



INSTITUTO POLITÉCNICO NACIONAL ESCUELA SUPERIOR DE CÓMPUTO

ESCOM

Trabajo de titulación

“Sistema de Monitoreo para Casa Habitación”

Para cumplir con la opción de titulación de trabajo terminal en la carrera de

“Ingeniería en Sistemas Computacionales”

Presentan

**Hernández López Alejandro
Rosas Santillán Leonardo de Jesús
Zamora Marín René**

Directores

M. en C. Axel Ernesto Moreno Cervantes Dr. José Félix Serrano Talamantes





INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO



No. de Registro: 2014 –B030

Sistema de Monitoreo para Casa Habitación (SiMC)

TT 2014 –B030

Autores:

Hernández López Alejandro

Rosas Santillán Leonardo de Jesús

Zamora Marín René

Este documento presenta la propuesta de un sistema llamado SiMC, que está configurado para monitorear y comunicar las acciones de agentes externos, permitiendo la comunicación con un dispositivo móvil con el fin de realizar un monitoreo remoto para aumentar el índice de seguridad de la casa habitación, por medio de la aplicación de mensajería instantánea Telegram, teniendo comunicación con una tarjeta Raspberry, que se encargara de toma de fotos y detección de movimiento, así también como la adaptación de un botón que funciona como timbre o micrófono, según se elija su funcionamiento.

Palabras clave – Sistema Embebido, Telegram, Instrumentación, Dispositivos Móviles, Monitoreo a casa-habitación.

Directores:

M. en C. Moreno Cervantes Axel Ernesto

Dr. en C. Serrano Talamantes José Félix

Advertencia

“Este documento contiene información desarrollada por la Escuela Superior de Cómputo del Instituto Politécnico Nacional, a partir de datos y documentos con derecho de propiedad y por lo tanto, su uso quedará restringido a las aplicaciones que explícitamente se convengan.”

La aplicación no convenida exime a la escuela su responsabilidad técnica y da lugar a las consecuencias legales que para tal efecto se determinen. Información adicional sobre este reporte técnico podrá obtenerse en:

La Subdirección Académica de la Escuela Superior de Cómputo del Instituto Politécnico Nacional, situada en Av. Juan de Dios Bátiz s/n
Teléfono: 57296000 Extensión: 52000.

*The only way
To do great work
Is to love
What you do
-Steve Jobs*



INSTITUTO POLITECNICO NACIONAL
SUBDIRECCION ACADEMICA
DEPARTAMENTO DE FORMACION INTEGRAL E INSTITUCIONAL



COMISION ACADEMICA DE TRABAJOS TERMINALES

México, D.F a 17 de Diciembre de 2015

DR. FLAVIO ARTUTO SANCHEZ GARFIAS
PRESIDENTE DE LA COMISIÓN ACADÉMICA
DE TRABAJOS TERMINALES
P R E S E N T E

Por medio del presente, informamos que los alumnos que integran el **TRABAJO TERMINAL 2014-B030** titulado **"SIETEMA DE MONITOREO PARA CASA HABITACIÓN"**, concluyeron satisfactoriamente su trabajo.

El empastado del Reporte Técnico Final y el Disco Compacto (CD) fueron revisados ampliamente por sus servidores y corregidos, cubriendo el alcance y el objetivo planteados en el protocolo original y de acuerdo a los requisitos establecidos por la Comisión que usted preside.

ATENTAMENTE

M. en C. Axel Ernesto Moreno Cervantes

Dr. José Félix Serrano Talamantes

Agradecimientos

Terminar este proyecto no fue cosa sencilla y requiere de mucha fuerza y ayuda tanto física como emocional, es por ello que agradezco a todas aquellas personas que estuvieron ahí, demostrando su apoyo y motivación para conmigo.

En primer lugar a mi papá, quien a pesar de todas las adversidades que hemos pasado, me daba ánimos y siempre se preocupaba por cómo iba transcurriendo mi estancia en la escuela y como iba en el proyecto. Y a mi familia que siempre me alentaban a seguir adelante aún cuando pasaba tragos amargos.

A mis amigos, que siempre me dieron y me siguen dando consejos sobre la vida, algo que siempre se va a agradecer, y más cuando los sentimientos se ven involucrados independientemente de lo ocurientes, molestos y graciosos que pueden llegar a ser, en especial a Erik y Leo, sin olvidar a Victor, Rene, José, Alejandro González, Iván Carmona y Fernanda.

A una persona, que si Bien ella no lo sabe, es mi motor para seguir adelante, y dejar plasmado que pase lo que pase nunca quiero perder sus amistad, ya que me da consejos y me hace Reflexionar, incluso cuando quería rendirme y tirar la toalla tanto en la escuela como en el trabajo. En alta estima la tengo y jamás quisiera que eso termine.

Al M. en C. Axel Ernesto Moreno Cervantes y al Dr. en C. José Félix Serrano Talamantes porque fueron los pilares para realizar este proyecto y sin sus conocimientos en el área, no hubiéramos terminado el trabajo a tiempo. Además no sólo por ser nuestros directores, sino también por estar al pendiente de nuestro progreso y darnos su confianza para desarrollar el prototipo.

Por último a una persona que a pesar de que ya no está conmigo, me dió muchos consejos, me enseñó a superar cada obstáculo inconscientemente, a tratar de seguir adelante sin importar lo que pase a mi alrededor y saber cómo se deben cumplir las promesas, aún cuando mis ojos ya no vean y mi corazón ya no sienta. Mamá: en donde quiera que estés te agradezco todo el amor que me diste hasta el último momento y decir que de todas las promesas que hicimos solo resta una... Conocer al amor de vida.

- Alejandro Hernández López

Agradecimientos

Al terminar este trabajo con gran esfuerzo, superando adversidades quiero agradecer de manera especial y sincera a toda mi familia, a mi papa y a mi mama principalmente, ya que gracias a ellos logre cumplir esta gran meta, les agradezco porque siempre confiaron plenamente en mí y me apoyaron incondicionalmente, por guiarme en todo este largo camino, por ayudarme a levantarme cuando caía y no sabía cómo reponerme, por siempre estar ahí cuando los necesitaba, los quiero. Así como también a mis abuelitos que de igual manera siempre me apoyaron en todo momento.

Quiero expresar también mi más sincero agradecimiento a mis amigos Erik y Víctor (Los hornys), Rene, José y Alejandro (Daddy) por todo este tiempo que estuvimos durante la carrera, disfrutando momentos buenos y malos, pero siempre riendo aunque el momento no fuera el adecuado pero siempre viendo el lado divertido de la vida. Puedo decir con felicidad que encontré a unos grandiosos amigos que aunque a veces chingan los quiero y sé que siempre estarán ahí conmigo.

A mis grandes amigos Héctor, Ricardo, Alondra, Sr. Isra, mi queridísimo Javi, Diego y Andy que también tengo sólo palabras de agradecimiento, especialmente por aquellos momentos en los que estuvieron siempre a mi lado aconsejándome, con grandes reflexiones que me ayudaron a nunca perder mi objetivo y no desviarme del camino. Sr. Isra gracias por su apoyo incondicional en sin número de ocasiones sus grandiosos consejos. Richard y Alo gracias por nunca dudar de mí, por alentarme a superarme por ser mi familia. Javi gracias por siempre ayudarme con geniales reflexiones, por ser un gran amigo y saber que siempre puedo contar con usted.

Al Maestro en C. Moreno Cervantes Axel Ernesto y al Dr. en C. Serrano Talamantes José Félix les agradezco por su paciencia, disponibilidad para compartir su experiencia y amplio conocimiento para el desarrollo de este proyecto y sobre todo por nunca dudar de nuestras capacidades para realizar esto. Su colaboración fue de gran ayuda durante mis estancias en la escuela. Le agradezco también por sus siempre atentas y rápidas respuestas a las diferentes inquietudes surgidas durante el desarrollo de este trabajo, lo cual se ha visto también reflejado en los buenos resultados obtenidos. ¡Muchas gracias!

- Leonardo de Jesús Rosas Santillán

Agradecimientos

Agradezco a mis padres René y Estela, que gracias a ellos, me dieron la oportunidad de tener una educación de calidad durante todo el transcurso de mi vida, por inculcarme los valores y hacerme un hombre de bien, por siempre estar alado mío y siempre apoyándome de no desistir de mis acciones y elecciones tomadas.

A todos mis profesores que tuve durante todo mi transcurso en mi carrera académica, que me inculcaron diferentes valores y me enseñaron a siempre dar lo mejor de mí, sometiéndome a diferentes retos, así también como el desarrollo de diferentes habilidades que al final me ayudaron en la formación de un ingeniero en sistemas computacionales. Al M. en C. Axel Ernesto Moreno Cervantes y al Dr. José Félix Serrano Talamantes quienes nos orientaron, apoyaron y supervisaron durante el desarrollo de este trabajo terminal, brindándonos confianza y fe en nosotros para realizar trabajo terminal.

A mis compañeros de este trabajo terminal, Leonardo y Alejandro que gracias a ellos se pudo concluir con este proyecto, y que sin ellos no hubiera sido posible, a pesar de las diferencias que se tuvieron a lo largo de la implementación del trabajo terminal, siempre concluíamos tomando la mejor decisión para el proyecto. Así mismo les doy las gracias por su amistad, por su compañía a lo largo de la carrera y por todos esos momentos que vivimos durante nuestra formación que vivimos.

Por ultimo quiero dar gracias a todos mis amigos por el poyo, sus consejos y los momentos que vivimos a lo largo de esta maravillosa carrera. Doy gracias también a todas las personas que durante el transcurso de mi carrera siempre estuvieron alentándome en los momentos más difíciles de la carrera, en los momentos en los que parecía que no podría lograrlo y sin embargo ellos nunca dejaron de apoyarme y no dudaron jamás de mis capacidades, haciendo mis momentos muy amenos, llenos de felicidad y mucha alegría.

Finalmente gracias a todas las personas que contribuyeron a concluir esta maravillosa etapa de mi vida.

- René Zamora Marín

Índice

Capítulo 1 Introducción	15
1.1 Introducción:	15
1.2 Justificación.	16
1.3 Objetivos.	16
General	16
Específicos	17
1.4 Definición General.	17
1.5 Problemática.	17
1.5.1 Análisis de la Problemática.	18
1.6 Solución Propuesta.	19
1.7 Funcionamiento General.	20
Capítulo 2 Estado del Arte.....	21
2.1 SmartBell	21
2.2 NUCANO	22
2.3 RING.....	23
Capítulo 3 Marco Teórico.....	26
3.1 Sistemas Embebidos.	26
3.2 Sistemas Operativos.....	29
3.3 Computo Móvil.....	30
3.4 Protocolo.	32
3.5 Modelo Cliente – Servidor.....	33
Capítulo 4. Análisis	34
4.1 Análisis de Riesgo.	34
4.2 Requerimientos.	37
4.2.1 Requerimientos funcionales.	37
4.2.2 Requerimientos no Funcionales.	37
4.3 Reglas de Negocios.....	38
4.4 Tecnologías a Utilizar.	40
Python.....	40
Shell.....	40
Raspbian.	41

Telegram.....	41
Bibliotecas, Codecs, OSS Codecs	45
LUA.....	46
Sensor	46
Radiación Infrarroja	46
Sensor Piroeléctrico.....	47
Servicios (Daemons)	48
Micrófono.....	49
4.5 Hardware a Utilizar.....	49
4.6 Metodología a utilizar	50
4.7 Casos De Uso	52
4.7.1 Caso de uso “Tocar el Timbre”	53
4.7.2 Caso de Uso “Recibir Mensaje”	55
4.7.3 Caso de Uso “Responder Mensaje”	57
4.7.4 Caso de Uso “Escuchar Mensaje”	58
4.7.5 Caso de Uso “Capturar Fotografía”	60
4.8. Diagrama a Bloques.....	61
Capítulo 5. Implementación	62
5.1. Prototipo 1.....	62
5.1.1. Descripción.....	62
5.1.2. Objetivo.....	62
5.1.3. Materiales Necesarios	62
5.1.4. Desarrollo	62
5.2. Prototipo 2.....	71
5.2.1. Descripción.....	71
5.2.2. Objetivo	71
5.2.3. Materiales Necesarios	71
5.2.4. Desarrollo	71
5.3 Prototipo 3 Respuesta Automática de mensajes.	78
5.3.1 Descripción.....	78
5.3.2 Objetivo.....	78
5.3.1 Desarrollo	78

<i>Código Alarma.lua</i>	78
5.4 Prototipo 4.....	83
5.4.1 Descripción.....	83
5.4.2 Objetivo.....	83
5.4.3 Desarrollo	83
5.5 Prototipo 5.....	87
5.5.1 Descripción.....	87
5.5.2 Objetivo.....	87
5.5.3 Desarrollo	87
5.6 Prototipo 6.....	94
5.6.1 Descripción.....	94
5.6.2 Objetivo.....	94
5.6.3 Desarrollo	94
CONCLUSIONES.....	98
TRABAJO A FUTURO	99
REFERENCIAS	100
GLOSARIO.....	102

Índice de Tablas

Tabla 1. Comparación entre softwares de seguridad.....	25
Tabla 2. Características de Sistemas Embebidos (BeagleBone Black - Raspberry Pi).....	28
Tabla 3. Categoría tipo de riesgo.....	34
Tabla 4. Riesgos Existentes.....	36
Tabla 5. Valores de Impacto.....	36
Tabla 6. Requerimientos Funcionales.....	37
Tabla 7. Requerimientos No Funcionales.....	37
Tabla 8. Reglas de negocio.....	39
Tabla 9. Resumen "Tocar el Timbre".....	53
Tabla 10. Resumen "Recibir Mensaje" Usuario.....	55
Tabla 11. Resumen "Responder Mensaje" Usuario.....	57
Tabla 12. Resumen "Escuchar Mensaje" Visitante.....	59
Tabla 13. Resumen "Capturar Fotografía" Sensor.....	60

Índice de Ilustraciones

Ilustración 1. Robo en México 2006-2013.....	18
Ilustración 2. Robo a Casa Habitación con o sin Violencia 2006-2013. [4].....	19
Ilustración 3. Diagrama del Funcionamiento General.....	20
Ilustración 4. Diagrama de Funcionamiento General SmartBell. [5].....	21
Ilustración 5. Nucano Hardware. [6].....	22
Ilustración 6. Funcionamiento general de Nucano. [6].....	23
Ilustración 7. Funcionamiento Ring. [7].....	24
Ilustración 8 Funcionamiento algoritmo de Diffie-Hellman. [17].....	43
Ilustración 9. Diagrama del Protocolo de encriptación MTPProto. [18].....	44
Ilustración 10 Diagrama Distribución Software Raspberry Pi 2. [19].....	45
Ilustración 11 Sensor Pi HC-SR501. [23].....	47
Ilustración 12 Diagrama Rango de Alcance Sensor PIR. [23].....	47
Ilustración 13. Metodología de Construcción por Prototipos. [27].....	51
Ilustración 14. Caso de Uso General.....	52
Ilustración 15. Caso de Uso "Tocar el Timbre".....	53
Ilustración 16. Diagrama de Actividades "Tocar Timbre".....	54
Ilustración 17. Caso de Uso "Recibir Mensaje" Usuario.....	55
Ilustración 18. Diagrama de Actividades "Recibir Mensaje".....	56
Ilustración 19. Caso de Uso "Responder Mensaje".....	57
Ilustración 20. Diagrama de Actividades "Responder Mensaje".....	58
Ilustración 21. Caso de uso "Escuchar Mensaje".....	58
Ilustración 22. Diagrama de Actividades "Escuchar Mensaje".....	59
Ilustración 23. Caso de Uso "Capturar Imagen" Transeúnte y Visitante.....	60
Ilustración 24. Diagrama de Actividades "Capturar Fotografía".....	61
Ilustración 25. Pantalla de instalación del Sistema Operativo Raspbian.....	63

Ilustración 26. Menú Principal de Configuración del Sistema Operativo Raspbian.	64
Ilustración 27. Opción 3 – Iniciar el S.O. con interfaz gráfica.	65
Ilustración 28. Menú principal de la Opción 4.	65
Ilustración 29. Establecer el idioma del Sistema.	66
Ilustración 30. Zona horaria del sistema por continente.	66
Ilustración 31. Zona horaria del sistema por país.	67
Ilustración 32. Configuración del Teclado por marca.	68
Ilustración 33. Configuración del teclado por idioma y caracteres especiales.	68
Ilustración 34. Habilitar la cámara para el sistema embebido.	69
Ilustración 35. Submenú de la opción 8.	69
Ilustración 36. Comandos para actualizar el S.O.	71
Ilustración 37. Librerías para Telegram.	72
Ilustración 38. Descargar Telegram para Raspberry.	73
Ilustración 39. Crear el archivo makefile.	73
Ilustración 40. Compilación de Telegram.	74
Ilustración 41. Ejecución de Telegram.	75
Ilustración 42. Prueba de envío de mensaje de texto a dispositivo móvil.	76
Ilustración 43. Mensaje desde Raspberry.	76
Ilustración 44. Prueba de envío de mensaje a Raspberry.	77
Ilustración 45. Mensaje recibido desde dispositivo móvil.	77
Ilustración 46 Código Python para Automatización de Mensajes.	78
Ilustración 47 Comando para ejecutar Telegram.	79
Ilustración 48 Inicio de Telegram.	79
Ilustración 49 Creación de conversación en Telegram.	80
Ilustración 50 Comprobación de Conexión.	80
Ilustración 51 Envío mensaje "Ping".	81
Ilustración 52 Envío Mensaje "Escom".	81
Ilustración 53 Envío Mensaje "Audio".	82
Ilustración 54 Diagrama Sensor HC-SR501.	83
Ilustración 55 Diagrama Cableado a protoboard.	85
Ilustración 56 Script funcionamiento sensor en Python	85
Ilustración 57 Conversación de Telegram	86
Ilustración 58 Notificación de movimiento detectado por el sensor.	86
Ilustración 59 Código tg.sh.	87
Ilustración 60 Ejecución de comando crontab –e	88
Ilustración 61 Archivo crontab	88
Ilustración 62 Código Alarma.lua.	90
Ilustración 63 Código foto.lua.	90
Ilustración 64 Código Sensor.py.	91
Ilustración 65 Conversación de Telegram.	92
Ilustración 66 Comando reboot y notificación de reinicio de sistema.	92
Ilustración 67 Comando foto, envío y recepción de foto por Telegram.	93

Ilustración 68 Notificación de movimiento detectado y toma de fotos.....	93
Ilustración 69 Código boton.py	94
Ilustración 70 Botón de timbre esperando ser presionado para empezar con su función.....	95
Ilustración 71 Botón presionado y envió de notificación a Telegram.....	95
Ilustración 72 Grabación de audio desde micrófono	96
Ilustración 73 Envío de audio de grabación y recibo del audio en Telegram.	96
Ilustración 74 Regreso al estado original del botón para repetir su función	97

Capítulo 1 Introducción

1.1 Introducción:

Actualmente la tecnología ha revolucionado la forma en la que las personas se comunican, y gracias a esto surgen nuevas necesidades, las cuales se han ido solucionando con sistemas que nos proveen nuevas formas de compartir los que sucede a nuestro alrededor, lo que ha llevado a crear aplicaciones y algunas de ellas son usadas como medios de comunicación.

Dicha evolución tecnológica, electrónica e informática, ha inundado nuestro entorno con dispositivos que cada vez ofrecen más características y funciones, y esta vez nos enfocaremos a la tecnología en el hogar y los edificios denominada Domótica.

Los diccionarios franceses incorporan el término “domotique” a partir de 1998. Esta palabra, traducida al castellano por domótica, es originaria de la palabra latina “domus” (de la que ha derivado la raíz domo que quiere decir casa) y de la palabra francesa “informatique” (informática). Este término de uso común en España, no ha conseguido imponerse en diversos países de Hispanoamérica. [1]

La domótica se aplica a la ciencia y a los elementos desarrollados por ella que proporcionan algún nivel de automatización o automatismo dentro de la casa pudiendo ser desde un simple temporizador para encender y apagar una luz o aparato a una hora determinada, hasta los más complejos sistemas capaces de interactuar con cualquier elemento electrónico de la casa.

Con esto se entiende que la vivienda domótica es aquella que está conformada por una serie de automatismos en materia de electricidad, electrónica, robótica, informática y telecomunicaciones, con el objetivo de asegurar al usuario un aumento del confort, seguridad, ahorro energético, de las facilidades de comunicación, y de las posibilidades de entretenimiento.

Por otro lado, uno de los sistemas que se han encargado en procesar tareas en tiempo real y que han tenido un gran impacto mundial en la actualidad son los sistemas embebidos.

Los sistemas embebidos son dispositivos que son usados para controlar equipos, operación de maquinarias o plantas industriales completas. El término embebido está compuesto por circuitos integrados que son una parte integral del sistema en que se encuentran. Lo interesante de un sistema embebido, es que puede estar incrustado, de tal manera que queda oculto a cualquier ser humano, y por ende no se da cuenta de la presencia de dichos sistemas que pueden operar casi cualquier componente. [2]

Por lo general los sistemas embebidos se pueden programar directamente en el lenguaje ensamblador del microcontrolador o microprocesador incorporado sobre el mismo, o también, utilizando los compiladores específicos, pueden utilizarse lenguajes como C o C++, y en algunos casos, cuando el tiempo de respuesta de la aplicación no es un factor crítico, también pueden usarse lenguajes interpretados como JAVA, Python, PHP, etc.

1.2 Justificación.

Uno de los principales problemas en las grandes ciudades es el estar expuesto al robo de nuestra casa o departamento cuando no nos encontramos habitando en ella. La sociedad necesita implementar cierto métodos de monitoreo a su casa habitación y que mejor que sea mediante el uso de su teléfono celular, sin importar su ubicación, con el cual podrá estar al pendiente de quien toca o visita su casa.

Actualmente se ha optado en nuevos esquemas de seguridad y monitoreo que sean capaces de manejarse mediante el uso del teléfono celular. SiMC es una forma de asegurar, monitorear y proteger la casa o departamento, promoviendo un novedoso sistema de monitoreo, para estar al pendiente de nuestro inmueble, sin importar la ubicación, con el cual el usuario podrá atender y estar al tanto de lo que suceda en el exterior de la casa habitación.

El sistema está basado en la comunicación por medio de mensajería instantánea con el interfono de la casa. SiMC va dirigido a toda persona que cuente con un dispositivo móvil, Tablet o Computadora Personal que tenga cargado un sistema operativo Android 2.1 o superior, o iOS 6.0 o superior y con conexión a internet.

Para realizar este trabajo terminal se requerirán conocimientos sobre comunicaciones en red, sistemas operativos, dispositivos móviles e instrumentación.

1.3 Objetivos.

General:

- Implementar un sistema híbrido capaz de mandar y recibir mensajes de voz a través de una aplicación móvil en el momento de que una persona toque el timbre (o interfono) de la casa-habitación y tener una conversación sin necesidad de estar presente.

Específicos:

- Adaptar el hardware para realizar el envío y recepción de mensajes de audio.
- Realizar la comunicación entre Telegram y la tarjeta Raspberry Pi.
- Realizar el acoplamiento del sensor de proximidad con la cámara fotográfica integrada en la tarjeta Raspberry Pi.
- Un sistema que esté en funcionamiento de forma constante.
- Realizar un sistema accesible a un gran número de personas.
- Mantener un monitoreo sobre la casa habitación.

1.4 Definición General.

¿Qué es un sistema de monitoreo a casa habitación?

Es un sistema que está conformado por un software que ejerce el control de vigilancia a una casa habitación para monitorear y comunicar las acciones de agentes externos. De igual manera puede permitir la comunicación con un dispositivo móvil con el fin de permitir un monitoreo remoto que permitirá aumentar el índice de seguridad.

1.5 Problemática.

Hoy en día una de las preocupaciones más importantes es el cuidado de nuestra casa habitación, ya que en ella se encuentra el patrimonio que nuestra familia ha generado o adquirido a lo largo del tiempo. Esto conlleva a tener un mayor cuidado, monitoreo y vigilancia de nuestro bien.

Con el paso del tiempo esta preocupación ha aumentado, ya que personas ajenas con objetivos malintencionados atentan contra el bienestar de nuestra casa habitación con el fin de usurpar o robar los bienes encontrados en ella.

Debido a esto, se ha optado por el desarrollado y creación de nuevas tecnologías de vanguardia para aumentar la vigilancia y prevenir cualquier atentado contra ella, las cuales pueden tener un amplio sistema de seguridad, soportando incontables metodologías de vigilancia como monitoreo streaming hasta sistemas biométricos de seguridad.

Estos sistemas, debido a las grandes ventajas que proporcionan, llegan a tener un gran costo el cual, en muchas ocasiones, es inalcanzable para muchos usuarios que no cuentan con los recursos necesarios para poner obtener estas herramientas que apoyen su seguridad, y a su vez necesitan de aplicaciones propias que aumentan el consumo de recursos del sistema, esto conlleva a que dicho sistema se torne lento con el paso del tiempo y termine siendo de desagrado para el usuario.

1.5.1 Análisis de la Problemática.

En México, al igual que en varios países de América Latina y del Caribe, se está deteriorando, la vida social, la convivencia armónica y pacífica. Esto sucede por el crecimiento de la inseguridad, que se manifiesta en robos, asaltos, etc. No se trata de hechos aislados o infrecuentes, sino de una situación que se ha vuelto común y que tiene distintas manifestaciones y en la que participan diversos agentes.

El robo a casa habitación es un delito que debe estudiarse por su trascendencia, es un delito tan antiguo como las sociedades, el cual se va actualizando con la tecnología lo cual lo hace vanguardista, es un delito de fuero común pero tiene incidencia Nacional e internacional, es un delito que afecta todos los estratos sociales, todo esto lo hacen del delito de robo a casa habitación relevante, además es un delito que no solo afecta el patrimonio de las víctimas, sino también las afecta psicológicamente, les crea una sensación de paranoia, de inseguridad, ya que se da una violación de su hogar, su lugar de descanso, su refugio. [3]

El robo a casas habitación es el delito que comete una persona al apoderarse de pertenencias ajenas, invadiendo propiedades privadas. Es preocupante el número de robos a casas habitación, se incrementen año con año.

El robo a casa habitación en México representa entre el 15% y el 18% del total de los robos denunciados entre 2006 y 2013. (Ver Ilustración 1)

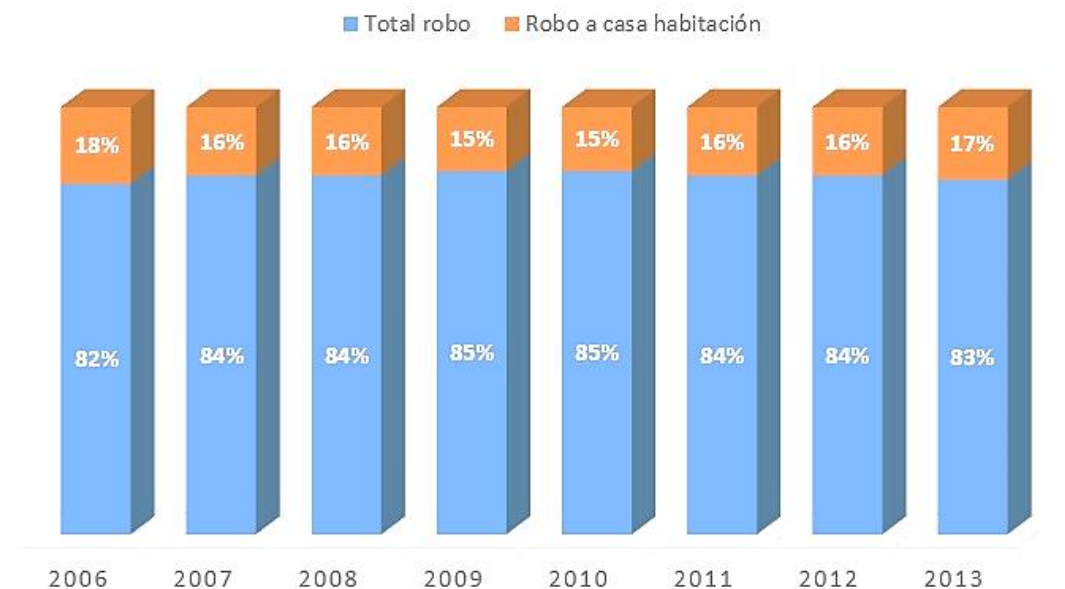


Ilustración 1. Robo en México 2006-2013.

Conforme a la información del Secretariado Ejecutivo del Sistema Nacional de Seguridad Pública (SESNSP), las denuncias por robo a casa habitación del 2006 a 2012 pasaron de 97 mil a 115, representando a un aumento del 19%.

De acuerdo a la Encuesta Nacional de Victimización y Percepción ENVIPE 2013, el robo a casa habitación es el sexto delito más frecuente, con una tasa de 2,656 delitos por cada 100 mil habitantes.

Existen dos formas en la que los robos a casas habitación pueden presentarse (Ver ilustración 2):

- Sin violencia, esto sucede cuando la familia no está en casa por algún motivo, ya que los delincuentes tienen como práctica vigilar las casas que quieren robar. [4]
- Con violencia, cuando la familia se encuentra en casa y reciben amenazas o agresiones físicas por parte de los delincuentes.

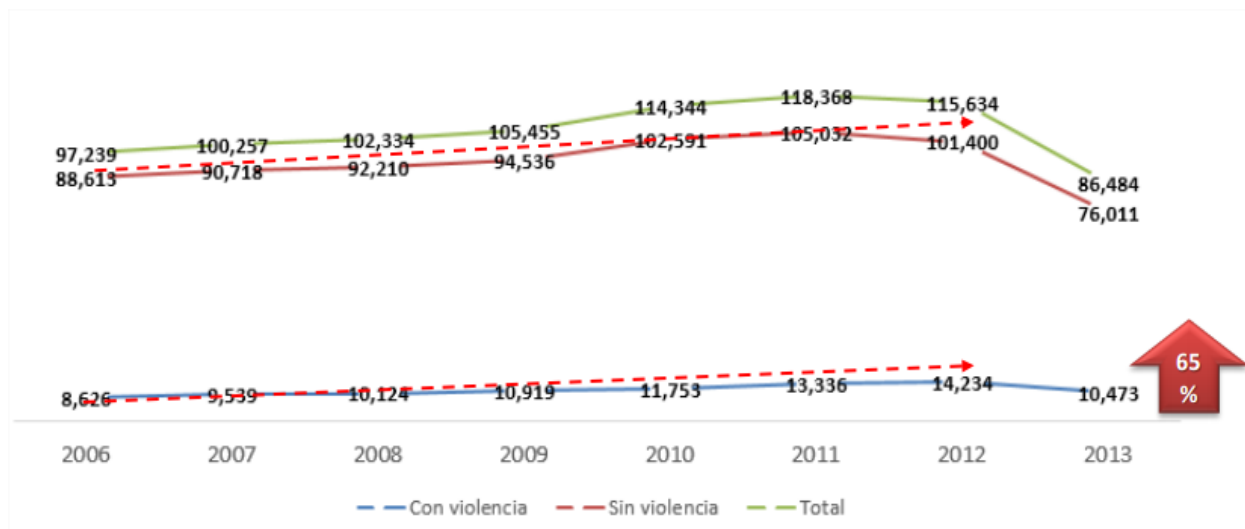


Ilustración 2. Robo a Casa Habitación con o sin Violencia 2006-2013. [4]

1.6 Solución Propuesta.

En el presente trabajo terminal se implementará un sistema híbrido conformado por una aplicación móvil en conjunto con un sistema embebido (interfono), con el fin de monitorear las actividades que se presenten en nuestra casa habitación y así poder prevenir un posible robo a la misma. El sistema híbrido enviará y recibirá mensajes de voz, los cuales el usuario podrá recibir y contestar mediante una aplicación en su dispositivo móvil en cualquier parte donde se encuentre siempre que se tenga acceso a la red. Este sistema podrá ser implementado en cualquier casa-habitación, departamento o negocio con la ventaja de tener un costo accesible para un gran número de personas.

Las condiciones para que este sistema funcione de manera correcta son conexión al cableado eléctrico para la alimentación del sistema embebido, y una conexión a internet ya sea mediante Ethernet o conexión inalámbrica, según el usuario lo prefiera. Dicha conexión se sugiere que tenga una velocidad mayor a 2Mbps para un mejor funcionamiento y rapidez de procesamiento de este.

1.7 Funcionamiento General.

Se puede observar la arquitectura general propuesta en la ilustración 3, el cual consta de un sistema embebido que en su interior tendrá instalado el Sistema Operativo Raspbian, el sistema embebido estará configurado y diseñado para realizar la función de un interfono, el cual permitirá enviar mensajes de voz por medio de un micrófono a un dispositivo móvil, así mismo, recibir mensajes de audio como respuesta del dispositivo móvil.

Adicionalmente, este sistema contará con una conexión a una cámara fotográfica, configurada con un sensor de proximidad, los cuales estarán encargados de tomar fotografías en el instante en el que el sensor detecte alguna actividad, mandando la imagen capturada al dispositivo móvil del propietario.

El usuario por medio de este sistema, tendrá la capacidad de llevar a cabo un monitoreo, además del control y seguridad a su casa habitación desde cualquier lugar con el apoyo de su dispositivo móvil.

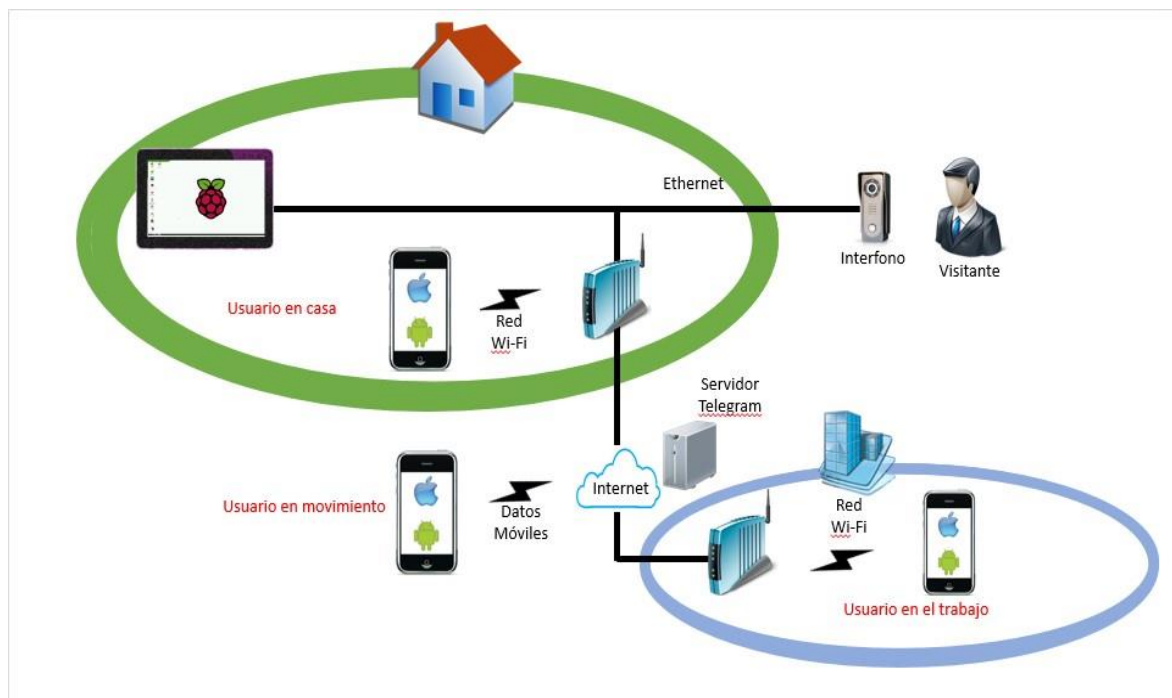


Ilustración 3. Diagrama del Funcionamiento General.

Capítulo 2 Estado del Arte

Los grandes avances en tecnología han permitido desarrollar una gran variedad de dispositivos, en cuanto a domótica se refiere, por lo que a continuación se listan aquellos que ya están en el mercado:

2.1 SmartBell

SmartBell combina la tecnología moderna con la demanda de los dueños de casa para ver y escuchar lo que sucede en la puerta principal, incluso cuando no hay nadie habitando en ella. Se permite una mayor comodidad y control cuando los visitantes.

SmartBell no necesita ningún conocimiento técnico para la instalación. Todo el mundo puede instalarlo sin pagar costos adicionales para un servicio para hacerlo. SmartBell es fácil de operar, operación del producto es en su mayoría auto-explicativo.

¿Cómo Funciona?

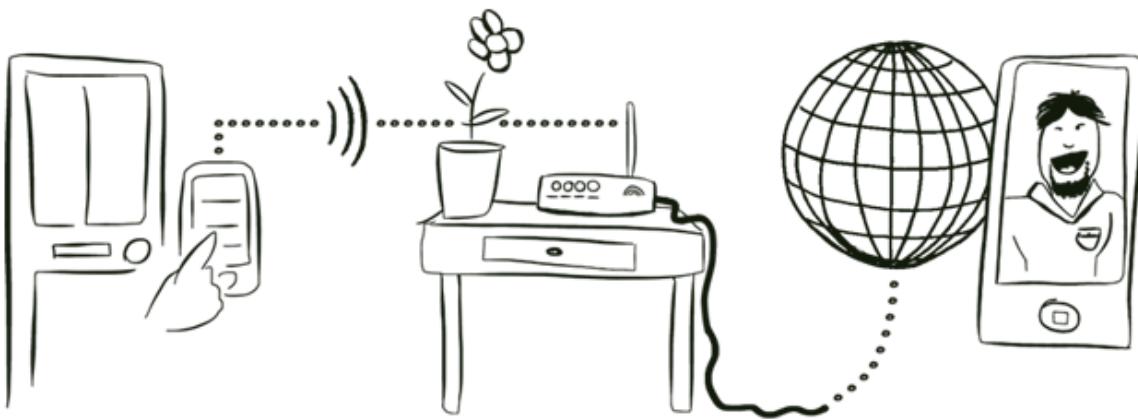


Ilustración 4. Diagrama de Funcionamiento General SmartBell. [5]

Un visitante toca el timbre en la puerta delantera. SmartBell se conecta a través de la red WiFi del router, y a través de internet a su smartphone. El usuario recibe una notificación instantánea que un visitante está en su casa y puede decidir si tomar o no la llamada.

Una vez que acepta la llamada, se establece un chat de vídeo entre el SmartBell y el smartphone. Puede comenzar a hablar con el visitante sin que él o ella sepan dónde está.

Compatibilidad

- iPhone 3GS, 4, 4S, 5, 5s/c (iOS 5.0 o superior)
- iPod Touch con iOS 5.0
- Dispositivos Android (2.3 o superior)

Precio: \$249Dlts. [5]

2.2 NUCANO

Nucano es un proyecto diferente e innovador, pues toma un camino diferente en cuanto a la domótica se refiere. En lugar de utilizar equipos de red convencionales, se ha implementado un centro de inteligencia en casa en un elegante timbre de puerta.

Nucano permite conectarse a más de 1.000 dispositivos compatibles existentes como cerraduras de las puertas, termostatos y cámaras. Da alertas móviles y puede incluso enviar vídeo cuando alguien visita el hogar.

Nucano consta de tres componentes:

- Nucano Smart Door Chime - Funciona como centro de inteligencia en casa.
- Nucano Smart Button - Versátil, inalámbrico y con pilas
- Nucano Mobile app - Controlar su casa desde su teléfono



Ilustración 5. Nucano Hardware. [6]

¿Cómo Funciona?

El dispositivo tiene tanto un radio WiFi y radio Z-Wave. Estos permiten que actúe como un hub, conectar otros dispositivos Z-Wave a Internet. Si ya se cuenta de una red domótica basado en Z-Wave, el Smart Chime puede conectar y actuar como un dispositivo básico Z-Wave, enmascarando todas sus características hub inteligentes.



Ilustración 6. Funcionamiento general de Nucano. [6]

Precio: En fase de pruebas y producción. [6]

2.3 RING

RING nace como una idea de Jamie Siminoff al comentarle a su esposa el poder reinventar el uso de los timbres en casa.

“¡Me encanta! No sólo para la conveniencia obvia sino para la seguridad. Será capaz de responder de forma segura la puerta desde cualquier lugar. Esto es como identificador de llamadas, pero para la puerta de principal”

Jamie Siminoff Wife's

“Nuestra Misión: Reducir el crimen en las comunidades”

Jamie Siminoff

RING permite responder a la puerta desde cualquier lugar utilizando cualquier Smartphone, cuenta con una cámara de alta definición con la visión nocturna, que proporciona una visión clara de su puerta durante el día y la noche.

Usa una batería recargable incorporada le permite instalar el timbre en cualquier lugar, o conectarlo al cableado de un timbre existente. Tiene alertas de movimiento, los cuales son enviados a través de la aplicación gratuita Ring TM (iOS® y Android TM) y notifican a los huéspedes.

Con Grabación en la Nube se asegura que nunca se perderá la actividad en su casa.

¿Cómo funciona?

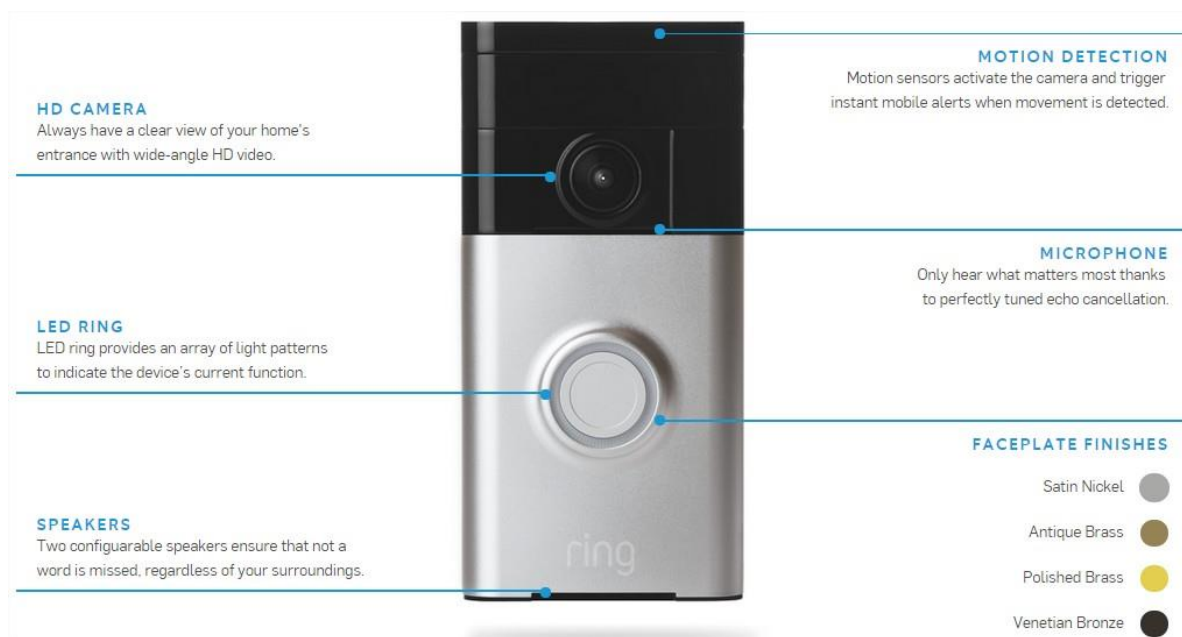


Ilustración 7. Funcionamiento Ring. [7]

Precio: \$199Dls. [7]

En la siguiente tabla se muestra una comparación entre los sistemas anteriormente descritos.

SOFTWARE	VENTAJAS	DESVENTAJAS	PRECIO EN EL MERCADO
SmartBell	<ul style="list-style-type: none"> ▪ Video llamada ▪ Múltiples dispositivos ▪ Voz y video con ruido casi nulo 	<ul style="list-style-type: none"> • Costo elevado • Alto consumo de datos para video llamada 	\$3,839
Nucano	<ul style="list-style-type: none"> • Video vigilancia • Seguridad en puertas • Sistema de luz y calefacción. • Fácil uso para el usuario 	<ul style="list-style-type: none"> • Costo altamente elevado • Control de Puertas (Sistema en desarrollo) • El sistema sigue en pruebas después de su lanzamiento. 	En desarrollo
Ring	<ul style="list-style-type: none"> • Responder a la puerta usando cualquier Smartphone. • Cámara HD, visión nocturna. • Alertas de movimiento. • Grabación en nube. 	<ul style="list-style-type: none"> • Fotografías demasiado pesadas para envió al Smartphone. • Tamaño de videos almacenados muy grandes. • Saturación del espacio en la nube por tamaño excesivo de grabaciones o fotografías. 	\$3,099

Tabla 1. Comparación entre softwares de seguridad.

Capítulo 3 Marco Teórico

3.1 Sistemas Embebidos.

Existen numerosas definiciones de sistemas embebidos, algunas son:

- “Un sistema embebido es cualquier dispositivo que incluye un computador programable, pero en sí mismo no es un computador de propósito general”.
- “Un sistema embebido es un sistema electrónico que contiene un microprocesador o microcontrolador; sin embargo, no pensamos en ellos como un computador”.
- “Las personas usan el término sistema embebido para referirse a cualquier sistema de cómputo escondido en algún producto o dispositivo”. [8]
- “Un sistema embebido es un sistema cuya función principal no es computacional, pero es controlado por un computador integrado. Este computador puede ser un microcontrolador o un microprocesador. La palabra embebido implica que se encuentra dentro del sistema general, oculto a la vista, y forma parte de un todo de mayores dimensiones”.

Un sistema embebido posee hardware de computador junto con software embebido como uno de sus componentes más importantes. Es un sistema computacional dedicado para aplicaciones o productos. Puede ser un sistema independiente o parte de un sistema mayor, y dado que usualmente su software está embebido en ROM (Read Only Memory) no necesita memoria secundaria como un computador. Un sistema embebido tiene tres componentes principales:

1. Hardware.
2. Un software primario o aplicación principal. Este software o aplicación lleva a cabo una tarea en particular, o en algunas ocasiones una serie de tareas.
3. Un sistema operativo que permite supervisar la(s) aplicación(es), además de proveer los mecanismos para la ejecución de procesos. En muchos sistemas embebidos es requerido que el sistema operativo posea características de tiempo real.

En la ilustración 8 se muestra la anatomía de un sistema embebido típico. Este diagrama muestra la arquitectura de hardware a alto nivel de un punto de acceso inalámbrico. La memoria FLASH es utilizada para almacenamiento de datos y programas de forma persistente. La memoria principal típicamente cuenta con algunos megabytes o cientos de megabytes, en los cuales se almacenan valores temporales para la ejecución de programas. En este ejemplo también se observa una interfaz Ethernet y una USB, un puerto serial y un chip que contiene la implementación del estándar IEEE 802.11. [8]

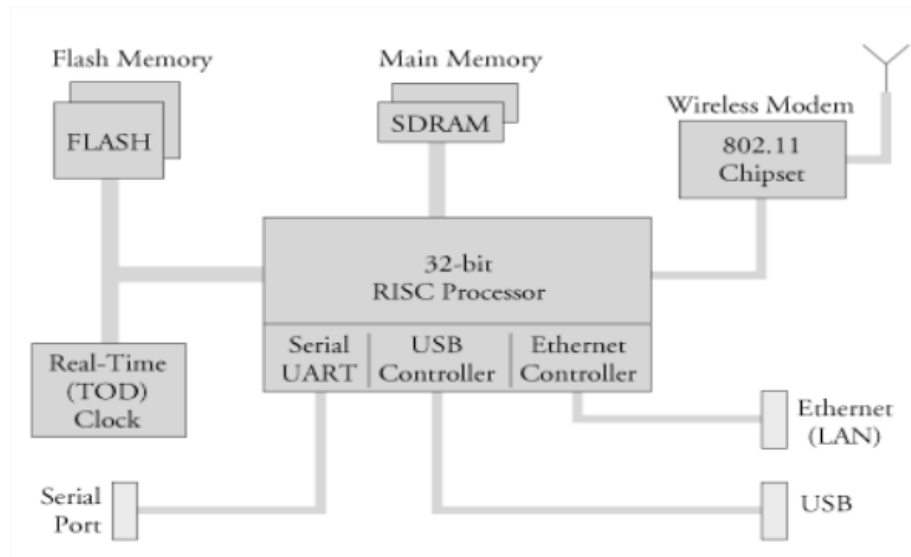


Ilustración 8. Diagrama General de un Sistema Embebido. [8]

En la siguiente Tabla se listan dos de los sistemas embebidos más conocidos y con características similares.

	BeagleBone Black	Raspberry Pi
Precio	\$1450	\$950
Procesador	1GHz TI Sitara AM3359 ARM C�rtex A8	700 MHz ARM1176JZFS
RAM	512 MB DDR3L @ 400 MHz	1 GB SDRAM @ 400 MHz
Almacenamiento	2 GB on-board eMMC, MicroSD	SD
Video	1 Micro-HDMI	1 HDMI, 1 Composite
Resoluciones Soportadas	1280�1024 (5:4), 1024�768 (4:3), 1280�720 (16:9), 1440�900 (16:10) all at 16 bit	Extensive from 640�350 up to 1920�1200, this includes 1080p
Audio	Stereo over HDMI	Stereo over HDMI, Stereo from 3.5 mm jack
Sistemas Operativos	Angstrom (Default), Ubuntu, Android,	Raspbian (Recommended), Ubuntu, Android,

	ArchLinux, Gentoo, Minix, RISC OS, others...	ArchLinux, FreeBSD, Fedora, RISC OS, others...
Alimentación	210-460 mA @ 5V under varying conditions	150-350 mA @ 5V under varying conditions
GPIO	65 Pins	8 Pins
Periféricos	1 USB Host, 1 Mini-USB Client, 1 10/100 Mbps Ethernet	4 USB Hosts, 1 Micro-USB Power, 1 10/100 Mbps Ethernet, RPi camera connector

Tabla 2. Características de Sistemas Embebidos (BeagleBone Black - Raspberry Pi).

Con estas características se optó por utilizar el sistema embebido Raspberry pi, la cual sus características se acoplan a nuestras necesidades, además debido a su costo, este puede hacerse más accesible a los usuarios finales.

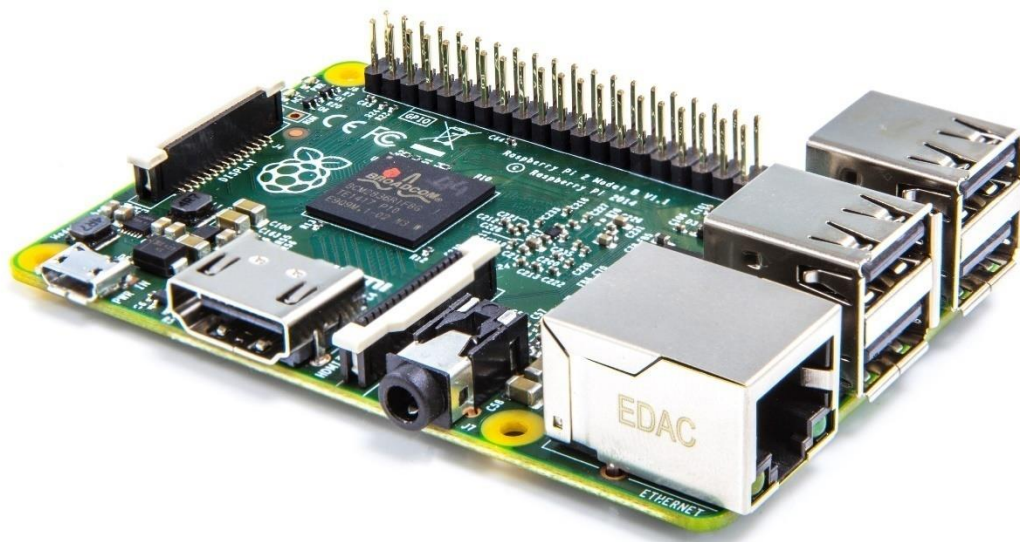


Ilustración 9. Raspberry Pi 2. [9]

3.2 Sistemas Operativos.

Un sistema operativo es un programa que actúa como intermediario entre el usuario y el hardware de un computador y su propósito es proporcionar un entorno en el cual el usuario pueda ejecutar programas, el objetivo principal de un sistema operativo es, entonces, lograr que el sistema de computación se use de manera cómoda, y el objetivo secundario es que el hardware del computador se emplee de manera eficiente.

Un sistema operativo es una parte importante de casi cualquier sistema de computación. Un sistema de computación puede dividirse en cuatro componentes: el hardware, el sistema operativo, los programas de aplicación y los usuarios.

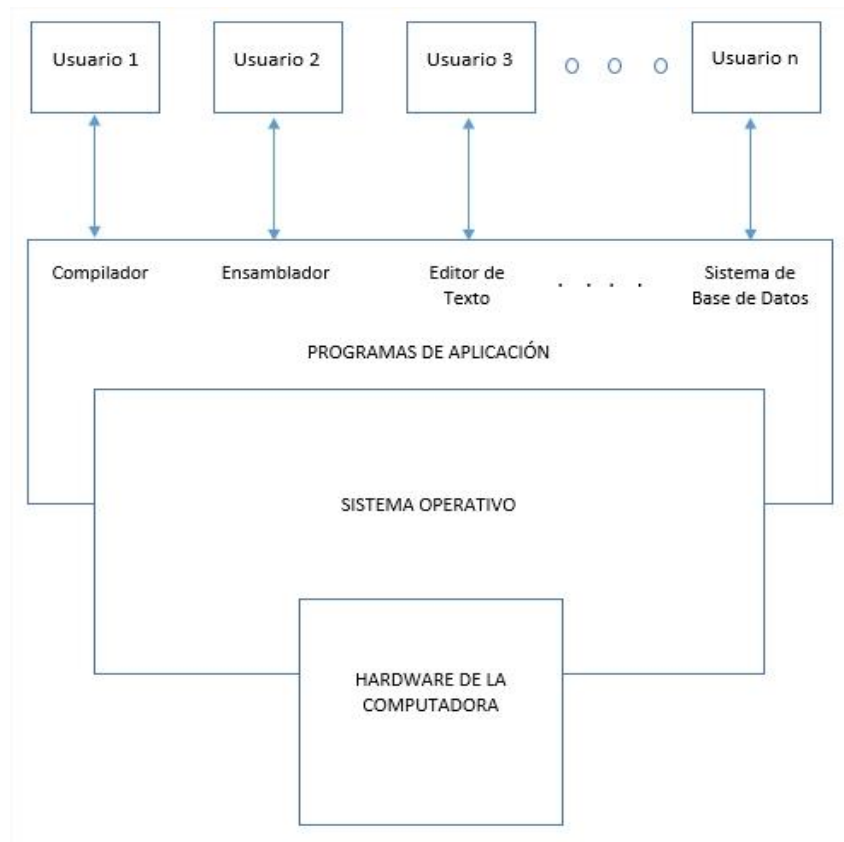


Ilustración 10. Abstracción de Componentes de un Sistema Operativo. [10]

El hardware (unidad central del procesamiento, memoria y dispositivos de entrada y salida) proporciona los recursos de computación básicos. Los programas de aplicación (compiladores, sistema de bases de datos, juegos de video y programas para negocios) definen la forma en que estos recursos se emplean para resolver los problemas de computación de los usuarios.

Puede haber distintos usuarios que intenten resolver problemas diferentes; por consiguiente, es posible que haya diferentes programas de aplicación. El sistema operativo controla y coordinan el uso del hardware entre los diversos programas de aplicación de los distintos usuarios. Un sistema operativo proporciona los medios necesarios para que los componentes de un computador usen adecuadamente durante el funcionamiento del sistema.

Los sistemas operativos existen porque son una manera razonable de solucionar el problema de crear un sistema de computación utilizable. El objetivo fundamental de los sistemas de computación es ejecutar los programas de los usuarios y facilitar la resolución de sus problemas. [11]

3.3 Computo Móvil.

Los avances tecnológicos en la miniaturización de dispositivos y en redes inalámbricas han llevado cada vez más a la integración de dispositivos de computación pequeños y portátiles. Estos dispositivos incluyen.

- Computadores portátiles.
- Dispositivos de mano (handheld), entre los que se incluyen asistentes digitales personales (PDA), teléfonos móviles, buscapersonas videocámaras o macaras digitales.
- Dispositivos que se pueden llevar puestos, como relojes inteligentes con funcionalidad semejante a la de los PDA's.
- Dispositivos insertados en aparatos, como lavadoras, sistemas de fidelidad, coches, etc.

La facilidad de transporte de muchos de estos dispositivos, junto con su capacidad para conectarse adecuadamente a redes en diferentes lugares, hace posible la computación móvil. Se llama computación móvil a la realización de tareas de computo mientras el usuario está en movimiento o visitando otros lugares distintos de su entorno habitual. En la computación móvil, los usuarios que están fuera de su hogar intranet disponen de la posibilidad de acceder a los recursos mediante los dispositivos que llevan con ellos. Pueden continuar accediendo a Internet; pueden acceder a los recursos de su intranet.

En la ilustración 11, se muestra a un usuario que está visitando una organización. Se puede apreciar la intranet de la casa del usuario y la de la organización anfitriona en el lugar que está visitando dicho usuario. Ambas intranets están conectadas al resto de Internet.

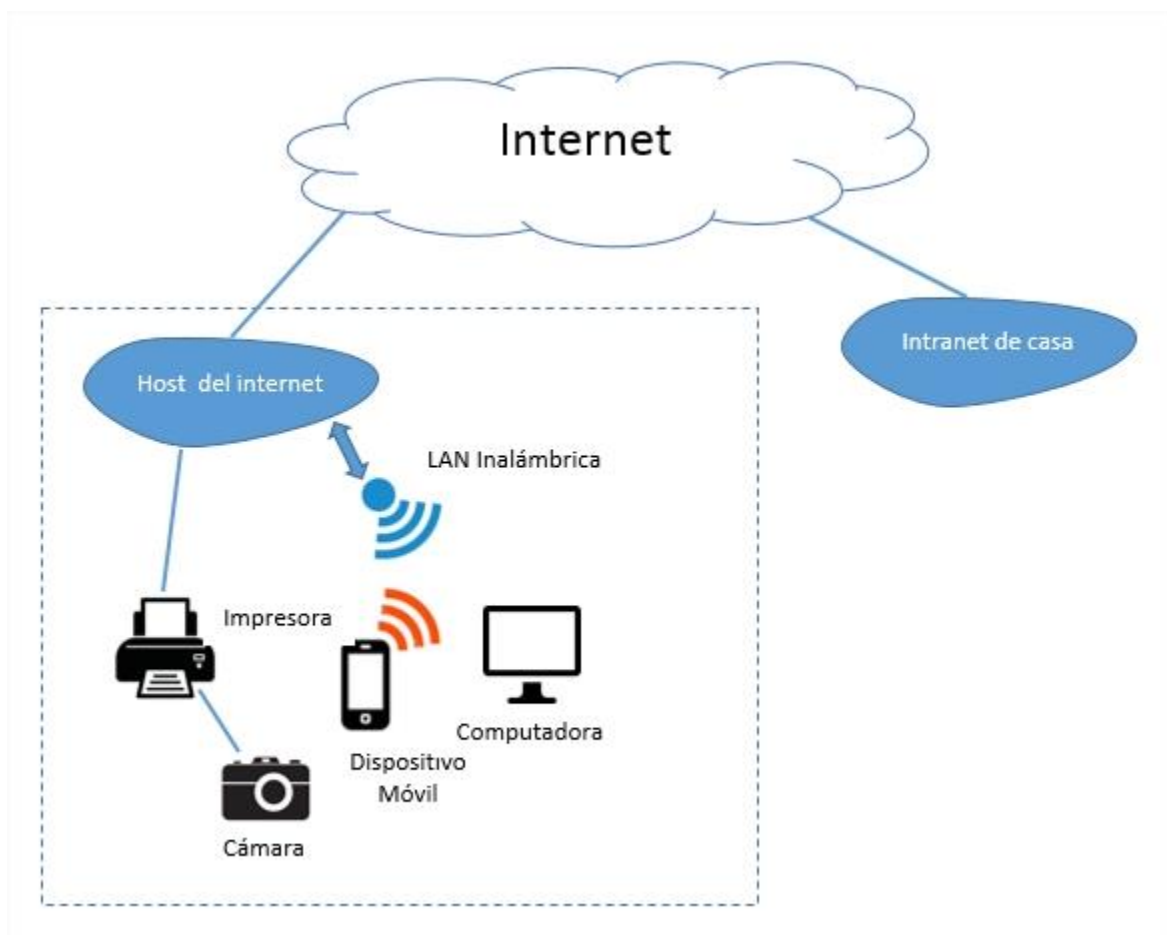


Ilustración 11. Diagrama de arquitectura de cómputo móvil.

El usuario tiene acceso a tres horas de conexión inalámbrica. Su computador portátil tiene un medio para conectarse a la red de área local (LAN) inalámbrica de su anfitrión. Esta red proporciona una cobertura de unos pocos cientos de metros. Se conecta con el resto de la intranet del anfitrión mediante una pasarela. El usuario dispone además de un teléfono móvil, que está conectado a internet por medio del protocolo de acceso inalámbrico (WAP) a través de una pasarela.

Con una infraestructura adecuada del sistema, el usuario puede realizar algunas tareas sencillas en el lugar del anfitrión utilizando los dispositivos que lleva. [11]

3.4 Protocolo.

El término protocolo se utiliza para referirse a un conjunto bien conocido de reglas y formatos que se han de utilizar para la comunicación entre procesos que realizan una tarea determinada. La definición de un protocolo tiene dos partes importantes:

- Una especificación de la secuencia de mensajes que se han de intercambiar.
- Una especificación del formato de los datos en los mensajes.

La existencia de protocolos bien conocidos posibilita que los componentes de software separados, que forman parte de sistemas distribuidos, puedan desarrollarse independientemente e implementarse en diferentes lenguajes de programación sobre computadores que quizás tengan diferente representación interna de los datos.

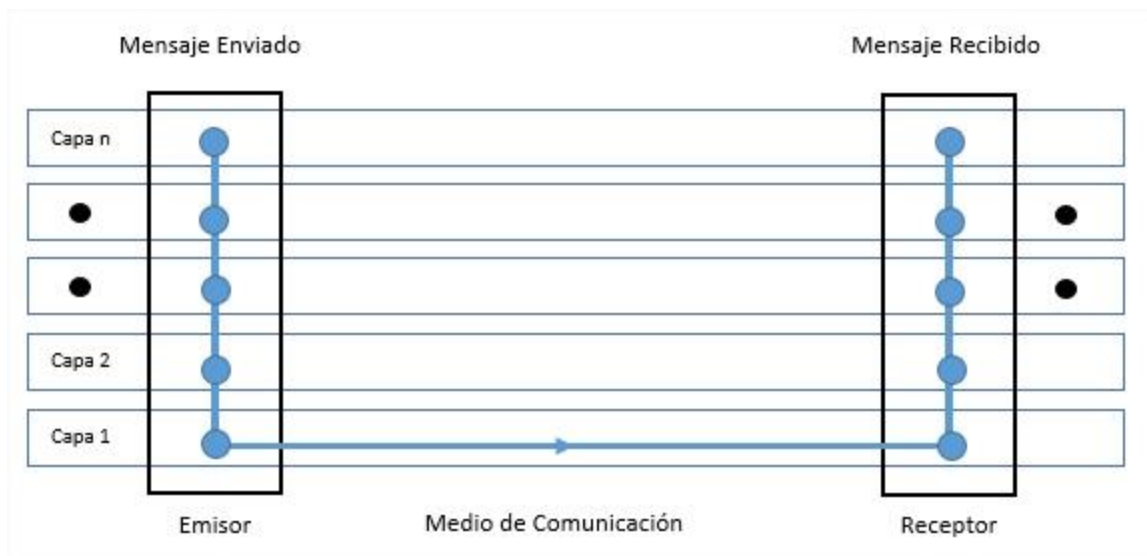


Ilustración 12. Protocolo para comunicación en diferentes capas.

Un protocolo está implementado por dos módulos de software ubicados en los computadores del emisor y del receptor. Por ejemplo, un protocolo de transporte transmite el mensaje de cualquier longitud desde un proceso emisor hacia un proceso receptor. Un proceso que desee transmitir un mensaje a otro proceso efectuará una llamada al módulo del protocolo de transporte, pasándole el mensaje en cierto formato.

Entonces, el software de transporte se pondrá a la tarea de transmitir el mensaje a su destino, subdividiéndolo en paquetes de un tamaño y formatos determinados, así estos podrán ser transmitidos al destino vía protocolo de red, otro protocolo de nivel inferior. El correspondiente módulo del protocolo de transporte en computador destino recibirá el paquete de su módulo del protocolo de nivel de red y realizará las transformaciones inversas para regenerar el mensaje antes de entregárselo al proceso receptor. [11]

3.5 Modelo Cliente – Servidor

El modelo arquitectónico cliente-servidor es un modelo de sistema en el que dicho sistema se organiza como un conjunto de servicios y servidores asociados, más unos clientes que acceden y usan los servicios. Los principales componentes de este modelo son:

1. Un conjunto de servidores que ofrecen servicios a otros subsistemas. Ejemplos de servidores son servidores de impresoras que ofrecen servicios de impresión, servidores de ficheros que ofrecen servicios de gestión de ficheros y servidores de compilación, que ofrecen servicios de compilación de lenguajes de programación.
2. Un conjunto de clientes que llaman a los servicios ofrecidos por los servidores. Estos son normalmente subsistemas en sí mismos. Puede haber varias instancias que un programa cliente ejecutándose concurrentemente.
3. Una red que permite a los clientes acceder a estos servicios. Esto no es estrictamente necesario ya que los clientes y los servidores podrán ejecutarse sobre una única máquina. En la práctica, sin embargo, la mayoría de los sistemas cliente-servidor se implementan como sistemas distribuidos.

Los clientes pueden conocer los nombres de los servidores disponibles y los servicios que estos proporcionan. Sin embargo, los servidores no necesitan conocer la identidad de los clientes o cuantos clientes tienen. Los clientes acceden a los servicios proporcionados por un servidor a través de llamadas de procedimientos remotos usando un protocolo de petición respuesta tal como el protocolo http usado en WWW. Básicamente, un cliente realiza una petición a un servidor y espera hasta que recibe una respuesta.

La ventaja más importante del modelo cliente servidor es que es una arquitectura distribuida.

- Se puede hacer un uso efectivo de los sistemas en red con muchos procesadores distribuidos.
- Es fácil añadir un nuevo servidor e integrarlo con el resto del sistema o actualizar los servidores de forma transparente sin afectar al resto del sistema.

Sin embargo, puede ser necesario realizar cambios a los clientes y servidores existentes para obtener los mayores beneficios de la integración a un nuevo servidor. Puede no haber un modelo de datos compartidos entre los servidores, y los subsistemas pueden organizar sus datos de formas diferentes. Esto significa que los modelos de datos específicos pueden establecerse para cada servidor con el fin de optimizar su rendimiento. [12]

Capítulo 4. Análisis

4.1 Análisis de Riesgo.

A continuación en la tabla 2 se describe de forma general los tipos de riesgos para poder aplicarlos a nuestro proyecto e implementar un plan de contingencia hacia el riesgo presentado.

Categoría	ID	Descripción
Riesgos de proyecto	RP	Los riesgos del proyecto amenazan al plan del proyecto, es decir, si los riesgos del proyecto se hacen realidad, es probable que la planeación temporal del proyecto se retrase y que los costos aumenten.
Riesgos Técnicos	RT	Los riesgos técnicos amenazan la calidad y la planificación temporal del software que hay que producir. Si un riesgo técnico se convierte en realidad, la implementación puede ser difícil o imposible.
Riesgos de Negocio	RN	Los riesgos del negocio amenazan la viabilidad del software a construir. Los riesgos del negocio a menudo ponen en peligro el proyecto o el producto
Riesgos Conocidos	RC	Los riesgos conocidos son todos aquellos que se pueden descubrir después de una cuidadosa evaluación del plan del proyecto, del entorno técnico y comercial en el que se desarrolla el proyecto y otras fuentes de información fiables.
Riesgos Predecibles	RR	Los riesgos predecibles se extrapolan de la experiencia en proyectos anteriores.
Riesgos Impredecibles	RI	RI Los riesgos impredecibles son el comodín de la baraja. Pueden ocurrir, pero son extremadamente difíciles de identificar por adelantado.

Tabla 3. Categoría tipo de riesgo.

A continuación, abarcamos todos los riesgos existentes, que nos ayudarán para tener un plan de contingencia a manera de indicar a cualquier persona como proceder en caso de que se presente.

Tipo	Categoría	ID	Descripción	Gestión/ Plan Contingencia	Probabilidad.
Interpretación errónea de los objetivos.	RC	3	Durante el diseño o desarrollo se puede interpretar de forma incorrecta alguno de los objetivos lo que conllevaría a un diseño erróneo del sistema.	Realizar constantemente revisiones detalladas de los requisitos y funcionalidades para comprobar que se cumplen con los objetivos verdaderos.	Baja
Diseño incorrecto.	RT	3	Al tener un diseño incorrecto provoca un cambio significativo en todas las etapas del sistema y retrasa el sistema.	Realizar un análisis detallado del como diseñar el sistema, basándose en los objetivos y funciones.	Alta
Cambio en los Objetivos.	RP	3	Al cambiar los objetivos en un momento inesperado puede cambiar todo el sistema haciendo perder el tiempo.	Realizar una evaluación del proyecto constantemente para comprobar que los objetivos son los adecuados para el desarrollo del sistema.	Media
Retraso del Diseño	RP	3	Desplazar la fecha estipulada por retraso de las demás etapas.	Cumplir con el cronograma de actividades en tiempo y forma, administrando el tiempo de forma correcta, además de acelerar el proceso del diseño para que no ocurra ningún contratiempo.	Alta
Retraso en la fase de pruebas	RP	3	Esto conlleva a un atraso en la entrega de los prototipos lo que causaría un	Agilizar el tiempo de pruebas, pero con un mayor	Alta

			aplazamiento en la fecha de entrega.	análisis, para disminuir el riesgo final.	
Falta de conocimiento de las tecnologías utilizadas.	RP	3	Los desarrolladores no están familiarizados con las tecnologías a utilizar, lo que puede conllevar a retrasos y pérdida de tiempo.	Asignar una etapa de familiarización con las tecnologías para un mejor entendimiento, con esto todos los desarrolladores será capaz de poder resolver cualquier situación de riesgo presentada, esto traerá un mejor control.	Media
Fallas de hardware.	RI	4	En fase de desarrollo o pruebas el hardware presenta alguna falla, que traiga como consecuencia retrasos y pérdida de tiempo.	Llevar revisiones constantes para ver el estado del hardware.	Baja

Tabla 4. Riesgos Existentes.

A continuación, se muestra los valores de impacto:

Valores de Impacto		
Impacto	ID	Descripción.
Catastrófico	4	El daño es irremediable.
Critico	3	Puede corregirse.
Marginal	2	Puede corregirse fácilmente.
Despreciable	1	No hay problema.

Tabla 5. Valores de Impacto.

4.2 Requerimientos.

4.2.1 Requerimientos funcionales.

Identificador	Requerimientos Funcionales
RF – 1	Capturar: Fotografía automática por aproximación a la puerta principal mediante el sensor. Mensaje de voz al accionar el interruptor de comunicación del interfono.
RF – 2	Envío: Mensaje de voz a dispositivo móvil. Mensaje de voz a sistema embebido.
RF – 3	Recibir: Mensaje de voz de dispositivo móvil. Mensaje de voz de sistema embebido.
RF – 4	El sistema embebido reproducirá los mensajes de audio recibidos por el propietario como respuesta al usuario externo.
RF – 5	El sistema Operativo (SO) que se utilizará es Raspbian para el Sistema Embebido.
RF – 6	Lenguaje de Programación Python.

Tabla 6. Requerimientos Funcionales.

4.2.2 Requerimientos no Funcionales.

Identificador	Requerimientos No Funcionales
RNF-1	Sistema de bajo costo.
RNF-2	Seguridad en envío y recepción de mensajes.
RNF-3	El Sistema Operativo (SO) que se utilizara es Android 2.2 y versiones superiores, así como también iOS 6.0 o posterior.
RNF-4	Conexión al cableado eléctrico para la alimentación del sistema embebido.
RNF-5	Conexión a Internet mediante cable Ethernet o conexión inalámbrica a una velocidad mayor a 2 Mbps.

Tabla 7. Requerimientos No Funcionales.

4.3 Reglas de Negocios.

En las reglas de negocio se describen las características y políticas del funcionamiento del sistema de prevención a robo a casa habitación, así como algunas de las restricciones de uso para un correcto funcionamiento.

Descripción de la regla			
Identificador	Nombre de la regla	Descripción de la Regla	Observaciones
RN-1	Captura de imagen de aproximación a puerta.	Si una persona se aproxima a la puerta principal de la casa, se tomará una foto.	1. Todas las capturas de imágenes, se harán en el momento que el sensor detecte algún movimiento.
RN-2	Envío de imagen al dispositivo móvil del usuario.	La foto capturada, será enviada al dispositivo móvil de usuario.	1. La captura de foto se enviará después de haber sido tomada, con el fin de corroborar la actividad suscitada en su domicilio. 2. El sistema embebido deberá contar con una conexión a internet
RN-3	Reproducción de mensaje de audio por interfono.	Cuando el sistema embebido reciba el mensaje de voz, este deberá reproducirlo por medio del interfono.	1. Los mensajes de voz recibidos, se reproducirán a través del interfono.
RN-4	Envío de mensajes de audio a dispositivo móvil del usuario.	El mensaje de sonido que grabara el sistema embebido se enviara al dispositivo móvil del usuario.	1. Los mensajes de voz previamente grabado por la persona del interfono, se enviarán para reproducirse en el dispositivo móvil del usuario. 2. El interfono deberá contar con una conexión a internet para el envío del mensaje de voz.
RN-5	Reproducción de mensaje de audio en el dispositivo móvil del usuario.	Cuando el sistema móvil reciba el mensaje de voz, el usuario podrá reproducirlo a través de la aplicación.	1. Los mensajes de voz recibidos, se reproducirán a través de los altavoces del dispositivo móvil.

RN-6	Grabación de mensaje de audio en el interfono	La persona frente al interfono deberá de presionar el botón del interfono, para poder comunicarse con el usuario.	1. La persona situada frente al interfono, no sabrá que el sistema embebido grabará ese mensaje de audio para posteriormente enviarlo al dispositivo móvil del usuario.
RN-7	Grabación de mensaje de audio en el dispositivo móvil.	El usuario grabara un mensaje de voz, para comunicarse con la persona del interfono	1. Los mensajes de voz que grabara el usuario, se harán presionando un botón táctil indicando el inicio de grabación y dejándolo de presionar señalando el final de grabación.
RN-8	Envío de mensaje al propietario.	Los mensajes de audio, así como las fotografías capturadas solo serán enviados a las personas autorizadas.	Al inicio de la configuración, se podrá establecer a quienes se les enviara las notificaciones adquiridas por el sistema, podrán ser solo una persona o un grupo de personas.

Tabla 8. Reglas de negocio.

4.4 Tecnologías a Utilizar.

Python.

Python es un lenguaje de scripting independiente de plataforma, preparado para realizar cualquier tipo de programa, desde aplicaciones Windows a servidores de red o incluso, páginas web. Es un lenguaje interpretado, lo que significa que no se necesita compilar el código fuente para poder ejecutarlo, lo que ofrece ventajas como la rapidez de desarrollo e inconvenientes como una menor velocidad. [13]

Cuenta con estructuras de datos eficientes y de alto nivel y un enfoque simple pero efectivo a la programación orientada a objetos. El intérprete de Python puede extenderse fácilmente con nuevas funcionalidades y tipos de datos implementados en C o C++ (u otros lenguajes accesibles desde C). Python puede usarse como un lenguaje de extensiones para aplicaciones personalizables. Este tutorial introduce de manera informal al lector a los conceptos y características básicas del lenguaje y el sistema de Python.

Características.

- La cantidad de bibliotecas que contiene, tipos de datos y funciones incorporadas en el propio lenguaje, que ayudan a realizar muchas tareas habituales sin necesidad de tener que programarlas desde cero.
- La sencillez y velocidad con la que se crean los programas.
- La cantidad de plataformas en las que podemos desarrollar, como Unix, Windows, OS/2, Mac y otros.
- Además, Python es gratuito, incluso para propósitos empresariales.

Shell

Una Shell de Unix o también shell, es el término usado en informática para referirse a un intérprete de comandos, el cual consiste en la interfaz de usuario tradicional de los sistemas operativos basados en Unix y similares como GNU/Linux.

Mediante las instrucciones que aporta el intérprete, el usuario puede comunicarse con el núcleo y por extensión, ejecutar dichas órdenes, así como herramientas que le permiten controlar el funcionamiento de la computadora.

Los comandos que aportan los intérpretes, pueden usarse a modo de guion si se escriben en ficheros ejecutables denominados shell-scripts, de este modo, cuando el usuario necesita hacer uso de varios comandos o combinados de comandos con herramientas, escribe en un fichero de texto marcado como ejecutable, las operaciones que posteriormente, línea por línea, el intérprete traducirá al núcleo para que las realice. Sin ser un shell estrictamente un lenguaje de programación, al proceso de crear scripts de shell se le denomina programación shell o en inglés, shell programming o shell scripting.

En el sentido más genérico del término, shell significa cualquier intérprete que los usuarios usen para escribir comandos. Su etimología proviene del uso natural de consolas en computadores funcionando bajo Unix antaño, cuando los usuarios conectaban al computador central, lo hacían mediante consolas, (shells) por las cuales, a través de un intérprete, hacían inicio de sesión y manejaban la computadora principal.

Posteriormente, con la proliferación de los computadores personales y su filosofía monousuario, un computador por usuario (entiéndase lo contrario de la filosofía inicial de Unix, un computador, muchos usuarios conectados por terminales), se desarrolló un software que emulase las características principales de las consolas físicas, a modo de poder seguir usándolas como clientes en un computador o núcleo que comprendiera la convención estándar usada para configurar y llevar a cabo tareas de administración de emergencia con servidores basados en Unix. [14]

Raspbian.

Raspbian es un sistema operativo libre basado en Debian optimizado para el hardware Raspberry Pi. Un sistema operativo es el conjunto de programas básicos y utilidades que hacen que su funcionamiento Raspberry Pi. Sin embargo, Raspbian ofrece más que un SO puro; viene con más de 35.000 paquetes, software pre-compilado en un formato que lo hace más funcional.

La construcción inicial de más de 35.000 paquetes de Raspbian, optimizado para un mejor rendimiento en el Raspberry Pi, se terminó en junio de 2012. Sin embargo, Raspbian está todavía en desarrollo activo con un énfasis en la mejora de la estabilidad y el rendimiento de la mayor cantidad de paquetes de Debian como sea posible.

Raspbian es una comunidad de software fundada y soportada por los propios miembros de forma libre y gratuita. Y aunque Raspbian es gratis, el desarrollo del sistema tuvo su costo. Así, los autores de este sistema operativo piden el apoyo de la comunidad vía donativos. [15]

Telegram.

Telegram es una aplicación de mensajería enfocada en la velocidad y seguridad, es súper rápida, simple y gratis. Telegram se puede usar en todos los dispositivos al mismo tiempo – los mensajes se sincronizan a la perfección a través de cualquiera los teléfonos, tablets o computadoras.

Telegram, puede enviar mensajes, fotos, vídeos y archivos de cualquier tipo (doc, zip, mp3, etc.), también se puede crear grupos de hasta 200 personas. Se puede escribir a los contactos del teléfono y encontrar personas a través de sus alias. Telegram es como SMS y correo electrónico combinados y puede hacerse cargo de todas las necesidades.

Telegram es apoyada por Pavel y Nikolai Durov. Pavel apoya a Telegram financiera e ideológicamente, mientras el aporte de Nikolai es tecnológico. Para hacer Telegram posible,

Nikolai desarrolló un protocolo de datos personalizado y único, que es abierto, seguro y optimizado para trabajar con múltiples centros de datos. Como resultado, Telegram combina seguridad, confiabilidad y velocidad en cualquier red. [16]

Criptografía Simétrica

Telegram cifra los mensajes de una forma simple: se cifra con una *clave* y se descifra en el otro lado con esa misma *clave*. Al estar los mensajes cifrados, el usuario que esté espiando el tráfico del móvil no sabrá de qué se está comunicando entre los usuarios.

El problema es *la clave*. La clave no puede ser la misma para todos los clientes porque entonces cualquiera podría descifrar los mensajes. Tiene que ser distinta para cada cliente.

Cada vez que un usuario instale la aplicación de Telegram se tiene que crear una *clave*, y esa *clave* tiene que enviarse al servidor para que pueda descifrar los mensajes. Obviamente, no podemos transmitir la clave en claro al servidor. Se tiene que buscar una forma de decirle al servidor cuál es la clave sin decirle cuál es la clave. Esta idea, que parece imposible, es en realidad un algoritmo muy sencillo. [16]

Diffie-Hellman, Intercambiando Claves De Forma Segura

La implementación de Diffie-Hellman se basa en matemáticas, más concretamente en grupos de enteros multiplicativos módulo p , con p primo. Sin embargo, lo que nos interesa de esa teoría matemática es una propiedad muy importante: es muy fácil operar un número a con otro b para que salga c , pero a partir de c es muy difícil saber qué números a y b lo han generado.

Pero es más fácil entender Diffie-Hellman con colores que con números. Este es el esquema del intercambio de claves (suponiendo que no sabemos cómo separar dos colores):

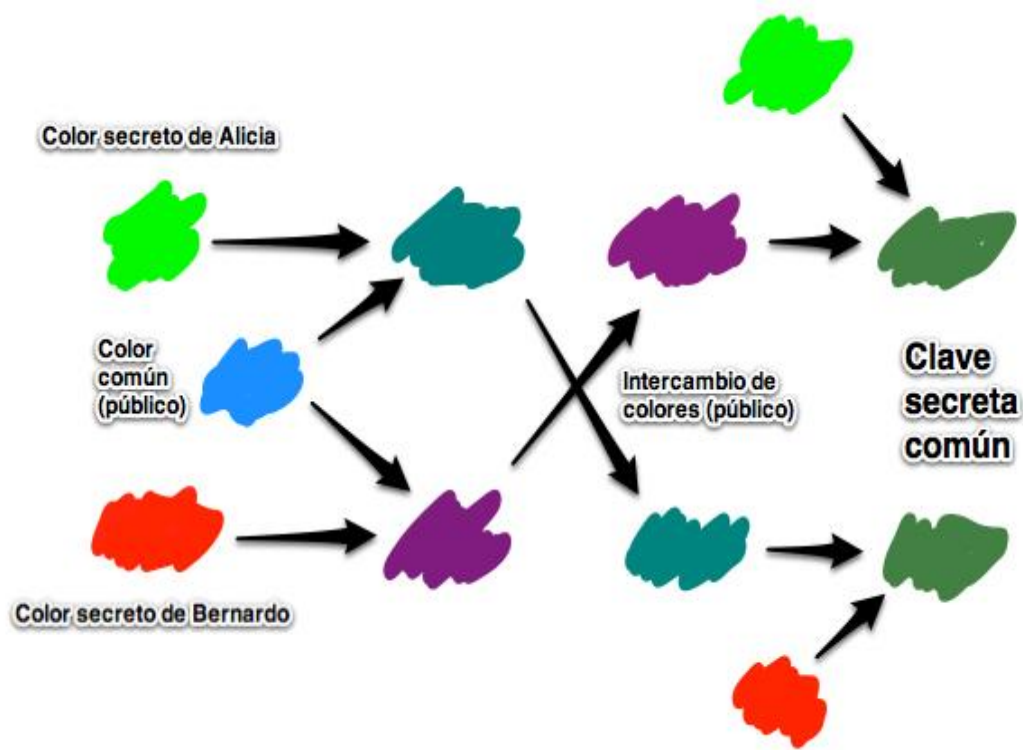


Ilustración 8 Funcionamiento algoritmo de Diffie-Hellman. [17]

Telegram usa el algoritmo de cifrado **AES con claves de 256 bits, en modo IGE** (Infinite Garble Extension). El modo tiene que ver con la entrada del algoritmo: AES cifra por bloques, cadenas de 128 bits. Dependiendo del modo en el que usemos el algoritmo, cifraremos los bloques provenientes del texto plano o los combinaremos de alguna forma con bloques cifrados anteriores.

Lo que sí importa es que AES-IGE es vulnerable a ciertos ataques de criptoanálisis. Por ejemplo, a CPA (Chosen Plaintext Attack). Este tipo de ataque consiste en cifrar ciertos textos planos que tú conoces y ver cuál es el resultado. A partir de esos textos (que no son elegidos al azar, sino que se forman específicamente para cada situación), se pueden inferir datos sobre la clave, reduciendo las posibilidades y aumentando la probabilidad de adivinarla por completo.

Para contrarrestar este tipo de ataques, Telegram implementa dos características ingeniosas. La primera es que, cuando se cifra un mensaje, no se cifra sólo el mensaje, sino que a él se añade un *salt*, una cadena aleatoria que define el servidor de Telegram, la hora actual y un número de secuencia que indica el orden del mensaje.

De esta forma, un atacante no puede cifrar con AES-IGE el texto que él quiera, ya que la aplicación siempre añade esos tres valores fijos que él no controla.

La segunda característica es que además de cifrar los mensajes, queremos comprobar que no se han modificado por el camino (accidentalmente o por un atacante). Para ello usamos un MAC (Message Authentication Code, o código de verificación de mensaje).

Este código se obtiene a partir del mensaje, y es (casi) único. Si el mensaje cambia, aunque sólo sea en una tilde, el MAC será distinto. Es parecido a las funciones hash o de huella digital, aunque no exactamente igual.

La clave con la que se cifra el mensaje depende de su contenido.

A la hora de enviar el mensaje, Telegram obtiene su huella digital mediante el algoritmo SHA-1. El resultado es una clave que identifica al mensaje. Pero además ese resultado se combina junto con la clave compartida entre cliente y servidor para obtener una nueva clave con la que cifrar el mensaje.

Es decir, que la clave con la que se cifra cada mensaje depende de su propio contenido. Este hecho da mucha robustez al cifrado de Telegram y además impide varios ataques a los que son vulnerables algunos de sus componentes. El esquema final del cifrado es el que sigue:

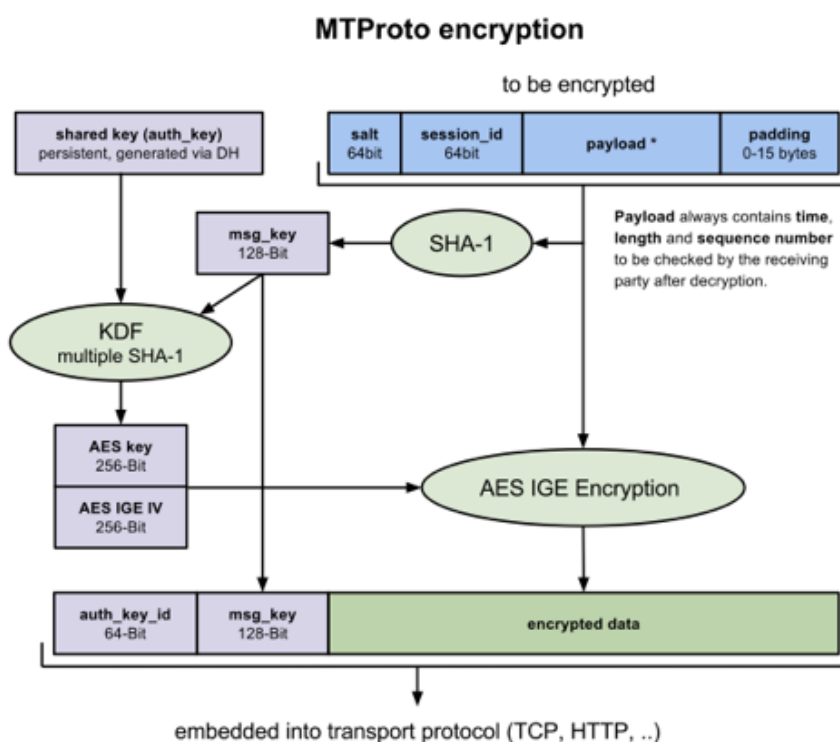


Ilustración 9. Diagrama del Protocolo de encriptación MTPROTO. [18]

Tras cifrar el mensaje, se envía un paquete que contiene el mensaje cifrado, la huella digital SHA-1 del texto sin cifrar y un número que identifica la clave compartida que se usa.

Al recibir el mensaje en el otro lado de la conexión, se recrea la clave de cifrado usando la clave compartida y la huella del mensaje, se descifra el texto y se comprueba que la huella digital concuerda. También se comprueba que el *salt* sea correcto (igual al que el servidor haya definido), que la fecha y hora sea razonable (por ejemplo, no aceptaremos un mensaje con fecha dos años en el futuro) y que el número de secuencia sea el apropiado.

Así, con esa comprobación múltiple, Telegram se asegura que nadie más puede leer el mensaje y que además ha llegado sin ningún tipo de modificación.

Bibliotecas, Codecs, OSS Codecs

Dos códecs con licencia serán proporcionados en el lanzamiento, MPEG4 y H.264. Los codecs de las licencias tienen un gran impacto en el rendimiento del dispositivo, por lo cual solo hay sólo dos en esta etapa. Hay Codecs sin licencia como en MPEG2, VC1, etc., pero por el momento no van a ser acelerados por la GPU. [19]

OPEN vs CLOSED SOURCE

Las bibliotecas secundarias para la aceleración de gráficos son de código cerrado y son proporcionados por el proveedor de SoC. La Fundación no tiene control sobre la naturaleza cerrada de estas bibliotecas. Dado que la gran mayoría de la gente sólo tiene que utilizar las bibliotecas de este tipo, se traspasó una solución para conseguir el alto rendimiento de los gráficos. Vale la pena señalar que no hay otros dispositivos SoC con un rendimiento de gráficos similares que son de código abierto.

Hay unos pocos controladores para el SoC que están vinculados en el kernel, estos son GPL y por lo tanto OSS. Uno de estos controladores es la interfaz de las bibliotecas del espacio de usuario a la GPU. Las bibliotecas lado del usuario utilizan este "conductor" para comunicarse con el GPU y decirle lo que debe hacer.

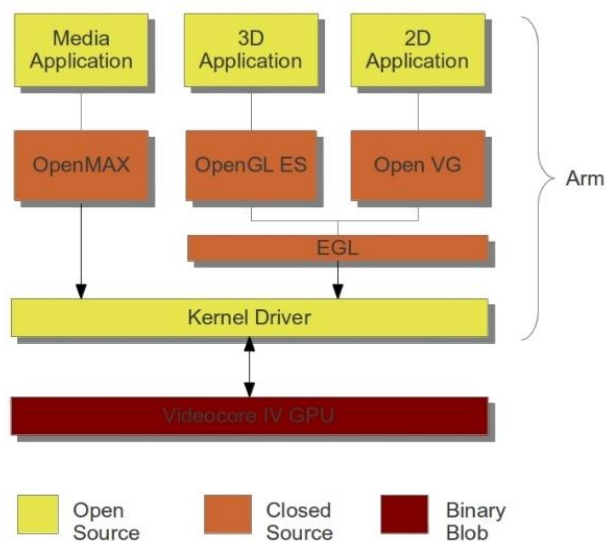


Ilustración 10 Diagrama Distribución Software Raspberry Pi 2. [19]

LUA

Es un lenguaje de programación extensible diseñado para una programación procedimental general con utilidades para la descripción de datos. También ofrece un buen soporte para la programación orientada a objetos, programación funcional y programación orientada a datos. Se pretende que Lua sea usado como un lenguaje de script potente y ligero para cualquier programa que lo necesite. Lua está implementado como una biblioteca escrita en C limpio (esto es, en el subconjunto común de ANSI C y C++).

Siendo un lenguaje de extensión, Lua no tiene noción de programa principal (main): sólo funciona embebido en un cliente anfitrión, denominado programa contenedor o simplemente anfitrión (host). Éste puede invocar funciones para ejecutar un trozo de código Lua, puede escribir y leer variables de Lua y puede registrar funciones C para que sean llamadas por el código Lua. A través del uso de funciones C, Lua puede ser aumentado para abarcar un amplio rango de diferentes dominios, creando entonces lenguajes de programación personalizados que comparten el mismo marco sintáctico. La distribución de Lua incluye un programa anfitrión de muestra denominado lua, que usa la biblioteca de Lua para ofrecer un intérprete de Lua completo e independiente. Lua es software libre, y se proporciona, como es usual, sin garantías, como se establece en su licencia. [20]

Sensor

Un sensor es un dispositivo capaz de detectar magnitudes físicas o químicas, llamadas variables de instrumentación, y transformarlas en variables eléctricas. Las variables de instrumentación pueden ser, por ejemplo: temperatura, intensidad lumínica, distancia, aceleración, inclinación, desplazamiento, presión, fuerza, torsión, humedad, movimiento, pH, etc. Una magnitud eléctrica puede ser una resistencia eléctrica, una capacidad eléctrica (como en un sensor de humedad), una tensión eléctrica (como en un termopar), una corriente eléctrica (como en un fototransistor), etc.

Un sensor se diferencia de un transductor en que el sensor está siempre en contacto con la variable de instrumentación con lo que puede decirse también que es un dispositivo que aprovecha una de sus propiedades con el fin de adaptar la señal que mide para que la pueda interpretar otro dispositivo. Como por ejemplo el termómetro de mercurio que aprovecha la propiedad que posee el mercurio de dilatarse o contraerse por la acción de la temperatura. Un sensor también puede decirse que es un dispositivo que convierte una forma de energía en otra. [21]

Radiación Infrarroja

La radiación infrarroja que existe en el espectro electromagnético a una longitud de onda que es más larga que la longitud de onda visible. La radiación infrarroja no se puede ver, pero si se puede detectar. Los objetos que generan calor también generan radiación infrarroja incluyendo a los animales y el cuerpo humano del cual la radiación más fuerte tiene una longitud de onda de 9.4 micras. Este rango infrarrojo no logra penetrar por muchos tipos de materiales, como lo son el vidrio y el plástico o por una ventana ordinaria. [22]

Sensor Piroeléctrico

El sensor piroeléctrico está hecho de un material cristalino que genera una pequeña carga eléctrica cuando es expuesto al calor en forma de radiación infrarroja. Los elementos del sensor son sensibles a la radiación en un amplio rango entonces se agrega una ventana que actúa como filtro para limitar la radiación de llegada a un rango de 8 a 14 micras donde es más sensible a la radiación del cuerpo humano. [22]

Sensor HC-SR501

El sensor de movimiento infrarrojo HC-SR501 es un sensor ideal para proyectos con la tarjeta Raspberry Pi.

Este sensor “PIR” o sensores de movimiento infrarrojo permiten detectar si existe movimiento dentro de su área de funcionamiento.

Todos los objetos emiten una pequeña cantidad de radiación infrarroja, y cuanto más caliente está un objeto, más radiación infrarroja emite. Los sensores PIR son capaces de detectar pequeños cambios en los niveles de radiación infrarroja en su área de detección.



Ilustración 11 Sensor Pi HC-SR501. [23]

Características.

- Sensor piroeléctrico (Pasivo) infrarrojo (También llamado PIR)
- El módulo incluye el sensor, lente, controlador PIR BISS0001, regulador y todos los componentes de apoyo.
- Rango de detección: 3 m a 7 m, ajustable mediante trimmer (Sx)
- Lente fresnel de 19 zonas, ángulo $< 100^\circ$
- Salida activa alta a 3.3 V
- Tiempo en estado activo de la salida configurable mediante trimmer (Tx)
- Re disparo configurable mediante jumper de soldadura
- Consumo de corriente en reposo: $< 50 \mu\text{A}$
- Voltaje de alimentación: 4.5 VDC a 20 VDC

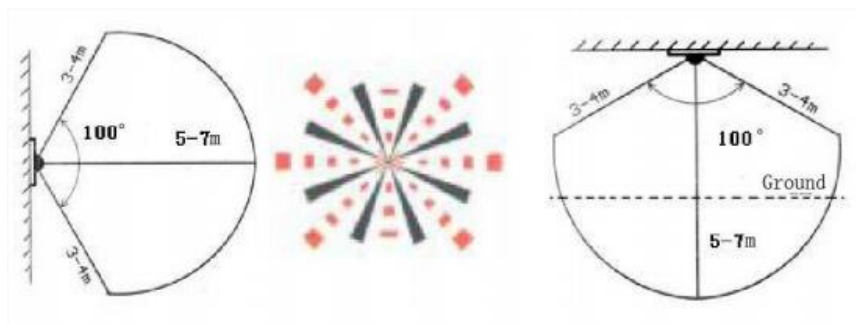


Ilustración 12 Diagrama Rango de Alcance Sensor PIR. [23]

Aplicaciones

- Productos de seguridad
- Iluminación automática
- Automatización y control industrial
- Puertas y timbres automáticos
- Juguetes

Servicios (Daemons)

Los *daemons* (o demonios) son procesos que se ejecuta en segundo plano. Estos demonios ejecutan diferentes funciones y proporcionan ciertos servicios, pero sin la interacción del [usuario](#). Los demonios pueden ser iniciados al arrancar el sistema, al entrar en un *runlevel* (nivel de ejecución) determinado, o simplemente cuando el usuario del sistema los inicie. [24]

Iniciando y parando los demonios o daemons

Los programas que se ejecutan como demonios pueden estar ubicados en cualquier parte del disco, pero tienen un punto en común: todos utilizan un script para ser iniciados/parados, y estos scripts se encuentran en el directorio: **/etc/init.d/** [25]

La sintaxis habitual para iniciar/parar demonios es:

/etc/init.d/<nombre_demonio>start (inicia el demonio)

/etc/init.d/<nombre_demonio> stop (detiene el demonio)

Micrófono

Un micrófono es un aparato que se usa para transformar las ondas sonoras en energía eléctrica y viceversa en procesos de grabación y reproducción de sonido; consiste esencialmente en un diafragma atraído intermitentemente por un electroimán, que, al vibrar, modifica la corriente transmitida por las diferentes presiones a un circuito. Un micrófono funciona como un transductor o sensor electroacústico y convierte el sonido (ondas sonoras) en una señal eléctrica para aumentar su intensidad, transmitirla y registrarla. Los micrófonos tienen múltiples aplicaciones en diferentes campos como en telefonía, ciencia, salud, transmisión de sonido en conciertos y eventos públicos, transmisión de sonido en medios masivos de comunicación como producciones audiovisuales (cine y televisión), radio, producción en vivo y grabado de audio profesional, desarrollo de ingeniería de sonido, reconocimiento de voz y VoIP.

Actualmente, la mayoría de los micrófonos utilizan inducción electromagnética (micrófonos dinámicos), cambio de capacitancia (micrófonos de condensador) o piezoelectricidad (micrófonos piezoeléctricos) para producir una señal eléctrica a partir de las variaciones de la presión de aire. Los Micrófonos usualmente requieren estar conectados a un preamplificador antes de que su señal pueda ser grabada o procesada y reproducida en altavoces o cualquier dispositivo de amplificación sonora. [26]

4.5 Hardware a Utilizar

- Tarjeta Raspberry Pi 2.
- Memoria micro SD de 32Gb.
- Sensor de movimiento piro eléctrico.
- Cámara fotográfica Noir Infrarroja.
- Micrófono dongle.

4.6 Metodología a utilizar

Se utilizará la metodología por prototipos, la cual es una metodología que permite establecer los requisitos del software a partir de la visualización de un producto de software inicial, el cual se llama prototipo.

Este prototipo, dada una especificación, evoluciona hasta tener bien formalizados los requisitos, e inclusive, hasta establecer ese software de prototipo (e un punto de evolución muy alto) como el software del sistema.

El prototipo debe ser fácilmente realizable y fácilmente modificable, para que el cliente y el técnico tengan una visualización oportuna y flexible del producto. Pressman [1992] nos ofrece, como el mismo lo dice, un guion de construcción de prototipos, el cual se explica a continuación.

Un cliente, a menudo, define un conjunto de objetivos generales para el software, pero no identifica los requisitos detallados de entrada, proceso o salida. En otros casos, el responsable del desarrollo del software puede no estar seguro de la eficacia de un algoritmo, de la capacidad de adaptación de un sistema operativo, o de la forma en que debería tomarse la interacción hombre-máquina. En estas y en otras muchas situaciones, un paradigma de construcción de prototipos puede ofrecer un mejor enfoque.

El paradigma de construcción de prototipos (Ilustración13) comienza la recolección de requisitos. El desarrollador y el cliente encuentran y definen los objetivos globales para el software, identifican los requisitos conocidos y las áreas del esquema en donde es obligatoria más definición. Entonces para un “diseño rápido”.

El diseño rápido se centra en una representación de esos aspectos del software que serán visibles para el usuario/cliente. El diseño rápido lleva a la construcción de un prototipo. El prototipo lo evalúa el cliente/usuario y se utiliza para refinar requisitos del software a desarrollar. La iteración ocurre cuando el prototipo se pone a punto de satisfacer las necesidades del cliente, permitiendo al mismo tiempo que el desarrollador comprenda mejor lo que se necesita hacer.

Lo ideal sería que el prototipo sirviera como un mecanismo para identificar los requisitos del software. Si se construye un prototipo de trabajo, el desarrollador intenta hacer usos de los fragmentos o aplica herramientas que permiten generar rápidamente programas de trabajo.



Ilustración 13. Metodología de Construcción por Prototipos. [27]

El prototipo puede servir como “primer sistema”. Aunque esta pudiera ser una versión idealizada. Es verdad que a los clientes y a los que desarrollan les gusta el paradigma de construcción de prototipos. A los usuarios les gusta el sistema real y a los que desarrollan les gusta construir algo inmediatamente. Sin embargo, la construcción de prototipos también puede ser problemática por las siguientes razones:

1. El cliente ve lo que parece ser una versión de trabajo de software, sin tener el conocimiento de que el prototipo es eso un prototipo, que con la prisa de hacer que funcione, no se ha tenido en cuenta la calidad del software global o la facilidad de mantenimiento a largo plazo. Cuando se informa de que el producto se debe construir otra vez para que se puedan mantener los niveles altos de calidad, el cliente no lo entiende y pide que se apliquen “unos pequeños ajustes” que se pueda hacer del prototipo un producto final. De forma demasiado frecuente la gestión de desarrollo del software es muy lenta.
2. El desarrollador, a menudo, hace compromisos de implementación para hacer que el prototipo funcione rápidamente. Se puede utilizar un sistema operativo o lenguaje de programación inadecuado simplemente porque está disponible y porque es conocido; un algoritmo eficiente se puede implementar simplemente para demostrar la capacidad. Después de algún tiempo, el desarrollador debe familiarizarse con estas selecciones, y olvidarse de las razones por las que son inadecuadas. La selección menos ideal ahora es una parte integral del sistema.

Aunque pueden surgir problemas, la construcción de prototipos puede ser un paradigma efectivo para la ingeniería de software. La clave es definir las reglas del juego al comienzo; es decir, el cliente y el desarrollador se deben poner de acuerdo en que el prototipo se construya para servir como un mecanismo de definición de requisitos. [27]

4.7 Casos De Uso

La ilustración 14 muestra los tres actores principales, así como sus respectivas funciones.

El visitante al llegar a la casa-habitación toca el timbre, con la finalidad de saber si está el cliente; enviará mensaje(s) en caso de que el usuario responda al llamado del mismo.

El usuario recibe mensajes a su dispositivo móvil de dos formas: notificación al momento de que el visitante toca el timbre o mensaje de respuesta del visitante; y puede ver una fotografía que es tomada con el sensor para ver quien visita su hogar. Este actor a su vez responderá mensajes como réplica del mensaje del visitante.

El sensor de proximidad solo tomará fotografías cuando alguien esté en un radio predeterminado, las cuales serán enviadas al usuario.

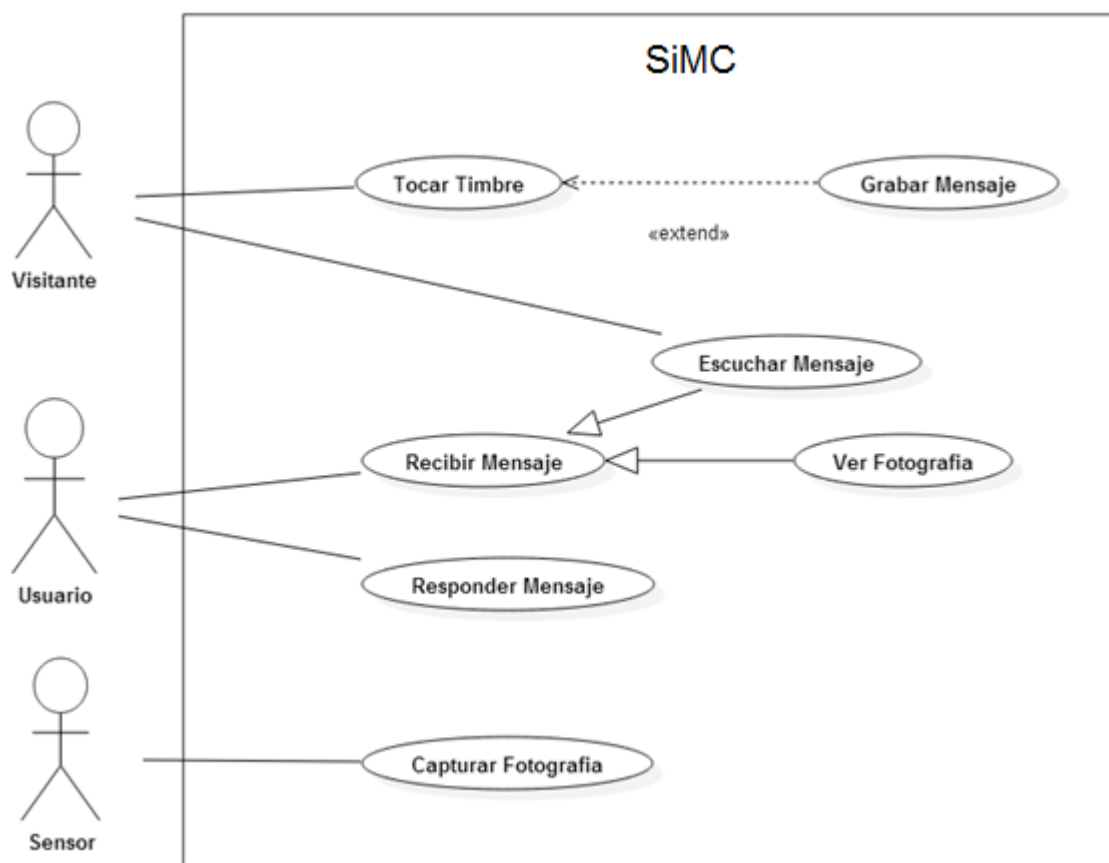


Ilustración 14. Caso de Uso General.

4.7.1 Caso de uso “Tocar el Timbre”

Descripción:

El visitante, al llegar a la casa-habitación, toca el timbre con la finalidad de saber si está el cliente. Grabar Mensaje extiende de Tocar Timbre puesto que no es indispensable que exista una comunicación entre usuario y visitante.

En caso de que el usuario desee entablar una conversación, el visitante deberá mantener presionado el botón para grabar el mensaje que le será enviado al usuario.

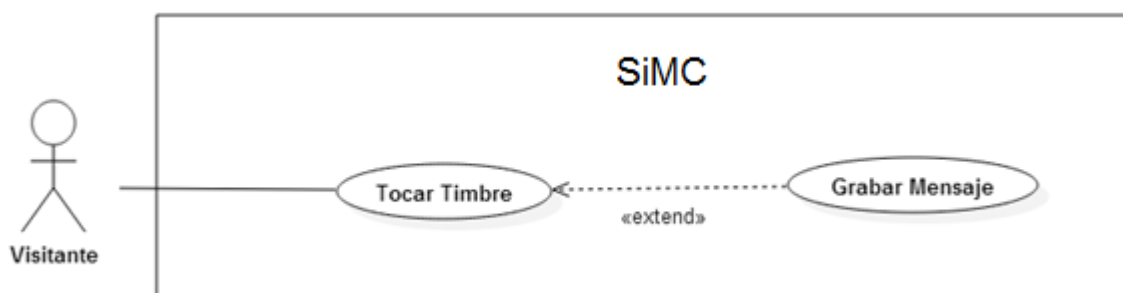


Ilustración 15. Caso de Uso "Tocar el Timbre".

Id Caso de Usos	CU-2
Nombre	Tocar el Timbre
Propósito	Tocar el timbre para saber si alguien habita el lugar.
Actores	Visitante
Entradas	Mecanismo de Activación (timbre).
Salidas	Envío de notificación al cliente. Envío de mensaje de voz.
Pre-condición	Presionar el mecanismo de activación (timbre)
Post-condición	Se recibe respuesta del cliente No se recibe respuesta
Flujo Principal	El visitante llega al domicilio, y presiona el timbre. El usuario de la aplicación, le llegara una notificación correspondiente a la actividad presentada. Si el visitante deja presionado el botón, se ejecutará el caso de uso “Grabar Mensaje”.

Tabla 9. Resumen "Tocar el Timbre"

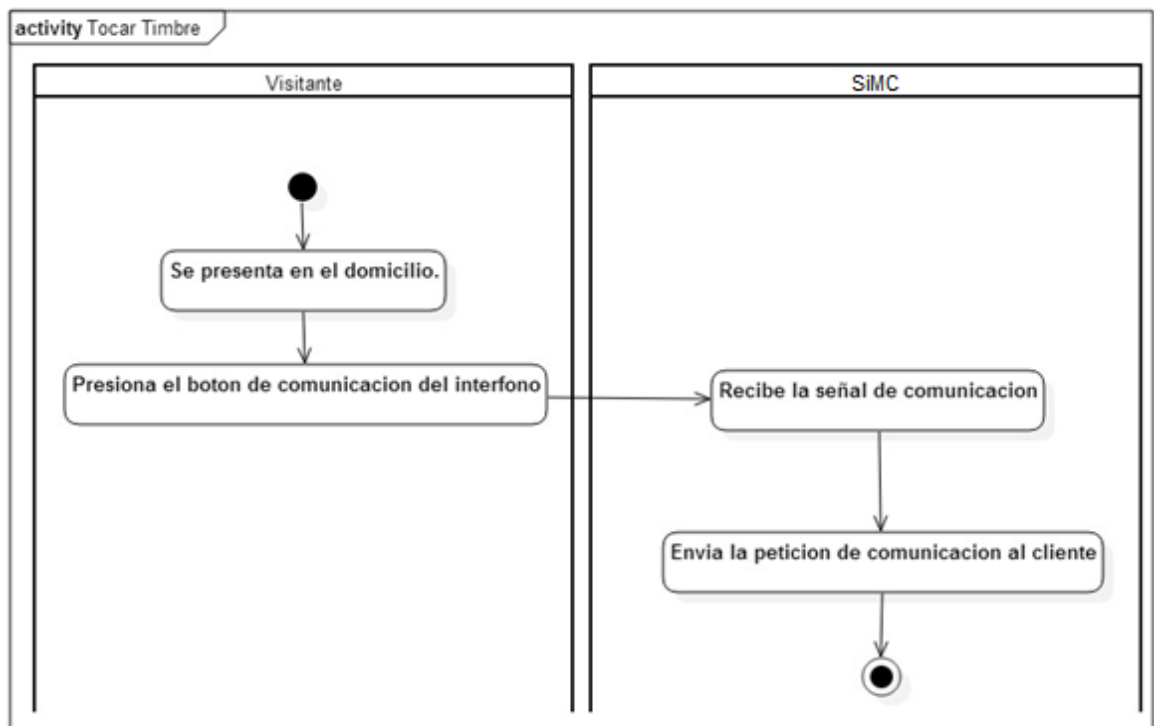


Ilustración 16. Diagrama de Actividades "Tocar Timbre".

4.7.2 Caso de Uso “Recibir Mensaje”

Descripción:

El usuario recibe mensajes a su dispositivo móvil. La primera opción es la notificación que recibe como acción de que el visitante haya tocado el timbre; caso contrario el mensaje de voz que recibe del visitante.

La segunda opción es un mensaje con la fotografía capturada con el sensor, con la finalidad de ver quien visita su hogar.

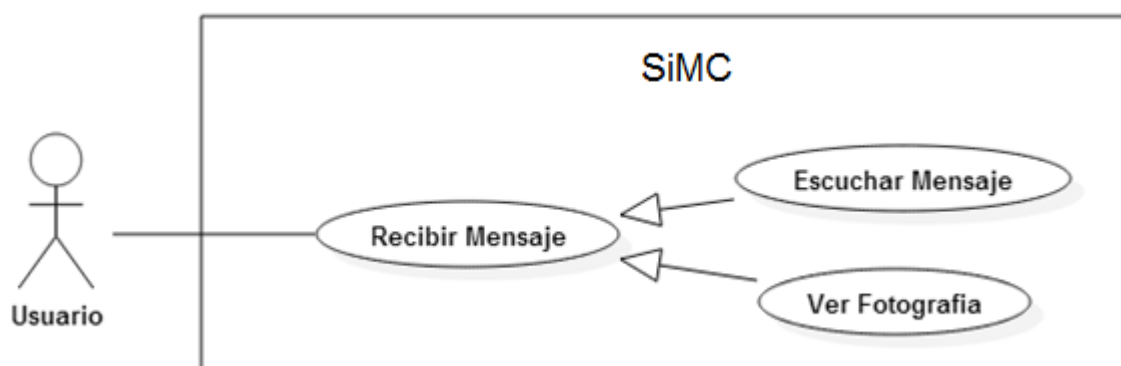


Ilustración 17. Caso de Uso "Recibir Mensaje" Usuario.

Id Caso de Usos	CU-3
Nombre	Recibir Mensaje
Propósito	Recibir Mensaje para notificar actividad presentada en domicilio.
Actores	Usuario
Entradas	Mensaje de voz del Visitante/Notificación. Fotografía
Salidas	Mensaje de voz hacia Visitante.
Pre-condición	El visitante toca el timbre. El visitante Graba mensaje. El sensor captura fotografía.
Post-condición	El usuario responde al Visitante.
Flujo Principal	Se presenta al usuario, una notificación en su celular. El usuario puede elegir si leer dicha notificación o ignorarla. Si el usuario elige leer notificación, se podrá continuar con el caso de uso “Responder Mensaje”.

Tabla 10. Resumen "Recibir Mensaje" Usuario.

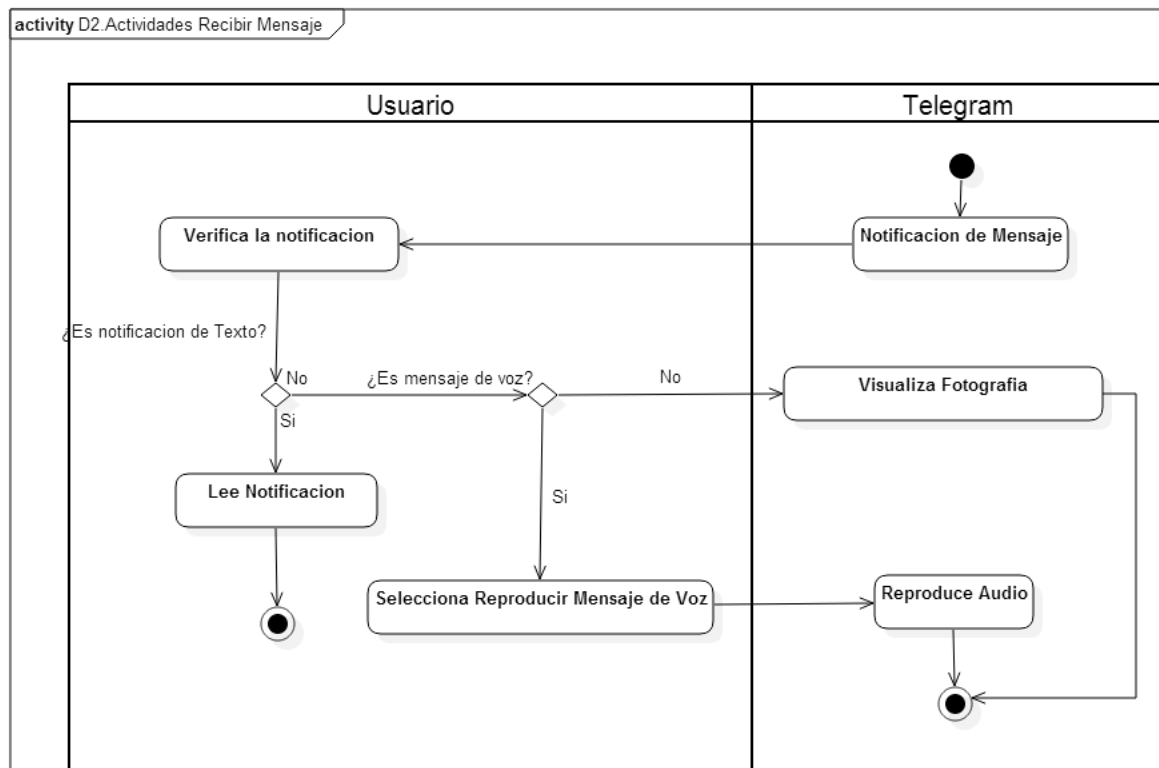


Ilustración 18. Diagrama de Actividades "Recibir Mensaje".

4.7.3 Caso de Uso “Responder Mensaje”

El usuario responderá con un mensaje de voz al visitante para entablar o seguir una conversación.

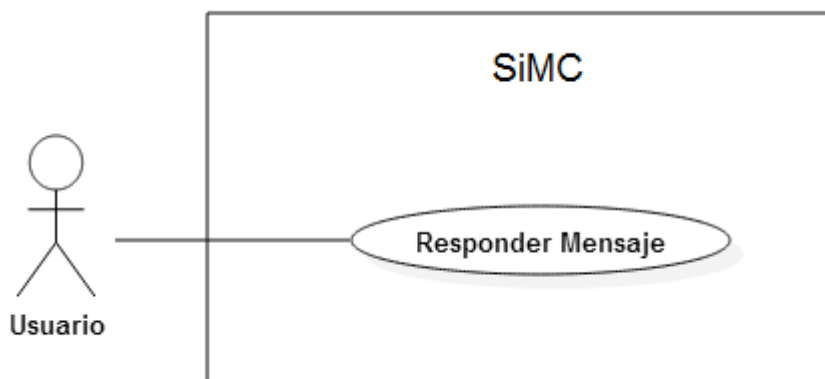


Ilustración 19. Caso de Uso "Responder Mensaje".

Id Caso de Usos	CU-4
Nombre	Responder Mensaje
Propósito	Recibir Mensaje
Actores	Usuario
Entradas	Mensaje de voz del Visitante.
Salidas	Mensaje de voz hacia el Visitante.
Pre-condición	El usuario responde al mensaje del visitante.
Post-condición	El usuario responde al Visitante.
Flujo Principal	<p>El usuario decide leer el mensaje recibido por el visitante. El usuario responde al mensaje del visitante (mensaje de voz).</p> <p>Se reproduce el mensaje en el interfono para el visitante.</p>

Tabla 11. Resumen "Responder Mensaje" Usuario.

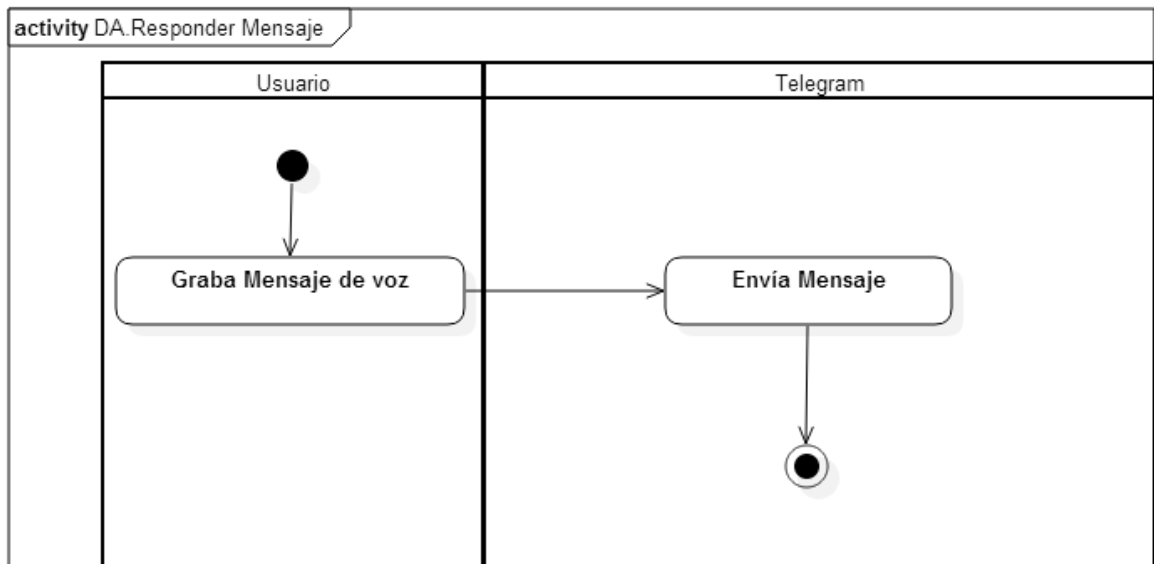


Ilustración 20. Diagrama de Actividades "Responder Mensaje".

4.7.4 Caso de Uso “Escuchar Mensaje”

El visitante escuchara el mensaje de voz, que el usuario le mandara como respuesta a su solicitud.

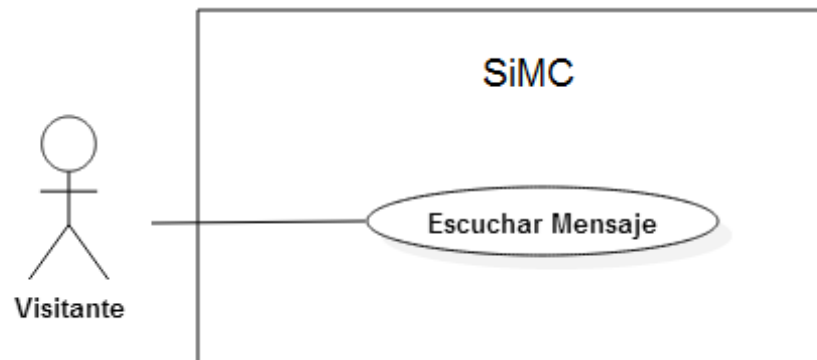


Ilustración 21. Caso de uso “Escuchar Mensaje”.

Id Caso de Usos	CU-5
Nombre	Escuchar Mensaje
Propósito	Escuchar mensaje, para saber la respuesta que el usuario envió.
Actores	Visitante
Entradas	Mensaje de voz del Usuario.
Salidas	Mensaje de voz al Usuario.
Pre-condición	El usuario responde y graba mensaje al visitante.
Post-condición	El Visitante responde al Usuario.
Flujo Principal	<p>El visitante espera por la respuesta del usuario. Una vez que el Usuario graba su mensaje, este se enviará y se reproducirá automáticamente por el interfono.</p> <p>En caso contrario no se reproducirá nada por el interfono.</p>

Tabla 12. Resumen "Escuchar Mensaje" Visitante.

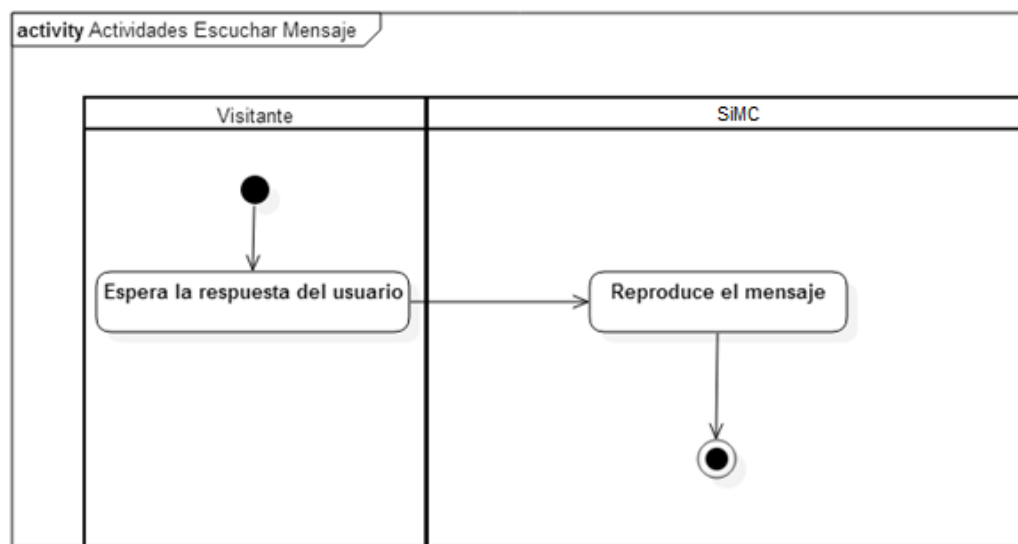


Ilustración 22. Diagrama de Actividades "Escuchar Mensaje".

4.7.5 Caso de Uso “Capturar Fotografía”

El sensor de proximidad solo tomará fotografías cuando alguien esté en un radio de alcance predeterminado. Dichas fotografía serán enviadas al usuario.

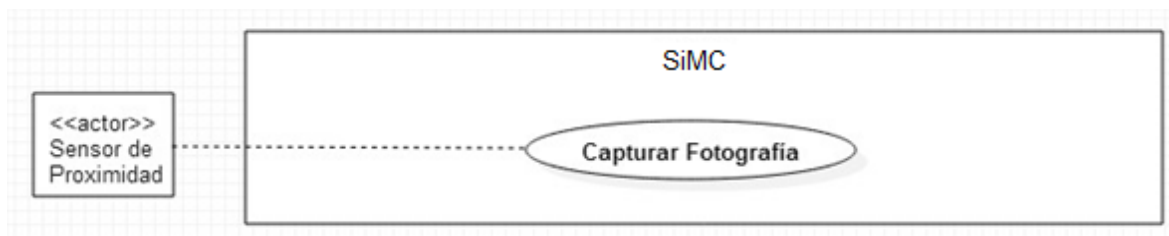


Ilustración 23. Caso de Uso "Capturar Imagen" Transeúnte y Visitante.

Id Caso de Usos	CU-6
Nombre	Capturar Fotografía
Propósito	Capturar fotografía y enviarla al usuario.
Actores	Sensor de Proximidad
Entradas	El sensor detecta movimiento.
Salidas	La fotografía se enviará al usuario.
Pre-condición	Debe haber movimiento en un radio determinado.
Post-condición	La imagen puede o no visualizarla el usuario.
Flujo Principal	Se presenta algún movimiento, el sensor de proximidad lo detecta, y ejecuta la acción de tomar fotografía.

Tabla 13. Resumen "Capturar Fotografía" Sensor.

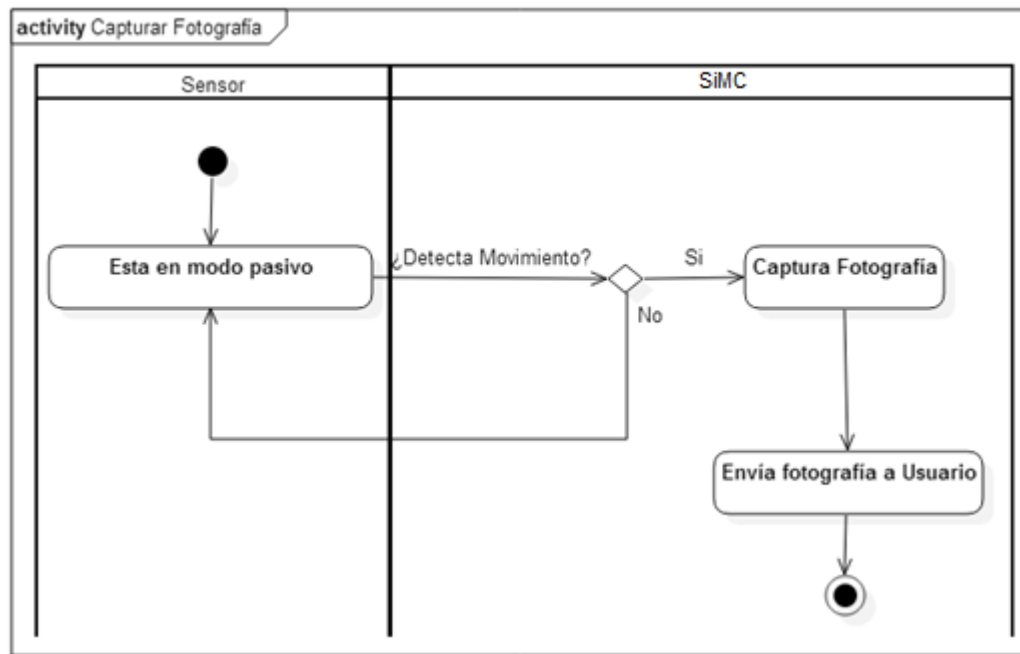
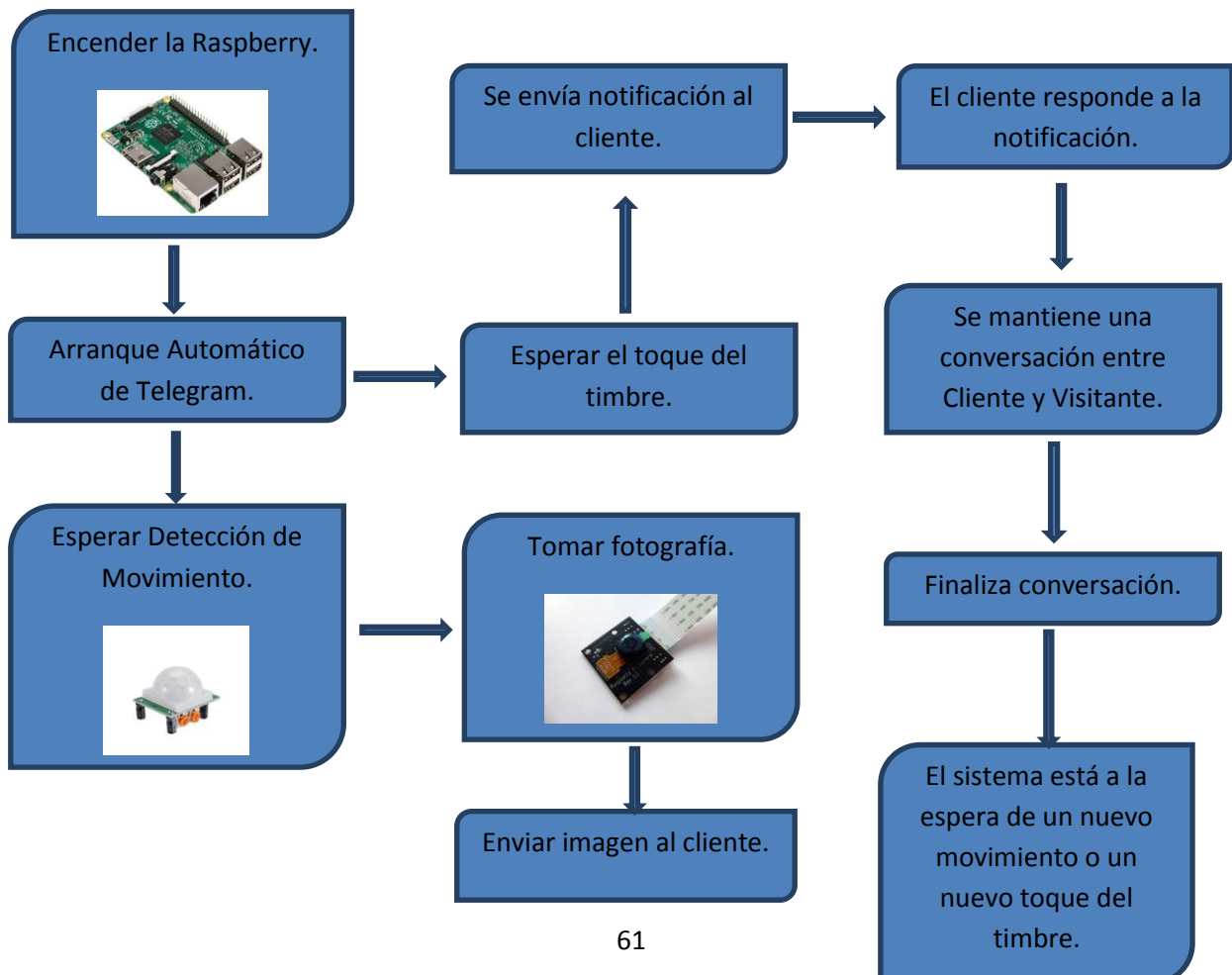


Ilustración 24. Diagrama de Actividades "Capturar Fotografía".

4.8. Diagrama a Bloques

El siguiente diagrama muestra el funcionamiento general del sistema.



Capítulo 5. Implementación

5.1. Prototipo 1

Arranque del sistema operativo en el sistema embebido Raspberry Pi 2

5.1.1. Descripción

Este prototipo muestra la instalación del sistema operativo Raspbian, así también el funcionamiento del sistema operativo en la Raspberry Pi 2.

5.1.2. Objetivo

Instalar y arrancar de manera satisfactoria el sistema operativo Raspbian en el sistema embebido.

5.1.3. Materiales Necesarios

- Tarjeta Micro SD 32Gb Clase 10
- Raspberry Pi 2
- Cable de alimentación
- Mouse
- Equipo portátil con lector de Tarjetas extraíble
- Memoria USB Booteable

5.1.4. Desarrollo

Paso 1. Descargar el software necesario

Se necesita el sistema operativo Raspbian para operar la tarjeta, o en su caso si es la primera vez que se hace uso de la misma. Si el usuario desconoce del sistema operativo que debe usar, se recomienda descargar NOOBS, el cual contiene todos los archivos necesarios para la instalación, así como las imágenes de los S.O. La descarga del material se encuentra en la siguiente URL: <https://www.raspberrypi.org/downloads/>

Paso 2. Extraer y copiar el S.O.

Una vez descargado el ZIP, se extraen todos los archivos y se copian directamente en la tarjeta micro SD.

Paso 3. Instalación y configuración inicial de Raspberry PI

Una vez la tarjeta ha sido preparada y NOOBS ha sido copiado, el sistema operativo se reinicia y el primer programa que se ejecuta es Pi Recovery, este programa sirve para instalar la versión del sistema operativo deseada. En la primera pantalla inicial, se selecciona Raspbian para instalar el sistema operativo.

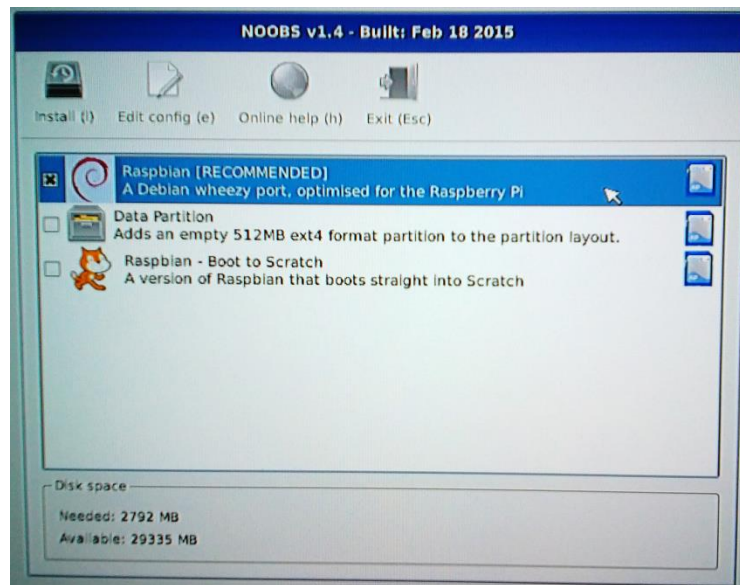


Ilustración 25. Pantalla de instalación del Sistema Operativo Raspbian.

Después que Raspbian se ha sido instalado, hay que configurar el Sistema Operativo para que funcione en español. A continuación, se muestran las opciones disponibles.

Al reiniciar el dispositivo el primer programa que se ejecuta se llama **raspi-config**, este programa solo se ejecuta en inglés. En caso de que queramos realizar una modificación o que hayamos omitido alguna configuración, solo tenemos que ejecutar el siguiente comando desde la terminal una vez instalado el S.O.:

```
1 sudo raspi-config
```

/****MENU PRINCIPAL****/

El primer menú que muestra contiene 9 diferentes opciones disponibles, a continuación, describimos cada una de las opciones disponibles:

/**Opción 1 – Expandir el Sistema de Archivos (Expand Filesystem) **/

Esta opción permite expandir el sistema operativo para que utilice todo el espacio disponible en la tarjeta. Cuando se instala Raspbian la imagen copiada en la tarjeta solo ocupa 2 GB, por lo tanto, es necesario ejecutar esta opción para que todo el espacio de la tarjeta SD sea utilizado.

Como utilizamos NOOBS para instalar el Sistema Operativo, no es necesario ejecutar esta opción ya que el sistema operativo ha sido expandido.

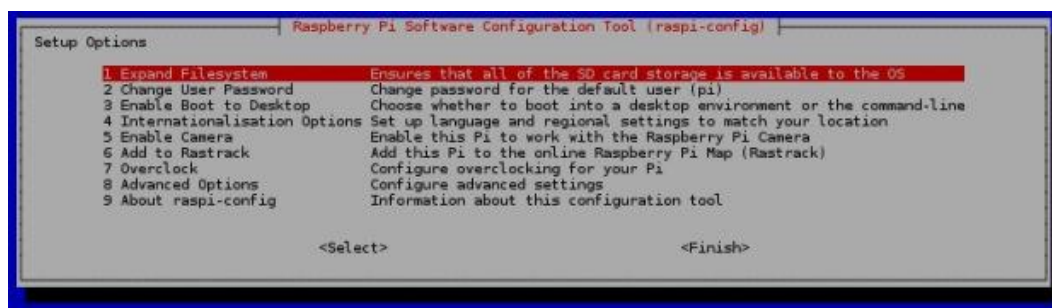


Ilustración 26. Menú Principal de Configuración del Sistema Operativo Raspbian.

/Opción 2 – Cambiar la contraseña del usuario (Change User Password) **/**

En el Raspberry Pi, y en general en sistemas Linux, existen diferentes tipos de usuario. Los dos que vienen predeterminados por el sistema son los usuarios “root” y “pi”

El más importante que es el administrador del sistema que se llama “root”, este tiene acceso privilegiado a todos los archivos, configuraciones y carpetas del sistema. El otro tipo de usuario son los comunes como lo es “pi”, este viene predeterminado con la contraseña “raspberry” por lo tanto cualquier persona podría acceder su sistema.

/Opción 3 – Activar el escritorio al iniciar (Entable Boot to Desktop) **/**

Esta opción permite que la Raspberry Pi, después de iniciar el sistema, comience inmediatamente el escritorio modo gráfico o en línea de comando. En caso de que inicie en modo de comando y después queramos ingresar al modo gráfico solo ingresamos el siguiente código:

1 Startx

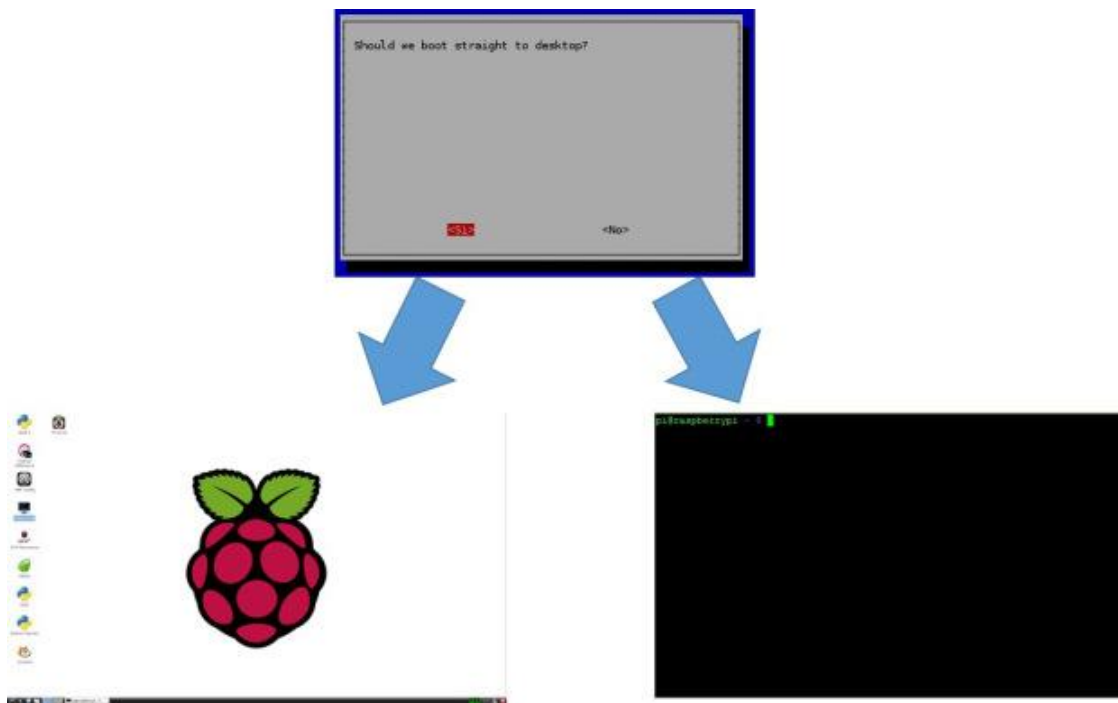


Ilustración 27. Opción 3 – Iniciar el S.O. con interfaz gráfica.

*/**Opción 4– Opciones de internacionalización (Internationalisation Options) **/*

Esta opción nos permite modificar el lenguaje del sistema operativo, la zona horaria y la distribución del teclado. Para nuestro proyecto establecemos México, TM -06:00Hrs. Y Teclado Latinoamericano.

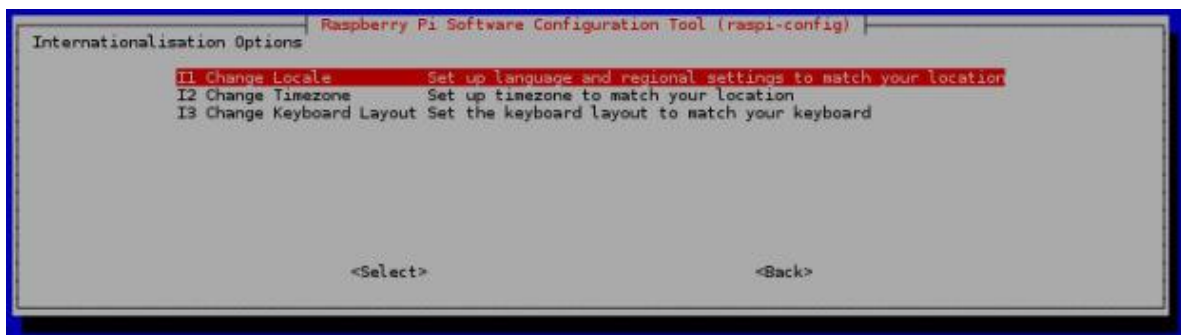


Ilustración 28. Menú principal de la Opción 4.

La primera opción sirve para indicar donde se encuentra ubicada la Raspberry, esta opción configura el lenguaje del sistema operativo, los caracteres, la denominación de la moneda, etc. Modo de codificación para el sistema:

es_MX. UTF-8 UTF-8

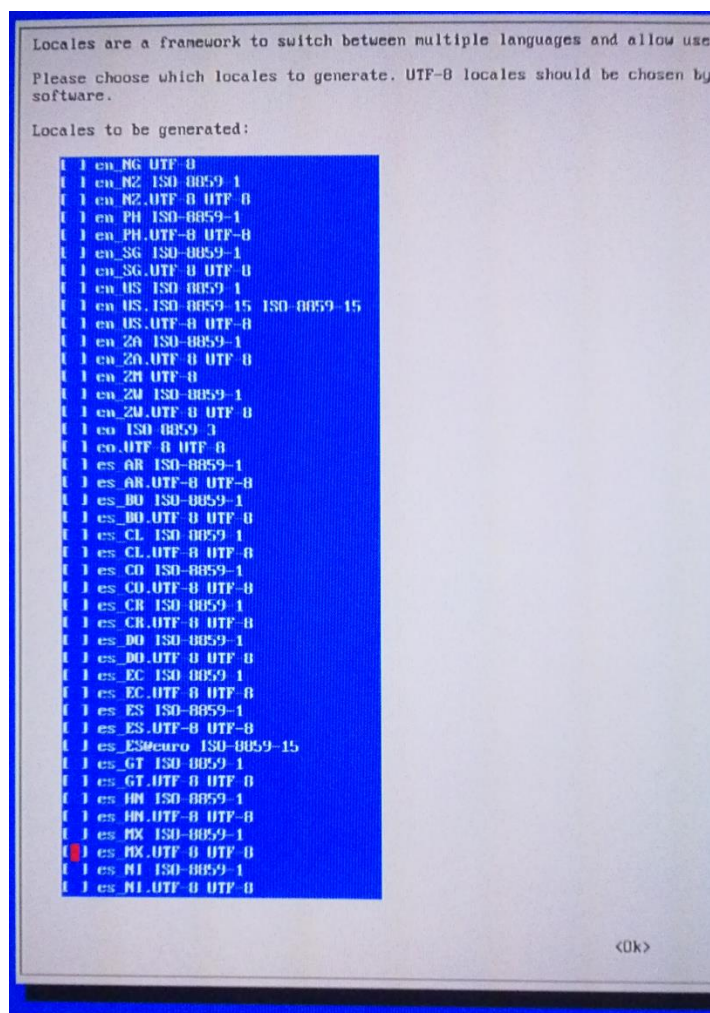


Ilustración 29. Establecer el idioma del Sistema.

La opción 2 sirve para cambiar la zona horaria del sistema la cual se ajusta de acuerdo a la ciudad donde estamos actualmente. Para este caso es: América->México

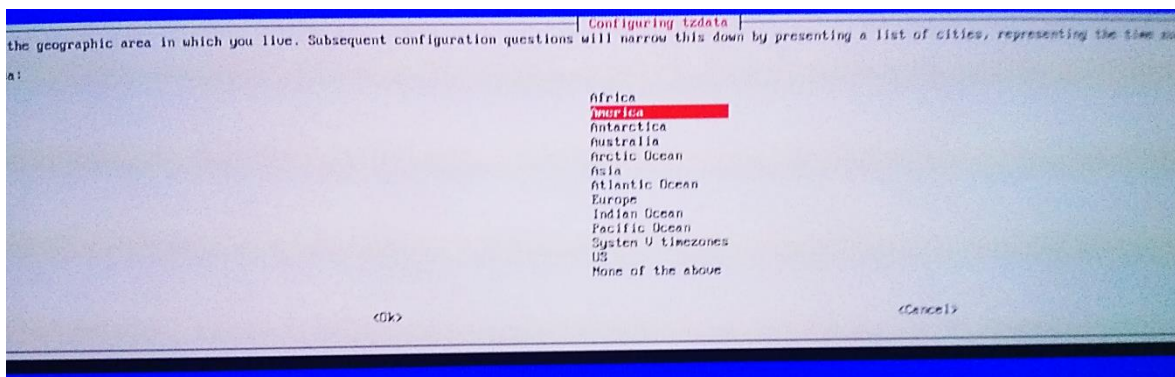


Ilustración 30. Zona horaria del sistema por continente.

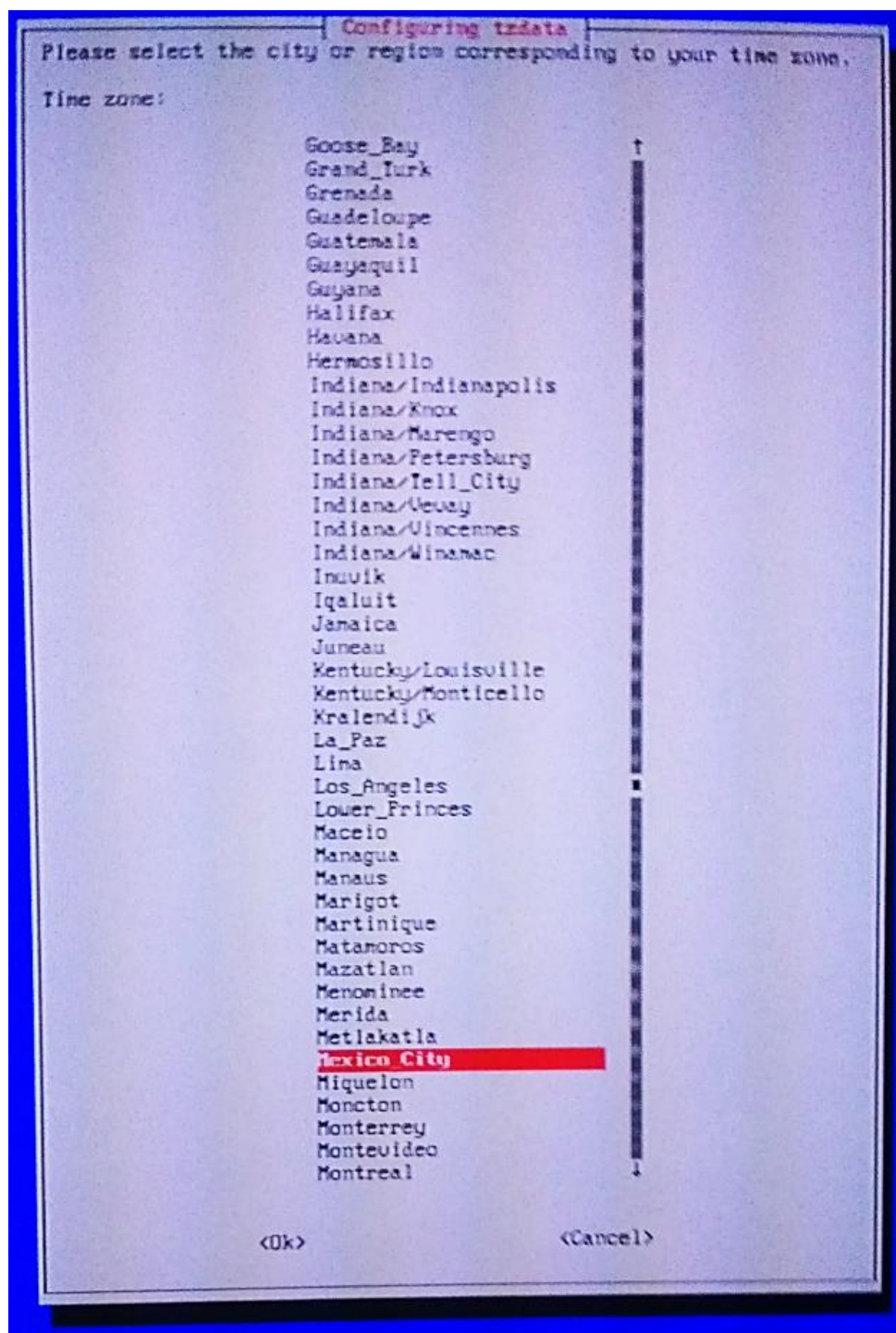


Ilustración 31. Zona horaria del sistema por país.

La opción 3 permite cambiar la configuración del teclado que estemos usando, si la marca y el tipo del teclado no aparece, se selecciona “PC genérico de 105 teclas”.

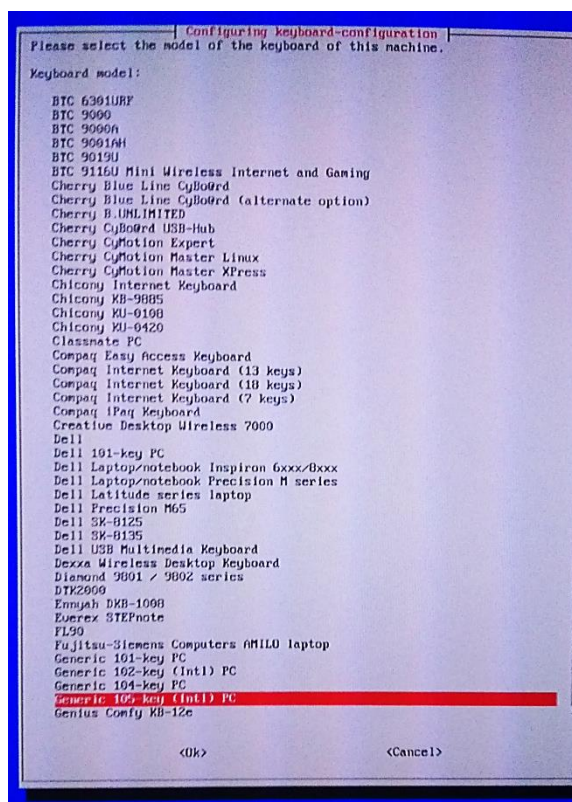


Ilustración 32. Configuración del Teclado por marca.

Se selecciona la distribución del teclado. Se recomienda dejar la opción predeterminada “Español Latinoamericano”



Ilustración 33. Configuración del teclado por idioma y caracteres especiales.

/Opción 5 – Activar la cámara (Enable Camera) **/**

Esta opción sirve para dar soporte a la cámara de Raspberry Pi, nos permite activar el puerto para que haya comunicación entre la CPU y el controlador de la cámara.

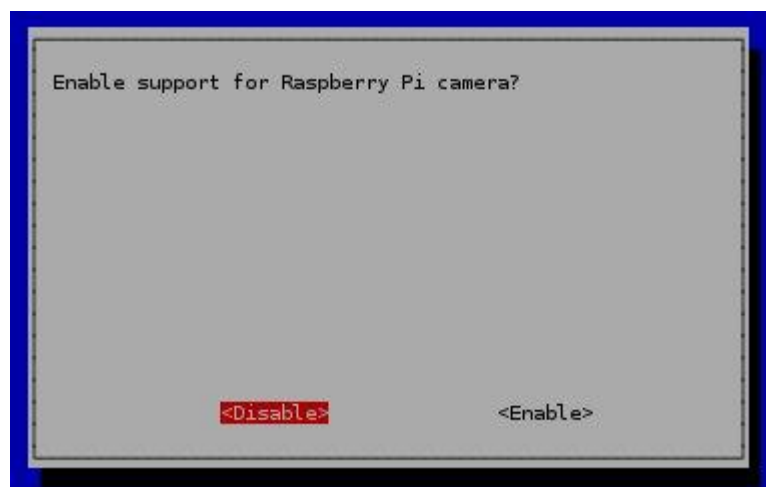


Ilustración 34. Habilitar la cámara para el sistema embebido.

*/**Opción 6 – Adicionar a Rastrack (Add to Rastrack) **/*

Esta opción permite que la Raspberry Pi sea rastreada por el sitio web Rastrack (<http://rastrack.co.uk>), este sitio no pretende registrar o recolectar información alguna. Es una herramienta para tener la estadística de donde se encuentran los Raspberry Pi en el mundo. Para nuestro proyecto omitimos esta opción.

*/**Opción 7 –Overclocking**/*

Esta opción permite nos aumentar la velocidad del procesador. Como nuestro proyecto no requiere de procesos que ocupen toda la memoria RAM y/o el procesador, omitimos también este paso.

*/**Opción 8 – Opciones avanzadas (Advanced Options) **/*



Ilustración 35. Submenú de la opción 8.

La opción A1 (Overscan) sirve para borrar las líneas negras en algunos monitores o televisores.

La opción A2 (Hostname), sirve para identificar la Raspberry Pi en nuestra red local. Por practicidad la dejaremos por default.

La opción A3 (Distribución de la memoria - Memory Split) nos permite seleccionar la cantidad de memoria compartida entre la CPU y la unidad de gráficos (GPU). Como nuestro proyecto no requiere de una pantalla, no es necesario configurar esta opción.

La opción A4 (Activar SSH - Enable SSH) se utiliza para acceder el Raspberry Pi remotamente desde un cliente SSH. SSH significa “Secure SHell” el cual es una forma segura de conectarse al Raspberry Pi a través de la red. Activamos esta opción ya que nuestro proyecto requerirá de esta función, ya que no necesitaremos utilizar ni un monitor, ni teclado, ni mouse adicionales para poder controlar nuestro dispositivo.

La última opción A5 (Actualizar - Update) se utiliza para descargar una actualización del sistema, librerías, etc. Solo basta con estar conectados a la red y utilizar el siguiente comando:

1 <code>sudo apt-get update</code>

5.2. Prototipo 2

Arranque de la aplicación de mensajería instantánea (Telegram) en el sistema embebido Raspberry Pi 2.

5.2.1. Descripción

Este prototipo muestra la instalación y el funcionamiento de Telegram en el sistema embebido.

5.2.2. Objetivo

Instalar y arrancar la aplicación de mensajería instantánea (Telegram), así como también hacer pruebas de envío de mensajes de texto a cualquier dispositivo disponible.

5.2.3. Materiales Necesarios

- Raspberry Pi 2
- Cable de alimentación
- Mouse
- Teclado
- Dispositivo móvil

5.2.4. Desarrollo

Paso 1. Descargar e instalar Telegram en Raspberry

Por recomendación es necesario actualizar el sistema con los siguientes comandos:

```
1 sudo apt-get update
2 sudo apt-get upgrade
```

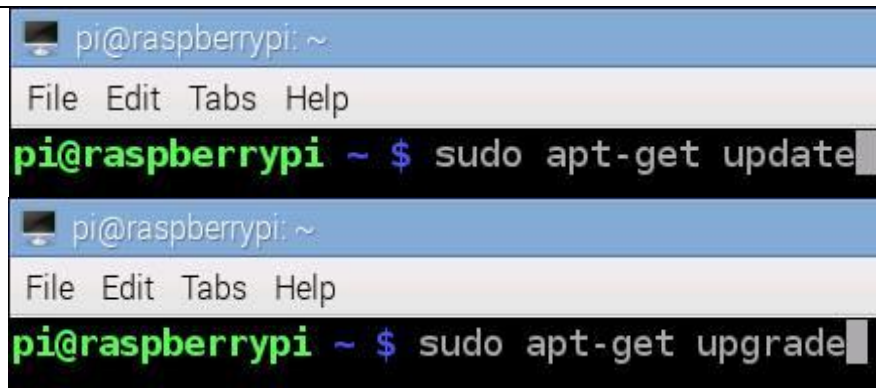
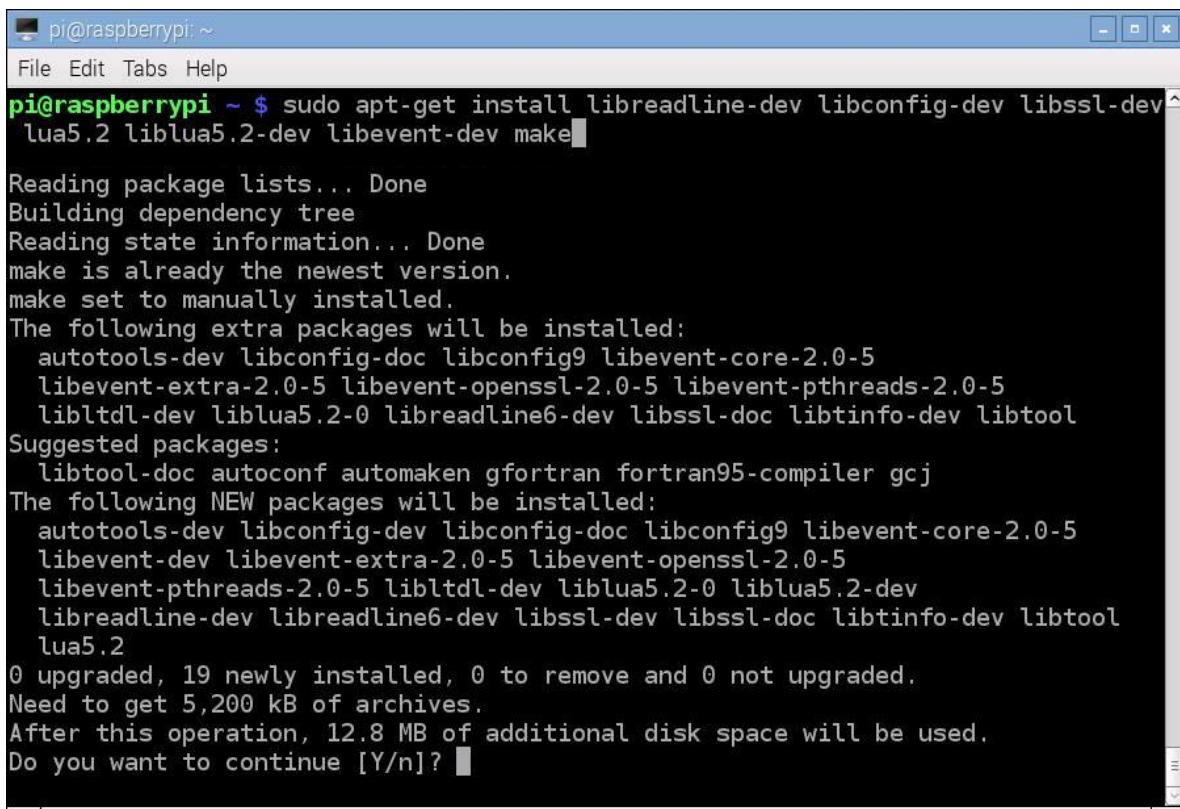


Ilustración 36. Comandos para actualizar el S.O.

Posteriormente se necesitan algunas librerías para utilizar telegram, haciendo uso del siguiente comando:

```
1 sudo apt-get install libreadline-dev libconfig-dev libssl-dev lua5.2 liblua5.2-dev
libevent-dev libnotify-bin lua-lgi glib-2.0 libjansson-dev libpython-dev make
```



```
pi@raspberrypi ~ $ sudo apt-get install libreadline-dev libconfig-dev libssl-dev lua5.2 liblua5.2-dev libevent-dev make
Reading package lists... Done
Building dependency tree
Reading state information... Done
make is already the newest version.
make set to manually installed.
The following extra packages will be installed:
  autotools-dev libconfig-doc libconfig9 libevent-core-2.0-5
  libevent-extra-2.0-5 libevent-openssl-2.0-5 libevent-pthreads-2.0-5
  libltdl-dev liblua5.2-0 libreadline6-dev libssl-doc libtinfo-dev libtool
Suggested packages:
  libtool-doc autoconf automake gfortran fortran95-compiler gcj
The following NEW packages will be installed:
  autotools-dev libconfig-dev libconfig-doc libconfig9 libevent-core-2.0-5
  libevent-dev libevent-extra-2.0-5 libevent-openssl-2.0-5
  libevent-pthreads-2.0-5 libltdl-dev liblua5.2-0 liblua5.2-dev
  libreadline-dev libreadline6-dev libssl-dev libssl-doc libtinfo-dev libtool
  lua5.2
0 upgraded, 19 newly installed, 0 to remove and 0 not upgraded.
Need to get 5,200 kB of archives.
After this operation, 12.8 MB of additional disk space will be used.
Do you want to continue [Y/n]? █
```

Ilustración 37. Librerías para Telegram.

Después ejecutaremos unas líneas de comando para configurar que la pantalla no entre en modo suspensión, ya que esa opción deshabilita por completo la tarjeta Raspberry.

```
1 sudo nano /etc/kbd/config
```

Una vez abierto el archivo buscamos la siguiente línea y la ponemos como se muestra continuación:

```
- BLANK_TIME=0
- POWERDOWN_TIME=0
```

Por ultimo solo resta descargar telegram desde GitHub.com, hacer. /configure y make. Todo esto se describe a continuación con los comandos:

```
1 git clone --recursive https://github.com/vysheng/tg.git && cd tg
```



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi ~ $ git clone --recursive https://github.com/vysheng/tg.git && cd telegram  
Cloning into 'tg'...  
remote: Counting objects: 3513, done.  
remote: Total 3513 (delta 0), reused 0 (delta 0), pack-reused 3513  
Receiving objects: 100% (3513/3513), 2.45 MiB | 874 KiB/s, done.  
Resolving deltas: 100% (2379/2379), done.  
Submodule 'tgl' (https://github.com/vysheng/tgl.git) registered for path 'tgl'  
Cloning into 'tgl'...  
remote: Counting objects: 702, done.  
remote: Total 702 (delta 0), reused 0 (delta 0), pack-reused 702  
Receiving objects: 100% (702/702), 685.70 KiB | 634 KiB/s, done.  
Resolving deltas: 100% (490/490), done.  
Submodule path 'tgl': checked out 'b3dcce35110f5c995366318c2886065287815d09'  
Submodule 'tl-parser' (https://github.com/vysheng/tl-parser) registered for path 'tl-parser'  
Cloning into 'tl-parser'...
```

Ilustración 38. Descargar Telegram para Raspberry.

1 ./configure

```
pi@raspberrypi: ~/tg  
File Edit Tabs Help  
pi@raspberrypi ~/tg $ ./configure  
checking for C compiler default output file name... a.out  
checking for suffix of executables...  
checking whether we are cross compiling... no  
checking for suffix of object files... o  
checking whether we are using the GNU C compiler... yes  
checking whether gcc accepts -g... yes  
checking for gcc option to accept ISO C89... none needed  
checking for sqrt in -lm... yes  
checking for library containing clock_gettime... -lrt  
checking for library containing backtrace... none required  
checking for event_base_new in -levent... yes  
checking how to run the C preprocessor... gcc -E  
checking for grep that handles long lines and -e... /bin/grep  
checking for egrep... /bin/grep -E  
checking for ANSI C header files... yes  
checking for sys/types.h... yes  
checking for sys/stat.h... yes  
checking for stdlib.h... yes  
checking for string.h... yes  
checking for memory.h... yes  
checking for strings.h... yes  
checking for inttypes.h... yes  
checking for stdint.h...
```

Ilustración 39. Crear el archivo makefile.

Antes de compilar Telegram en la ruta home/pi/tg/tgl, modificamos los archivos queries.c y strcutures.c para corregir errores de audio y video.

En el archivo queries.c quitamos la siguiente línea:

```
2489 D->type = CODE_input_document_file_location;
```

Y agregamos las siguientes líneas:

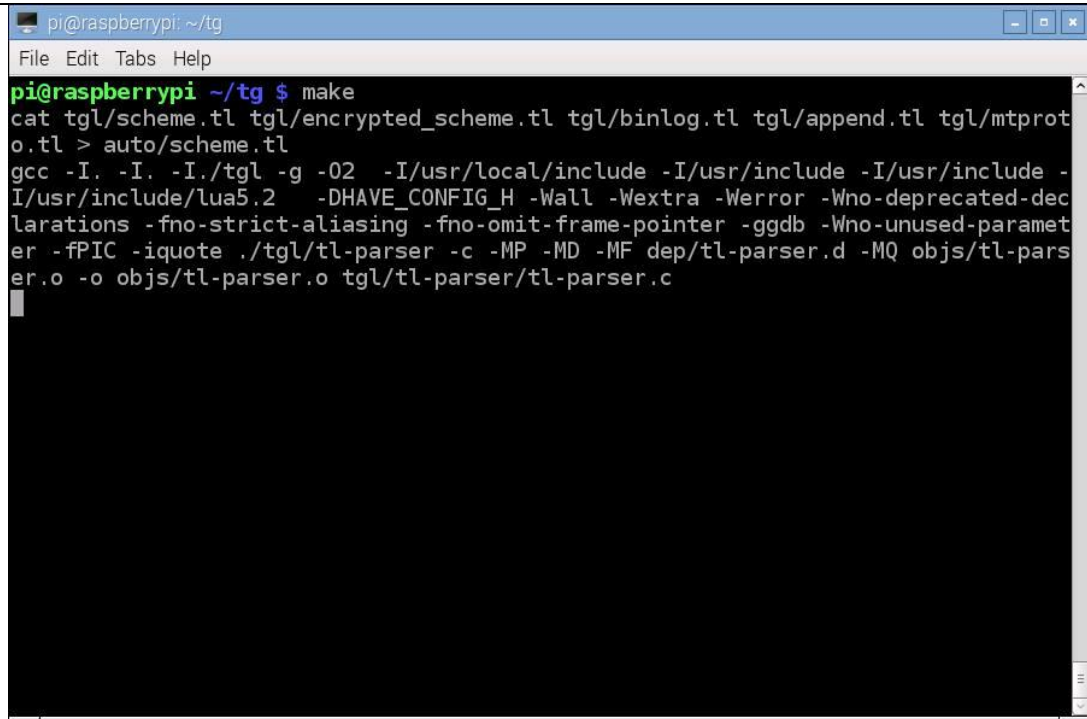
```
2489 if (V->flags & TGLDF_VIDEO) {  
2490     D->type = CODE_input_video_file_location;  
2491 } else if (V->flags & TGLDF_AUDIO) {  
2492     D->type = CODE_input_audio_file_location;  
2493 } else {  
2494     D->type = CODE_input_document_file_location;  
2495 }
```

Una vez terminado de agregar el código, guardamos y abrimos el archivo structure.c y agregamos la siguiente línea:

```
605 D->flags = TGLDF_VIDEO;  
606
```

Finalmente compilamos Telegram.

1 make



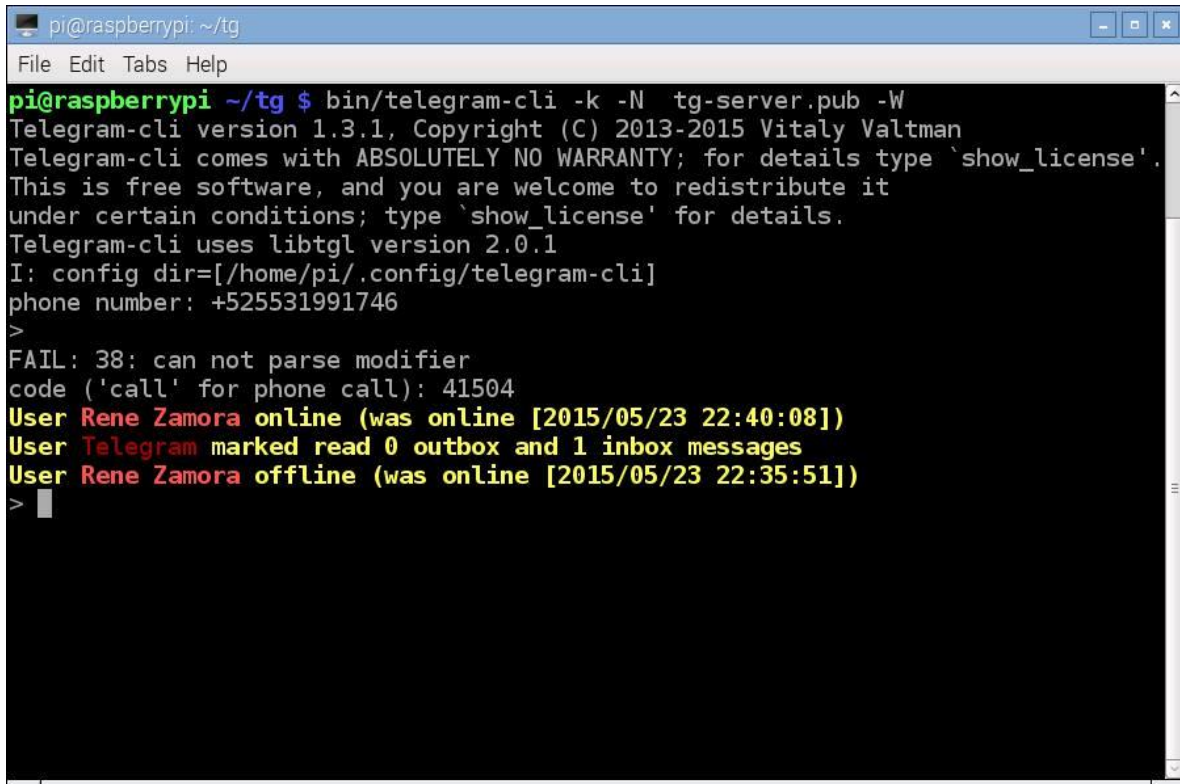
```
pi@raspberrypi: ~/tg  
File Edit Tabs Help  
pi@raspberrypi ~/tg $ make  
cat tgl/scheme.tl tgl/encrypted_scheme.tl tgl/binlog.tl tgl/append.tl tgl/mtprot  
o.tl > auto/scheme.tl  
gcc -I. -I. -I./tgl -g -O2 -I/usr/local/include -I/usr/include -I/usr/include -  
I/usr/include/lua5.2 -DHAVE_CONFIG_H -Wall -Wextra -Werror -Wno-deprecated-dec  
larations -fno-strict-aliasing -fno-omit-frame-pointer -ggdb -Wno-unused-paramet  
er -fPIC -iquote ./tgl/tl-parser -c -MP -MD -MF dep/tl-parser.d -MQ objs/tl-pars  
er.o -o objs/tl-parser.o tgl/tl-parser/tl-parser.c
```

Ilustración 40. Compilación de Telegram.

Paso 2. Ejecutar la aplicación de mensajería instantánea y pruebas de mensajes

Para abrirla aplicación solo es necesario el siguiente comando:

```
1 bin/telegram-cli -k -N tg-server.pub
```



```
pi@raspberrypi: ~/tg
File Edit Tabs Help
pi@raspberrypi ~/tg $ bin/telegram-cli -k -N tg-server.pub -W
Telegram-cli version 1.3.1, Copyright (C) 2013-2015 Vitaly Valtman
Telegram-cli comes with ABSOLUTELY NO WARRANTY; for details type `show_license'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show_license' for details.
Telegram-cli uses libtgc version 2.0.1
I: config dir=[/home/pi/.config/telegram-cli]
phone number: +525531991746
>
FAIL: 38: can not parse modifier
code ('call' for phone call): 41504
User Rene Zamora online (was online [2015/05/23 22:40:08])
User Telegram marked read 0 outbox and 1 inbox messages
User Rene Zamora offline (was online [2015/05/23 22:35:51])
>
```

Ilustración 41. Ejecución de Telegram.

Si es la primera vez que se instala la aplicación, el sistema pedirá un número de teléfono el cual estará asociado a la aplicación y así poder entablar comunicación con otros dispositivos.

Pedirá además el código de verificación, el cual será enviado como mensaje de texto al teléfono registrado en el paso anterior.

Para empezar una conversación solo basta con escribir en la terminal el siguiente comando:

1 msg <peer> <text>; Si el usuario tiene espacios, sustituirlo por “_”
--

```
pi@raspberrypi: ~/tg/tg
File Edit Tabs Help
with system default action
view_file_thumb <msg-id> Downloads file to downloads dirs. Then tries to
open it with system default action
view_photo <msg-id> Downloads file to downloads dirs. Then tries to open it
with system default action
view_user_photo <user> Downloads file to downloads dirs. Then tries to open it
with system default action
view_video <msg-id> Downloads file to downloads dirs. Then tries to open it
with system default action
view_video_thumb <msg-id> Downloads file to downloads dirs. Then tries to
open it with system default action
view <msg-id> Tries to view message contents
visualize_key <secret chat> Prints visualization of encryption key (first 16
bytes sha1 of it in fact)
> Leo Rosas Santillan
FAIL: 38: can not find comamnd 'Leo'
> msg Leo_Rosas_Santillan SyPR0c: Mensaje Enviado desde RaspberryPi2
*** reply_id=0, disable=0
139 [11:04] Leo Rosas Santillan <<< SyPR0c: Mensaje Enviado desde RaspberryPi2
>
```

Ilustración 42. Prueba de envío de mensaje de texto a dispositivo móvil.



Ilustración 43. Mensaje desde Raspberry.

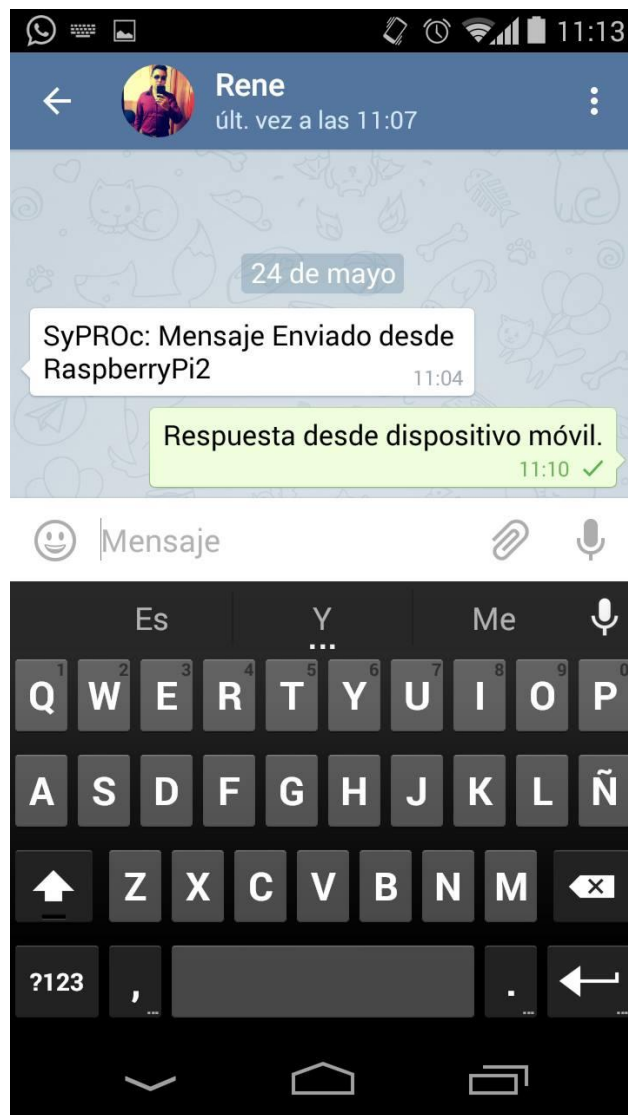


Ilustración 44. Prueba de envío de mensaje a Raspberry.

```

pi@raspberrypi: ~/tg/tg
File Edit Tabs Help
[0%Down]> quit
halt
pi@raspberrypi ~/tg/tg $ bin/telegram-cli -N tg-server.pub
Telegram-cli version 1.3.1, Copyright (C) 2013-2015 Vitaly Valtman
Telegram-cli comes with ABSOLUTELY NO WARRANTY; for details type `show_license'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show_license' for details.
Telegram-cli uses libtgml version 2.0.1
I: config dir=[/home/pi/.config/telegram-cli]
141 [11:10] Leo Rosas Santillan >>> Respuesta desde dispositivo móvil.
>

```

Ilustración 45. Mensaje recibido desde dispositivo móvil.

5.3 Prototipo 3 Respuesta Automática de mensajes.

Respuesta automática de mensajes de Telegram en el sistema embebido Raspberry Pi 2

5.3.1 Descripción

Este prototipo muestra la programación de la respuesta automática para los mensajes de Telegram del sistema embebido, utilizando el lenguaje de programación Python, para definir el comportamiento que se tendrá al recibir determinados mensajes en la conversación de Telegram.

El archivo Alarma.lua, contendrá el comportamiento de cada uno de los mensajes definidos, de esta manera al teclear alguno de esos mensajes, se realizara una acción en específico.

5.3.2 Objetivo

Programar en Python el comportamiento que tendrán los mensajes automatizados al recibir determinados mensajes mediante Telegram por parte del usuario.

5.3.1 Desarrollo

Código Alarma.lua

```
USER_ALLOWED = "Leonardo_Rosas" --Usuario al que se permitira el control remoto de la aplicacion.
function on_msg_receive (msg) -- Esta funcion se ejecutara cuando el mensaje recibido sea de
    -- tipo mensaje de texto

    if msg.out then
        return
    end

    -- Allowed only our ID's
    if (msg.from.print_name == USER_ALLOWED) then -- Se realiza la validacion de que solo se ejecutara
        -- cuando el usuario sea identico
        -- al que se le autorizo para controlar la aplicacion.
        if (msg.text=='ping' or msg.text=='Ping') then -- Este proceso se ejecutara cuando la cadera recibida
            -- sea identica a ping
            send_msg (msg.from.print_name, 'Mensaje Automatico de Respuesta', ok_cb, false) -- Si se valida de forma
            -- correcta se manda una respuesta
            -- automatica de texto al usuario.

        end

        if (msg.text=='escom' or msg.text=='Escom') then -- Este proceso se ejecutara cuando la cadera recibida
            -- sea identica a escom
            send_photo(msg.from.print_name, '/home/leonardo/tg/img/escom.jpg', ok_cb, false) --Se enviara la imagen
            -- correspondiene al dispositivo
            --movil del usuario

            send_msg (msg.from.print_name, 'Enviando imagen', ok_cb, false)

        end

        if (msg.text=='ipn' or msg.text=='Ipn') then -- Este proceso se ejecutara cuando la cadera recibida
            -- sea identica a ipn
            send_photo(msg.from.print_name, '/home/leonardo/tg/img/ipn.png', ok_cb, false) --Se enviara la imagen
            -- correspondiene al dispositivo
            --movil del usuario

            send_msg (msg.from.print_name, 'Enviando imagen', ok_cb, false)

        end

        if (msg.text=='audio' or msg.text=='Audio') then -- Este proceso se ejecutara cuando la cadera recibida sea identica a audio
            os.execute('lua5.2 ' .. SCRIPTS_FOLDER .. 'video.lua')
            send_audio(msg.from.print_name, '/home/leonardo/.telegram-cli/downloads/download_355823914326163496.oga', ok_cb, false)

        end

    end
end
```

Ilustración 46 Código Python para Automatización de Mensajes.

Pruebas:

Para poder ejecutar Telegram-cli en la Raspberry es necesario ejecutar el siguiente comando.

```
leonardo@leonardo-VirtualBox: ~/tg$ bin/telegram-cli -N -k tg-server.pub -W -s scripts/Alarma.lua
```

Ilustración 47 Comando para ejecutar Telegram.

Donde:

Bin/Telegram-cli: Es el cliente de la aplicación de Telegram.

N: Muestra el Id del mensaje que se envíen o reciban.

K: Para conectarse al servidor de telegram mediante la llave asignada.

Tg-server.pub: Servidor al que se conectara el cliente.

S: Significa que se usara un script que va a interactuar con la aplicación

Scripts/Alarma.lua: Ruta del Script que se ejecutara.

Cuando de Enter al comando nos mostrara la siguiente pantalla, donde nos dita que se ha inicializado de forma correcta Telegram.

```
leonardo@leonardo-VirtualBox: ~/tg$ bin/telegram-cli -N -k tg-server.pub -W -s scripts/Alarma.lua
Telegram-cli version 1.3.3, Copyright (C) 2013-2015 Vitaly Valtman
Telegram-cli comes with ABSOLUTELY NO WARRANTY; for details type `show_license'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show_license' for details.
Telegram-cli uses libtcl version 2.0.3
Telegram-cli includes software developed by the OpenSSL Project
for use in the OpenSSL Toolkit. (http://www.openssl.org/)
Telegram-cli uses libpython version 2.7.6
I: config dir=[/home/leonardo/.telegram-cli]
>
```

Ilustración 48 Inicio de Telegram.

Para realizar las pruebas será necesario crear una nueva conversación con el número asociado a la Raspberry.



Ilustración 49 Creación de conversación en Telegram.

Para la primera prueba solo tendremos que enviar cualquier mensaje de texto para comprobar la conexión.

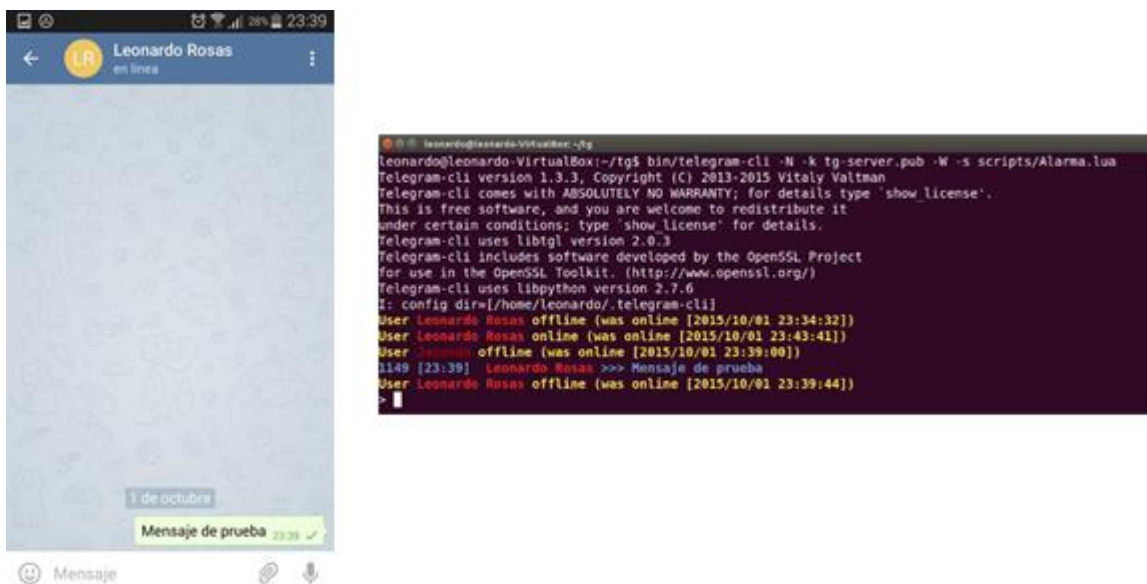


Ilustración 50 Comprobación de Conexión.

Ahora procederemos a mandar el primer mensaje con respuesta automática, enviando la palabra “Ping”.



```
User Leonardo Rosas offline (was online [2015/10/01 23:34:32])
User Leonardo Rosas online (was online [2015/10/01 23:43:41])
User Jaconda offline (was online [2015/10/01 23:39:00])
1149 [23:39] Leonardo Rosas >>> Mensaje de prueba
User Leonardo Rosas offline (was online [2015/10/01 23:39:44])
User Leonardo Rosas online (was online [2015/10/01 23:48:44])
1150 [23:43] Leonardo Rosas >>> Ping
1151 [23:43] Leonardo Rosas <<< Mensaje Automatico de Respuesta
```

Ilustración 51 Envío mensaje "Ping".

Ahora enviaremos la cadena de texto “Escom” para que nos envíe automáticamente una imagen predeterminada.



```
I: config dir=[/home/leonardo/.telegram-cli]
User Leonardo Rosas online (was online [2015/10/01 23:53:59])
1152 [23:49] Leonardo Rosas >>> Escom
1153 [23:49] Leonardo Rosas <<< Enviando imagen
1154 [23:49] Leonardo Rosas >>> [photo]
```

Ilustración 52 Envío Mensaje "Escom".

Ahora se enviar la cadena de texto “Audio” para que nos envíe automáticamente el audio predeterminado.



Ilustración 53 Envío Mensaje "Audio".

5.4 Prototipo 4

Acoplamiento del Sensor Pi en el sistema embebido Raspberry Pi 2.

5.4.1 Descripción

Este prototipo muestra el acoplamiento, programación y funcionamiento del sensor Pi en el sistema embebido, ajustando la sensibilidad y el tiempo en el cual sensorará, acoplado a unos puertos de entrada y salida de la tarjeta Raspberry, y con el archivo sensor.py, se programara el funcionamiento del sensor, configurando puertos, comportamiento e intervalo en el que sensorará.

5.4.2 Objetivo

Acoplar e instalar el sensor pi para detectar movimiento presente y así poder configurarlo con el archivo sensor.py para posteriormente realizar la automatización de un mensaje de notificación mediante Telegram.

5.4.3 Desarrollo

Ajustar el sensor

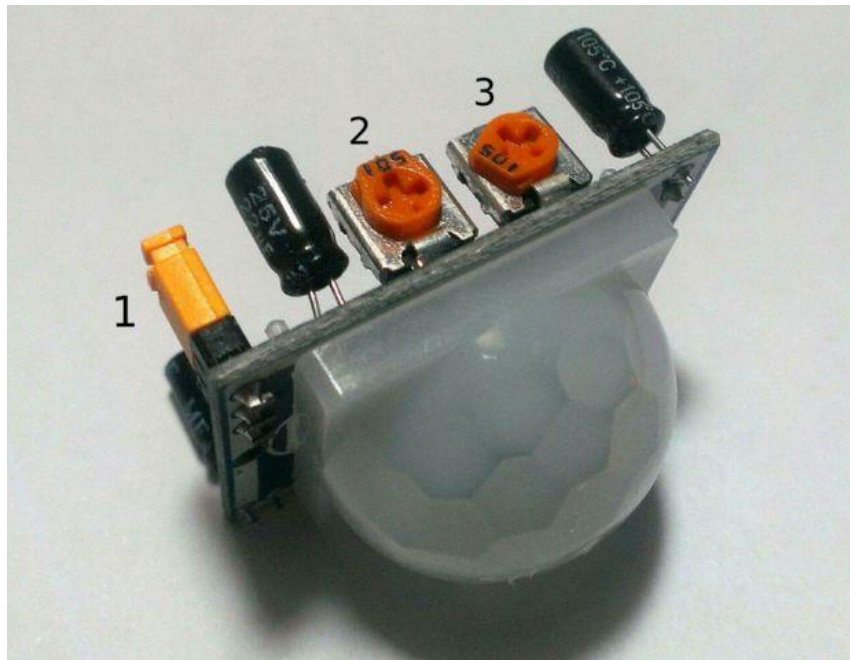


Ilustración 54 Diagrama Sensor HC-SR501.

El sensor incorpora 2 potenciómetros y un jumper que nos permiten modificar su comportamiento y adaptarlo a nuestras necesidades: precisión de detección, tiempo de reactivación, comportamiento ante detecciones repetitivas.

Los elementos de ajuste son los siguientes:

1. Selector de modo: nos permite cambiar entre el modo de funcionamiento continuo o el modo de repetición. En modo continuo, si el sensor detecta movimiento de manera continuada mantendrá una señal continua. En el modo de repetición, el sensor se activará al detectar movimiento y volverá luego a su estado normal, si vuelve a detectar movimiento se volverá a activar y completará otro ciclo, pero no funcionará de manera continua, aunque detecte movimiento repetidas veces. El primer modo es ideal para usarlo como detector de movimiento que permita iluminar una zona, ya que se activará mientras detecte presencia.
Si tomamos como primer contacto el que se encuentra en la esquina del sensor, puenteando los contactos 2 y 3 se activa el modo continuo, y por el contrario haciéndolo con los contactos 1 y 1 se activa el modo de repetición. Siempre deberá activarse un modo u otro.
2. Ajuste de sensibilidad: aumenta o disminuye la sensibilidad del sensor, con ello podemos ajustar la distancia a la que se activará y/o la cantidad de movimiento necesario para activar el sensor (por ejemplo, para distinguir una persona de una mascota). Teniendo el sensor como en la imagen, el mínimo se encontrará en el lado izquierdo y el máximo en el derecho. Los potenciómetros se pueden ajustar con la ayuda de un destornillador.
3. Ajuste del temporizador: aumenta o disminuye el tiempo que se activará el sensor una vez detecte presencia, el rango va aproximadamente desde unos 3 segundos hasta unos 5 minutos. Si el sensor se encuentra en modo continuo, el tiempo de activación será como mínimo el ajustado, no existiendo máximo si el sensor detecta continuamente presencia mientras se encuentra activado.

Diagrama cableado a protoboard.

El siguiente diagrama muestra la conexión entre el sensor y la tarjeta Raspberry, así también como los puertos a los que se conectara, los puertos 2 (Voltaje) y 6 (Tierra) que son de entrada y el puerto 11 de salida que serán necesarios para la comunicación entre ambos dispositivos.

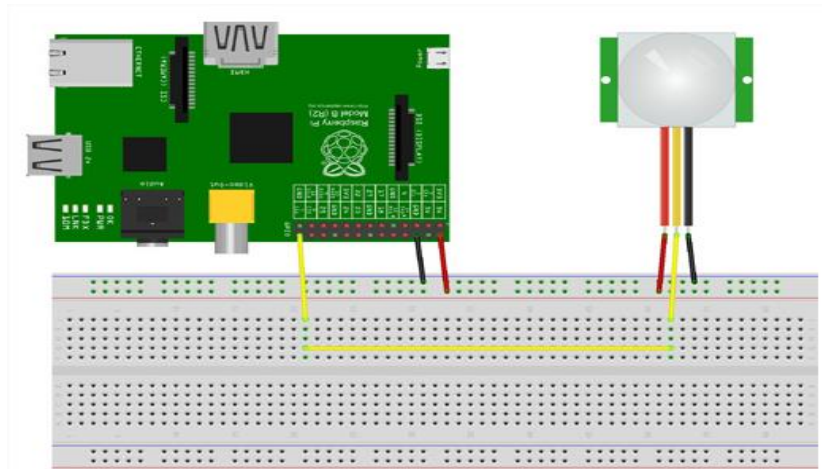


Ilustración 55 Diagrama Cableado a protoboard.

Script en Python.

```
import RPi.GPIO as GPIO      #Importamos la libreria GPIO
import time                  #Importamos time
from time import gmtime, strftime #importamos gmtime y strftime
GPIO.setmode(GPIO.BCM)      #Configuramos los pines GPIO como BCM
PIR_PIN = 7                  #Usaremos el pin GPIO n°7
GPIO.setup(PIR_PIN, GPIO.IN) #Lo configuramos como entrada

#GPIO.setup(17, GPIO.OUT)    #Configuramos el pin 17 como salida (para un

try:
    while True: #Iniciamos un bucle infinito
        if GPIO.input(PIR_PIN): #Si hay señal en el pin GPIO n°7
            # GPIO.output(17,True) #Encendemos el led
            time.sleep(1)         #Pausa de 1 segundo
            timex = strftime("%d-%m-%Y %H:%M:%S", gmtime()) #Creamos una cadena
            print timex + " MOVIMIENTO DETECTADO" #La sacamos por pantalla
            time.sleep(1) #Pausa de 1 segundo
            # GPIO.output(17,False) #Apagamos el led
            time.sleep(1)         #Pausa de 1 segundo y vuelta a empezar
except KeyboardInterrupt: #Si el usuario pulsa CONTROL + C...
    print "quit"          #Anunciamos que finalizamos el script
    GPIO.cleanup()         #Limpiamos los pines GPIO y salimos
```

Ilustración 56 Script funcionamiento sensor en Python

Pruebas.

Conversación de Telegram, mientras el sensor monitorea algún movimiento.

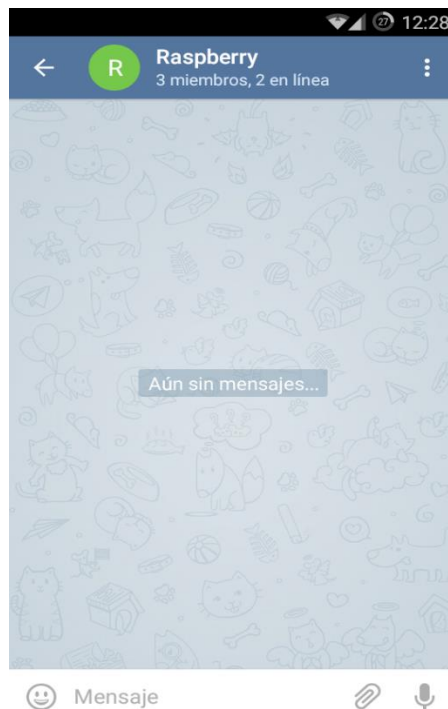


Ilustración 57 Conversación de Telegram

Movimiento detectado, se envía notificación a Telegram, se recibe la notificación detectada por el sensor.

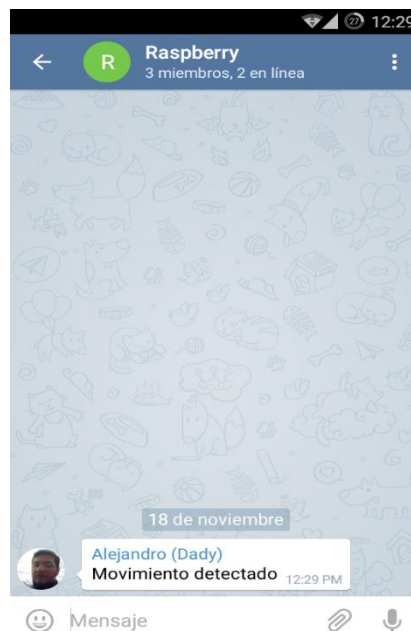


Ilustración 58 Notificación de movimiento detectado por el sensor.

5.5 Prototipo 5

Automatización del Sensor Pi con la cámara Pi Noir en el sistema embebido.

5.5.1 Descripción

Este Prototipo muestra la automatización del funcionamiento que tendrá el sensor Pi al sensor alguna presencia y el comportamiento de la cámara para tomar foto, configurando Telegram como un daemon con los archivos tg.sh y crontab, y con los archivos Alarma.lua, foto.lua y sensor.py, para el comportamiento del sensor Pi y cámara Pi Noir.

5.5.2 Objetivo

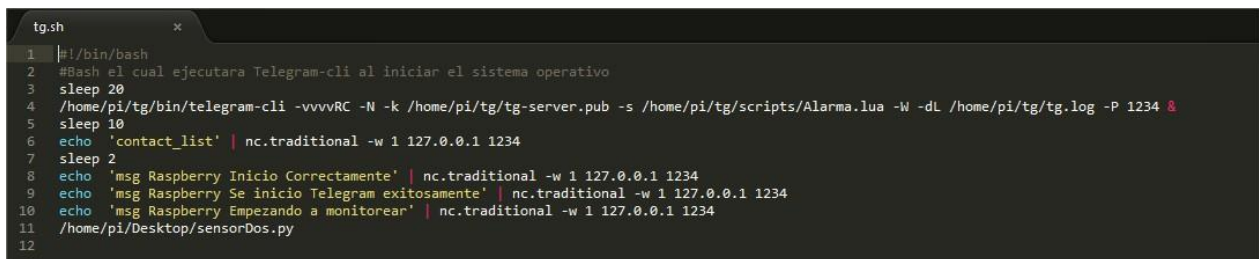
Automatizar el sensor Pi, para el funcionamiento principal del sistema embebido, tomando una fotografía cuando el sensor Pi, sense alguna presencia.

5.5.3 Desarrollo

Para realizar la automatización de las funcionalidades del sistema con Telegram, primero es necesario convertir la aplicación de mensajería instantánea de Telegram como un servicio o proceso propio del sistema operativo de la tarjeta Raspberry, de esta manera aseguramos que cuando se inicie o reinicie la tarjeta, Telegram se ejecutara de forma automática, sin necesidad de tener que ejecutarlo.

Código tg.sh

Para el proceso de automatización, creamos el archivo tg.sh, en el cual definimos la carpeta donde se localiza Telegram, dándole los permisos para que se inicie automáticamente al arrancar el sistema operativo, posteriormente escribimos las funcionalidades que le daremos, en este caso la notificación de que Telegram se ejecutó de forma correcta y que se empieza a monitorear.



```
1 #!/bin/bash
2 #Bash el cual ejecutara Telegram-cli al iniciar el sistema operativo
3 sleep 20
4 /home/pi/tg/bin/telegram-cli -vvvRC -N -k /home/pi/tg/tg-server.pub -s /home/pi/tg/scripts/Alarma.lua -W -dL /home/pi/tg/tg.log -P 1234 &
5 sleep 10
6 echo 'contact_list' | nc.traditional -w 1 127.0.0.1 1234
7 sleep 2
8 echo 'msg Raspberry Inicio Correctamente' | nc.traditional -w 1 127.0.0.1 1234
9 echo 'msg Raspberry Se inicio Telegram exitosamente' | nc.traditional -w 1 127.0.0.1 1234
10 echo 'msg Raspberry Empezando a monitorear' | nc.traditional -w 1 127.0.0.1 1234
11 /home/pi/Desktop/sensorDos.py
12
```

Ilustración 59 Código tg.sh

Posteriormente procedemos a modificar el crontab del sistema operativo, para que así se ejecute el código del archivo tg.sh y así poder hacer que Telegram se ejecute de forma automática.

Para esto en una terminal ejecutamos la línea de comando: crontab -e.

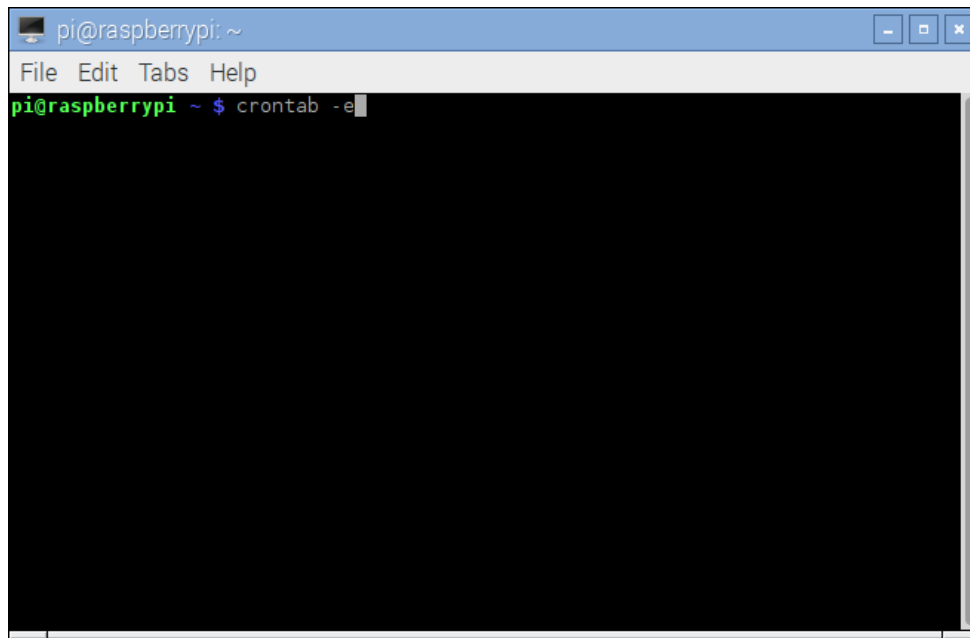


Ilustración 60 Ejecución de comando crontab -e

Posteriormente se abrirá el archivo crontab, una vez ahí procedemos a agregar la línea de código: @reboot /home/pi/Desktop/tg.sh, de esta manera le indicamos que cada que se inicie el sistema operativo, proceda a ejecutar el archivo tg.sh localizado en la carpeta de Desktop del sistema.

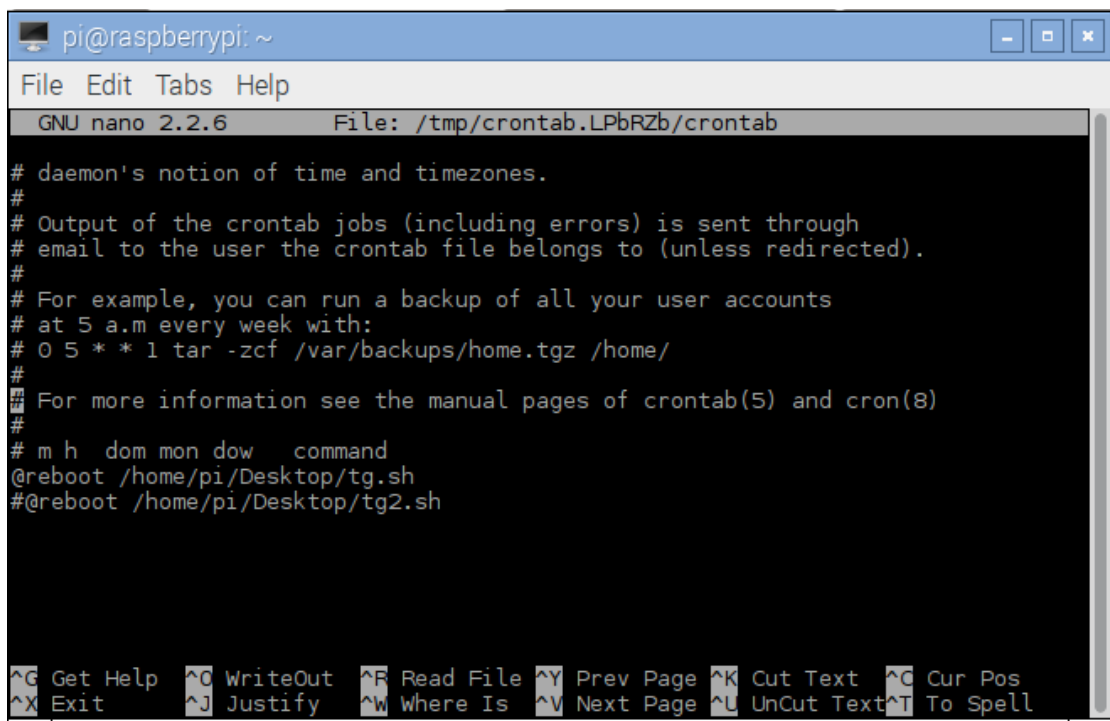


Ilustración 61 Archivo crontab

Una vez terminado con el proceso anterior, guardamos los cambios y de esta manera ya convertimos a Telegram como proceso del sistema operativo. De esta forma ya solo procedemos a crear los scripts que contendrán la funcionalidad del sistema.

Código Alarma.lua

Se implementará el código llamado Alarma.lua, que contendrá el comportamiento del sistema híbrido con el Telegram, al recibir ciertos valores mensajes, los cuales harán que Telegram conteste de forma automática de acuerdo al mensaje recibido.

```

USER_ALLOWED = "Leonardo_Rosas"
USER_ALLOWED2 = "Rene"

function on_msg_receive (msg)
    if msg.out then
        return
    end

    function save_file(extra, success, file)
        if success then
            print(file)          ---filename where the media is stored
            print(extra[0])      ---sender name
            print(extra[1])      ---msg.id
            print(extra[2])      ---msg.date
            print('omxplayer' ..file)
            os.execute('omxplayer' ..file)
            ---the extra[] values where these, which where written to the a = {}
            ---array. Later you can include code here to handle the specific
            ---media files.
        end
    end

    -- Allowed only our ID's
    if (msg.from.print_name == USER_ALLOWED or msg.from.print_name == USER_ALLOWED2) then
        a = {}
        a[0] = msg.from.print_name
        a[1] = msg.id
        a[2] = msg.date
        if msg.media then      ---check if photo
            if msg.media.type == 'document' then

                print('Tipo de Respuesta: ' ..msg.media.type)
                load_audio(msg.id, save_file, a)
            end
        end

        if (msg.text=='ping' or msg.text=='Ping') then
            send_msg ('Raspberry', 'Mensaje Automatico de Respuesta', ok_cb, false)
        end

        if (msg.text=='reiniciar' or msg.text=='Reiniciar') then
            os.execute('sudo reboot now' ..file)
            print('leonidas92')
        end
    end
end

function on_our_id (id)
end

function on_secret_chat_created (peer)
end

function on_user_update (user)
end

function on_chat_update (user)
end

```

```

function on_get_difference_end ()
end

function on_binlog_replay_end ()
end

function ok_cb(extra, success, result)
end

```

Ilustración 62 Código Alarma.lua.

Código foto.lua

El código del archivo foto.lua, describe la configuración de la cámara fotográfica Pi Noir, como son la ruta en la cual se almacenara, la resolución de la foto tomada, tamaño y el nombre que tendrá.

```

FOLDER = "/home/pi/tg/media/"
TOSEND_FOLDER = "/home/pi/tg/toSend/"
--Quality 0 - 100
QUALITY = 25

--name_folder = os.date("%H_%M_%S_%m_%d_%Y")

name_picture= os.date("%H_%M_%S_%m_%d_%Y")

--os.execute('mkdir '..FOLDER..'/'..name_folder)

os.execute('raspistill -o ' .. FOLDER ..name_picture..'jpg -t 1 -vf -h 720 -w 1080 -rot 270 -q '... QUALITY )
--os.execute('raspistill -o ' .. FOLDER .. name_folder .. '/' ..name_picture..'jpg -vf -h 720 -w 1080 -q 25')
--raspistill -o imagen2.jpg -h 720 -w 1080
os.execute('cp ' .. FOLDER .. '/' ..name_picture..'jpg ' .. TOSEND_FOLDER ..'foto.jpg')

```

Ilustración 63 Código foto.lua.

Código Sensor.py

El código del archivo Sensor.py, describe el comportamiento que tendrá el sensor Pi, al sensar alguna presencia y tomar una fotografía con la cámara Pi Noir.

```

import RPi.GPIO as GPIO
import time

# Use BCM GPIO references
# instead of physical pin numbers
GPIO.setmode(GPIO.BOARD)

# Define GPIO to use on Pi
GPIO_PIR = 11

print "PIR Module Test (CTRL-C to exit)"

# Set pin as input
GPIO.setup(GPIO_PIR,GPIO.IN)      # Echo

Current_State = 0
Previous_State = 0

try:

    print "Waiting for PIR to settle ..."

    # Loop until PIR output is 0
    while GPIO.input(GPIO_PIR)==1:
        Current_State = 0

    print " Ready"

    # Loop until users quits with CTRL-C
    while True :

        # Read PIR state
        Current_State = GPIO.input(GPIO_PIR)

        if Current_State==1 and Previous_State==0:
            # PIR is triggered
            print " Motion detected!"
            os.execute('raspistill -o foto.jpg -h 480 -w 640 -t 1 -rot 270 -q 50')
            echo('send_photo Raspberry /home/pi/tg/toSend/foto.jpg | nc.traditional 1 127.0.0.1 1234')
            # Record previous state
            Previous_State=1
        elif Current_State==0 and Previous_State==1:
            # PIR has returned to ready state
            print " Ready"
            Previous_State=0

        # Wait for 1 milliseconds
        time.sleep(0.1)

except KeyboardInterrupt:
    print " Quit"
    # Reset GPIO settings

```

Ilustración 64 Código Sensor.py.

Funcionamiento del sistema

El sistema se está ejecutando, esperando a recibir algún comando establecido (ping, foto, reboot) o detectar algún movimiento.

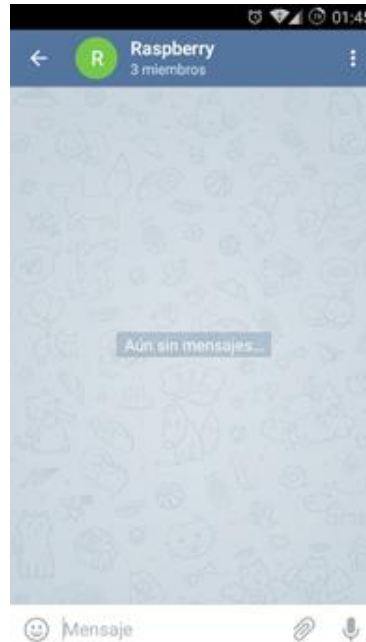


Ilustración 65 Conversación de Telegram.

Se escribe el comando reboot para reiniciar el sistema, y se inicie automáticamente el sistema junto con Telegram para comenzar a monitorear algún movimiento que se presente en la casa habitación.

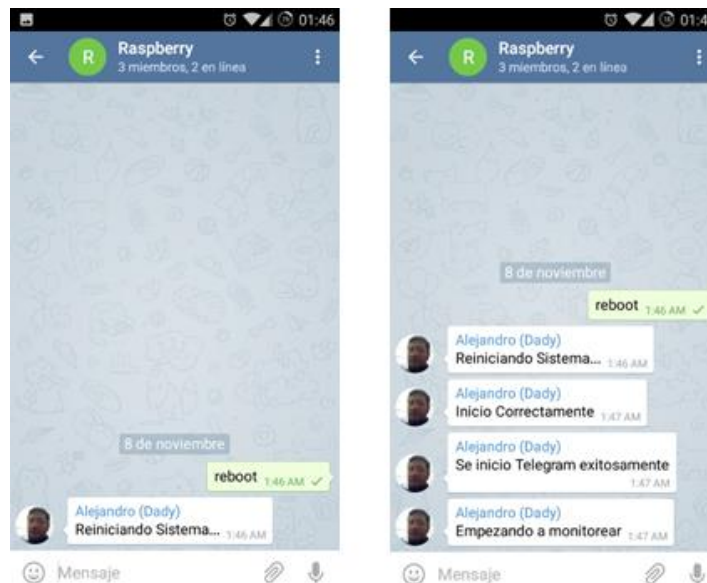


Ilustración 66 Comando reboot y notificación de reinicio de sistema.

Se escribe el comando foto, para tomar una foto instantánea desde la tarjeta y recibirla por Telegram.



Ilustración 67 Comando foto, envío y recepción de foto por Telegram.

Se reciben notificaciones de que se ha detectado movimiento, así también se toman fotos del movimiento detectado y se envían a Telegram.



Ilustración 68 Notificación de movimiento detectado y toma de fotos.

5.6 Prototipo 6

Programación del micrófono dongle para grabe los mensajes de voz desde el sistema embebido.

5.6.1 Descripción

Este prototipo muestra la programación del micrófono dongle que tendrá la función de grabar mensajes de voz, una vez que el botón del sistema embebido sea oprimido. La configuración se dará mediante el lenguaje de programación Python con el archivo botón.py, el cual incluirá el comportamiento que tendrá el botón al ser oprimido, menos de 3 segundos tipo timbre y mayor a 3 segundos como grabadora de voz, enviando la grabación o notificación de timbre mediante Telegram.

5.6.2 Objetivo

Programar el botón que funcionara como un timbre o grabador de mensajes de voz, para que mande notificaciones mediante el uso de la aplicación de mensajería instantánea Telegram

5.6.3 Desarrollo

Se implementará un código en Python que controle el funcionamiento del micrófono, permitiendo crear un uso de timbre para dar aviso de timbre a través de un botón y como micrófono como medio de comunicación con el usuario si se sigue apretando el botón más de 3 segundos.

Código botón.py

Se implementará el código llamado botón.py, que contendrá el comportamiento del micrófono funcionando como timbre y micrófono según sea el tiempo en que se aprieta el botón, teniendo comunicación con Telegram para enviar mensaje como alerta o envió de audio de la grabación que se realice mediante el micrófono de la tarjeta Raspberry.

```
boton.py
1 #!/usr/bin/python
2 import RPi.GPIO as GPIO
3 import time
4 import os
5
6 GPIO.setmode(GPIO.BOARD)
7
8 GPIO.setup(38, GPIO.IN, pull_up_down=GPIO.PUD_UP)
9 pressed_time=0
10
11 while True:
12     if GPIO.input(38):
13         print 'Boton libre'
14         time.sleep(1)
15     else:
16         print 'Boton Presionado'
17         pressed_time=pressed_time+1
18         while GPIO.input(38):
19             pass
20             if pressed_time<=2:
21                 if pressed_time==1:
22                     os.system('echo \'msg Raspberry Hay alguien tocando el timbre \' | nc.traditional -w 1 127.0.0.1 1234')
23                     #print 'Menor a 3 segundos'
24                     #time.sleep(1)
25             elif pressed_time==2:
26                 print pressed_time
27                 print 'Grabando audio'
28                 os.system('arecord -D plughw:1,0 -V stereo -c 1 --duration=7 audio.wav')
29                 print 'Enviando audio...'
30                 os.system('echo \'send_audio Raspberry /home/pi/Desktop/audio.wav \' | nc.traditional -w 1 127.0.0.1 1234')
31                 #os.system('pkill -9 arecord')
32                 if GPIO.input(38)==0:
33                     print 'Grabando audio'
34                     os.system('arecord -D plughw:1,0 -V stereo -c 1 --duration=5 audio2.wav')
35                     print 'Enviando audio...'
36                     os.system('echo \'send_audio Raspberry /home/pi/Desktop/audio2.wav \' | nc.traditional -w 1 127.0.0.1 1234')
37                     #os.system('pkill -9 arecord')
38                     pressed_time=0
39             else:
40                 #print 'Mayor a 3 segundos'
41                 #time.sleep(1)
42                 pressed_time=0
```

Ilustración 69 Código boton.py

Funcionamiento del sistema

Sistema de micrófono en funcionamiento, esperando a que el botón sea apretado para empezar con alguna de sus 2 funciones principales (timbre o grabación de audio).

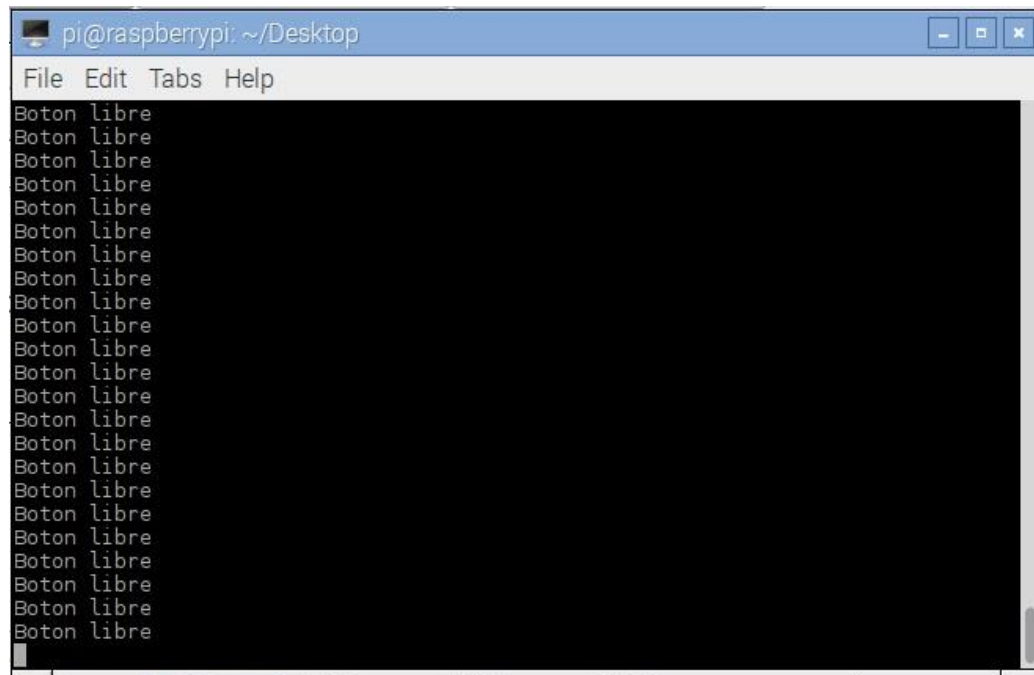


Ilustración 70 Botón de timbre esperando ser presionado para empezar con su función.

Botón de la tarjeta Raspberry ha sido presionado y se envía notificación a Telegram

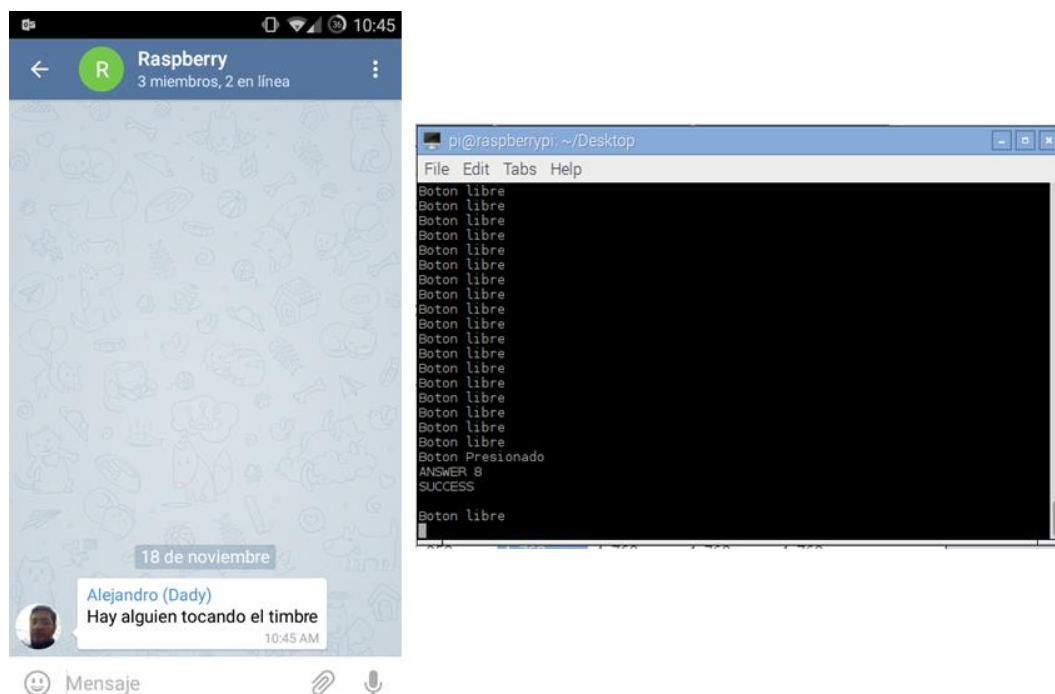
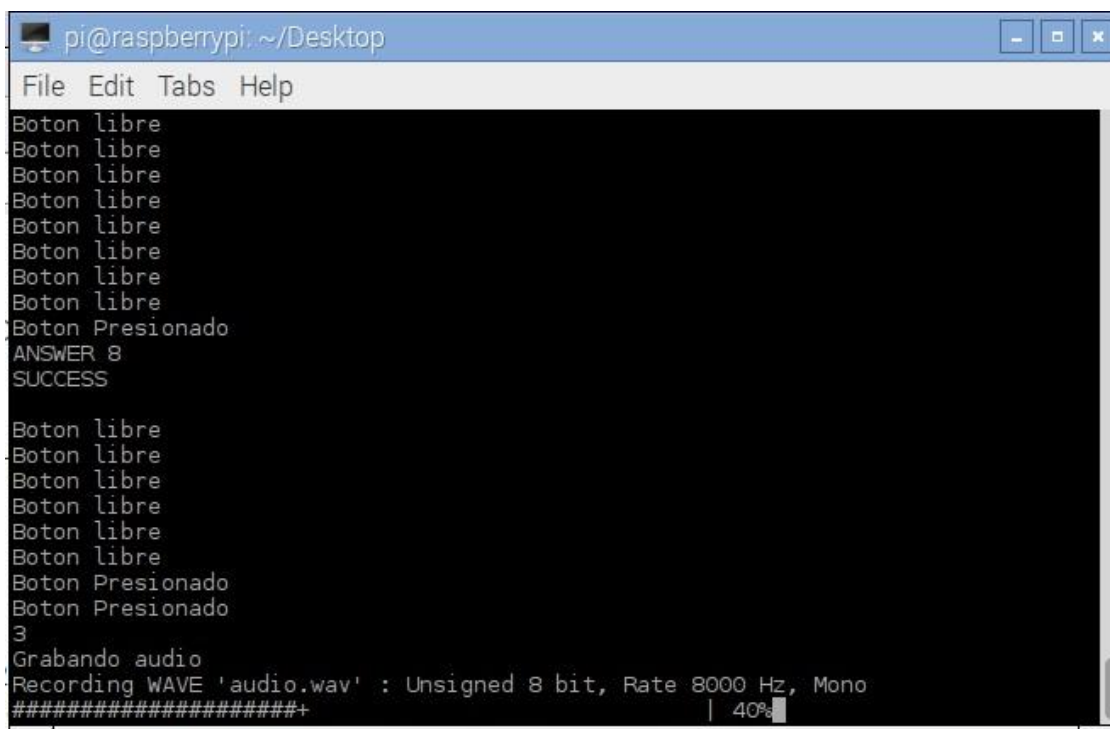


Ilustración 71 Botón presionado y envió de notificación a Telegram

El botón de timbre de la tarjeta Raspberry ha sido presionado más de 3 segundo y se empieza a grabar desde el micrófono.



```
pi@raspberrypi: ~/Desktop
File Edit Tabs Help
Boton libre
Boton libre
Boton libre
Boton libre
Boton libre
Boton libre
Boton libre
Boton libre
Boton Presionado
ANSWER 8
SUCCESS

Boton libre
Boton libre
Boton libre
Boton libre
Boton libre
Boton libre
Boton Presionado
Boton Presionado
3
Grabando audio
Recording WAVE 'audio.wav' : Unsigned 8 bit, Rate 8000 Hz, Mono
#####+ | 40%
```

Ilustración 72 Grabación de audio desde micrófono

Termina grabación desde el micrófono de la tarjeta Raspberry, se guarda en la tarjeta y posteriormente es enviado a Telegram para que lo reciba el usuario.

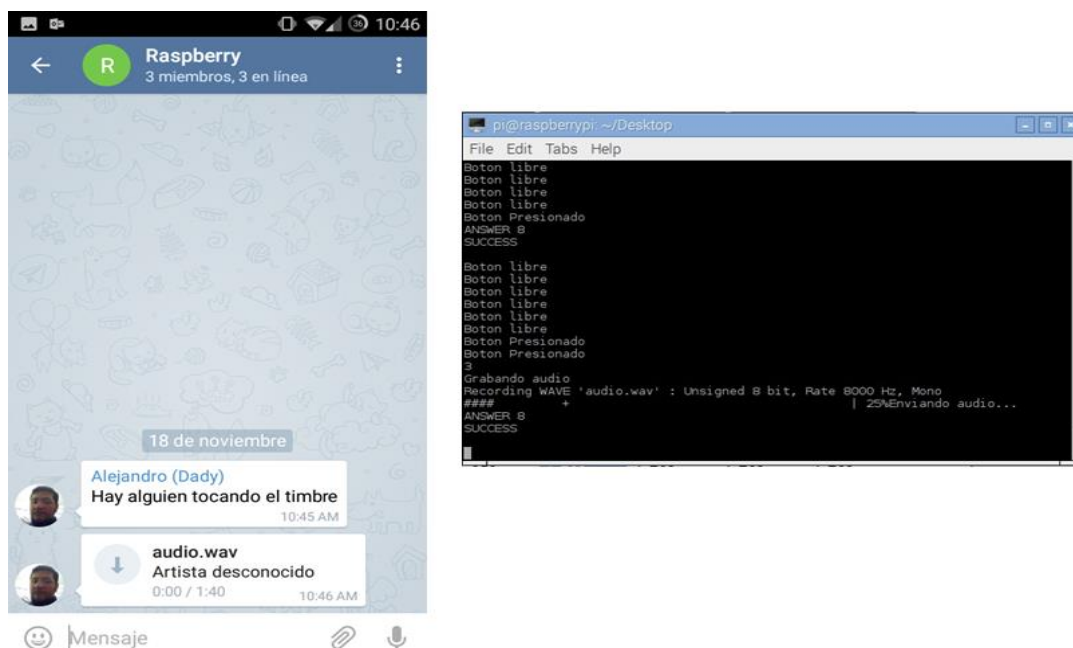
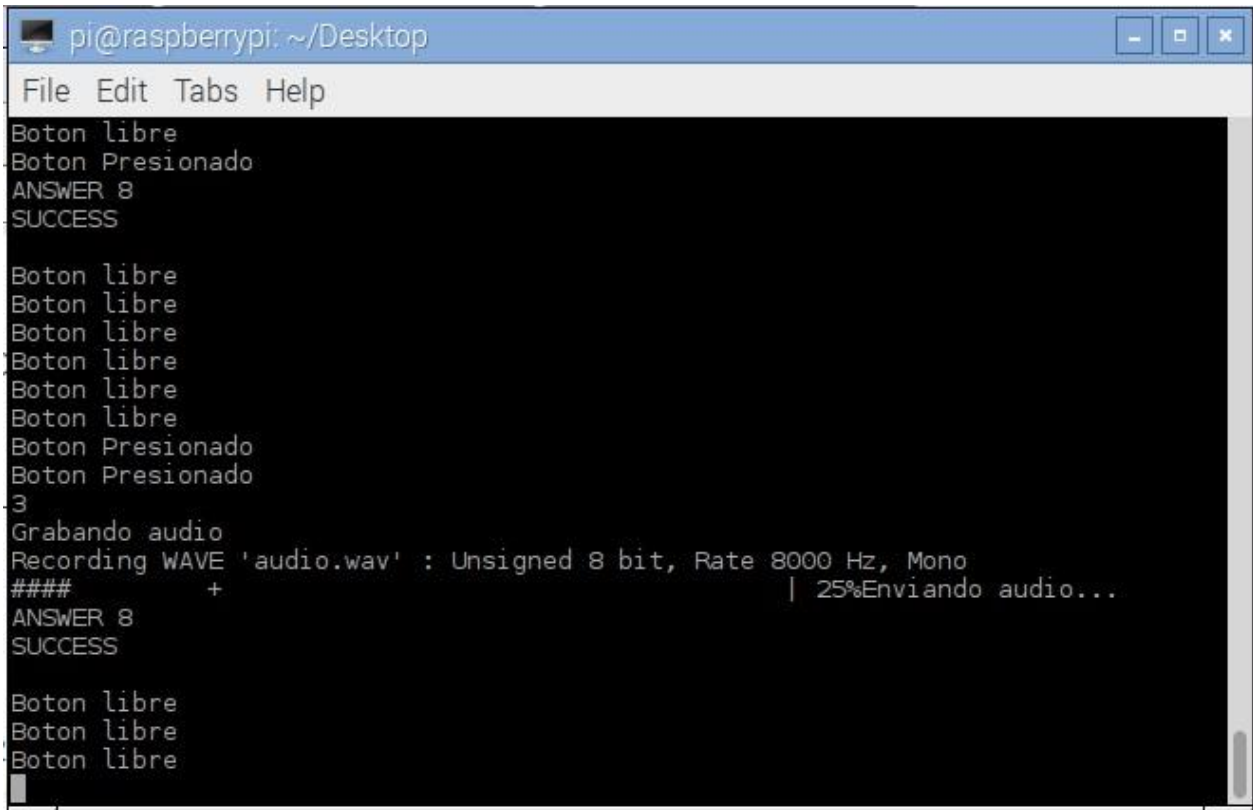


Ilustración 73 Envío de audio de grabación y recibo del audio en Telegram.

Terminado con el proceso de grabación y envío del audio grabado, el funcionamiento del botón regresa a su estado original, esperando hacer presionado para enviar notificación o grabar, repitiendo así su función para lograr un funcionamiento de interfono.



```
pi@raspberrypi: ~/Desktop
File Edit Tabs Help
Boton libre
Boton Presionado
ANSWER 8
SUCCESS

Boton libre
Boton libre
Boton libre
Boton libre
Boton libre
Boton libre
Boton libre
Boton Presionado
Boton Presionado
3
Grabando audio
Recording WAVE 'audio.wav' : Unsigned 8 bit, Rate 8000 Hz, Mono
####      +      | 25%Enviando audio...
ANSWER 8
SUCCESS

Boton libre
Boton libre
Boton libre
```

Ilustración 74 Regreso al estado original del botón para repetir su función

CONCLUSIONES

Un sistema de monitoreo a casa habitación, para su buen funcionamiento necesita conexión permanente a internet para garantizar una buena comunicación del sistema hacia el usuario.

La tarjeta Raspberry necesita un suministro de energía mayor a 750 mA y 5 V. Durante la conexión de la tarjeta a la corriente, evitar el uso de extensiones USB, ya que no todas suministran la corriente y el voltaje necesario para el encendido de la tarjeta. Si se necesita de una extensión, debe ser solo una y no varias unidas.

La tarjeta SD no debe ser removida de la tarjeta Raspberry, aun cuando la tarjeta está apagada pero conectada a la corriente. Si se remueve puede ocasionar un fallo en el sistema operativo de la tarjeta, teniendo que reinstalar todo de nuevo.

Al trabajar en la toma de fotos como medio de monitoreo, se debe verificar la calidad y tamaño de la imagen, tomar fotos con excelente calidad de imagen conlleva a un mayor espacio de almacenamiento y por ende un consumo mayor de datos y tiempo para su envío.

Algunas funcionalidades, como escuchar mensajes de voz, ver videos y visualizar archivos, se tuvieron que modificar debido a un problema con el código de Telegram en GitHub (structures.c y queries.c) para que cumpla con las funcionalidades normales de la aplicación.

En general con la tarjeta Raspberry Pi es posible no solamente el poder monitorear una casa habitación con un sensor y una cámara fotográfica, sino que nos permite adaptar otros dispositivos que ayuden a la escalabilidad del sistema, implementando novedosas formas de seguridad haciendo un posible uso de una aplicación de mensajería instantánea. De esta manera se podrían crear diversos sistemas tanto de monitoreo como de seguridad implementados a un fin específico.

TRABAJO A FUTURO

Se propone como trabajo a futuro de este sistema lo siguiente:

- Implementar nuevos métodos de grabación al dejar un mensaje de voz, para hacer más interactivo el sistema con el usuario.
- Implementación de un método de video al sistema como medio de monitoreo a casa habitación.
- Considerar el uso de más de 2 cámaras Pi Noir, para garantizar una foto segura de la persona que toca la puerta.
- Implementación de un sistema de alarma para puerta y ventanas.
- Implementación de un sistema para poder controlar el cierre de puertas y ventanas, así también como controlar el apagado y encendido de las luces de la casa habitación.
- Reconocimiento facial.
- Traducción de texto a audio y viceversa.
- Comunicación con fines de prevención.

REFERENCIAS

- [1] Domotica, Consejería de Economía e Innovación Tecnológica de Madrid, «Madrid.org,» 2007. [En línea]. Available: <http://www.madrid.org/bvirtual/BVCM005729.pdf>.
- [2] Sistema Embebido, Facultad de Ciencias Exactas, Físicas y Naturales; Universidad Nacional de Córdoba, «Ingeniería en Computación,» 2015. [En línea]. Available: <http://computacion.efn.uncor.edu/descripci%C3%B3n>.
- [3] Robo a Casa Habitación Vargas, Hugo, «Scribd,» 06 Julio 2012. [En línea]. Available: <http://es.scribd.com/doc/99279861/Robo-a-Casa-Habitacion#scribd>.
- [4] Asociación Civil, «México Unido Contra la Delincuencia A.C.,» 2012. [En línea]. Available: <http://mucd.org.mx/recursos/Contenidos/Estudiosycifras/documentos2/Datos%20sobre%20delitos%20en%20Mexico.pdf>.
- [5] Smartbell, «Kickstarter,» 04 Enero 2014. [En línea]. Available: <https://www.kickstarter.com/projects/1256599792/smartbell-wi-fi-doorbell-for-video-chats-to-ios-an>. [Último acceso: Noviembre 2014].
- [6] Nucano, «Kickstarter,» 28 Junio 2014. [En línea]. Available: https://www.kickstarter.com/projects/632247255/nucano-smart-door-chime-reimagining-home-intelligence?ref=nav_search. [Último acceso: Noviembre 2014].
- [7] Ring, Siminoff, Jamie, «Ring,» Agosto 2014. [En línea]. Available: <https://ring.com>.
- [8] C. Hallinan, Embedded Linux, Prentice Hall, 2006.
- [9] Raspberry Inc., «Raspberry,» 2006. [En línea]. Available: <https://www.raspberrypi.org/products/raspberry-pi-2-model-b/>. [Último acceso: Enero 2015].
- [10] A. Silbershatz, Sistemas Operativos, Addison Wesley, 2008.
- [11] G. Coulouris, Sistemas Distribuidos, Tercera ed., Addison-Wesley Iberoamericano, 2001.
- [12] G. L. R., Administración de Servidores Linux (Ubuntu/Fedora).
- [13] Álvarez Miguel Ángel; Python, «Desarrolloweb.com,» 1999. [En línea]. Available: <http://www.desarrolloweb.com/articulos/1325.php>.

- [14] Norberto Barraza Calvillo; Shell, «Prezi,» 14 Octubre 2014. [En línea]. Available: <https://prezi.com/w7j6ztqciawp/shell-y-sistema-de-ventanas-unix/>.
- [15] Raspbian.org, «Raspbian,» 2006. [En línea]. Available: <https://www.raspbian.org/>. [Último acceso: Marzo 2015].
- [16] Telegram, «Telegram,» 14 Agosto 2013. [En línea]. Available: <https://telegram.org/>. [Último acceso: Febrero 2015].
- [17] Wikipedia, Diffie Hellman, «Wikipedia,» 20 Mayo 2015. [En línea]. Available: <https://es.wikipedia.org/wiki/Diffie-Hellman>.
- [18] MTProto, «Telegram,» 2013. [En línea]. Available: <https://core.telegram.org/mtproto>.
- [19] H. James; Codecs for Raspberry+Pi, «Raspberry.org,» 20 Enero 2012. [En línea]. Available: <https://www.raspberrypi.org/blog/libraries-codecs-oss/>.
- [20] Ierusalimschy Roberto; LUA, «LUA.org,» 28 Septiembre 2011. [En línea]. Available: <http://www.lua.org/manual/5.1/es/manual.html>.
- [21] Wikipedia, Sensor, «Wikipedia,» Wikipedia, 13 Octubre 2015. [En línea]. Available: <https://es.wikipedia.org/wiki/Sensor>.
- [22] Universidad de Guadalajara, «<http://proton.ucting.udg.mx/>,» Centro Universitario de Ciencias Exactas e Ingenierías , [En línea]. Available: <http://proton.ucting.udg.mx/~mariocc/piro.html>.
- [23] Sensor de Movimiento, «fpaez,» 9 Agosto 2014. [En línea]. Available: <http://fpaez.com/sensor-de-movimiento-infrarojo-hc-sr501/>.
- [24] Daemons (Linux), «archLinux,» 2 Septiembre 2015. [En línea]. Available: [https://wiki.archlinux.org/index.php/Daemons_\(Español\)](https://wiki.archlinux.org/index.php/Daemons_(Español)).
- [25] Demonios, iniciando y parando los demonios, «Adslayuda,» 2015. [En línea]. Available: <http://www.adslayuda.com/linux-demonios.html>.
- [26] Aldrin Enforcer, Definición de Microfóno, «Scribd,» 17 Noviembre 2011. [En línea]. Available: <http://es.scribd.com/doc/73014596/Definicion-de-microfono#scribd>.
- [27] R. S. Pressman, Ingeniería del Software: Un enfoque práctico, R. S. Pressman, Ed., Mc Graw Hill, 2002.

GLOSARIO

- SiMC: Sistema de Monitoreo para Casa Habitación.
- Casa Habitación: Edificación construida para ser habitada por una persona o un grupo de personas.
- Sistema Embebido: Sistema de computación diseñado para realizar una o varias funciones específicas en tiempo real.
- Android: Sistema operativo orientado a dispositivos móviles, basado en una versión modificada del núcleo de Linux.
- iOS: Sistema operativo móvil desarrollado por Apple Inc para su producto iPhone.
- Telegram: Servicio de mensajería por Internet desarrollado por los hermanos Nikolai y Pavel Durov.
- Raspberry: Ordenador de placa reducida o de bajo coste.
- Driver / controlador: [programa informático](#) que permite al [sistema operativo](#) interactuar con un [periférico](#), haciendo una abstracción del [hardware](#) y proporcionando una [interfaz](#) para utilizar el dispositivo.