280))
2380            elif k == 4:
2381                screen.blit(CuadritoPeque, (-20, -170))
2382            elif k == 5:
2383                screen.blit(CuadritoPeque, (-20, -70))
2384            pygame.display.flip()
2385            x=0
2386            while x <= VelocidadCursor:
2387                if GPIO.input(21) == GPIO.HIGH:
2388                    sonidoSeleccion()
2389                    if k == 1:
2390
driver.execute_script("document.querySelectorAll('.scrollable.scrollable-
y')[1].scrollBy(0,-50)")
2391                    elif k == 2:
2392
driver.execute_script("document.querySelectorAll('.scrollable.scrollable-
y')[1].scrollBy(0,50)")
2393                    elif k == 3:
2394                        try:
2395
driver.execute_script("document.querySelector('.btn-circle.btn-corner.z-
depth-1.bubbles-corner-button.bubbles-go-down.tgico-
arrow_down.rp').click()")
2396                        except:
2397                            None
2398                    elif k == 4:
2399                        driver.minimize_window()
2400                        SeleccionInterfazRes()
2401                    elif k == 5:
2402                        formato = ""
2403                        PlantillaSocial = "Desactivada"
2404                        driver.minimize_window()
2405                        Selecciondeinterfaz()
2406                    else:
2407                        driver.minimize_window()
2408                        SeleccionContactos()
2409                time.sleep(0.01)
2410                x=x+10
2411            k = k + 1
2412            if k >= 6:
2413                k = 0
2414
2415#########################################################################
2416
2417    def interfazFila6():
2418        while True:
```

```
2419                global frase
2420                global prefrase
2421                global PlantillaAlfabetica
2422                global Prediccion1
2423                global Prediccion2
2424                global Prediccion3
2425                global UltimaFraseIngresada
2426                global CuadriculaContactos
2427                global ContactoElegido
2428                RevisarMensajesNoLeidos()
2429                EstadoRedGSM = EstadoConexion()
2430                previsualizacion(frase,CuaVoz,-35,0,EstadoRedGSM)
2431                x=0
2432                while x <= VelocidadCursor:
2433                    if GPIO.input(21) == GPIO.HIGH:
2434                       database.ActualizacionDatos()
2435                       voz(frase)
2436                       RepetirFrase(interfazAlfabetica)
2437                    time.sleep(0.01)
2438                    x=x+10
2439                previsualizacion("Enviar",CursorABCEnviar,0,-
13,EstadoRedGSM)
2440                x=0
2441                while x <= VelocidadCursor:
2442                    if GPIO.input(21) == GPIO.HIGH:
2443                       voz(frase)
2444                       if PlantillaSocial == "Activada":
2445                           if formato == "SMS":
2446                               screen.fill(ColorDeFondo)
2447                               EnviarMensajeSMS(ContactoElegido, frase)
2448                           elif formato == "WHATSAPP":
2449                               driver.set_window_size(1100, 720)
2450
driver.switch_to.window(driver.window_handles[0])
2451                               IncluirMensajeWhatsApp(frase)
2452                           elif formato == "TELEGRAM":
2453                               driver.set_window_size(1100, 720)
2454
driver.switch_to.window(driver.window_handles[1])
2455                               IncluirMensajeTelegram(frase)
2456                       else:
2457                           CuadriculaContactos = "Enviar"
2458                           interfazSocial()
2459                    time.sleep(0.01)
2460                    x=x+10
2461                previsualizacion("Borrar todo",CuaNuevaFrase,-
313,0,Esta oRedGSM)
2462                x=0
2463                while x <= VelocidadCursor:
2464                    if GPIO.input(21) == GPIO.HIGH:
2465                       sonidoSeleccion()
2466                       frase=""
2467                       Prediccion1=""
2468                       Prediccion2=""
2469                       Prediccion3=""
2470                       UltimaFraseIngresada=""
2471                       interfazAlfabetica()
```

```
2472                time.sleep(0.01)
2473                x=x+10
2474            previsualizacion(frase,CuaNumero,465,0,EstadoRedGSM)
2475            TiempoEspera(interfazNumerica)
2476            previsualizacion(frase,CuaInicio,0,15,EstadoRedGSM)
2477            TiempoEspera(interfazAlfabetica)
2478
2479#####################################################################
2480
2481    def interfazFila3Num():
2482        while True:
2483            global frase
2484            global prefrase
2485            global PlantillaNumerica
2486            global CuadriculaContactos
2487            RevisarMensajesNoLeidos()
2488            EstadoRedGSM = EstadoConexion()
2489            PlantillaNumerica="Activa"
2490            previsualizacionNumerica(frase,CuaVoz,43,0,EstadoRedGSM)
2491            x=0
2492            while x <= VelocidadCursor:
2493                if GPIO.input(21) == GPIO.HIGH:
2494                   voz(frase)
2495                   PlantillaNumerica="Desactivada"
2496                   RepetirFrase(interfazNumerica)
2497                time.sleep(0.01)
2498                x=x+10
2499
previsualizacionNumerica(frase,CuaEspacio,6,0,EstadoRedGSM)
2500            x=0
2501            while x <= VelocidadCursor:
2502                if GPIO.input(21) == GPIO.HIGH:
2503                   sonidoSeleccion()
2504                   frase=frase + " "
2505                   interfazNumerica()
2506                time.sleep(0.01)
2507                x=x+10
2508            preborrar(frase)
2509
previsualizacionNumerica(prefrase,CuaBorrar,5,0,EstadoRedGSM)
2510            x=0
2511            while x <= VelocidadCursor:
2512                if GPIO.input(21) == GPIO.HIGH:
2513                   sonidoSeleccion()
2514                   borrar(frase)
2515                   interfazNumerica()
2516                time.sleep(0.01)
2517                x=x+10
2518            previsualizacionNumerica("Borrar todo",CuaNuevaFrase,-
25,0,EstadoRedGSM)
2519            x=0
2520            while x <= VelocidadCursor:
2521                if GPIO.input(21) == GPIO.HIGH:
2522                   sonidoSeleccion()
2523                   frase=""
2524                   interfazNumerica()
2525                time.sleep(0.01)
```

```
                x=x+10

previsualizacionNumerica("Enviar",CuaEspacio,725,0,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                   voz(frase)
                   if PlantillaSocial == "Activada":
                       if formato == "SMS":
                           screen.fill(ColorDeFondo)
                           EnviarMensajeSMS(ContactoElegido, frase)
                       elif formato == "WHATSAPP":
                           driver.set_window_size(1100, 720)

driver.switch_to.window(driver.window_handles[0])
                           IncluirMensajeWhatsApp(frase)
                       elif formato == "TELEGRAM":
                           driver.set_window_size(1100, 720)

driver.switch_to.window(driver.window_handles[1])
                           IncluirMensajeTelegram(frase)
                   else:
                       CuadriculaContactos = "Enviar"
                       interfazSocial()
                time.sleep(0.01)
                x=x+10

previsualizacionNumerica(frase,CuaInicio,0,0,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor:
                if GPIO.input(21) == GPIO.HIGH:
                   sonidoSeleccion()
                   PlantillaNumerica="Desactivada"
                   interfazNumerica()
                time.sleep(0.01)
                x=x+10

#####################################################################

    def interfazSocial():
        while True:
            global PlantillaPictografica
            global PlantillaAlfabetica
            global PlantillaNumerica
            global PlantillaSocial
            global SeleccionPlantillaPictografica
            global formato
            global CuadriculaContactos
            RevisarMensajesNoLeidos()
            EstadoRedGSM = EstadoConexion()
            seleccion(OpcionesSocial,CuaAlfa,0,0,EstadoRedGSM)
            x=0
            while x <= VelocidadCursor :
                if GPIO.input(21) == GPIO.HIGH:
                   sonidoSeleccion()
                   formato = "SMS"
                   SeleccionContactos()
```

```
2579                time.sleep(0.01)
2580                x=x+10
2581            seleccion(OpcionesSocial,CuaAlfa,427,0,EstadoRedGSM)
2582            x=0
2583            while x <= VelocidadCursor:
2584                if GPIO.input(21) == GPIO.HIGH:
2585                   sonidoSeleccion()
2586                   formato = "WHATSAPP"
2587                   SeleccionContactos()
2588                time.sleep(0.01)
2589                x=x+10
2590            seleccion(OpcionesSocial,CuaAlfa,850,0,EstadoRedGSM)
2591            x=0
2592            while x <= VelocidadCursor:
2593                if GPIO.input(21) == GPIO.HIGH:
2594                   sonidoSeleccion()
2595                   formato = "TELEGRAM"
2596                   SeleccionContactos()
2597                time.sleep(0.01)
2598                x=x+10
2599            seleccion(OpcionesSocial,CuaRegresar,40,-15,EstadoRedGSM)
2600            x=0
2601            while x <= VelocidadCursor:
2602                if GPIO.input(21) == GPIO.HIGH:
2603                   sonidoSeleccion()
2604                   CuadriculaContactos = ""
2605                   if PlantillaSocial=="Activada":
2606                        Selecciondeinterfaz()
2607                   if PlantillaPictografica == "Activada":
2608                        if SeleccionPlantillaPictografica ==
"PlantillaHospital":
2609                            interfazGrafica()
2610                        else:
2611                            interfazGraficaCasa()
2612                   if PlantillaAlfabetica == "Activada":
2613                        interfazAlfabetica()
2614                   else:
2615                        interfazNumerica()
2616                time.sleep(0.01)
2617                x=x+10
2618
2619    def AgendaContactos():
2620        global Lista_Contactos
2621        Fuente = pygame.font.Font(None, 50)
2622        Fuente2 = pygame.font.Font(None, 75)
2623        screen.fill(ColorDeFondo)
2624        screen.blit(PictoRegreso, (0, 0))
2625        BotonInicio = Fuente.render("Volver", 1, (Negro))
2626        BotonMP = Fuente.render("Menú principal", 1, (Negro))
2627        screen.blit(BotonInicio, (1140, 640))
2628        screen.blit(BotonMP, (500, 640))
2629        i = 0
2630        j = 0
2631        for key in Lista_Contactos:
2632            if i > 2:
2633                j = j + 1
2634                i = 0
```

```
2635            N = key
2636            T = Lista_Contactos[key]
2637            Nombre = Fuente.render(N,1,(Negro))
2638            Telefono = Fuente.render(T,1,(Negro))
2639            screen.blit(Nombre, (40+(425*i), 25+(115*j)))
2640            screen.blit(Telefono, (40+(425*i), 75+(115*j)))
2641            if Mensajes_Leidos.get(N) == False:
2642                pygame.draw.circle(screen,Negro,(368+(425*i),
86+(115*j)),25)
2643                Notificacion = Fuente2.render("!",1,(Blanco))
2644                screen.blit(Notificacion, (358+(425*i), 60+(115*j)))
2645            i = i + 1
2646
2647    def SeleccionContactos():
2648        global formato
2649        global CuadriculaContactos
2650        global PlantillaSocial
2651        global frase
2652        RevisarMensajesNoLeidos()
2653        k = 0
2654        while True:
2655            AgendaContactos()
2656            if k == 5:
2657                screen.blit(CuaNuevaFrase, (-325,90))
2658                pygame.display.flip()
2659                x=0
2660                while x <= VelocidadCursor:
2661                    if GPIO.input(21) == GPIO.HIGH:
2662                        sonidoSeleccion()
2663                        formato = ""
2664                        PlantillaSocial = "Desactivada"
2665                        CuadriculaContactos = ""
2666                        Selecciondeinterfaz()
2667                    time.sleep(0.01)
2668                    x=x+10
2669                k = k + 1
2670            elif k >= 6:
2671                k = 0
2672                screen.blit(CuadritoPeque, (0,0))
2673                pygame.display.flip()
2674                x=0
2675                while x <= VelocidadCursor:
2676                    if GPIO.input(21) == GPIO.HIGH:
2677                        sonidoSeleccion()
2678                        if PlantillaSocial == "Activada":
2679                            interfazSocial()
2680                        if CuadriculaContactos == "Enviar":
2681                            formato = ""
2682                            interfazSocial()
2683                        elif CuadriculaContactos == "Agregar" or
CuadriculaContactos == "Prueba":
2684                            interfazConfiguracionSocial()
2685                    time.sleep(0.01)
2686                    x=x+10
2687            else:
2688                screen.blit(CursorFilaContactos, (0, 0+(115*k)))
2689                pygame.display.flip()
```

```
                k = k + 1
                x=0
                while x <= VelocidadCursor:
                    if GPIO.input(21) == GPIO.HIGH:
                        sonidoSeleccion()
                        frase = frase.capitalize()
                        if k == 1:
                            interfazFila1Contactos()
                        if k == 2:
                            interfazFila2Contactos()
                        if k == 3:
                            interfazFila3Contactos()
                        if k == 4:
                            interfazFila4Contactos()
                        if k == 5:
                            interfazFila5Contactos()
                    time.sleep(0.01)
                    x=x+10

    def interfazFila1Contactos():
        global frase
        global formato
        k = 0
        while True:
            AgendaContactos()
            if k >= 3:
                k = 0
                screen.blit(CuaInicio, (0,0))
                pygame.display.flip()
                x=0
                while x <= VelocidadCursor:
                    if GPIO.input(21) == GPIO.HIGH:
                        sonidoSeleccion()
                        SeleccionContactos()
                    time.sleep(0.01)
                    x=x+10
            else:
                screen.blit(CursorColumnaContactos, (0+(425*k), 0))
                pygame.display.flip()
                k = k + 1
                x=0
                while x <= VelocidadCursor:
                    if GPIO.input(21) == GPIO.HIGH:
                        sonidoSeleccion()
                        if k == 1:
                            ContactoPos = 0
                        if k == 2:
                            ContactoPos = 1
                        if k == 3:
                            ContactoPos = 2
                        if PlantillaSocial == "Activada":
                            if formato == "SMS":
                                VerChatSMS(ContactoPos)
                            elif formato == "WHATSAPP":
                                driver.set_window_size(1100, 720)

driver.switch_to.window(driver.window_handles[0])
```

```
                                SeleccionarChatWhatsApp(ContactoPos)
                                VerChatWhatsApp()
                            elif formato == "TELEGRAM":
                                driver.set_window_size(1100, 720)

driver.switch_to.window(driver.window_handles[1])
                                MismaSeleccion =
SeleccionarChatTelegram(ContactoPos)
                                VerChatTelegram()
                        if CuadriculaContactos == "Enviar":
                            if formato == "SMS":
                                EnviarMensajeSMS(ContactoPos, frase)
                            elif formato == "WHATSAPP":
                                driver.set_window_size(1100, 720)

driver.switch_to.window(driver.window_handles[0])
                                SeleccionarChatWhatsApp(ContactoPos)
                                IncluirMensajeWhatsApp(frase)
                            elif formato == "TELEGRAM":
                                driver.set_window_size(1100, 720)

driver.switch_to.window(driver.window_handles[1])
                                SeleccionarChatTelegram(ContactoPos)
                                IncluirMensajeTelegram(frase)
                        elif CuadriculaContactos == "Agregar":
                            interfazAgregarContacto(ContactoPos)
                        elif CuadriculaContactos == "Prueba":
                            interfazPruebaSMS(ContactoPos)
                    time.sleep(0.01)
                    x=x+10

    def interfazFila2Contactos():
        global frase
        global formato
        k = 0
        while True:
            AgendaContactos()
            if k >= 3:
                k = 0
                screen.blit(CuaInicio, (0,0))
                pygame.display.flip()
                x=0
                while x <= VelocidadCursor:
                    if GPIO.input(21) == GPIO.HIGH:
                        sonidoSeleccion()
                        SeleccionContactos()
                    time.sleep(0.01)
                    x=x+10
            else:
                screen.blit(CursorColumnaContactos, (0+(425*k), 115))
                pygame.display.flip()
                k = k + 1
                x=0
                while x <= VelocidadCursor:
                    if GPIO.input(21) == GPIO.HIGH:
                        sonidoSeleccion()
                        M = frase
```

```
2799                        if k == 1:
2800                            ContactoPos = 3
2801                        if k == 2:
2802                            ContactoPos = 4
2803                        if k == 3:
2804                            ContactoPos = 5
2805                        if PlantillaSocial == "Activada":
2806                            if formato == "SMS":
2807                                VerChatSMS(ContactoPos)
2808                            elif formato == "WHATSAPP":
2809                                driver.set_window_size(1100, 720)
2810
driver.switch_to.window(driver.window_handles[0])
2811                                SeleccionarChatWhatsApp(ContactoPos)
2812                                VerChatWhatsApp()
2813                            elif formato == "TELEGRAM":
2814                                driver.set_window_size(1100, 720)
2815
driver.switch_to.window(driver.window_handles[1])
2816                                MismaSeleccion =
SeleccionarChatTelegram(ContactoPos)
2817                                VerChatTelegram()
2818                        if CuadriculaContactos == "Enviar":
2819                            if formato == "SMS":
2820                                EnviarMensajeSMS(ContactoPos, frase)
2821                            elif formato == "WHATSAPP":
2822                                driver.set_window_size(1100, 720)
2823
driver.switch_to.window(driver.window_handles[0])
2824                                SeleccionarChatWhatsApp(ContactoPos)
2825                                IncluirMensajeWhatsApp(frase)
2826                            elif formato == "TELEGRAM":
2827                                driver.set_window_size(1100, 720)
2828
driver.switch_to.window(driver.window_handles[1])
2829                                SeleccionarChatTelegram(ContactoPos)
2830                                IncluirMensajeTelegram(frase)
2831                        elif CuadriculaContactos == "Agregar":
2832                            interfazAgregarContacto(ContactoPos)
2833                        elif CuadriculaContactos == "Prueba":
2834                            interfazPruebaSMS(ContactoPos)
2835                    time.sleep(0.01)
2836                    x=x+10
2837
2838    def interfazFila3Contactos():
2839        global frase
2840        global formato
2841        k = 0
2842        while True:
2843            AgendaContactos()
2844            if k >= 3:
2845                k = 0
2846                screen.blit(CuaInicio, (0,0))
2847                pygame.display.flip()
2848                x=0
2849                while x <= VelocidadCursor:
2850                    if GPIO.input(21) == GPIO.HIGH:
```

```
2851                        sonidoSeleccion()
2852                        SeleccionContactos()
2853                    time.sleep(0.01)
2854                    x=x+10
2855            else:
2856                screen.blit(CursorColumnaContactos, (0+(425*k), 230))
2857                pygame.display.flip()
2858                k = k + 1
2859                x=0
2860                while x <= VelocidadCursor:
2861                    if GPIO.input(21) == GPIO.HIGH:
2862                        sonidoSeleccion()
2863                        M = frase
2864                        if k == 1:
2865                            ContactoPos = 6
2866                        if k == 2:
2867                            ContactoPos = 7
2868                        if k == 3:
2869                            ContactoPos = 8
2870                        if PlantillaSocial == "Activada":
2871                            if formato == "SMS":
2872                                VerChatSMS(ContactoPos)
2873                            elif formato == "WHATSAPP":
2874                                driver.set_window_size(1100, 720)
2875
driver.switch_to.window(driver.window_handles[0])
2876                                SeleccionarChatWhatsApp(ContactoPos)
2877                                VerChatWhatsApp()
2878                            elif formato == "TELEGRAM":
2879                                driver.set_window_size(1100, 720)
2880
driver.switch_to.window(driver.window_handles[1])
2881                                MismaSeleccion =
SeleccionarChatTelegram(ContactoPos)
2882                                VerChatTelegram()
2883                        if CuadriculaContactos == "Enviar":
2884                            if formato == "SMS":
2885                                EnviarMensajeSMS(ContactoPos, frase)
2886                            elif formato == "WHATSAPP":
2887                                driver.set_window_size(1100, 720)
2888
driver.switch_to.window(driver.window_handles[0])
2889                                SeleccionarChatWhatsApp(ContactoPos)
2890                                IncluirMensajeWhatsApp(frase)
2891                            elif formato == "TELEGRAM":
2892                                driver.set_window_size(1100, 720)
2893
driver.switch_ o.window(driver.window_handles[1])
2894                                SeleccionarChatTelegram(ContactoPos)
2895                                IncluirMensajeTelegram(frase)
2896                        elif CuadriculaContactos == "Agregar":
2897                            interfazAgregarContacto(ContactoPos)
2898                        elif CuadriculaContactos == "Prueba":
2899                            interfazPruebaSMS(ContactoPos)
2900                    time.sleep(0.01)
2901                    x=x+10
2902
```

```
2903    def interfazFila4Contactos():
2904        global frase
2905        global formato
2906        k = 0
2907        while True:
2908            AgendaContactos()
2909            if k >= 3:
2910                k = 0
2911                screen.blit(CuaInicio, (0,0))
2912                pygame.display.flip()
2913                x=0
2914                while x <= VelocidadCursor:
2915                    if GPIO.input(21) == GPIO.HIGH:
2916                       sonidoSeleccion()
2917                       SeleccionContactos()
2918                    time.sleep(0.01)
2919                    x=x+10
2920            else:
2921                screen.blit(CursorColumnaContactos, (0+(425*k), 345))
2922                pygame.display.flip()
2923                k = k + 1
2924                x=0
2925                while x <= VelocidadCursor:
2926                    if GPIO.input(21) == GPIO.HIGH:
2927                       sonidoSeleccion()
2928                       M = frase
2929                       if k == 1:
2930                          ContactoPos = 9
2931                       if k == 2:
2932                          ContactoPos = 10
2933                       if k == 3:
2934                          ContactoPos = 11
2935                       if PlantillaSocial == "Activada":
2936                           if formato == "SMS":
2937                              VerChatSMS(ContactoPos)
2938                           elif formato == "WHATSAPP":
2939                              driver.set_window_size(1100, 720)
2940
driver.switch_to.window(driver.window_handles[0])
2941                              SeleccionarChatWhatsApp(ContactoPos)
2942                              VerChatWhatsApp()
2943                           elif formato == "TELEGRAM":
2944                              driver.set_window_size(1100, 720)
2945
driver.switch_to.window(driver.window_handles[1])
2946                              MismaSeleccion =
SeleccionarChatTelegram(ContactoPos)
2947                              VerChatTelegram()
2948                       if CuadriculaContactos == "Enviar":
2949                          if formato == "SMS":
2950                              EnviarMensajeSMS(ContactoPos, frase)
2951                          elif formato == "WHATSAPP":
2952                              driver.set_window_size(1100, 720)
2953
driver.switch_to.window(driver.window_handles[0])
2954                              SeleccionarChatWhatsApp(ContactoPos)
2955                              IncluirMensajeWhatsApp(frase)
```

```
                            elif formato == "TELEGRAM":
                                driver.set_window_size(1100, 720)

driver.switch_to.window(driver.window_handles[1])
                                SeleccionarChatTelegram(ContactoPos)
                                IncluirMensajeTelegram(frase)
                        elif CuadriculaContactos == "Agregar":
                            interfazAgregarContacto(ContactoPos)
                        elif CuadriculaContactos == "Prueba":
                            interfazPruebaSMS(ContactoPos)
                    time.sleep(0.01)
                    x=x+10

    def interfazFila5Contactos():
        global frase
        global formato
        k = 0
        while True:
            AgendaContactos()
            if k >= 3:
                k = 0
                screen.blit(CuaInicio, (0,0))
                pygame.display.flip()
                x=0
                while x <= VelocidadCursor:
                    if GPIO.input(21) == GPIO.HIGH:
                        sonidoSeleccion()
                        SeleccionContactos()
                    time.sleep(0.01)
                    x=x+10
            else:
                screen.blit(CursorColumnaContactos, (0+(425*k), 460))
                pygame.display.flip()
                k = k + 1
                x=0
                while x <= VelocidadCursor:
                    if GPIO.input(21) == GPIO.HIGH:
                        sonidoSeleccion()
                        M = frase
                        if k == 1:
                            ContactoPos = 12
                        if k == 2:
                            ContactoPos = 13
                        if k == 3:
                            ContactoPos = 14
                        if PlantillaSocial == "Activada":
                            if formato == "SMS":
                                VerChatSMS(ContactoPos)
                            elif formato == "WHATSAPP":
                                driver.set_window_size(1100, 720)

driver.switch_to.window(driver.window_handles[0])
                                SeleccionarChatWhatsApp(ContactoPos)
                                VerChatWhatsApp()
                            elif formato == "TELEGRAM":
                                driver.set_window_size(1100, 720)

```

```
driver.switch_to.window(driver.window_handles[1])
                                MismaSeleccion =
SeleccionarChatTelegram(ContactoPos)
                                VerChatTelegram()
                        if CuadriculaContactos == "Enviar":
                            if formato == "SMS":
                                EnviarMensajeSMS(ContactoPos, frase)
                            elif formato == "WHATSAPP":
                                driver.set_window_size(1100, 720)

driver.switch_to.window(driver.window_handles[0])
                                SeleccionarChatWhatsApp(ContactoPos)
                                IncluirMensajeWhatsApp(frase)
                            elif formato == "TELEGRAM":
                                driver.set_window_size(1100, 720)

driver.switch_to.window(driver.window_handles[1])
                                SeleccionarChatTelegram(ContactoPos)
                                IncluirMensajeTelegram(frase)
                        elif CuadriculaContactos == "Agregar":
                            interfazAgregarContacto(ContactoPos)
                        elif CuadriculaContactos == "Prueba":
                            interfazPruebaSMS(ContactoPos)
                    time.sleep(0.01)
                    x=x+10

######################################################################

    def RepetirFrase(plantillaAnterior):
        while True:
            RevisarMensajesNoLeidos()
            global frase
            global Tono
            global Idioma
            global PlantillaAlfabetica
            global PlantillaNumerica
            global PlantillaPictografica
            global Prediccion1
            global Prediccion2
            global Prediccion3
            global CuadriculaContactos
            global ContactoElegido
            if frase!="":
                Repetir(CuaRepetirFrase,0,0)
                x=0
                while x <= VelocidadCursor:
                    if GPIO.input(21) == GPIO.HIGH:
                        sonidoSeleccion()
                        union = ("espeak "+ Idioma + Tono + " '" +
frase  + "' --stdout|aplay")
                        os.system(union)
                        pygame.time.delay(VelocidadCursor)
                        RepetirFrase(plantillaAnterior)
                    time.sleep(0.01)
                    x=x+10
                Repetir(CursorRepEnviar,0,0)
                x=0
```

```
3063                 while x <= VelocidadCursor:
3064                     if GPIO.input(21) == GPIO.HIGH:
3065                        sonidoSeleccion()
3066                        if PlantillaSocial == "Activada":
3067                            if formato == "SMS":
3068                                EnviarMensajeSMS(ContactoElegido,
frase)
3069                            elif formato == "WHATSAPP":
3070                                driver.set_window_size(1100, 720)
3071
driver.switch_to.window(driver.window_handles[0])
3072                                IncluirMensajeWhatsApp(frase)
3073                            elif formato == "TELEGRAM":
3074                                driver.set_window_size(1100, 720)
3075
driver.switch_to.window(driver.window_handles[1])
3076                                IncluirMensajeTelegram(frase)
3077                        else:
3078                            CuadriculaContactos = "Enviar"
3079                            interfazSocial()
3080                     time.sleep(0.01)
3081                     x=x+10
3082                 Repetir(CuaRepetirFraseInicio,796,0)
3083                 x=0
3084                 while x <= VelocidadCursor:
3085                     if GPIO.input(21) == GPIO.HIGH:
3086                        sonidoSeleccion()
3087                        frase=""
3088                        Prediccion1=""
3089                        Prediccion2=""
3090                        Prediccion3=""
3091                        if PlantillaPictografica == "Activada":
3092                           if SeleccionPlantillaPictografica ==
"PlantillaHospital":
3093                               interfazGrafica()
3094                           else:
3095                               interfazGraficaCasa()
3096
3097                        elif PlantillaNumerica == "Activada":
3098                             interfazNumerica()
3099                        else:
3100                            interfazAlfabetica()
3101                     time.sleep(0.01)
3102                     x=x+10
3103                 Repetir(CuaRegresar,3,6)
3104                 TiempoEspera(plantillaAnterior)
3105                 RepetirFrase(plantillaAnterior)
3106             else:
3107                 None
3108
3109#####################################################################
3110
3111     database = DataBase()
3112     database.ConsultaListaContactos()
3113     database.ConsultaHistorialMensajes()
3114     database.CerrarBD()
3115     ComunicacionModulo()
```

```
3116    CargarParametrosSocial()
3117    CargarWhatsAppTelegram()
3118    Selecciondeinterfaz()
```