

Grado en Ingeniería Electrónica Industrial y Automática

Curso 2018-2019

Trabajo Fin de Grado

Sistema de control y asistencia a personas dependientes

Autor

Luis Cambero Piqueras

Tutor

David Griol Barres

Leganés, octubre de 2019



Esta obra se encuentra sujeta a la licencia *Creative Commons* **Reconocimiento – No Comercial – Sin Obra Derivada**

*“Todo hombre puede ser, si se lo propone,
escultor de su propio cerebro.”*

Santiago Ramón y Cajal

RESUMEN

En el presente Trabajo Fin de Grado se ha desarrollado un sistema para el control y la asistencia a personas dependientes, cuyo objetivo principal es poder prestar asistencia no urgente a distancia de forma automática, basándose en los datos recogidos del entorno de la persona. De este modo, se pretende que las personas en situación de dependencia, que son las que más asistencia suelen precisar, dispongan de una herramienta extra que sea capaz de solicitar asistencia, aunque la persona no pueda.

Las personas dependientes suelen necesitar de otra persona o sistema que las controlen y las ayuden en las tareas diarias, por ello, la automatización de estos controles y las solicitudes de asistencia permite reducir costes a la vez que se hace de forma más precisa e inteligente. La automatización actual en este campo no está muy avanzada. Actualmente, los servicios de teleasistencia dependen de la pulsación de un botón por parte del usuario para percatarse de la necesidad asistencial. En los casos en los que, por el motivo que sea, el usuario no puede utilizar dicho botón, la asistencia puede demorarse de forma excesiva. En este trabajo, se ha utilizado una interfaz conversacional como sustituto de este botón, de forma que el usuario solicita asistencia solo con el uso de su voz. Intentando cubrir los casos en los que la persona no pudiera hacer uso de su voz, el sistema incorpora una serie de sensores para la obtención de datos de los cuales se pueda extraer un patrón anómalo correspondiente a una situación que requiera asistencia.

Para la realización del trabajo, se ha realizado un estudio previo sobre el concepto de persona en situación de dependencia y sobre las necesidades que estas personas tienen. Además, se incluye una revisión de las tecnologías usadas, así como de su evolución y la justificación de su elección, mostrando también otras similares o alternativas. Se hace hincapié en las interfaces conversacionales, ya que son parte diferenciadora del sistema desarrollado al permitir al usuario una interacción más natural y libre de dependencias de cualquier otro dispositivo externo. Otras tecnologías, como los algoritmos de aprendizaje automático, los cuales permiten el reconocimiento de patrones, también juegan un papel muy importante, ya que permite detectar situaciones de posible peligro o alerta sin la supervisión de un técnico. Para la detección de estos patrones, se recogen datos de distintas variables del entorno mediante sensores, que están permanentemente transmitiendo sus datos a través de Internet.

El documento incorpora una descripción general, que proporciona una visión sobre las distintas partes del sistema, y una descripción detallada, en la cual se explica el trabajo realizado de forma técnica. Además, se ha incorporado la gestión del proyecto, el marco regulador y el impacto sobre el entorno socio-económico. Incluye también la evaluación realizada que, junto con los objetivos iniciales, se han usado para extraer las conclusiones una vez finalizado el presente Trabajo Fin de Grado.

Palabras clave: dependencia, teleasistencia, interfaz conversacional, altavoz inteligente, aprendizaje automático, IoT.

INDICE DE CONTENIDOS

RESUMEN	i
1. INTRODUCCIÓN	1
1.1. Motivación	1
1.2. Objetivos	3
1.3. Fases del desarrollo.....	5
1.3.1. Planificación	5
1.3.2. Ejecución	5
1.3.3. Evaluación	5
1.3.4. Documentación	6
1.4. Medios empleados	6
1.4.1. Recursos hardware.....	6
1.4.2. Recursos software	6
1.5. Estructura de la memoria.....	7
2. ESTADO DEL ARTE	10
2.1. Interfaces conversacionales	10
2.1.1. ¿Qué es una interfaz conversacional?	11
2.1.2. Interfaces conversacionales disponibles actualmente	13
2.1.2.1. Alexa.....	13
2.1.2.2. Cortana	14
2.1.2.3. Siri.....	15
2.1.2.4. Assistant	16
2.1.3. Dispositivos que integran una interfaz conversacional.....	16
2.2. Altavoces inteligentes	18
2.2.1. Google Home.....	18
2.2.2. Amazon Echo	19
2.2.3. Apple HomePod	20
2.3. Aprendizaje automático	20
2.3.1. Tipos de algoritmos	21
2.3.2. Técnicas de clasificación.....	22
2.3.3. Isolation Forest.....	23
2.3.4. Librerías de algoritmos de aprendizaje automático.....	25
2.3.4.1. TensorFlow	25
2.3.4.2. Keras.....	25
2.3.4.3. Scikit-learn.....	26

2.4. Dialogflow	26
2.5. Firebase	28
2.6. Raspberry Pi	29
2.7. Sistemas de teleasistencia.....	30
2.7.1. Componentes del sistema	30
2.7.2. Modalidades según el tipo de accionamiento	31
2.7.3. Modalidades según el tipo de respuesta	32
2.7.4. Modalidades según el tipo de demanda de asistencia	32
2.8. La persona dependiente.....	32
2.8.1. Determinantes demográficos de la dependencia	34
3. DESCRIPCIÓN GENERAL DEL SISTEMA.....	37
3.1. Presentación del sistema	37
3.2. Elementos del sistema	39
3.2.1. Interfaz conversacional	39
3.2.2. Control y análisis de los datos	41
3.2.2.1. Obtención de los datos	42
3.2.2.2. Algoritmo de aprendizaje automático	43
3.2.2.3. Alertas y avisos.....	44
3.2.3. Base de datos	44
3.2.4. Página de configuración	45
4. DESCRIPCIÓN DETALLADA DEL SISTEMA.....	48
4.1. Desarrollo de la interfaz conversacional	48
4.1.1. Creación de un agente	49
4.1.2. Creación de intenciones	49
4.1.3. Creación de Entidades.....	51
4.1.4. Uso de los contextos	53
4.1.5. Integración con Google Assistant.....	54
4.1.6. Conexión con Firebase	55
4.2. Recogida de datos del entorno	57
4.2.1. Sensores	57
4.2.1.1. Sensor de temperatura y humedad	58
4.2.1.2. Sensor de presencia	59
4.2.1.3. Sensor de calor.....	59
4.2.2. Software para la lectura de los sensores	60
4.3. Análisis de los datos mediante algoritmo de aprendizaje automático	61
4.3.1. Desarrollo en un <i>Notebook</i>	62

4.3.2. Implementación en el sistema	64
4.4. Gestión de avisos y alertas	65
4.4.1. Servidor	65
4.4.2. Conexiones y configuración	67
4.5. Base de datos	68
4.5.1. Creación de la base de datos.....	68
4.5.2. Estructura de la base de datos.....	68
4.6. Página de configuración	71
4.6.1. Estructura y diseño.....	71
4.6.2. Funcionalidades.....	72
5. EVALUACIÓN Y GESTIÓN DEL PROYECTO	75
5.1. Metodología de evaluación.....	75
5.2. Resultados de la evaluación	77
5.3. Planificación temporal.....	78
6. MARCO REGULADOR Y ENTORNO SOCIO-ECONÓMICO	82
6.1. Marco regulador.....	82
6.2. Entorno socio-económico	83
6.2.1. Presupuesto.....	85
7. CONCLUSIONES Y TRABAJO FUTURO	90
7.1. Conclusiones.....	90
7.2. Trabajo futuro	93
GLOSARIO	96
BIBLIOGRAFÍA	98

INDICE DE FIGURAS

Figura 1.1: Personas que reciben una prestación por dependencia en España [2]	2
Figura 1.2: Ejemplo de un botón para solicitud de asistencia en un sistema de teleasistencia ...	3
Figura 2.1: Google Assistant ejecutándose en un teléfono Android.....	11
Figura 2.2: Logotipo de Alexa	14
Figura 2.3: Logotipo de Cortana	15
Figura 2.4: Logotipo de Siri.....	15
Figura 2.5: Logotipo de Assistant	16
Figura 2.6: Google Assistant ejecutándose en un reloj inteligente	17
Figura 2.7: Google Home.....	19
Figura 2.8: Amazon Echo	19
Figura 2.9: Apple HomePod	20
Figura 2.10: Comparativa del número de ramas en Isolarion Forest [16]	24
Figura 2.11: Logotipo de TensorFlow	25
Figura 2.12: Logotipo de Keras.....	26
Figura 2.13: Logotipo de Scikit-learn.....	26
Figura 2.14: Logotipo de Dialogflow	26
Figura 2.15: Diagrama de funcionamiento de Dialogflow [21]	27
Figura 2.16: Logotipo de Firebase	28
Figura 2.17: Raspberry Pi 3 modelo B+	30
Figura 2.18: Pirámide de población española con discapacidad frente al total [27]	34
Figura 2.19: Proyección de la evolución de la población de la tercera edad en España [25]	35
Figura 3.1: Esquema de interacción entre las partes del sistema.....	38
Figura 3.2: Ejemplo ilustrativo del funcionamiento de la interfaz conversacional proactiva.....	40
Figura 3.3: Diagrama del flujo de los datos obtenido de los sensores.....	41
Figura 3.4: Página de configuración	46
Figura 4.1: Ajustes del agente creado para la interfaz conversacional.....	49
Figura 4.2: Ejemplo de frases de entrenamiento y respuestas de una intención.....	50
Figura 4.3: Creación de una entidad	52
Figura 4.4: Adición de una entidad en una intención	53
Figura 4.5: Contextos de entrada y salida	54
Figura 4.6: Ajustes de la interfaz conversacional para la integración con Google Assistant	55
Figura 4.7: Función en Firebase para consultar la temperatura	56
Figura 4.8: Circuito equivalente para los pines GPIO	57
Figura 4.9: Pines GPIO de la Raspberry Pi 3 modelo B+	58
Figura 4.10: Imagen y esquema de conexión del DHT11	58
Figura 4.11: Imagen y esquema de conexión del HC-SR501	59
Figura 4.12: Imagen y esquema de conexión del KY-026.....	59
Figura 4.13: Bucle principal para la lectura de los sensores	61
Figura 4.14: Aspecto de un Notebook de Python	62
Figura 4.15: Conjunto de datos de ejemplo con anomalías.....	63
Figura 4.16: Curvas ROC para los subconjuntos de entrenamiento y test.....	64
Figura 4.17: Ejemplo de lectura de un sensor desde el servidor	66
Figura 4.18: Función para enviar los avisos por correo electrónico	67
Figura 4.19: Subárbol de la base de datos para la información del usuario	69
Figura 4.20: Subárbol de la base de datos para la información de los sensores	70
Figura 4.21: Formulario para la configuración de un nuevo sensor	71

Figura 4.22: Configuración de nuevos sensores..... 73

Figura 5.1: Diagrama de Gantt 80

Figura 6.1: Expectativa de ventas de altavoces inteligentes en todo el mundo [36] 83

Figura 6.2: Principales usos de los altavoces inteligentes [37] 84

INDICE DE TABLAS

Tabla 4.1: Intenciones disponibles en la interfaz conversacional.....	51
Tabla 4.2: Entidades creadas en la interfaz conversacional.....	53
Tabla 5.1: Preguntas del cuestionario de evaluación.....	76
Tabla 5.2: Resultados de la evaluación	77
Tabla 5.3: Fechas y duración de cada tarea	78
Tabla 6.1: Presupuesto de personal.....	85
Tabla 6.2: Presupuesto de materiales.....	87
Tabla 6.3: Resumen de costes.....	88

CAPÍTULO 1

INTRODUCCIÓN

Este primer capítulo de introducción al Trabajo Final de Grado se exponen las motivaciones que han llevado a desarrollar un sistema para la asistencia a personas dependientes mediante el uso de la voz. También se incluyen los objetivos que se han completado para la realización del proyecto y las fases de desarrollo por las que se ha pasado. Por último, se detalla la estructura de la presente memoria indicando el contenido de cada sección.

1.1. Motivación

Se considera que una persona está en situación de dependencia personal cuando para desarrollar las actividades de su vida diaria necesita de algún tipo de ayuda de alguien o algo. El grado de dependencia de una persona se determina en función a diferentes escalas, que dependen de la legislación vigente en cada país. En algunos casos la persona presenta un grado de dependencia alto de forma repentina debido a accidentes o enfermedades. Sin embargo, en otros casos son grados de dependencia leve que aparecen con la edad. En estos casos la persona envejece y va perdiendo facultades poco a poco, lo que hace que puedan valerse por sí mismas en la mayoría de las actividades de su vida cotidiana, pero necesiten ayuda puntual.

En 2030, aproximadamente un cuarto de la población española superará los 65 años, además, la población de entre 30 y 50 años disminuirá [1]. Este escenario nos presenta una pirámide de población invertida, donde la base, las generaciones más jóvenes, son la menor parte de la población. A la vista de las previsiones, se hace muy necesario la creación de sistemas que permitan atender todas esas necesidades de las personas mayores que surgirán en el futuro con el menor personal posible, ya que la población activa tenderá a ser menor. Esta atención se debe realizar manteniendo siempre la máxima calidad posible.

En España se aprobó en 2007 un sistema público de atención a personas dependientes, pero solo un 26% de las personas que tienen reconocida una situación de dependencia tienen acceso a las prestaciones económicas o de servicios pertinentes [2]. Los servicios que se les ofrece a estas personas son muy variados, y tratan de adecuarse a las necesidades de la persona en cuestión. La Figura 1.1 muestra un resumen sobre las prestaciones que reciben las personas dependientes en España hasta 2017.



Figura 1.1: Personas que reciben una prestación por dependencia en España [2]

De entre todos los servicios que se ofrecen, el de teleasistencia representa aproximadamente el 16%. A estos datos de uso, habría que sumarle las personas de avanzada edad que, sin tener un grado de dependencia reconocido, hacen uso de este servicio ya que les permite vivir de forma independiente, pero teniendo la asistencia necesaria en caso de necesitarla. Como se comentaba anteriormente, la población está cada vez más envejecida y, además, es habitual que las personas de avanzada edad vivan solas, lo que dificulta el aviso de una emergencia o la gestión de estas. Es por todo esto que el sistema de teleasistencia es uno de los servicios sociosanitario más utilizado en todo el territorio nacional.

El sistema de teleasistencia clásico que se viene usando hasta ahora consta de un terminal que se conecta a la línea telefónica y que por tanto puede realizar llamadas. Este terminal se complementa con un botón que está conectado de forma inalámbrica con el terminal y que la persona debe llevar siempre encima. En caso de emergencia o cualquier otra necesidad, la persona debe pulsar el botón. El sistema es sencillo, pero requiere que la persona en cuestión lleve el botón siempre encima, para lo que el botón está dispuesto en forma de pulsera o colgante. La Figura 1.2 muestra un ejemplo de este botón destinado a la solicitud de asistencia.



Figura 1.2: Ejemplo de un botón para solicitud de asistencia en un sistema de teleasistencia

Aunque este sistema es efectivo y se ha estado usando hasta el día de hoy y de hecho se sigue usando, tiene una fuerte dependencia del botón de solicitud de ayuda. Sin este botón el sistema es prácticamente inservible, ya que la única manera de solicitar asistencia sería desde el terminal instalado en la vivienda, que posee otro botón con la misma función, pero obliga a la persona a desplazarse en una situación en la que puede no ser posible.

Estas situaciones de peligro pueden darse cuando la persona no lleve el botón de aviso consigo en todo momento. Incluso en los casos en que llevándolo puesto, se les puede olvidar que lo llevan. Estos casos pueden parecer aislados o muy remotos, pero pueden ocurrir con cierta frecuencia, si recordamos que estamos hablando de personas con algún tipo de dependencia. Es por esto por lo que este sistema tiene una dependencia muy grande del botón que, por motivos muy diversos como ya se ha comentado, podría no estar al servicio de la persona cuando de verdad lo necesite. Es por esto por lo que un sistema de teleasistencia que no precisase de botón sería de mucha utilidad.

La motivación fundamental de este Trabajo Final de Grado es la utilización de una interfaz conversacional mediante el uso de la voz, de forma que se prescinde del botón y se permite a la persona pedir ayuda en cualquier momento, simplemente usando la voz como si hubiera otra persona en la habitación escuchando. Además, se busca complementar este asistente con una serie de sensores que controlen el entorno y proporcionen avisos en caso de reconocer un peligro. En definitiva, se busca modernizar y automatizar el sistema clásico de teleasistencia mediante las nuevas tecnologías disponibles, haciéndolo más efectivo y accesible, a la vez que más amigable al ser menos intrusivo en la vida de la persona.

1.2. Objetivos

El principal objetivo del presente Trabajo Final de Grado es desarrollar un sistema de asistencia a personas con dependencia leve, que necesitan de un sistema que les pueda prestar la ayuda necesaria o hacer las gestiones para hacerles llegar esa ayuda siempre que la necesite eliminando la presencia de un botón de solicitud de ayuda. Se busca generar una interacción mucho más natural de la persona con el sistema ya que solo debe de utilizar la voz para realizar cualquier acción.

Haciendo uso de la gran cantidad de herramientas disponibles hoy en día, tanto en local como en la nube, se pretende desarrollar un sistema que sea útil y fácil de usar y que, a su vez, no sea intrusivo en la vida de la persona a la que asiste. Se va a crear un sistema que consta de principalmente de dos partes, una interfaz conversacional embebida en un altavoz inteligente con la que la persona podrá interactuar, y una segunda parte consistente en unos dispositivos que controlen el entorno en el que se encuentra la persona y puedan avisar de valores o comportamientos anómalos. Los objetivos parciales para poder desarrollar el sistema son:

- Análisis de los sistemas de teleasistencia, o similares, que existen actualmente y los servicios que ofrecen.
- Estudio de las situaciones que se pueden dar en las personas dependientes a las que van destinados estos sistemas.
- Construir un sistema lo más modular posible, de forma que se adapte a diferentes necesidades.
- Análisis de los dispositivos existentes actualmente sobre los que se puede implementar un asistente por voz.
- Implementar una interfaz conversacional proactiva.
- Análisis de las bases de datos no relacionales en tiempo real, así como otras tecnologías de almacenamiento de datos.
- Desarrollo de un software embebido en una placa electrónica de prototipado, que sea capaz de controlar los distintos elementos de hardware que tenga conectados.
- Reconocimiento de datos anómalos mediante el uso de algoritmos de aprendizaje automático.

Además de estos, los objetivos personales que se busca conseguir al final del presente Trabajo Final de Grado son los siguientes:

- Ampliar conocimientos en el ámbito de la automatización y más concretamente en el uso de asistentes por voz.
- Ampliar conocimientos en el ámbito del internet de las cosas, el uso de sensores conectados permanentemente a internet transmitiendo sus datos.
- Adquirir técnicas y hábitos de investigación y trabajo que permiten desarrollar sistemas destinados a la automatización de tareas utilizando tanto software, mediante programas en diferentes lenguajes, como hardware, mediante el uso de distintos sistemas electrónicos.

- Desarrollar un sistema útil, que pueda ser usado, poniendo así la primera piedra para la mejora de los servicios clásicos de teleasistencia y de asistencia en general, mejorando por ende la calidad de vida de sus usuarios.

1.3. Fases del desarrollo

Para el desarrollo de este Trabajo Fin de Grado se han aplicado los conocimientos adquiridos en las diferentes asignaturas cursadas como parte del Grado en Ingeniería Electrónica Industrial y Automática. Otros conocimientos adquiridos como parte de mi experiencia laboral y espíritu autodidacta también han sido de gran utilidad para tratar de realizar el mejor trabajo posible.

El desarrollo se ha dividido en diferentes fases con el objetivo de poder seguir una metodología de trabajo adecuada. Las fases y su contenido se detalla a continuación.

1.3.1. Planificación

En esta primera fase se ha realizado una investigación sobre el estado actual de los sistemas de teleasistencia, que prestan servicio a las personas dependientes. Se ha estudiado su funcionamiento y observado sus puntos de posible mejora. De esta investigación se han obtenido los objetivos de este Trabajo Fin de Grado. Se han revisado las tecnologías disponibles para la implementación propuesta y otras tecnologías alternativas para contrastar las diferentes posibilidades y se han seleccionado los recursos necesarios para la realización del proyecto.

1.3.2. Ejecución

En esta fase se ha realizado un boceto inicial de las funcionalidades deseadas para implementar en el sistema basándose en los objetivos marcados. Se han introducido en este primer boceto las tecnologías necesarias para la ejecución. Basándose en este diseño, se ha procedido al desarrollo del sistema. Se ha buscado, como se detalla en capítulos posteriores, crear un sistema lo más modular posible que pueda ser adaptable a cualquier usuario.

1.3.3. Evaluación

En esta fase se han realizado pruebas a distintos niveles. Las pruebas del correcto funcionamiento del sistema se han ido realizando durante el desarrollo del mismo, de forma que se van probando pequeñas características para asegurar su correcto funcionamiento. Se han realizado también pruebas con distintas personas para comprobar el funcionamiento del conjunto del sistema. De esta forma también se han recogido posibles puntos de mejora. Por último, se ha realizado la evaluación del sistema mediante una encuesta al conjunto de personas que han utilizado el sistema, con una serie de preguntas para conocer de forma independiente su percepción del funcionamiento del sistema.

1.3.4. Documentación

En esta fase se ha documentado todo el trabajo realizado. Para ello se ha redactado el presente documento, donde se explica con detalle todo el trabajo realizado para completar el presente Trabajo Fin de Grado. Se ha creado también una presentación de este proyecto para realizar su defensa ante el tribunal.

1.4. Medios empleados

Para el desarrollo del presente Trabajo Final de Grado se han empleado los siguientes recursos:

1.4.1. Recursos hardware

- Ordenador portátil MSI (Intel Core i7-8550U, Gráfica MX150, 16GB de RAM)
- Google Home Mini
- Raspberry Pi 3 Model B (1.2 GHz Quad-core ARM Cortex-A53, 1GB RAM)
- Sensores de prototipado (humedad y temperatura, llama y presencia)
- Cables de conexión y *protoboard*
- Pantalla externa LG (23.8", 1920x1080, IPS)

1.4.2. Recursos software

- Ubuntu 18.04
- Microsoft Visual Studio Code
- Python 3.5
- Scikit-learn 0.21
- Node.js 10.16
- Dialogflow
- Firebase
- Repositorio Git en Gitlab
- GitKraken (cliente de Git para Linux)
- Windows 10
- Microsoft Office 365 ProPlus
- XODO PDF Reader
- Draw.io
- ClickUp, para la organización de las tareas
- Google Chrome v.72

1.5. Estructura de la memoria

La memoria está estructurada en 7 capítulos. En este epígrafe se explica el contenido de todos ellos y la función que cumplen dentro de esta memoria. Además, se incluye también los apartados de glosario y bibliografía.

Capítulo 1: Introducción

En este primer capítulo se introduce la motivación que ha llevado a la realización de este Trabajo Final de Grado, así como los objetivos fijados para su realización. También se expone el material, tanto software como hardware, que se ha empleado en la realización del proyecto y las fases en las que se ha dividido el desarrollo del sistema. Por último, se hace una pequeña presentación de la estructura que presenta la memoria y del contenido de cada apartado.

Capítulo 2: Estado del arte

En este capítulo se da una visión general del estado en el que se encuentran actualmente las tecnologías y medios que se van a utilizar en este proyecto. Se hará un repaso por las diferentes interfaces conversacionales disponibles, también denominados chatbots. Se revisarán las distintas formas que existen de implementarlos para su uso. Se explicarán también brevemente los algoritmos de aprendizaje de automático y el porqué de su gran potencial. Además, se presentará la plataforma Raspberry Pi, con la cual se pueden realizar prototipos en el ámbito de la electrónica y la informática. Se analizarán también las características que presentan los sistemas de teleasistencia clásicos. Por último, se realizará un estudio de las situaciones de dependencia que se dan y cuáles son sus componentes principales.

Capítulo 3: Descripción general del sistema

En este capítulo se da una visión general del sistema desarrollado para explicar su funcionamiento y como se resuelve el problema planteado. Se explican sus elementos y partes más representativos, así como la relación entre ellos.

Capítulo 4: Descripción detallada del sistema

En este capítulo se profundiza sobre la descripción dada en el capítulo anterior y se detalla de forma íntegra la composición y funcionamiento de cada una de las partes del trabajo realizado de forma más técnica. En cada sección de este capítulo se exponen todas las funcionalidades que lo componen y como se han realizado.

Capítulo 5: Evaluación y gestión del proyecto

En este capítulo se expone la evaluación del sistema realizada, como se ha realizado dicha evaluación y los resultados obtenidos de la misma. Se expone también la planificación temporal del proyecto completo de forma gráfica con un diagrama de Gantt.

Capítulo 6: Marco regulador y entorno socio-económico

Es este capítulo se busca analizar las regulaciones que son, o podrían ser, aplicables a la solución propuesta. Además, se añade un análisis detallado del impacto tanto social como económico que tendría la solución propuesta, junto con un presupuesto que describe el coste de aplicación tanto de costes directos como indirectos.

Capítulo 7: Conclusiones y trabajo futuro

En este último capítulo se realiza una valoración general del trabajo realizado una vez finalizado, explicando las ideas principales y conclusiones que se han obtenido de su realización. Además, se explican cuáles pueden ser las líneas futuras de trabajo en base a este proyecto.

Glosario

En este apartado se recopilan por orden alfabético todo los términos y conceptos técnicos que son de difícil comprensión o poco descriptivos, por lo que se ofrece una descripción detallada de dicho concepto para facilitar la lectura y comprensión de esta memoria.

Bibliografía

En este apartado se recopilan todas las referencias bibliográficas que se han utilizado para la consulta o extracción de datos durante la redacción de la presente memoria, y que han servido por tanto también como base para el desarrollo completo de este Trabajo Final de Grado. La bibliografía sigue el estilo IEEE.

Capítulo 2

ESTADO DEL ARTE

El presente capítulo tiene como propósito dar una visión general del panorama actual en las tecnologías y materias tratadas en este trabajo. Para ello se hará en primer lugar un estudio de lo que es una interfaz conversacional, la forma en la que reconoce el lenguaje natural, procesa los datos y responde de la misma forma. Se procederá a hacer un breve resumen de cómo funciona la interacción por voz, especificando como es la interacción hombre-máquina. En segundo lugar, se hará una revisión de los dispositivos que soportan este tipo de asistentes. Estos dispositivos son muy variados por lo que se expondrán sus principales características y sus diferencias.

A continuación, se realizará un breve resumen de los tipos de algoritmos de aprendizaje automático y se explicará más en detalle uno de ellos. También se expondrán las cualidades y características de la plataforma Raspberry Pi, la cual permite realizar prototipos de forma económica en el ámbito de la electrónica y la informática.

Por último, se realizará una revisión de cómo funcionan los sistemas de teleasistencia disponibles, que características y funcionalidades tienen, para poder así tener un claro punto de partida en el desarrollo del proyecto. Por último, se entrará a profundizar en las situaciones que se le pueden presentar a una persona dependiente, porque se producen, síntomas previos que se pudiesen detectar, así como las características demográficas de este grupo de población.

2.1. Interfaces conversacionales

Las interfaces conversacionales están sufriendo un crecimiento exponencial, ya que las diferentes empresas se han dado cuenta de que es una forma sencilla y natural para el usuario de interactuar con sus productos. En la última campaña navideña, 1,3 millones de españoles tenían pensado comprar alguno de los altavoces inteligentes disponibles en el mercado y que integran una interfaz conversacional [3]. El crecimiento de estos sistemas ha sido tan exponencial que el asistente de Amazon, Alexa, se

colapsó el día de Navidad debido a que el uso que se le dio superó cualquier expectativa [4]. Su uso actualmente, aunque muy variado, todavía no aprovecha todo su potencial. Es por ello por lo que muchas empresas están desarrollando sus propias interfaces conversacionales para interactuar de una forma más natural con sus clientes. De hecho, ya nos encontramos con estas interfaces cuando llamamos diferentes servicios de atención al cliente.

2.1.1. ¿Qué es una interfaz conversacional?

Una interfaz conversacional es un software que hace de intermediario entre sistemas computacionales y el usuario, automatizando tareas con la mínima interacción posible entre el usuario y la máquina. Para ello, se busca que la interacción entre el hombre y la máquina se haga de la forma más natural posible, por lo que el usuario utiliza su lenguaje natural, ya sea hablado o escrito, para comunicar sus intenciones a la máquina. Esta debe hacer un reconocimiento del lenguaje natural, ya que no funciona con comandos preestablecidos, y responder con la misma naturalidad. La Figura 2.1 muestra como es el uso de la interfaz conversacional Google Assistant en un teléfono móvil inteligente.

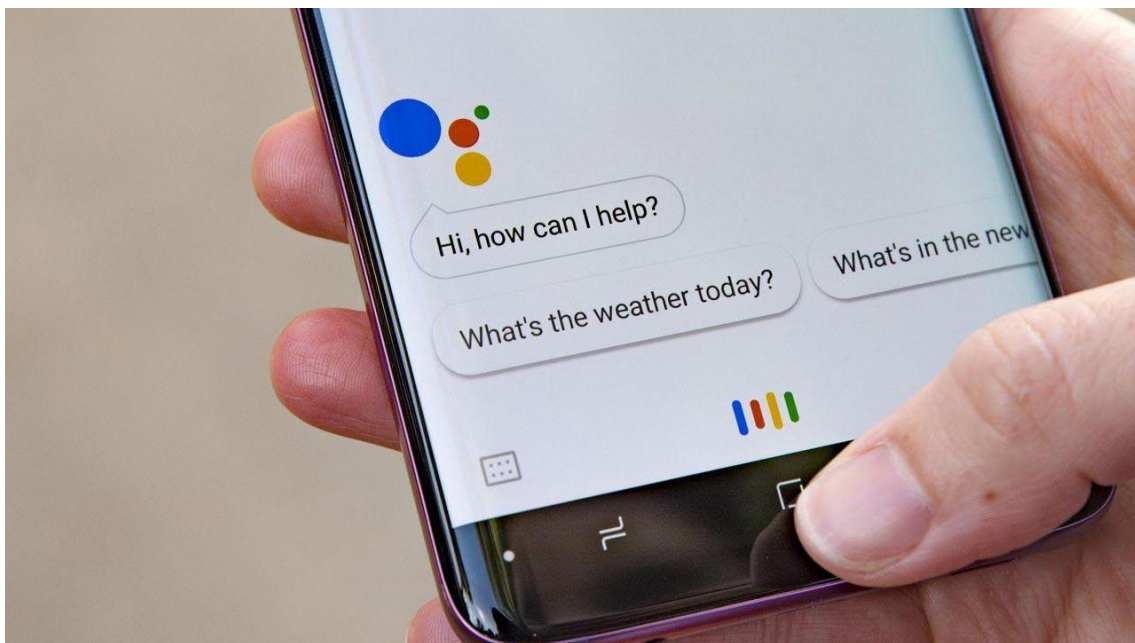


Figura 2.1: Google Assistant ejecutándose en un teléfono Android

La interfaz conversacional tiene sus orígenes en un proyecto militar llamado CALO que buscaba agrupar tecnologías de inteligencia artificial para construir un asistente cognitivo [5]. Posteriormente, el *manager* de este proyecto formó junto con algunas personas del centro de inteligencia artificial un equipo para la creación de Siri. Unos 20 años antes del lanzamiento de la primera computadora personal de IBM, esta misma empresa desarrolló el IBM Shoebox, el primer sistema capaz de ejecutar el reconocimiento de voz. Solo era capaz de reconocer 16 palabras y los números del 0 al 9. Más tarde, en la década de los 70, el proyecto Harpy consiguió dominar un millar de

palabras, lo que lo asemejaba a un niño de 3 años. En 1994 IBM presentó un teléfono inteligente con una interfaz conversacional integrada que sentó las bases de para lo que conocemos hoy en día. La primera interfaz conversacional realmente funcional fue instalada en un *smartphone* en 2011, cuando Apple, después de comprar Siri Inc lo incorporó en su iPhone 4S [6].

El nombre de interfaz conversacional, o bot conversacional proviene del inglés “chatbot”, aunque el nombre de bot se suele aplicar a los más rudimentarios, como veremos más adelante. A estas interfaces también se les denomina asistentes virtuales, los más avanzados, aunque esta denominación se puede confundir con la profesión de asistente virtual, la cual desarrolla una persona desde su casa o una oficina personal realizando tareas de asistencia técnica o administrativa a personas o empresas de forma totalmente online, externalizando así este servicio.

Las interfaces conversacionales se podrían clasificar en tres tipos dependiendo del nivel de inteligencia que demuestren. El CTO de IBM Watson, la inteligencia artificial cognitiva de IBM, Rob High propone una división en tres fases diferentes para clasificar estas interfaces inteligentes [7]. Estas tres fases son las siguientes:

- Las interfaces que se encuentran en la primera fase, normalmente denominados “chatsbots”, son las que realizan una interpretación del comando que recibe, la asocia a una tarea específica y entrega una respuesta.
- En una segunda fase posterior se situaría interfaces capaces de enzarzarse en una conversación y comprender los deseos del usuario. En estas interfaces, las preguntas serían cada vez más concisas para averiguar cuál es la intención del usuario. Por lo tanto, estas interfaces serían capaces de resolver problemas de cierta complejidad.
- En la última fase, la más avanzada, la interfaz construiría su propia personalidad. La interfaz debería conocer profundamente al usuario, para poder construir una personalidad afín al usuario y por tanto que el usuario confíe en él. Además, la comprensión del lenguaje debe de ser óptima y la respuesta de una calidad alta.

Para la interacción con estos sistemas, existen varias posibilidades. El más común y que se está utilizando más últimamente es la interacción por voz. Esta presenta múltiples ventajas ya que el usuario no necesita saber leer o escribir. Además, es un sistema manos libres, lo que significa que el usuario puede estar realizando otra tarea mientras mantiene una conversación con el asistente. A diferencia de la interacción mediante texto, que veremos a continuación, con la voz se puede hacer uso de la entonación para darle más riqueza a la conversación. La interacción con la interfaz conversacional se puede realizar también mediante el uso de texto. Estos asistentes se implementan sobre todo en páginas web o en aplicaciones de mensajería, ya que son lugares que se visitan desde un dispositivo que cuenta con un teclado. Uno de los puntos fuertes de estos sistemas es el hecho de que no necesitan transformar la voz del usuario a texto, lo que les facilita el trabajo, pero, por otro lado, se necesita algún dispositivo con teclado y pantalla para poder interactuar con ellos. Por último, existen algunos asistentes

con los que se puede interactuar mediante imágenes, como Bixby de Samsung, aunque apenas son usados.

La mayoría de las interfaces existentes hoy en día se encuentran en la primera o la segunda fase. Una conversación entre personas no es un proceso sencillo. La mayoría de los niños pequeños no dicen su primera palabra hasta los 9 meses de media y no es hasta los 3 años que pueden hacerse entender [8]. Además, por experiencia sabemos que incluso de adultos cometemos errores, ya que comunicarse mediante la voz no es un proceso sencillo, y por lo tanto replicarlo en un software es muy complicado. No obstante, se están realizando avances que auguran un futuro muy prometedor para estas interfaces conversacionales.

2.1.2. Interfaces conversacionales disponibles actualmente

En la actualidad, como se comentaba anteriormente, las interfaces conversacionales están en auge. Existen multitud de ellas, ya que como veremos más adelante, cualquier persona puede crear una. No obstante, veremos solo las que están más desarrolladas y son de una calidad considerable. Todas ellas además comparten la característica de que están diseñadas para la conversación en general, es decir, se les pueden hacer preguntas sobre tiempo, deportes, economía, búsqueda de información, etc. Muchas de ellas están integradas en asistentes completos, es decir, que además de la voz se pueden usar otros métodos de interacción.

2.1.2.1. Alexa

Alexa es una interfaz conversacional desarrollada por Amazon. Esta permite reproducir música, hacer comprobaciones del tráfico o del tiempo o buscar noticias entre otras. Además, al estar desarrollada por Amazon, también permite realizar compras en el sitio web de la compañía y crear listas de seguimiento de productos. En 2017, Amazon contaba con una división de más de 5000 personas trabajando en su interfaz conversacional Alexa, lo que hace visible su fuerte apuesta por estas tecnologías. Actualmente Alexa está disponible en inglés, alemán, japonés, francés, italiano y español.

Su lanzamiento se produjo en 2014 junto con los primeros dispositivos Echo, de los que hablaremos posteriormente. Su creación fue inspirada por el sistema de conversación que aparece en las películas de Star Trek. Según sus creadores, el nombre de Alexa se eligió por contener la 'x', la cual es una consonante fácilmente reconocible por el asistente. La Figura 2.2 muestra el logotipo de la interfaz Alexa de Amazon.



Figura 2.2: Logotipo de Alexa

Alexa se encuentra normalmente embebida en los Amazon Echo, que son los altavoces inteligentes desarrollados por la compañía. También se puede encontrar en su aplicación móvil y en los dispositivos de otras marcas gracias a que Amazon les permite integrar las capacidades de Alexa en sus productos. Además, esta interfaz conversacional cuenta con las denominadas “skills”, que son el equivalente a las aplicaciones de los *smartphones*. Con ellas se pueden añadir nuevas funcionalidades creadas por terceros al asistente.

2.1.2.2. Cortana

Cortana es una interfaz conversacional desarrollada por Microsoft. Su desarrollo comenzó en 2009 y no se dio a conocer al público hasta 2013. Se desarrolló como una parte clave para el cambio de imagen que se buscaba en Windows. Para su diseño el equipo entrevistó a personas cuya profesión era la de asistente, lo que inspiró una serie de características únicas de Cortana. Su nombre sale de la franquicia de videojuegos Halo, perteneciente a Microsoft, y en la cual un personaje de inteligencia artificial lleva ese nombre.

Cortana se encuentra integrada en prácticamente todas las aplicaciones y dispositivos de Microsoft, desde Skype hasta la Xbox. Actualmente Cortana está disponible en inglés, chino, francés, alemán, italiano, japonés, portugués y español. La Figura 2.3 muestra el logotipo de la interfaz Cortana de Microsoft.



Figura 2.3: Logotipo de Cortana

2.1.2.3. Siri

Siri es una interfaz conversacional que comenzó siendo desarrollada por el centro de inteligencia artificial de SRI Venture group, y más tarde por Apple, después de que esta adquiriera Siri. Este asistente nació como una rama del proyecto CALO, el cual se describió como uno de los mayores proyectos de inteligencia artificial de la historia, y se centra en las áreas de interfaces conversacionales, reconocimiento del contexto personal y la delegación de servicios.

Siri está presente en todos los dispositivos de la compañía de la manzana y actualmente está disponible en veinte idiomas incluido el español. Como el resto de las interfaces de las otras compañías, Siri te permite una interacción conversacional con la mayoría de las aplicaciones de los dispositivos que la integran, como recordatorios, contactos, el tiempo, música, reloj, etc. La Figura 2.4 muestra el logotipo de la interfaz Siri de Apple.



Figura 2.4: Logotipo de Siri

2.1.2.4. Assistant

Assistant es una interfaz conversacional desarrollada por Google. A diferencia de Google Now, el cual se podría considerar su antecesor, Assistant es capaz de participar en conversaciones bidireccionales. Este asistente fue presentado por Google en mayo de 2016 junto con el que sería el principal dispositivo que haría uso del asistente, el Google Home.

Esta interfaz conversacional puede ajustar alarmas, cambiar los ajustes del dispositivo, responder preguntas, etc. En la presentación también se mostró Duplex, una evolución donde se le podía pedir al asistente que nos pidiese cita en la peluquería, y esta realizaba la llamada, de una manera sorprendentemente natural se comunicaba de forma autónoma con el personal de la peluquería para solicitar una cita. Esta es una funcionalidad que todavía no está disponible al público, pero que deja ver el gran potencial que tiene esta interfaz. La Figura 2.5 muestra el logotipo de la interfaz Assistant de Google.

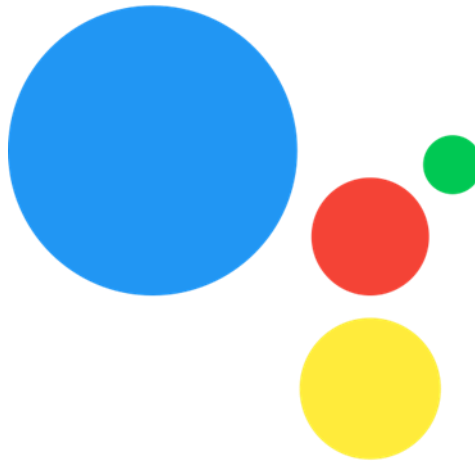


Figura 2.5: Logotipo de Assistant

2.1.3. Dispositivos que integran una interfaz conversacional

Las interfaces conversacionales pueden encontrarse en una gran variedad de dispositivos. Los requisitos que deben cumplir estos dispositivos son variados. Dependiendo de la forma en la que se pueda interactuar con el asistente, serán necesarios elementos diferentes. Si, por ejemplo, el interfaz solo admite la entrada por teclado, será necesaria una forma de introducir el texto. Si por el contrario el asistente admite la entrada por voz, se necesitará de un micrófono.

Por norma general, todos necesitarán de una conexión a internet, para analizar la entrada y proporcionar una salida, ya que no se encuentran almacenados en local en el dispositivo. La mayoría de los asistentes tratados anteriormente cuentan con la posibilidad de interactuar con ellos mediante un teclado y una pantalla o mediante el uso

de la voz gracias a un micrófono y un altavoz, dado que están pensados para usarse principalmente en dispositivos móviles.

Lo más habitual es encontrar estos asistentes que hacen uso de interfaces conversacionales en dispositivos móviles, ya que es un dispositivo que cuenta con todos los requisitos necesarios para integrarlo y además es un dispositivo que tiene la gran mayoría de la población. También los podemos encontrar en páginas web y aplicaciones móviles. El auge de este tipo de tecnologías ha hecho que nos podamos encontrar estos asistentes en prácticamente cualquier dispositivo con conexión a internet. Hoy en día podemos encontrar esta tecnología en automóviles, relojes inteligentes e incluso electrodomésticos. La Figura 2.6 muestra la interfaz conversacional Google Assistant siendo usada desde un reloj inteligente.



Figura 2.6: Google Assistant ejecutándose en un reloj inteligente

No obstante, el dispositivo que más ha aumentado su presencia en nuestras vidas en los últimos meses son los altavoces inteligentes. Estos son dispositivos diseñados y pensados exclusivamente para la interacción con las interfaces conversacionales. Se confunden entre el resto de los elementos de nuestra casa y te permiten tener acceso a una interfaz conversacional muy completa, generalmente solo mediante el uso de la voz, lo que hace que sean extremadamente fáciles de usar. Además, son compatibles con una gran cantidad de dispositivos de domótica e IoT, lo que permite al usuario vincular todos estos elementos y poder controlarlos mediante el uso de la voz.

2.2. Altavoces inteligentes

Un altavoz inteligente es un dispositivo bastante simple, que cuenta con un micrófono, para recoger las peticiones del usuario, y un altavoz, para poder proporcionar la respuesta. Está conectado a internet, generalmente mediante Wi-Fi. Es capaz de recibir peticiones de todo tipo usando lenguaje natural y proporcionar una respuesta. Tanto el procesamiento del lenguaje natural de la petición del usuario, como el procesamiento de la respuesta necesaria se realizan generalmente fuera del dispositivo, en los servidores de la compañía propietaria de la interfaz conversacional.

La lista de altavoces inteligentes en el mercado crece diariamente, dada la alta demanda que están teniendo. Solo durante el año pasado, los envíos de este tipo de dispositivos aumentaron un 197% hasta llegar a más de 22 millones de unidades [9]. Este aumento ha venido muy unido a la presentación de los principales asistentes en su versión en español. En el tercer trimestre de 2017 el líder indiscutible de ventas era Amazon con un casi un 75% del mercado. Sin embargo, en el mismo periodo de 2018, cuenta solo con un 32%, seguido muy de cerca por Google con casi un 30% [10]. El tercero en la lista es Apple, con cifras más modestas.

Además de las expectativas de ventas de estas compañías, que son muy positivas, está apareciendo otras compañías como Xiaomi, que entrarán con fuerza los próximos años.

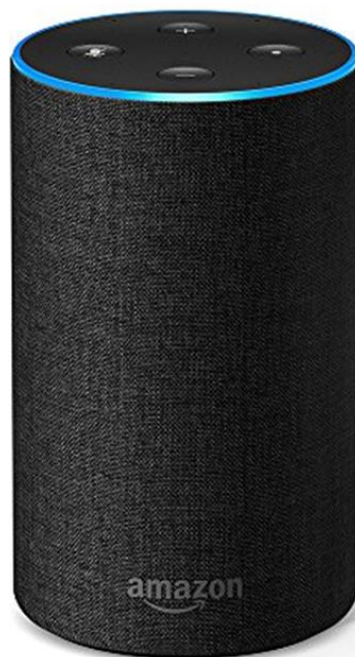
2.2.1. Google Home

Fue anunciado en mayo de 2016 para competir contra el dispositivo de Amazon. Google Home es el principal altavoz inteligente de la compañía, que también cuenta con una versión mini y otra max, las cuales cuentan con las mismas funcionalidades y se diferencian en tamaño y precio. Además, cuenta con el Google Home Hub, el cual cuenta con una pantalla táctil de 7", usada para complementar las respuestas con material visual. En su versión estándar cuenta con un altavoz de 50mm y dos radiadores pasivos [11]. Todos ellos cuentan con un botón físico que permite desactivar el funcionamiento de los micrófonos. La Figura 2.7 muestra una imagen del altavoz inteligente Google Home.

*Figura 2.7: Google Home*

2.2.2. Amazon Echo

Los dispositivos Echo empezaron a ser desarrollados por Amazon en 2010, pero no fue hasta 2014 cuando se empezaron a distribuir. Al inicio solo se podía acceder a ellos por invitación. El Echo cuenta con diferentes versiones, todas con la misma interfaz conversacional, que varían en su tamaño y calidad de audio. Además, algunas versiones cuentan con una pantalla para proporcionar información extra. En su versión estándar dispone de un altavoz de 16mm y un subwoofer de 63mm [12]. Todas las versiones cuentan con un botón físico que permite desactivar el funcionamiento de los micrófonos. La Figura 2.8 muestra una imagen del altavoz inteligente Amazon Echo.

*Figura 2.8: Amazon Echo*

2.2.3. Apple HomePod

El HomePod fue presentado por Apple en febrero de 2018. Es el único altavoz inteligente del que dispone la compañía actualmente. Este dispositivo cuenta con siete altavoces distribuidos a lo largo de su contorno y con un subwoofer de 4" enfocado hacia la parte superior. Además, cuenta también con siete micrófonos para poder recibir peticiones por voz desde cualquier posición a su alrededor y una pantalla en su parte superior para mostrar información. La Figura 2.9 muestra una imagen del altavoz inteligente Apple HomePod.



Figura 2.9: Apple HomePod

2.3. Aprendizaje automático

El aprendizaje automático o aprendizaje de máquinas (del inglés, *machine learning*) es una rama de la inteligencia artificial. Su objetivo es desarrollar técnicas que doten a los ordenadores de la capacidad de aprender. Se dice que un algoritmo aprende cuando es capaz de reducir el error en los resultados con el paso de los ejemplos. La habilidad de predecir los datos de salida no es una cualidad que el algoritmo posea cuando se concibe, sino que la va aprendiendo [13]. Más concretamente, son algoritmos que son capaces de generalizar el comportamiento y reconocer los patrones presentes en la información que se le suministra.

Los algoritmos de aprendizaje automático se utilizan cuando se cuenta con una gran cantidad de datos con muchas variables y por tanto es difícil ver de forma manual cual es el patrón que estos datos siguen. El algoritmo encuentra las reglas y ecuaciones que siguen los datos, que de hacerlo de forma manual sería tremendamente complejas, como puede

ser el reconocimiento de voz. Si los datos cambian el algoritmo es capaz de ajustar las reglas que ha encontrado a esos nuevos datos, por lo que también son muy usados en aplicaciones donde las reglas de una tarea cambian constantemente, como en el caso de la detección de fraudes.

El aprendizaje automático es similar a la minería de datos, ya que ambos buscan los patrones que puedan estar en los datos. Al contrario que en la minería de datos, donde los datos se extraen para la comprensión humana, los algoritmos de aprendizaje automático utilizan el patrón encontrado para ajustar las acciones del programa en consecuencia.

2.3.1. Tipos de algoritmos

Los algoritmos de aprendizaje automático se agrupan en función de los datos que se le proporcionan para aprender y la salida de estos [14]. Los principales grupos son los siguientes:

- **Aprendizaje supervisado:** Este tipo de algoritmos producen una correlación entre los datos de entrada y los de salida del sistema analizado. Estos algoritmos son entrenados con una serie de ejemplos de los cuales se sabe la entrada del sistema y la salida. Al conocer la salida, se puede calcular el error que comete el algoritmo en sus predicciones y por tanto optimizar esta función de coste para mejorar los resultados. Se denomina supervisado ya que se puede conocer el error de las predicciones del modelo.
- **Aprendizaje no supervisado:** Este tipo de algoritmos reciben como únicos datos las entradas del sistema a modelar y no se proporcionan las salidas. Por ello se denominan no supervisados, ya que no se puede calcular el error que comete ya que no se conocen las categorías de esos ejemplos. El algoritmo debe ser capaz de reconocer los patrones para poder etiquetar las nuevas entradas.
- **Aprendizaje semi-supervisado:** Este tipo de algoritmos combinan los dos tipos anteriores. Se tienen en cuenta tanto datos categorizados como aquellos de los que no se conoce su categoría.
- **Aprendizaje por refuerzo:** Este tipo de algoritmos recibe como información de entrada el feedback que recoge del entorno que le rodea, como respuesta a las acciones que realiza. El algoritmo aprende mediante el sistema de prueba y error. El algoritmo debe aprender cómo se comporta el entorno en el que se encuentra mediante la optimización de la señal de valor, la cual le indica cuando toma la decisión correcta y cuando se equivoca.

2.3.2. Técnicas de clasificación

Como se ha comentado en los epígrafes anteriores, el objetivo de los algoritmos de aprendizaje automático es modelar un sistema e inferir el patrón que siguen los datos que se le proporcionan y usar estas reglas que ha ido aprendiendo para dar una predicción correcta a nuevos datos del mismo sistema. Para conseguir este objetivo, se han desarrollado diferentes técnicas. Las principales técnicas utilizadas son las siguientes [15]:

- **Árboles de decisión:** Este es un modelo predictivo que dado un conjunto de datos fabrica diagramas lógicos de predicción similares a los basados en reglas. De esta forma se etiquetan sucesos que ocurren consecutivamente. El árbol está compuesto por nodos, que se dividen en dos o más ramas. Los datos llegan a los distintos nodos del árbol a través de las ramas, y es en los nodos donde se decide por qué rama continua según la función de evaluación de ese nodo. En cada nodo se toma una decisión que va categorizando los datos.
- **Algoritmos genéticos:** Estos algoritmos son búsquedas heurísticas que tratan de imitar el proceso de selección natural. Se genera una primera población de individuos y sobre parte de esta se aplicarán mutaciones. Esto hará que unos individuos den mejores soluciones que otros, por lo que se seleccionarán solo aquellos que den las mejores respuestas.
- **Redes neuronales artificiales:** Estos algoritmos, al igual que los algoritmos genéticos, también son de inspiración biológica, pero en este caso se basan en el funcionamiento de los sistemas nerviosos de los animales. La unidad básica de estos algoritmos es la neurona. Esta recibe como entradas las salidas de las neuronas de la capa anterior y las pondera en función a un peso, que son los parámetros que debe aprender. La suma de estas entradas ponderadas es pasada a una función de activación que produce la salida de la neurona, que será una entrada en las neuronas de la siguiente capa. Las neuronas se agrupan en capas, de tal forma que cada neurona de una capa recibe como entrada las salidas de todas las neuronas de la capa anterior.
- **Máquinas de soporte vectorial:** Este tipo de algoritmos crean un modelo que representa los datos en un espacio que queda separado por un hiperplano de la manera más amplia posible. Este hiperplano está definido por el vector entre los dos puntos de las dos clases más cercanos. Se busca con esto la “separación óptima”, es decir, el hiperplano con la máxima distancia con los puntos de cada categoría más cercanos al hiperplano.
- **Clustering:** Estos algoritmos, también denominados de agrupamiento, analiza las características de cada elemento para agrupar aquellos que cuenten con características similares. Normalmente, los elementos de un

mismo grupo comparten propiedades comunes y está suficientemente diferenciadas de las propiedades de otros grupos.

- **Redes bayesianas:** Este es un modelo probabilístico en forma de grafo que representa un conjunto de variables aleatorias y las dependencias condicionales que estas tienen. Cada nodo de este grafo representa una de las variables aleatorias y cada nodo tiene asociada una función de probabilidad. Una red bayesiana representa las relaciones probabilísticas entre las entradas y las salidas, de manera que dadas ciertas entradas se puede calcular las probabilidades de que se den ciertas salidas.

2.3.3. Isolation Forest

Isolation Forest es un algoritmo de aprendizaje automático no supervisado basado en árboles de decisión. No tiene una traducción oficial al castellano, pero podría traducirse por “bosque de aislamiento”, ya que utiliza multitud de árboles de decisión para aislar los datos anómalos. El algoritmo se basa en la idea de que aislar las anomalías dividiendo los datos de forma aleatoria supone sustancialmente menos iteraciones que aislar un dato no anómalo, ya que estos últimos están agrupados.

Funciona de la siguiente manera. Supongamos un conjunto de datos que tienen dos características, Q1 y Q2.

1. Se selecciona aleatoriamente un punto del conjunto de datos para comprobar si es una anomalía o no.
2. Para cada característica, se establece el rango posible para la división del conjunto de datos entre el máximo y el mínimo dato de esa característica. Es decir, si el valor más alto del conjunto de datos en esa característica fuese 10 y el mínimo 5, la división del conjunto de datos por esa característica sería entre el 10 y el 5.
3. Se selecciona aleatoriamente una de sus características, Q1 ó Q2.
4. Se selecciona aleatoriamente un valor dentro del rango para la característica seleccionada en el paso 3. Si el valor seleccionado es superior al punto que se desea aislar en esa característica, se cambia el máximo del rango por este nuevo valor seleccionado. Lo mismo si el valor fuese inferior, en cuyo caso se ajustaría el mínimo.
5. Se repiten los pasos 3 y 4 hasta que el punto que se quiere comprobar es el único dentro del rango que se ha ido ajustando para todas las características.

Aquellos puntos del conjunto de datos que estén separados del resto y por tanto no sigan el patrón de los datos, necesitarán menos iteraciones del algoritmo para ser

aislados, ya que al estar más separados el rango dentro del que quedarán aislados podrá ser mayor. Estas separaciones sucesivas del conjunto de datos van generando un árbol de decisión, el cual tendrá sustancialmente menos ramas si corresponde a una anomalía. Este proceso se repite para todos los puntos una cantidad x de veces, por lo que se generan multitud de árboles, de ahí la denominación de bosque [16].

En la Figura 2.10 podemos observar la diferencia entre el número de iteraciones (y por tanto el número de ramas del árbol) necesarias para aislar un dato no anómalo (x_i) de otro que si lo es (x_o).

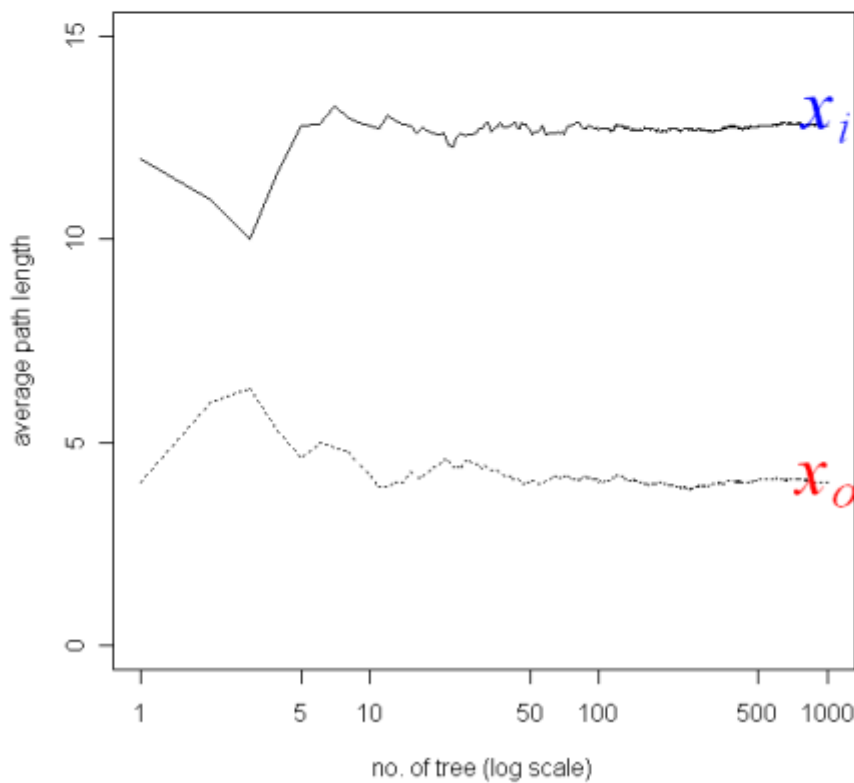
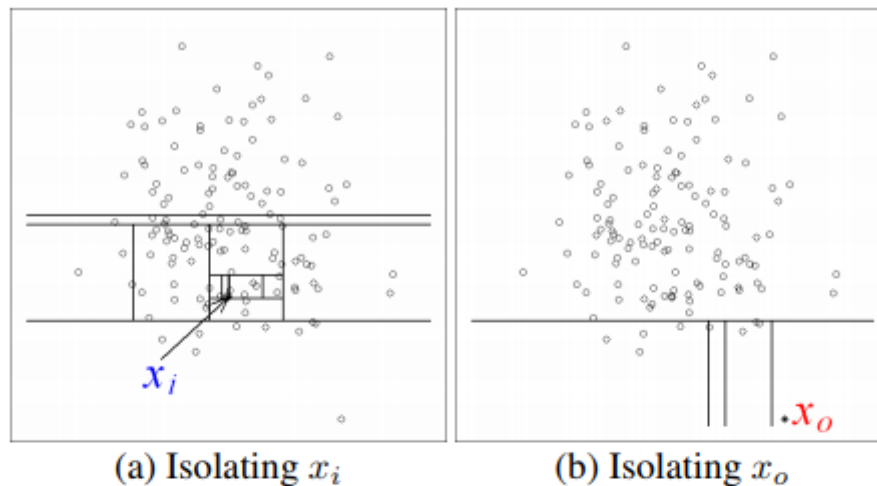


Figura 2.10: Comparativa del número de ramas en Isolation Forest [16]

Existen variaciones de este algoritmo que buscan hacerlo más eficiente, ya que al ser no supervisado es difícil controlar su rendimiento. Algunas variaciones consisten en analizar solo los datos que caigan dentro de una ventana e ir moviendo esa ventana a lo largo del conjunto de datos [17].

2.3.4. Librerías de algoritmos de aprendizaje automático

Cuando se desea implementar un algoritmo de aprendizaje automático, no es habitual escribir todo el código necesario desde cero, salvo que se esté realizando alguna investigación al respecto. Para la producción, se utilizan librerías que ya implementan las funcionalidades y los algoritmos necesarios.

2.3.4.1. TensorFlow

TensorFlow es una librería de código abierto para la computación numérica desarrollada por Google que proporciona una API de Python. En los últimos años también se han desarrollado APIs en otros lenguajes de programación como JavaScript. Proporciona funcionalidades de diferenciación automática de manera que genera grafos computacionales. Los nodos en estos grafos representan operaciones matemáticas, mientras que las uniones entre nodos representan los tensores (matrices multidimensionales) a los que afectan esas operaciones [18]. Está diseñada principalmente para su uso en redes neuronales, para lo cual optimiza las operaciones matriciales y su cálculo en GPUs (tarjetas gráficas). La característica principal de esta librería es su bajo nivel de abstracción. Con ella se puede crear la arquitectura de una red neuronal desde cero diseñando el grafo correspondiente. La Figura 2.11 muestra el logotipo de la librería TensorFlow.



Figura 2.11: Logotipo de TensorFlow

2.3.4.2. Keras

Keras es una librería de código abierto para Python inicialmente desarrollada como parte del proyecto ONEIROS y que corre sobre otras librerías y *frameworks* como TensorFlow, CNTK o Theano. Proporciona funcionalidades a nivel de capas para que la creación de redes neuronales artificiales sea una composición de capas [19]. Esta librería pierde parte de la flexibilidad que aporta TensorFlow al no poder predefinir las capas, lo que impide al usuario su creación personalizada. Esto se convierte en una ventaja cuando no se necesita de un diseño desde cero, ya que permite un desarrollo

mucho más rápido y sencillo. Por ello Keras se encuentra en un nivel de abstracción superior a TensorFlow. La Figura 2.12 muestra el logotipo de la librería Keras.



Figura 2.12: Logotipo de Keras

2.3.4.3. Scikit-learn

Scikit-learn es una librería de código abierto para Python. Proporciona modelos completos que pueden ser configurables mediante el ajuste de sus hiperparámetros. Scikit-learn por tanto se encuentra en un nivel superior de abstracción respecto a TensorFlow y Keras. De esta forma, aunque se pierde flexibilidad se aumenta la velocidad de desarrollo [20]. Proporciona algoritmos de regresión, clasificación, análisis, búsqueda de hiperparámetros, etc. Prácticamente la totalidad de los algoritmos relacionados con el *machine learning* están implementados en esta librería. Se creó como parte del proyecto SciPy (Python científico) junto con otras librerías (Numpy, Pandas, etc.) con las que está totalmente integrada. La Figura 2.13 muestra el logotipo de la librería Scikit-learn.



Figura 2.13: Logotipo de Scikit-learn

2.4. Dialogflow

Para la creación de interfaces conversacionales que se puedan integrar en diferentes dispositivos y sistemas existen una serie de plataformas disponibles con las que se pueden construir. Estas plataformas existen con el fin de acumular todo el conocimiento y las técnicas para la construcción de interfaces conversacionales, y así facilitar esta tarea. De todas las disponibles, Dialogflow [21] es una de las más conocidas, sobre todo después de su compra por parte de Google. En la Figura 2.14 se muestra el logotipo de Dialogflow.



Figura 2.14: Logotipo de Dialogflow

Dialogflow es una API para la creación de interfaces conversacionales basada en inteligencia artificial. Transforma la consulta a texto y este en datos procesables como un objeto JSON. Su principal característica es que hace coincidir de forma automática la consulta del usuario con la intención más adecuada, basándose en el modelo de aprendizaje automático del agente, lo que consigue una interacción muy natural con el usuario.

Cuando el usuario hace uso de este servicio, ya sea mediante el uso de la voz o mediante un mensaje de texto, esto llega a la plataforma que es la encargada de averiguar cuál es la intención del usuario. En función de la intención del usuario llama a un servicio u otro, para que le proporcione la respuesta que necesita. Cuando ya tiene la respuesta, se la manda de vuelta al usuario usando el medio que esté disponible [21]. En la Figura 2.15 se muestra un esquema del flujo que se sigue cuando se realiza una consulta a una interfaz conversacional desarrollada en Dialogflow.

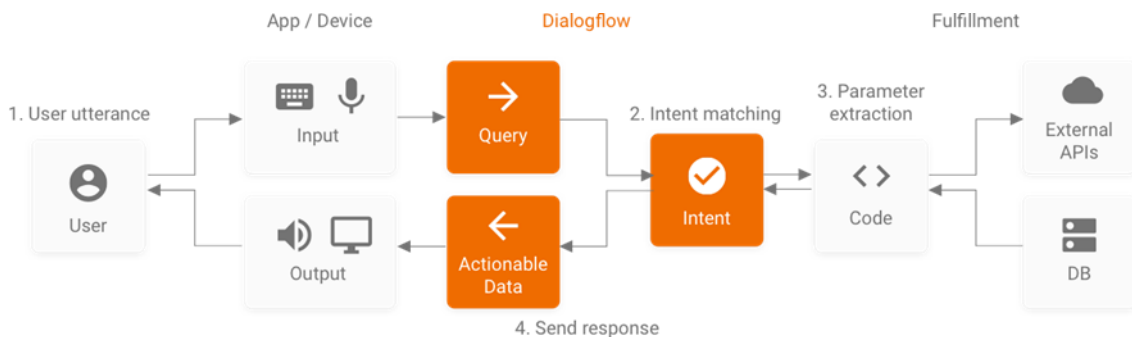


Figura 2.15: Diagrama de funcionamiento de Dialogflow [21]

Dialogflow se basa en una serie de conceptos que se exponen a continuación para hacer el reconocimiento de las intenciones del usuario. Estos conceptos son los siguientes:

- **Agentes (*agents*):** Son módulos a los cuales se les encarga el reconocimiento de ciertas intenciones comunes normalmente relacionadas.
- **Intenciones (*intents*):** Son las intenciones, lo que el usuario quiere. Definen las posibles preguntas o peticiones que nuestra interfaz conversacional puede ser capaz de responder.
- **Entidades (*entities*):** Son los parámetros que una intención puede contener. Debido a que una misma realidad puede denominarse de formas distintas, las entidades recogen estas formas y las interpretan todas como el mismo concepto.
- **Acciones (*actions*):** Es todo aquello que se ejecuta cuando se detecta una intención del usuario.
- **Contextos (*contexts*):** Permiten relacionar varias intenciones de forma que se pueda usar la información extraída en uno también en el resto, y

de esa forma hacer más fluida la conversación ya que no necesitas volver a indicar todo desde el principio.

- **Fulfillment:** Permite el uso de APIs externas para la recopilación de información, de forma que se puede responder de forma personalizada al usuario al recopilar la información que necesita de servicios externos.

Con estos conceptos del funcionamiento de la plataforma, es posible construir interfaces conversacionales muy completas. Además, las interfaces creadas con Dialogflow pueden integrarse en una gran variedad de servicios distintos como Google Assistant, Amazon Alexa o Microsoft Cortana entre otros.

2.5. Firebase

Firebase es un conjunto de herramientas y utilidades orientadas a la creación de aplicaciones en la nube. Las herramientas estas destinadas a la creación de aplicaciones tanto móviles (Android e iOS) como aplicaciones web [22]. En la Figura 2.16 se muestra el logotipo de Firebase.



Figura 2.16: Logotipo de Firebase

Algunas de las herramientas que presta Firebase son las siguientes:

- **Base de datos en tiempo real (*realtime database*):** Es una base de datos de tipo no relacional que se almacena en formato JSON en la nube.
- **Autenticación (*authentication*):** Este servicio simplifica el inicio de sesión, ya que permite que el usuario se registre usando su cuenta en otros servicios.
- **Almacenamiento (*storage*):** Permite almacenar archivos de cualquier clase, que no se admitirían en el JSON de la base de datos.
- **Alojamiento (*hosting*):** Sirve tu aplicación en red sin necesidad de montar un servidor propio, contando además con todos los estándares de seguridad.
- **Funciones (*functions*):** Son pequeñas funciones de código que se ejecutan cuando se hace una petición a la URL que genera cada función.

Además de los mencionados anteriormente, también tiene otras utilidades y herramientas como las analíticas y de monetización de la aplicación, pero no son de tanta relevancia en este trabajo.

2.6. Raspberry Pi

Raspberry Pi es un ordenador de placa única de bajo coste desarrollado por la Raspberry Pi Foundation en Reino Unido. Su objetivo fue, y es, el de fomentar la enseñanza de la informática en los colegios e institutos. Aunque la placa no es expresamente hardware libre, el software, que es una versión adaptada de Debian, sí que lo es.

Los primeros diseños estaban basados en el micro ATmega644 de Atmel. En el 2009 se crea la fundación Raspberry Pi y en febrero de 2012 fue lanzado el primer modelo, la Raspberry Pi 1 modelo A. En los primeros seis meses desde el lanzamiento de la placa se vendieron alrededor de medio millón de unidades. Actualmente existen 4 placas, cada una con diferentes variantes.

La Raspberry Pi 1, que cuenta con los modelos A, B y B+, fue la primera placa de la fundación. El primer modelo no contaba con puerto Ethernet, cosa que se solucionó en los modelos B. Disponía de 26 conectores GPIO, salida HDMI, conector Jack 3.5mm y un único puerto USB. Su procesador era un Broadcom BCM2835 a 700MHz y contaba con 256Mb de memoria RAM. Los modelos B doblaron la memoria RAM y aumentaron el número de puertos USB.

La Raspberry Pi 2 fue lanzada en 2014 y solo contó con modelo B. Sustituyó el procesador por el BMC2836 de cuatro núcleos a 900MHz. Cuenta con un 1Gb de memoria RAM y aumenta el número de pines GPIO a 40.

La Raspberry Pi 3 fue lanzada en 2016, inicialmente el modelo B. Posteriormente se presentaron los modelos B+ y A. Esta placa cuenta con un procesador de cuatro núcleos a 1,2GHz y mantiene la misma cantidad de memoria RAM de su antecesor. Esta placa ha sido la primera en incluir Wi-Fi y Bluetooth sin necesidad de adaptadores. El modelo B+ mejora el procesador y la conectividad, mientras que el modelo A es una versión reducida y de menor coste. En la Figura 2.17 se muestra una imagen de una Raspberry Pi Modelo B+ en la cual se ven todos los conectores que tiene disponibles esta placa.



Figura 2.17: Raspberry Pi 3 modelo B+

Además de estas placas, la fundación también ha creado las Raspberry Pi Zero. Estas placas son más pequeñas y menos potentes, lo que a su vez repercute en el coste, lo que también es menor. El primer modelo fue lanzado en 2015, con un coste de tan solo 5 dólares. Aun así, es un 40% más potente que la primera Raspberry que fue lanzada. Cuenta con un procesador BCM2835 de un núcleo a 1GHz y 512Mb de RAM. El modelo W de esta placa integra también conexión Wi-Fi y Bluetooth. [23]

2.7. Sistemas de teleasistencia

Los sistemas de teleasistencia son servicios preventivos de asistencia domiciliaria. Estos sistemas están pensados para personas mayores o con una discapacidad que le genere un cierto grado de dependencia. Tienen por objeto mejorar la calidad de vida de sus usuarios facilitando su independencia, al asegurar una intervención inmediata en caso de necesidad médica o personal, lo que contribuye a disminuir significativamente los ingresos en centros hospitalarios o residenciales, con el correspondiente ahorro económico que esto supone [24].

Los sistemas se pueden dividir en diferentes tipos en función a diferentes factores, como pueden ser el tipo de accionamiento, el tipo de respuesta o los servicios prestados, entre otros.

2.7.1. Componentes del sistema

Los sistemas de teleasistencia están generalmente formados por los mismos componentes y agentes. Los elementos considerados son tanto técnicos como humanos.

- **Gestor del servicio:** Empresa (pública o privada) que proporciona el servicio al usuario. Es la empresa que gestiona todo el sistema ya que cuanta con toda la información del usuario, y por tanto de sus necesidades.
- **Suministradores:** Empresas que suministran el equipamiento necesario a los gestores del servicio.
- **Proveedor de comunicaciones:** Son los operadores que proporcionan la conexión de línea fija o móvil.
- **Usuario:** Persona que recibe la prestación de los servicios de teleasistencia como elemento de apoyo a su independencia diaria.
- **Cliente del servicio:** Persona o entidad que paga el servicio. No tiene por qué coincidir con el usuario del sistema, ya que muchas veces este tipo de sistemas están subvencionados.
- **Terminal adaptada del servicio:** Elemento que se instala en la vivienda del usuario y que sirve para la comunicación del usuario con el gestor del servicio.
- **Pulsador de emergencia:** Dispositivo que debe llevar el usuario consigo en todo momento, que cuenta con un botón que al pulsarlo se lanza la solicitud de ayuda.

2.7.2. Modalidades según el tipo de accionamiento

- **Sistemas activos:** Es la modalidad más común y conocida. Necesita de una unidad de control remoto para su funcionamiento, con la que el usuario activa el terminal instalado en su vivienda, y a través del cual puede pedir ayuda. El control remoto puede ser un colgante o pulsera que integra un pulsador.
- **Sistemas pasivos:** La alarma se activa en función a la medición de distintos sensores instalados en la vivienda. El usuario no interviene en la activación del sistema.
- **Sistemas semiactivos:** El gestor del servicio se pone en contacto con el usuario en periodos regulares para comprobar su estado y el correcto funcionamiento del sistema.

2.7.3. Modalidades según el tipo de respuesta

- **Sin unidad móvil:** El servicio se presta exclusivamente a través del sistema instalado en la vivienda por el gestor de este. Se movilizan los recursos necesarios y se notifica a las personas de contacto.
- **Con unidad móvil:** El gestor del sistema cuenta con las llaves de la vivienda del usuario (con su consentimiento) de forma que pueden desplazar a una persona a la vivienda para realizar una asistencia más precisa y de mejor calidad.

2.7.4. Modalidades según el tipo de demanda de asistencia

- **Urgencias vitales:** El usuario requiere una atención inmediata (caídas, quemaduras, etc.).
- **Situaciones no críticas:** El usuario requiere de atención, pero la situación no es de carácter urgente (información, soledad, etc.).
- **Automatización y control:** Se monitorizan las actividades de la vida diaria del usuario haciendo uso de los sensores instalados en la vivienda.

2.8. La persona dependiente

El consejo europeo, tal como refleja el libro blanco sobre la dependencia elaborado por el gobierno de España [25], define la dependencia como “la necesidad de ayuda o asistencia importante para las actividades de la vida cotidiana”. En una definición más precisa lo indica como “un estado en el que se encuentran las personas que, por razones ligadas a la falta o la pérdida de autonomía física, psíquica o intelectual, tienen necesidad de asistencia y/o ayudas importantes a fin de realizar los actos corrientes de la vida diaria y, de modo particular, los referentes al cuidado personal” [25].

Por lo tanto, se deben identificar tres factores para poder hablar de una persona en situación de dependencia. El primer factor es que la persona en cuestión debe presentar alguna limitación tanto física como psíquica o intelectual. El segundo factor haría referencia a la incapacidad de la persona para realizar actividades cotidianas de manera autónoma. Por último, el tercer factor al que se hace referencia en la definición es la necesidad de algún tipo de asistencia por parte de un tercero.

Siguiendo el criterio establecido por la OMS en mayo de 2001, se puede seguir la siguiente clasificación para interpretar las alteraciones que acaban provocando la situación de dependencia en la persona:

- **Déficit en el funcionamiento:** Pérdida de una o varias funciones fisiológicas o psíquicas. También se incluyen la pérdida o malformación de alguna parte del cuerpo.
- **Limitación de la actividad:** Dificultades en la ejecución de actividades que pueda presentar la persona. Estas limitaciones, debido a gran cantidad de variaciones, suelen clasificarse en varios grados.
- **Restricción de la participación:** Problemas de participación en situaciones vitales, comparados con la participación esperada de una persona sin discapacidad.
- **Elementos barrera:** Factores del entorno que condicionan su autonomía y crean dependencia.
- **Discapacidad:** Término genérico usado para referirse a déficits, límites en la actividad o restricciones de la participación.

Siguiendo la línea de acontecimientos marcada anteriormente, una situación de dependencia es el resultado de un proceso que comienza con la aparición de un déficit como consecuencia de un accidente o enfermedad. Este déficit, en mayor o menor medida, lleva asociada una limitación de la actividad. Estas limitaciones en ocasiones pueden subsanarse con la adaptación del entorno, pero cuando esto no es posible, se produce una restricción de la participación. Este proceso acaba por concretarse en una situación de dependencia.

Las situaciones de dependencia tienen una estrecha relación con la edad, pues las personas de avanzada edad son el grupo de población que presentan mayores limitaciones. Dentro del grupo de población de la tercera edad, los individuos con dependencia no están distribuidos de forma uniforme. A partir de los 80 años, aproximadamente, la cantidad de individuos con dependencia aumenta de forma muy considerable.

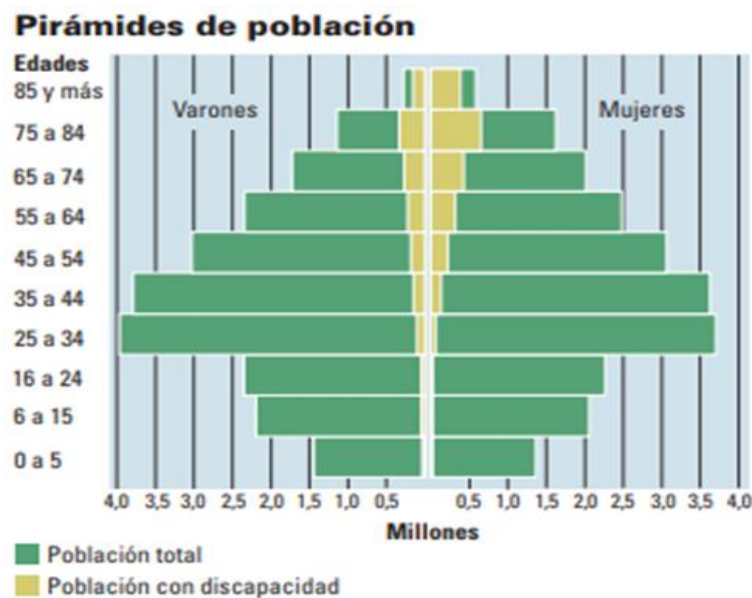
Por todo esto, es que la dependencia es un problema asociado al envejecimiento demográfico de la población. No obstante, la dependencia se puede dar en individuos de cualquier edad, desde el nacimiento hasta el fallecimiento. Dado que las limitaciones pueden deberse a una enfermedad o un accidente, este problema puede surgir a cualquier edad. La dependencia recorre toda la estructura de la población.

La estrecha relación entre la dependencia y la edad no significa necesariamente que la vejez traiga asociada una situación de dependencia. En ocasiones la dependencia no llega a aparecer. Esto se debe tanto a factores ambientales como genéticos, además de distintos hábitos de vida. Esto quiere decir que se puede prevenir la dependencia [26].

2.8.1. Determinantes demográficos de la dependencia

Como se comentaba anteriormente, la dependencia tiene una estrecha relación con la edad. Debido al envejecimiento de la población, la necesidad de atención a situaciones de dependencia ha aumentado de forma notable en los últimos años y lo va a seguir haciendo durante las próximas décadas.

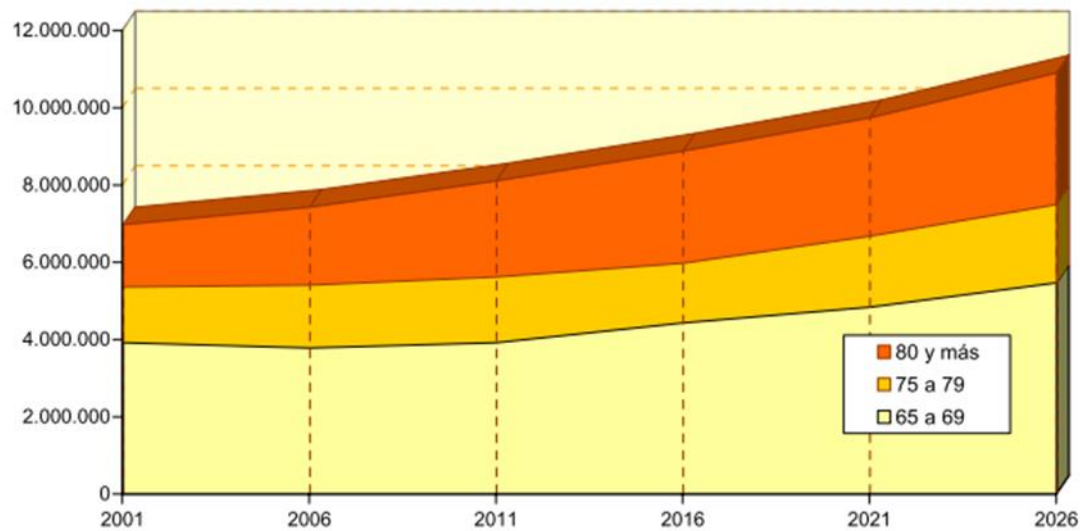
Teniendo como referencia a la población española, en los últimos años ha experimentado un crecimiento considerable. El número de individuos mayores de 65 años se ha duplicado en tan solo treinta años (1970-2000) llegando a ser casi un 17 por ciento del total de la población. Se espera además que en los próximos años siga aumentando [27]. En la Figura 2.18 se muestra la pirámide poblacional de España frente a la población con discapacidad en 2008. En la Figura 2.19 se muestra estimación de evolución de la población de España desde 2001 hasta 2026, realizando una separación por franjas de edad.



Fuente estadística utilizada: Encuesta de Discapacidad, Autonomía personal y situaciones de Dependencia (EDAD-2008), procedente del INE

Figura 2.18: Pirámide de población española con discapacidad frente al total [27]

Evolución proyectada de la población mayor española por tramos de edad, 2001-2026



Fuente: INE, Proyecciones de población calculadas a partir del Censo de 2001

Figura 2.19: Proyección de la evolución de la población de la tercera edad en España [25]

En conjunto, existe un mayor número de mujeres dependientes que de hombres en los grupos de población de avanzada edad. Sin embargo, en personas más jóvenes, el número de varones dependientes es ligeramente superior.

Como se puede observar la previsión es al alza, lo que conllevará una presión extra sobre el sistema de provisión de cuidados a personas dependientes. Se convierte por tanto en un problema fundamental para todas las sociedades modernas, que por lo general presentan siempre el fenómeno del envejecimiento.

Capítulo 3

DESCRIPCIÓN GENERAL DEL SISTEMA

En el presente capítulo se va a dar una visión general del sistema desarrollado durante el Trabajo Fin de Grado. En el capítulo no se entrará a explicar cómo se han realizado y cómo funcionan las diferentes partes del sistema de forma técnica, ya que ello se realizará en el siguiente capítulo. Este capítulo está destinado a exponer la solución desarrollada y explicar por qué se han elegido esta y no otra. Además, se dará una visión general de la interconexión de las diferentes partes del proyecto.

Se presentará el sistema desarrollado, explicando las partes que lo forman, así como sus características más relevantes. Se explicará cómo es la estructura del sistema y la comunicación entre las partes. Se busca con ello dar a entender el funcionamiento y la elección de la solución. Además, se realiza un análisis de los recursos que se han utilizado y son necesarios para realizar este proyecto.

3.1. Presentación del sistema

El sistema desarrollado en el presente Trabajo Fin de Grado está destinado a poder ofrecer a personas en situación de dependencia un sistema que de forma automática pueda solicitar la ayuda que estas personas necesiten. Como se comentaba en capítulos anteriores, las pirámides de población de los países desarrollados se están invirtiendo y esto hace que cada vez existan más personas en situación de dependencia viviendo en su casa. Por ello este sistema busca aportar una solución a este problema, al automatizar procesos para el control y la asistencia a estas personas.

El sistema está compuesto por tres partes principalmente: 1) Una interfaz conversacional que interactúa con el usuario a través de un altavoz inteligente Google Home. 2) Una serie de sensores que miden parámetros del entorno en el que se encuentra el usuario con la intención de poder dar alertas en caso de que las mediciones

sean anómalas. 3) Una base de datos en la nube con actualización en tiempo real. En la Figura 3.1 se muestran las distintas partes que componen el sistema y las tecnologías que se han empleado, así como la interacción entre las partes.

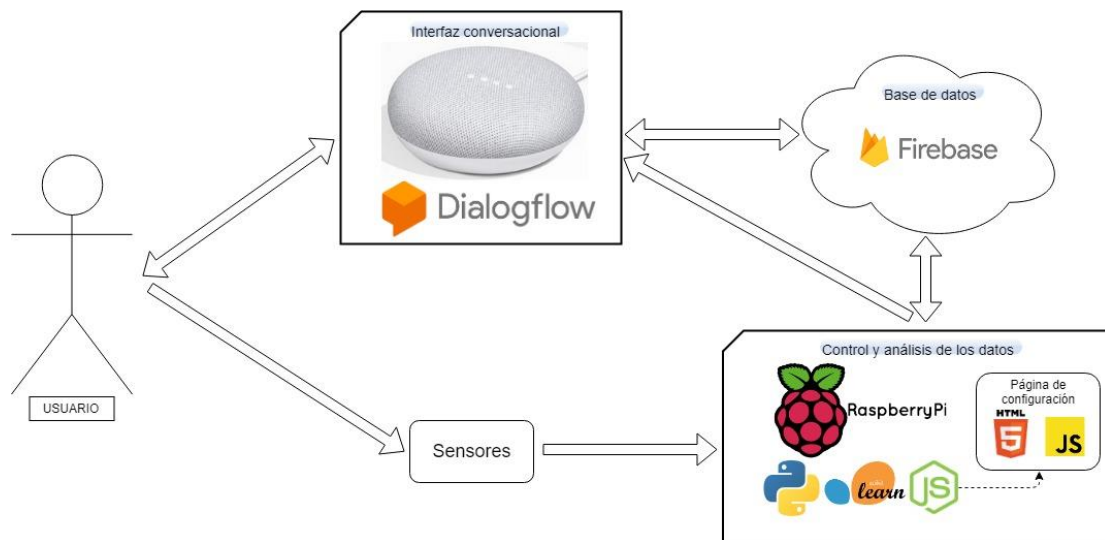


Figura 3.1: Esquema de interacción entre las partes del sistema

El usuario puede interaccionar con la interfaz conversacional, por lo que puede preguntarte o solicitar ayuda solo con el uso de la voz. De esta forma, se consigue que el usuario no dependa de un dispositivo externo (habitualmente un botón, ver epígrafe 2.7) para poder solicitar ayuda ya que, únicamente invocando a la interfaz conversacional mediante el uso de la voz, esta atenderá sus peticiones. De esta forma se elimina el riesgo de que el usuario no porte consigo el dispositivo, y por tanto no pueda solicitar asistencia. Se emplea la voz ya que es la forma más natural de comunicarse para los seres humanos. Incluso una persona que no sepa leer o escribir sí que sabe comunicarse usando su voz.

No siempre ocurre que el usuario esté capacitado para solicitar asistencia, ya que pueden darse casos de desmayos o situaciones de inconsciencia. Por ello, el sistema cuenta con una serie de sensores que miden diferentes parámetros del entorno domiciliario en el que se encuentra el usuario. Para tratar de detectar estas situaciones en las que el usuario necesita de la intervención de una tercera persona, pero no es capaz de solicitarla, se analizan los datos recogidos por los sensores. Estos datos son analizados con algoritmos de aprendizaje automático en lugar del uso de reglas fijas, de forma que se puedan detectar anomalías correctamente en función del patrón de comportamiento del usuario.

Todos estos datos son almacenados en la nube, de forma que son accesibles desde cualquier parte. Además, la base de datos usada proporciona funcionalidades para notificar cambios en los datos en el mismo momento en que se producen estos cambios.

3.2. Elementos del sistema

Como se comentaba en el epígrafe anterior, el sistema está compuesto por distintas partes. Cada una de estas partes están a su vez compuestas por otros elementos, cuyo funcionamiento se expondrá a continuación.

Además de las funcionalidades de las que están compuestas las distintas partes, se expondrán también los recursos y tecnologías utilizadas para el desarrollo de estas.

3.2.1. Interfaz conversacional

La interfaz conversacional es una de las partes fundamentales del sistema, ya que es la forma en la que el usuario interacciona con el sistema. Esta interfaz se comunica mediante el uso de voz con el usuario, y de esta forma atiende sus peticiones o proporciona alertas.

Como se comentaba anteriormente, el uso de la voz es una de las cualidades diferenciadoras frente a otros sistemas que también proporcionan asistencia a personas dependientes. Es diferenciador ya que todo el mundo, sea cual sea su nivel de estudios, está capacitado para comunicarse mediante el uso de la voz. Esto hace que sea la voz un medio idóneo para poder solicitar asistencia sin la necesidad de un dispositivo que la persona tenga que llevar consigo constantemente.

Solo usando su voz el usuario puede consultar el estado del sistema o solicitar aquello que necesite, sin importar la estancia de la vivienda en la que se encuentre, ya que tanto el micrófono como el altavoz son de gran alcance. En caso de ser una vivienda muy grande sería necesario el uso de más de un altavoz inteligente, que trabajarían de forma coordinada.

La principal cualidad que se ha tratado de implementar es que sea una interfaz conversacional proactiva. De esta forma, el usuario no solo obtiene respuestas a las preguntas o peticiones que realice, sino que, además, el sistema emplea el uso de la voz para alertar al usuario de posibles peligros de su entorno cuando estos se producen (reconocidos por los sensores). En la Figura 3.2 se muestra una imagen de un altavoz inteligente Google Home Mini con un ejemplo de conversación proactiva motivada por la medición de un sensor.



Figura 3.2: Ejemplo ilustrativo del funcionamiento de la interfaz conversacional proactiva

Para recoger las solicitudes del usuario por medio de la voz se ha utilizado un Google Home Mini, pero la funcionalidad puede ser implementada en cualquier altavoz inteligente del mismo fabricante. Se ha elegido este altavoz, entre otros motivos por su reducido coste, lo que permite que pueda ser adquirido por cualquier persona.

Por otra parte, al ser el fabricante del altavoz también el propietario de la tecnología utilizada para el desarrollo e implementación de la interfaz conversacional, se asegura una mejor compatibilidad.

Para desarrollar la interfaz conversacional se utilizó Dialogflow (ver epígrafe 2.4). Esta es una herramienta disponible online para la creación de interfaces conversacionales (también conocidas como chatbots). Esta interfaz se diseña y posteriormente se implementa como una aplicación adicional en el altavoz inteligente.

De esta forma se consigue que el altavoz mantenga toda la funcionalidad que trae de fábrica y, además, añadir las funcionalidades del sistema desarrollado en este proyecto. Así, se combina con todas utilidades que posee Google Assistant y se consigue una solución más completa.

Esta interfaz se publica de una forma similar a la publicación de aplicaciones móviles y puede ser instalada por el usuario de forma sencilla.

3.2.2. Control y análisis de los datos

Como se comentaba, existen momentos en los que el usuario puede no ser capaz de solicitar asistencia por sus propios medios. Por ello, se ha implementado un sistema que recoge mediante sensores algunos de los parámetros del entorno domiciliario en el que se encuentra el usuario.

Estos datos son usados para lanzar avisos que puedan alertar al usuario de peligros que hayan surgido en su entorno. Estos avisos, son lanzados mediante el altavoz inteligente como parte de la interfaz conversacional.

Para obtener los datos se utilizan una serie de sensores conectados a una placa Raspberry Pi 3. Se ha elegido esta placa no solo para realizar la lectura de los sensores, sino también para ser el cerebro del sistema. A diferencia de otras placas de propósitos parecidos como pueda ser Arduino, la Raspberry Pi monta un sistema Linux, lo que no restringe la programación a un único lenguaje y tiene suficiente potencia como para ejecutar los programas necesarios.

La Figura 3.3 muestra de forma gráfica el flujo de datos desde que son medidos por el sensor hasta que, en caso necesario, se lance una alerta. Como se puede observar, en la Raspberry Pi conviven dos partes en diferentes lenguajes. La parte de Python es la encargada de la lectura de los sensores y del análisis de estas mediciones mediante un algoritmo de aprendizaje automático, mientras que la parte de Node.js levanta un servidor en las Raspberry Pi desde el cual se lanzaran las alertas y se servirá la página de configuración.

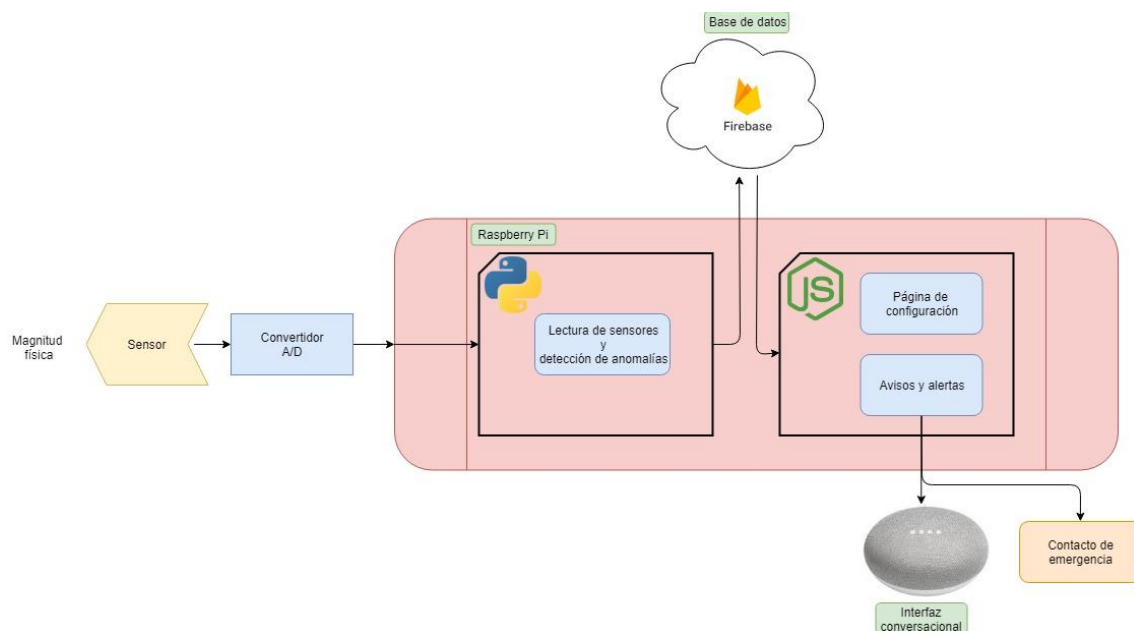


Figura 3.3: Diagrama del flujo de los datos obtenido de los sensores

Los datos son recogidos y analizados en función del tipo de sensor. Para un sensor de tipo detector de llamas, cuando se active se procede a avisar directamente al usuario, ya que se debe comprobar la veracidad de la alerta inmediatamente. Para sensores de presencia o similares, se va a utilizar un algoritmo de aprendizaje

automático. Esto elimina la rigidez de establecer unas reglas fijas, y permite al sistema adaptarse a los parámetros del entorno del usuario y a los patrones de comportamiento de este. Además, permite reconocer datos anómalos, aunque los parámetros del entorno cambien con el tiempo (como podría ser la temperatura en función de la época del año). De igual manera puede adaptarse a los cambios en la rutina del usuario.

El objetivo de este análisis es poder detectar cuando el usuario se sale de su rutina, ya que esto podría significar la existencia de algún tipo de problema.

3.2.2.1. Obtención de los datos

Los datos del entorno en el que se encuentra ubicado el sistema se obtienen mediante el uso de sensores. Estos sensores están conectados a la Raspberry Pi mediante los pines GPIO de esta. En este proyecto se implementan una serie de sensores que se expondrán con detalle en el siguiente capítulo. No obstante, y tal y como se verá en el último capítulo de conclusiones y trabajo futuro, el sistema está diseñado de tal forma que añadir nuevos tipos de sensores sea algo sencillo.

Los pines GPIO de la Raspberry Pi son entradas y salidas digitales. Esto quiere decir, que es necesario convertir la señal analógica de los sensores en una señal digital. Por esto, en aquellos sensores que no cuenten con el acondicionamiento adecuado será necesario acondicionar la señal analógica y convertirla a digital antes de conectarlos a un pin de la placa.

La Raspberry Pi ejecuta un *script* de Python el cual es el encargado de realizar la lectura de los sensores. Python es un lenguaje de programación interpretado que está orientado a objetos. Es de alto nivel y su sintaxis es muy fácilmente legible, lo que facilita su depuración y lo dota de una curva de aprendizaje suave. Además, es el lenguaje más utilizado para la implementación de algoritmos de aprendizaje automático, lo que facilita encontrar librerías y recursos.

Este *script* es el encargado de comprobar las lecturas en los pines donde hay sensores conectados de forma periódica. Para que el sistema sea lo más modular posible, este *script* descarga de la base de datos la configuración de los sensores. De esta forma, si se quiere realizar algún cambio en la configuración solo se debe acudir a la página de configuración (que se explicará en epígrafes posteriores) y realizar los cambios.

Todos los datos que se van recogiendo se procesan para analizar si suponen una anomalía en el patrón que siguen esos datos (como se explica en el siguiente epígrafe) y se suben de manera inmediata a la base de datos, de forma que el resto del sistema pueda consultarlos.

3.2.2.2. Algoritmo de aprendizaje automático

Los datos recogidos del entorno son analizados para comprobar si la nueva medición obtenida sigue el patrón del resto de datos recogidos o si por el contrario representa una anomalía.

Una anomalía en el patrón puede significar la existencia de una situación en la que la persona requiera algún tipo de asistencia. Así, por ejemplo, si la persona entra todos los días en la cocina a las nueve de la mañana (siendo esto recogido por sensores de presencia), el hecho de que un día no entrase en una hora más o menos próxima significa que ha ocurrido algo que ha alterado su rutina, y por lo menos se debe comprobar que es lo que ha producido el cambio. De esta forma, se obtiene un preaviso que puede prevenir una situación de una gravedad mayor.

Los patrones no son reglas de comportamiento que permanezcan inmutables con el paso del tiempo. Es por esto por lo que es necesario ir adaptando el modelo de decisión. El uso de unas reglas fijas daría como resultado un modelo demasiado rígido que sería incapaz de adaptarse a los cambios y que proporcionaría avisos falsos o pasaría por alto avisos reales.

Para evitar esta rigidez, se ha optado por usar un algoritmo de aprendizaje automático, el cual aprenda el patrón de los datos obtenidos previamente y sea capaz de discernir si la nueva medición realizada se puede considerar una anomalía o no. De esta forma, al utilizarse un modelo entrenado con los datos previos el patrón va cambiando con los datos, y por ello se obtiene un mejor resultado.

Dado que la detección de anomalías es un problema en el que se tienen pocos ejemplos de lo que es una anomalía, se debe seleccionar uno que sea capaz de separar los datos que están agrupados según un patrón, los que se considerarían datos normales, de los que están aislados y no siguen el patrón, que son considerados anomalías.

De todos los algoritmos existentes en el estado del arte en el momento de la realización de este proyecto, se ha elegido el algoritmo de Isolation Forest, cuyo funcionamiento ya ha sido explicado en el capítulo anterior (ver epígrafe 2.3.3). Da buenos resultados, aunque el número de características disponibles sea bajo, lo que lo convierte en idóneo para este sistema.

Dentro del mismo *script* de Python utilizado para la obtención de los datos se realiza también el análisis de estos mediante el algoritmo de aprendizaje automático mencionado anteriormente. Los datos del histórico son descargados de la base de datos y se utilizan para entrenar el algoritmo. Para contemplar la estacionalidad de los patrones, el algoritmo se reentrena todas las noches con los datos de los últimos días. De esta forma, si los patrones cambiasen en un tiempo razonable, se entiende que es un cambio de hábitos y no un comportamiento anómalo.

Para implementar el algoritmo de Isolation Forest en Python se ha utilizado la librería Scikit-learn (ver epígrafe 2.3.4.3).

3.2.2.3. Alertas y avisos

Una vez que se han recogido los datos, se han analizado y etiquetado en función de si se consideran o no datos anómalos, es necesario generar avisos en caso de que una medición de un sensor haya sido etiquetada como anómala.

Para llevar a cabo esta funcionalidad, se ha implementado un servidor en la Raspberry Pi mediante Node.js. Implementar un servidor es necesario ya que se necesita establecer comunicación con el Google Home para poder mandar los mensajes de alerta a través de él, y de esta forma contar con una interfaz conversacional proactiva que es uno de los objetivos del proyecto. Además de los avisos que se envían a través del altavoz inteligente, también se ha implementado una funcionalidad para avisar a un contacto de emergencia de estas alertas. En la página de configuración se puede añadir un correo electrónico de la persona que se desea notificar, de manera que recibirá un correo electrónico cada vez que se reconozca un dato anómalo, o cuando el usuario solicite asistencia.

Para el servidor se ha elegido Node.js ya que permite utilizar JavaScript como lenguaje de programación. JavaScript es un lenguaje de programación interpretado y orientado a objetos. Este lenguaje es el más usado en su forma del lado del cliente, por lo que es sencillo encontrar recursos y soporte.

Los avisos enviados a través del Google Home como parte de la interfaz conversacional no se realizan de una forma “nativa”. Con el termino nativo hago referencia al hecho de que Dialogflow no implementa la posibilidad de generar una salida de voz en la interfaz conversacional si previamente no ha sido invocada por el usuario. Por ello se emplea la funcionalidad de *broadcasting* (difusión en castellano) que permite distribuir un mensaje seleccionado por el usuario en todos los altavoces inteligentes conectados en la misma red. Para esta difusión se utiliza una librería creada por la comunidad a tal efecto ante esta carencia. Como punto negativo que se ha encontrado, se puede destacar el hecho de que este mensaje no espera respuesta, y por tanto estos avisos no pueden (no deberían) realizar una pregunta directa al usuario, ya que, aunque este la contestase, el altavoz no estaría escuchando esa respuesta. En el último capítulo de conclusiones y trabajo futuro se indican posibles mejoras a esta carencia.

3.2.3. Base de datos

Los datos en este proyecto son una parte muy importante. Uno de los objetivos es el reconocimiento de patrones para poder detectar anomalías y dar así avisos. Como se ha ido desarrollando a lo largo de este capítulo, para implementar esta funcionalidad se utiliza un algoritmo de aprendizaje automático, el cual necesita de datos para poder aprender.

Estos datos son almacenados en Firebase, la cual es una base de datos no relacional (NoSQL). Es una base de datos alojada en la nube y los datos son

almacenados en formato JSON, sincronizándose en tiempo real con cada cliente conectado. Al tener sincronización en tiempo real, es idónea para implementar la funcionalidad de alertas y avisos, ya que cuando se añada un nuevo dato, automáticamente será leído por el servidor implantado en la Raspberry Pi (ver epígrafe anterior). Además, el formato JSON es fácilmente manejable en JavaScript, que es el lenguaje que se emplea en el servidor utilizado.

Firebase cuenta además con la posibilidad de añadir funciones, que se ejecutan en la nube. Esto es muy útil, ya que algunas funciones como puede ser la consulta de algunos datos o funcionalidades de la interfaz conversacional se pueden implementar sin necesidad de utilizar la Raspberry Pi, lo que dota de cierta modularidad extra al sistema. Las funciones implementadas directamente en Firebase son utilizadas para realizar una consulta a la base de datos cuando es requerido por alguna intención de la interfaz conversacional.

Tener la base de datos implementada en la nube permite tener los datos siempre disponibles desde cualquier dispositivo con conexión a internet, además de evitar su pérdida en caso de error o fallo si estuviera alojada en un dispositivo local. Cuanta con planes de pago, aunque para este proyecto solo han sido necesarios los gratuitos, debido a que para las pruebas el volumen de datos y de tráfico no ha sido muy elevado. En ambas modalidades se cuenta con un servicio de análisis del funcionamiento y del volumen de tráfico que ayuda al dimensionamiento de la base de datos necesaria.

El tráfico de datos podría llegar a ser elevado en ciertas implementaciones del sistema, ya que, para asegurar que todos los elementos del sistema trabajan con los mismos datos, todos ellos los descargan de la base de datos independientemente de donde se hayan generado. Esto significa, que por ejemplo en la Raspberry Pi, donde conviven la parte de la obtención de los datos y su análisis y la parte de las alertas y avisos, estas dos partes no comparten los datos directamente. En su lugar, el *script* de Python los sube a la base de datos en la nube y el servidor de Node.js los descarga desde la base de datos. Este funcionamiento, que aumenta el tráfico de datos en Firebase, permite asegurar que no se darán problemas de retrasos que pudieran hacer que distintas partes trabajen con datos distintos.

3.2.4. Página de configuración

Uno de los objetivos de este Trabajo Fin de Grado es el de crear un sistema lo más modular posible. Para ello, se debe poder añadir o quitar sensores de forma que cada usuario decida cuales son los que desea implementar. Para cumplir con esto, se ha desarrollado una página de configuración donde el usuario puede cambiar la configuración del sistema, así como ver su estado actual.

En todas las partes del sistema se tiene en cuenta la posibilidad de añadir o quitar sensores del sistema, así como cambiar el correo electrónico de la persona de contacto, y otras funcionalidades. Por ello, todos los aspectos están parametrizados

para funcionar según de unos datos de configuración que se establecen y se suben a la base de datos desde esta página.

Esta página de configuración se aloja en el servidor Node.js que se ejecuta en la Raspberry Pi, por tanto, se puede acceder a ella desde cualquier dispositivo conectado a la misma red. Se han contemplado una serie de sensores y ubicaciones posibles, por lo que solo se podrá elegir entre estos, aunque se pueden combinar como se desee. Añadir más tipos de sensores y ubicaciones de estos necesitaría de la modificación de algunas partes del código, sin embargo, dado que esta parametrizado como se comentaba anteriormente, estos cambios serían mínimos.

La página muestra los sensores conectados (si los hubiera) junto con su última medición. Los datos se muestran agrupados por tipo. Además, se puede añadir o cambiar el correo electrónico al que se notificarán los avisos. Cuando se añade un nuevo sensor, se ha escoger la ubicación en la que se instalará, tipo de sensor que es para poder realizar una correcta lectura y el pin GPIO de Raspberry Pi al que irá conectado. La Figura 3.4 muestra una captura de pantalla de la página web creada para configurar el sistema.

Dependents Assistant

Email de notificaciones

ejemplo@ejemplo.com

Guardar

Añadir sensor

Temperatura		
Habitación	23 °C	
Salón	20 °C	

Humedad		
Salón	21%	

Presencia		
Parece que todavía no hay sensores de presencia.		

Fuego		
Salón	0	

Figura 3.4: Página de configuración

La página web se ha desarrollado con HTML5 y JavaScript, que son las tecnologías más usadas para el desarrollo de webs. Además, se ha utilizado el framework Bootstrap para darle una mejor apariencia y estilo.

Capítulo 4

DESCRIPCIÓN DETALLADA DEL SISTEMA

En el presente capítulo se va a dar una descripción detallada del sistema desarrollado durante el Trabajo Fin de Grado. Se entrará a explicar cómo se han realizado y cómo funcionan las diferentes partes del sistema de forma más técnica. Se explicarán todos los detalles del trabajo realizado y se expondrán todas las piezas que componen el proyecto. La mayor parte del código y elementos mostrados en las figuras de este capítulo están en inglés, ya sea porque la herramienta utilizada solo está disponible en ese idioma o por que se ha decidido utilizarlo para evitar mezclar idiomas, como pasa sobre todo en el caso del código.

4.1. Desarrollo de la interfaz conversacional

Como se explicaba en capítulos anteriores, una de las partes más importantes del sistema desarrollado es la interfaz conversacional. Esta permite al usuario comunicarse con el sistema y también recibir avisos de este. Esta interfaz conversacional se ha desarrollado con la herramienta Dialogflow, disponible online. Como se describía anteriormente (ver epígrafe 2.4) esta herramienta está estructurada principalmente mediante la figura de los *intents*. Con ellos se puede reconocer las intenciones del usuario. A lo largo de este capítulo se utilizará intenciones, el termino en castellano, para referirse a esta figura.

Además, otra figura importante son las *entities*, que permiten reconocer conceptos dentro de las intenciones del usuario. A lo largo de este capítulo se utilizará entidades, el termino en castellano, para referirse a esta figura. Con estos dos elementos clave pueden desarrollarse la mayoría de las funcionalidades de la interfaz conversacional. A continuación, se detalla el trabajo realizado con esta herramienta para la creación de la interfaz conversacional.

4.1.1. Creación de un agente

El primer paso para la creación de la interfaz conversacional es la creación del agente. Un agente es la forma en la que se organizan en Dialogflow las interfaces conversacionales. De esta forma, se puede crear un agente para temas de meteorología, otro para asuntos turísticos o para asistencia, como es el caso de este proyecto.

Se ha de definir el nombre que se le va a dar, que no tiene por qué ser el mismo que luego se usará para la invocación de la interfaz conversacional. Se define también el idioma con el que se va a trabajar, en este caso el español, y la zona horaria de referencia. Por último, hay que vincular el nuevo agente a un proyecto de Google en la nube (*Google Cloud*). Si no existe se puede crear uno nuevo de forma automática. Con esto ya se ha creado el agente. Ahora es necesario realizar algunas configuraciones. La Figura 4.1 muestra una captura de pantalla de algunos de los ajustes más relevantes a la hora de crear un agente.

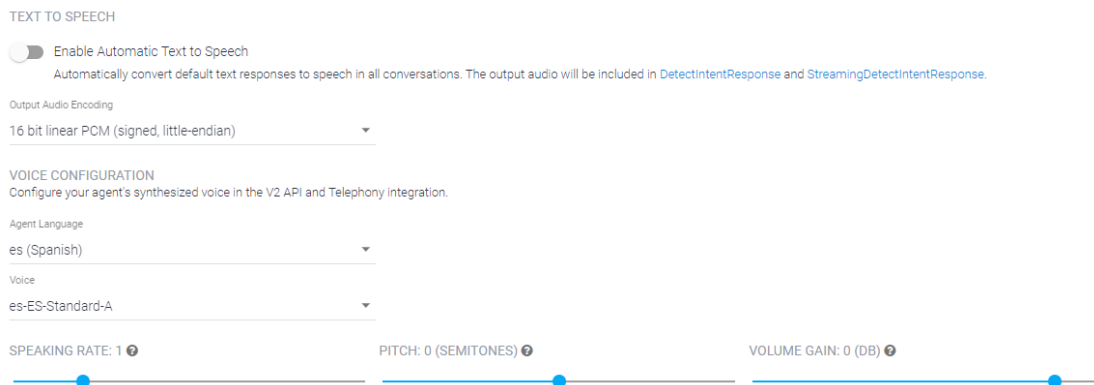


Figura 4.1: Ajustes del agente creado para la interfaz conversacional

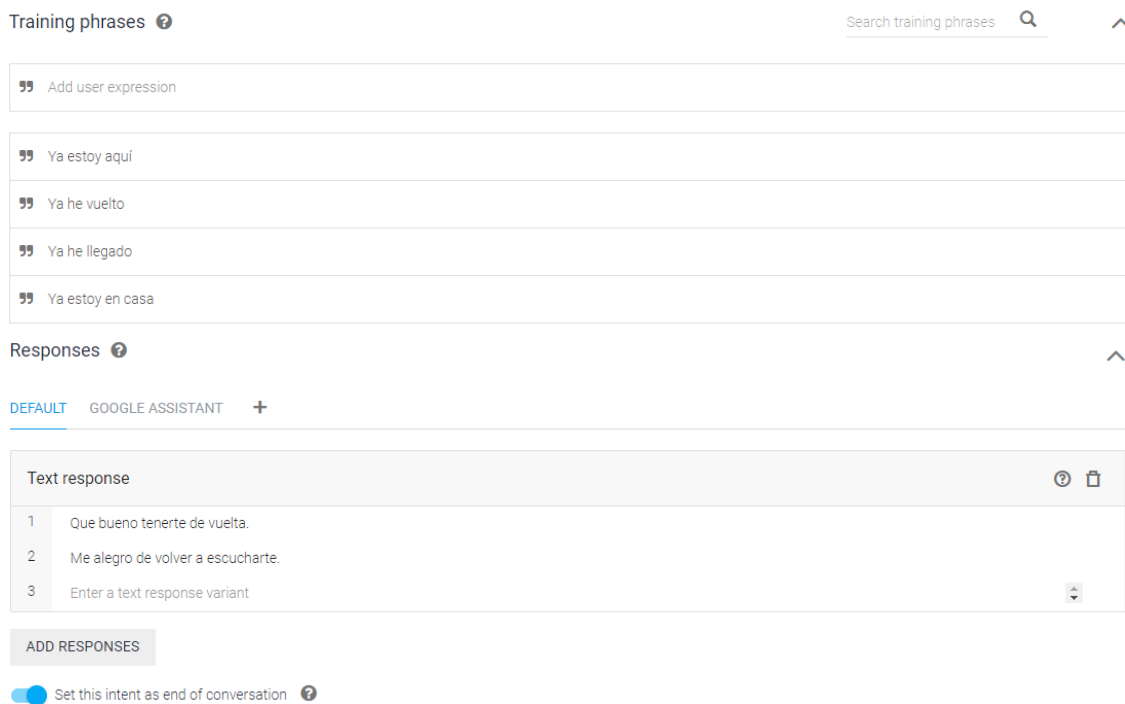
Se debe comprobar que los ajustes relativos a la interacción por voz son los adecuados y ajustándolos en caso necesario. Se está diseñando una interfaz conversacional que utiliza la voz para la interacción por lo que es necesario dedicar un tiempo al correcto ajuste de estos parámetros. Además, se comprueba también la versión de la API que se está usando, ya que la versión que se configura por defecto dejara de tener soporte a lo largo del año 2019. Con todo esto comprobado, el agente ya está listo para comenzar.

4.1.2. Creación de intenciones

Como se explicaba, las intenciones (*intents*) son la manera que tiene Dialogflow de reconocer cual es la intención del usuario. Lo primero que se ha de definir en la creación de una intención, después del nombre por el que lo vas a identificar, son las frases de entrenamiento. Con estas frases se define el fin que tendrá esa intención. Una de las ventajas de Dialogflow es que hace uso de la inteligencia artificial en el reconocimiento de las intenciones del usuario, de forma que, aunque su petición no concuerde al 100% con una de las frases de entrenamiento dadas, es capaz de

reconocer si se tiene la misma intención y por tanto ejecutar la intención correcta. Cuantas más frases de entrenamiento se introduzcan más fácil le será a la interfaz conversacional identificar correctamente las intenciones.

Una vez que ya se han definido las frases que servirán a la interfaz conversacional para aprender a reconocer la intención del usuario para llamar a una intención en concreto, se han de definir las respuestas que este dará. Si se está construyendo una intención sencilla la respuesta no necesitará recabar información de ningún sitio y será directa, por lo que podrá definirse directamente dentro de la interfaz conversacional. La Figura 4.2 muestra una captura de pantalla de una de las intenciones creadas a modo de ejemplo para ilustrar el proceso de creación de intenciones en Dialogflow.



The screenshot shows the Dialogflow console interface. At the top, there's a 'Training phrases' section with a search bar and a list of phrases: 'Ya estoy aquí', 'Ya he vuelto', 'Ya he llegado', and 'Ya estoy en casa'. Below this is the 'Responses' section, which is currently set to 'DEFAULT'. It shows a 'Text response' list with three items: 'Que bueno tenerte de vuelta.', 'Me alegro de volver a escucharte.', and 'Enter a text response variant'. There is an 'ADD RESPONSES' button and a toggle switch for 'Set this intent as end of conversation' which is currently turned on.

Figura 4.2: Ejemplo de frases de entrenamiento y respuestas de una intención

En este caso, no es necesario que la conversación se extienda más por lo que se marca para que esta intención sea el final de la conversación. No obstante, es necesario guardar esta información de que el usuario ya está en su casa, por lo que se utiliza un *fulfillment*. Esto es una conexión que se establece entre la base de datos y la interfaz conversacional, que será detallada en epígrafes posteriores.

En una intención también se puede definir el contexto en el que se puede enmarcar, los eventos que pueden activarlo, los parámetros que se deben de reconocer dentro de las frases de activación o las distintas configuraciones de uso del *fulfillment*. Estos a pesar de encontrarse dentro de la creación y configuración de una intención se explicarán en epígrafes separados para dar una descripción más detallada del proceso.

Las intenciones disponibles en la interfaz conversacional son las que se muestran en la Tabla 4.1. No obstante, puede hacerse uso de todas las funcionalidades

de las que dispone Google Assistant al estar la interfaz conversacional integrada con este asistente.

Intenciones	Uso
Bienvenido	Saluda al usuario cada vez que invoca la interfaz conversacional.
Error	Indica al usuario que no entendió su petición y pide que la repita de nuevo.
Llegar a casa	El usuario indica que ya está en casa y la interfaz conversacional lo anota en la base de datos.
Salir de casa	El usuario indica que sale de casa y la interfaz conversacional lo anota en la base de datos.
Emergencia	El usuario notifica que tiene algún tipo de problema y necesita asistencia.
Tipo de emergencia	El usuario puede especificar qué es lo que necesita. En caso de que no lo haga en primera instancia se le redirige a esta intención.
Comprobar temperatura	El usuario puede preguntar por la medición de temperatura del sensor que desee o por la media de todos los disponibles.
Comprobar humedad	El usuario puede preguntar por la medición de humedad del sensor que desee o por la media de todos los disponibles.

Tabla 4.1: Intenciones disponibles en la interfaz conversacional

4.1.3. Creación de Entidades

Las entidades (*entities*) permiten en Dialogflow reconocer conceptos. Esto significa que, por ejemplo, el concepto ciudad puede tener muchas palabras o nombres propios asociados que hagan referencia al concepto de ciudad. Por ello, si necesitamos que nuestro usuario nos indica la ciudad en la que se encuentra, debemos definir previamente que entendemos por ciudad. De esta forma, no será necesario introducir una gran cantidad de frases de entrenamiento en una intención, todas ellas iguales, pero con distintas ciudades. Podemos crear el concepto ciudad que englobe todas las ciudades y añadirlo en la frase de entrenamiento de la intención.

Para crear una entidad es necesario definir las palabras que quieres englobar dentro de ese concepto. De esta forma, todas ellas serán reconocidas como el mismo concepto dentro de las intenciones. Además, estas palabras podrían tener sinónimos, por lo que también se han de definir. Al igual que pasaba con las frases de entrenamiento de las intenciones, cuantos más ejemplos se añaden más fácil es para la interfaz conversacional reconocerlos. En la Figura 4.3 se pone como ejemplo de la creación de una de estas entidades, donde aparecen las ubicaciones contempladas de los sensores. Estas palabras que definen el concepto pueden aumentar de forma

automática gracias a que Dialogflow puede reconocer similitudes. Esto le da mucha más versatilidad.


☒ Define synonyms  ☒ Allow automated expansion



banio	banio, baño
cocina	cocina
comedor	comedor
despacho	despacho
dormitorio	dormitorio
entrada	entrada, recibidor
habitacion	habitacion, habitación
pasillo	pasillo
salon	salon, salón
terraza	balcon, balcón, terraza
Click here to edit entry	

Figura 4.3: Creación de una entidad

Una vez se tiene la entidad, hay que añadirla a las intenciones donde sea necesaria. En el caso de la intención que se utiliza para consultar la temperatura de los sensores es necesario el uso de la entidad de ubicación, ya que se debe poder consultar la temperatura de un solo sensor, ya que se quiere saber la medición en esa estancia, o la media de todos ya que lo que se busca es conocer la temperatura de la vivienda. Como se muestra en la Figura 4.4, se añade un parámetro a la intención para poder reconocer las distintas posibles ubicaciones del sensor.

En la Figura 4.4 se ve resaltado en amarillo la frase de entrenamiento que contiene una de las palabras que representa el concepto de la ubicación del sensor. Esto significa que la interfaz conversacional entiende que debe buscar ese concepto y no una palabra específica. En la parte inferior se muestra el concepto que debe estar presente. En este caso no es necesario, pero se puede marcar un concepto como obligatorio, de forma que, si el usuario no lo comunicase, se definirían preguntas adicionales para poder obtenerlo.

Training phrases 

Search training phrases  


” Add user expression


” ¿Que temperatura hace en el **salon**?





” ¿Que temperatura hace?

” ¿Cual es la temperatura que hace?

” ¿Cual es la temperatura?

Action and parameters 

Enter action name 

REQUIRED 	PARAMETER NAME 	ENTITY 	VALUE	IS LIST 
<input type="checkbox"/>	location	@location	\$location	<input type="checkbox"/>
<input type="checkbox"/>	Enter name	Enter entity	Enter value	<input type="checkbox"/>

[+ New parameter](#)

Figura 4.4: Adición de una entidad en una intención

En la Tabla 4.2 se muestran las entidades creadas como parte de la interfaz conversacional.

Entidad	Uso
Ubicación	Representa las posibles ubicaciones de los sensores dentro de una vivienda.
Sensor	Representa los distintos tipos de sensores que maneja el sistema.

Tabla 4.2: Entidades creadas en la interfaz conversacional

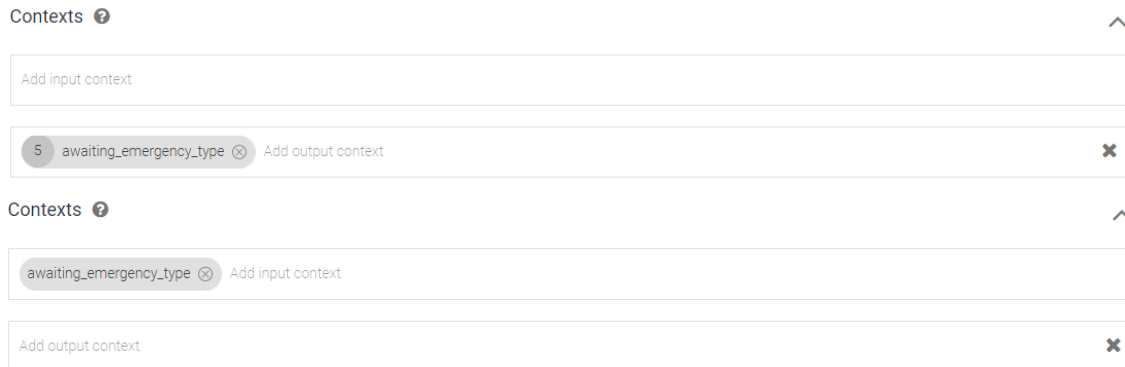
4.1.4. Uso de los contextos

Puede darse el caso, que para recoger información del usuario referente a un mismo tema se necesiten varias intenciones. Esto puede deberse a que se necesita de un proceso de varias preguntas y respuestas. Para ello se utilizan los contextos.

Los contextos pueden enlazar varias intenciones, de forma que cuando el usuario esté interactuando con la interfaz conversacional tenga la sensación de que esta es capaz de seguir una conversación, ya que enlaza preguntas enmarcadas en el mismo contexto. La principal diferencia entre volver a preguntar utilizando un contexto o utilizando una entidad no encontrada en una intención es que en la primera se deja la puerta abierta a recoger más información que puede ser útil.

Para enlazar intenciones utilizando los contextos es necesario definir un contexto de entrada y un contexto de salida. De esta forma, una intención con un contexto de salida será redirigida a otra intención que tenga ese mismo contexto, pero marcado

como entrada. En la Figura 4.5 se muestra un ejemplo empleado en las intenciones destinadas a que el usuario pueda solicitar asistencia.



The screenshot shows two panels of the Dialogflow console. The top panel, titled 'Contexts', shows an 'Add input context' field and a list of contexts. One context is selected: '5 awaiting_emergency_type' with a close button. Below it is an 'Add output context' field. The bottom panel, also titled 'Contexts', shows the 'awaiting_emergency_type' context selected in the 'Add input context' field, and an 'Add output context' field with a close button.

Figura 4.5: Contextos de entrada y salida

En este caso el uso de contextos es importante ya que, si el usuario no especifica el motivo de la necesidad de asistencia, es necesario redirigirlo a otra intención que pregunte por ese motivo.

4.1.5. Integración con Google Assistant

Una de las ventajas de Dialogflow es que se puede integrar con una gran cantidad de aplicaciones y servicios. Entre las integraciones disponibles se encuentran Google Assistant, Facebook Messenger, Slack, Viber, Twitter, Twilio, Skype, Telegram, Kik, Line, Cisco Spark, Amazon Alexa y Microsoft Cortana.

Debido a que en este proyecto se cuenta con el altavoz inteligente Google Home Mini, la integración se realizará con Google Assistant. Para la integración con este asistente, es posible añadir invocaciones explícitas e implícitas de la interfaz conversacional. Las invocaciones explícitas son aquellas que permiten llamar a la interfaz conversacional. Por ejemplo, “Ok Google, quiero hablar con mi asistente”. Aquí se reconocerá la intención de bienvenida. Por otra parte, las invocaciones implícitas permiten saltarse esta intención de bienvenida para ir lanzar directamente la intención deseada. Por ejemplo, “Ok Google, dile a mi asistente que necesito ayuda”. Aquí la intención que se lanzará será la de emergencia directamente y permitirá ahorrar tiempo. Se han añadido las intenciones relacionadas directamente con la asistencia como invocaciones implícitas.

Para completar la integración es necesario también realizar una serie de ajustes en la página de configuración *Actions on Google*. Desde esta página se gestionan todos los servicios que Google ofrece a los desarrolladores. La Figura 4.6 muestra una captura de pantalla del panel de configuración para la integración de la interfaz conversacional con Google Assistant.

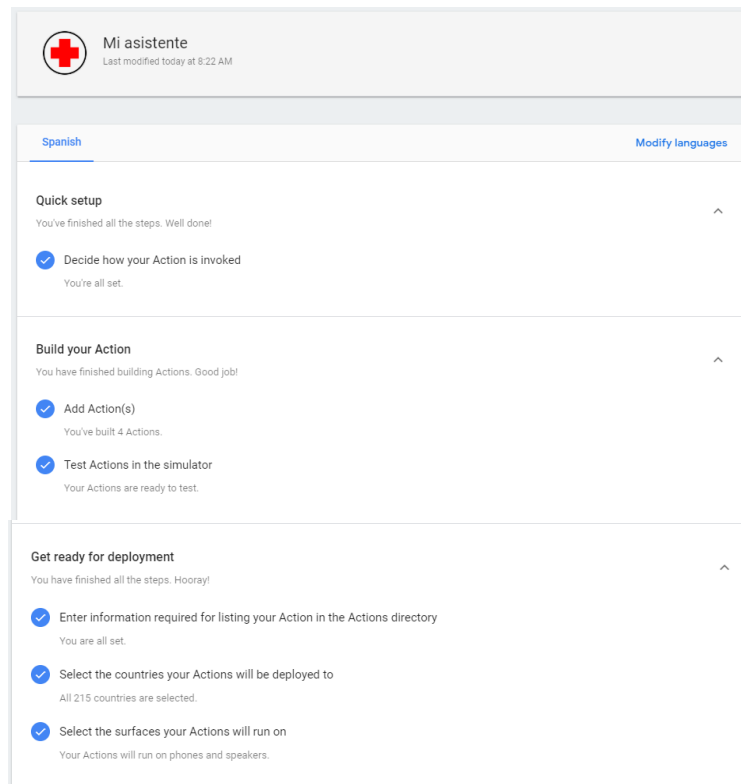


Figura 4.6: Ajustes de la interfaz conversacional para la integración con Google Assistant

Completando estas configuraciones básicas ya se puede comprobar el funcionamiento de la interfaz conversacional desde cualquier dispositivo que cuente con el asistente de Google. Además, completar estas configuraciones permite también publicar la interfaz conversacional para que pueda ser usada por el público en general.

4.1.6. Conexión con Firebase

Como se ha explicado en el epígrafe referente a la creación del agente (ver epígrafe 4.1.1), cuando se crea un agente, este ha de estar vinculado a un proyecto en la nube de Google. Esto permite integrar distintos servicios dentro del mismo proyecto, y por tanto también permite integrar Firebase dentro del proyecto.

En Firebase se crea una nueva aplicación en el mismo proyecto en el que está integrado el agente. La aplicación de Firebase no solo cuenta con la base de datos, sino también con todas las funcionalidades explicadas anteriormente (ver epígrafe 2.5). Una vez hecho esto, se le debe indicar al agente que el *fulfillment* que debe usar son las funciones de Firebase. Para ello se debe añadir la URL de Firebase que contiene las funciones necesarias para que la interfaz conversacional pueda consultar la base de datos. De esta forma, cuando en una intención se marque que es necesario el uso del *fulfillment*, el agente creado para la interfaz conversacional lanzará una petición contra esa URL, la cual mediante las funciones que se explicarán a continuación reconoce la intención y los parámetros y devuelve la información solicitada.

Cuando estas funciones reciben una petición identifican de que intención proceden y en función de eso se deriva la información a una función más pequeña que es la encargada de realizar la petición a la base de datos, utilizando si existieran, parámetros extraídos de la intención con el uso de una entidad.

Para la creación de estas funciones es necesario instalar en un sistema Linux los paquetes “*firebase-admin*” y “*firebase-functions*”, mediante los cuales se puede hacer uso del comando “*firebase deploy*” en la consola, el cual busca errores de sintaxis y actualiza las funciones que están disponibles en la nube para sincronizarlas con las que se han desarrollado en local. La Figura 4.7 muestra un extracto del código desarrollado en JavaScript para crear una función en Firebase que consulte la temperatura en la base de datos.

```
// Run the proper function handler based on the matched Dialogflow intent name
let intentMap = new Map();
intentMap.set('Check Temperature', checkTemperature);

// Check the Last measure of the temperature sensor on the database
function checkTemperature(agent) {
  return database.ref('sensors/data/lastMeasurement/temperature').once('value', snapshot => {
    const temperatureData = snapshot.val();
    const parameters = request.body.queryResult.parameters;
    const temperatureSensorData = temperatureData[parameters.location];
    if (parameters) {
      try {
        agent.add('La temperatura es de ' + temperatureSensorData.measure + ' ' + temperatureSensorData.units + '.');
      } catch (error) {
        agent.add('Ese sensor no está disponible.');
```

Figura 4.7: Función en Firebase para consultar la temperatura

Como se puede observar en este ejemplo, cuando el usuario quiere consultar los datos de temperatura, y por tanto hace uso de la intención destinada a tal efecto, esta información llega a las funciones de Firebase. En ella, se mapea la intención para seleccionar la función que atenderá la petición, en este caso comprobar la temperatura. Después, se hace una consulta a la base de datos mediante la cláusula “*once*”, la cual lee un dato de la ruta especificada pero no permanece a la escucha de los cambios. Una vez se han obtenido los datos, se comprueba si la petición realizada por Dialogflow contiene parámetros de alguna entidad. En caso de que se hayan especificado parámetros se devuelve la medición de temperatura del parámetro indicado y en caso contrario se devuelve la media.

Todas las funciones desarrolladas para Firebase siguen la misma estructura. Se busca la función que da respuesta a la intención recibida, se consulta en la base de datos, se comprueba, si procede, si en la petición se incluye algún parámetro procedente de una entidad y se devuelve la información.

No todas las intenciones tienen una función asociada, ya que no en todos los casos Dialogflow necesitará información de la base de datos.

4.2. Recogida de datos del entorno

Para poder dar *feedback* al usuario sobre el estado del entorno, es necesario que el sistema recoja datos de este. Para ello se utilizan una serie de sensores conectados a una placa electrónica Raspberry Pi. En este proyecto se ha implementado un número reducido de sensores, de forma que se pueda comprobar el correcto funcionamiento del sistema. No obstante, se ha diseñado el código de forma que añadir nuevos tipos de sensores sea sencillo.

El código que realiza la lectura de los sensores se ha implementado en Python, en su versión 3.5. Como se explicará a continuación, se ha implementado mediante el uso de un *virtual environment* (entorno virtual) para evitar problemas de dependencias.

4.2.1. Sensores

Los sensores recogen distintas magnitudes físicas que transforman en una señal eléctrica. Estos sensores se conectan a los pines GPIO de la Raspberry Pi. Esta placa cuenta con 40 pines. 26 de estos terminales son de entrada / salida, con resistencias *pull-up* y *pull-down* programables por software. En la Figura 4.8 se muestra el esquemático del circuito eléctrico de los pines GPIO de la Raspberry Pi.

Equivalent Circuit for Raspberry Pi GPIO pins

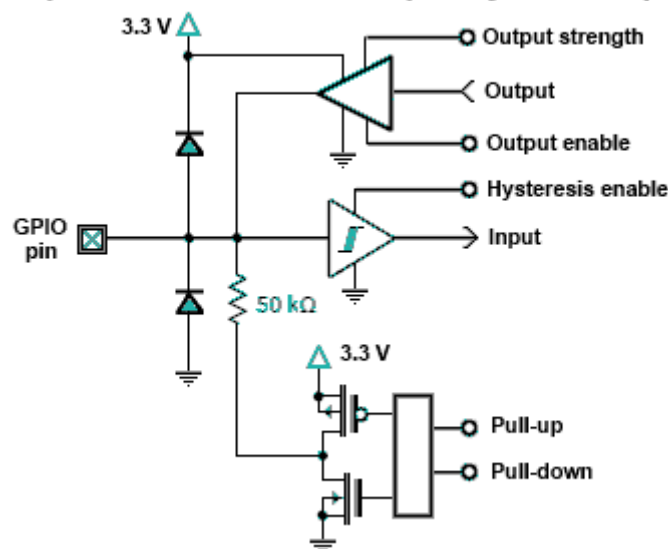


Figura 4.8: Circuito equivalente para los pines GPIO

Estos pines son los que se usarán para la conexión de los sensores con la placa. Esta placa no cuenta con entradas analógicas, solo digitales, por lo que la señal que llegue de los sensores debe ser digital. En la Figura 4.9 se muestra la disposición de todos los pines GPIO con los que cuenta la Raspberry Pi.

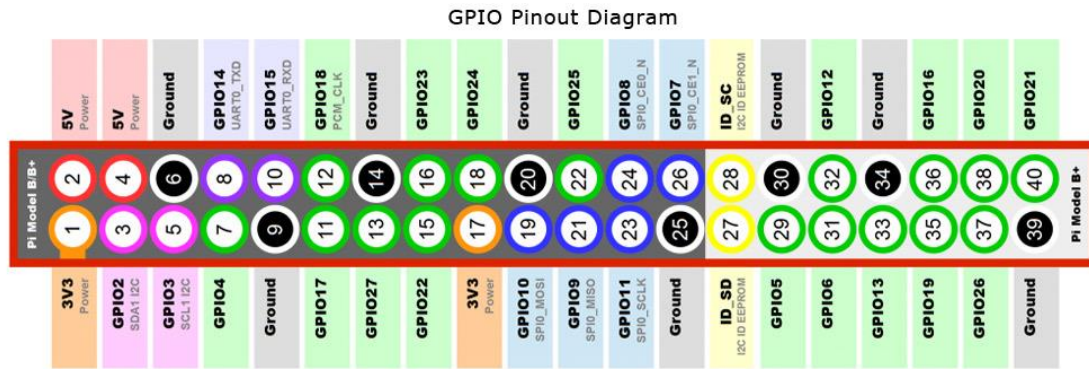


Figura 4.9: Pines GPIO de la Raspberry Pi 3 modelo B+

Todos los sensores implementados comparten alimentación proveniente de la placa a 5 voltios y están conectados al mismo punto de referencia en la placa, los pines marcados como *ground*. La alimentación da hasta un máximo de 1.5 amperios, suficiente para alimentar los sensores.

4.2.1.1. Sensor de temperatura y humedad

El sensor utilizado para la medición de la temperatura y la humedad ha sido el DHT11. Este sensor cuenta con un rango de medición de entre 0 y 50 grados centígrados (°C) de temperatura y de entre el 20 y el 90 por ciento de humedad relativa (RH). Da una precisión de ± 2 °C y de $\pm 5\%$ RH [28]. Estas características son suficientes para un entorno controlado como es una vivienda. Su alimentación está entre 3 y 5.5 voltios y consume un máximo de 0.5 miliamperios, por lo que es compatible con la alimentación que proporciona la placa. En la Figura 4.10 se muestra una imagen del sensor y su esquema de conexión.

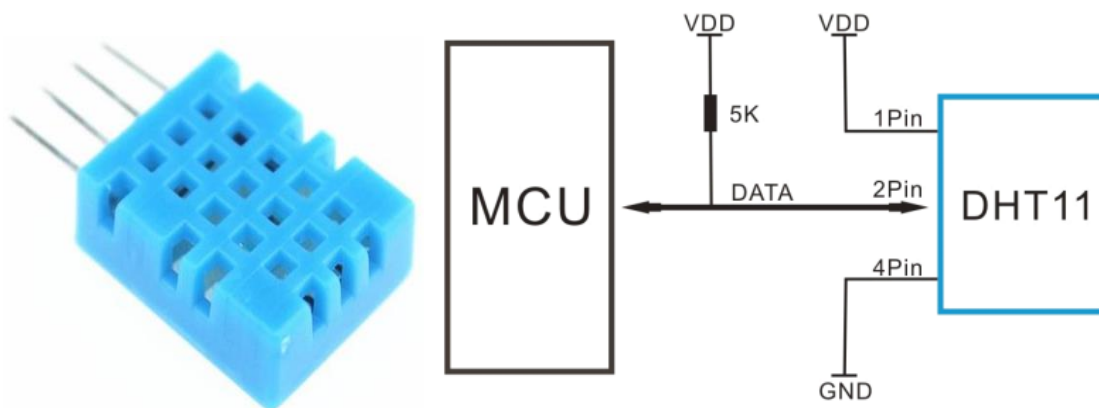


Figura 4.10: Imagen y esquema de conexión del DHT11

Cuando el cable de conexión es menor de 20 metros, como es el caso de este proyecto ya que se ha implementado sobre una *protoboard*, es necesario el uso de una resistencia *pull-up* de 5K.

4.2.1.2. Sensor de presencia

El sensor utilizado para la detección de presencia ha sido el HC-SR501. Este sensor cuenta con un piezoeléctrico sensible a los infrarrojos, encapsulado dentro de una lente Fresnel. Esto le da una distancia de detección máxima de 7 metros y un cono de detección de 110° de apertura. Se alimenta a una tensión de 5 voltios y su consumo es inferior a un miliamperio [29].

Cuenta con dos potenciómetros y un *jumper* con los que se pueden realizar ciertos ajustes sobre el sensor. Uno de los potenciómetros permite ajustar el tiempo de disparo de la señal mientras que el otro permite ajustar la sensibilidad. El jumper es utilizado para configurar si se genera un impulso o impulsos repetidos en las detecciones. En la Figura 4.11 se muestra una imagen del sensor junto con una explicación de la función de cada pin y potenciómetro.

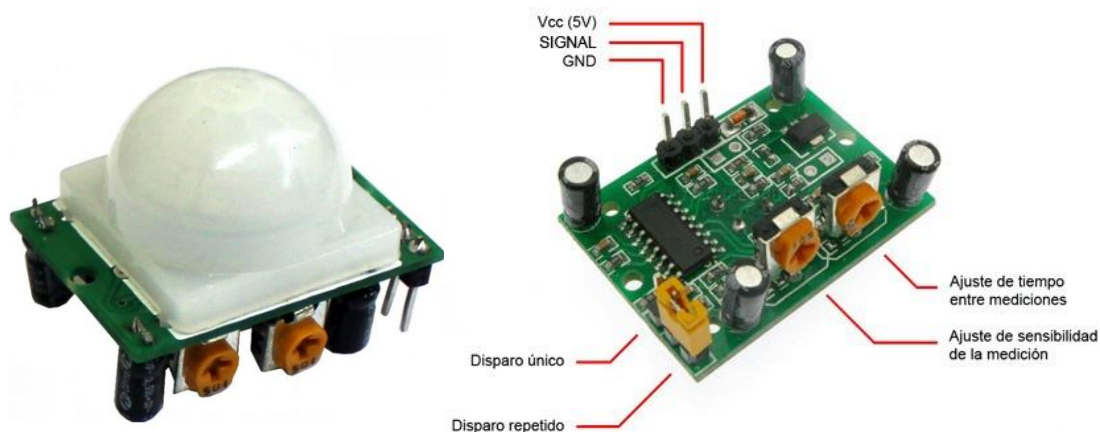


Figura 4.11: Imagen y esquema de conexión del HC-SR501

4.2.1.3. Sensor de calor

El sensor utilizado para la detección de calor (calor intenso como puede ser el fuego) ha sido el KY-026. Este sensor cuenta con un ángulo de detección de hasta 60° y es capaz de detectar longitudes de onda en el infrarrojo desde los 760nm hasta los 1100nm. Se alimenta a una tensión de entre 3.3 y 5.5 voltios y consume 15 miliamperios. Cuenta con una salida digital, donde se indica si se ha detectado llama o no (en función del límite de decisión establecido en un potenciómetro) y otra salida analógica. En la Figura 4.12 se muestra una imagen del sensor junto con la función de sus pines.

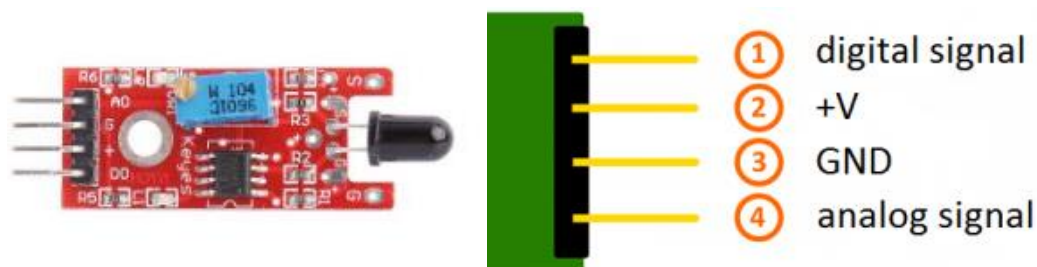


Figura 4.12: Imagen y esquema de conexión del KY-026

4.2.2. Software para la lectura de los sensores

La Raspberry Pi monta un sistema operativo Linux, por lo que se pueden utilizar lenguajes de programación de alto nivel. Python es el lenguaje que cuenta con las librerías oficiales para la lectura de los pines GPIO. Además, en Python se pueden utilizar los llamados *virtual environment*, que son entornos en los que se genera una instalación a parte de la instalación local de Python. Con esto se pueden instalar paquetes y librerías sin que afecte al resto del sistema, ya que se mantienen separados.

Gracias a mantener estos entornos separados, se puede crear un archivo de dependencias necesarias. Este archivo se genera automáticamente mediante la ejecución de un comando y almacena todas las librerías instaladas en el entorno en el que se ha ejecutado. Esto permite que, si se lleva la instalación del sistema a otra máquina, con este archivo generado se pueden instalar todas las dependencias y librerías necesarias de forma automática.

Para aumentar la automatización del proceso, se han desarrollado también dos *scripts* en bash para la instalación y puesta en marcha del código de Python. Ejecutando estos *scripts* no es necesario ejecutar uno a uno todos los comandos necesarios.

El código de Python sube los datos preprocesados obtenidos de los sensores a Firebase. Para ello es necesario autorizar al código para acceder a la base de datos. Esto se realiza descargando unas credenciales que proporciona Firebase, con las cuales el programa se autentica como autorizado a acceder a la base de datos. Estas credenciales se descargan desde el apartado de cuentas de servicio en la configuración del proyecto de Firebase.

El código desarrollado en Python establece una clase llamada “Sensor”, la cual representa un sensor cualquiera y sobre la que pivota todo el código. Esta clase tiene como atributos el tipo de sensor, la ubicación, las unidades de la medición, la medición en sí, si el sensor se encuentra disponible o no, si necesita de una interrupción, el modelo que analiza los datos anómalos y si la medición realizada se considera anómala o no. Todos estos atributos tienen sus correspondientes métodos para acceder a ellos o para modificarlos.

Cuando el *script* se inicia, se descargan los datos de configuración de los sensores establecidos mediante la página de configuración. Con esta información, se crea un objeto de la clase sensor por cada sensor establecido en la configuración. Una vez se tienen todos los sensores, comienza el bucle principal del programa. En la Figura 4.13 se muestra una captura de pantalla del bucle principal del código desarrollado en Python para la lectura de los sensores.

```
setUpSensors()
print("The sensor controller of the Dependents Assistant app is working")
while True:
    try:
        # Update ml model training once a day
        if getCurrentTime().hour == 2:
            for sensor in sensors:
                if not (sensor.getSensorCategory() in ('fire')):
                    trainData = getLastSensorMeasures(sensor)
                    mlModel = IsolationForest(
                        behaviour='new', contamination='auto')
                    mlModel.fit(trainData)
                    sensor.setMlModel(mlModel)

        # Checks all sensors
        for sensor in sensors:
            updateSensorMeasure(sensor)

        time.sleep(updateTime)

    except KeyboardInterrupt:
        print("The sensor controller of the Dependents Assistant app is stopping")
        GPIO.cleanup()
        sys.exit()
```

Figura 4.13: Bucle principal para la lectura de los sensores

El proceso es bastante sencillo y está implementado mediante funciones, de forma que el bucle principal queda más limpio y legible. Una vez se han configurado los sensores se entra en un bucle infinito que solo se detiene cuando el usuario lo interrumpe. Los modelos de aprendizaje automático de los sensores se actualizan una vez al día a las dos de la madrugada, de forma que no interfiera en el uso normal. El resto del tiempo el programa está leyendo cada cierto tiempo los sensores, analiza si puede tratarse de una anomalía, computa la media si hubiese más de un sensor del mismo tipo y sube los datos a la base de datos.

4.3. Análisis de los datos mediante algoritmo de aprendizaje automático

Como se ha explicado en el capítulo anterior, se hace uso de un algoritmo de aprendizaje automático para detectar posibles datos anómalos. El algoritmo utilizado es Isolation Forest (ver epígrafe 2.3.3) y se ha implementado mediante la librería de Python Scikit-learn (ver epígrafe 2.3.4.3). Para comprobar su efectividad y realizar pruebas, se ha implementado primero en IPython Notebook, el cual es muy utilizado por la comunidad científica ya que permite ejecutar código de Python en fragmentos separados denominados celdas. Esto da la posibilidad de generar gráficas intermedias y depurar mejor el código. Se ha utilizado la herramienta Colaboratory de Google, con la cual se pueden crear estos *notebooks* online y ejecutarlos en cualquier ordenador, sin necesidad de instalación de ningún tipo.

Una vez se ha comprobado la validez del algoritmo, se implementado en el código de Python desarrollado para la lectura de los sensores, de forma que en el dato que se suba a Firebase aparezca si se considera anómalo o no.

4.3.1. Desarrollo en un *Notebook*

Debido a que cualquier error que se produzca durante el desarrollo detendría la ejecución del código, se ha tomado la decisión de realizar la implementación primero en un *notebook*, que son usados precisamente para estos fines, ya que se pueden ejecutar partes independientes del código separados en celdas y en el orden que se desee. De esta forma, se puede desarrollar más rápidamente y se evitan errores de concepto, ya que se pueden sacar gráficas y datos intermedios para realizar comprobaciones. En la Figura 4.14 se muestra el aspecto que tiene uno de estos *notebooks*. Son celdas de código que se ejecutan de forma independiente, dando cada una el resultado para la que está destinada.

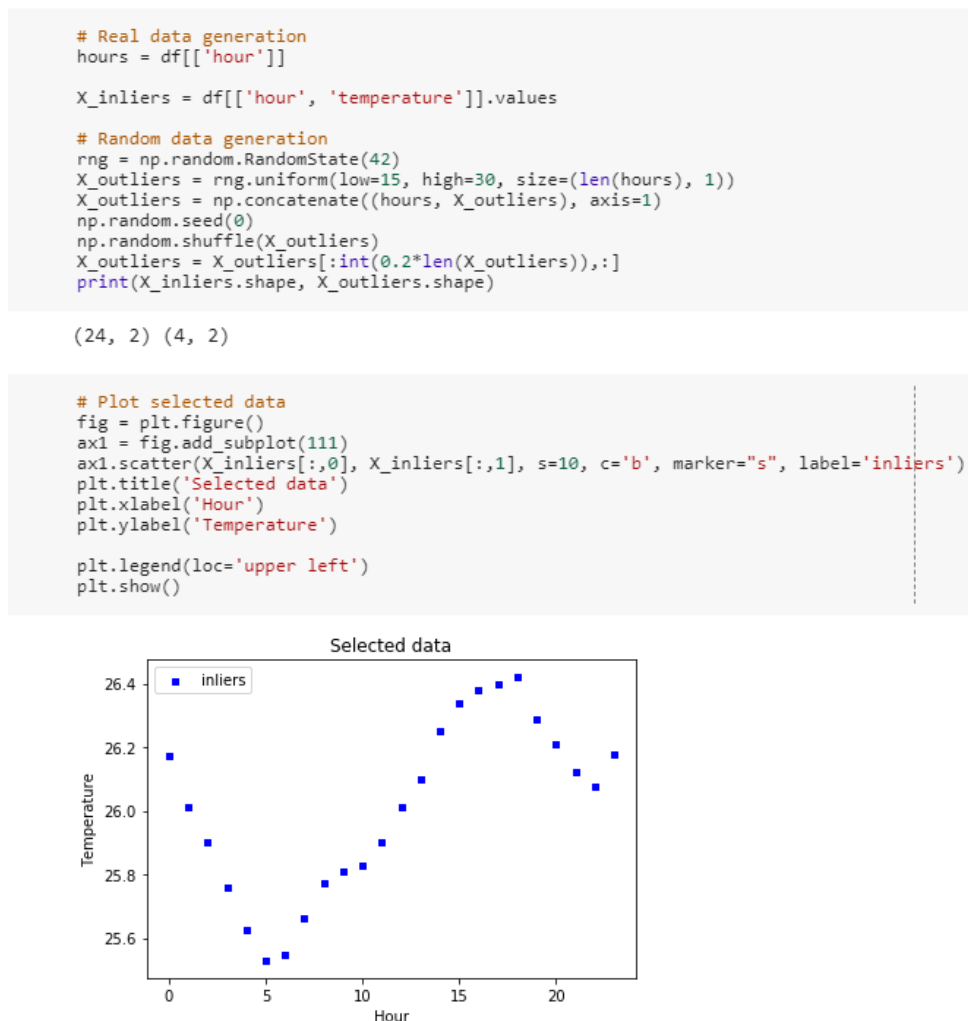


Figura 4.14: Aspecto de un *Notebook* de Python

Para realizar el desarrollo y prueba del algoritmo, se ha utilizado un conjunto de datos de temperatura a modo de ejemplo, extraído de un repositorio público de internet [30]. Estos datos están en formato CSV, por lo que se ha usado la librería Pandas para trabajar con ellos. Los datos de temperatura de este conjunto están en grados Fahrenheit, por lo que primero se realiza una conversión a grados Celsius. El conjunto de datos tiene una medición cada minuto, pero la temperatura no es un parámetro que cambie rápidamente, por lo que se selecciona solo una muestra por hora para trabajar mejor.

Estos datos no presentan anomalías, por lo que para entrenar al algoritmo es necesario introducir algunos datos que sí puedan representar una anomalía. Para ello, se introducen de manera aleatoria algunos datos, quedando el conjunto de datos como se muestra en la Figura 4.15.

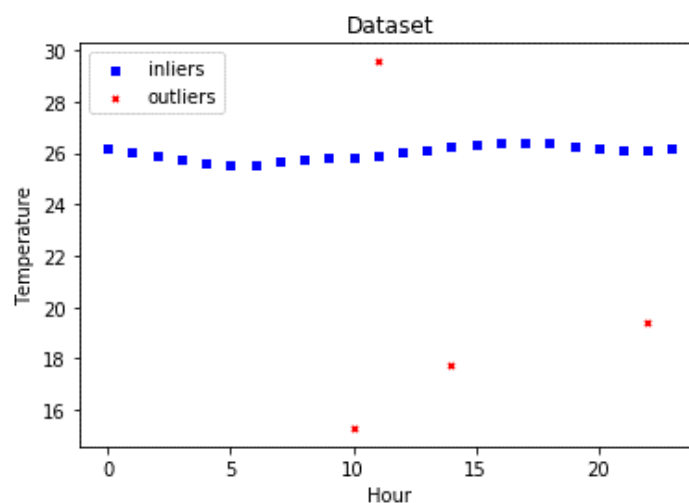


Figura 4.15: Conjunto de datos de ejemplo con anomalías

A continuación, se divide el conjunto de datos, de forma que se pueda realizar el entrenamiento con una parte de los datos, y luego comprobar su precisión con unos datos que el algoritmo no ha visto nunca. Se crean dos subconjuntos, el de entrenamiento que contendrá de forma aleatoria en 70 por ciento de los datos originales y el de test que contendrá en 30 por ciento restante.

Con esto, se procede a realizar el entrenamiento utilizando el algoritmo Isolation Forest. Para ello se crea el modelo utilizando el objeto de Scikit-learn del mismo nombre y se utiliza el método *fit* para entrenar el algoritmo, solo con el subconjunto de entrenamiento.

Una vez finalizado el entrenamiento, que debido a la reducción de mediciones hecha al principio solo toma unos pocos segundos, hay que comprobar la validez del entrenamiento, viendo si realiza predicciones suficientemente correctas. Al ser un algoritmo de aprendizaje no supervisado, a priori no se sabe de antemano si un dato es una anomalía o no. No obstante, este desarrollo se ha creado para comprobar la validez del algoritmo y por tanto si se sabe que datos son anomalías y cuáles no, por lo que se puede realizar esta evaluación.

Para realizar esta comprobación, se utiliza la curva ROC. Esta curva es una gráfica de la sensibilidad frente a la especificidad en un sistema de clasificación según va cambiando el umbral de discriminación. También puede interpretarse como la representación del ratio de verdaderos positivos frente a falsos positivos. Por ello, el área bajo esta curva indica como de buenos son las predicciones hechas por el algoritmo. Un área bajo la curva de 0.5 es equivalente a lanzar una moneda. Para que se considere un resultado bueno el área bajo la curva debe ser superior a 0.75.

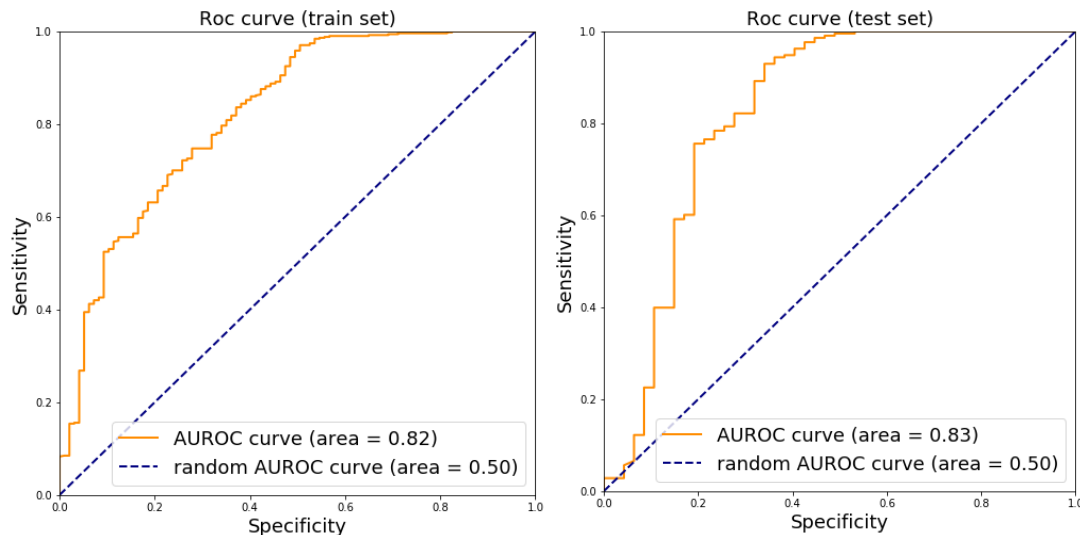


Figura 4.16: Curvas ROC para los subconjuntos de entrenamiento y test

Como se puede ver en la Figura 4.16, los resultados obtenidos son bastante buenos. El área bajo la curva es en ambos casos superior a 0.8, lo que se considera un buen resultado. La curva representada por una línea azul discontinua sirve de referencia e indica cuando las predicciones son indistinguibles de lanzar una moneda.

4.3.2. Implementación en el sistema

Una vez se tiene la metodología y se ha comprobado que se obtienen buenos resultados de este algoritmo, se procede a la implementación en el *script* de Python que realiza la lectura de los sensores. Esta implementación, gracias al paso anterior, es mucho más sencilla y asegura mejores resultados.

Se añade a la clase “Sensor” mencionada anteriormente (ver epígrafe 4.2.2) un atributo que almacene el modelo creado para cada sensor en concreto, que ha sido entrenado con los datos obtenidos de ese sensor. Cada vez que se realiza una medición nueva, esta pasa por el modelo, el cual determina si puede tratarse de una anomalía o no, y anota el resultado en un atributo de la clase destinado a tal efecto.

El modelo de cada sensor se reentrena cada noche con los datos obtenidos en los días anteriores. Se busca con esto evitar falsos positivos debido a los cambios estacionales, en el caso de la temperatura y humedad, o cambios en la rutina en el caso de los sensores de presencia, por ejemplo.

4.4. Gestión de avisos y alertas

Cuando una medición de los sensores es considerada como anómala por el algoritmo, ya que se sale del patrón, o directamente se ha salido de los límites fijados por el usuario, se debe generar un aviso de este evento, ya que podría tratarse de una situación en la que la persona precise asistencia.

Para realizar esto se ha implementado un servidor Node.js en la Raspberry Pi que descarga todas las nuevas mediciones que se suben a la base de datos y comprueba si es necesario generar un aviso o no. Estos avisos pueden seguir dos vías distintas. Cuando se considere necesario avisar al usuario, el aviso generará un mensaje de voz que se emitirá a través del altavoz inteligente Google Home Mini utilizado en el proyecto. Cuando se considere necesario avisar al contacto de emergencia se enviará un correo electrónico a la dirección introducida en la página de configuración. También se pueden generar ambos tipos de avisos simultáneamente.

4.4.1. Servidor

El servidor ejecutado en las Raspberry Pi está desarrollado sobre Node.js y permite establecer una conexión con el Google Home. Además de eso, tiene otras funcionalidades como enviar los correos electrónicos de aviso y servir la página de configuración. Para poder ejecutarlo primero se ha de instalar Node.js. La instalación y ejecución del servidor está automatizada con *scripts* de bash al igual que el software desarrollado en Python para la lectura de los sensores. Todas las librerías y dependencias necesarias se instalan automáticamente ejecutando estos *scripts*.

El servidor se ejecuta en local en las Raspberry Pi en un puerto que se ha configurado para que sea el 3000. Este puerto se puede cambiar fácilmente sin necesidad de tocar el código ya que se encuentra dentro de un JSON de configuración que se explicará posteriormente. Este puerto será el que de acceso a la página de configuración, indicando en el navegador la IP de la Raspberry seguido del puerto mencionado, o aquel que se configure.

Cuando el servidor se ejecuta, establece conexión con el Google Home. La configuración de esta conexión, así como el resto de las configuraciones disponibles se explicarán en el siguiente apartado.

Una vez se ha conseguido establecer conexión con el Google Home, se establece una conexión con Firebase para descargar las actualizaciones de los datos. Para poder descargar los datos que se actualicen en la base de datos se utiliza el método “on”, el cual establece una escucha permanente en la referencia que se le indique. La referencia es la ruta dentro del árbol que se genera en la base de datos donde se aplica el método que se indica. Por ello, se establece este método en todos aquellos datos que sean susceptibles de cambiar durante el uso del sistema.

Para conocer el correo electrónico al que se deben mandar los avisos, este es el primer dato en descargarse, y se establece una escucha por si durante la ejecución el usuario realizase algún cambio, que esté siempre actualizado. De la misma forma se descarga la configuración establecida por el usuario sobre los sensores, lo que permite conocer sus parámetros y los límites establecido por el usuario.

Para conocer las últimas mediciones de los sensores el proceso es ligeramente diferente. Dado que de un mismo tipo de sensor pueden existir múltiples dispositivos, es necesario descargar los datos de aquel que proporciona nuevas mediciones, pero no del resto. Para ello, se utiliza el método “on” en una referencia donde se encuentran todos los sensores del mismo tipo, y se añade la cláusula “child_changed” para indicar que solo se necesitan aquellos datos que hayan cambiado. En la Figura 4.17 se muestra un ejemplo de una de las funciones que realiza una lectura de los datos del sensor en la base de datos.

```
firebaseDB.ref('sensors/data/lastMeasurement/fire').on('child_changed', function (snapshot) {  
  const fireSensorName = snapshot.key;  
  const fireSensorData = snapshot.val();  
  // console.log('Fire data: ', fireSensorData, '\n');  
  if (fireSensorData.measure === 1) {  
    console.log('ALERT: Fire on ' + fireSensorName + '\n');  
    sendTextInput(broadcast + 'Se ha detectado fuego en ' + fireSensorName + ', por favor compruebelo.');
```

Figura 4.17: Ejemplo de lectura de un sensor desde el servidor

En aquellos sensores que tengan implementados el reconocimiento de patrones, se comprobará además si el nuevo dato está marcado como una anomalía o no (en el ejemplo de la Figura 4.17 al tratarse de un sensor de fuego no se implementa esta característica). En caso de haberse identificado como una anomalía, envía una notificación al contacto de emergencia para que tenga conocimiento de este evento.

Como se observa en la Figura 4.17, se generan dos avisos. Uno mediante la función “sendTextInput”, la cual es la encargada de mandar un mensaje de voz a través del altavoz en el que está implementada la interfaz conversacional. El otro aviso se realiza mediante la función “sendEmail”, la cual manda un correo electrónico a la dirección fijada como contacto de emergencia en la página de configuración.

La primera hace uso de la librería *Google Assistant SDK* [31], la cual habilita al sistema a mandar mensajes de voz a través del altavoz haciendo uso de la funcionalidad de difusión. La segunda hace uso de la librería *nodemailer* [32], la cual gestiona el envío de los correos electrónicos. La Figura 4.18 se muestra un extracto del código desarrollado para mandar los correos electrónicos al contacto asignado a tal efecto.

```
// Sends an email when a alert is triggered
function sendEmail(msgSubject, msgText) {
  const mailOptions = {
    from: emailAccount.auth.user,
    to: destinationEmail,
    subject: msgSubject,
    text: msgText
  };
  mailer.sendMail(mailOptions, function (error, info) {
    if (error) {
      console.log(error);
    } else {
      console.log('Email sent: ' + info.response);
    }
  });
}
```

Figura 4.18: Función para enviar los avisos por correo electrónico

4.4.2. Conexiones y configuración

Como se ha comentado en el apartado anterior, el servidor establece una serie de conexiones y comunicaciones que requieren de una configuración previa. En este apartado se explica cómo se realizan esas configuraciones. El servidor establece una conexión con el altavoz que implementa la interfaz conversacional para poder mandar los mensajes mediante voz. Se debe de conectar con Firebase para poder descargar los datos de los sensores. También necesita la información de una cuenta de correo electrónico activa para poder mandar las notificaciones. Además, cuenta con algunos parámetros de configuración que lo hacen un poco más flexible y evitan tener que modificar el código.

En este apartado no se mostrarán capturas del proceso ya que los datos que en ellas aparecerían son de carácter sensible. Todas las credenciales que se explican a continuación quedan guardadas en una carpeta que no es subida a ningún repositorio en la nube, con la intención de preservar la seguridad.

Para la conexión con el Google Home, la librería utilizada requiere descargar un archivo OAuth2 JSON de Google, de forma que acredites tu usuario. Esta descarga debe realizarse desde la consola de APIs de desarrolladores de Google. Una vez se tiene el archivo, solo se ha de indicar su ruta en el archivo de configuración creado para que se tenga en cuenta.

Para la conexión con Firebase, también es necesario la descarga de unas credenciales, que den acceso a la lectura de los datos, ya que la base de datos por defecto no lo permite por motivos de seguridad. La descarga de estas credenciales se realiza desde la consola de Firebase, en el apartado de cuentas de servicio en la configuración del proyecto.

Para que la librería empleada pueda mandar las notificaciones necesarias, se debe introducir en un archivo JSON las credenciales de la cuenta de correo electrónico

que se desean usar para este fin. De esta forma, la librería puede utilizar la cuenta con el fin de notificar los avisos.

Por último, en el archivo de configuración, se puede cambiar el puerto desde el que se accederá al servidor, que no se notifique el inicio del sistema mediante un mensaje de voz o el idioma utilizado para la comunicación de los mensajes.

4.5. Base de datos

La base de datos, como se ha comentado ya en varios capítulos y epígrafes anteriores, esta implementada en Firebase. Es una de las partes más sencillas del proyecto, pero a la vez una de las más importantes. En ella se almacenan todos los datos de los sensores desde que se inicia el sistema. Esta recopilación de datos es la que permite establecer un algoritmo de aprendizaje automático que aprenda sobre todos estos datos recogidos. Además, también permite que las distintas partes del sistema trabajen siempre con los mismos datos, lo que le aporta integridad. También ayuda a establecer parámetros de configuración, que de otra forma estarían fijos en el código del sistema.

4.5.1. Creación de la base de datos

La creación de una base de datos en Firebase es un proceso sencillo. Si ya se tiene un proyecto de Google en la nube (como es el caso debido al proyecto de Dialogflow) solo se debe vincular a este proyecto. Cuando se crea un proyecto de Firebase no se crea solo una base de datos, sino todo un ecosistema destinado al desarrollo de aplicaciones en la nube. El sistema desarrollado solo utiliza la base de datos en tiempo real y las funciones, comentadas anteriormente (ver epígrafe 4.1.6).

Cuando se crea el proyecto en Firebase, se debe indicar la ubicación, de entre las posibles, donde se quiere que se establezca la base de datos. Esta elección es necesaria ya que proporcionará una menor latencia.

4.5.2. Estructura de la base de datos

Firebase proporciona dos tipos de almacenamiento. Uno tiene el nombre de *Cloud Firestore*, la cual se base en el almacenamiento de documentos similares a JSON, y la otra se llama *Realtime database* (base de datos en tiempo real), que es la empleada en este proyecto.

La base de datos en tiempo real almacena los datos en un único árbol de datos de tipo JSON. Las consultas son profundas, lo que significa que se devuelve todo el subárbol de los nodos que se consulten. Además, es una solución regional, ya que los

datos se almacenan en un único servidor regional, lo que mejora la latencia en detrimento de la disponibilidad.

La base de datos se ha estructurado de tal forma que se intenta evitar en la medida de lo posible la anidación de datos, haciéndola más escalable. Se ha dividido en dos ramas, una para almacenar los datos referentes al usuario y otra para almacenar la información recogida por los sensores y su configuración.

En la rama referente al usuario se guardan datos como el correo electrónico de contacto o su estado (si está en casa o no, etc.). Además, también se incluye un campo que sirve para la notificación de las solicitudes de asistencia. Cuando el usuario solicita asistencia, en este campo se guarda esa solicitud y el motivo de esta, datos que serán descargados por el sistema de avisos para generar la notificación correspondiente. En la Figura 4.19 se muestra una captura de pantalla del árbol creado en la base de datos para guardar los datos referentes al usuario.

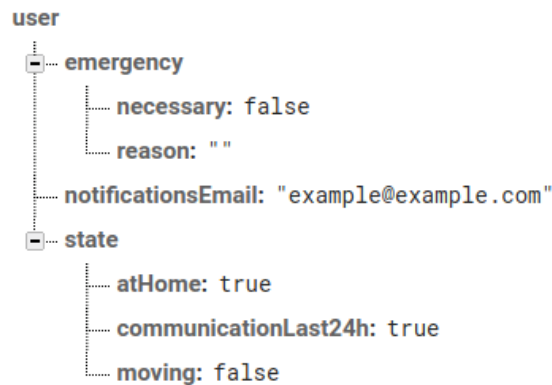


Figura 4.19: Subárbol de la base de datos para la información del usuario

En la rama referente a los sensores se guardan los datos necesarios para su configuración y las medidas que los sensores han proporcionado. Además, también guarda las categorías o tipo de sensores contemplados en el sistema, de forma que todas las partes del sistema estén sincronizadas y trabajen con los mismos tipos. En cuanto a las configuraciones, se guardan por separado por cada sensor, de modo que no existen configuraciones globales por tipo de sensor. Se almacena la ubicación en la que se encuentra, las unidades en las que proporciona la medición, el puerto de la Raspberry Pi donde se va a conectar y unos límites superior e inferior que fija el usuario para ese sensor. Si la medición del sensor cruza alguno de los límites marcados se generará un aviso.

Los datos almacenados de las mediciones de los sensores están divididos en dos ramas. Ambas tienen una estructura muy similar. Una de ellas guarda el histórico de todos los sensores. En ella, todos los datos recogidos durante el funcionamiento del sistema son almacenados junto con la fecha y la hora en la que se realizó la medición. Esto permite implementar el algoritmo de aprendizaje automático y entrenarlo con datos reales del lugar donde estás implantado el sistema.

La otra rama guarda solamente la medición más reciente de los sensores y la media de todos los sensores del mismo tipo. Con esto se agiliza el proceso de consulta para la gestión de los avisos, ya que no es necesario descargar todo el histórico. Además, la media ya ha sido calculada al recoger las mediciones de los sensores, lo que también aumenta la velocidad del proceso. En la Figura 4.20 se muestra una captura de pantalla del árbol creado en la base de datos para guardar los datos referentes a los sensores.

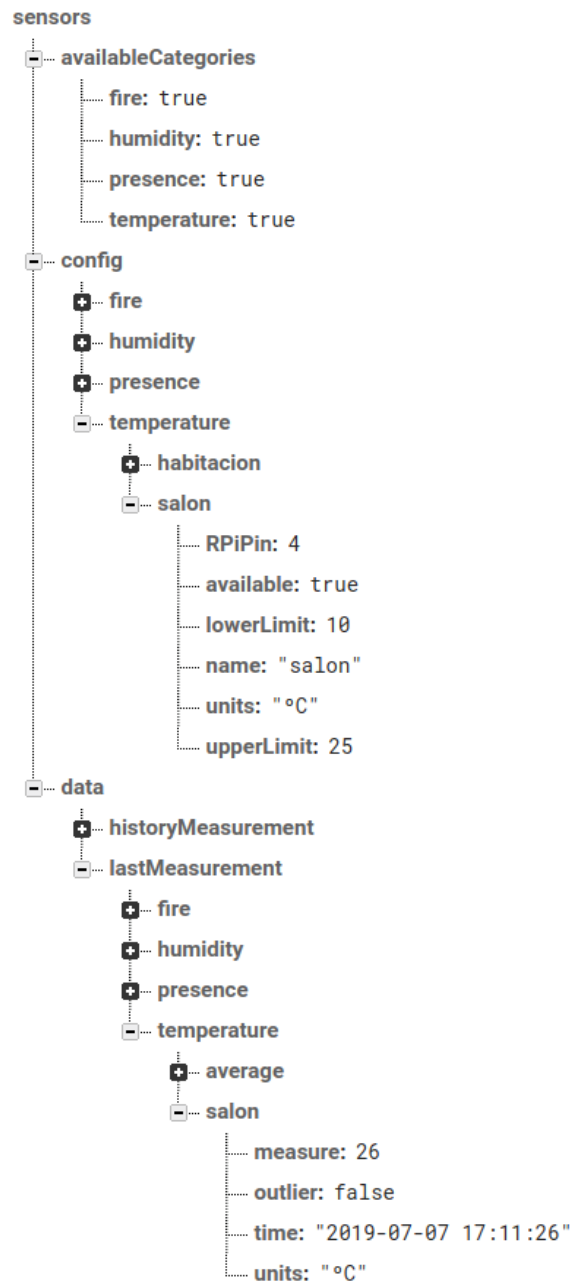


Figura 4.20: Subárbol de la base de datos para la información de los sensores

4.6. Página de configuración

La página de configuración es una parte fundamental del proyecto, ya que permite que el sistema se adapte a las necesidades del usuario de forma sencilla y se puedan cambiar parámetros del sistema sin necesidad de realizar cambios de código o similares. Esto dota al sistema de más modularidad, que es uno de los objetivos del presente proyecto.

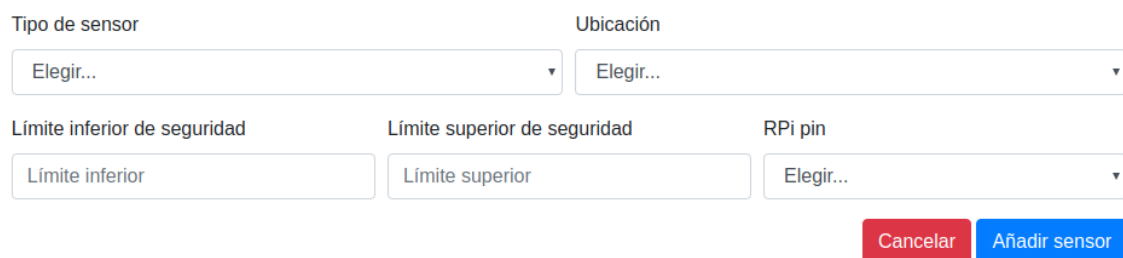
La página es mostrada por el servidor que se ejecuta en la Raspberry Pi, como se ha explicado anteriormente. La página se muestra en el puerto 3000 de la placa, por lo que para acceder a ella desde cualquier otro dispositivo solo se debe introducir la dirección IP de la Raspberry Pi seguido de dos puntos y el puerto indicado. De esta forma se puede acceder, por ejemplo, desde un *smartphone* conectado a la red.

En los siguientes apartados se explica cómo se ha diseñado y porqué se le ha dado esta estructura y diseño. La premisa principal a la hora del desarrollo de esta página de configuración ha sido la de crear una interfaz sencilla y amigable, pero sin perder las funcionalidades que se necesitaban implementar. La página está desarrollada con HTML5 y JavaScript, que son los principales lenguajes utilizados en el desarrollo de páginas web. Con HTML5 se ha creado la estructura y ubicado los elementos y con JavaScript se han creado todas las funciones que se conectan con la base de datos y van actualizando la página, de forma que no sea estática.

4.6.1. Estructura y diseño

El diseño de la página es simple y limpio. No existen diferentes páginas entre las que navegar, todos los datos están disponibles en la primera visión que se tiene. Para conseguir un diseño más elaborado se ha utilizado el *framework* Bootstrap, el cual proporciona una gran cantidad de elementos con un diseño cuidado. El diseño general puede verse en la Figura 3.4.

En la parte superior de la página, bajo el título se puede encontrar un cuadro de texto donde se puede introducir el correo electrónico al que se desean enviar las notificaciones. A la misma altura, pero en el lado derecho se encuentra el botón que permite añadir sensores al sistema. Cuando se presiona se despliega sobre el botón un formulario como el que se muestra en la Figura 4.21.



Formulario para la configuración de un nuevo sensor:

- Tipo de sensor:
- Ubicación:
- Límite inferior de seguridad:
- Límite superior de seguridad:
- RPI pin:
- Botones:

Figura 4.21: Formulario para la configuración de un nuevo sensor

Cuando se completan los campos del formulario y se presiona el botón añadir sensor, toda la configuración establecida se añade a la base de datos en el lugar apropiado y ya está lista para que el resto de las partes del sistema puedan trabajar con ella.

El resto de la página está ocupada por las mediciones de los sensores. Estas están agrupadas por el tipo de sensor y se muestran junto a sus unidades de medida (cuando estén disponibles). También están acompañadas de un botón que permite eliminar el sensor, de forma que el sistema no lo tiene en cuenta. En caso de no existir sensores de un tipo se muestra un mensaje indicándolo.

4.6.2. Funcionalidades

Todos los elementos explicados en el apartado anterior se crean de forma dinámica, dado que no se sabe el número de sensores o los tipos de estos disponibles, la página debe construirse en función a los datos alojados previamente en la base de datos. Para el caso inicial en el que no hay configuraciones previas en la base de datos, se establece un diseño por defecto.

Para crear los elementos de forma dinámica se utiliza el método “*createElement*” de JavaScript para generar una nueva etiqueta en el HTML y se le añaden las clases necesarias para que su posición y diseño sean los adecuados. Los elementos también se pueden eliminar, con el botón destinado a tal efecto, que representa un cubo de basura. Al eliminar un elemento, no solo hay que eliminarlo visualmente, sino también detener la actualización de su referencia en la base de datos.

El código en JavaScript que ejecuta la página genera una clase “Sensor” en la cual se almacenan las configuraciones de los distintos sensores descargadas desde la base de datos. Para realizar una actualización en tiempo real de los datos que muestra la página, se establece una escucha a la referencia de la base de datos que contiene las últimas mediciones de los sensores. En la Figura 4.22 se muestra una captura de pantalla del código desarrollado en JavaScript para la adición de sensores desde la página de configuración.

```
// Set a new sensor on the db
function addNewSensor() {
  const sensorName = document.getElementById('inputSensorLocation').value;
  const sensorCategory = document.getElementById('inputSensorType').value;
  const sensorLowerLimit = document.getElementById('inputSensorLowerLimit').value;
  const sensorUpperLimit = document.getElementById('inputSensorUpperLimit').value;
  const sensorRPIPin = document.getElementById('inputSensorRPIPin').value;
  const sensorUnits = setNewSensorUnits(sensorCategory);
  database.ref('sensors/config/' + sensorCategory + '/' + sensorName).set({
    name: sensorName,
    RPIPin: parseInt(sensorRPIPin),
    available: false,
    units: sensorUnits,
    lowerLimit: parseInt(sensorLowerLimit),
    upperLimit: parseInt(sensorUpperLimit)
  }, function () {
    document.getElementById('addNewSensorForm').style.display = 'none';
    document.getElementById('addSensorButton').style.display = 'flex';
  });
}
```

Figura 4.22: Configuración de nuevos sensores

Cada vez que se actualiza el correo electrónico destinado al envío de las notificaciones se actualiza también el valor almacenado en la base de datos. No precisa de un botón explícito de guardado.

Capítulo 5

EVALUACIÓN Y GESTIÓN DEL PROYECTO

En el presente capítulo se describe la evaluación que se ha llevado a cabo sobre el sistema de asistencia y control de personas dependientes que se ha desarrollado en este Trabajo Fin de Grado. Se explica la forma en la que se ha realizado la evaluación y se exponen los resultados obtenidos de la misma.

Además, se incluye también la planificación temporal que se ha seguido durante el desarrollo del proyecto de una forma pormenorizada. Se muestra la división del proyecto en distintas tareas y fases con su fecha de inicio y fin.

5.1. Metodología de evaluación

Para evaluar el sistema desarrollado se ha realizado un cuestionario a una serie de personas que han podido probar el sistema. Se ha buscado crear un cuestionario con preguntas lo más concretas posibles sobre diferentes aspectos: utilidad, funcionalidad, nivel de satisfacción, facilidad de uso y conocimiento sobre las tecnologías empleadas. Mediante este cuestionario se ha podido valorar el funcionamiento del sistema y como este es percibido por los usuarios.

El cuestionario consta de 10 preguntas, en las que se debe de dar una valoración del 1 al 5, siendo el 1 la valoración más desfavorable y 5 la valoración más favorable. Solo se puede dar una única valoración por pregunta. De esta forma, se busca tener un cuestionario suficientemente preciso para evaluar el sistema desarrollado, pero no excesivamente largo, lo que haría que al usuario le resultase tedioso.

La muestra de usuarios que han rellenado el cuestionario es muy heterogénea. De esta forma, se obtienen puntos de vista distintos que permitan encontrar mejoras para el sistema desarrollado. Los perfiles de las personas que han completado el

cuestionario de evaluación son: 1) Personas jóvenes que no se encuentran en situación de dependencia y cuya relación con la tecnología es estrecha. De este grupo se han obtenido 5 muestras. 2) Personas de mediana edad, que no se encuentran en situación de dependencia, pero no nacieron con las tecnologías actuales y por lo tanto han aprendido a usar algunas de ellas posteriormente a su juventud. De este grupo se han obtenido 5 muestras. 3) Personas de avanzada edad que se encuentran, en mayor o menor grado, en situación de dependencia ya que requieren de algún tipo de asistencia. De este grupo se han obtenido 4 muestras. El total de muestras obtenidas ha sido de 14 usuarios en un rango de edad de entre 21 y 89 años.

Para que los usuarios pudiesen rellenar el cuestionario, previamente debían probar el funcionamiento del sistema. Para ello, se les ha proporcionado el sistema ya funcionando y se les ha explicado las funcionalidades con las que cuenta. A partir de este punto, se les ha dejado hacer uso libre del mismo, pidiéndoles que trataran de hacer uso de todas las funcionalidades y de todo aquello que viesen conveniente. Para probar todas las funcionalidades, se les dejaba el sistema un par de días pidiéndoles que se acordasen de realizarle peticiones de vez en cuando.

La evaluación se ha realizado con la ayuda de la aplicación de Formularios de Google (<https://docs.google.com/forms/u/0/>) en la que se ha creado el cuestionario con las preguntas que se indican en la Tabla 5.1.

Cuestionario de evaluación					
1. Evalúe su experiencia con la tecnología	1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>
2. ¿Es capaz de valerse por sí mismo en todas las situaciones del día a día?	1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>
3. ¿La interfaz conversacional entendió todas sus peticiones?	1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>
4. ¿Las respuestas y avisos que proporciona la interfaz conversacional son claras y se entienden fácilmente?	1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>
5. ¿Le resultó fácil añadir o eliminar sensores del sistema o cambiar cualquier otra configuración?	1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>
6. ¿Cómo de preciso fue el sistema al proporcionar avisos?	1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>
7. ¿Su contacto de emergencia fue avisado siempre que lo solicitó?	1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>
8. En caso de haber utilizado cualquier sistema de teleasistencia previamente, ¿considera que este sistema supone una mejora?	1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>
9. En general, ¿el sistema le pareció fácil de usar?	1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>
10. En general, ¿el sistema le pareció útil?	1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>

Tabla 5.1: Preguntas del cuestionario de evaluación

5.2. Resultados de la evaluación

Los resultados obtenidos en la evaluación del sistema se muestran a continuación en Tabla 5.2. En ella se ha resumido la puntuación obtenida en cada una de las preguntas usando la media, para ver la valoración media, la mediana, para comprobar la valoración que se encuentra en la posición intermedia y la desviación típica, para ver la discrepancia de valoraciones en las respuestas.

Pregunta	Media	Mediana	Desviación típica
1. Evalúe su experiencia con la tecnología	3.2	3	1.7
2. ¿Es capaz de valerse por sí mismo en todas las situaciones del día a día?	3.8	4.5	1.6
3. ¿La interfaz conversacional entendió todas sus peticiones?	3.8	4	0.5
4. ¿Las respuestas y avisos que proporciona la interfaz conversacional son claras y se entienden fácilmente?	4.4	4.5	0.6
5. ¿Le resultó fácil añadir o eliminar sensores del sistema o cambiar cualquier otra configuración?	4.2	4	0.8
6. ¿Cómo de preciso fue el sistema al proporcionar avisos?	3.9	4	0.7
7. ¿Su contacto de emergencia fue avisado siempre que lo solicitó?	4.2	4	0.4
8. En caso de haber utilizado cualquier sistema de teleasistencia previamente, ¿considera que este sistema supone una mejora?	4	4	0.7
9. En general, ¿el sistema le pareció fácil de usar?	4.4	4	0.5
10. En general, ¿el sistema le pareció útil?	4.6	5	0.5

Tabla 5.2: Resultados de la evaluación

En términos generales se han obtenido unos resultados favorables. Los puntos mejor valorados han sido la utilidad del sistema, la facilidad de uso y la claridad con la que la interfaz conversacional se comunica con el usuario. Además, estos puntos cuentan con poca desviación, por lo que sabemos que los usuarios no discrepan en estos aspectos.

Como puntos más débiles, se encuentran la precisión del sistema de avisos y la precisión de la interfaz conversacional a la hora de entender las intenciones del usuario. De estos aspectos peor evaluados se puede deducir que el sistema todavía debe de ser mejorado para su uso en situaciones reales. No obstante, es un sistema que hace uso de inteligencia artificial, la cual aprende según va recopilando ejemplos (experiencia) y,

por lo tanto, en un periodo de prueba tan corto es complicado su correcto funcionamiento. Las dos primeras preguntas del cuestionario, a pesar de tener una media baja, no son valoraciones negativas, ya que en ella el usuario debe auto valorarse y sirven para observar si la muestra de usuarios empleada es homogénea o heterogénea. Como se indicaba en el epígrafe anterior, la muestra de usuarios es muy heterogénea, tal y como se ven en los resultados de la evaluación, tanto en la media como en la desviación típica de las dos preguntas iniciales.

En general, se han obtenido unos buenos resultados de la evaluación, que reflejan la consecución de los objetivos fijados al inicio de este Trabajo Fin de Grado.

5.3. Planificación temporal

Para la realización del presente Trabajo Fin de Grado se ha realizado una planificación temporal mediante el uso de un diagrama de Gantt. Este diagrama sirve para planificar temporalmente el desarrollo de un proyecto y sus tareas. El tiempo empleado por día para la realización de las tareas ha sido variable. Debido a la universidad y al trabajo el tiempo disponible para emplear en el desarrollo de este proyecto no ha podido ser superior a las cuatro horas diarias entre semana. Durante sábados y domingos el tiempo empleado ha llegado hasta las seis y ocho horas. También durante ciertas épocas no he podido realizar avances.

Las tareas están enmarcadas dentro de cuatro fases, tal como se explicó en el primer capítulo de este documento. En la Tabla 5.3 se muestran las tareas que se han desarrollado para completar el Trabajo Fin de Grado.

Fecha de inicio del proyecto		02/02/2019	
Fecha de finalización del proyecto		10/08/2019	
Duración total		189 días	
TAREA	INICIO	FIN	DURACIÓN
PLANIFICACIÓN	02/02/2019	10/03/2019	36 días
Estudio de las situaciones de dependencia	02/02/2019	06/02/2019	4 días
Análisis de los sistemas actuales de asistencia	07/02/2019	10/02/2019	3 días
Estudio de las interfaces conversacionales	11/02/2019	17/02/2019	6 días
Estudio de otras tecnologías necesarias	18/02/2019	03/03/2019	13 días
Estudio de algoritmo de aprendizaje automático	04/03/2019	10/03/2019	6 días
EJECUCIÓN	16/03/2019	16/06/2019	90 días
Diseño del sistema	16/03/2019	30/03/2019	12 días
Desarrollo	01/04/2019	16/06/2019	76 días
EVALUACIÓN	24/06/2019	13/07/2019	19 días
Pruebas de desarrollo	24/06/2019	30/06/2019	6 días
Evaluación del sistema	01/07/2019	13/07/2019	12 días
DOCUMENTACIÓN	11/03/2019	10/08/2019	152 días
Redacción de la memoria	11/03/2019	30/07/2019	141 días
Preparación de la presentación	01/08/2019	10/08/2019	9 días

Tabla 5.3: Fechas y duración de cada tarea

Las tareas se han ido realizando en orden secuencial, excepto la redacción de la memoria, la cual se ha ido completando con los avances conseguidos y se ha revisado una vez finalizada. Teniendo en cuenta que las horas dedicadas al día para la realización del proyecto han sido irregulares, con días en los que no he podido avanzar y otros en los que dedicaba una jornada laboral, la media de horas dedicadas a la realización de este trabajo es de aproximadamente 2,5 horas diarias. Por lo tanto, la duración total del proyecto en horas es de 472,5.

Con las tareas mostradas en la Tabla 5.3 se ha creado el diagrama de Gantt. Las tareas para desarrollar aparecen en el eje vertical, ordenadas de forma jerárquica, mientras que el eje horizontal contiene el tiempo. Las barras horizontales representan el tiempo destinado a la realización de cada tarea. El diagrama de Gantt mostrado en la Figura 5.1 se ha diseñado en Excel.

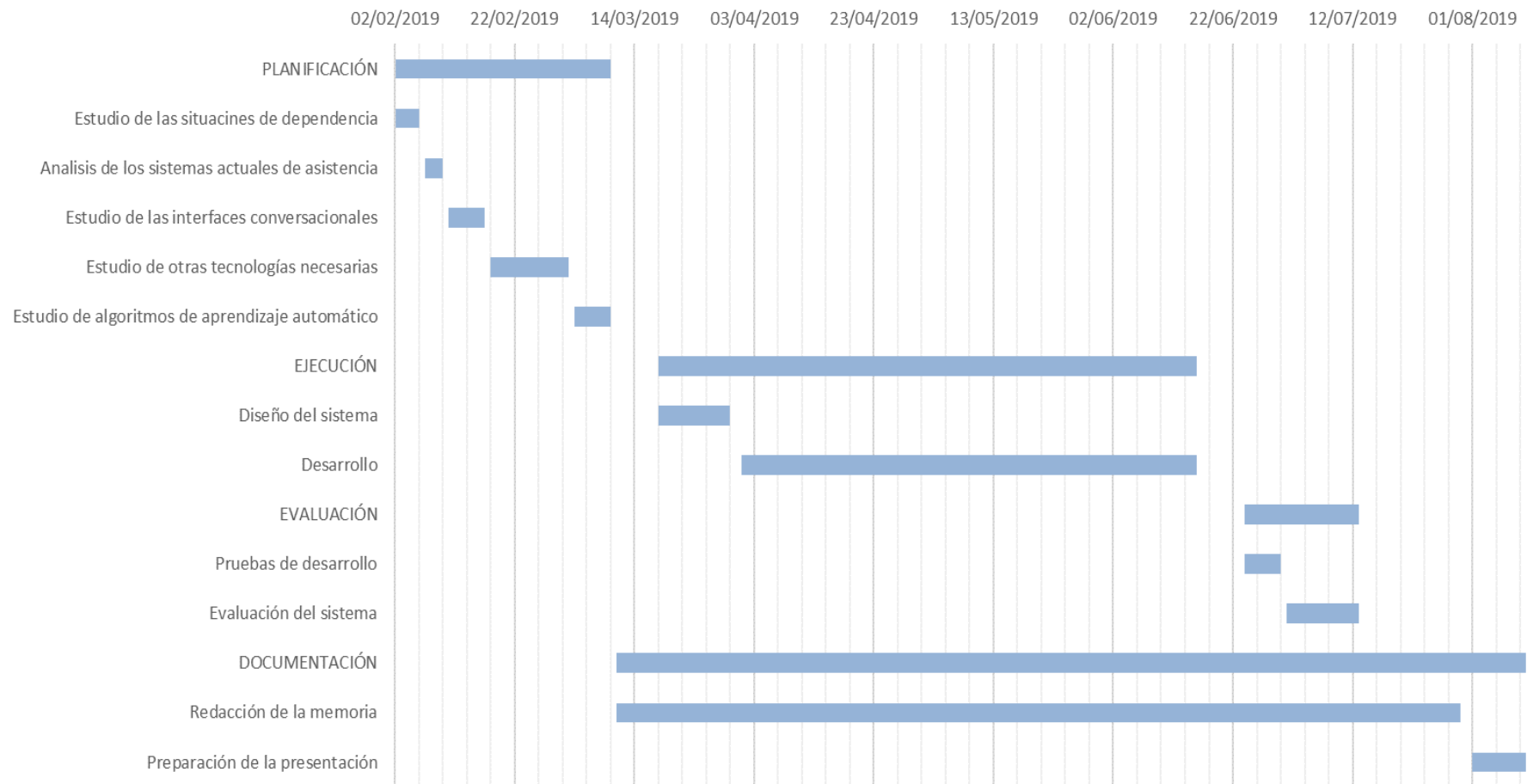


Figura 5.1: Diagrama de Gantt

Capítulo 6

MARCO REGULADOR Y ENTORNO SOCIO-ECONÓMICO

En el presente capítulo se exponen todas aquellas normativas o leyes que puedan afectar a la implantación o uso del sistema de control y asistencia a personas dependientes desarrollado en este Trabajo Fin de Grado. De esta forma se tienen en cuenta los aspectos necesarios para evitar sanciones tanto administrativas como de posibles usuarios, lo que además avala el sistema desarrollado.

También se realiza un análisis del entorno social y económico que rodea a las tecnologías utilizadas en este Trabajo Fin de Grado y a las situaciones donde se podría implantar y ser de utilidad.

6.1. Marco regulador

Cualquier sistema o aplicación que se desarrolle debe estar regulado por una serie de leyes y normas que garanticen un uso correcto del mismo, tanto por parte de los desarrolladores como de los usuarios. Las condiciones legales que un sistema como el desarrollado en el presente trabajo debería de cumplir son las siguientes:

- **Protección de datos:** es probablemente la condición legal más importante, ya que un sistema como el descrito recoge una gran cantidad de datos. La Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales [33] adapta al reglamento nacional el Reglamento (UE - 2016/679) General de Protección de Datos [34]. Esta normativa establece “las normas relativas a la protección de las personas físicas en lo que respecta al

tratamiento de los datos personales y las normas relativas a la libre circulación de tales datos.” [34]. Las sanciones por el incumplimiento de esta pueden alcanzar los 20 millones de euros.

- **Protección intelectual:** Si el sistema necesita un nombre, un logotipo o cualquier otro distintivo para ser comercializado debe de inscribirse en el Registro de la Propiedad Intelectual. La Ley 17/2001, de 7 de diciembre, de Marcas [35] reformada a través del Real Decreto 306/2019, de 26 de abril, establece “el régimen jurídico de los signos distintivos, categoría jurídica que configura uno de los grandes campos de la propiedad industrial”.
- **Licencias y condiciones de uso:** Se deben de establecer unos términos y condiciones de uso que el usuario debe de aceptar previa lectura de estos. De esta forma se dejan claras las responsabilidades tanto de desarrolladores como de usuarios en caso de uso fraudulento del sistema.
- **Menores:** En el caso de que el sistema fuese usado por menores de edad, las precauciones que se han de tomar deben de ser mayores. Será necesaria la autorización del tutor del menor de forma que quede constancia de su consentimiento expreso.

6.2. Entorno socio-económico

Los altavoces inteligentes son dispositivos que han visto como en poco tiempo ha aumentado su uso de forma casi exponencial. Se espera además que estos dispositivos superen en un futuro no muy lejano a las *tablets*. Y es que el concepto de casa inteligente hacia el que nos movemos parece que claramente será controlado con la voz y no con el uso de una pantalla. En la Figura 6.1 se muestra una gráfica elaborada por Canalys acerca de la estimación en la evolución del uso de distintos dispositivos.

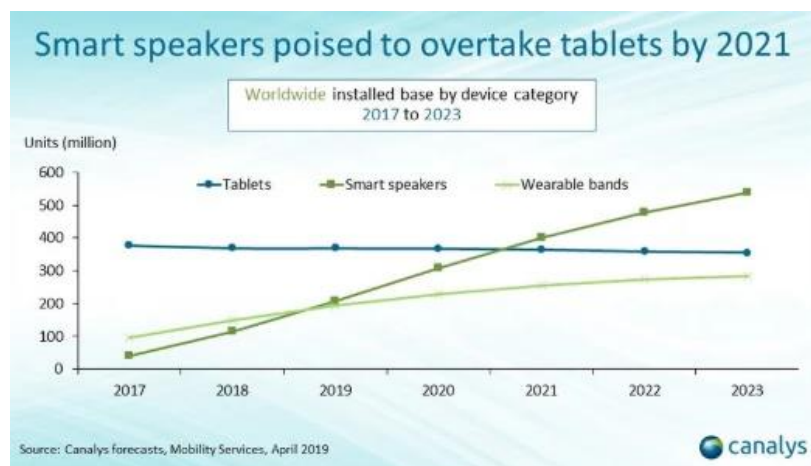


Figura 6.1: Expectativa de ventas de altavoces inteligentes en todo el mundo [36]

A la vista de estas previsiones, es de esperar que el impacto tanto social como económico de estos dispositivos sea enorme. Ya no se diseñarán tantas aplicaciones para su uso en una pantalla, sino que pivotarán hacia un diseño para su uso mediante la voz y una interacción cada vez más natural con el usuario. Esto provocará grandes cambios en la industria del desarrollo de aplicaciones móviles. En la Figura 6.2 se muestra una gráfica desarrollada por Statista acerca del uso que se da a los altavoces inteligentes.

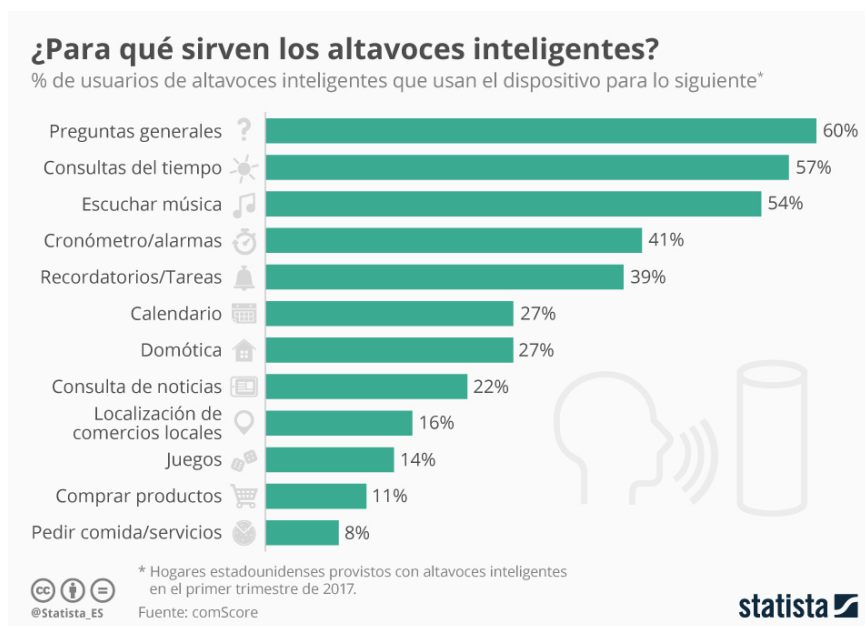


Figura 6.2: Principales usos de los altavoces inteligentes [37]

El uso que actualmente se da a los altavoces inteligentes está relacionado con las funcionalidades que por defecto trae el dispositivo. Los usos relacionados con aplicaciones de terceros son todavía escasos. No obstante, ante el panorama de crecimiento que se espera, es muy probable que la industria invierta en el desarrollo de funcionalidades para estos dispositivos y veamos cómo sus posibilidades de uso aumentan.

Es por todo ello que este sector en crecimiento generará un mercado aun mayor del que ya tiene, con el consecuente impacto económico positivo y la creación de puestos de trabajo.

El proyecto desarrollado en el presente Trabajo Fin de Grado busca apoyarse en ese crecimiento, y en el hecho de la facilidad y naturalidad en el uso de los altavoces inteligentes, para mejorar los servicios asistenciales de las personas dependientes. Tal y como se comentaba en el capítulo primero de este documento, la pirámide poblacional se invierte cada día más y no hay ningún indicio de este cambio de tendencia. Es por ello por lo que a cada día que pasa hay más personas que pueden estar en situación de dependencia y menos personas disponibles para prestarles la atención que necesitan. La esperanza de vida también aumenta por lo que el tiempo que una persona pueda encontrarse en situación de dependencia aumenta en consecuencia.

Debido a esto, es necesario automatizar procesos para poder atender a un mayor número de usuarios con un menor número de personas. Solo así se conseguiría atender la demanda asistencial de estas personas. Esto supondría un beneficio enorme para la sociedad, que solucionaría un problema al que ahora se enfrenta.

6.2.1. Presupuesto

En este apartado se realizará un análisis de los costes totales de la ejecución del presente Trabajo Fin de Grado. Se incluyen tanto los costes directos, que son los costes que genera directamente la realización del proyecto, como los costes indirectos, que no tienen una relación explícita con cada una de las partes del proyecto pero que son necesarios.

Dentro de los costes directos se enmarcan los costes de personal y los costes del material utilizado. Para los costes relacionados con el personal, se han tenido en cuenta las horas totales de trabajo destinadas a este proyecto (ver epígrafe 5.3). Se ha estimado un salario, para un graduado en ingeniería con menos de 2 años de experiencia a jornada completa en Madrid, de aproximadamente 25.000€ brutos al año. Normalmente a una empresa le supondría un coste del 166% de su salario bruto, lo que significa un coste de unos 41.500€ anuales. Teniendo en cuenta 250 días laborables al año, es coste por hora se aproximaría a 20,75€. Teniendo esto en cuenta se elabora la Tabla 6.1.

Días trabajados	Horas/día	Horas totales	Coste hora	Coste total
189	2,5	472,5	20,75€	9804,38€

Tabla 6.1: Presupuesto de personal

Para los costes relacionados con los materiales necesarios, se han tenido en cuenta los materiales que ya se tenían previamente y los que se han adquirido para la realización del proyecto. También se han añadido aquellos elementos que son necesarios para la realización del proyecto y son gratuitos en el momento de la realización del proyecto. En la Tabla 6.2 se muestra el presupuesto de materiales:

UNIDAD	DESCRIPCIÓN	N.º UNIDADES	PRECIO UNITARIO	PRECIO FINAL
CAPÍTULO I: ELEMENTOS HARDWARE				
Ud	Ordenador portátil MSI Procesador Intel Core i7-8550U, gráfica Nvidia MX150, 16GB de memoria RAM, almacenamiento SSD 512GB, pantalla de 14" FHD (1920*1080), batería de 4 celdas Ion de litio y 3290mAh.	1	1.199,00€	1.199,00€

Ud	Google Home Mini Conexión inalámbrica Wi-Fi 802.11b/g/n/ac (2,4 o 5 GHz), reconocimiento de voz de largo alcance y alimentación a 5V 1.8A.	1	59,99€	59,99€
Ud	Raspberry Pi 3 Model B Procesador 1.2 GHz Quad-core ARM Cortex-A53, 1GB de memoria RAM.	1	39,90€	39,90€
Ud	Pantalla externa LG Diagonal de 23.8", resolución nativa Full HD (1080p) 1920 x 1080 a 75 Hz, IPS, relación de aspecto 16:9 y clase energética A+.	1	119,00€	119,00€
Ud	Sensor de temperatura y humedad DHT11 Rango de entre 0 y 50 grados centígrados (°C y de entre el 20 y el 90% de humedad relativa (RH). Precisión de ± 2 °C y de $\pm 5\%$ RH. Alimentación entre 3V y 5.5V y 0.5mA.	1	1,45€	1,45€
Ud	Sensor de llama KY-026 Ángulo de detección de hasta 60° y longitudes de onda desde los 760nm hasta los 1100nm. Alimentación entre 3.3V y 5.5V, 15 mA.	1	2,24€	2,24€
Ud	Sensor de presencia HC-SR501 Distancia de detección de hasta 7 metros y cono de detección de 110° de apertura. Alimentación a 5V y 1mA.	1	2,27€	2,27€
Ud	Protoboard Placa de prototipado para la interconexión de elementos con 830 puntos y dos filas de alimentación.	1	3,10€	3,10€
Ud	Cables de conexión Cables para la conexión de los elementos macho-hembra, de 20cm de longitud. Pack de 40.	1	1,02€	1,02€
Subtotal.....			1.428,06€	

CAPÍTULO II: ELEMENTOS SOFTWARE

Ud	Ubuntu 18.04 Distribución Ubuntu del sistema operativo Linux basado en UNIX en su versión 18.04 LTS.	1	0,00€	0,00€
Ud	Microsoft Visual Studio Code Editor de código de Microsoft en su versión 1.36.	1	0,00€	0,00€

Ud	Python 3.5			
	Lenguaje de programación Python en su versión 3.5.	1	0,00€	0,00€
Ud	Scikit-learn 0.21			
	Librería Scikit-learn de modelos de aprendizaje automático y herramientas para el desarrollo en Python en su versión 0.21.	1	0,00€	0,00€
Ud	Node.js 10.16			
	Entorno multiplataforma en tiempo de ejecución para el lado del servidor basado en el lenguaje JavaScript, en su versión 10.16.	1	0,00€	0,00€
Ud	Dialogflow			
	Herramienta online para la creación de interfaces conversacionales, integrable con Google Assistant.	1	0,00€	0,00€
Ud	Firebase			
	Herramienta online para la gestión y almacenamiento de datos en tiempo real.	1	0,00€	0,00€
Ud	GitKraken			
	Ciente de Git para la gestión de versiones del código.	1	0,00€	0,00€
Ud	Windows 10 Home			
	Sistema operativo Windows 10 Home	1	69,99€	69,99€
Ud	Microsoft Office 365 ProPlus			
	Suite ofimática para la creación de documentos, tablas, gráficas, etc.	1	95,99€	95,99€
Ud	XODO PDF			
	Lector de documentos PDF, que permite también la edición y toma de notas sobre los mismos.	1	0,00€	0,00€
Ud	Draw.io			
	Herramienta online para la creación de diagramas y esquemas.	1	0,00€	0,00€
Ud	ClickUp			
	Herramienta online para la organización de tareas, que sigue la metodología SCRUM.	1	0,00€	0,00€
Subtotal.....				165,98€
TOTAL.....				1.594,04€

Tabla 6.2: Presupuesto de materiales

Por tanto, el total de los costes directos es la suma de los costes de personal y de los costes de material, que asciende a 11.398,42€.

Dentro de los costes indirectos se enmarcarían conceptos como el consumo de luz o la factura de Internet, entre otros. Estos recursos no se pueden asignar a ninguno de los elementos de forma directa, pero son igualmente necesarios para la realización del proyecto. Los costes indirectos se estima que pueden suponer un 20% de los costes directos. Por tanto, el total de los costes indirectos asciende a 2.279,68€. En la Tabla 6.3 se muestra un resumen de los costes.

TIPO DE COSTES	PRESUPUESTADO
Costes directos	11.398,42€.
Costes indirectos	2.279,68€
TOTAL	13.678,10€

Tabla 6.3: Resumen de costes

El presupuesto total de este proyecto asciende a la cantidad de TRECE MIL SEISCIENTOS SETENTA Y OCHO CON DIEZ EUROS.

Capítulo 7

CONCLUSIONES Y TRABAJO FUTURO

En este último capítulo se realiza un balance del trabajo realizado durante el presente Trabajo Fin de Grado. Basándose en los resultados obtenidos, se observa que las metas y objetivos fijados en el primer capítulo de este documento han sido alcanzados. Como resultado, a continuación, se desarrollan las conclusiones obtenidas.

Además de las conclusiones, se incluye un apartado sobre las posibles líneas de trabajo futuro que se podrían desarrollar sobre, o con base en, este proyecto. Se indican las posibles mejoras con el fin de aumentar la utilidad del sistema desarrollado en las páginas anteriores.

7.1. Conclusiones

En el presente Trabajo Fin de Grado se ha desarrollado un sistema que persigue la mejora en la asistencia no urgente a distancia a todo tipo de personas que la puedan necesitar, pero poniendo el foco en las personas en situación de dependencia. El objetivo es doble, por un lado, se busca la modernización de los sistemas actuales que prestan este servicio incorporando las últimas tecnologías y por tanto haciéndolo más accesible y funcional y, por otro lado, desarrollar un sistema que de forma autónoma pueda controlar el entorno de la persona y solicitar la asistencia que necesite, por lo que la intervención humana será menor pero más precisa.

Debido al auge que han tenido recientemente los asistentes por voz, ayudados también por su uso mediante los denominados altavoces inteligentes, decidí que estos eran ideales para desarrollar un sistema de asistencia. Estos asistentes proporcionan una interacción con la tecnología muy natural, por lo que a día de hoy los considero como la mejor herramienta para que las personas menos familiarizadas con la tecnología puedan usarla. Con esta idea en mente, y con la idea de que la tecnología debe servir para mejorar la vida de todas las personas, y no solo de aquellos que saben usarla, decidí centrarme precisamente en un colectivo que, generalmente por edad, no

son usuarios de mucha tecnología. Además, debido a malas experiencias de familiares cercanos con los servicios de teleasistencia, decidí buscar una mejor solución a los inconvenientes que estos presentan.

Los sistemas de teleasistencia utilizan un botón que el usuario puede pulsar cuando necesita asistencia, pero esta fuerte dependencia del botón para solicitar la asistencia hace que si, por el motivo que sea, el usuario no tiene el botón a mano, el sistema es prácticamente inútil. Se utiliza un botón ya que el sistema apenas ha sufrido ninguna evolución. Con este proyecto se buscaba construir el prototipo de sistema que pueda convertirse en el futuro de los sistemas de asistencia en remoto, eliminando la dependencia del botón y haciéndolo lo más modular posible, de forma que se adapte a todo tipo de usuarios.

Para diseñar dicho sistema, se han empleado múltiples tecnologías, pero el usuario solo interactúa con las partes diseñadas a tal efecto, que son las más sencillas. Como altavoz inteligente decidí usar un Google Home Mini, ya que disponía de él previamente. La decisión de utilizar un altavoz inteligente comercial y no diseñar uno propio, fue que el diseño de un altavoz inteligente hubiera requerido de muchos más recursos de los que disponía, y limitaba el proyecto. Por el contrario, al utilizar uno comercial, me permitía añadirle más funcionalidades distintas, más allá de una interfaz conversacional. El desarrollo de la interfaz conversacional no es algo complicado, pero necesita de un proceso de prueba y error y sobre todo de aprendizaje, para comprobar cuáles son las intenciones más frecuentes del usuario. Por otro lado, utilizar un altavoz ya existente ha limitado y modificado algunos aspectos de diseño.

Generar una interfaz conversacional proactiva, que iniciara una comunicación por voz con el usuario cuando fuese necesario (porque un sensor ha detectado un peligro, por ejemplo) era una de las prioridades. Al inicio esto fue un problema, ya que una de las limitaciones que imponía el altavoz inteligente elegido (aunque también la imponían otros en el mercado por motivos de seguridad) era precisamente que la interfaz conversacional iniciase una conversación. Tras muchos intentos y pruebas distintos, finalmente se consiguió que la interfaz lanzase avisos cuando el sistema fuese necesario, haciendo uso de la funcionalidad de difusión del altavoz. No obstante, aunque se consigue que la interfaz conversacional sea proactiva, esta no escucha la respuesta del usuario, lo cual podría ser útil en ciertas circunstancias y se propone como una línea de trabajo futuro.

Otro de los principales objetivos era el de crear un sistema modular, que se le puedan añadir o eliminar sensores de forma sencilla, de manera que el usuario pueda elegir que va a necesitar. Para esto, es fundamental escribir todo el sistema en función de unas variables y que, dependiendo del valor de esas variables, sea cual sea su valor, el sistema funcione correctamente. Este objetivo ha requerido de mucha organización y tiempo, ya que todas las partes del sistema deben funcionar correctamente compenetradas. Para que, como se decía anteriormente, el usuario pueda hacer un uso sencillo, estas variables que controlan y rigen el funcionamiento del sistema son establecidas por el usuario desde una página de configuración.

La creación de esta página de configuración es fundamental para completar el objetivo de modularidad y sencillez. Desde ella el usuario puede cambiar la configuración del sistema y de esta forma adaptarlo a sus necesidades.

Uno de los aspectos sobre los que se apoya la mejora de las interfaces conversacionales y que las hace cada vez más naturales es la inteligencia artificial. Con algoritmos de aprendizaje automático se pueden reconocer patrones, tanto en la voz, como en cualquier otro dato y, por tanto, quise implementar un pequeño algoritmo de aprendizaje automático. Con ello el sistema es capaz de reconocer patrones en los datos que recoge y proporcionar un aviso cuando un nuevo dato se sale del patrón. El punto fuerte de los algoritmos de aprendizaje automático, y por lo que decidí implementarlo, es que pueden generalizar una situación, y no siguen unas reglas fijas. Esto permite que se adapten al problema que se está tratando de forma personalizada a los patrones del usuario. Además, mi interés en todo lo que rodea a la inteligencia artificial es muy alto, por lo que este trabajo ha sido una oportunidad perfecta para aplicar conocimientos, pero sobre todo para ampliarlos.

Para todo el procesamiento de los datos y las mediciones de los sensores se ha utilizado una Raspberry Pi. Esta placa, si bien es cierto que no está destinada a profanos en la informática, es relativamente fácil de usar y tiene un rendimiento suficiente por un precio muy contenido. Por ello, he utilizado una, aunque el usuario final al que está destinado no sepa nada al respecto, ya que solo se han de conectar unos cables y el proceso es sencillo. Durante todo el desarrollo de este sistema de asistencia y control he ido ampliando y estructurando la base de datos, según iban apareciendo las distintas necesidades. La elección de una base de datos en la nube me permitía acceder a ella desde cualquier dispositivo además de ser un seguro contra la pérdida de datos por un fallo.

Para la realización de este Trabajo Fin de Grado, he tenido que adquirir una gran cantidad de conocimientos, sobre tecnologías que sabía que existían pero que nunca había usado, pero también de otras tecnologías que he ido descubriendo a base de investigar. Aprender es algo que me encanta y por ello no me he querido limitar a realizar un proyecto con todo aquello que ya sabía, sino que he preferido usar mis conocimientos como base para aprender nuevas cosas y explorar nuevas áreas de conocimiento.

Tras realizar un estudio de distintas tecnologías necesarias y de los sistemas de teleasistencia actuales, además de investigar sobre la dependencia, se ha logrado desarrollar con éxito un sistema de control y asistencia que implemente una interfaz conversacional proactiva para la interacción con el usuario. Con este trabajo, se da un paso más hacia la modernización de la asistencia remota a las personas que más lo necesitan.

Como conclusión final a este proyecto, se puede destacar que los objetivos que se habían fijado para el mismo se han alcanzado, y que tanto los conocimientos adquiridos como el desarrollo del sistema me han aportado una experiencia muy positiva y satisfactoria.

7.2. Trabajo futuro

Una vez finalizado el Trabajo Fin de Grado, se analizan los posibles puntos sobre los que se podrían desarrollar nuevas funcionalidades o implementar mejoras que no se han incluido por decisiones de diseño o por suponer un problema que excede los objetivos del proyecto pero que, no obstante, se han tenido en cuenta y se han meditado antes de excluirlos del proyecto. A continuación, se indican una serie de puntos sobre los que se podría ampliar este Trabajo Fin de Grado y que lo harían más completo:

- **Invocaciones a la interfaz conversacional:** Este primer punto está estrechamente relacionado con el siguiente. En este proyecto se ha desarrollado una interfaz conversacional que cuenta con una serie de funcionalidades. La herramienta que se ha utilizado para su creación permite su integración en otros servicios, pero dado que el altavoz inteligente del que se disponía era un Google Home Mini, se decidió integrarlo en el asistente de la misma compañía. Esta integración es muy ventajosa ya que permite disponer de la interfaz conversacional desarrollada en cualquier dispositivo que también cuente con el asistente de Google. El punto negativo se da ya que cada vez que se quiera hablar con la interfaz conversacional desarrollada, se deba invocar primero al asistente de Google. Estas dos invocaciones se pueden generar en la misma frase y esto no impide su correcto funcionamiento, pero es un paso extra que sería útil eliminar. Seguramente la integración con el asistente de Google deba mantenerse, debido a las ventajas que presenta, pero quizá solo en dispositivos que no sean el altavoz inteligente, y destinar este exclusivamente al sistema.
- **Altavoz inteligente propio:** Siguiendo la idea del punto anterior, al disponer en el proyecto de un altavoz inteligente Google Home, es necesaria la integración con su asistente ya que no es posible utilizar únicamente un asistente desarrollado por terceros. Además, no está permitido implementar una conversación que se inicie por el asistente, únicamente se pueden lanzar mensajes, pero no se puede recoger una respuesta del usuario si este no pregunta primero. Esto dificulta la interacción en ciertas ocasiones. Esto a su vez deriva en otra molestia, que no impide su uso pero que sería útil eliminar, y es que se necesita siempre la invocación del usuario. Esto es por motivos obvios de seguridad y funcionalidad, ya que grabar absolutamente todo lo que pasa es peligrosos desde el punto de vista de la privacidad. No obstante, con el desarrollo de un altavoz inteligente exclusivo para este sistema, se podrían realizar cambios como la fórmula de invocación o la conversación iniciada por el sistema. Esto lo haría un sistema mucho más completo.
- **Creación de una identificación de usuario:** El sistema desarrollado no cuenta con ningún sistema para la autenticación del usuario, ya que dentro del entorno doméstico no sería necesario. Sin embargo, esto

permitiría encriptar los datos para ese usuario y hacer el sistema más seguro.

- **Aumentar el número de sensores:** Esto es algo que, tal y como se ha diseñado el sistema (uno de los objetivos) es sencillo. Aumentar el número de sensores permite recoger más datos y por tanto dar un mayor rango de alertas. No obstante, cuanto mayor sea el volumen de datos, mayor deberá ser la capacidad de procesamiento, y en ese momento sería bueno valorar el analizar los datos en la nube, en lugar de en local.
- **Análisis cruzado de los datos:** Para poder detectar patrones complejos en los datos, es necesario realizar combinaciones y un preprocesamiento de estos. Esto le quita algo de modularidad, ya que las combinaciones a priori no se pueden realizar de forma automática sin la supervisión de una persona que valide los resultados que aporta el análisis de estas combinaciones. Un análisis más elaborado daría más información, pero para detectar los parámetros más importantes es necesaria una gran cantidad de datos.

GLOSARIO

API (Application Programming Interface): La interfaz de programación de aplicaciones es un conjunto de funcionalidades y rutas que ofrece cierta biblioteca o programa para que sean utilizadas por otro software externo.

Bash (Bourne-again Shell): Es un lenguaje que interpreta ordenes de consola en sistemas UNIX, y es el intérprete por defecto en la mayoría de las distribuciones GNU/Linux.

Conectores GPIO: Conectores de entrada o salida de propósito general (General Purpose Input/Output en inglés) que pueden ser programados para desempeñar distintas funciones.

CSV (Comma-Separated Values): Es un archivo que representa los datos en forma de tabla y que los almacena las columnas separadas por comas y las filas separadas por un salto de línea.

Framework: Es un conjunto estandarizado de prácticas, conceptos y métodos que enfocan un problema desde un punto de vista particular.

Hiperparámetros: Son los parámetros que definen la configuración de un modelo de aprendizaje automático y de cuya elección dependerá el rendimiento del modelo.

IoT (Internet of Things): Internet de las cosas es un concepto que hace referencia a la interconexión de todos los objetos de la vida cotidiana, mediante la inclusión de una conexión a Internet en estos objetos.

IP: Es un número que identifica de manera jerárquica a un elemento conectado a una red que utiliza el modelo TCP/IP.

JSON (JavaScript Object Notation): Es un formato de texto destinado al intercambio de datos.

Latencia: En redes, es la suma de todos los retrasos temporales que sufre una señal que atraviesa la red.

Librería (de código): Es un conjunto de funciones programadas en un lenguaje específico que aportan una serie bien definida de funcionalidades.

Script: En informática, se refiere a un archivo de órdenes, un programa no muy complejo destinado a una tarea concreta.

SDK (Software Development Kit): El kit de desarrollo de software es un conjunto de herramientas destinadas a la creación de una aplicación en un sistema concreto.

Smartphone: El teléfono inteligente es un dispositivo móvil con las capacidades de un teléfono que además incluye otras capacidades típicas de los ordenadores, gracias a los potentes procesadores que llevan instalados.

Subwoofer: Altavoz capaz de reproducir las dos primeras octavas del espectro audible de las 10 que lo forman.

URL (Uniform Resource Locator): Localizador de recurso uniforme es un estándar que designa y localiza los recursos disponibles dentro de una red, como por ejemplo Internet.

BIBLIOGRAFÍA

- [1] D. Fernandez, «El salto,» 11 Octubre 2018. [En línea]. Disponible: www.elsaltodiario.com/cuidados/espana-sera-cada-vez-mas-vieja-y-no-tendra-quien-cuide-a-sus-dependientes. [Último acceso: Septiembre 2019].
- [2] R. Pascual Cortes, «Cinco días,» 11 Octubre 2017. [En línea]. Disponible: www.cincodias.elpais.com/cincodias/2017/10/10/midiner/1507645811_272900.html. [Último acceso: Septiembre 2019].
- [3] D. Jalón, «Europa Press,» 26 Noviembre 2018. [En línea]. Disponible: www.europapress.es/comunicados/sociedad-00909/noticia-comunicado-mas-13-millones-espanoles-piensen-comprar-altavoz-inteligente-black-friday-navidad-20181126095017.html. [Último acceso: Septiembre 2019].
- [4] M. Contreras, «Clipset,» Enero 2019. [En línea]. Disponible: www.clipset.20minutos.es/amazon-alexa-caida-navidad-2018/. [Último acceso: Septiembre 2019].
- [5] E. Naone, «MIT Technology Review,» 24 Febrero 2009. [En línea]. Disponible: www.technologyreview.com/news/412191/tr10-intelligent-software-assistant/. [Último acceso: Septiembre 2019].
- [6] A. Springer, «Welt,» 20 Marzo 2012. [En línea]. Disponible: www.welt.de/newsticker/dpa_nt/infonline_nt/computer_nt/article106206488/Von-IBM-Shoebox-bis-Siri-50-Jahre-Spracherkennung.html. [Último acceso: Septiembre 2019].
- [7] P. Bejerano, «Blogthinkbig,» 20 Marzo 2018. [En línea]. Disponible: www.blogthinkbig.com/interfaces-conversacionales-las-tres-fases-de-la-tecnologia. [Último acceso: Septiembre 2019].
- [8] Armando, «Bebes y mas,» 4 Marzo 2015. [En línea]. Disponible: www.bebesymas.com/desarrollo/las-primeras-palabras-como-y-cuando-empiezan-a-hablar. [Último acceso: Septiembre 2019].
- [9] J. Sánchez, «ABC,» 28 Noviembre 2018. [En línea]. Disponible: www.abc.es/tecnologia/informatica/hardware/abci-ventas-altavoces-inteligentes-disparan-201811270331_noticia.html. [Último acceso: Septiembre 2019].
- [10] «Canalys,» 15 Noviembre 2018. [En línea]. Disponible: www.canalys.com/newsroom/amazon-reclaims-top-spot-in-smart-speaker-market-in-q3-2018. [Último acceso: Septiembre 2019].
- [11] «Google Home,» [En línea]. Disponible: www.store.google.com/product/google_home. [Último acceso: Septiembre 2019].

- [12] «Amazon,» [En línea]. Disponible: www.amazon.es/Amazon-Echo-Altavoz-Inteligente-Alexa/dp/B079PFKDZC. [Último acceso: Septiembre 2019].
- [13] S. Theodoridis, *Machine Learning, A Bayesian and Optimizarion Perspective*, Elsevier, 2015.
- [14] «Mathworks,» [En línea]. Disponible: es.mathworks.com/discovery/machine-learning.html. [Último acceso: Septiembre 2019].
- [15] S. E. Everywhere, «Stanford University,» [En línea]. Disponible: see.stanford.edu/Course/CS229. [Último acceso: Septiembre 2019].
- [16] F. T. Liu, K. M. Ting y Z.-H. Zhou, «Departamento de Ciencia y Tecnología de la Computación, Universidad de Nanjing,» [En línea]. Disponible: cs.nju.edu.cn/zhoush/zhoush.files/publication/icdm08b.pdf?q=isolation-forest. [Último acceso: Septiembre 2019].
- [17] Z. Ding y M. Fei, «An Anomaly Detection Approach Based on Isolation Forest Algorithm for Streaming Data using Sliding Window,» *IFAC Proceedings Volumes*, vol. 40, nº 20, pp. 12-17, 2013.
- [18] Google, «TensorFlow,» [En línea]. Disponible: www.tensorflow.org/guide. [Último acceso: Septiembre 2019].
- [19] «Keras,» [En línea]. Disponible: www.keras.io. [Último acceso: Septiembre 2019].
- [20] «Scikit-learn,» [En línea]. Disponible: www.scikit-learn.org/stable/documentation.html. [Último acceso: Septiembre 2019].
- [21] «Dialogflow,» [En línea]. Disponible: www.dialogflow.com/docs. [Último acceso: Septiembre 2019].
- [22] «Firebase,» [En línea]. Disponible: www.firebase.google.com/docs/?hl=es-419. [Último acceso: Septiembre 2019].
- [23] «Raspberry Pi,» [En línea]. Disponible: www.raspberrypi.org. [Último acceso: Septiembre 2019].
- [24] A. Matin Dimas, «Fundación Alzheimer España,» [En línea]. Disponible: www.alzfae.org/fundacion/459/teleasistencia-que-es-en-que-consiste-como-contratarlo. [Último acceso: Septiembre 2019].
- [25] Ministerio de trabajo y asuntos sociales. Gobierno de España, *Atención a las personas en situación de dependencia en España (libro blanco)*, Madrid, 2004.
- [26] J. Garcés, S. Carretero, F. Ródenas y C. Alemán, «A review of programs to aliviate the burden of informal caregivers of dependent persons,» *Archives of Gerontology and Geriatrics*, vol. 50, nº 3, pp. 245-259, 2010.

- [27] I. N. d. Estadística, «INE,» Octubre 2009. [En línea]. Disponible: www.ine.es/revistas/cifraine/1009.pdf. [Último acceso: Septiembre 2019].
- [28] Mouser Electronics, Inc, «Mouser,» [En línea]. Disponible: www.mouser.com/ds/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf. [Último acceso: Septiembre 2019].
- [29] «Epitrán,» [En línea]. Disponible: www.epitrán.it/ebayDrive/datasheet/44.pdf. [Último acceso: Septiembre 2019].
- [30] johnmyleswhite, «Github,» [En línea]. Disponible: raw.githubusercontent.com/johnmyleswhite/room_temperatures/master/temperature_s.csv. [Último acceso: Septiembre 2019].
- [31] endoplasmic, «Github,» [En línea]. Disponible: www.github.com/endoplasmic/google-assistant. [Último acceso: Septiembre 2019].
- [32] «Nodemailer,» [En línea]. Disponible: www.nodemailer.com/about/. [Último acceso: Septiembre 2019].
- [33] J. d. Estado, «Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales,» *Boletín Oficial del Estado*, vol. I, nº 294, p. 119788, 2018.
- [34] P. E. y. C. d. I. U. Europea, «Reglamento general de protección de datos,» *Diario Oficial de la Unión Europea*, vol. I, p. 119, 2016.
- [35] J. d. Estado, «Ley 17/2001, de 7 de diciembre, de Marcas,» *Boletín Oficial del Estado - Legislación Consolidada*, nº 294, 2001.
- [36] K. Wiggers, «Venturebeat,» 14 Abril 2019. [En línea]. Disponible: www.venturebeat.com/2019/04/14/canalys-200-million-smart-speakers-will-be-sold-by-2019/. [Último acceso: Septiembre 2019].
- [37] G. Moreno, «Statista,» 6 Junio 2017. [En línea]. Disponible: es.statista.com/grafico/9676/apple-se-apunta-al-mercado-de-los-altavoces-inteligentes/. [Último acceso: Septiembre 2019].
- [38] A. Nijholt, «Google home: Experience, support and re-experience,» *Science Direct*, vol. 178, nº 3, pp. 612-630, 2018.
- [39] C. Wilson, «Benefits and risks of smart home technologies,» *Energy Policy*, vol. 103, pp. 72-83, 2017.
- [40] J. L. Zeni Montenegro, C. André da Costa y R. da Rosa Roghi, «Survey of conversational agents in health,» *Expert Systems with Applications*, vol. 129, pp. 56-57, 2018.
- [41] D. Griol Barres, «Desarrollo y evaluación de diferentes metodologías para la gestión automática del diálogo,» Noviembre 2007.

