



Escuela Técnica Superior de Ingeniería Informática y de Telecomunicaciones

Máster Universitario en Ingeniería Informática

Curso 2022/2023

MONITORIZACIÓN DE MAYORES CON DISPOSITIVOS DOMÓTICOS PARA DETECTAR DETERIORO COGNITIVO Y RIESGOS EN CASA

Autor

Álvaro de la Flor Bonilla

Tutores

María José Rodríguez Fórtiz

Miguel J. Hornos Barranco

RESUMEN

La detección temprana del deterioro cognitivo es un aspecto vital para tratarlo exitosamente, debido a que hoy en día podemos ser capaces de reducir la velocidad con la que avanza este deterioro, incluso llegar a “congelar” sus síntomas. Sin embargo, detectar las enfermedades relacionadas con el deterioro cognitivo puede ser extremadamente complicado, debido a su aparición “silenciosa” y a que cada vez más personas viven solas, sin la vigilancia de sus allegados. Esto hace que sea muy complicado de detectar hasta que no se encuentran en un estado avanzado.

En el contexto de nuestro proyecto, presentamos la domótica como una solución innovadora para detectar de forma temprana casos de deterioro cognitivo en personas mayores, de una forma automatizada y mucho antes incluso de que sus familiares más cercanos sean conscientes de ello. Básicamente, la domótica (aplicada en el campo de nuestro proyecto) consiste en la integración de la tecnología en los hogares para mejorar la calidad de vida de sus residentes, distribuyendo sensores inteligentes en el entorno del usuario que permiten detectar cambios en la actividad cotidiana de las personas, alertar a cuidadores o tutores, e incluso activar dispositivos para actuar en el propio hogar, tratando de evitar ciertos tipos de peligro.

En el presente trabajo se analizará cómo la domótica puede ayudar a detectar algún tipo de deterioro cognitivo u otras enfermedades relacionadas, basándose en una serie de sensores que analizarán comportamientos inusuales que se alejan de patrones preestablecidos, además de prevenir de forma activa posibles peligros relacionados que puedan ocurrir en relación con estos comportamientos extraños, tales como automatizaciones de luces para el guiado o alerta automática a cuidadores, entre otros.

ABSTRACT

Early detection of cognitive impairment is a vital aspect of successful treatment, because nowadays we may be able to slow down the rate at which cognitive impairment progresses, even “freezing” its symptoms. However, detecting diseases related to cognitive impairment can be extremely complicated, due to their “silent” onset and the fact that more and more people live alone, without the vigilance of those close to them. This makes it very difficult to detect until they are in an advanced stage.

In the context of our project, we present home automation as an innovative solution to detect early cases of cognitive impairment in older people, in an automated way and long before even their closest relatives are aware of it. Basically, home automation (applied in the field of our project) consists of the integration of technology in homes to improve the quality of life of its residents, distributing intelligent sensors in the user's environment that allow detecting changes in the daily activity of people, alerting caregivers or guardians, and even activating devices to act in the home itself, trying to avoid certain types of danger.

This project will analyze how home automation can help to detect some kind of cognitive impairment or other related diseases, based on a series of sensors that will analyze unusual behaviors that deviate from pre-established patterns, as well as actively prevent possible related dangers that may occur in relation to these strange behaviors, such as automation of lights for guidance or automatic alerts to caregivers, among others.

ÍNDICE DE LA MEMORIA DEL PROYECTO

1	Introducción	10
1.1	Justificación.....	10
1.2	Objetivos	11
1.2.1	General.....	11
1.2.2	Específicos	11
1.3	Motivación	12
1.4	Estructura del resto de la memoria	13
2	Estado del arte	14
2.1	Necesidades de monitorización y apoyo a personas mayores.....	14
2.2	Inicios de la domótica	16
2.3	Monitorización en el hogar.....	20
2.3.1	Etapla inicial: Introducción de sistemas de seguridad básicos y tecnologías de monitorización en el hogar	20
2.3.2	Avance en la domótica: Integración y automatización inteligente en el hogar	21
2.3.3	Monitorización inteligente: Hacia una monitorización personalizada y proactiva.....	22
2.3.4	Monitorización personal.....	23
2.4	API (<i>Application Programming Interfaces</i>)	29
2.4.1	Google Homegraph API.....	32
2.4.2	Amazon Alexa	35
2.4.3	Tuya API.....	37
2.4.4	Tabla comparativa de API.....	39
2.5	Protocolos de comunicación.....	40
2.6	Unidades de control: microcontroladores y microprocesadores.....	41
2.7	MQTT vs API.....	42
2.7.1	Diferencias	42
2.7.2	Beneficios de una API	43
2.8	Domótica actual. Estallido de los altavoces inteligentes	44

2.9	Dispositivos domóticos	48
2.9.1	Sensores de movimiento	48
2.9.2	Sensores de temperatura y humedad	52
2.9.3	Sensores lumínicos	56
2.9.4	Sensores adicionales.....	59
2.9.5	Actuadores	60
2.9.6	Pulseras inteligentes.....	63
2.10	Trabajos Relacionados	67
3	Propuesta.....	70
3.1	Descripción	70
3.1.1	¿Qué innovación aporta?.....	72
3.1.2	¿Qué se propone hacer?.....	73
3.2	Metodología	74
3.3	Desarrollo.....	75
3.3.1	Iteración 1: Microservicio “sensors”	75
3.3.2	Iteración 2: Microservicio “analyzer”	87
3.3.3	Iteración 3: Microservicio “execution”	95
3.3.4	Manual de uso	97
3.4	Prototipado de la aplicación móvil	102
3.5	Planificación temporal	104
3.6	Presupuesto del desarrollo	109
3.6.1	Coste de personal	109
3.6.2	Costes de material.....	110
4	Conclusiones y trabajos futuros	111
4.1	Conclusiones.....	111
4.2	Posibles futuras mejoras en funcionalidad y tecnologías	112
5	Referencias.....	114

ÍNDICE DE FIGURAS

Figura 2.1 Interfaz gráfica de Google Home (Google).....	30
Figura 2.2 Flujo de datos en Homegraph (Smith, 2020).....	33
Figura 2.3 Alexa Discovery Response	36
Figura 2.4 Alexa.PowerController	37
Figura 2.5 Cómo funciona una skill de Alexa	46
Figura 2.6 Arquitectura de interacción entre sujeto y altavoz propuesta para Alexa.....	47
Figura 2.7 Sensor MOES.....	48
Figura 2.8 Precio MOES Sensor	49
Figura 2.9 Sensor Yueyang	49
Figura 2.10 Precio Yueyang Sensor	50
Figura 2.11 Sensor Splenssy	50
Figura 2.12 Precio Splenssy Sensor	51
Figura 2.13 Sensor PHOVOLT	51
Figura 2.14 Precio PHOVOLT Sensor	52
Figura 2.15 Phovolt Sensor de termómetro.....	53
Figura 2.16 Precio Phovolt temperatura.....	53
Figura 2.17 Molczov Sensor termómetro	54
Figura 2.18 Precio Molczov Sensor	54
Figura 2.19 Fengchuang Sensor termómetro.....	55
Figura 2.20 Precio Fengchuan temperatura.....	55
Figura 2.21 Ailgely Termómetro higrómetro	55
Figura 2.22 Precio Ailgely temperatura	56
Figura 2.23 Kokonm sensor de luz.....	57
Figura 2.24 Precio Kokonm.....	57
Figura 2.25 Lechnical sensor de luz.....	57
Figura 2.26 Precio Lechnical.....	58
Figura 2.27 Juwaacoo sensor lumínico.....	58
Figura 2.28 Precio Juwaacoo.....	59

Figura 2.29 Relé inteligente MOES.....	60
Figura 2.30 Precio relé MOES	60
Figura 2.31 Relé proAMEDEN	60
Figura 2.32 Precio relé proAMEDEN	61
Figura 2.33 Interruptor de sensor táctil BSEED.....	61
Figura 2.34 Precio interruptor BSEED	62
Figura 2.35 Tira LED FACTORLED	62
Figura 2.36 Precio tira LED FACTORLED	63
Figura 2.37 Apple Watch Series 8.....	65
Figura 2.38 Samsung Galaxy Watch 5.....	65
Figura 2.39 Empática E4	66
Figura 2.40 Diagrama de la aplicación del proyecto de Delgado Sanchis (2016)	67
Figura 2.41 Propuesta de hogar inteligente	68
Figura 3.1 Diagrama de la arquitectura del sistema propuesto	71
Figura 3.2 IoT Platform Dashboard	76
Figura 3.3 Credenciales de la aplicación.....	76
Figura 3.4 Dispositivos virtuales disponibles.....	77
Figura 3.5 UX de Smart Industry.....	78
Figura 3.6 Llamada a la API Tuya.....	79
Figura 3.7 Controlador TuyaDeviceController	80
Figura 3.8 Controlador DeviceController	81
Figura 3.9 Uso de DeviceController	82
Figura 3.10 Parametrización del controlador.....	83
Figura 3.11 MongoDB vs Redis	85
Figura 3.12 Ejemplo de key compartida en redis	86
Figura 3.13 Controlador microservicio redis-gateway RedisController	86
Figura 3.14 Diagrama de flujo del primer microservicio.....	87
Figura 3.15 Estructura del proyecto	88
Figura 3.16 Implementación filtro DisableSensorsMovementAlertFilter.....	89

Figura 3.17 Creación de la alerta DisableSensorsMovementAlert.....	89
Figura 3.18 Configuración de valores iniciales para el análisis de eventos	90
Figura 3.19 Modelado de una alerta y su tipología.....	93
Figura 3.20 Implementación del analizador de signals en AnalyzeService.....	94
Figura 3.21 Diagrama de flujo del segundo microservicio	94
Figura 3.22 Ejecución del microservicio Execution	96
Figura 3.23 Conclusiones del reporte	96
Figura 3.24 Diagrama de flujo del tercer microservicio.....	97
Figura 3.25 Healthcheck del microservicio Sensors.....	98
Figura 3.26 Logs para alertas de tipo AletType.ACTION	100
Figura 3.27 Ciclo normal de la aplicación usando Postman	101
Figura 3.28 Flujo de inicio de la aplicación.....	102
Figura 3.29 Funcionamiento principal de la aplicación	103
Figura 3.30 Personalización de los sensores.....	104
Figura 3.31 Diagrama de Gantt.....	106

ÍNDICE DE TABLAS

Tabla 1 – Comparativa entre API estudiadas	40
Tabla 2 – Comparativa de sensores de movimiento	52
Tabla 3 – Comparativa entre sensores de temperatura y humedad	56
Tabla 4 – Comparativa entre sensores de luminosidad.....	59
Tabla 5 – Datos proporcionados por el Samsung Watch 5	64
Tabla 6 – Comparativa entre trabajos relacionados	69
Tabla 7 – Relación de alertas, eventos de activación y fuente de los datos	89

1 INTRODUCCIÓN

1.1 Justificación

Una de las principales características de la sociedad humana es la forma de relacionarnos entre nosotros. La sociedad actual vive cambios constantes que impactan significativamente en nuestras interacciones, incluso en las estructuras familiares.

Poniendo foco en la etapa de la vejez, la principal tendencia observable durante los últimos años es que **cada vez más personas mayores viven solas** y su **cuidado se convierte en un total desafío para sus allegados**, con la dificultad añadida de que, a medida que envejecemos, cuerpo y mente experimentan cambios que afectan directamente a nuestra capacidad para realizar las tareas cotidianas, incluso al cuidado de nosotros mismos.

No tener la posibilidad de llevar un seguimiento continuo del estado de las personas mayores puede suponer un **riesgo** en relación con **no detectar problemas a tiempo**. A nivel personal, detectar los cambios que tenemos nosotros mismos es complicado y más aún cuando se tratan de problemas relacionados con **deterioros cognitivos**, sin dejar a un lado problemas comunes, como caídas o algún tipo de **situación peligrosa** similar.

La alternativa ideal, en el caso de que una familia no se pueda hacerse cargo de sus allegados es la contratación de profesionales que se hagan cargo de ellos, pero en la mayoría de las ocasiones, por razones económicas, es imposible optar a ello. Otra opción sería apostar por la tecnología, pero **en la actualidad no hay sistemas especializados** que tengan **costes asequibles** para un público general.

Lo que este proyecto pretende conseguir es dar solución a este problema. Sustituir (o incluso complementar) mediante una propuesta basada en elementos tecnológicos la labor de seguimiento que haría un familiar o profesional, pero **manteniendo unos estándares económicos asequibles** que permitan que sea accesible para cualquier familia. Realizar una **monitorización continua** que se anticipe a las anomalías detectadas, e incluso reaccione de forma activa cuando las detecte.

Por anomalía nos referimos a cualquier tipo de problema detectado tras el análisis de los datos obtenidos de la monitorización, ya sean posibles casos de deterioro cognitivo, o cualquier otro tipo de situación que puede desencadenar algún tipo de problema o peligro.

La tecnología aplicada a la monitorización de mayores permitirá solventar el desafío que representa el cuidado de estas personas que, por cualquier tipo de circunstancias, tienen que vivir solas, a pesar de necesitar un cuidado más cercano. Mediante la evaluación continua y la provisión de recursos adecuados, se pretende **garantizar la seguridad**, el **bienestar** y la **calidad de vida** de esta población en cierto modo vulnerable.

Es fundamental asegurar las necesidades de las personas que viven solas, para garantizar una sociedad más inclusiva y solidaria. Si no pueden tener un seguimiento personal, debemos aportar soluciones que intenten mitigar las diferencias sociales.

1.2 Objetivos

1.2.1 General

El objetivo primordial de este proyecto es desarrollar una solución basada en la tecnología domótica **para detectar y prevenir el deterioro cognitivo** en personas mayores y/o en aquellas con necesidades especiales. Se pretende utilizar una gran cantidad de sensores domóticos de diversa índole, que permitan recopilar todos los datos posibles sobre las actividades y comportamientos habituales que realizan las personas que viven solas. También se pretende que el sistema que se desarrolle sea capaz de **identificar** si la persona mayor ha tenido un accidente y/o **necesita ayuda**.

Por tanto, se desarrollará una plataforma de monitoreo inteligente encargada de recopilar datos de todos los sensores. Esta plataforma será capaz de procesar y analizar los datos de forma eficiente y precisa.

Tras la recopilación de los datos, el principal objetivo es ser capaz de **identificar patrones y comportamientos extraños que puedan suponer peligro para el usuario**. Se utilizarán técnicas de análisis de datos basadas en reglas fijas para detectar estas anomalías.

En los casos donde se identifiquen situaciones peligrosas, se podrá **alertar a cuidadores o profesionales** enviando notificaciones a través de una aplicación móvil o servicio de mensajería, para que se puedan tomar medidas inmediatas en caso de emergencia. Por otro lado, también se accionará actuadores para la seguridad activa, en caso de necesidad. Por ejemplo, se podría construir un camino de luces que guíe al usuario a un lugar seguro.

1.2.2 Específicos

Existen varias metas concretas que se pretenden alcanzar para lograr el objetivo general de este proyecto.

- **Analizar el estado de la domótica actual y su evolución a lo largo de los años** para comprender los cambios ocurridos desde sus inicios hasta la actualidad.
- **Investigar el mercado actual de dispositivos domóticos** para seleccionar los más económicos y adecuados en función del sistema de procesamiento que se utilizará.
- **Desarrollar un sistema de procesamiento de datos** para recopilar, almacenar y analizar los datos en tiempo real, incluyendo un algoritmo de detección de patrones anormales de los datos de los sensores, para identificar situaciones de riesgo o que puedan ser un signo de deterioro cognitivo.
- **Configurar los diferentes sensores** para que puedan ser controlados mediante las API más adecuada de *Tuya*, estableciendo reglas y umbrales de detección para generar y enviar alertas en caso de riesgo a las personas pertinentes.

- **Facilitar la interoperabilidad entre los sensores domóticos y el sistema que gestione los datos**, para justificar así el uso de controladores. Los controladores permiten establecer una capa de abstracción que se encarga de traducir y adaptar los datos provenientes de los sensores al formato requerido por la aplicación que vamos a desarrollar.
- **Diseñar una interfaz de usuario** de una aplicación *Android* para visualizar los datos de los sensores y alertas generadas por el sistema.
- **Organizar un entorno de pruebas** para mejorar la precisión y evitar falsas alarmas. Evaluar el sistema será primordial para que la detección temprana de situaciones de deterioro cognitivo o de riesgo para las personas mayores permita reducir el número de accidentes en el hogar.
- **Realizar una comparativa de los sistemas de monitorización existentes** para llegar a la conclusión de cuáles son los dispositivos que nos permitan realizar el mejor seguimiento posible de los usuarios al precio más asequible.

1.3 Motivación

Desde hace años, cada vez más personas mayores viven solas, sin contar con la asistencia y el apoyo de otros. Ya sea por motivos personales o laborales, muchas personas se tienen que enfrentar a la dura situación de separarse de sus familias y no poder aportar la ayuda de la que son conscientes que les hace falta.

La tecnología nos ayuda, nos hace crecer, nos enseña y nos hace la vida un poco más fácil. Sin embargo, en el contexto de las personas mayores, en ocasiones parecen vivir una realidad aparte. En la sociedad de consumo en la que vivimos, se prioriza mucho más el rendimiento económico de un producto o servicio por encima de las posibilidades de mejora de calidad de vida, bienestar o seguridad que pueda aportar. Como resultado, debido a que comercialmente no suele ser muy rentable, la mayoría de empresas no suelen tener una línea de desarrollo de productos para personas mayores.

En este punto es donde surge mi motivación. La tecnología de dispositivos domóticos puede marcar la diferencia en la vida de estas personas vulnerables. Podemos crear mecanismos de seguimiento que permitan detectar problemas cognitivos y situaciones de riesgo de manera temprana.

Desde siempre, he tenido un especial interés por los dispositivos domóticos. Aspectos como luces, persianas, sistemas de riego y climatización han sido completamente automatizados en mi hogar y, con la implantación de rutinas, han hecho que mi vida sea mucho más fácil y cómoda. Sin embargo, nunca he apreciado el potencial que puede tener esta tecnología para las personas mayores, ese gran colectivo olvidado.

Quiero contribuir a cerrar la brecha tecnológica y crear conciencia sobre las ventajas y beneficios que estos dispositivos pueden proporcionar, dado que creo que podemos hacer

que la tecnología sea una aliada en el cuidado y bienestar de las personas mayores, que merecen todo nuestro apoyo y atención.

1.4 Estructura del resto de la memoria

En la siguiente sección se realizará un análisis del estado del arte en el que se hará un recorrido de la evolución de la domótica desde su comienzo hasta nuestros días, tomando como referencia la influencia que ha tenido los altavoces inteligentes en su desarrollo y expansión. Durante este estudio, se profundizará en la evolución de la sociedad y cómo a lo largo de los años los cambios en ella han propiciado grandes evoluciones en el sector tecnológico. Además, se profundizará con más detalle en las diferentes opciones de cómo es posible acceder y tomar el control de los diferentes dispositivos domóticos.

Posteriormente, se realizará una propuesta de sistema de monitorización que pretendemos diseñar e implementar, en la que explicaremos en detalle cómo funciona y qué problemas solventa. Tras esta propuesta se presentarán los diferentes casos de uso del sistema propuesto.

Por último, se finalizará el documento con las conclusiones y posibles mejoras aplicables a nuestra propuesta.

2 ESTADO DEL ARTE

En esta sección realizaremos un recorrido por las diferentes etapas que ha atravesado la domótica en nuestros hogares y las causas que han motivado un uso tan habitual como el que se realiza actualmente. Se realizará además una explicación del funcionamiento de las diferentes tecnologías usadas y su evolución a lo largo de los años. Comenzaremos justificando porqué la domótica puede ser útil para monitorizar a personas mayores.

2.1 Necesidades de monitorización y apoyo a personas mayores

La sociedad humana vive en un constante e incansable cambio que a medida de los años se acentúa cada vez más. No solo no referimos a nivel de productos que surgen y crean una necesidad que antes ni imaginábamos, esta evolución está afectando a la propia forma de relacionarnos con algo tan personal como la familia.

La tecnología une y acorta fronteras, pero ¿realmente es cierto? En los últimos años, a la par del crecimiento de los dispositivos inteligentes, se ha disparado un dato muy preocupante, y es que **cada vez hay más personas mayores que viven solas**, a pesar de necesitar ayuda en su día a día. Existen varias razones que explican este auge:

- **Cambios en las estructuras familiares:** en muchos países, las estructuras familiares han cambiado en las últimas décadas, con menos personas viviendo en hogares multigeneracionales y más personas viviendo solas. Esto significa que las personas mayores pueden tener menos opciones de vivienda y reciben menos cuidado de sus seres queridos. Actualmente no es nada extraño que existan casos en los que los hijos de un matrimonio vivan a cientos de kilómetros de sus padres, incluso en otros países.
- **Mayor esperanza de vida:** las personas están viviendo más tiempo que antes, lo que significa que hay más personas mayores que necesitan atención y cuidado. A menudo, esto significa que estas personas mayores viven solas durante más tiempo. De hecho, existen casos en que los propios hijos que ya son mayores (edades superiores a 75 años) siguen cuidando a sus padres mucho más mayores. Además, vivir más años suele relacionarse con la aparición de más enfermedades, lo que requiere más cuidados.
- **Movilidad y ubicación geográfica:** la movilidad geográfica también puede ser un factor. Muchas personas mayores tienen hijos y otros familiares que viven lejos de ellos, lo que hace que sea más difícil recibir ayuda y cuidado. No pueden permitirse dejar su trabajo para cuidarlos. Suelen ser casos en los que siendo aún jóvenes se mudaron lejos de su ciudad natal y no se plantean volver una vez han rehecho su vida en otra localidad lejana. El modelo de sociedad en el que las sucesiones de generaciones viven en la misma localidad cada vez es menos usual y se está perdiendo.

- **Recursos limitados:** Las personas mayores con recursos financieros limitados pueden tener dificultades para encontrar opciones de vivienda y atención asequibles. A menudo, esto significa que algunos de ellos tienen que depender de ellos mismos, de servicios gubernamentales o de organizaciones benéficas para obtener ayuda. En España, por ejemplo, se puede llegar a subvencionar residencias para este tipo de personas, pero para ello deben pasar por procesos de evaluación que en ocasiones se alargan demasiado en el tiempo para su aprobación e incluyen una gran cantidad de requisitos.

A medida que envejecemos, el cuerpo y la mente pueden experimentar **cambios que afecten a la capacidad para realizar actividades diarias** y cuidar de sí mismos. Algunos de estos cambios, entre los más comunes hoy en día, pueden ser:

- **Problemas de salud:** las personas mayores pueden experimentar problemas de salud, como enfermedades crónicas y discapacidades físicas, y problemas de salud mental, que pueden dificultar la realización de actividades de la vida diarias, como la alimentación, el aseo y la movilidad.
- **Cambios cognitivos:** a medida que envejecemos, nuestro cerebro experimenta un deterioro cognitivo que pueden afectar a la memoria, la toma de decisiones y la capacidad de pensar con claridad. Estos cambios pueden hacer que sea difícil para las personas mayores realizar actividades cotidianas y tomar decisiones importantes.
- **Pérdida de seres queridos:** con el paso de los años, es más probable que experimentemos la pérdida de seres queridos, lo que puede afectar nuestra salud mental y emocional. Esto puede hacer que las personas mayores necesiten apoyo emocional y psicológico para hacer frente a la pérdida.
- **Cambios en la movilidad:** A medida que envejecemos, también podemos experimentar un deterioro físico que implique cambios en la movilidad, como la disminución de la fuerza muscular y la flexibilidad. Esto puede hacer que sea más difícil realizar actividades diarias, como subir escaleras o levantar objetos pesados.

Todos los problemas nombrados con anterioridad resultan de especial interés, pero para este proyecto se ha puesto especial énfasis en lo relativo al **deterioro cognitivo**. Este deterioro puede ser difícil de detectar en personas mayores que viven solas, porque pueden no tener a alguien que esté cerca de ellos para notar los cambios en su comportamiento y en su capacidad de realizar actividades diarias. Además, muchas personas mayores pueden tratar de ocultar o minimizar los síntomas de cambios cognitivos, como la pérdida de memoria o la confusión, por vergüenza o miedo a ser estigmatizados. Es común la negación en casos como éstos, ya que estas personas suelen auto-convencerse de que solo es un problema temporal derivado del cansancio o el estrés. En su opinión, admitir este problema podría conllevar perder independencia en la toma de decisiones importantes (Fundación Pasqual Maragall, 2022).

Los cambios cognitivos pueden ser especialmente peligrosos en personas mayores que viven solas, porque pueden afectar su capacidad para cuidar de sí mismos y realizar actividades diarias. Por ejemplo, pueden tener dificultades para recordar cuándo tomar sus medicamentos o cuándo tienen una cita con el médico. También pueden tener problemas para realizar tareas básicas como cocinar, limpiar y manejar el dinero.

Además, los cambios cognitivos pueden **aumentar el riesgo de caídas, accidentes y otros incidentes que pueden ser peligrosos** para las personas mayores que viven solas. Por ejemplo, pueden olvidar apagar la estufa o el horno después de cocinar, lo que podría provocar un incendio. También pueden tener dificultades para recordar cómo usar los dispositivos de seguridad del hogar, como cerraduras de puertas y ventanas, lo que podría aumentar el riesgo de robos y otras amenazas para su seguridad.

Lo que queremos expresar es que tanto los cambios cognitivos como los físicos suelen ser **difíciles de detectar en personas mayores que viven solas** y pueden ser especialmente peligrosos porque pueden afectar su capacidad para cuidar de sí mismos y realizar actividades diarias. Es importante que las personas mayores que viven solas sean **evaluadas regularmente** por un profesional y que se les brinde apoyo y recursos adecuados para garantizar su seguridad y bienestar.

Aunque en la siguiente sección se detallará aún más, el principal foco de este estudio es ser capaz de identificar estos casos de deterioro cognitivo que hemos comentado con ayuda de la domótica, y realizar una serie de actuaciones a partir de los datos recopilados.

2.2 Inicios de la domótica

La domótica, en su sentido más amplio, se refiere a la integración de tecnologías y sistemas en el hogar para automatizar y controlar diversas funciones, como la iluminación, la climatización, la seguridad, los electrodomésticos y otros dispositivos. Esta tecnología tiene como objetivo mejorar la comodidad, la eficiencia energética, la seguridad y la calidad de vida en general.

En una de las primeras publicaciones españolas sobre este tema, titulada “*La mutación de la vivienda*” (Chaparro, 2003), se hace una revisión muy interesante sobre el tema. Ya el título de la publicación, respecto a nuestro punto de vista, es un completo acierto, teniendo en cuenta el momento que vivimos hoy en día.

Actualmente, no podemos tener la misma concepción de vivienda que en la década de los noventa o dos mil, hemos vivido la **masificación y estandarización de la tecnología** como algo normal, y aún más si cabe en los últimos años, con el estallido de una pandemia mundial. En un mundo más abierto e interconectado, apostamos cada vez más por aislarnos. Nuestra oficina ahora está a tres pasos de nuestra habitación, las plataformas “*streaming*” como Netflix han traído las salas de cine al salón de nuestra casa e incluso hacer la compra lo tenemos a golpe de un clic. Ocio, trabajo o descanso, ahora todo se concentra en la misma estancia.

En la publicación de Chaparro (2003) se indica cómo la propia evolución de la tecnología está induciendo cambios en la función de la vivienda tal y como la conocemos hasta ahora. La domótica no es más que el estallido de una necesidad evidente. El foco de nuestra vida gira más que nunca en nuestro hogar y consecuentemente el mercado y desarrollo de tecnologías no es ajeno a todo esto. En los últimos años más que nunca están apareciendo productos que no solo dan cabida a solucionar problemas, sino que generan necesidades que no éramos conscientes de ellas. Uno de estos productos, son los altavoces inteligentes, de los que más adelante hablaremos.

En la propia revisión anterior, también se hace referencia al proyecto **de diálogo en lenguaje natural** de la Universidad de Sevilla (Quesada Moreno et al., 2001). Es sorprendente cómo en ese proyecto, ya en el año 2001, se hace (de forma bastante acertada) una aproximación de un sistema de reconocimiento de voz asociada a productos domóticos, en este caso mostrando ejemplos del control de iluminación de distintas estancias de un domicilio.

Realmente, la aproximación es tan real que es igual a lo que tenemos hoy en día con altavoces inteligentes como “Google Home” o “Alexa” (siendo éstos, dos de los más extendidos). Básicamente se establece un conjunto de reglas en base a diálogos estáticos, más sencillos que los algoritmos de inteligencia artificial que se utilizan hoy en día. Tanto Google como Amazon de igual forma se han encargado de desarrollar sus propios sistemas de reconocimiento del lenguaje natural, tal y como se propone en el artículo anterior. No somos conscientes de la capacidad de recopilación de datos que poseen estas dos compañías la cual les ha ayudado, entre otras cosas, a llegar a alcanzar un nivel muy avanzado de comprensión de órdenes, que no se basan reconocer o responder sentencias simples y fijas, sino que pueden llegar a mantener una conversación habitual con una persona. Además, ambas compañías se han encargado de desarrollar un sistema público y abierto para el control de lo que hoy en día llamamos, internet de las cosas o IoT de sus siglas en inglés, para fortalecer el uso de sus plataformas en coalición con los altavoces inteligentes, en forma de API públicas (ver API) y accesibles para todo desarrollador interesado.

En el estudio de Chaparro (2003) que nombramos con anterioridad, se nombran multitud de protocolos entre el usuario final y los accesorios domóticos, entendiendo estos como el conjunto de sensores y actuadores inteligentes. Sin embargo, a diferencia de esta propuesta, lo que se ha logrado crear hoy en día es una **estandarización**, con unas API sencillas de utilizar y muy baratas de implementar. Con ello queremos decir que la inmensa mayoría de los desarrolladores han dejado atrás su preocupación por el desarrollo de tecnologías de procesamiento de lenguaje natural e interconexión entre dispositivos inteligentes, el principal el foco actual es **desarrollar el mejor y más atractivo producto** (con sensores, actuadores...) que pueda ser utilizado por los estándares propuestos por grandes empresas como Google y Amazon. En definitiva, el usuario genérico quiere comprar un altavoz, buscar un accesorio compatible (ya sea una bombilla, termostato, enchufe...), seleccionar la red WiFi y que comience a formar parte del ecosistema inteligente de su hogar que quiere construir o del que ya dispone y quiere ampliarlo.

Lo principal que queremos señalar con el resumen anterior es que la compra de un altavoz inteligente (en gran parte de los casos) determinará el producto *IoT* que se comprará y no al contrario, ya que el usuario que quiere apostar por la domótica en su hogar necesita configurar un ecosistema compatible, y actualmente éste vendrá regido por el altavoz (en la mayoría de las ocasiones).

Durante la primera década de los 2000, el desarrollo de la domótica comenzó a despuntar, pero dando un enfoque más centrado al **ahorro energético**. El uso de dispositivos inteligentes para la gestión eficiente fue prácticamente el **único uso** que se le daba a la domótica y consecuentemente la monitorización de personas mayores para su cuidado asistencial ni si quiera se consideraba. En una sociedad donde la **conexión a internet** aún **no era masiva** en todos los hogares resultaba muy difícil invertir y centrar el presupuesto en este aspecto. Por tanto, el principal problema observable durante estos años (para que no se enfocara la domótica en el control asistencial) fue que el público objetivo para esta tecnología se centró en dos grupos diferenciados:

Por un lado, nos encontramos con **grandes comercios y oficinas** que esperaban que su inversión en domótica tuviese como fruto **ahorros en costes** relacionados con la adaptación al entorno como iluminación automática y la regulación de temperatura (eficiencia energética).

Por otro lado, aunque en una medida mucho más inferior, estarían los **entusiastas de la tecnología**, a los que no les importaba realizar un desembolso importante para la fecha, y cuyo objetivo no era el ahorro, sino más bien su **pasión y afición** por los nuevos **avances** y productos **tecnológicos**.

Por más que nos hemos centrado en tratar de encontrar posibles soluciones domóticas para personas mayores con anterioridad al año 2014, son muy pocas las citas que pueden encontrarse, y la mayoría de ellas se focalizan en el control de iluminación, calefacción... alejándose de nuestra idea de control asistencial. Señalamos esta fecha con tanta exactitud porque fue cuando se presentó el primer altavoz inteligente (Amazon Alexa) y que propició un boom de la tecnología domótica.

La aparición de los altavoces inteligentes, como Amazon Echo (Alexa) y Google Home propició un aumento de la demanda de soluciones domóticas en gran medida debido a su capacidad para controlar múltiples dispositivos conectados en el hogar a través de comandos de voz y la integración con sus asistentes.

Antes de la aparición de estos altavoces, como ya bien hemos comentado anteriormente, la domótica era costosa, poco accesible y compleja, requiriendo cableado y dispositivos específicos para monitorización. Su instalación quedaba reducida a profesionales con sólidos conocimientos de programación, por lo que se reservaba a entornos lujosos o profesionales como empresas. Era una tecnología **no al alcance de todos**.

Gracias a los altavoces inteligentes, la situación cambió radicalmente, ya que permitían realizar una interacción mucho más fácil. La instalación de cualquier dispositivo se convirtió en algo trivial, no se necesitaba cableado y bastaba hacer uso de protocolos

como bluetooth, con lo que el sistema resultante muy cómodo de usar. Así, por ejemplo, con un simple “*Alexa, me voy*” podías apagar todas las luces de la casa. El usuario podría ordenar apagar las luces o dar cualquier orden similar al sistema. Estas nuevas funcionales para interactuar con los productos, además de la reducción en costes, produjo el *boom* de la tecnología.

Entre las pocas publicaciones que mencionaba anteriormente, es bastante interesante la propuesta presentada por Castro Morales et al. (2009). Parte de sus ideas encajan a la perfección con la asistencia a personas mayores tal y como la queremos plantear en nuestros días. Por ejemplo, una de las principales propuestas y que más interesante nos parece dentro de esta propuesta es la detección de camas vacías. Como se presenta en el proyecto, su idea principal es la de avisar al supervisor de ese turno de que el residente ha abandonado la cama durante el período de sueño. Sin embargo, no se pretende hacer nada más allá que activar una mera alarma. Con esto queremos decir que los datos recopilados no se tienen en cuenta para un futuro análisis de las rutinas del usuario (tendencias a la hora de despertarse, por ejemplo).

Hoy en día no son escasas las situaciones en las que personas mayores, dependen únicamente de sí mismas para su día a día. Es muy difícil en este tipo de ocasiones darse cuenta de nuestros cambios a medida que pasa el tiempo sin una opinión o punto de vista ajeno, ya que suelen ser muy leves los cambios en el tiempo y no somos conscientes de ellos. Por eso, sería ideal tener un **sistema** que sea **capaz de detectar comportamientos** que pudieran indicar el inicio de un proceso que en un futuro quizás desemboque en algún tipo de **deterioro cognitivo**, siendo éste el objetivo de la propuesta que se presenta en este TFM.

Por otro lado, queremos señalar que el **control del sueño es un aspecto fundamental**. La puesta en común de la detección de la ocupación de la cama, junto con otros sistemas de seguimiento, permitiría conocer los hábitos de los usuarios. El análisis de éstos, utilizando un análisis contrastado, incluso podría **prevenir de posibles problemas**. El buen sueño es primordial para la consolidación de la memoria y el aprendizaje, pero también es un pilar fundamental para el funcionamiento cognitivo. Dormir poco o que el sueño sea de mala calidad puede afectar negativamente a multitud de aspectos como el procesamiento de la información, el razonamiento, la memoria. Se ha demostrado que la **falta de sueño REM** (sueño profundo) puede **afectar negativamente a nuestro cerebro** en el sentido de que no se completa la eliminación de proteínas tóxicas en éste, lo cual puede derivar en enfermedades neurodegenerativas, como el **Alzheimer**. También se ha descubierto que la interrupción del sueño puede aumentar el nivel de *beta-amiloide*, proteína relacionada con el desarrollo del Alzheimer, como se indica en el estudio “*Privación de sueño y neurodegeneración*” (Sánchez Blanco, 2022).

Otro sistema propuesto bastante interesante en el estudio que nombramos anteriormente (Castro Morales et al., 2009) es un detector de caídas donde en esta ocasión utiliza el dispositivo “*Tunstall Fall Detector*” para ello. Sin embargo, una vez más, su finalidad (en el estudio que nombramos) está más destinada al aviso inmediato a un supervisor en una residencia. ¿Cómo podría aplicarse hoy en día? Con el desarrollo tecnológico con el que

se cuenta, situaciones como éstas podrían detectarse automáticamente mediante sensores, cámaras y sistemas de reconocimiento, y desencadenar rutinas estándar, como protocolos en los que el usuario tendrá que confirmar que no necesita ayuda, mientras que, en caso contrario, se activaría un protocolo de auxilio. Si el usuario comienza a actuar extrañamente (movimientos no habituales por la noche, por ejemplo) podríamos comenzar a alertar de manera activa. En caso de que se produzca una caída podríamos iniciarse un protocolo automatizado con dispositivos domóticos, como encender luces, alertar al responsable o cuidador, tranquilizar al usuario e incluso llamar a un profesional de la salud con imagen en tiempo real de lo que ha ocurrido.

En los siguientes apartados, realizaremos un especial hincapié en la evolución de los sistemas de monitorización, tanto personales como del hogar, los cuales, junto con el estallido de los altavoces inteligentes, propiciaron la consolidación de la domótica como la conocemos en nuestros días.

2.3 Monitorización en el hogar

La evolución de la monitorización personal de nuestros hogares y su integración con la domótica del hogar ha sido notable en las últimas décadas. Podemos señalar tres etapas principales: etapa principal, avance en la domótica y monitorización inteligente.

2.3.1 Etapa inicial: Introducción de sistemas de seguridad básicos y tecnologías de monitorización en el hogar

En los primeros pasos de la evolución de la monitorización personal de los hogares, se implementaron sistemas de seguridad básicos con el objetivo de proteger la propiedad y sus habitantes. Estos sistemas se basaban en tecnologías rudimentarias y proporcionaban una capa de seguridad inicial. A continuación, se describen algunas de las tecnologías clave utilizadas en esta etapa:

- **Sistemas de seguridad básicos:** Los sistemas de seguridad iniciales consistían en dispositivos de alarma que se activaban en caso de detección de intrusiones en la propiedad. Estos sistemas utilizaban sensores magnéticos en las puertas y ventanas para detectar aperturas no autorizadas, activando una alarma sonora o enviando una señal a una central de seguridad. Prácticamente la mayoría consistía en poner alarmas en puertas y ventanas con alertas sonoras si se producía un acceso no identificado.
- **Sensores de localización puntual:** Para mejorar la detección de intrusiones, se introdujeron sensores de movimiento en áreas estratégicas de la propiedad. Estos sensores, basados en tecnología infrarroja o ultrasónica, eran capaces de detectar cambios en el patrón de movimiento en una zona determinada. Cuando se detectaba un movimiento inesperado, se generaba una señal de alarma. Para entenderlo un poco mejor, es muy similar a lo que se utiliza en la actualidad para el cierre de puertas automáticas en chocheras, detectando mediante una línea infrarroja punto a punto en el borde de la puerta que no sigue estando el vehículo en esa zona.

- **Cerraduras electrónicas:** Con el objetivo de reforzar la seguridad y proporcionar un mayor control de acceso, se comenzaron a utilizar cerraduras electrónicas en lugar de las cerraduras tradicionales. Estas cerraduras electrónicas podían ser controladas de forma remota mediante claves numéricas o tarjetas de acceso. Además, algunos modelos permitían la programación de horarios de acceso y la capacidad de registrar y auditar los eventos de apertura, las típicas credenciales de acceso que hoy en día seguimos utilizando en tornos en las entradas de edificios, por ejemplos.

Aunque en esta etapa inicial se hicieron avances significativos en la seguridad residencial, las tecnologías utilizadas eran limitadas en su alcance y capacidad. Los sistemas de seguridad eran independientes y no estaban integrados con otros dispositivos o sistemas en el hogar. Además, la monitorización era principalmente reactiva, es decir, se activaba una vez que se detectaba una intrusión o violación de seguridad.

2.3.2 Avance en la domótica: Integración y automatización inteligente en el hogar

La evolución de la monitorización personal en los hogares ha dado lugar a la segunda etapa, marcada por el avance en la domótica. En esta fase, se ha producido una integración más estrecha de dispositivos y sistemas para permitir un mayor control y automatización inteligente del hogar. La domótica ha abierto un amplio abanico de posibilidades para la monitorización personalizada y la gestión eficiente de los recursos. Las tecnologías y avances clave en esta etapa fueron:

- **Automatización del hogar:** la integración de dispositivos y sistemas permite la automatización de varias funciones en el hogar, como la iluminación, la temperatura, las persianas, los electrodomésticos, entre otros. Mediante la programación o el control remoto, los usuarios pueden establecer horarios o activar acciones basadas en eventos específicos. Por ejemplo, es posible configurar la iluminación para que se encienda automáticamente al llegar a casa o programar el termostato para ajustar la temperatura según la hora del día.
- **Control remoto:** el control remoto de los dispositivos del hogar ha adquirido una importancia significativa en esta etapa. Las aplicaciones móviles y las interfaces de usuario intuitivas permiten a los usuarios controlar y supervisar sus sistemas domésticos desde cualquier lugar mediante dispositivos como teléfonos inteligentes o tabletas. Además, la conectividad a Internet y la comunicación inalámbrica facilitan la gestión remota de los dispositivos, lo que proporciona una mayor comodidad y flexibilidad.
- **Sensores inteligentes:** se han introducido sensores más avanzados que mejoran la monitorización personalizada en el hogar. Estos sensores son capaces de detectar una amplia gama de variables, como la presencia de personas, cambios en la temperatura, la humedad, la calidad del aire y otros factores ambientales. La información recopilada por estos sensores se utiliza para tomar decisiones

automatizadas, optimizar el consumo energético y mejorar la comodidad en el hogar. Por ejemplo, los sensores de movimiento pueden activar la iluminación o el sistema de climatización cuando se detecta la presencia de alguien en una habitación.

2.3.3 Monitorización inteligente: Hacia una monitorización personalizada y proactiva

La evolución de la monitorización personal en los hogares ha dado lugar a la etapa de "*Monitorización inteligente*", en la que se han introducido tecnologías más avanzadas y soluciones innovadoras para proporcionar una monitorización personalizada y proactiva en el ámbito residencial. Esta etapa se centra en aprovechar la inteligencia artificial, el aprendizaje automático y la integración de datos para mejorar la seguridad, el bienestar y la eficiencia del hogar. A continuación, se describen las tecnologías y aplicaciones clave en esta fase:

- **Sistemas de videovigilancia avanzados:** Se han desarrollado sistemas de videovigilancia más sofisticados que van más allá de las cámaras de seguridad convencionales. Estos sistemas utilizan cámaras de alta definición y visión nocturna, y están equipados con análisis de video basados en inteligencia artificial. Estos análisis permiten la detección de objetos, reconocimiento facial, seguimiento de personas y detección de actividades inusuales. Además, se han implementado funciones de grabación en la nube, lo que garantiza el almacenamiento seguro de las imágenes y permite el acceso remoto a través de dispositivos conectados.

Ya en 2008 como ejemplo de este avance podemos señalar por ejemplo la publicación "*The CAS-PEAL Large-Scale Chinese Face Database and Baseline Evaluations*" (Gao et al., 2008). Una base de datos con más de 10000 imágenes que se utilizó inicialmente para desafíos y competencias de reconocimiento social. Pero lo que es realmente interesante es que desde esa época hasta hoy hemos sido capaces de desarrollar una tecnología que permite la identificación de personas en tiempo real. Hoy en día, técnicamente no es necesario ningún tipo de dispositivo de identificación físico para identificar a una persona, simplemente una captura de imagen analizada con un software específico puede usarse para reconocer a una persona. De hecho, en países como China se ha impuesto por ley la obligatoriedad de habilitar un sistema de reconocimiento facial en productos tecnológicos, según sus palabras textuales para "proteger los derechos e intereses legítimos de los ciudadanos en el ciberespacio" (de la Cal, 2019). Con esto se quiere decir que a partir de 2019 quien contrate nuevos servicios de telefonía móvil en China tendrá que pasar por un escaneo facial.

- **Sensores inteligentes de salud, bienestar y análisis del comportamiento:** Se han desarrollado sensores inteligentes para monitorizar la salud y el bienestar de los residentes en el hogar. Estos sensores suelen estar en dispositivos *wearables* y pueden medir parámetros fisiológicos como la frecuencia cardíaca, la presión arterial, los niveles de oxígeno en sangre y la actividad física. Los datos recopilados

por estos sensores se pueden analizar mediante algoritmos de aprendizaje automático para detectar patrones y anomalías, lo que permite el seguimiento continuo de la salud y la detección temprana de problemas.

Mediante el análisis de datos y algoritmos de aprendizaje automático, se pueden detectar patrones de comportamiento en el hogar. Esto permite la identificación de actividades anormales o situaciones de riesgo, como caídas o ausencia prolongada de actividad, especialmente en personas mayores o dependientes. Por ejemplo, los sensores de movimiento y las cámaras pueden detectar caídas y enviar alertas a los cuidadores o al personal médico para una respuesta rápida.

- **Integración con asistentes virtuales:** Los sistemas de domótica se han integrado con asistentes virtuales, como Amazon Alexa, Google Assistant o Apple Siri. Estos asistentes permiten controlar los dispositivos y sistemas del hogar mediante comandos de voz, lo que agiliza la interacción y facilita el control *hands-free*. Los usuarios pueden dar instrucciones verbales para encender o apagar luces, ajustar la temperatura, reproducir música y realizar otras acciones, lo que brinda una experiencia más intuitiva y cómoda en el hogar.

2.3.4 Monitorización personal

Los sistemas de monitorización personal han cobrado una gran relevancia en la actualidad, ya que permiten a las personas hacer un seguimiento de su salud y bienestar de manera autónoma y precisa. Algunos de los sensores inteligentes mencionados en la sección anterior permiten estudiar una cantidad enorme de parámetros fisiológicos. En esta sección pondremos un mayor énfasis en aquellos especialmente de interés para el cuidado de las personas mayores. Estos sistemas tienen un gran potencial en el cuidado de las personas mayores, ya que pueden ayudar a detectar problemas de salud y ofrecer recomendaciones para mejorar el bienestar y la calidad de vida. Entre los principales, podemos señalar los siguientes:

- **Sensores de actividad física**

Los sensores de actividad física pueden ser especialmente útiles para las personas mayores que tienen problemas de movilidad. Estos dispositivos pueden medir la cantidad de actividad física que realiza una persona y ofrecer recomendaciones para aumentarla, así como detectar patrones de actividad que puedan indicar problemas de salud.

- **Sensores de caídas**

Los sensores de caídas son dispositivos que se colocan en la ropa o en el cuerpo de una persona y que detectan cuándo esta se cae. En caso de una caída, el dispositivo puede enviar una alerta a un cuidador o a un servicio de emergencia para que se pueda prestar asistencia de manera inmediata.

Hoy en día, hay productos que ofrecen esta funcionalidad, como por ejemplo algunos modelos de Apple Watch¹, aunque existen infinidad de más ejemplos. En Tai et al., (2020), se hace una clara referencia a lo que se comentó anteriormente. Año tras año aumenta la población anciana y las cuestiones que son relativas al cuidado de personas mayores están adquiriendo un peso inevitable y de gran importancia ya que incluso puede representar un nicho de mercado poco explorado hoy en día a partir de la venta de dispositivos de monitorización. En concreto, en el estudio de Tai que nombramos, se hace un análisis exhaustivo del modelo HLPFN (High-Level Fuzzy Petri Net) para la detección de caídas. Es decir, basándose en el modelo HLPFN se realiza un razonamiento difuso que permite predecir los movimientos de los usuarios todo ello en tiempo real. Es decir, mediante la recogida de los datos de sus acelerómetros de tres ejes y la aplicación del modelo de reconocimiento desarrollado, se ha conseguido un porcentaje de 97.72%, 94.67% y 95.12% de acierto para clasificación de caídas hacia delante, atrás y derecha e izquierda respectivamente.

En la actualidad, el sistema de detección de caídas del Apple Watch ha sido todo un éxito, no es solo una función en pruebas. Muestra de ello podemos señalar el caso recogido en febrero de 2022 en la web Applesfera (Bernal Raspall, 2022) en donde un Apple Watch salvó la vida de un hombre mayor tras sufrir una caída y perder el conocimiento en temperaturas bajo cero. En el documento se explica con detalle cómo un hombre de edad avanzada norteamericano sufrió una fuerte caída en una zona con temperaturas muy frías en el país y perdió el conocimiento. El reloj, al detectar la caída y que el propietario no respondía contactó inmediatamente con los servicios de emergencia proporcionando la ubicación del usuario (coordenadas exactas utilizando el GPS del dispositivo). Las asistencias fueron capaces de localizarlo y mandar las asistencias en 12 minutos tras la llamada del reloj. En este mismo artículo, se detalló cómo utilizando una vez más los datos de los acelerómetros de los que dispone el reloj también se han mejorado los algoritmos de análisis de tal forma que se es capaz de detectar además choques fuertes (por ejemplo, un accidente de tráfico) y actuar solicitando auxilio como en el caso anterior.

- **Sensores de temperatura corporal**

Los sensores de temperatura corporal se utilizan para medir la temperatura del cuerpo de una persona. Estos dispositivos pueden ser especialmente útiles para las personas mayores que tienen problemas de salud crónicos, como la diabetes o la enfermedad cardiovascular, ya que pueden ayudar a detectar problemas de salud en etapas tempranas.

Estos sensores utilizan tecnología termopila, que es capaz de medir la temperatura corporal sin estar en contacto con ella, mediante la radiación infrarroja que emite la piel y convirtiéndola en una señal eléctrica procesable. Además, el uso de la muñeca (en los casos de los relojes) es ideal ya que es una zona del cuerpo que suele mantener una temperatura estable y semejante a la central del cuerpo.

¹ Detección de caídas en Apple Watch: <https://support.apple.com/es-es/HT208944>

- **Sensores de electrocardiograma (ECG)**

Los sensores de ECG son dispositivos que se utilizan para medir la actividad eléctrica del corazón. Estos dispositivos pueden ser especialmente útiles para las personas mayores que tienen problemas cardíacos, ya que pueden ayudar a detectar problemas de arritmia o de otros tipos de trastornos cardíacos. También pueden ayudar a detectar situaciones de estrés o la realización de ejercicio físico.

Una de las más interesantes añadidas al reloj de Apple que ya detectaba caídas, es el ECG. En un artículo publicado por la revista Europapress (EuropaPress, 2020), se destaca cómo el reloj fue capaz de detectar en una mujer alemana de 80 años síntomas de isquemia cardiaca que había pasado desapercibida en una prueba de ECG de 12 canales realizada en el hospital.

Este tipo de funcionalidad relativa al ECG, incorporadas a dispositivos *wearables*, como en el Apple Watch o en el Samsung Galaxy Watch, funcionan mediante un sensor en la parte posterior del reloj, que permite medir la actividad eléctrica a través de la piel en la muñeca del usuario. Para ello, el usuario debe colocar su dedo en un punto indicado actuando éste de electrodo y a partir de ese momento el reloj comienza a medir la actividad eléctrica del corazón generando un registro que incluso podría (o debería) ser analizado por un médico. Con este dato, pueden detectarse irregularidades en el ritmo cardíaco (por ejemplo, la fibrilación auricular).

En el trabajo de Sprenger et al. (2022), se realizó un estudio para validar la fiabilidad de estos dispositivos (un Apple Watch en este caso). El final del estudio determina que el software utilizado es factible y fiable en sus registros, ya que, de una muestra de 100 usuarios, fue capaz de detectar derivaciones precordiales en el 98% de las ocasiones observándose además una correlación entre la amplitud de las ondas del ECG estándar y las que se mostraban por software utilizando el reloj de Apple en esta ocasión. Estas derivaciones precordiales se utilizan para obtener una evaluación completa del ritmo cardíaco y su función eléctrica ya que proporcionan información sobre la actividad del corazón en distintas zonas, ayudando así a detectar anomalías y trastornos cardíacos.

- **Sistemas de monitorización cardiaca**

La monitorización cardiaca en los dispositivos inteligentes es una funcionalidad clave que permite a los usuarios realizar un seguimiento de su frecuencia cardíaca en tiempo real y obtener información sobre su salud cardiovascular. Estos dispositivos utilizan tecnologías específicas para medir y monitorizar el ritmo cardíaco de forma continua o bajo demanda.

Los smartwatches y pulseras de actividad suelen estar equipados con sensores ópticos de frecuencia cardíaca. Estos sensores utilizan luces LED y fotodetectores para medir el flujo sanguíneo en la muñeca del usuario. El sensor emite luces verdes o rojas que se reflejan en los vasos sanguíneos y los fotodetectores miden los cambios en la cantidad de luz absorbida para determinar los latidos del corazón.

El proceso de medición de la frecuencia cardíaca puede variar ligeramente según el dispositivo, pero generalmente consiste en fotopletismografía (PPG), y sigue estos pasos:

1. El sensor óptico de frecuencia cardíaca emite luces verdes o rojas en la muñeca del usuario.
2. Las luces se reflejan en los vasos sanguíneos y son captadas por los fotodetectores del dispositivo.
3. Los fotodetectores miden los cambios en la cantidad de luz absorbida, que está relacionada con los latidos del corazón.
4. Los datos capturados se procesan mediante algoritmos específicos para determinar la frecuencia cardíaca del usuario.
5. La frecuencia cardíaca se muestra en tiempo real en la pantalla del dispositivo o se sincroniza con una aplicación móvil o plataforma en la nube.

Es importante tener en cuenta que los sensores ópticos de frecuencia cardíaca en los dispositivos inteligentes no son tan precisos como los ECG médicos de grado clínico. Sin embargo, ofrecen una estimación bastante precisa de la frecuencia cardíaca en condiciones normales de uso.

Es posible que los dispositivos inteligentes también ofrezcan funciones adicionales relacionadas con la monitorización cardíaca, como la detección de arritmias o la estimación de la variabilidad de la frecuencia cardíaca, que pueden proporcionar información adicional sobre la salud cardiovascular del usuario. Estas funciones suelen basarse en algoritmos más avanzados que analizan los patrones y variaciones en la frecuencia cardíaca.

- **Sistemas de monitorización del sueño**

Estos dispositivos pueden medir la calidad del sueño de una persona y ofrecer recomendaciones para mejorarla. Los sistemas de monitorización del sueño pueden ser especialmente útiles para las personas mayores con problemas cognitivos, ya que los trastornos del sueño son comunes en este grupo de edad y pueden empeorar los síntomas cognitivos.

Por ejemplo, en personas mayores con enfermedad de Alzheimer, se ha demostrado que los trastornos del sueño son comunes y pueden afectar la calidad de vida del paciente y de sus cuidadores. La monitorización del sueño mediante dispositivos portátiles o sistemas en el hogar puede ayudar a detectar patrones de sueño anormales, como la apnea del sueño o la fragmentación del sueño, que puedan estar afectando la cognición.

Además, los sistemas de monitorización del sueño pueden ser útiles para identificar otros problemas de salud que puedan estar afectando la cognición de las personas mayores, como la depresión, la ansiedad o la enfermedad de Parkinson (Ferradans Rodríguez & Soto González, 2017).

Con los datos recopilados mediante la monitorización del sueño, los profesionales de la salud pueden realizar ajustes en la medicación, en la dieta o en la actividad física del paciente, para mejorar la calidad del sueño y reducir los síntomas cognitivos. También pueden ofrecer recomendaciones para mejorar el entorno de sueño, como reducir la exposición a la luz artificial, evitar ruidos o mantener una temperatura adecuada en la habitación.

Los sensores de monitorización del sueño de los relojes inteligentes como el Apple Watch² o el Samsung Galaxy Watch³ funcionan a través de una combinación de sensores de movimiento, monitores de frecuencia cardíaca y la técnica de fotoplethysmografía (PPG), según se indica en la propia web del fabricante y en artículo.

Los sensores de movimiento, como el acelerómetro y el giroscopio, detectan el movimiento y la posición del reloj en la muñeca del usuario. Estos sensores son capaces de registrar la cantidad de movimiento del usuario durante la noche, lo que puede ayudar a determinar cuánto tiempo se tarda en conciliar el sueño, cuántas veces se despierta durante la noche y cuánto tiempo se pasa en diferentes fases del sueño.

Además, los relojes inteligentes también están equipados con monitores de frecuencia cardíaca que utilizan sensores ópticos para medir la frecuencia cardíaca del usuario durante la noche. Los cambios en la frecuencia cardíaca pueden indicar cuándo el usuario está en diferentes fases del sueño, como el sueño profundo o el sueño REM.

La combinación de datos recopilados por los sensores de movimiento, monitores de frecuencia cardíaca y la técnica de PPG permite a los relojes inteligentes calcular la cantidad de sueño que ha tenido el usuario y la calidad de ese sueño. Algunos relojes inteligentes, como el Apple Watch y el Samsung Galaxy Watch, también utilizan algoritmos de aprendizaje automático obteniendo un patrón de sueño de una persona con mediciones durante varios días, y así poder proporcionar puntuaciones de calidad de sueño y recomendaciones para mejorar el sueño del usuario.

- **Sistemas de monitorización de movimiento con wearables (personal)**

Los sistemas de monitorización de movimiento en smartwatches y pulseras de actividad utilizan una combinación de sensores y algoritmos para detectar y registrar el movimiento del usuario. Estos dispositivos suelen contar con acelerómetros, giroscopios y sensores ópticos de frecuencia cardíaca para recopilar datos sobre el movimiento y la actividad física.

Acelerómetro: El acelerómetro mide la aceleración lineal en diferentes direcciones. Detecta los cambios en la velocidad y dirección del movimiento. Por ejemplo, registra los pasos dados al caminar o correr, así como los movimientos del brazo al hacer ejercicio.

Giroscopio: El giroscopio mide la orientación y el movimiento rotacional. Permite detectar movimientos como giros, movimientos de muñeca y cambios de dirección.

Sensor GPS: este sensor funciona mediante la recepción de señales de satélites en órbita y permite calcular la posición geográfica exacta en que se encuentra el usuario, permitiendo hacer un registro del recorrido de sus movimientos.

² PPG Apple Watch: <https://support.apple.com/en-us/HT204666>

³ PPG Samsung Galaxy Watch: <https://www.samsung.com/es/apps/samsung-health-monitor/>

Altímetro: permite al usuario obtener información detallada de la altitud durante sus actividades, proporcionando información muy interesante para personas que disfrutan de deportes con cambios de elevación significativos.

Estos sensores capturan datos de movimiento y actividad en tiempo real, que luego se procesan y analizan utilizando algoritmos específicos. Los algoritmos interpretan los datos recopilados para determinar la cantidad de pasos dados, la distancia recorrida, las calorías quemadas y otros parámetros relevantes de la actividad física.

- **Sistemas de monitorización del estado de ánimo y estrés**

Los sistemas de monitorización del estado de ánimo pueden ser especialmente útiles para las personas mayores que tienen problemas de salud mental, como la depresión o la ansiedad. Estos sistemas pueden medir el estado emocional de una persona y ofrecer recomendaciones para mejorar su bienestar mental, pero cabe destacar que no hay sensores específicos para medir emociones, sino que las estimaciones se hacen con técnicas de análisis y clasificación sobre los datos.

En el mercado, existen relojes como el Samsung Galaxy Watch que son capaces de cuantificar el estrés, midiendo ciertos biomarcadores (Samsung, 2021). Samsung utiliza una combinación de la frecuencia cardíaca y su variabilidad entre latidos (VFC). Nuestro sistema nervioso autónomo (SNA) es el responsable de ajustar nuestra frecuencia cardíaca, presión arterial y respiratoria y además es quien se encarga de nuestras respuestas de huida (estrés alto) y de los estados relajados (estrés bajo).

- **Sistema de atención social continuada y personalizada**

Por otro lado, si nos centramos en el ámbito nacional y más concretamente en Andalucía cabe destacar que se facilita la adquisición de forma gratuita del servicio de teleasistencia⁴. Programa que, dejando a un lado la gran labor que realiza acompañando a las personas mayores en su día a día en caso de que lo necesiten, también proporciona un sistema de auxilio en forma de “*botón del pánico*”: pequeño dispositivo que una vez pulsado, pone en marcha un sistema de auxilio en caso de emergencia. El gran problema es que a pesar de que la idea es fantástica, no tiene ningún tipo de automatización.

El actuador solo funciona si el usuario pulsa el botón y visto todo lo que se puede llegar a hacer con el análisis de datos, tras lo mostrado con relojes inteligentes como el Apple Watch o el Samsung Watch, sería excepcional añadir algún tipo de mejora en este servicio.

En lo relativo a nuestro proyecto, ya se han hecho algunos intentos de incorporar algunos de los sensores que hemos nombrado anteriormente. Por ejemplo, Alor-Hernández et al. (2022) proponen el uso de uno de estos dispositivos (pulsera inteligente compatible con *Google Fit*), de tal forma que mediante una **medición continuada y un post-procesado** de los datos el sistema es capaz **de detectar y prevenir la aparición de hipertensión en**

⁴ Servicio andaluz de teleasistencia:

<https://www.juntadeandalucia.es/agenciadeserviciosocialesydependencia/index.php/m-teleasistencia/m-que-ofrece>

los usuarios, mediante el análisis de variables fisiológicas como pueden ser frecuencia cardiaca, pasos realizados, distancia recorrida y presión arterial y mediante un algoritmo de aprendizaje automático propio, pero en este caso **la toma de datos se realiza sola y de forma continuado**, el usuario no tiene que tomar ningún dato de forma manual y los informes son accesibles a través de los altavoces inteligentes mediante su cuenta asociada.

Si volvemos al foco del control de personas de edad avanzada, dispositivos que permitan acceder a **datos de la salud** del usuario (presión arterial, sueño, pasos, frecuencia cardiaca...), combinados con sistemas de análisis de datos y recomendación, **supondrían un avance significativo**, para que el usuario no tenga que manejar ningún tipo de dispositivo para conocer su estado actual. Estos sistemas podrían incluso advertir de forma automática y activa al usuario y a sus cuidadores que hay algún tipo de problema, por ejemplo, encendiendo las luces para despertar al usuario y activando una llamada automática al cuidador (medición continua automática).

De hecho, hay estudios que han comprado la viabilidad del uso de estos dispositivos (*wearables* inteligentes) para estudios médicos. En Olmedo-Aguirre, et al., (2022) se resaltan los desafíos y avances en el desarrollo de dispositivos portátiles para el monitoreo de la salud. A pesar de los avances en la medición de variables fisiológicas, aún **existen limitaciones en lo que se puede medir**, especialmente en áreas de difícil acceso. Se busca desarrollar dispositivos más pequeños, con mayor vida útil de la batería y métodos de comunicación eficientes. Se exploran nuevas tecnologías como IoT y IA, y se considera la importancia de biosensores más precisos y **dispositivos invasivos**. Aunque se reconocen las limitaciones del estudio, se destaca el **potencial de los dispositivos portátiles** en el monitoreo de enfermedades crónicas y se resalta la necesidad de investigar y mejorar en este campo.

La normalización del uso de *wearables* y dispositivos de *IoT* que ha ocurrido tras la comercialización masiva de altavoces inteligentes como hemos discutido en este documento es evidente. Sin embargo, con ello también ha aparecido una gran cantidad de nuevos dispositivos que más que funcionales se categorizan como de control. Con dispositivos de control nos referimos a por ejemplo termostatos inteligentes que aparte de poder ser controlados a distancia, son capaces de acceder a internet para adaptar su temperatura en función del clima que vaya a hacer, cerraduras inteligentes interconectadas a aplicaciones como la de *Airbnb* y que permiten crear llaves virtuales (código PIN) temporales para los huéspedes (Nuki, 2022) o la gran variedad de relés inteligentes útiles para infinitud de situaciones. Toda esta intercomunicación entre dispositivos y aplicaciones se ha visto favorecidas gracias al uso de las *API*.

2.4 API (*Application Programming Interfaces*)

El pilar fundamental de este proyecto gira en torno a la información recopilada del día a día de personas en su hogar, centrándonos especialmente en aquellas que viven solas. Como bien se explicó en el apartado 1.1 *Justificación*, mediante el uso de sensores distribuidos por todo el domicilio, repartidos estratégicamente, se tratará de detectar casos de deterioro cognitivo u otros problemas (como una caída) mediante el análisis de los datos recopilados. Además, en las situaciones en que sea posible, como se detallará posteriormente, la

información también se utilizará para tomar decisiones sobre la realización de ciertas acciones (encendido y apagados de luces, contactar con cuidadores, ...)

En cuanto a los sensores, en el mercado actual existe una cantidad muy diversa de productos que permiten detectar cambios de diversa índole en el entorno. Entre estos cambios podemos destacar algunos muy comunes, como pueden ser sensores de movimiento, que permitan detectar presencia en distintas estancias, de temperatura, de luz, de humedad, etc. Una gran parte de estos productos comparten una peculiaridad, que se está extendiendo cada vez más, y es que permiten interactuar con ellos mediante aplicaciones propias, de tal forma que el usuario no necesita programarlos, sino simplemente configurarlos para poder acceder a sus funciones (en la mayoría de los casos basta con conectarlos a una red Wi-Fi). El uso de estos sensores se ha extendido hasta el punto de que existen otros productos que lo integran internamente y son capaces de interactuar con los cambios en el medio de forma inteligente como en el caso de bombillas, persianas, aspiradores inteligentes...

En la Figura 2.1 se puede observar un ejemplo práctico del funcionamiento de la domótica actual utilizando el ecosistema de Google (Google Home). En el caso de la “Luz de la despensa”, el usuario puede saber si está encendida, regular su intensidad, su color e incluso programar un horario de funcionamiento.

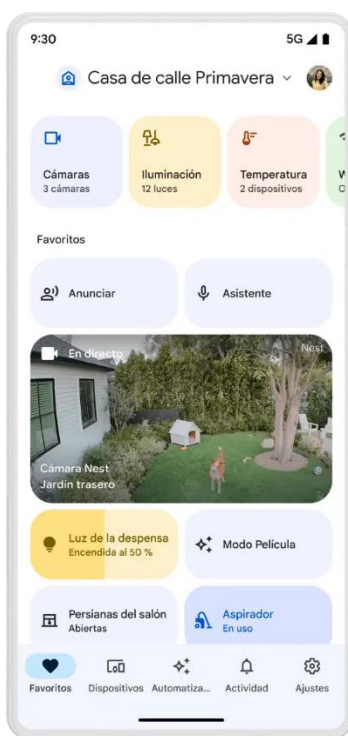


Figura 2.1 Interfaz gráfica de Google Home (Google⁵)

Con una aplicación como ésta, el usuario tendría acceso a la situación de todos sus dispositivos. En el caso de sensores de movimiento, el funcionamiento se presentaría como

⁵ Información sobre la interfaz de Google Home: https://home.google.com/intl/en_uk/the-latest/

un conjunto de alertas que se registran en la aplicación cada vez que el sensor detecte algún tipo de cambio en el entorno. Sin embargo, hay una **limitación** enorme, que está relacionada con el factor humano, ya que estas **alertas por sí solas no desencadenan ningún tipo de acción** hasta que el usuario nos las analice y decida qué hacer en consecuencia tras estudiarlas.

Imagínese el lector que se dispone de un sensor capaz de detectar cuándo una planta tiene un nivel de humedad bajo y, a su vez, también se dispone de un actuador que permite regar las plantas. Ambos dispositivos pueden controlarse remotamente, pero depende de la acción del usuario. Es decir, un nivel bajo de humedad no activa automáticamente el regado.

En cuanto al ejemplo anterior extrapolado a nuestro caso particular, tendremos una serie de alertas como resultado de nuestros sensores, donde aparte de realizar una actuación ante el estudio de la información obtenida, nuestro primer problema partirá de cómo tratar estos datos en crudo. Más en concreto, el problema realmente será como automatizar la recopilación y el análisis de los datos, ya que no podemos depender de un usuario accediendo a la aplicación y mucho menos que se encargue de publicar los datos de forma manual en algún tipo de sistema de almacenamiento de datos.

Ante este problema, una solución factible sería intentar realizar una **automatización** y centralización de la recogida de datos a partir de las distintas API que permiten usar los varios proveedores de los sensores que se pretenden instalar.

Una **API** (siglas de la expresión en inglés “*Application Programming Interfaces*”), no es otra cosa que una especificación formal que establece cómo un módulo software puede interactuar e intercambiar información con otro. Es decir, permite establecer una comunicación entre dos aplicaciones a través de un conjunto de reglas previamente establecidos. Una API es una especie de intermediario que permite a diferentes programas o aplicaciones que se comuniquen entre sí de manera estandarizada y ordenada, con la principal ventaja de que los programadores no necesitan conocer los detalles internos de cada uno de ellos.

Por exponer algunos casos, existen diversos tipos de API que podríamos usar para sensores, pero casi la totalidad con la que nos hemos encontrado para consultar los servicios que detallaremos a continuación son API REST o API SOAP.

Una **API REST** (*Representational State Transfer*) es una interfaz web que utiliza el protocolo *HTTP* para enviar y recibir los datos en diferentes formatos, como pueden ser *XML* o *JSON*. En esencia, se basa en los verbos *HTTP* (*GET*, *POST*, *PUT* y *DELETE*, entre otros) para realizar operaciones sobre los recursos, y utiliza una URL única para cada recurso, facilitando así la navegación y comprensión de la estructura de la API. En resumen, una *API REST* suele ser más simple, flexible y escalable que las *API SOAP*, y no requieren el uso de un lenguaje de descripción de servicios.

Por otro lado, una **API SOAP** (*Simple Object Access Protocol*) es una interfaz web que utiliza el protocolo *HTTP* para enviar y recibir datos en formato *XML*. Utiliza una estructura

más rígida y compleja que las *API REST*, ya que requiere de un lenguaje de descripción de servicios (*WSDL – Web Services Description Language*) que define la estructura de los mensajes y las operaciones disponibles. Las *API SOAP* utilizan diferentes verbos *HTTP* para realizar operaciones sobre los recursos y requieren de una mayor cantidad de código para implementarlas y consumirlas.

Actualmente, las ***API REST* son más populares** y ampliamente utilizadas que las *API SOAP*. Las razones son:

- **Simplicidad.** Son más sencillas de entender y usar que las *API SOAP*, ya que no requieren del uso de un lenguaje de descripción de servicios (*WSDL*) ni de una estructura rígida para los mensajes.
- **Flexibilidad.** Permiten el intercambio de datos en diferentes formatos, como *XML* o *JSON*, lo que las hace más flexibles y adaptables a diferentes plataformas y dispositivos.
- **Escalabilidad.** Son más escalables, ya que no dependen de una estructura rígida para los mensajes, lo que las hace más fáciles de modificar y actualizar.
- **Popularidad.** Son más populares y ampliamente utilizadas que las *API SOAP*, lo que significa que hay una mayor cantidad de recursos y herramientas disponibles para trabajar con ellas.

La elección de una u otra *API* es determinante para nuestro proyecto. Debido a que determinarán factores claves, como son la escalabilidad, velocidad y seguridad.

En las siguientes secciones se realizará un análisis entre los principales proveedores de servicios *API* disponibles para este tipo de sensores que nos gustaría utilizar, así como de los fabricantes principales de igual forma. La elección de una u otra *API* determinará el fabricante de los diferentes sensores y actuadores que se utilizarán para este proyecto, ya que no todos los fabricantes permiten la utilización de sus dispositivos mediante *API*, pero es un aspecto fundamental en nuestro proyecto.

2.4.1 Google Homegraph API

La *API* de *Google Homegraph*⁶ es una interfaz de programación de aplicaciones que permite a los desarrolladores acceder y controlar dispositivos inteligentes compatibles con el ecosistema de Google Home. En pocas palabras, es un conjunto de herramientas y funciones que permiten a los dispositivos inteligentes comunicarse entre sí y con el asistente virtual de Google.

En la actualidad, existen una gran cantidad de productos que son compatibles con el altavoz inteligente de Google y su asistente. Google Assistant es el asistente virtual de Google que utiliza tecnología de lenguaje natural y aprendizaje automático para proporcionar respuestas a preguntas y realizar acciones a través de comandos de voz (**que incorpora el altavoz**). Es el núcleo principal de la plataforma de Google Home y es el medio a través

⁶ Documentación de Google Homegraph API: <https://developers.home.google.com/reference/home-graph/rpc>

del cual los usuarios interactúan con los dispositivos inteligentes y las aplicaciones creadas con la API de Google Homegraph.

La relación entre Homegraph API y Google Assistant es que la Homegraph API proporciona a Google Assistant acceso a la información sobre los dispositivos domésticos inteligentes conectados al hogar, como luces, termostatos, cerraduras, altavoces y otros dispositivos. Con esta información, Google Assistant puede controlar y gestionar estos dispositivos a través de comandos de voz y texto. Ante el input del usuario el asistente de voz procesa la orden de voz y crea una orden ejecutable, que en primer lugar actualizará los datos en la base de datos del proveedor y tras esto mandará otra orden a Homegraph API para que actualice el estado del dispositivo (ver Figura 2.2).

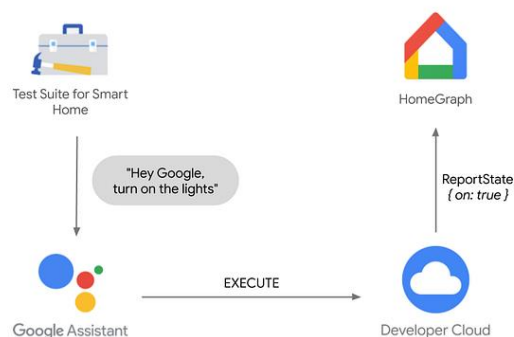


Figura 2.2 Flujo de datos en Homegraph (Smith, 2020)

Homegraph API, por otro lado, proporciona información sobre los dispositivos domésticos inteligentes conectados al hogar a través del modelo de datos Homegraph de Google. Este modelo de datos almacena información sobre la estructura y el estado del hogar inteligente, incluyendo los **dispositivos conectados, sus atributos y su estado actual**. Los **desarrolladores pueden acceder a esta información a través de Homegraph API** para usarlos en aplicaciones y servicios que se pueden brindar experiencias personalizadas y contextuales para el hogar inteligente.

Por tanto, la API de Google Homegraph y Google Assistant están muy relacionadas, ya que ambos forman parte de la misma plataforma para el control de dispositivos inteligentes y el asistente virtual.

La API Homegraph consta de **varios endpoints** (URL específica o punto de acceso en una API), cada uno con una función específica. A continuación, se detallan los *endpoints* más importantes de la API de Google Homegraph⁷.

- **SYNC:** punto de entrada para la API de Google Home. Este *endpoint* se utiliza para informar a Google sobre los dispositivos inteligentes que se han integrado con la plataforma. La información proporcionada incluye la descripción de los dispositivos,

⁷ Documentación de la API de HomeGraph: <https://developers.home.google.com/reference/home-graph/rpc>

como el nombre, la ubicación, el tipo de dispositivo y sus capacidades, como si se pueden encender o apagar, si tienen capacidad de regulación, etc.

- **QUERY:** con este *endpoint*, es posible obtener información sobre el estado actual de los dispositivos, como si están encendidos o apagados, la temperatura actual de un termostato, etc. Esto es útil para aplicaciones que necesitan proporcionar información precisa a los usuarios.
- **EXECUTE:** se utiliza para enviar comandos a los dispositivos integrados en la plataforma. Con este *endpoint*, es posible encender o apagar dispositivos, establecer la temperatura de un termostato, reproducir música en un altavoz inteligente, etc. Este *endpoint* permite a las aplicaciones controlar los dispositivos mediante comandos de voz.
- **REPORT STATE:** se utiliza para informar a Google sobre el estado actual de los dispositivos. Este *endpoint* se utiliza después de que se haya enviado un comando para actualizar el estado de los dispositivos.
- **REQUEST SYNC:** se utiliza para solicitar una actualización de la lista de dispositivos integrados en la plataforma. Este *endpoint* se utiliza cuando se agregan nuevos dispositivos a la plataforma. Después de llamar a este *endpoint*, Google realizará una nueva solicitud SYNC para obtener la información actualizada de los dispositivos.
- **DISCONNECT:** se utiliza para desconectar un dispositivo de la plataforma de Google Home. Este *endpoint* se utiliza cuando se retira un dispositivo de la plataforma o cuando se desconecta una cuenta de usuario de la plataforma.

Hasta el momento, parece que esta API abre infinitud de posibilidades para el desarrollo de la aplicación que deseamos realizar. Lamentablemente, la API de Google HomeGraph está **diseñada** principalmente para que los **fabricantes de dispositivos** inteligentes puedan **implementarla en sus productos** y permitir que los usuarios controlen sus dispositivos a través de Google Assistant.

Esto quiere decir que, a pesar de que es cierto que, a través de ciertas estrategias en la que interviene el uso de software de terceros, sería posible el control y acceso total a los distintos dispositivos asociados con nuestra cuenta de Google, **no es su principal aplicación** y **no se diseñó para este fin**, por lo que aumentaríamos la complejidad del proyecto, promoviendo incluso una **posible incompatibilidad en un futuro**.

Homegraph, por tanto, más que una herramienta para el control de los dispositivos asociados se ha convertido simplemente en un intermediario entre el altavoz inteligente y su software de reconocimiento de voz (Google Assistant) y la propia API interna que utilizan los distintos proveedores, la cual puede ser o no abierta al usuario estándar para su uso libre.

Alguna de las soluciones para poder usar la API propuesta anteriormente requeriría desarrollar un sistema parecido a **Home Assistant**⁸, un sistema de automatización del hogar de código abierto que permite a los usuarios controlar y monitorear una amplia variedad de dispositivos y servicios en su hogar. Es una plataforma domótica que integra dispositivos inteligentes y servicios en un solo sistema, lo que permite a los usuarios controlar y monitorear su hogar de manera más fácil y eficiente.

2.4.2 Amazon Alexa

Amazon Alexa⁹ es una plataforma de interacción inteligente (como Google Assistant) que permite a los usuarios interactuar con dispositivos y diferentes servicios mediante comandos de voz. Para permitir que los desarrolladores creen funcionalidades (**habilidades**) para Alexa, Amazon proporciona un conjunto de diferentes tipos de API que se pueden utilizar para integrar e interaccionar con los dispositivos compatibles con Alexa.

Existen **varias API destinadas a diferentes funciones** y muy variadas, por ejemplo: “*Video Skill API*”, que permite crear habilidades que integren contenido de video en dispositivos compatibles con Alexa, y “*Custom Skill API*”, que permite crear habilidades que pueden integrarse con servicios web, como, por ejemplo, reservar una mesa en un restaurante.

Es ésta una de las razones por las que el desarrollo mediante el uso de los recursos proporcionados por Amazon complica un poco todo el proyecto, ya que existe la posibilidad de que en el futuro se requiera la integración de la API de video para mostrar una señal al usuario (siendo esto último un ejemplo), lo que implicaría una implementación completamente diferente. Esta complejidad adicional puede plantear desafíos en términos de integración y mantenimiento a largo plazo.

Para la detección y control de dispositivos inteligentes Amazon proporciona la API llamada “*Alexa Smart Home Skill API*”, la cual tiene una lógica un poco extraña, ya que en lugar de separar funcionalidad utilizando **distintos endpoints**, hace uso de un **parámetro** llamado “**event**” **para diferenciar la función del endpoint**. Es decir, a diferencia del caso de Google que ofrece varios *endpoints* diferentes, uno para cada acción, Amazon solo ofrece uno. Este *endpoint* (“<https://api.eu.amazonalexa.com/v3>”) recibe un objeto JSON que en función del identificador “*event*” del cuerpo del contenido define la capacidad que se quiere realizar. Por ejemplo, el “*event*” con identificador “*Alexa.Discovery*” se usa siempre si queremos descubrir los dispositivos disponibles. Tras esto, obtendremos una respuesta similar a la Figura 2.3.

⁸ Web oficial de Home Assistant: <https://www.home-assistant.io/>

⁹ Web oficial de Amazon Alexa: <https://developer.amazon.com/es-ES/alexa>

```
{
  "event": {
    "header": {
      "namespace": "Alexa.Discovery",
      "name": "Discover.Response",
      "payloadVersion": "3",
      "messageId": "Unique identifier, preferably a version 4 UUID"
    },
    "payload": {
      "endpoints": [
        {
          "endpointId": "Unique ID of the endpoint",
          "manufacturerName": "Sample Manufacturer",
          "description": "Smart Light by Sample Manufacturer",
          "friendlyName": "Living Room Light",
          "additionalAttributes": {
            "manufacturer": "Sample Manufacturer",
            "model": "Sample Model",
            "serialNumber": "U111122334455",
            "firmwareVersion": "1.24.2546",
            "softwareVersion": "1.036",
            "customIdentifier": "Sample custom ID"
          },
          "displayCategories": ["LIGHT"],
          "cookie": {},
          "capabilities": [
            {
              "type": "AlexaInterface",
              "interface": "Alexa.PowerController",
              "version": "3",
              "properties": {
                "supported": [
```

Figura 2.3 Alexa Discovery Response¹⁰

Entre otras cosas, la respuesta que hemos mostrado a parte de la información propia de los dispositivos disponibles también muestra qué acciones están disponibles para estos dispositivos, ya que, como en la ocasión anterior, las acciones definen el evento a usar en la misma API.

Utilizando el ejemplo anterior, como puede observarse en la Figura 2.3, el dispositivo dispone de la capacidad “Alexa.PowerController”, la cual puede ser utilizada para modificar su estado de encendido a apagado y viceversa. En la Figura 2.4 se muestra el cuerpo del JSON que será enviado en la llamada POST a la API. Tras realizar la llamada post el sensor modificará su estado y se activará, ya que se usa el comando “TurnOn” en el evento con identificador “Alexa.PowerController”.

¹⁰ Documentación de la API de Amazon: <https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-discovery.html>

```
{
  "directive": {
    "header": {
      "namespace": "Alexa.PowerController",
      "name": "TurnOn",
      "messageId": "Unique version 4 UUID",
      "correlationToken": "Opaque correlation token",
      "payloadVersion": "3"
    },
    "endpoint": {
      "scope": {
        "type": "BearerToken",
        "token": "OAuth2.0 bearer token"
      },
      "endpointId": "Endpoint ID",
      "cookie": {}
    },
    "payload": {}
  }
}
```

Figura 2.4 Alexa.PowerController¹¹

2.4.3 Tuya API

Tuya Smart¹² es una empresa de tecnología de la información con sede en China que se dedica al desarrollo de soluciones para Internet de las cosas. Fundada en 2014, Tuya Smart ofrece una plataforma de desarrollo de *IoT* para fabricantes y desarrolladores de productos inteligentes, lo que les permite integrar fácilmente tecnologías inteligentes en sus productos y servicios.

Ofrece una amplia gama de dispositivos domóticos que se pueden integrar fácilmente en una casa inteligente. Estos dispositivos incluyen interruptores de luz inteligentes, enchufes inteligentes, cámaras de seguridad, cerraduras inteligentes, termostatos inteligentes, sensores de movimiento, sensores de puertas y ventanas, entre otros.

Los dispositivos domóticos de Tuya Smart se pueden controlar y monitorear a través de su aplicación móvil de una forma fácil, **lo que permite a los usuarios controlar sus hogares desde cualquier lugar con una conexión a Internet** lo que brinda una mayor flexibilidad y comodidad en la gestión de sus hogares. La aplicación móvil también ofrece otras características, como programación de horarios, control de voz a través de asistentes

¹¹ Documentación de la API de amazon: <https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-response.html>

¹² Información sobre Tuya: <https://www.tuya.com/about>

virtuales, como Amazon Alexa y Google Assistant, y alertas de seguridad para una mayor tranquilidad.

Existe además una plataforma de desarrollo de IoT para otros fabricantes y desarrolladores de productos inteligentes, lo que les permite integrar fácilmente tecnologías inteligentes en sus productos y servicios. Esto significa que los usuarios pueden encontrar una amplia variedad de dispositivos domóticos que utilizan la tecnología de Tuya Smart, lo que les brinda una mayor flexibilidad en la creación de una casa inteligente personalizada y adaptada a sus necesidades específicas.

En cuanto a desarrollo, Tuya Smart ofrece una API pública para que los desarrolladores puedan integrar sus dispositivos inteligentes en sus propias aplicaciones. La API de Tuya Smart permite a los desarrolladores controlar y monitorear dispositivos inteligentes, así como acceder a los datos recopilados por los dispositivos.

Además, Tuya Smart también ofrece una plataforma de desarrollo de IoT online que permite a los desarrolladores crear sus propios dispositivos inteligentes utilizando la tecnología de Tuya Smart.

A través de la API, los desarrolladores pueden realizar una serie de operaciones, como:

- **Controlar** los dispositivos inteligentes de **forma remota**, de igual forma que se hace con la aplicación móvil.
- **Recopilar datos** de los dispositivos inteligentes, como la temperatura, el consumo de energía y el estado del dispositivo, entre otros.
- **Programar acciones** para los dispositivos inteligentes, como **encender o apagar** un dispositivo en un momento específico.
- **Configurar notificaciones** para alertas de seguridad, como cuando se detecta movimiento en una cámara de seguridad o cuando se abre una puerta o ventana.

Además de estas funcionalidades, la API de Tuya Smart también ofrece una integración fácil con otras plataformas y servicios, como Amazon Alexa y Google Assistant. Esto permite a los usuarios controlar sus dispositivos inteligentes mediante comandos de voz, lo que proporciona una mayor comodidad y accesibilidad.

La API de Tuya Smart ofrece varios *endpoints* para permitir a los desarrolladores interactuar con sus dispositivos inteligentes. A continuación, se describen algunos de los *endpoints* disponibles en la API y su propósito (Tuya, 2023):

- **GET /v1.0/devices:** Este *endpoint* devuelve información detallada sobre los dispositivos asociados a la cuenta de un usuario, como su identificador único, el tipo de dispositivo, el nombre y el estado actual del dispositivo.
- **PUT /v1.0/devices/{device_id}:** Este *endpoint* se utiliza para actualizar el estado de un dispositivo específico. Por ejemplo, si se trata de un dispositivo de iluminación, este *endpoint* se puede utilizar para encender o apagar la luz.

- **GET /v1.0/devices/{device_id}/status:** Este *endpoint* devuelve el estado actual de un dispositivo específico, como si está encendido o apagado, y cualquier otro estado relevante del dispositivo.
- **POST /v1.0/devices/{device_id}/commands:** Este *endpoint* permite enviar comandos personalizados a un dispositivo específico, lo que permite a los desarrolladores crear funcionalidades personalizadas para sus dispositivos.
- **GET /v1.0/devices/{device_id}/datapoints:** Este *endpoint* devuelve los valores de los puntos de datos de un dispositivo, que pueden incluir información como la temperatura, la humedad, la calidad del aire, el consumo de energía y otros datos relevantes del dispositivo.
- **GET /v1.0/device/{device_id}/functions:** Este *endpoint* devuelve una lista de funciones disponibles para un dispositivo específico, lo que permite a los desarrolladores conocer las capacidades de un dispositivo en particular.
- **GET /v1.0/families:** Este *endpoint* devuelve una lista de todas las familias disponibles de dispositivos inteligentes que son compatibles con la plataforma de Tuya Smart.
- **POST /v1.0/token?grant_type=1:** Este *endpoint* se utiliza para obtener un token de acceso que se requiere para acceder a la API de Tuya Smart.

Éstos son solo algunos ejemplos de los *endpoints* disponibles en la API de Tuya Smart. Cada *endpoint* tiene un propósito específico y permite a los desarrolladores interactuar con los dispositivos inteligentes y acceder a información relevante de una manera personalizada y controlada.

2.4.4 Tabla comparativa de API

En la Tabla 1 se muestra una comparativa entre las diferentes API descritas en esta sección.

Característica	Google HomeGraph	Amazon API	Tuya API
Enfoque principal	Fabricantes de productos inteligentes	Desarrolladores de habilidades Alexa	Usuarios y desarrolladores
Disponibilidad de dispositivos	Amplia variedad, pero no todos los fabricantes están integrados	Amplia variedad, pero no todos los fabricantes están integrados	Amplia variedad, la mayoría de los fabricantes están integrados
Funciones específicas de los endpoints	No permite controlar dispositivos y monitorizarlos de forma nativa.	El control de los dispositivos a través de mensajería de eventos complejas	Permite controlar y monitorear dispositivos mediante <i>endpoints</i> sencillos
Documentación	Completa, pero compleja de entender y utilizar	Completa, pero compleja de entender y utilizar	Completa, fácil de entender y utilizar

Acceso a los datos	Solo los fabricantes pueden acceder a los datos de los usuarios	Los desarrolladores de habilidades pueden acceder a los datos de los usuarios	Los usuarios pueden elegir compartir sus datos con los desarrolladores de la aplicación
Integración con otros servicios	Amplia integración con otros servicios de Google	Amplia integración con otros servicios de Amazon	Amplia integración con otros servicios de terceros
Costo	Gratis, pero solo para fabricantes asociados con Google	Gratis, pero con limitaciones y requisitos de suscripción	Gratis para los desarrolladores y los usuarios

Tabla 1 – Comparativa entre API estudiadas

De la comparativa anterior se observa que la API de Tuya destaca por estar **diseñada para los usuarios y desarrolladores**, en lugar de solo para los fabricantes de nuevos dispositivos domóticos, lo cual la hace mucho **más accesible** para un público general. Además, la documentación de esta API es **fácil de entender y utilizar**, lo que facilita la creación de aplicaciones.

2.5 Protocolos de comunicación

A la explosión de las diversas tecnologías para el hogar que hemos visto hasta ahora contribuyó mucho la evolución tanto en los protocolos de comunicación como la evolución de microcontroladores y microprocesadores (ver 2.6 Unidades de control: microcontroladores y microprocesadores). Esta evolución ha sido crucial para el avance de la domotización y el conjunto de *API* utilizadas ya que han permitido una mayor interconexión entre los dispositivos domóticos, la estandarización de la comunicación, un mayor poder de procesamiento y la creación de interfaces de programación que facilitan el desarrollo de aplicaciones inteligentes para el hogar. En conjunto, han impulsado la accesibilidad, la inteligencia y la personalización de la domótica para los usuarios.

Estos protocolos permiten la comunicación entre los dispositivos del hogar y los sistemas de control. Aunque hay diversos protocolos de comunicación, algunos de los más usados son:

- **MQTT**¹³ (Message Queuing Telemetry Transport): es un protocolo de mensajería ligero y de bajo consumo de ancho de banda, diseñado para conectar dispositivos IoT (Internet of Things) de forma eficiente y confiable. Proporciona una comunicación basada en publicación-suscripción, donde los dispositivos pueden publicar mensajes en un tema y los suscriptores pueden recibir los mensajes de dicho tema. MQTT es ampliamente utilizado en la automatización del hogar debido a su capacidad de conexión confiable y su bajo consumo de energía. En la siguiente sección lo describiremos con más detalle en comparación con el uso de APIs.

¹³ MQTT: <https://mqtt.org/>

- **Zigbee¹⁴**: es un estándar de comunicación inalámbrica de baja potencia y corto alcance (10 metros en interior y 75 metros en exterior), diseñado para la automatización del hogar y la construcción de redes de sensores. Utiliza una topología de malla en la que los dispositivos pueden comunicarse entre sí a través de múltiples saltos. Zigbee ofrece una buena capacidad de interoperabilidad y es adecuado para aplicaciones que requieren una baja tasa de transferencia de datos y una vida útil prolongada de la batería.
- **Z-Wave**: es otro estándar de comunicación inalámbrica de baja potencia utilizado en la automatización del hogar de corto alcance (30 metros en interior y 100 metros en exterior). Proporciona una comunicación de radiofrecuencia de baja velocidad en la banda de frecuencia de 800-900 MHz. Z-Wave utiliza una topología de malla similar a Zigbee y se centra en la fiabilidad y la seguridad de la comunicación.
- **WiFi y Bluetooth**: fueron y son clave en el desarrollo de la domótica ya que permitió la conexión inalámbrica a internet y entre dispositivos en el hogar, facilitando además de las tareas de control, las futuras actualizaciones de firmware.

2.6 Unidades de control: microcontroladores y microprocesadores

En la automatización del hogar, se emplean microcontroladores y microprocesadores para el control y la gestión de los dispositivos y sistemas. Algunos de los microcontroladores y microprocesadores más utilizados son:

- **Arduino¹⁵**: es una plataforma de prototipado electrónico ampliamente utilizada en la domótica. Utiliza microcontroladores AVR de Atmel y proporciona una interfaz fácil de usar para el desarrollo de proyectos de automatización del hogar.
- **Raspberry Pi¹⁶**: es una computadora de placa única (SBC) que puede utilizarse para la automatización del hogar. Incorpora un microprocesador ARM y ofrece una amplia gama de capacidades de conectividad y procesamiento.
- **ESP8266/ESP32¹⁷**: Los microcontroladores ESP8266 y ESP32 son ampliamente utilizados en aplicaciones de IoT y domótica.

Estos microcontroladores ofrecen conectividad Wi-Fi, lo que permite una fácil comunicación y control de dispositivos en red.

¹⁴ Zigbee vs Z-Wave: <https://www.xataka.com/seleccion/zigbee-z-wave-que-que-se-diferencian-que-marcas-domotica-compatibles>

¹⁵ Arduino: <https://www.arduino.cc/en/about>

¹⁶ Raspberry-Pi: <https://www.raspberrypi.com/about/>

¹⁷ ESP8266 vs ESP32: <https://descubrearduino.com/esp32-vs-esp8266/>

En nuestra opinión, la aparición de estas tres unidades de control han sido realmente uno de los principales impulsos al desarrollo domótico. En resumen, un dispositivo domótico se estructura en tres partes principales, que son: el sensor o el actuador en sí (un sensor lumínico o un relé respectivamente, por ejemplo) y un controlador encargado de la lógica con los que se podrían diseñar dispositivos domóticos de forma casera.

Como ejemplo, existen multitud de centros educativos que desde hace tiempo realizan proyectos con elementos muy similares a los propuestos justo antes (yo mismo fui alumno de una asignatura en secundaria donde se realizó), relés inteligentes para ser más exactos. Una simple alargadera era seccionada en uno de los polos, el cual se conectaba al relé, siendo este gobernado lógicamente por un Arduino al que a su vez se le conectaba un microcontrolador *ESP8266* para tener acceso a internet. Por último, se utilizaba una Raspberry Pi para controlar vía WiFi nuestro Arduino y a su vez crear algún tipo de *endpoint* para acceder remotamente a este, y con ello se fabricaba un actuador inteligente casero.

2.7 MQTT vs API

En un primer momento puede parecer que *MQTT* y una *API* (utilicemos como ejemplo la de Tuya) no tienen relación alguna, pero ambas tecnologías pueden ser usadas para manejar dispositivos domóticos. Son dos tecnologías diferentes, pero que pueden trabajar juntas para permitir el control de dispositivos domóticos a través de Internet.

En la actualidad, una gran mayoría de los dispositivos domóticos utilizan *MQTT* para comunicarse internamente, sin embargo, el usuario puede interactuar con ellos por medio de distintas API. En este apartado aclararemos las diferencias entre ambas tecnologías.

2.7.1 Diferencias

La *API* de Tuya proporciona un conjunto de interfaces de programación de aplicaciones que permiten a los desarrolladores interactuar con los dispositivos domóticos conectados a la plataforma Tuya. Esto significa que los desarrolladores pueden utilizar la *API* para enviar y recibir comandos de control a través de Internet, y así controlar los dispositivos domóticos conectados a la plataforma Tuya.

Por otro lado, *MQTT* es un protocolo de comunicación ligero y eficiente diseñado específicamente para la comunicación entre dispositivos de *IoT*. *MQTT* permite el intercambio de mensajes de control y estado entre dispositivos domóticos conectados, facilitando la transmisión de información relevante para el funcionamiento de dichos dispositivos en un entorno *IoT*.

Por lo tanto, una posible relación entre *MQTT* y la *API* de Tuya sería que la *API* de Tuya podría utilizarse para enviar comandos de control de dispositivos domóticos a través de *MQTT*. De esta manera, los dispositivos domóticos podrían ser controlados mediante el uso de mensajes *MQTT* publicados en los temas adecuados.

En resumen, con todo esto queremos decir que cada uno de los dispositivos de los que vamos a hacer uso utiliza *MQTT* para su comunicación a nivel de proveedor, pero los distintos fabricantes han hecho la labor de abstraer toda esta lógica, de tal forma que

podemos hacer uso de sus productos simplemente por medio de las distintas API propuestas.

2.7.2 Beneficios de una API

Uno de los grandes avances según nuestra opinión que se ha alcanzado en nuestros días es poder utilizar una *API* propia por el fabricante en lugar de tener que diseñar un sistema de mensajería *MQTT* por nosotros mismos. Es decir, internamente el fabricante utiliza el protocolo *MQTT* para la intercomunicación de sus productos, pero han diseñado un contrato accesible (en la mayoría de los casos mediante una *API REST*) que permite al programador abstraerse y centrarse en la mera funcionalidad que quiere implementar. Algunas de las razones por las que una API pública puede ser preferible pueden ser:

- **Estándar de comunicación:** las API públicas generalmente siguen estándares y protocolos de comunicación ampliamente utilizados, lo que facilita la integración con diferentes sistemas y plataformas. Esto significa que puedes interactuar con la API utilizando diferentes lenguajes de programación y dispositivos, sin la necesidad de implementar y mantener un protocolo específico como *MQTT* en cada dispositivo.
- **Escalabilidad:** al utilizar una API pública, puedes aprovechar la infraestructura y la escalabilidad proporcionadas por el proveedor de la API. No tienes que preocuparte por la gestión de los aspectos técnicos de la comunicación, como el enrutamiento y la confiabilidad, ya que estos son manejados por el proveedor de la API.
- **Abstracción de complejidad:** una API pública puede simplificar la comunicación y ocultar la complejidad subyacente de los protocolos de bajo nivel como *MQTT*. Esto permite que los desarrolladores se centren en la lógica de negocio y la funcionalidad específica del dispositivo en lugar de preocuparse por los detalles de implementación de la comunicación.
- **Exposición a una comunidad más amplia:** al utilizar una API pública, puedes conectarte y colaborar con una comunidad más amplia de desarrolladores que utilizan la misma interfaz. Esto puede facilitar la colaboración, compartir conocimientos y aprovechar las soluciones y herramientas existentes desarrolladas por otros miembros de la comunidad.

Sin embargo, es importante tener en cuenta que la elección entre una API pública y *MQTT* directamente en dispositivos inteligentes depende de los requisitos específicos del proyecto y la arquitectura general. **MQTT puede ser más adecuado** en escenarios donde se requiere una comunicación de **baja latencia**, alto rendimiento y una gran cantidad de dispositivos conectados, como en sistemas *IoT* masivos. En nuestro caso, **la prioridad absoluta es el bajo coste y la interoperabilidad**. La utilización de una API pública abre el abanico de posibilidades de dispositivos que serán compatibles, ya que los desarrolladores solo tendrán que centrarse en cómo utilizar los datos obtenidos, dejando a un lado los sistemas de comunicación entre sensores, por ejemplo.

Lo que principalmente se quiere proponer es un sistema en el que el usuario no tenga que elegir un sensor o actuador muy específico, sino dar prioridad absoluta al precio y que solo tenga que ser compatible con uno de los grandes desarrolladores (Google, Amazon o Tuya).

2.8 Domótica actual. Estallido de los altavoces inteligentes

Como hemos comentado antes, en el año 2014, y solo para una selección de miembros a modo de testeo (se lanzaría de forma oficial al año siguiente) apareció **Amazon Echo**, siendo el primer altavoz inteligente estandarizado del mercado (Lacoma, 2022).

Este producto consistía en un cilindro con altavoces en su parte inferior, indicadores de uso y teclas de funcionamiento en la parte superior y un conjunto de micrófonos (siete en concreto), que le permitía tener un alcance de 360 grados. Son varios los motivos que propiciaron el éxito de este producto.

El primero de ellos, siendo a nivel técnico un gran paso hacia delante, es que se permitía interactuar utilizando el lenguaje natural, sin necesidad de utilizar comandos estáticos, a excepción de uno; ¡Alexa! Únicamente utilizando este comando de voz se puede empezar a interactuar con el altavoz. En un primer momento, Alexa solo era capaz de ejecutar órdenes “sencillas”, como dar la hora, temporizadores, poner música u opciones algo más difíciles, como responder cuestiones complejas a partir de búsquedas en internet.

Pero lo que realmente, desde nuestro punto de vista, hizo interesante a Alexa fue permitir el uso de “**skills**”. A grandes rasgos, las “**skills**” de Alexa, no son otra cosa que **aplicaciones** con las que puede **interactuar este asistente virtual**. Cada una de estas aplicaciones son (o pueden ser) desarrolladas por terceros.

Por defecto, Amazon Alexa (siendo Alexa el asistente de los altavoces inteligentes de Amazon) cuenta con una serie de “**skills**” básicas, que podían irse ampliando a medida que el usuario encontrara interesante cierta funcionalidad. Estas funcionalidades podían ser juegos, noticias, etc. En definitiva, apareció un nuevo concepto en el que los límites aún eran desconocidos, dependía de la originalidad de los desarrolladores en este nuevo campo.

El gran paso hacia delante (y que posteriormente propiciaría el impulso de la domótica), fue que **Amazon creó un estándar** e hizo **accesible** el uso del **lenguaje natural** a todo el mundo **mediante su API**. Este conjunto de facilidades se denomina **Amazon Skills Kit** y, en realidad, es un conjunto de herramientas y APIs que permiten a los desarrolladores crear nuevas habilidades (**skills**) para el asistente virtual que permiten el uso de dispositivos domóticos inteligentes. El primer paso para crear una **skill** consiste en la definición del modelo de interacción, como los usuarios interactúan con la **skill**. Este modelo se define utilizando el Lenguaje de Modelo de Interacción de Alexa (ASK CLI), un lenguaje basado en JSON que describe cómo los usuarios interactúan con la **skill**. Una vez definido este modelo, comienza la creación del código de la **skill** donde se creará la interfaz de interacción con los diferentes dispositivos (Amazon, s.f.).

La llegada de los altavoces inteligentes, como **Amazon Echo y Google Home**, causó un **boom** significativo en el uso de **dispositivos inteligentes y la adopción de soluciones domóticas**. Esto se debe a varias razones clave que han transformado la forma en que interactuamos con la tecnología en nuestros hogares.

En primer lugar, los altavoces inteligentes introdujeron una interfaz de voz natural y accesible que revolucionó la forma en que interactuamos con los dispositivos y sistemas domésticos. En lugar de depender de pantallas táctiles, teclados o mandos a distancia, podemos controlar y acceder a la tecnología simplemente hablando con el altavoz. Esto facilita la interacción, especialmente para aquellos que no están familiarizados con la tecnología o tienen dificultades para usar interfaces convencionales y lo hace realmente atractivo.

El lanzamiento de Google Home fue una respuesta directa al éxito de Amazon Echo (Alexa) ya que demostró que había un mercado para este tipo de dispositivos, y Google, que ya tenía su propio asistente de voz incluido en dispositivos Android (como Siri de Apple). Se dio cuenta de que tenía que competir con Amazon en este espacio. En años posteriores Apple también replicó esta estrategia con el lanzamiento de “Homepod”, su altavoz inteligente.

Además, estos altavoces comenzaron a actuar como un centro de control centralizado para la domótica. Permiten controlar una variedad de dispositivos inteligentes, como luces, termostatos, cerraduras, cámaras de seguridad, electrodomésticos, entre otros, desde un solo lugar. Esto simplifica la experiencia del usuario al proporcionar una forma conveniente de administrar y automatizar diferentes aspectos de la casa con comandos de voz o aplicaciones móviles asociadas. El éxito que comenzaron a tener motivó a todos los fabricantes a apostar por ellos y compatibilizar sus productos con estos altavoces. Esto fomenta la interoperabilidad y facilita la expansión del ecosistema domótico, ya que los usuarios pueden agregar nuevos dispositivos y servicios sin preocuparse por la compatibilidad.

Además, se comenzó a ofrecer funciones de automatización y rutinas que permiten crear escenarios personalizados y automatizar tareas diarias. Los usuarios pueden programar acciones específicas basadas en horarios, eventos o condiciones predefinidas. Por ejemplo, al decir “Buenas noches”, el altavoz puede apagar las luces, ajustar la temperatura y reproducir música relajante, creando una experiencia personalizada y conveniente.

Con todo lo anterior queremos decir que los desarrolladores a partir de este momento encontraron un procesador de lenguaje natural totalmente gratuito que mejoraba sus productos y facilitaba el desarrollo de sus productos. Es decir, a partir de este momento, **el foco se centró en la funcionalidad del propio dispositivo**, dejando a un lado la conexión y protocolos de comunicación. Los dispositivos inteligentes quedaban asociados a los altavoces inteligentes creando un ecosistema donde se podría acceder a ellos mucho más fácilmente. No era necesario centrarse en la intercomunicación entre los diferentes dispositivos, el público interesado en el producto comenzaba a crecer de forma exponencial a partir de este momento.

Figura 2.5 Cómo funciona una *skill* de Alexa

Hasta ese momento, el gran problema que existía respecto a la posibilidad de una extensión masiva de la domótica, entendiendo esta como el uso en todos los hogares, era que cada fabricante (de dispositivos domóticos) tenía una forma de particular de utilizar sus productos. Es decir, **cada producto tenía una forma diferente de interactuar con ellos en función de quién fuese el fabricante**, bien mediante el uso de API propias, controladores totalmente abiertos donde el usuario tendría que programarlos... La aparición de altavoces inteligentes como Alexa Echo y Google Home hicieron que **todos los fabricantes quisieran ser compatibles con ellos**, para aumentar sus ventas ya que el público objetivo crecería, y ocurrió. La concepción de la domótica en el hogar pasó a comprenderse como algo sencillo.

En años anteriores a la aparición de los altavoces como Amazon Alexa o Google Home, desarrollar un hogar inteligente obligaba a enfrentarse a varios problemas. El primero de ellos podría ser el uso de un único proveedor, ya que de esta forma solo habría que adaptar toda la instalación a un comportamiento e intercomunicación definido por el proveedor para todos sus productos, lo que podría significar que una determinada funcionalidad (utilicemos como ejemplo los detectores de humo), no fuese ofrecida por la marca.

La segunda vía consistía en utilizar diferentes proveedores, lo que suponía en la mayoría de los casos que el usuario se haría cargo de averiguar cómo resolver la interoperabilidad entre los diferentes productos. Es decir, si se apostaba por usar diferentes fabricantes (que en la mayoría de los casos usaban APIs propias), el usuario tendría que resolver como intercomunicar todas estas.

La domótica actual por tanto sigue una filosofía “**Plug & Play**”. Comprar, conectar a la red a la red WiFi y comenzar a funcionar. El usuario tendría una sede central donde poder controlar cada uno de sus dispositivos. Es decir, si todos los dispositivos domóticos comparten la misma API y esta es accesible, se podría realizar un software que actúe externamente, de tal forma que independientemente de la marca usada, podría controlarlo todos y que estos interactúen entre ellos sin complicaciones. Iluminación, interruptores, termostatos y una infinidad de nuevos. Esta sede central podría ser su cuenta de Amazon (Alexa), Google (Google Assistant) o su cuenta de Tuya asociada, actualmente como vimos en la comparativa de las APIs, son actualmente los grandes líderes en domótica.

Llegados a este punto, el lector podría pensar que los desarrollos que permite esta interfaz se limitan a la configuración (entendiendo esta como encendido, apagado, temporización...), pero no es así. El conjunto de funcionalidades va más allá, y permite incluso obtener datos médicos válidos a partir del análisis de los datos recopilados.

Como primer acercamiento del entorno que nos encontramos analizando aplicado a la tercera edad, Niño Rivera (2018) propone el desarrollo de una aplicación para el control de la tensión arterial utilizando el asistente Alexa que se ha comentado anteriormente.

En la Figura 2.6 se muestra el diseño finalmente implementado en el que se le permite llevar un control del registro de sus datos médicos a un usuario (en este caso tensión arterial). El funcionamiento es sencillo y en definitiva consiste en un registro de los valores de tensión arterial de un usuario mediante ordenes de voz a los que posteriormente se le da persistencia. En lugar de guardarlo por escrito, se ofrece la posibilidad de almacenarlos en una base de datos. En este mismo proyecto, Niño Rivera (2018) menciona varios estudios futuros bastante interesantes.



Figura 2.6 Arquitectura de interacción entre sujeto y altavoz propuesta para Alexa

El primero de ellos, consiste en realizar un control en tiempo real de los datos tomados. Es decir, una vez se cuente con todos los valores, sería bastante sencillo realizar un estudio de la tendencia de estos, analizar desviaciones que puedan suponer peligro de tal forma que se pueda advertir a los pacientes e incluso llegar a contactar directamente con un equipo médico. Durante todo el estudio, se menciona la tensión arterial, pero sería muy sencillo pivotar esta aplicación a otras patologías de control diario, como bien podrían ser la diabetes, donde un análisis de la tendencia de datos es vital.

A pesar de lo interesante del proyecto, es cierto que las personas de edad avanzada presentan ciertas dificultades en el manejo de las nuevas tecnologías, y más aún si el proceso para realizarlo supone una interacción manual con un dispositivo previamente, ya que, tal y como se plantea, no está automatizado, sino que es el propio usuario quien se toma la tensión y le tiene que indicar al altavoz inteligente los datos mediante comandos de voz. Las personas mayores pueden ser reticentes a todo este proceso, ya que pueden considerar más útil y simple tomar estos valores a mano y guardar los datos en una agenda, por ejemplo. Algo distinto sería si con tan solo un comando de voz como *“tómame las pulsaciones”* se automatizara todo. El estudio, sin embargo, si plantea acceder al historial de datos mediante comandos de voz (*“Alexa, ¿qué tensión tenía ayer?”*).

2.9 Dispositivos domóticos

En esta sección detallaremos los diferentes tipos de dispositivos que serían necesarios para desarrollar una aplicación capaz de monitorizar a una persona mayor en su hogar y detectar tanto problemas que requieren de atención (por ejemplo, caídas o algún tipo de descuido) como recopilar los datos que permitirán descubrir casos o indicios de deficiencia cognitiva.

Para todas las comparativas de precio utilizaremos la extensión Keepa¹⁸, una herramienta que permite rastrear el histórico de precios de productos en Amazon, el seguimiento de ventas y su disponibilidad. La comparativa de precios se realiza mediante gráficas y tablas en los que se detallan la evolución de precios de los diferentes vendedores. El uso de Keepa permite, por ejemplo, identificar cuándo un producto ha rebajado su precio de verdad o simplemente el vendedor ha incrementado su valor días antes para provocar una sensación al consumidor de gran rebaja.

2.9.1 Sensores de movimiento

La idea de los sensores de movimiento es poder realizar un registro de todas las acciones que realiza el usuario. Con los datos recogidos, se pueden analizar los recorridos que una persona hace a lo largo de su hogar para tratar de identificar situaciones extrañas que puedan significar un deterioro cognitivo. A continuación, describiremos cuatro sensores de este tipo y mostraremos una tabla comparativa de sus principales características.

MOES Smart ZigBee PIR Sensor de Movimiento



Figura 2.7 Sensor MOES¹⁹

Este sensor de movimiento tiene como peculiaridad que es compatible con los 3 servicios de Gateway más importante hasta el momento: Alexa, Google Assistant y Tuya.

Entre las características que llaman la atención podemos señalar varias. La primera de ellas es que no necesita estar conectado a la red continuamente, sino que funciona con baterías CR2 de 3V que puede durar hasta dos años. Lo consideramos importante, ya que este

¹⁸ Web oficial de Keepa: <https://keepa.com>

¹⁹ Enlace sensor MOES: <https://amzn.eu/d/ex4CkKb>

hecho facilitará muchísimo la instalación, al permitir al usuario poder elegir cualquier lugar que considere oportuno para instalar el sensor, sin necesidad de depender de una conexión eléctrica cerca, o realizar obras para llevar dicha conexión donde se requiera.

Tiene un ángulo de detección de movimiento variable de entre 120 y 150 grados (el campo visual humano es de 150 grados, como referencia). Además, cuenta con una detección inteligente de presencia humana. Gracias a ello, el sensor es capaz de activarse solo y únicamente cuando detecta un ser humano, con lo cual se descartarán posibles falsos positivos, como el movimiento de mascotas en casa.



Figura 2.8 Precio MOES Sensor

En cuanto a su coste, en el momento de redacción de este documento tiene un precio de 23,99€, con un mínimo histórico de 20,29€, por lo que podría adquirirse aún más barato. Es importante tener en cuenta el precio, ya que para la elaboración de este proyecto un objetivo es que se apueste por el bajo coste.

Yueyang Sensor de movimiento



Figura 2.9 Sensor Yueyang²⁰

²⁰ Enlace sensor Yueyang: <https://amzn.eu/d/fO08Taq>

Este sensor comparte características con anterior (MOES Smart ZigBee), pero a diferencia de aquel, su ángulo de funcionamiento es de solo 120 grados y 5 metros, en contra de los 150 y 15 metros del anterior.

Este sensor se alimenta con 3 baterías de 1.5V, que pueden dar una duración de 5 a 6 meses (en contraposición a los dos años del anterior). Sin embargo, en los comentarios del producto se suceden las críticas respecto a sus problemas de conectividad y sus continuas pérdidas de señal.



Figura 2.10 Precio Yueyang Sensor

En cuanto a su precio (25,99€), es el más caro hasta ahora de la comparativa, y aunque ha rebajado su precio, se encuentra en uno de sus máximos históricos.

Splenssy Sensor de movimiento



Figura 2.11 Sensor Splenssy²¹

Este sensor cuenta con la funcionalidad que requerimos de detección de presencia, como los anteriores, si detecta cualquier tipo de movimiento en su ángulo de detección enviará una alerta. En este caso, cuenta con un ángulo de detección de 115 grados y puede detectar hasta a 9 metros de distancia. Al igual que el primer caso, cuenta con identificación de presencia humana, con lo cual es capaz de diferenciar entre seres humanos y mascotas. Una vez más, este sensor es capaz de funcionar con baterías/pilas con una duración real de unos 5 meses, según los comentarios de los usuarios en Amazon.

²¹ Enlace sensor Splenssy: <https://amzn.eu/d/hinxj4g>



Figura 2.12 Precio Splenssy Sensor

En cuanto a su precio (19,99€), es el más barato de este tipo actualmente en el mercado, manteniendo una tendencia continuista durante los últimos meses, por lo que no se espera ninguna bajada de precio señalada.

PHOVOLT Sensor de puerta

Figura 2.13 Sensor PHOVOLT²²

A diferencia del caso anterior, el mecanismo de detección de movimiento de este sensor es completamente diferente. El dispositivo está compuesto por el conjunto de dos placas alargadas que quedan unidas por un imán cuando se colocan. Cuando éstas se separan, se activa la alarma de posible apertura (entendiéndose ésta como un movimiento).

Para la idea inicial de nuestro proyecto, descartamos este tipo de sensores, porque no detectan el movimiento en sí, sino la apertura de puertas o cajones. Aunque en cierto modo sí puede llegar a ser interesante en ciertas ocasiones, como, por ejemplo, colocarlo en el frigorífico, donde su apertura muy repetitiva durante la noche o su no apertura durante el día podría ser signo de deterioro cognitivo y motivo de alarma. Esta operación puede ser extrapolada a otras situaciones, como cajones de la mesita de noche del usuario donde haya medicación, o cajones de uso habitual.

²² Enlace sensor Phovolt: <https://amzn.eu/d/7dm2B4c>

Una vez más, este dispositivo puede funcionar con baterías externas, por lo que no es necesario que esté conectado continuamente a la red eléctrica. De hecho, es capaz de mandar una alerta cuando se va a agotar la batería, de unos 8 meses según usuarios reales.



Figura 2.14 Precio PHOVOLT Sensor

En este caso, el sensor se encuentra en su máximo histórico de precio (15,19€), pero se ha podido comprar por apenas 10,11€. Es casi la mitad de precio que el sensor anterior, pero en su contra está que ofrece unas prestaciones mucho más limitadas para nuestro propósito, en comparativa con los anteriores.

TABLA COMPARATIVA

Modelo	Ángulo Max.	Distancia Max.	Batería	Precio	Detección Humana
MOES	150 °	15 metros	2 años	23.99 €	Sí
Phovolt	-	-	1 año	15.19 €	No
Yueyang	120 °	5 metros	5/6 meses	25.99 €	No
Splenssy	115 °	9 metros	5 meses	19.99 €	Sí

Tabla 2 – Comparativa de sensores de movimiento

Definitivamente, el mejor sensor es el de la compañía MOES, aunque sí es cierto que sería interesante, como hemos visto en el análisis anterior, el sensor Phovolt para ciertas situaciones, aunque no es un sensor de movimiento como tal.

2.9.2 Sensores de temperatura y humedad

Se pretende usar sensores de temperatura y humedad para identificar situaciones o casos particulares, como sistemas de calefacción/refrigeración mal configurados por el usuario o valores de temperatura y humedad extremos, para una vez más, intentar detectar alguna situación que consideremos extraña o peligrosa.

Los datos recogidos por este tipo de sensores ayudarán a configurar además actuaciones automáticas, como apagar la calefacción en caso de que el usuario lo olvide o incluso avisar a algún tutor o cuidador de que la temperatura en el hogar está aumentando, pero se desconocen las razones por las que se está produciendo (un fuego en el hogar como caso extremo, por ejemplo).

Phovolt Sensor de termómetro higrómetro



Figura 2.15 Phovolt Sensor de termómetro²³

Este sensor es capaz de monitorizar tanto la temperatura como la humedad en tiempo real. Su tamaño es la palma de una mano aproximadamente, por lo que puede ser colocado en cualquier lugar. Aunque la lógica de alertas queda administrada en principio por nuestro *backend* externo, es capaz de configurarse este tipo de alarmas desde el propio sensor. Además, funciona tanto con Google Assistant como con Alexa, por lo que el usuario podrá interactuar con este dispositivo también a través de su dispositivo.

Entre sus características, este dispositivo tiene un margen de funcionamiento que va desde los -20°C hasta 60°C y un rango de medición de humedad que va desde el 0% al 100% de RH (*Relative Humidity*), teniendo una precisión de 1°C para la temperatura y 5% RH para la humedad. Funciona a través de pilas y no es necesaria la conexión a la red eléctrica.

No se proporcionan datos de la batería, pero hay usuarios que reportan hasta 2 meses de duración.



Figura 2.16 Precio Phovolt temperatura

Su precio actual es de 19.26€, siendo actualmente el máximo histórico, ya que llegó a costar tan solo 12.99€. Actualmente el precio lleva bastante tiempo siendo estable, pero podría llegar a aumentar aún más.

²³ Enlace sensor Phovolt: <https://amzn.eu/d/4EGIArX>

Molczov Termómetro Higrómetro Interior



Figura 2.17 Molczov Sensor termómetro²⁴

Cuenta con exactamente las mismas funcionalidades que el producto anterior y también funciona con baterías, por lo que no es necesario conectarlo a la red eléctrica. Los rangos de funcionamiento en este caso son de entre -10°C y 55°C para la temperatura y de entre 10% y 90% RH para los valores de humedad. Sus márgenes de error son iguales al caso anterior, es decir, 1°C para la temperatura y 5% RH para la humedad.

En este caso, hay información mucho más extensa (aportada por un usuario) referente a la batería. Si se utiliza la lectura continuada cada minuto, el usuario indica que no ha sido capaz de durar más de un día el dispositivo encendido. Si se configura el valor máximo de lectura (cada dos horas), la duración de la batería se alarga a los tres meses.



Figura 2.18 Precio Molczov Sensor

El precio del producto es actualmente de 14.99€, siendo su mínimo histórico, por lo que es un buen momento de compra si se decide adquirir este producto.

²⁴ Enlace sensor Molczov: <https://amzn.eu/d/1YpvcFz>

Fengchuang Sensor de humedad & temperatura



Figura 2.19 Fengchuang Sensor termómetro²⁵

Este sensor también comparte características con los otros termómetros analizados previamente. El rango de funcionamiento de este termómetro va desde los -10°C hasta los 50°C en el caso de la temperatura y para la humedad trabaja desde el 0% hasta el 95% RH con unos márgenes de error de 1°C para la temperatura y 5% RH para la humedad. No hay datos de la batería en este caso, pero funciona con el mismo voltaje y características que el caso anterior, por lo que suponemos que será similar.



Figura 2.20 Precio Fengchuan temperatura

El precio actual es de 13.03€ y tuvo un mínimo histórico de 12.65€, por lo que no ha variado mucho su precio a lo largo del tiempo.

Ailgely Termómetro higrómetro



Figura 2.21 Ailgely Termómetro higrómetro²⁶

²⁵ Enlace sensor Fengchuang: <https://amzn.eu/d/1H57n1w>

²⁶ Enlace sensor Ailgely: <https://amzn.eu/d/9nDUAFI>

Este termómetro cuenta con todas las funciones de los anteriores, pero a diferencia del resto cuenta con una pantalla donde visualmente pueden observarse los valores, por lo que no es necesario interactuar con el altavoz inteligente para los valores.

Para este termómetro se informa que tiene unos valores máximos de -10°C a 55°C para la temperatura y de 10% a 99% para los valores de humedad, con una precisión de 0.3°C en el caso de la temperatura y un 3% en el caso de la humedad. Como contra, este termómetro necesita de un dispositivo intermedio (“bridge”) para poder usarlo de forma inteligente, el cual tiene un valor aproximado de 30€ (Ailgely Gateway).



Figura 2.22 Precio Ailgely temperatura

Su precio actual es de 29.99€, a lo que habría que sumarle el precio del Gateway. Se encuentra en un máximo histórico que se mantiene estable desde hace unos meses, habiendo llegado a tener un precio de 19.99€.

TABLA COMPARATIVA

Modelo	Rango temperatura	Rango humedad	Precisión	Precio	Batería Max	Necesita Bridge
Phovolt	-20°C $+60^{\circ}\text{C}$	0-100% RH	$1^{\circ}\text{C}/5\%$ RH	23.99 €	2 meses	No
Molczov	-10°C $+55^{\circ}\text{C}$	10-90% RH	$1^{\circ}\text{C}/5\%$ RH	15.19 €	3 meses	No
Fengchuang	-10°C $+50^{\circ}\text{C}$	0-95% RH	$1^{\circ}\text{C}/5\%$ RH	25.99 €	2/3 meses	No
Ailgely	-10°C $+55^{\circ}\text{C}$	10-99% RH	$0.3^{\circ}\text{C}/3\%$ RH	19.99 €	-	Sí

Tabla 3 – Comparativa entre sensores de temperatura y humedad

Como podemos observar, el mejor producto en relación calidad precio es el modelo Molczov, aunque si es cierto que el resto tienen características prácticamente similares, pero a un mayor precio.

2.9.3 Sensores lumínicos

Los sensores lumínicos serán utilizados para conocer si el usuario suele olvidar apagar luces, dejar ventanas abiertas o cerradas a horas no acordes a lo normal, o incluso detectar si los usuarios no siguen horarios regulares en sus rutinas (apagar o encender luces, abrir ventanas a una determinada hora...).

Kokonm Sensor de luz



Figura 2.23 Kokonm sensor de luz²⁷

Se trata de un sensor capaz de comparar diferentes rangos lumínicos en una escala de 1000 valores. Es decir, el resultado obtenido es un valor en ese rango, siendo la unidad de medida usada LUX. Tiene un rango de funcionamiento de 180°, superior al campo visual humano.



Figura 2.24 Precio Kokonm

Su precio actual es de 23.78€, y actualmente es el mínimo historio del producto, por lo que es un buen momento para su compra.

Lechnical Sensor de luz

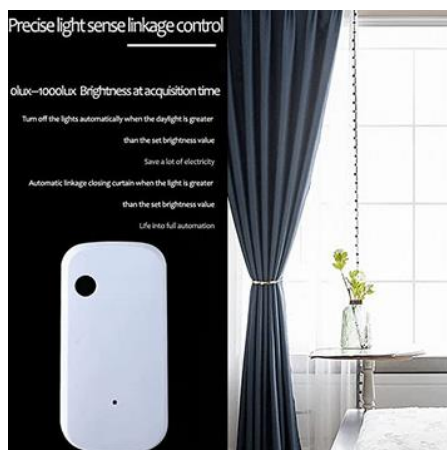


Figura 2.25 Lechnical sensor de luz²⁸

²⁷ Enlace sensor Kokonm: <https://amzn.eu/d/8hIRpOU>

²⁸ Enlace sensor Lechnical: <https://amzn.eu/d/0qETUZY>

Este sensor no tiene prácticamente ninguna diferencia con el anterior. Parece el mismo sensor, pero con otro branding y un precio algo mayor, pero es el más vendido de Amazon.

La única diferencia apreciable con el anterior es que, según la descripción del producto, alcanza un nivel máximo de 1500 LUX de detección de potencia lumínica.



Figura 2.26 Precio Lechnical

En cuanto a su precio, actualmente es de 25.23€, manteniéndose estable desde marzo de este año, aunque su mínimo histórico es de 23.95€. Una variación nada significativa, aunque, como en cualquier otro producto, podría producirse una variación significativa en cualquier momento.

Juwaacoo Sensor de luz



Figura 2.27 Juwaacoo sensor lumínico²⁹

Este sensor, igual que los anteriores, es compatible con todo el ecosistema Tuya, pero a diferencia de ellos, tiene mucha mayor sensibilidad, ya que es capaz de detectar hasta los 30000 LUX de potencia, 29000 LUX más que los sensores anteriores.

El proveedor no proporciona datos de duración de batería, simplemente que utiliza pilas, pero no aclara cuál es el tiempo en funcionamiento sin cambiarlas con el que es capaz de funcionar.

²⁹ Enlace sensor Juwaacoo: <https://amzn.eu/d/fxZgDpX>



Figura 2.28 Precio Juwaacoo

Su precio actual es de 28.26€, siendo su máximo histórico y en un período de encarecimiento de precios, siendo su mínimo histórico 25.67€. A pesar de la diferencia de precio, sus características son muy superiores a la de sus rivales, ya que por apenas un poco más de coste se consigue un aumento en sensibilidad de nada menos que 29000 LUX.

TABLA COMPARATIVA

Modelo	LUX Máximo	Precio	Batería Max
Kokonm	1000	28.26 €	2 meses
Lechnical	1000	25.23 €	3 meses
Juwaacoo	30000	28.26 €	2/3 meses

Tabla 4 – Comparativa entre sensores de luminosidad

En este caso, el modelo *Juawaacoo* a pesar de tener un mayor coste, merece la pena realizar la inversión por la diferencia en LUX máximo con el que opera.

2.9.4 Sensores adicionales

En los apartados anteriores, se han discutido el uso de sensores para medir factores tan habituales como pueden ser movimientos, temperatura o luminosidad. Hoy en día se pueden tener en cuenta muchas más variables que podrían ser interesantes de incluir para los estudios de deterioro cognitivo. Por ejemplo, analizadores de la calidad del aire, que supondrían conocer si se está llevando a cabo una buena ventilación del hogar, o incluso prevenir situaciones de peligro, como dejar el gas abierto.

En el trabajo de Guerrero Ulloa et al. (2023), se presenta el desarrollo del sistema IdeAir, una solución de bajo costo y eficiente en energía para medir la calidad del aire interior y alertar a las personas cuando el nivel de toxicidad es peligroso. El sistema consta de un dispositivo para medir la calidad del aire, servicios web y una aplicación móvil. Dispositivos como éste podrían ser añadidos a un hogar inteligente como otra fuente de datos para retroalimentar sistemas de análisis en busca de signos de deterioro cognitivo. Otro tipo de sensor interesante sería el de consumo eléctrico. Existen multitud de sensores que permiten ver el consumo de un aparato en tiempo real, lo que permitiría controlar y vigilar dispositivos que podrían originar algún tipo de peligro (estufas, cocinas eléctricas, secadoras...).

2.9.5 Actuadores

Relés

MOES Interruptor Wifi Mini Inteligente



Figura 2.29 Relé inteligente MOES³⁰

Este módulo es adecuado para cualquier toma de pared de Reino Unido y de la Unión Europea con un tamaño muy compacto que facilita la instalación. Es compatible con otros interruptores inteligentes para permitir la asociación de múltiples controladores y también se puede utilizar como de 1 o 2 vías (2 interruptores controlan 1 luz). Admite comandos de voz ya sea usando Amazon Alexa o Google Assistant. Es compatible además con Tuya Smart.



Figura 2.30 Precio relé MOES

Actualmente su precio es de 20,99€ marcando uno de sus máximos, pero llegó a poderse comparar por 17,59€ anteriormente.

proAMEDEN Interruptor inteligente

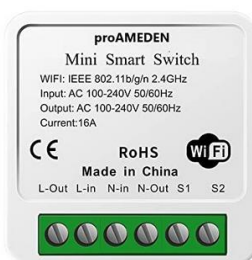


Figura 2.31 Relé proAMEDEN³¹

³⁰ Enlace de compra de relé inteligente MOES: <https://amzn.eu/d/eNtrS42>

³¹ Enlace de compra de relé proAMEDEN: <https://amzn.eu/d/fQftDJu>

Es un módulo de interruptor inteligente de pequeño tamaño. Como en el caso anterior, es compatible su uso mediante comandos de voz a través de Amazon Alexa o Google Assistant así como con la aplicación de Tuya Smart.

Tiene exactamente las mismas características que el relé anterior, en este caso, también puede usarse en 1 o 2 líneas.



Figura 2.32 Precio relé proAMEDEN

Actualmente tiene un precio de 13,99€, manteniéndose esta cifra desde diciembre del año 2022, por lo que su precio es bastante estable. Al **compartir exactamente las mismas características** y ser más barato, **se ha decidido utilizar este producto**. Sin embargo, debido a razones de coste, finalmente en la implementación de la prueba de concepto no se ha llegado a utilizar.

Iluminación

BSEED Interruptor de Sensor Táctil



Figura 2.33 Interruptor de sensor táctil BSEED³²

Compatible con Tuya Smart, Amazon Alexa y Google Assistant permite transformar un interruptor tradicional en uno táctil que además puede ser controlado remotamente.

³² Enlace de compra interruptor BSEED: <https://amzn.eu/d/df1fuUF>

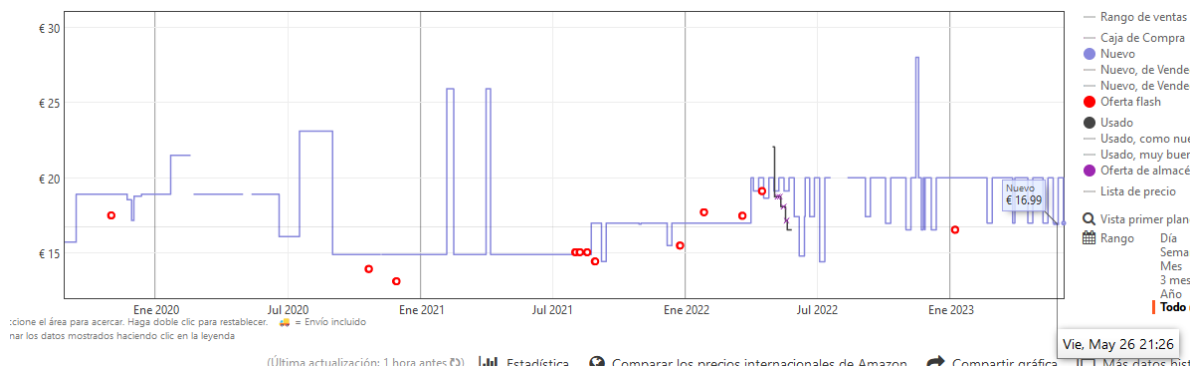


Figura 2.34 Precio interruptor BSEED

Su precio actual es muy inestable y varía entre 16,99€ y 19,99€. Desde julio del año 2022 como puede apreciarse en la gráfica su precio mantiene una tendencia alcista.

Tira led inteligente WiFi FACTORLED



Figura 2.35 Tira LED FACTORLED³³

Control con Amazon Alexa, Google Assistant y Tuya Smart. Cuenta con regulación tanto de la temperatura de la luz (desde luz blanca hasta luz cálida) y colores RGB. Además, incluye mando a distancia y transformador de 12v para su funcionamiento a 220V. La tira tiene 5 metros de longitud y 12 milímetros de ancho.

³³ Enlace de compra de FACTORLED: <https://amzn.eu/d/5UBoKUX>



Figura 2.36 Precio tira LED FACTORLED

Actualmente tiene un precio de 26,95€, pero sigue una tendencia alcista que no para de crecer desde mayo de 2022.

2.9.6 Pulseras inteligentes

La tecnología, tanto de los relojes como de las pulseras inteligentes, ha avanzado significativamente en los últimos años, ofreciendo un conjunto de características y funciones muy variadas que llaman la atención de los usuarios, muchas de ellas centradas en la monitorización de su salud y bienestar. En particular, nos gustaría hacer una especial distinción del Apple Watch y del Samsung Galaxy Watch, ya que son dos de los relojes inteligentes más populares y avanzados para cada ecosistema (Apple y Android, respectivamente). Éstos tienen el potencial suficiente para ser herramientas valiosas en la propuesta del control asistencial que presentamos.

Además, tanto el Apple Watch como el Samsung Galaxy Watch cuentan con una amplia gama de sensores y tecnologías, como acelerómetros, giroscopios, sensores de luz y GPS, que pueden utilizarse para medir y recopilar datos sobre la actividad y el movimiento de una persona. Estos datos pueden ser utilizados para detectar patrones y cambios en la actividad física y la movilidad de una persona, lo que podría ser útil para detectar problemas cognitivos y de movilidad tempranos.

Una de las características más destacadas de estos relojes es su capacidad para monitorizar de manera precisa y continua el ritmo cardíaco de una persona, lo que puede ser especialmente útil en la detección de problemas cardíacos, ansiedad o estrés. Además, estos relojes inteligentes también pueden medir el nivel de actividad física de una persona, la cantidad de calorías quemadas y la calidad del sueño, lo que permite a los profesionales médicos y cuidadores tener una imagen más completa de la salud general de una persona.

Otra característica interesante es que estos relojes inteligentes pueden integrarse fácilmente con otras aplicaciones de salud y bienestar, lo que permite una monitorización más exhaustiva y personalizada. Por ejemplo, aplicaciones de seguimiento de la dieta, la ingesta de agua y el sueño pueden integrarse con los datos recopilados por el reloj para proporcionar una visión completa de la salud y el bienestar de una persona. Respecto a esto último, hay que hacer una diferenciación entre el dispositivo de Samsung y el de Apple.

En el caso de Samsung, hay una manera de acceder a los datos, pero no consiste en una API estándar, y el proceso para obtener los datos se complica un poco, porque Samsung impide acceder a estos datos directamente, al no almacenarlos en el cloud, sino localmente

en el dispositivo. En su lugar, hay que construir una aplicación intermedia que actúe a modo de *bridge*, ya que, para que se habilite el acceso, solo es válido solicitarlos desde el dispositivo móvil al que se encuentra conectado el reloj, actuando como si fuese una llave de acceso (Samsung, s.f.). A partir de ese momento, se pueden acceder a los valores que se muestran en la Tabla 5 extraída de la web oficial de Samsung³⁴.

Available data
All steps → Recuento de pasos
Blood glucosa → Valor de glucosa en sangre
Blood oxygen saturation → Saturación de oxígeno en sangre
Blood pressure → Presión sanguínea
Exercise sesión → Calorías consumidas, actividad del ejercicio y distancia recorrida
Exercise (heart rate) → monitorización cardiaca
Exercise (power) → Intensidad del ejercicio (en función de la actividad seleccionada)
Exercise (speed) → velocidad del ejercicio
Exercise (VO2max) → cantidad máxima de oxígeno en sangre
Heart rate → monitorización cardiaca durante el día
Sleep sesión → registros de sueño
Sleep stage → etapas del sueño
Weight / Body composition → peso y composición corporal

Tabla 5 – Datos proporcionados por el Samsung Watch 5

Por otro lado, con Apple pasa exactamente igual, los datos no se almacenan en la nube, por lo que localmente hay que diseñar algún tipo de aplicación para que el dispositivo que se encargue de almacenarlos. En este caso, la documentación se expone en el llamado “*HealthKit*”, donde se proporciona la información de cómo acceder a los datos (Apple, 2023).

Para el desarrollo de una aplicación para este dispositivo, Apple requiere usar un ordenador de la compañía, los cuales son bastante costosos.

³⁴ Documentación de los datos recogidos por el Samsung Galaxy Watch 5: <https://developer.samsung.com/health/blog/en-us/2022/12/21/accessing-samsung-health-data-through-health-connect>

Apple Watch



Figura 2.37 Apple Watch Series 8³⁵

El Apple Watch es un reloj inteligente que fue lanzado por Apple (el Watch Series 1) en 2015. Desde la primera versión, el dispositivo tiene una pantalla táctil OLED, con un sistema operativo propio (*watchOs*) que permite utilizar el dispositivo mediante gestos, toques y la corona digital.

Como hemos nombrado anteriormente, el reloj recopila multitud de datos del usuario como conteo de pasos, frecuencia cardíaca, presión arterial, electrocardiogramas, seguimiento de automático de actividades físicas y sueño y multitud de más funcionalidades. Además de mostrar la hora, este reloj permite responder llamadas, enviar mensajes de texto y diversas aplicaciones compatibles.

Samsung Galaxy Watch 5



Figura 2.38 Samsung Galaxy Watch 5³⁶

Al igual que el dispositivo de Apple, el Galaxy Watch es una serie de relojes desarrollada por Samsung. Cuenta con una pantalla AMOLED táctil, que cuenta con el sistema operativo

³⁵ Apple Watch Series 8 página oficial: <https://www.apple.com/es/apple-watch-series-8/>

³⁶ Samsung Galaxy Watch 5: <https://www.samsung.com/es/watches/galaxy-watch/galaxy-watch5-40mm-graphite-bluetooth-sm-r900nzaaphe/>

Android Wear. Ofrece diversas funciones de seguimiento de la salud, como la medición de la frecuencia cardíaca, el seguimiento del sueño, el conteo de pasos, el monitoreo del estrés y la detección de caídas. También cuentan con funciones de seguimiento de ejercicio y deportes, incluyendo GPS incorporado.

Empática E4



Figura 2.39 Empática E4³⁷

La *Empática E4* es un dispositivo de muñeca portátil diseñado para medir y registrar datos fisiológicos. Es fabricada por la empresa Empática que está especializada en tecnología de monitoreo de salud y bienestar.

En concreto, esta pulsera está diseñada para capturar datos biométricos en tiempo real, incluyendo entre estos datos la frecuencia cardíaca, la variabilidad del ritmo cardíaco, la sudoración, la temperatura de la piel y acelerómetro para el control del movimiento y la actividad física.

Normalmente, este dispositivo es usado principalmente para la investigación científica, psicología, salud mental, entrenamiento deportivo y el bienestar personal.

La Empática E4 cuenta con numerosos sensores de los que carece los dispositivos de Samsung y Apple, sin embargo, para la propuesta que se presenta en apartado 3, son más interesantes los que cuentan estos dos últimos dispositivos.

Tabla comparativa

M.	Frec. Card.	Presión arterial	ECG	GPS	Batería	Precio	SO	Pasos
Samsung	Sí	Sí	Sí	Sí	2 días	219€	Android Wear	Sí
Apple	Sí	Sí	Sí	Sí	2 días	269€	watchOS	Sí
Empática E4	Sí	No	No	No	1.5 días	700€	Sistema propio (sin nombre)	No

³⁷ Empática E4: <https://www.empatica.com/en-eu/research/e4/>

2.10 Trabajos Relacionados

En esta sección se detallarán alguno de los proyectos que comparten o bien características a nivel técnico que encargarían con nuestra propuesta o bien la propia propuesta es similar a la que se pretende lograr en este proyecto.

En la publicación de Delgado Sanchis (2016) se presenta un proyecto cuyo planteamiento (a nivel tecnológico), puede ser muy similar a lo que se puede requerir en el control y monitorización de personas mayores. En la Figura 2.40 se muestra el esquema de la aplicación.

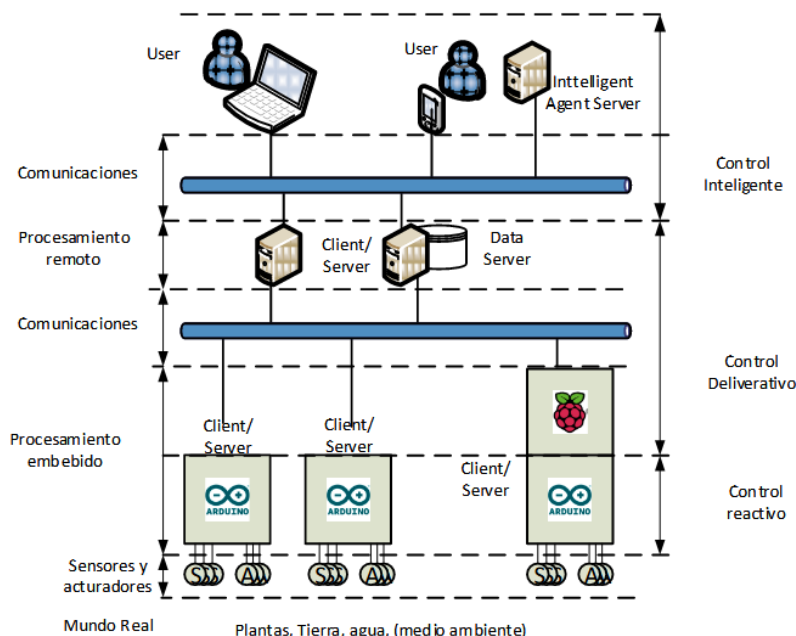


Figura 2.40 Diagrama de la aplicación del proyecto de Delgado Sanchis (2016)

Por un lado, nos encontramos con la capa reactiva. El conjunto de elementos del jardín cuenta con una **serie de sensores y actuadores** conectados a una placa Arduino, que es la encargada de consultar los datos y desencadenar las diferentes acciones por medio de los actuadores respectivamente.

Como bien se indicó anteriormente, los sensores están conectados a una placa Arduino que los gestiona. A su vez, esta placa Arduino se conecta a una Raspberry que se encargará de la **publicación de los diferentes datos a través del protocolo MQTT**. Estos datos que se publican al servidor tienen dos funciones principales.

Por un lado, el estudio de los datos desencadenará la acción de los diversos actuadores, en este caso por ejemplo podría ser la activación de un sistema de riego, algún tipo de protección contra el exceso de sol o un sistema de humificación.

De manera simultánea, cada publicación en **la cola MQTT puede ser consultada por el usuario en un servidor web**. Los datos publicados son adaptados de tal forma que mediante una visión rápida puede conocer cuando se activaron los actuadores, el estado

en tiempo real de los diferentes elementos del jardín e incluso activar de forma manual algún tipo de actuador si, según la opinión del usuario, ante los datos recogidos por los sensores es necesario realizarlo.

Aunque parezca un poco raro a primera vista, **es el mismo esquema que se necesita para monitorizar a personas mayores**. Dispositivos que planteen la monitorización de algo, una capa inteligente que se haga cargo del estudio de los datos recopilados durante la monitorización y finalmente una actuación tras el análisis de los datos.

Por otro lado, la publicación Vargas & Salvador (2016) puede ser un ejemplo de la combinación de todos los elementos en un mismo sistema (ver la Figura 2.41). En esta propuesta, se hace un claro énfasis al control de personas dependientes. La función que se plantea es la **instalación de un conjunto de sensores repartidos a lo largo del hogar e intercomunicados entre ellos** (con una clara apuesta por la interoperabilidad a pesar de ser heterogéneos entre ellos) de tal forma que al procesar los datos obtenidos junto con los registros de los *wearables* puedan realizarse actuaciones externas. Es decir, plantea construir un espacio vigilado mediante sensores que permitirá monitorizar la movilidad de los mayores dentro de un hogar y desencadenar funciones. Por ejemplo, algunas de estas funciones pueden ser encender la luz o incluso puertas, mediante el uso de actuadores según esta propuesta.

Además de las funciones de actuación, también se destaca la posibilidad de notificar de situaciones de riesgo para la salud, como los casos en los que se produce una caída en la que instantáneamente se contacte con el servicio médico.

En concreto, para esta propuesta se utilizan **tres tipos de sensores**. Los de presencia, colocados estratégicamente a lo largo del hogar para conocer la ubicación de las personas, los ambientales, y por último los portátiles, que realmente son acelerómetros que permitirían conocer caídas.

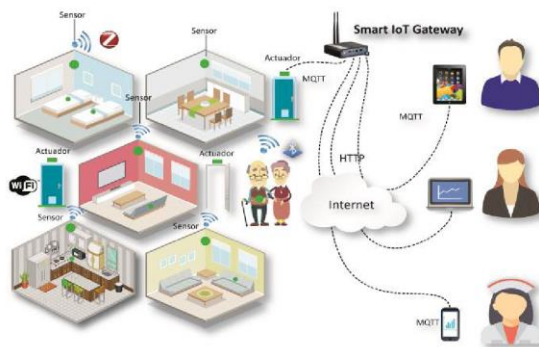


Figura 2.41 Propuesta de hogar inteligente

Hasta ahora, esta es la propuesta más acorde con el propósito de monitoreo que se pretende conseguir. Sin embargo, según nuestra opinión, encontramos una serie de problemas que describiremos a continuación.

En primer lugar, la identificación de usuario a monitorizar en hogares en los que viven más personas. Como bien se indica en el artículo, casi la totalidad de las personas mayores, a

pesar de tener dificultades, prefieren seguir viviendo en sus hogares. En la gran mayoría de casos, se conforma una unidad familiar compuesta por dos personas, por tanto, ¿es capaz de identificar el sistema quién es la persona?

Este problema se puede afrontar utilizando sistemas tan complejos como identificación de rostro mediante inteligencia artificial, o utilizando algo mucho más sencillo, como dispositivos *wearables* que, mediante balizas de proximidad bluetooth, identifiquen al sujeto.

El otro problema que podemos discutir es la interoperabilidad. Aunque es cierto que el diseño es completamente válido y funcional, basarlo todo en una implementación MQTT requiere de cierta programación. Aunque el consumo de “*topics*” tal y como actualmente funciona en *MQTT* es muy básico y sencillo, requiere programar los dispositivos. Por eso, nunca será tan sencillo para un público general, y mucho menos en el entorno al que va dirigido este estudio. Por ello, se debe presentar al usuario un sistema totalmente transparente, no como en los inicios de la domótica donde el usuario era incluso en el encargado de diseñar toda la lógica. Es decir, en nuestro caso queremos proponer un sistema en el que el usuario pueda comprar cualquier sensor del mercado y que la única configuración que tenga que realizar sea conectarlo a la red *WiFi* y asociarlo a su cuenta, no queremos obligarlo a comprar un determinado desarrollo hecho por nosotros o que tenga que hacer algún tipo de modificación, como configurar los *topics MQTT*. Como podrá leer en el apartado de propuesta, simplemente bastará si el sensor es compatible con Tuya, nuestro sistema se encargará del resto.

La orientación que se busca en nuestro caso es de la de “*Plug & Play*”. El usuario solo deberá conectar a la red al dispositivo, asociarlo a su cuenta y podrá desentenderse de él. Ningún tipo de configuración adicional será necesaria en un entorno ideal.

Tabla comparativa

<i>Característica</i>	<i>Delgado Sanchis (2016)</i>	<i>Vargas & Salvador (2016)</i>
Sector objetivo	Agricultura	Sociosanitario
Uso MQTT	Sí	Sí
Uso de API	Sí	No
Número diferente de sensores utilizados	2	3
Número diferente de actuadores utilizados	3	1
Acceso remoto	Sí	Sí

Tabla 6 – Comparativa entre trabajos relacionados

3 PROPUESTA

3.1 Descripción

La **detección temprana de problemas cognitivos** en personas mayores es esencial para poder brindarles un tratamiento efectivo y mejorar la calidad de vida de aquellas que podrían llegar a estar afectadas. Sin embargo, como hemos comentado en las secciones anteriores, esta detección puede resultar **especialmente difícil en el caso de personas mayores que viven solas**, ya que, al no tener un seguimiento continuo, los problemas solo son detectados en etapas muy avanzadas. En suma, **situaciones de peligro**, como caídas, pueden ser peligrosas para aquellos que viven solos y **queremos ser capaces de detectarlas rápidamente**, para poder actuar lo más pronto posible en los casos de accidentes graves.

Por esta razón, es vital contar con un sistema de control asistencial que permita **monitorizar de forma continuada a las personas** y conseguir nuestro objetivo final, que es detectar los posibles casos de problemas cognitivos y asistir a las personas mayores en caso de que se produzca algún tipo de peligro. El sistema inteligente que proponemos contará con diferentes **dispositivos domóticos**, conformados por sensores de movimiento y **un reloj inteligente** que proporcionará datos del usuario, tales como frecuencia cardíaca, temperatura corporal, monitorización del sueño...

Con la información recopilada, se utilizarán técnicas de **análisis de datos basadas en reglas estáticas inicialmente**, aunque en un futuro se podría hacer evolucionar el sistema, aplicando técnicas de inteligencia artificial, como *machine learning*, con la que se intentarán detectar casos señalados por los especialistas como de especial relevancia, en el sentido de que puedan significar una señal de posible deterioro cognitivo.

Concretando aún más, se propone un sistema basado en diferentes sensores compatibles con el proveedor **Tuya**. Se ha decidido utilizar este proveedor por dos razones principales. La primera de ellas es porque Tuya es el principal proveedor y fabricante chino; con ello queremos señalar que la mayoría de los dispositivos vendidos en comercios como Amazon son compatibles con este fabricante, y uno de los principales objetivos de nuestro proyecto es **garantizar la máxima interoperabilidad y compatibilidad** con dispositivos actuales, tratando de evitar productos poco comunes. La segunda razón es porque la documentación de Tuya es la más extensa y actualizada, permitiendo realizar más acciones distintas de una forma mucho más fácil que la proporcionada por sus competidores, como Google o Amazon.

En nuestro diseño (véase la Figura 3.1), los datos serán recopilados a través de un *backend* externo (nuestra aplicación puede ser desplegada en un servidor local o en la nube), que contará con **tres microservicios principales**. El **primer microservicio** se encarga de **escuchar continuamente los cambios** que se produzcan en los diferentes sensores que se utilicen, como los de temperatura, movimiento y luz (aunque para nuestra prueba de concepto solo se utilizara los de movimiento y un reloj). Si se produce algún tipo de cambio, se lanzará un evento que se registrará en un sistema de persistencia de alta velocidad, como Redis. Es decir, el microservicio se encargará de captar los cambios en el sensor de

movimiento y publicará el evento detectado en una cola de Redis, siendo ésta una base de datos en memoria que se utilizará como intermediaria para la comunicación entre los diferentes microservicios.

El **segundo microservicio**, ante un nuevo evento, **analizará los datos** y, mediante las técnicas de análisis que comentamos, se encargará de **detectar patrones anormales** en el comportamiento de los sujetos. En nuestro caso particular, se pueden utilizar reglas fijas para detectar estas situaciones, como, por ejemplo, que se entre y salga de una habitación 5 veces seguidas en un período breve de tiempo. Además, en función de las reglas implementadas, es decir, si se cumplen ciertas condiciones, se publicarán eventos para que entre en acción el tercer microservicio.

Finalmente, el **tercer microservicio se encargará de actuar** como se haya programado si se detecta alguna anomalía.

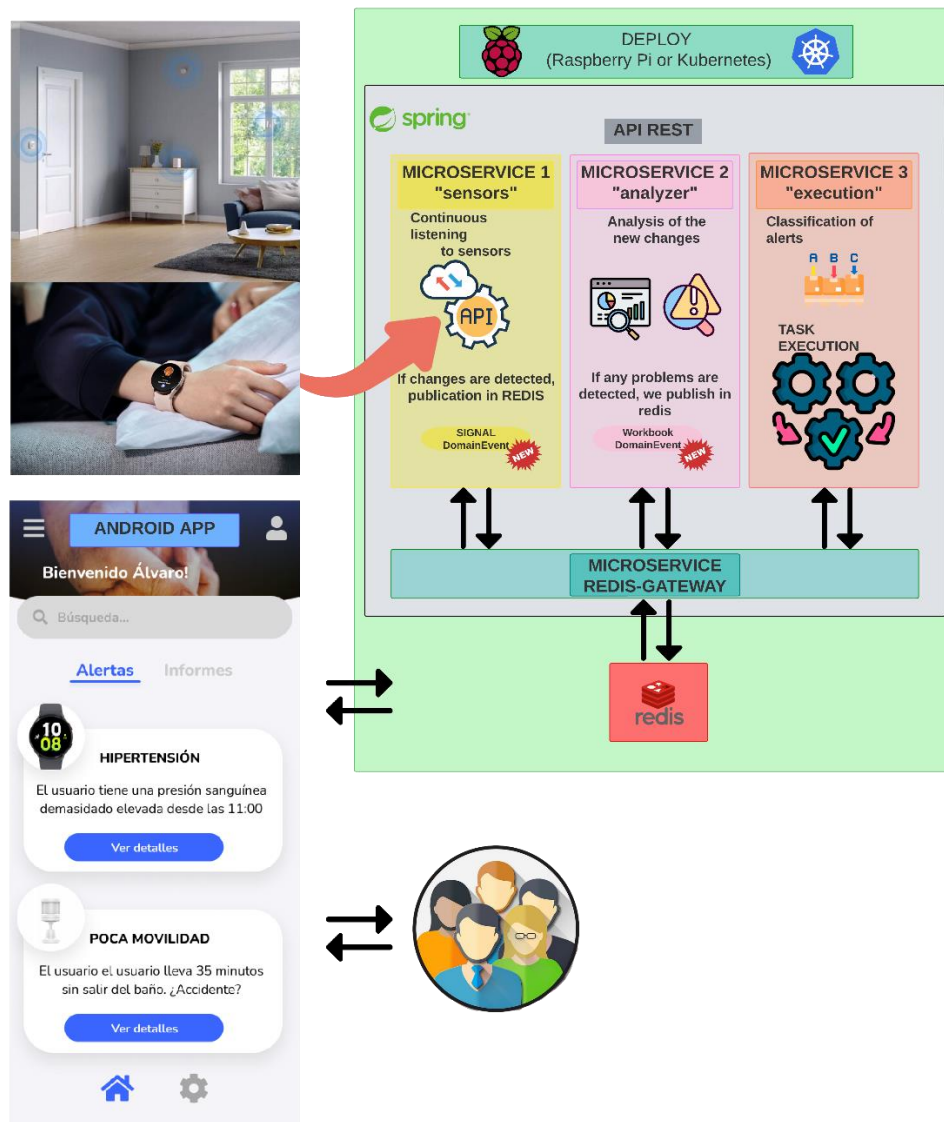


Figura 3.1 Diagrama de la arquitectura del sistema propuesto

Esta actuación se llevará a cabo en tres niveles:

- **Seguridad pasiva:** se pretende que, en casos donde el segundo microservicio identifique una situación de riesgo, nuestro sistema reaccione antes de que ocurra un problema. Por ejemplo, si se detecta que el usuario comienza a hacer recorridos extraños durante la noche, se irán encendiendo las luces automáticamente, generando un recorrido hasta un lugar seguro (Tuya permite el posicionamiento relativo de los productos dentro del hogar) y proporcionando alguna distracción (como encender una televisión) para que cese su recorrido. Por el momento, esta seguridad pasiva consistirá en lanzar algún tipo de alerta al cuidador o tutor de la persona que use nuestro sistema.
- **Informes analíticos:** con toda la casuística de los datos recopilados, y tras la acción del segundo servicio, se realizará un informe final que muestre si hay posibilidad de que haya un deterioro cognitivo.
- **Aplicación de control:** aplicación móvil que podrá ser utilizada por la familia, con la que podrá nivelar los ajustes para cada usuario, considerando cambios en su comportamiento y tratando de evitar falsos positivos que se detecten en el segundo microservicio de forma errónea. Esta aplicación servirá para llevar un control manual de las diferentes notificaciones de los sensores, así como también se podrán recibir notificaciones a través de ella.

Como dato importante, el usuario deberá fijar en esta aplicación dónde se encuentra cada sensor y actuador. De esta forma, en caso de que haya que realizar un camino de luces hasta alguna estancia o **analizar el recorrido que está realizando la persona mayor**, podrá usarse esta información. Esto será tan sencillo como indicar el nombre de la habitación donde se encuentre el dispositivo e indicarle al sistema manualmente como están conectadas las habitaciones (ejemplo: “Habitación 1” conecta con “Pasillo 1” y “Pasillo 1” conecta con “Habitación 3”).

3.1.1 ¿Qué innovación aporta?

La propuesta que hemos planteado ofrece varias innovaciones importantes en el campo del control asistencial y la domótica. En primer lugar, la utilización de dispositivos domóticos, que se han vuelto cada vez más accesibles y comunes en los hogares, para recoger datos que ayuden a detectar problemas cognitivos en las personas mayores, es una innovación importante en sí misma.

Tradicionalmente, los controles asistenciales se han realizado mediante la instalación de dispositivos muy específicos y costosos que requerían de una configuración muy personalizada que en la mayoría de los casos es compleja e incluso invasiva. Con esta propuesta, se utiliza la **tecnología ya presente en los hogares y asequible**, como los sensores domóticos y los relojes inteligentes, para realizar el **control asistencial**. Además, será un sistema completamente independiente, ya que no se necesitará la acción

supervisada para el análisis de los datos, sino que el **sistema será capaz de tomar datos, analizarlos y reaccionar de forma autónoma**.

Además, la utilización de una **arquitectura basada en microservicios y la mensajería de eventos** ofrecen varias ventajas técnicas importantes. En primer lugar, se ofrece una mayor escalabilidad del sistema, ya que cada microservicio puede ser optimizado de forma independiente en función de la carga de trabajo, pudiendo aumentar o disminuir la capacidad del servicio en cuestión para satisfacer la demanda, pero manteniendo un rendimiento óptimo. Además, la separación de las diferentes responsabilidades en diferentes microservicios permite mejorar para cada sistema su modularidad, lo que facilita su mantenimiento y evolución.

El uso de **mensajería de eventos** permite que el acoplamiento sea débil entre los diferentes componentes del sistema. Cada servicio puede funcionar de forma independiente, enviando y recibiendo a través de un bus de mensajes, por lo que no se necesita conocer los detalles internos de cada microservicio, facilitando así la evolución y el mantenimiento del sistema a medida que se añadan en el futuro nuevos servicios.

Por último, la propuesta apuesta por el uso de *Tuya API*, en una modalidad de uso no solamente enfocada al control de dispositivo, sino más amplia, siendo capaz de interactuar con un ecosistema completo de dispositivos compatibles. Se diseñará un sistema *Plug&Play* en el que el usuario no tendrá que realizar sucesivas configuraciones sobre el dispositivo. Con tan solo conectarlo a la red WiFi local, nuestro sistema se encargará del resto.

3.1.2 ¿Qué se propone hacer?

Para llevar a cabo esta propuesta, se diseñará una prueba de concepto basada en un sistema de control asistencial basado en sensores que puedan ser controlados mediante la aplicación de Tuya y el reloj inteligente Samsung Galaxy Watch. Al no poseer el reloj por motivos de coste, se simularán los datos a los que, según su documentación, podemos acceder, y que son los que se describen en la Tabla 5.

Tras lo visto en el apartado 2.9 Dispositivos domóticos, de los dispositivos analizados, se ha utilizado realmente *“MOES Smart ZigBee PIR Sensor de Movimiento”*. Idealmente se hubiesen comprado todos, pero por motivos de coste se ha decidido priorizar y solo se ha seleccionado éste y el reloj Samsung, para el simularemos sus resultados, como comentamos anteriormente.

El *backend* externo, como detallamos anteriormente, estará compuesto de tres microservicios que se ejecutarán en un dispositivo de bajo coste, como una Raspberry Pi, cumpliendo con nuestro compromiso de que la **solución no sea de un precio elevado**. Aunque idealmente podría ser desplegado usando *Kubernetes* (la aplicación se “dockerizará”).

El primer microservicio en esta primera fase de prueba de concepto se encargará de recopilar todos los datos del sensor de movimiento y del reloj, publicándolos los eventos en Redis. Respecto al reloj, se creará un servicio que sea capaz de generar datos reales

(simulándolos). Estos datos, según la parametrización (variables utilizadas al llamar al controlador) que se utilicen al llamar al servicio generarán valores aleatorios que estarán por encima o por debajo de los márgenes establecidos como recomendados.

El segundo microservicio analizará los eventos publicados, y tras someterlos a un proceso de análisis identificará situaciones que considere problemáticas y creará nuevos eventos, que analizará el tercer microservicio. En este segundo microservicio, aparte de los datos reales recogidos por el sensor de movimiento, se simularán datos de otros sensores que podrían añadirse al sistema pero que no se ha hecho por **motivos de coste**, como se comentó anteriormente.

Por último, el tercer microservicio se encargará de actuar en función de los eventos que cree el segundo microservicio. Se realizarán acciones que pueden ser, por ejemplo: el encendido de luces, contactar con el cuidador en caso de que sea necesario, y la generación de informes en caso de que detecte alguna anomalía de tipo cognitiva de la persona mayor. Por el momento, la funcionalidad de accionar luces no se desarrollará en esta prueba de concepto.

Además, se diseñará la interfaz de una pequeña aplicación que permita a los cuidadores recibir alertas y configurar las reglas para ajustarlas de un modo más individualizado a cada usuario, de manera que desencadenen las actuaciones y avisos de la forma más adecuada en cada caso.

Debe tenerse en cuenta que se creará un servicio que permita interactuar con cada microservicio y Redis, de tal manera que la comunicación entre los diferentes microservicios se basará únicamente en la creación y consumición de los eventos producidos entre ellos, no existirá un contacto directo como tal.

3.2 Metodología

Para el desarrollo de este proyecto, se podría seguir una metodología de desarrollo ágil, como Scrum, que permitiría una gestión eficiente del proyecto y una mayor adaptabilidad a los cambios que pudieran surgir durante el proceso. A continuación, se describen las etapas que se seguirían en esta metodología para cada *sprint* o iteración:

- **Planificación:** se establecerán los objetivos a alcanzar y se definirán los requisitos del proyecto. Además, se definirán los roles y responsabilidades del equipo y se creará el *backlog* del producto.
- **Sprint planning:** se definirán los objetivos específicos del *sprint* y se seleccionarán las tareas a realizar. También se estimará el tiempo y recursos necesarios para cada tarea.
- **Desarrollo:** durante el *sprint*, el equipo trabajará en las tareas definidas en el *sprint planning* y se realizarán reuniones diarias de seguimiento para revisar el progreso y resolver cualquier problema que surja.
- **Revisión del sprint:** al finalizar el *sprint*, se realizará una revisión del trabajo realizado y se presentará a los stakeholders los resultados obtenidos.

- ***Sprint retrospective***: se llevará a cabo una reunión para analizar el *sprint* y proponer mejoras para el siguiente.

Este ciclo se repetirá en cada sprint, lo que permitirá una gestión eficiente del proyecto y una mayor adaptabilidad a los cambios que pudieran surgir durante el proceso.

Es vital que tener en cuenta la importancia de realizar pruebas y validaciones continuas, para asegurar la calidad del software y minimizar errores en la implementación.

En el contexto de nuestra aplicación, se **desarrollarán 3 iteraciones**, tal y como se describirá en el apartado 3.3 *Desarrollo*. En resumen, el objetivo de cada una de estas iteraciones es modelar, implementar y testear uno de los microservicios principales de nuestra propuesta.

En la primera iteración se desarrollará el microservicio encargado de administrar los sensores de los usuarios, en la segunda iteración el microservicio encargado de analizar los datos recopilados y finalmente en la tercera iteración el microservicio encargado de actuar ante el análisis realizado previamente en el anterior microservicio.

En la siguiente sección se explicará con mucho más detalle cada una de estas iteraciones.

3.3 Desarrollo

El desarrollo se dividió en tres iteraciones, en la que en cada una de ellas se implementó por completo uno de los microservicios principales de la aplicación final. A continuación, se detallará que se realizó en cada una de estas iteraciones. En todos los microservicios que se han desarrollado se ha usado *Spring Boot* y *Maven*. Además, se ha “*dockerizado*” toda la aplicación como posteriormente se explicará.

Todo el código de la aplicación está disponible en <https://github.com/alvarodelaflor/tfm-ugr>. Este repositorio cuenta con una licencia *MIT* (*Massachusetts Institute of Technology*). Bajo esta licencia se permite a cualquier persona utilizar, modificar, distribuir y utilizar el proyecto tanto para fines comerciales como no comerciales.

Además, para esta primera prueba de concepto tan solo se utilizará como dispositivo real el sensor de movimiento “*MOES Smart ZigBee*”. Por motivos de coste se descartó el uso de más sensores, aunque se simuló el comportamiento que tendría el dispositivo *Samsung Smart Watch 5*.

3.3.1 Iteración 1: Microservicio “sensors”

Durante este período, el objetivo principal fue desarrollar un módulo que fuese capaz hacer uso de la *API* de *Tuya Smart* para obtener los datos de los sensores utilizados. En este apartado explicaremos el desarrollo que hemos realizado para construir los dos primeros microservicios, tanto el microservicio “**sensors**” como el microservicio secundario “**redis-gateway**”, este último imprescindible para lograr la comunicación entre el resto de los microservicios.

El primer paso fue conseguir las credenciales de desarrollador de la plataforma, la cuales son necesarias para hacer uso de los diferentes *endpoints* que permite la aplicación. Para

ello, hay que dirigirse a la web de desarrolladores de la compañía conocida como *Tuya IoT Platform*³⁸.

Una vez tengamos acceso a la plataforma necesitaremos obtener las credenciales del usuario (tanto el *accessId* como el *accessKey*), registrar nuestro proyecto personal en la plataforma y por último asociar a este proyecto cada uno de los dispositivos que utilizará, que podrán ser físicos (reales) o pueden ser virtualizados.

El primer lugar nos tendremos que dirigir a la sección *Cloud* y dentro de esta al apartado *Development*, lo que mostrará el panel Dashboard, tal y como puede verse en la Figura 3.2. Aparecerá una nueva ventana donde deberemos crear el nuevo proyecto para nuestra aplicación a partir de una serie de pasos muy sencillos e iterativos usando la opción “*Create Cloud Project*”.

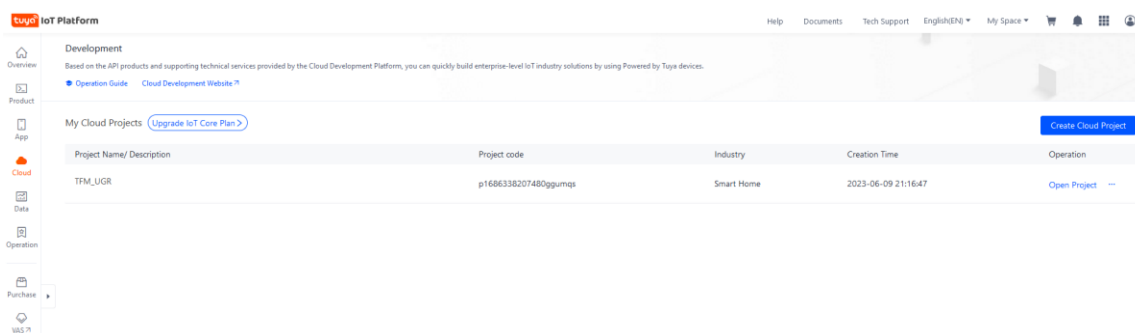


Figura 3.2 IoT Platform Dashboard

Una vez terminado el proceso de crear el proyecto (donde se requerirá información personal del desarrollador), la primera ventana que aparecerá mostrará las credenciales como se muestra en la Figura 3.3. Es muy importante almacenarlas en un lugar seguro, ya que un uso inadecuado podría conllevar a que se aplicasen costes de uso a nuestro proyecto.

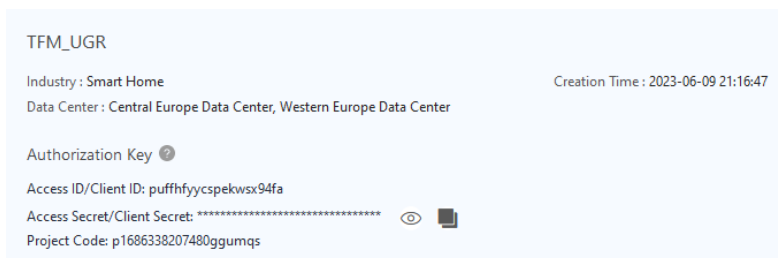


Figura 3.3 Credenciales de la aplicación

Tras obtener las credenciales, podríamos empezar a hacer uso de la API, sin embargo, carece de sentido si aún no hemos añadido dispositivos con los que podamos interactuar.

Para añadir los dispositivos a nuestra aplicación, la plataforma obliga a dar de alta a diferentes usuarios, por lo que varios desarrolladores pueden trabajar en el mismo proyecto

³⁸ Tuya IoT Platform: <https://iot.tuya.com/>

de forma independiente evitando concurrencias en los datos ya que a cada usuario, aunque forme parte del mismo proyecto, se le asigna individualmente un dispositivo.

Para añadir un dispositivo al proyecto existen dos opciones, añadir un dispositivo real o uno virtualizado. La opción de dispositivos virtualizados es muy interesante ya que como puede ver en la Figura 3.4, hay una gran cantidad de opciones disponibles, prácticamente todos los dispositivos que son compatible con Tuya Smart actualmente. Durante el desarrollo del proyecto hicimos uso de esta funcionalidad añadiendo a nuestra aplicación una gran cantidad de dispositivos, pero debido a un uso muy continuado y al gran uso de peticiones sobre la API que se realizaron por medio de un *CRON* (gestiona tareas que deben realizarse de forma regular), nuestra IP fue bloqueada y a partir de ese momento solo se nos permitió realizar consultas sobre dispositivos físicos (reales).

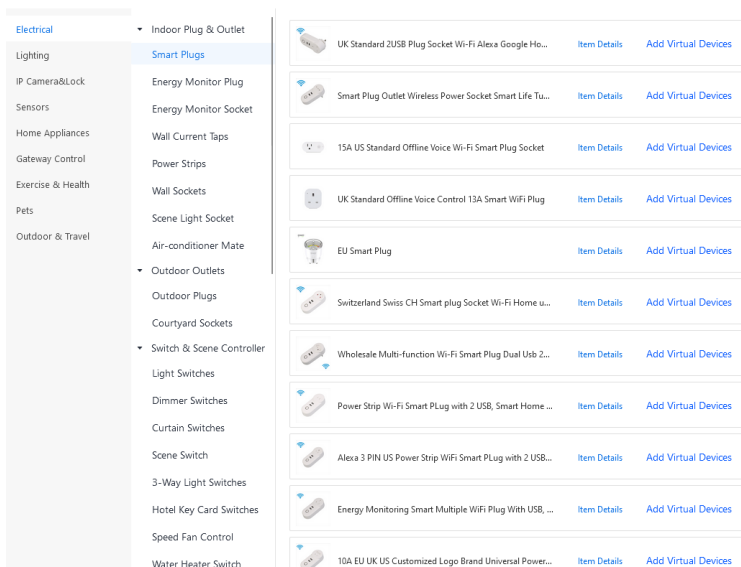


Figura 3.4 Dispositivos virtuales disponibles

Por otro lado, para hacer uso de un sensor real, la plataforma requiere que descarguemos una aplicación en nuestro dispositivo móvil llamada *Smart Industry*. Esta aplicación simula el comportamiento que tendría la aplicación *Tuya Smart* real, pero habilita una serie de características para desarrolladores. Es decir, podemos controlar y acceder a toda la información de nuestros dispositivos mediante la aplicación.

Deberemos iniciar sesión con las credenciales del usuario desarrollador que se crearon anteriormente y añadir el dispositivo mediante la aplicación, a través de la web no es posible. En esta aplicación como puede ver en la Figura 3.5, para cada uno de los dispositivos, puede acceder a ellos para modificar su configuración. Por ejemplo, apagando o desactivando la funcionalidad de detección en caso de los sensores PIR, o bien acceder al conjunto de alertas que se han producido.

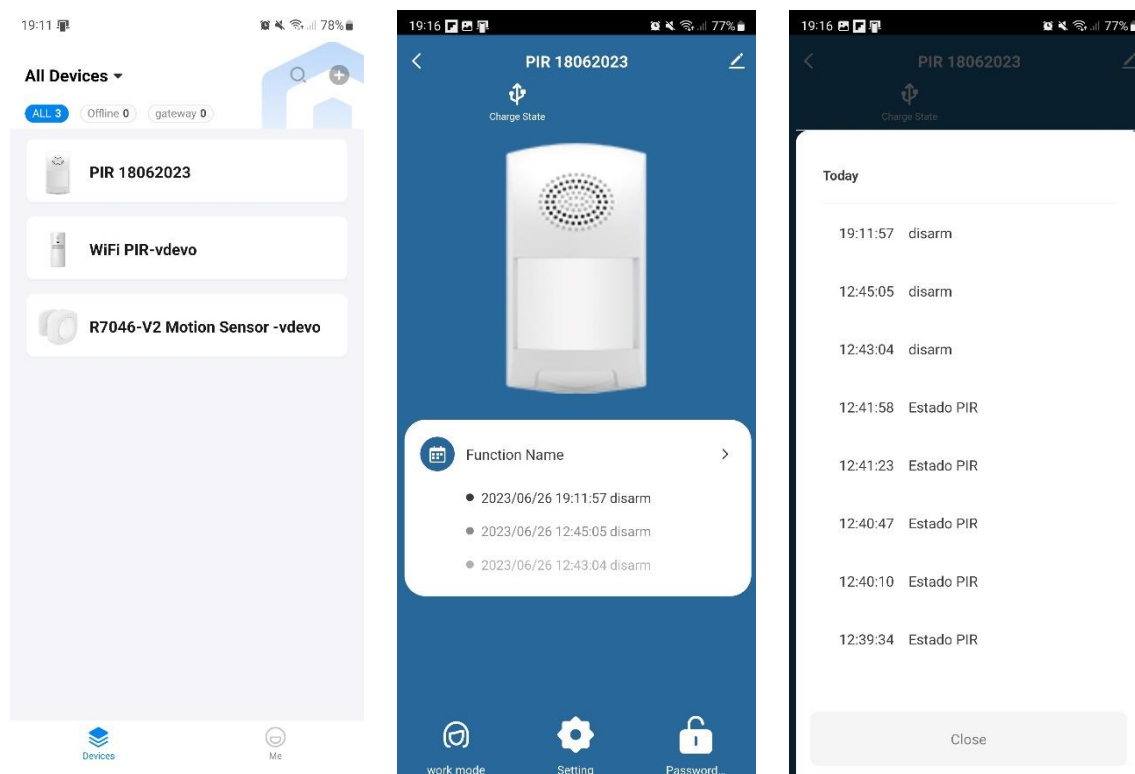


Figura 3.5 UX de Smart Industry

A partir de este momento, tenemos todos los requisitos para comenzar a trabajar con la *API* de forma automatizada. El conjunto de todos los *endpoints* disponibles solo es accesible a través de la plataforma si tiene la sesión iniciada en ella, disponible en la sección *Service API* del panel *Cloud*.

El siguiente paso es ser capaz de conocer todos los dispositivos asociados a la cuenta del usuario, una vez tengamos esta información podremos interactuar con ellos. Gracias al *endpoint Get Device List*³⁹ podemos conocer todos los dispositivos que tiene el usuario asociado a su cuenta y lo más importante de todo, el *ID* de cada uno de ellos.

Sin embargo, para realizar cualquier tipo de consulta a su *API*, *Tuya* requiere que utilicemos un token de autenticación, que será añadido a la cabecera de cada petición a su *API*.

Por tanto, el primer paso que se desarrolló en el microservicio “*sensors*” fue construir el conector *TuyaConnector* que permite hacer consultas a la *API* de forma genérica para reducir al máximo la complejidad de las llamadas.

En la Figura 3.6 puede ver un fragmento del código del conector, en concreto el método *execute*. Con este método tan solo se debe indicar el *access token*, el *path* (endpoint) que se quiere consultar, así como el tipo de *request* y el *body* de este si fuese necesario. Para obtener este *access token*, se ha implementado un método igual de simple “*getToken()*”, al que se llamará cada vez que queramos hacer uso de la *API* y que devolverá el *access token*

³⁹ Get Device List: <https://developer.tuya.com/en/docs/cloud/dc413408fe?id=Kc09y2ons2i3b>

que requiere el método *execute* como parámetro, las cabeceras de autorización para el resto de consultas se crearán de forma automática.

```

Alvaro de la Flor Bonilla
public static String execute(String accessToken, String path, String method, String body, Map<String, String> customHeaders) {
    try {
        // Verify developer information
        if (MapUtils.isEmpty(Constant.CONTAINER)) {
            throw new TuyaCloudSDKException("Developer information is not initialized!");
        }

        String url = Constant.CONTAINER.get(Constant.ENDPOINT) + path;

        Request.Builder request;
        if ("GET".equals(method)) {
            request = getRequest(url);
        } else if ("POST".equals(method)) {
            request = postRequest(url, body);
        } else if ("PUT".equals(method)) {
            request = putRequest(url, body);
        } else if ("DELETE".equals(method)) {
            request = deleteRequest(url, body);
        } else {
            throw new TuyaCloudSDKException("Method only support GET, POST, PUT, DELETE");
        }

        if (customHeaders.isEmpty()) {
            customHeaders = new HashMap<>();
        }

        Headers headers = getHeader(accessToken, request.build(), body, customHeaders);
        request.headers(headers);
        request.url(Constant.CONTAINER.get(Constant.ENDPOINT) + getPathAndSortParam(new URL(url)));
        Request requestBuild = request.build();
        requestToCurl(requestBuild);
        Response response = doRequest(requestBuild);
        return response.body().string();
    } catch (Exception e) {
        throw new TuyaCloudSDKException(e.getMessage());
    }
}

```

Figura 3.6 Llamada a la API Tuya

Tras el desarrollo del conector que acabamos de explicar, realizamos la implementación completa del controlador para Tuya. En concreto se diseñaron cuatro *endpoints* para este controlador (ver la Figura 3.7). En esta documentación no se mostrará el servicio que implementa la funcionalidad del controlador, pero se puede acceder a ella a través del código adjunto a esta memoria.

1. ***getDevices()*** mediante el cual seremos capaces de conocer todos los dispositivos que tiene el usuario asociado a su cuenta. En concreto, este *endpoint* nos es útil para obtener el *ID* de los dispositivos a los que posteriormente accederemos a sus logs.
2. ***getLogsByDeviceId(String deviceId, LocalDateTime startTime, LocalDateTime endTime)*** permitirá dado un determinado dispositivo identificado por su ID, obtener todos los registros (movimientos, interrupciones, pérdida de conexión...) que se tienen de él en el intervalo de una fecha dada.
3. ***getLogs(LocalDateTime startTime, LocalDateTime endTime)*** es una combinación de los dos recursos anteriores, por cada uno de los dispositivos que se

encuentren en el primer método, se le aplica el segundo para obtener los logs de todos los dispositivos asociados a la cuenta del usuario.

4. **`getPirSignal(LocalDateTime startTime, LocalDateTime endTime)`** utiliza el recurso anterior y filtra los resultados, ya que los valores que devuelve la API de Tuya son un poco confusos. Por ejemplo, un *log* con *value* 0 representa que se ha perdido la conexión, mientras que un *value* “pir” indica un movimiento detectado.

```
package com.alvarodelaflor.sensors.web;

import ...

2 usages  Alvaro de la Flor Bonilla
@RestController
@RequestMapping("/tuya/devices")
public class TuyaDeviceController {

4 usages
    @Autowired
    TuyaDeviceService tuyaDeviceService;

    Alvaro de la Flor Bonilla
    @GetMapping("/")
    public DeviceResults getDevices() {
        return tuyaDeviceService.getDevices();
    }

    Alvaro de la Flor Bonilla
    @GetMapping("/{logs/{deviceId}")
    public DeviceLogs getLogsByDeviceId(
        @PathVariable("deviceId") String deviceId,
        @DateTimeFormat(iso = DateTimeFormat.ISO.DATE_TIME, fallbackPatterns = { "dd/MM/yyyy HH:mm" }) @RequestParam(value = "startTime") LocalDateTime startTime,
        @DateTimeFormat(iso = DateTimeFormat.ISO.DATE_TIME, fallbackPatterns = { "dd/MM/yyyy HH:mm" }) @RequestParam(value = "endTime") LocalDateTime endTime
    ) {
        return tuyaDeviceService.getLogsByDeviceId(deviceId, startTime, endTime);
    }

    Alvaro de la Flor Bonilla
    @GetMapping("/{logs}")
    public List<DeviceLogs> getLogs(
        @DateTimeFormat(iso = DateTimeFormat.ISO.DATE_TIME, fallbackPatterns = { "dd/MM/yyyy HH:mm" }) @RequestParam(value = "startTime") LocalDateTime startTime,
        @DateTimeFormat(iso = DateTimeFormat.ISO.DATE_TIME, fallbackPatterns = { "dd/MM/yyyy HH:mm" }) @RequestParam(value = "endTime") LocalDateTime endTime
    ) {
        return tuyaDeviceService.getLogs(startTime, endTime);
    }

    Alvaro de la Flor Bonilla
    @GetMapping("/{pirSignal}")
    public List<TuyaPirSignal> getPirSignal(
        @DateTimeFormat(iso = DateTimeFormat.ISO.DATE_TIME, fallbackPatterns = { "dd/MM/yyyy HH:mm" }) @RequestParam(value = "startTime") LocalDateTime startTime,
        @DateTimeFormat(iso = DateTimeFormat.ISO.DATE_TIME, fallbackPatterns = { "dd/MM/yyyy HH:mm" }) @RequestParam(value = "endTime") LocalDateTime endTime
    ) {
        return tuyaDeviceService.getPirSignals(startTime, endTime);
    }
}
```

Figura 3.7 Controlador TuyaDeviceController

Llegados a este momento éramos capaces de detectar cualquier movimiento con un sensor Tuya, y podíamos hacer pruebas ya que contábamos con él físicamente. Dada la posibilidad de virtualizar dispositivos decidimos tomar ventaja de ello y hacer uso de todos los sensores e incluso relojes inteligentes que permitía la plataforma.

Sin embargo, como comentamos anteriormente, nuestra IP fue bloqueada y no podíamos usar dispositivos virtualizados **de la plataforma Tuya**. Debido a que solo mediante el uso de sensores de movimiento es prácticamente imposible detectar casos de deterioro cognitivo se decidió que, aunque no se disponía del reloj Samsung Galaxy Watch 5 físicamente, podríamos optar por simular nosotros mismos en nuestro microservicio los datos de este reloj (qué es más conocido que el ofrecido por Tuya) e incorporarlos a nuestro sistema (ver 2.9.6 Pulseras inteligentes). Es decir, aunque no tengamos el reloj, se puede diseñar la arquitectura de un servicio capaz de generar de manera aleatoria dentro de nuestro microservicio los valores que generaría este reloj si una persona lo llevara y en un futuro poder incorporar este reloj de forma real a nuestro sistema. Realmente lo que hicimos

fue replicar el servicio que realizar Tuya en su plataforma para virtualizar dispositivos, pero nosotros mismos en nuestro microservicio.

Para ello, se ha construido un nuevo controlador *DeviceController*. El objetivo de este controlador es, además de utilizar los datos reales relativos a los sensores compatibles con *Tuya*, simular los datos que obtendríamos a partir del uso del reloj de Samsung. En la implementación del servicio que utiliza el controlador, además de recopilar los datos reales de los sensores de Tuya es capaz de generar valores que devolvería el reloj de Samsung.

Como pueden observar en la Figura 3.8, además de las fechas para definir el intervalo como en el caso anterior, se definen una serie de nuevas variables que explicaremos a continuación.

```

Alvaro de la Flor Bonilla
@RestController
@RequestMapping("/devices")
public class DeviceController {

    @Autowired
    public DeviceService deviceService;

    Alvaro de la Flor Bonilla
    @GetMapping("/signal")
    public Signal ping(
        @DateTimeFormat(iso = DateTimeFormat.ISO.DATE_TIME, fallbackPatterns = { "dd/MM/yyyy HH:mm" }) @RequestParam(value = "startTime") LocalDateTime startDateTime,
        @DateTimeFormat(iso = DateTimeFormat.ISO.DATE_TIME, fallbackPatterns = { "dd/MM/yyyy HH:mm" }) @RequestParam(value = "endTime") LocalDateTime endDateTime,
        @RequestParam(required = false, value = "debug") String debug,
        @RequestParam(required = false, value = "fakeSamsungValue") String fakeSamsungValue,
        @RequestParam("username") String username
    ) {
        return this.deviceService.getAndSaveAllDeviceSignals(startDateTime, endDateTime, FakeSignal.fromDebugParam(debug), FakeSamsungValue.fromValue(fakeSamsungValue), username);
    }
}

```

Figura 3.8 Controlador DeviceController

El primer parámetro que puede llamar la atención es el valor **debug**. Si se asigna este valor como **“samsung”** se activa la **simulación de los datos** de un dispositivo Samsung (ver la Figura 3.9). Es decir, por cada llamada a nuestro controlador, si no se encuentra activado este parámetro se devolverán los registros de los sensores Tuya, como hemos visto anteriormente (método **getPirSignal** de TuyaDeviceController).

En el caso concreto de activar el modo *debug* simple (posteriormente veremos casos más complejos), se simularán datos relativos a **constantes básicas del usuario**. Estas constantes básicas son el número de pasos, la glucosa en sangre, la saturación sanguínea, la presión arterial y el pulso. Cada vez que se hace uso de este *endpoint* con el valor *debug* asignado a **“samsung”** se establecen unos **valores aleatorios (nunca se repite el resultado anterior)**, manteniendo unos resultados que **están dentro de los márgenes recomendados por expertos**.

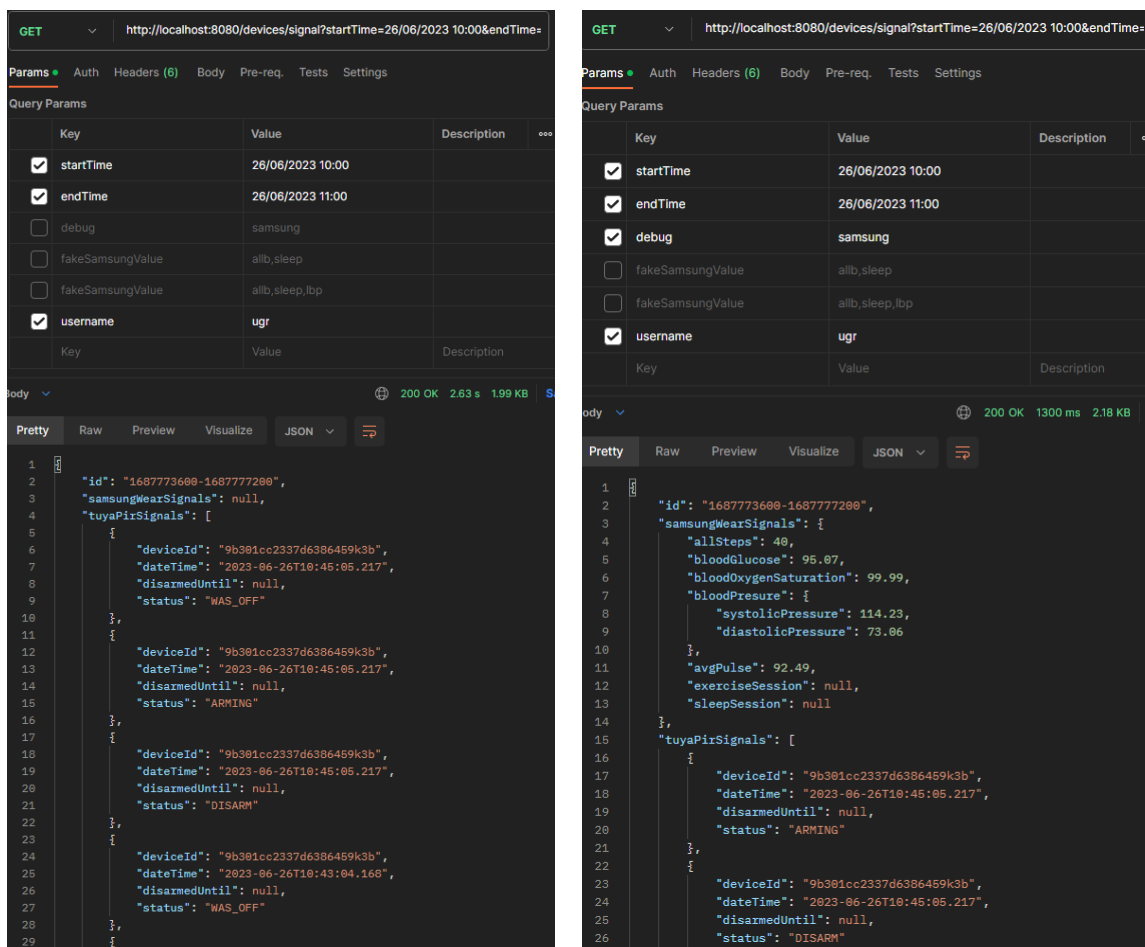


Figura 3.9 Uso de DeviceController

Para poder simular justo el caso contrario, valores que no es establecen respecto a los márgenes recomendados, y poder así recrear sintomatología relacionada con el deterioro cognitivo, se ha añadido la parametrización en el controlador del valor **“fakeSamsungValue”** (ver la Figura 3.10). Dependiendo del valor que se utilice, variará la configuración de esta simulación.

1. **“allb”**: hace referencia a **“all bad”**. Si se utiliza esta configuración todos los valores quedarán fuera de los valores recomendados, de esta forma podremos simular valores que corresponden con síntomas relacionados con deterioro cognitivo.
2. **“sleep”**: si se utiliza este parámetro se añadirán los registros de sueño que es capaz de recoger el reloj. Estos registros incluyen las diferentes etapas del sueño del usuario y sus intervalos (*AWAKE*, *LIGHT*, *DEEP* y *REM*), la saturación de oxígeno media durante el sueño, el pulso medio, la duración total del sueño y si se ha registrado el día completo o es posible que aún se registren más datos. Hay que destacar que todos los valores de sueño se mantendrán dentro de los rangos recomendados, pero si se utiliza en combinación de **“sleep”** el valor **“allb”**, todos los valores de sueño también se saldrán de los rangos recomendados.

3. **“lbp”**: se ha añadido este caso para simular una situación muy concreta. Por defecto, si utilizamos la parametrización para que los resultados salgan de los márgenes recomendados para la presión arterial, se establecen valores aleatorios que están por encima de los máximos. Utilizar este parámetro permite establecer valores que están por debajo del mínimo requerido solo para la presión arterial.

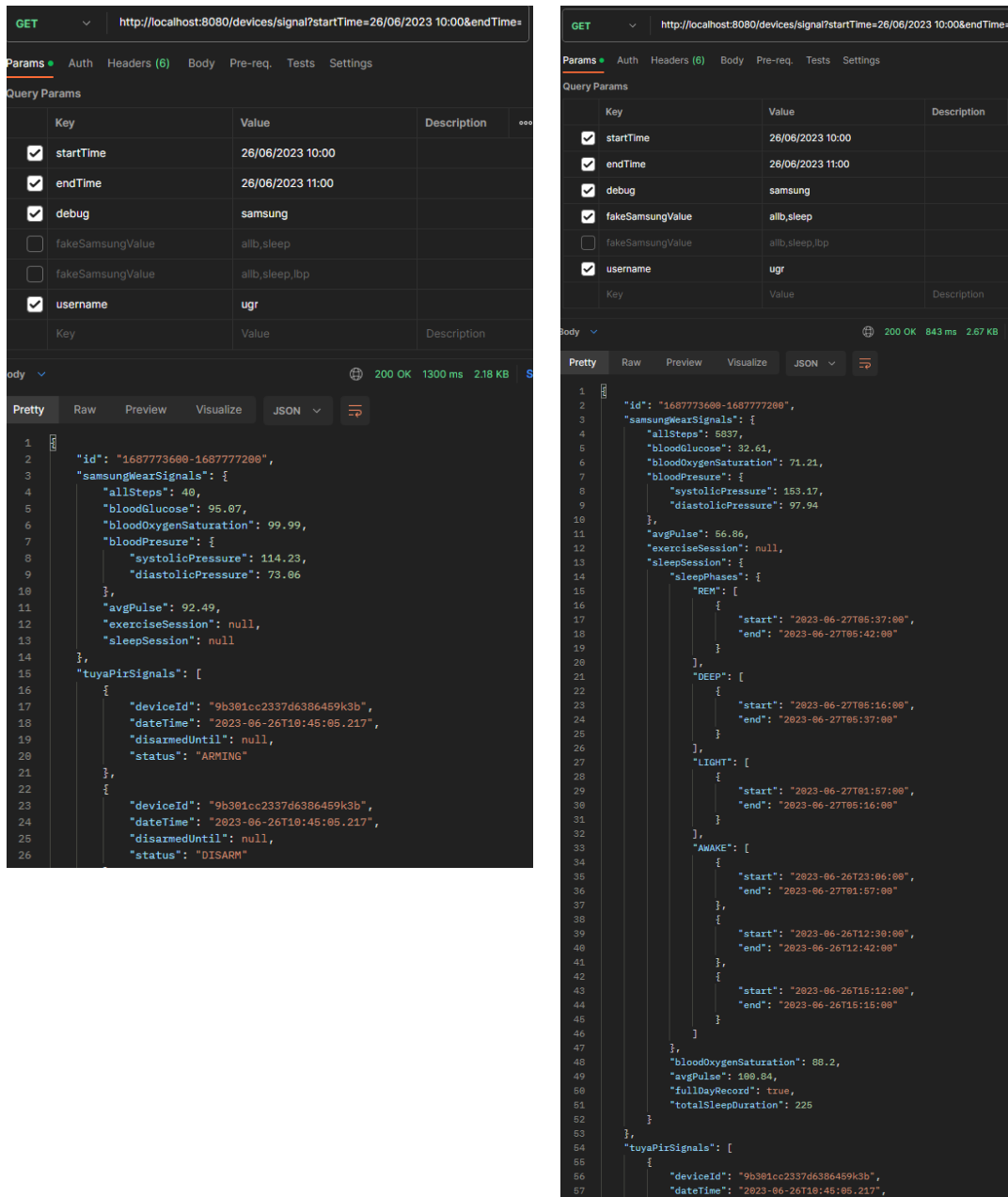


Figura 3.10 Parametrización del controlador

4. **“username”**: se utiliza este valor para hacer referencia al usuario de los datos. El sistema está preparado para interactuar con más de una cuenta a la vez. Este parámetro será utilizado para dos cosas. En primer lugar, utilizaremos este nombre

de usuario para asociar el *accessToken* y el *accessId* correspondiente del usuario, y además utilizaremos como clave el nombre del usuario para almacenar todos los valores que necesitamos que se almacenen de forma persistente en la base de datos.

Cada vez que ejecutamos este endpoint, como se puede apreciar en la Figura 3.10, se crea un objeto (evento) de tipo **Signal**. Este objeto contiene tanto la información de todos los movimientos que se hayan podido detectar por los sensores de movimiento asociados por el usuario (*tuyaPirSignals*), como de los registros de datos del reloj inteligente, en caso de que se encuentre activado el modo *debug* (*samsungWearSignals*).

Una vez se haya recopilado toda la información este nuevo objeto, se almacena de forma persistente en una base de datos Redis, a la espera de que este evento (objeto **Signal**) sea analizado por el siguiente microservicio **Analyzer** que explicaremos posteriormente.

La principal razón por la que hemos usado esta metodología de “eventos” con microservicios independientes en lugar de realizar todo en un único monolito ha sido simplemente una idea estratégica. Este primer microservicio tendrá como requerimiento únicamente escuchar todos los sensores por cada uno de los usuarios y almacenarlos para que posteriormente sean analizados. De esta manera, en un futuro será mucho más sencillo escalar la aplicación, asignando más o menos recursos a cada sistema en función de las necesidades. Por otro lado, la caída de un microservicio no conllevará que el otro no pueda seguir trabajando, ya que puede consumir el conjunto de señales en cola que le queden por analizar.

Además, desde un punto de vista comercial, para rentabilizar este sistema se puede proponer un servicio *freemium*, un modelo de negocio en el que parte del producto que se oferta es gratuito, pero existe una modalidad de pago para disfrutar de ciertas funciones. En nuestra aplicación podríamos establecer como función extra (la que no se ofrece gratis en la modalidad *freemium*) la modificación de frecuencia de ejecución de los diferentes microservicios.

Es evidente que cuanto más alta sea esta frecuencia más coste computacional (si desplegamos nuestro sistema en la nube) tendrá. Se puede plantear establecer un valor fijo de, por ejemplo, dos análisis a la hora y en el caso de que se quiera modificar este valor se tendrá que acceder a una modalidad de pago. Esta idea se establece con vistas a sufragar los posibles gastos que supongan desplegar el sistema en línea para que el usuario ni si quiera tenga que comprar un dispositivo donde desplegar el sistema, como por ejemplo, una Raspberry Pi o un ordenador de bajo coste. El uso de un sistema en red simplificaría aún más el uso de nuestra propuesta.

Como bien hemos comentado, este servicio se comunicará con el resto mediante eventos almacenados en una base de datos Redis. Cualquier tipo de base de datos como Oracle, Mongo o MySQL han sido descartadas por motivos de velocidad. Queremos proponer un sistema cuyo principal pilar sea la rapidez de comunicación entre los diferentes microservicios, usar uno de los servicios anteriores castigaría el rendimiento de nuestro sistema.

Figura 3.11 MongoDB vs Redis⁴⁰

Como puede ver en la Figura 3.11, en base de datos como por ejemplo MongoDB, a medida que se aumenta el número de usuarios tanto el rendimiento como la latencia se ven claramente afectados. El comportamiento en otras bases de datos como Oracle o MySQL es prácticamente similar al mostrado por MongoDB, a medida que crecen los usuarios, también se perjudica al rendimiento en mucha mayor medida que en sistemas como Redis.

En concreto, de entre los sistemas de almacenamiento disponibles, se ha elegido Redis porque cumple exactamente con el objetivo clave valor que queremos conseguir. Es decir, para un usuario y una acción (evento) se requiere persistir un conjunto de datos.

En este servicio en concreto se ha diseñado de tal forma que la clave estará compuesta por el tipo de evento, en este caso **SIGNAL** y el nombre del usuario. Utilizando como ejemplo el nombre de usuario “*ugr*” la clave que utilizará el sistema para persistir los datos en redis será “**SIGNAL|ugr**”.

Vista esta configuración, se podría llegar a pensar que, si el siguiente servicio (analizador) no llega a consumir el evento a tiempo en la siguiente recogida de datos de los sensores, se sobrescribirá la información, ya que se usará la misma *key* “**SIGNAL|(username)**”. Sin embargo, esto es incorrecto, ya que, además de este *hash key* se usa adicionalmente un *ID*. En nuestro caso hemos elegidos utilizar el período de tiempo en formato *timestamp* para definir inequívocamente la diferenciación entre dos eventos.

En la Figura 3.12 puede observar un ejemplo de lo comentado anteriormente. A pesar de que se ha recopilado la información de los sensores dos veces, la primera de ellas sin ningún dato y la segunda con datos del reloj de Samsung, pueden compartir *hash key* en Redis, ya que quedan diferenciados ambos registros por la *ID* utilizada. Esto permite que cuando el servicio analizador de datos recopile toda la información de un usuario con tan solo el nombre de usuario pueda obtener todos los datos con independencia al intervalo de fecha utilizado.

⁴⁰ MongoDB vs Redis: <https://scalegrid.io/blog/comparing-in-memory-databases-redis-vs-mongodb-percona-memory-engine/>

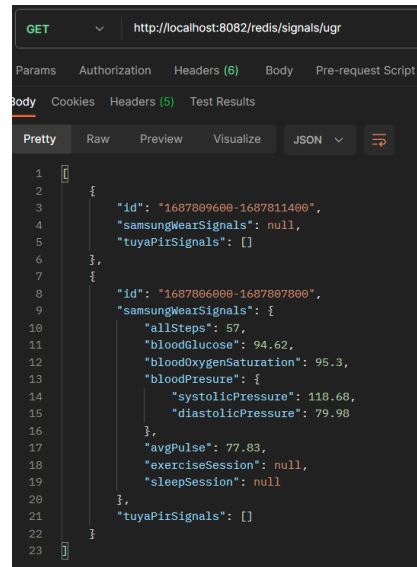


Figura 3.12 Ejemplo de key compartida en redis

Además, como bien se adelanta en la Figura 3.12, se ha creado un microservicio independiente llamado “**redis-gateway**”. Este microservicio está encargado tanto de abstraer toda la lógica para guardar, borrar y seleccionar los “eventos” que hemos guardado en Redis para todo el resto de los microservicios. En concreto, el microservicio “**sensors**” requiere como mínimo que se sea capaz de guardar los eventos de tipo **Signal** que hemos explicado con anterioridad. De manera adicional se han añadido la lógica para ser capaz de leer cada evento **Signal** a partir de un nombre de usuario y borrar cada uno de los eventos a partir del nombre de usuario y su **ID** en específico (ver la Figura 3.13)

```

Alvaro de la Flor Bonilla
@PostMapping("/signals/{username}")
public void saveSignal(
    @RequestBody Signal signal,
    @PathVariable("username") String username
) {
    this.redisService.saveSignal(signal, username);
}

Alvaro de la Flor Bonilla
@GetMapping("/signals/{username}")
public List<Signal> getAllSignalsByUsername(
    @PathVariable(value = "username") String username
) {
    return this.redisService.getAllSignalsByUser(username);
}

Alvaro de la Flor Bonilla
@DeleteMapping("/signals/{username}/{id}")
public void deleteSignalByUsernameAndId(
    @PathVariable(value = "id") String id,
    @PathVariable(value = "username") String username
) {
    this.redisService.deleteSignal(id, username);
}

```

Figura 3.13 Controlador microservicio redis-gateway RedisController

Finalizado esta etapa, somos capaces de leer todos los movimientos que detecten los sensores Tuya registrados en la cuenta de un usuario, así como simular de forma aleatoria

los datos que devolvería un Samsung Galaxy Watch 5, tanto para valores que están dentro de los márgenes recomendados como fuera de ellos. Estos datos somos capaces de persistirlos en una base de datos Redis como un evento **“signal”**, que se encuentra a la espera de ser analizado por el siguiente microservicio, **“analyzer”**. En Figura 3.14 se puede ver el diagrama de flujo del microservicio que hemos descrito en esta aplicación.

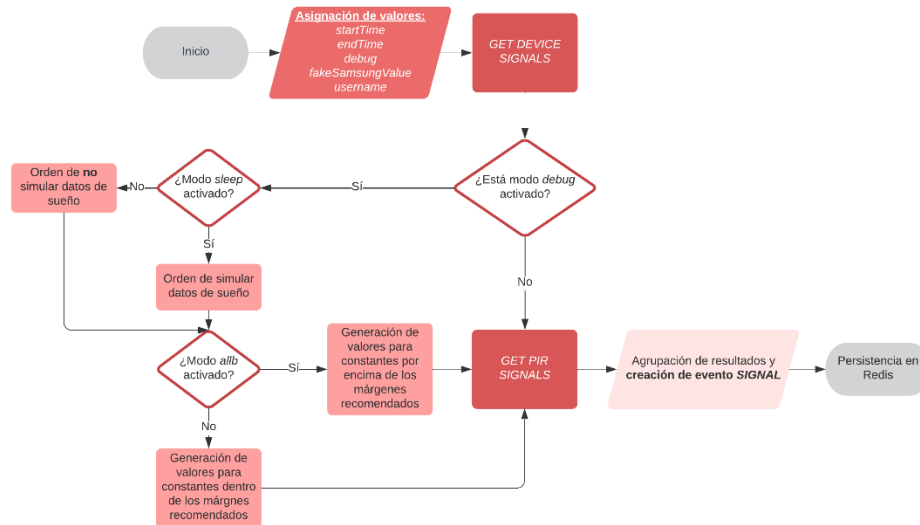


Figura 3.14 Diagrama de flujo del primer microservicio

3.3.2 Iteración 2: Microservicio “analyzer”

El objetivo del microservicio que se quiere implementar en esta etapa es una lógica capaz de recuperar todos los eventos **“Signal”** que se han creado para un usuario y llegar a una conclusión con ellos. Esta conclusión será un evento de tipo **“Workbook”** que será utilizado por un tercer microservicio (**“execution”**), que estará encargado de llevar a cabo algún tipo de acción.

Antes de avanzar con el proyecto, se decidió que sería una buena idea **“dockerizar”** todo el desarrollo en este momento, ya que si se hiciese más tarde con la aplicación mucho más avanza sería más complicado. **“Dockerizar”** una aplicación consiste en crear un contenedor **Docker** que encapsule todos los componentes necesarios para ejecutar una aplicación de forma independiente y reproducible. Docker es una plataforma de virtualización a nivel de sistema operativo que permite empaquetar una aplicación junto con sus dependencias en un contenedor ligero y portátil. De esta forma, cualquier equipo compatible con Docker podrá ejecutar nuestra propuesta sin necesitar instalar nada, tan solo ejecutando el contendor.

Además, una vez finalizado el desarrollo del microservicio **“sensors”**, se procedió a extraer todas las clases que conforman el dominio de la aplicación en un proyecto Maven separado, de tal forma que todos los microservicios pudieran compartir los modelos (definición de eventos) sin tener que repetir código para cada uno de los microservicios. Idealmente se debería haber creado un repositorio **Nexus** (herramienta de gestión de repositorios), incorporar este **“proyecto de modelos”** a él e incluir el dominio de la aplicación como

dependencia a cada uno de los microservicios, pero para esta prueba de concepto complicaría un poco más el desarrollo. Alternativamente, como puede observar en la Figura 3.15, se ha decidido crear un proyecto *Maven* general en el que cada módulo representa un microservicio, meramente para agilizar el desarrollo, pero cada microservicio puede desplegarse y funcionar de forma separada.

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.alvarodelaflor</groupId>
  <artifactId>tfm-code</artifactId>
  <version>1.0.0</version>
  <packaging>pom</packaging>

  <modules>
    <module>domain</module>
    <module>sensors</module>
    <module>analyzer</module>
    <module>execution</module>
    <module>redis-gateway</module>
  </modules>

  <properties>
```

Figura 3.15 Estructura del proyecto

Gracias al microservicio **“redis-gateway”** que se desarrolló en la primera iteración, recuperar todos los eventos pendientes de procesar para un usuario es tan simple como llamar al *endpoint* **“/redis/signals/{username}”**.

El siguiente paso es construir el modelo de las alertas. Pretende señalar un signo relacionado con deterioro cognitivo que puede detectarse a partir de comportamientos del usuario o por medio de sus constantes vitales. Cada una de estas alertas tendrá un filtro relacionado el microservicio **“analyzer”** que se encargará de detectar si de entre todos los registros del evento **Signal** para ese usuario existe alguna situación relacionada con la definición de la alerta. Para ello, se han revisado varios artículos en la búsqueda de indicadores de demencia asociado a variables fisiológicas y para aquellos que utilizaban variables a las que tenemos acceso se ha creado el correspondiente filtro y alerta.

En la Tabla 7 se describe las diferentes alertas que se han implementado en nuestra aplicación, indicándose para cada una de ellas el artículo o fuente de referencia que nos ha servido de inspiración para crearlo.

Alerta	Evento de activación	Fuente
DisableSensorsMovementAlert	Número de veces que se desconectan los sensores	Gálvez Díaz (2016)
RepeatedMovementAlert	Número de movimientos repetidos en un breve período de tiempo	(Alzheimer's Association, s. f.)
AwakeningsAlert	Aumento de los despertares	Echavarri y Erro (2007)
DayTimeAlert	Aumento de las siestas diurnas	Echavarri y Erro (2007)

RemAlert	Poca cantidad de sueño en fase REM	Pase et al. (2017)
WakeUpEarlyAlert	Despertar excesivamente temprano	Echavarrí y Erro (2007)
BradycardiaAlert	Tener una media de pulsaciones durante el día por debajo de 60 ppm	Bayón et al. (2014)
HighBloodPressureAlert	Tener la presión sanguínea con valores demasiado altos	Vicario et al. (2010)
LowBloodPressureAlert	Tener la presión sanguínea con valores demasiado bajos	Ceraso, D. (2001)

Tabla 7 – Relación de alertas, eventos de activación y fuente de los datos

- ***DisableSensorsMovementAlert***

Tal y como describe Gálvez Díaz (2016), esta alerta trata de detectar los casos en los que las personas que han percibido su propio deterioro cognitivo “*ponen en marcha mecanismos de defensa para ocultar sus fallos ante los demás*”. Es un síntoma habitual que estas personas traten de evitar que sus familiares más cercanos se percaten de ello y le resten importancia al problema.

Respecto a la implementación del filtro de esta alerta (***DisableSensorsMovementFilter***) se tratará de contabilizar el número de veces que el usuario desconecta los sensores y si ese número sobrepasa una cifra preestablecida se lanzará la alerta (ver la Figura 3.16).

```

Alvaro de la Flor Bonilla
@Override
public Optional<CommonAlert> isRuleValid(Signal signal, ValueService valueService) {
    Map<String, List<LocalDateTime>> disableSensorsOccurrences = getDisableSensorsOccurrences(signal);
    Integer numberOfTimes = getNumberOfOccurrences(disableSensorsOccurrences);
    return numberOfTimes < valueService.getDisableSensorsOccurrences() ?
        Optional.empty()
        :
        Optional.of(getCommonAlert(disableSensorsOccurrences, numberOfTimes));
}

```

Figura 3.16 Implementación filtro ***DisableSensorsMovementAlertFilter***

En el caso de que se lance la alerta, tal y como puede ver en la Figura 3.17, se devolverá un objeto del tipo de la alerta que ha activado (en este caso ***DisableSensorsMovementAlert***) que extenderá siempre a la clase general ***CommonAlert***. Esta alerta contará con el número de veces que el usuario ha desactiva el sensor de movimiento y a qué hora lo ha hecho.

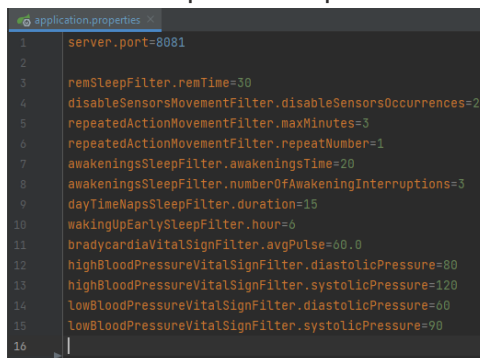
```

1 usage Alvaro de la Flor Bonilla
public CommonAlert getCommonAlert(Map<String, List<LocalDateTime>> disableSensorsOccurrences, Integer numberOfTimes) {
    return DisableSensorsMovementAlert.builder() DisableSensorsMovementAlertBuilder<capture of ?, capture of ?>
        .dateTimeList(disableSensorsOccurrences) capture of ?
        .numberOfTimes(numberOfTimes)
        .build();
}

```

Figura 3.17 Creación de la alerta ***DisableSensorsMovementAlert***

Además, para todas las implementaciones de filtros que hemos realizado en este microservicio, todo valor prefijado que es evaluado para desencadenar una alerta es fácilmente configurable y editable en función de las características del usuario. Por el momento, todos estos valores quedan configurados en el “*application.properties*” (ver la Figura 3.18) de nuestro sistema, por lo que con tan solo reiniciar el microservicio quedarán establecidos los nuevos valores. En un futuro esta configuración se pretende que sea tan fácil de modificar como establecer los valores en una ventana de configuración de los sensores de la aplicación para dispositivos móviles, de manera independiente para cada usuario.



```

1 server.port=8081
2
3 remSleepFilter.remTime=30
4 disableSensorsMovementFilter.disableSensorsOccurrences=2
5 repeatedActionMovementFilter.maxMinutes=3
6 repeatedActionMovementFilter.repeatNumber=1
7 awakeningsSleepFilter.awakeningsTime=20
8 awakeningsSleepFilter.numberOfAwakeningInterruptions=3
9 daytimeNapsSleepFilter.duration=15
10 wakingUpEarlySleepFilter.hour=6
11 bradycardiaVitalSignFilter.avgPulse=60.0
12 highBloodPressureVitalSignFilter.diastolicPressure=80
13 highBloodPressureVitalSignFilter.systolicPressure=120
14 lowBloodPressureVitalSignFilter.diastolicPressure=60
15 lowBloodPressureVitalSignFilter.systolicPressure=90
16

```

Figura 3.18 Configuración de valores iniciales para el análisis de eventos

Cada una de las alertas que vamos a ver a continuación incorpora en su dominio el subtipo de alerta que es. Este subtipo de alerta queda modelado por el enumerado **AlertType** y puede tener dos valores:

- **AlertType.INFORM:** Indica que en el tercer microservicio se debe tener en cuenta esta información para llevar a cabo el análisis del informe de deterioro cognitivo que se le realizará al usuario a partir de todas las alertas creadas. Todas las alertas que se definirán a continuación tienen este subtipo, salvo una (**LowBloodPressureAlert**).
- **AlertType.ACTION:** Indica que se ha detectado una situación peligrosa y el tercer microservicio debería actuar cuanto antes, ya sea realizando una acción o notificando a los tutores o cuidadores.

A continuación, veremos en detalle cada una de las alertas que se han implementado:

- **RepeatedMovementAlert**

Es común que personas con patologías relacionadas como el Alzheimer u otro tipo de deterioro cognitivo se les olvide a veces dónde están o cómo llegaron a ese lugar (Alzheimer’s Association, s. f.). Incluso pueden llegar a confundir su propio hogar y no identificar donde se encuentran dentro de éste.

En nuestro caso particular, la implementación de esta alerta pretende contabilizar el número de veces que un usuario repite una acción relacionada con entrar o salir de la misma estancia (baño, habitación, dormitorio...). Se determina que si se repite esta acción un número de veces durante un período de tiempo máximo establecido se activará la alarma. Una vez más, como en todas las implementaciones de filtros,

se hace uno de la parametrización de los valores evaluables que activan la alerta mediante el servicio **valueService**. Nos gustaría recordar que este servicio se utiliza en todos los filtros, para no usar ningún valor fijo en las condiciones de activación.

La alerta que se crea detallará el número de veces que ha repetido una acción por cada uno de los sensores repartidos en el hogar, además del intervalo de tiempo (fecha y hora) en el que se ha producido este movimiento repetido.

- ***AwakeningsAlert***

Tal y como se detalla en el artículo de Echavarri y Erro (2007), se ha comprobado que en la enfermedad de Alzheimer (EA), el sueño se caracteriza por un aumento de los despertares, tanto en duración como en frecuencia.

En el filtro que se ha implementado (***AwakeningsSleepFilter***) a la hora de lanzar esta alerta se contabilizan el número de sucesos en los que el usuario ha pasado de un estado de sueño a un estado **AWAKE**. En la configuración de la alerta se ha añadido información como el número de veces que ocurre este suceso (incluyendo fecha y hora del suceso) así como el tiempo total que el usuario ha permanecido despierto hasta dormirse otra vez.

- ***DayTimeAlert***

Un número elevado de siestas durante el día puede representar un síntoma de deterioro cognitivo, como se indica en el artículo de Echavarri y Erro (2007). En este artículo se afirma que “*En la enfermedad de Alzheimer (EA), el sueño se caracteriza por un aumento de las siestas diurnas*”.

En este caso en la implementación del filtro hacer una selección de las veces que ha dormido el usuario durante el día. Si la duración total de estos sueños es superior al máximo preestablecido se lanzará la alerta, que contará con la información completa de estas siestas diurnas, es decir, para cada etapa de sueño que se tuvo (**AWAKE**, **LIGHT**, **DEEP** o **REM**) se almacena la fecha y hora de cada uno de ellos.

- ***RemAlert***

En esta alerta se pretende detectar los casos en los que a pesar de que el usuario es capaz de dormir, el cómputo total de minutos de sueño REM, es muy bajo. Como se detalla en el estudio de Pase et al. (2017), un menor sueño REM se traduciría en un riesgo más alto de demencia.

En la implementación de nuestro filtro ***RemSleepFilter***, se contabiliza todos los minutos de sueño REM que ha tenido el usuario durante los ciclos de sueño de ese día. En el caso de que sea inferior al establecido como valor mínimo por defecto, se lanzará la alerta que contará con la duración total de minutos de sueño REM.

- **WakeUpEarlyAlert**

Este filtro es muy sencillo y pretende detectar los casos en los que las personas mayores se despiertan excesivamente temprano. En el estudio de Sarrais & de Castro Manglano (2007) se hace referencia al trastorno del ritmo circadiano en el que el sujeto se despierta espontáneamente a primeras horas de la madrugada y puede representar indicios de deficiencia cognitiva.

En el filtro de esta alerta, su implementación (**WakeUpEarlySleepFilter**) tiene en cuenta que se está en el último ciclo de sueño del usuario y se devuelve esta fecha y hora como información en la alerta.

- **BradycardiaAlert**

En el estudio realizado en Bayón et al. (2014) se hace referencia a la bradicardia (condición en la que la frecuencia cardíaca es anormalmente lenta) como un síntoma de la deficiencia cognitiva. En concreto se indica que *“la bradicardia (< 60 ppm) es estadísticamente más frecuente en los pacientes con DFT (demencia frontotemporal)”*. Es decir, se ha llegado a la conclusión de que *“la bradicardia es más frecuente en enfermos con deterioro cognitivo ligero o demencia de otra etiología”*.

La implementación de este filtro (**BradycardiaVitalSignFilter**), es una de las más sencillas de todas, ya que únicamente se estudia el valor de pulsación media del usuario durante el día. Dato que es proporcionado por el smartwatch directamente sin necesidad de hacer ningún cálculo con pulsaciones durante el día.

- **HighBloodPressureAlert**

Según estudios como el de Vicario et al. (2010) los pacientes con hipertensión arterial muestran más deterioro cognitivo que los controles normales en diferentes estudios. Con ello se concluye que es una evidencia que tener unos valores medios altos de presión arterial pueden conllevar el desarrollo de alguna deficiencia cognitiva.

La implementación de esta alerta se lleva a cabo en el filtro **HighBloodPressureVitalSignFilter** y en este se pretende detectar si bien la presión sistólica y o bien la presión diastólica supera los márgenes establecidos como recomendados. La alerta que se creará contará con la información de la presión arterial media durante el día.

En la Figura 3.19 se muestra cómo una alerta puede contar con los dos tipos que explicamos anteriormente. Por ejemplo, el caso de presión sanguínea es un síntoma relacionado con un posible caso de deficiencia cognitiva, pero a su vez, pero ser un problema real que se debe tratar con la mayor brevedad que sea posible.

```
public class HighBloodPressureAlert extends CommonAlert implements Serializable {

    2 usages
    SamsungWearSignal.BloodPresure bloodPressure;
    @Builder.Default
    Double weight = 2.9;
    @Builder.Default
    String link = "https://www.sciencedirect.com/science/article/abs/pii/S1853002810700707";
    @Builder.Default
    String summary = "Los pacientes con hipertensión arterial muestran más deterioro cognitivo que los controles normales en diferentes estudios";
    @Builder.Default
    List<AlertType> alertType = Arrays.asList(AlertType.INFORM, AlertType.ACTION);
    @Builder.Default
    String descriptionName = "Hipertensión";
}
```

Figura 3.19 Modelado de una alerta y su tipología

- **LowBloodPressureAlert**

A diferencia del caso anterior, no se pretende detectar un signo de deficiencia cognitiva, sino alertar a posibles cuidadores o tutores del usuario de la aplicación de que el usuario puede padecer hipotensión, es decir, la presión sanguínea es más baja de lo normal, lo que podría ser una bajada de tensión repentina grave.

Esta alerta se ha implementado en el filtro **LowBloodPressureVitalSignFilter** y funciona exactamente igual que la anterior, pero detectando valores que queden por debajo de los valores establecidos como recomendados.

En este caso, la alerta es de tipo **AlertType.ACTION**, ya que no se ha encontrado ningún estudio que relacione la hipotensión con problemas relacionados a la deficiencia cognitiva, pero sí el peligro que puede conllevar la existencia de la hipotensión arterial severa (Ceraso, D., 2001)

En la Figura 3.20 puede ver un extracto de código de la implementación de **AnalyzeService** que resume lo que se realiza en el microservicio **analyzer**. Dado un listado de señales asociadas a un usuario (por medio de su *username*), que han sido creadas por el microservicio **sensors**, este microservicio captura todas ellas y las analiza, creando un nuevo evento llamado **Workbook** asociado al usuario, el cual será consumido por el siguiente microservicio, **execution**, haciendo uso de redis-gateway y el *endpoint* **"/redis/workbooks/{username}"**. Este evento contendrá todas las alertas que serán analizadas una a una.

```

1 usage  Alvaro de la Flor Bonilla *
public List<Workbook> analyzeSignals(List<Signal> signals, String username) {
    List<Workbook> workbookList = new ArrayList<>();
    signals.stream().forEach(signal -> {
        Optional<Workbook> workbook = checkSignal(signal);
        if (workbook.isPresent()) {
            workbookList.add(workbook.get());
            redisService.saveWorkbook(workbook.get(), username);
            redisService.deleteSignal(signal.getId(), username);
        } else {
            logger.info("There are no signs that require attention or are candidates for consideration in the report");
        }
    });

    return workbookList;
}

```

Figura 3.20 Implementación del analizador de signals en AnalyzeService

Cada vez que una señal es analizada, su registro es eliminado de Redis, evitando así contemplar dos veces un mismo registro. Además, tal y como pasaba antes, este nuevo evento también es almacenado en *Redis* utilizando la misma estrategia, el *hash key* será “**WORKBOOK**/(username)” y cada *workbook* contendrá una *ID* distinta, por lo que si el tercer microservicio no ha tenido tiempo de procesar el evento antes de que se cree un nuevo *workbook* no pasará nada, ya que no se sobrescribirán los datos y la arquitectura que se ha diseñado permite procesar más de un evento a la vez.

Tras recuperar la información del evento **signal**, el microservicio decidirá qué información del evento **signal** se debe tener en cuenta (y que se debe descartar) y finalmente se construirá el nuevo evento que consumirá el siguiente microservicio. Se finaliza así la funcionalidad completa de este microservicio. En la Figura 3.21 puede observar el diagrama de flujo de este segundo microservicio.

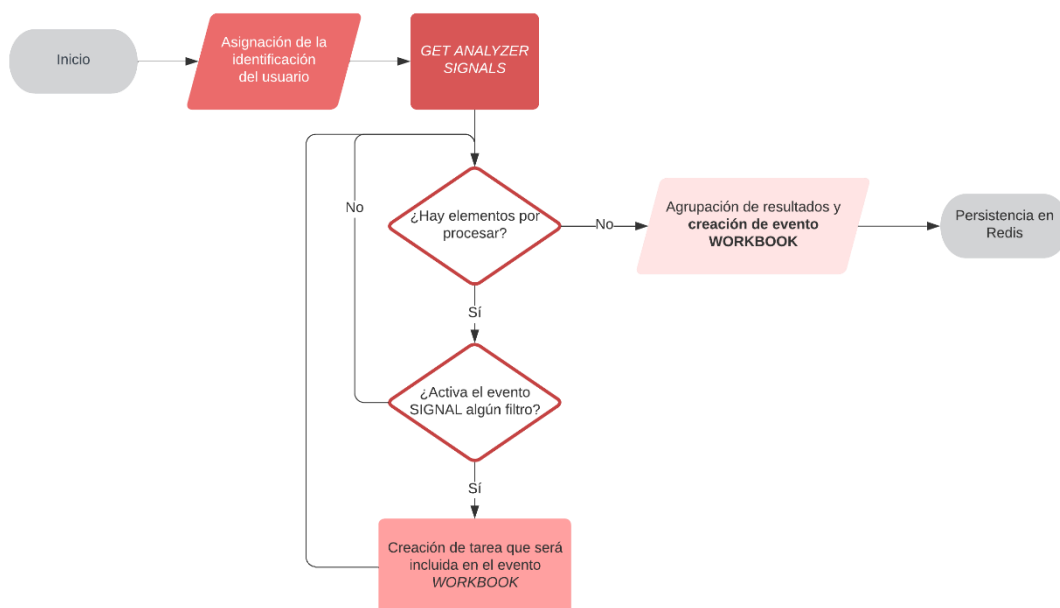


Figura 3.21 Diagrama de flujo del segundo microservicio

3.3.3 Iteración 3: Microservicio “execution”

El objetivo de este tercer microservicio es recopilar todos los eventos que cree el microservicio anterior y llevarlas a cabo. Como prioridad, en esta prueba de concepto nos hemos centrado en generar un informe lo suficientemente válido como para que importe información de calidad a un profesional médico. Por otro lado, también pretende realizar un conjunto de acciones rápidas para avisar con la mayor brevedad posible la existencia de un posible problema que requiere atención.

En cuanto a la generación del informe, por cada evento del tipo **workbook** se analizan cuáles de sus alertas son del tipo **AlertType.INFORM**. Para cada una de ellas se creará un resumen de porqué es de importancia tener en cuenta el comportamiento que se ha detectado, un enlace a la documentación del artículo científico que corrobora esta información y los datos de los sensores que han desatado la alerta, es decir, la copia del comportamiento que confirma este problema detectado, por ejemplo para el caso de siestas diurnas el conjunto de intervalos (horas) en la que se han producido y los diferentes estados (*AWAKE*, *REM*...) para cada uno de ellos. Los datos particulares para cada una de las alertas se explicaron en el apartado anterior.

En la Figura 3.22 se muestra un ejemplo del reporte (informe) que se obtiene al ejecutar el *endpoint* principal de este microservicio. Como dijimos, para cada usuario se analizan los eventos **workbook** pendientes, y para las alertas que se deben añadir al reporte (*TypoAlert.INFORM*), se realizará una pequeña introducción del problema y porqué es importante tenerlas en cuenta.

Además, se podrá ver que para cada alerta se añade una cifra calificada como *weight* (peso). Este parámetro, que se define a la hora de crear la alerta, muestra la importancia de este problema respecto al conjunto del reporte completo. La intención de esta variable es darle un peso a cada una de las alertas creadas, de tal forma que una vez concluido el informe se sumaran todos los pesos y el valor del cómputo completo determinará la probabilidad o gravedad del deterioro cognitivo que se señala que puede existir.

El peso de cada una de las alertas es aún una tarea pendiente. Es evidente que algunas evidencias pueden tener mucho más peso que otras, por ejemplo, en tener una poca cantidad de sueño REM respecto a despertarse muy temprano. En este caso la primera puede ser más significativa que la segunda. La correcta puntualización de estos pesos depende de la interacción de esta aplicación con profesionales, ya que es muy difícil evaluar cuanto de importante es más una alerta sin probar el sistema, pero podemos asegurar que tener en cuenta estos pesos puede ser muy beneficioso para la aplicación.

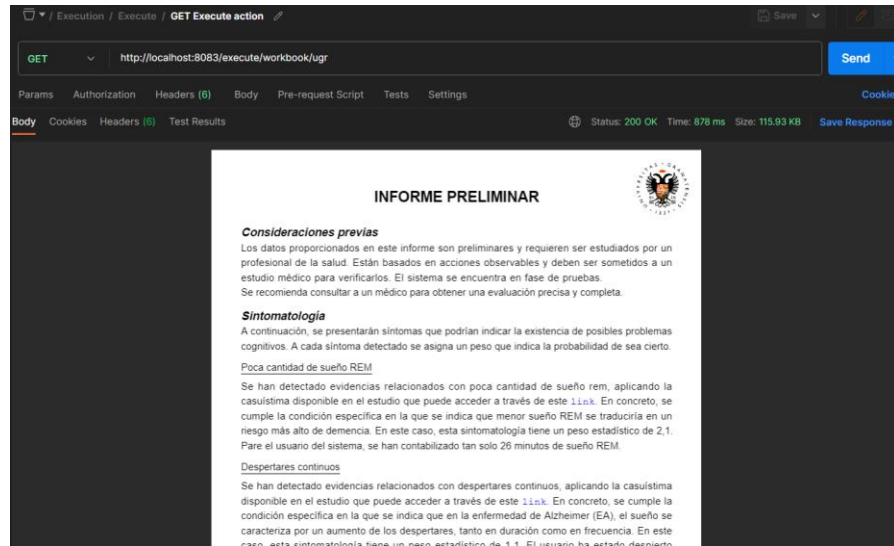


Figura 3.22 Ejecución del microservicio Execution

Así mismo, también queda pendiente establecer el valor máximo del sumatorio total a partir del que se debería llamar la atención o destacar algún problema. Como se puede ver en la Figura 3.23, al finalizar el reporte se hace el sumatorio de todas las alertas que incluyen el evento **workbook** y se indica el valor máximo que se debería tener

Además, los pesos y el sumatorio total deberían ir asociados a personas concretas, y deberán variar según sus características y circunstancias, así como también según su evolución.

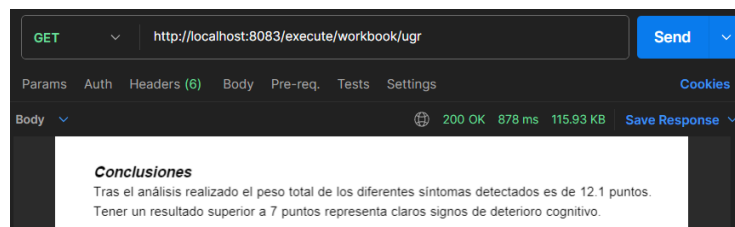


Figura 3.23 Conclusiones del reporte

En la Figura 3.24 se muestra el diagrama de este tercer microservicio. Sencillamente, es un bucle en el que se van añadiendo alertas al reporte en formato *PDF*, en el caso de que la alerta dentro del evento *workbook* lo requiera (si la alerta es del tipo *AlertType.Inform*).

Si esta alerta cuenta con el tipo *AlertType.Action* tal y como se muestra en el diagrama de flujo presentado en la Figura 3.24, se genera un mensaje en la consola *log* donde se esté ejecutando la aplicación.

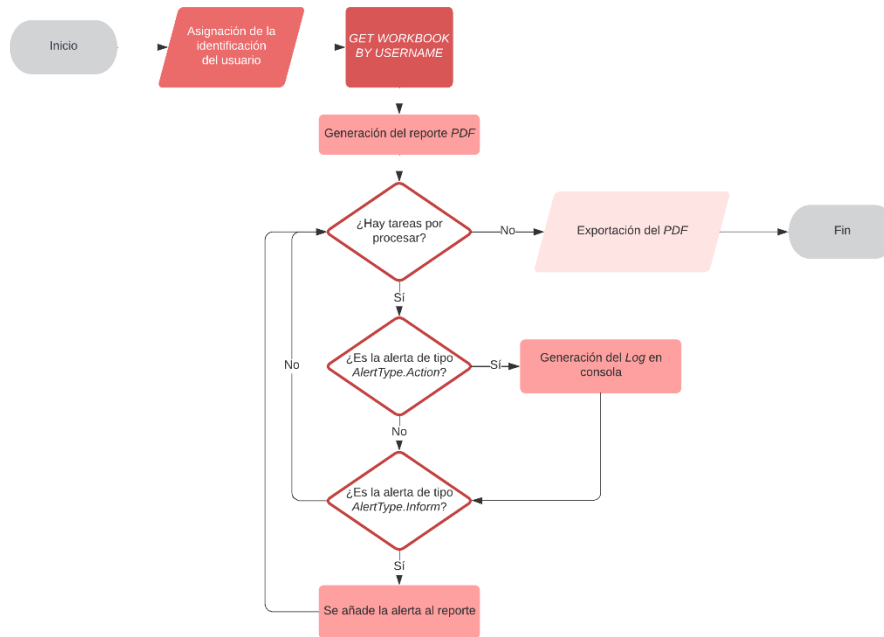


Figura 3.24 Diagrama de flujo del tercer microservicio

3.3.4 Manual de uso

A continuación, describimos cómo instalar el sistema en un docker y desplegar en él los microservicios necesarios para su ejecución como una prueba de concepto, así como instrucciones para esta ejecución.

Como detallamos en el proceso del desarrollo de la aplicación, toda la aplicación se ha *dockerizado* y cada microservicio puede utilizarse de forma independiente. También se ha *dockerizado* una instancia Redis, para facilitar el despliegue de la aplicación y solo necesitar *Docker* para ponerla en marcha. Se ha usado además *Docker Compose*, para gestionar multi-contenedores en *Docker*. El archivo ***docker-compose.yml*** que hemos utilizado tiene la siguiente estructura (servicios):

1. **Redis.** Contiene la imagen de Redis en su versión 1. Se despliega en el puerto por defecto, el 6379. Hemos querido añadir esta imagen para despreocuparnos de posibles problemas al configurar la base de datos en cualquier entorno donde se utilice finalmente la aplicación, ya sea un servidor en la nube o el dispositivo (PC, Raspberry Pi...) del usuario.
2. **Redis-Gateway.** Es el microservicio que actúa como intermediario entre el resto de los microservicios y la instancia Redis donde almacenamos nuestros “eventos”. La relación de comunicación entre los diferentes microservicios se realiza mediante este *gateway*, y en ningún momento se realiza una llamada directa entre dos microservicios. Utilizamos el puerto 8082 y el 8092 para depurar mediante *Remote JVM* si hemos usado *docker* para desplegarlo.

3. **Sensors.** Es el microservicio encargado de contactar con los diferentes sensores y crear los eventos tipo **Signal**. Para usar su API REST tendremos que utilizar el puerto 8080 y el 8091 para debugear mediante *Remote JVM*.
4. **Analyzer.** Es el microservicio encargado de analizar los eventos **Signal** y crear posteriormente los eventos **workbook**. Utiliza el puerto 8083 y para depurar el 8093 mediante *Remote JVM*.
5. **Execution.** Finalmente es el microservicio encargado de analizar los eventos **Workbook** y crear acciones ante estos. Utiliza el puerto 8082 y para depurar el 8092 mediante *Remote JVM*.

Aunque el proyecto que se entrega cuenta con los archivos compilación, es recomendable volver a compilarlo, por lo que, para ello, será necesario tener instalado *Maven*. Una vez instalado, podremos compilarlo simplemente con la opción “*mvn clean install*”. Esta instrucción deberá terminar con un mensaje “*BUILD SUCCESS*”, si todo ha ido bien.

El siguiente paso consiste en levantar la aplicación, para ello necesitamos tener instalado tanto *Docker* como *Docker Compose*. Utilizaremos la instrucción “*docker-compose up*” que se encargará de hacerlo. Esta instrucción iniciará todos los microservicios a la vez, pero si solo queremos utilizar uno de ellos, basta con utilizar por ejemplo “*docker-compose up sensors*” y solo se iniciará el microservicio *sensors*. Si es la primera vez que realiza esta acción Docker necesita hacer la *build* del proyecto y podría tardar algo más de tiempo. Si no se indica ningún mensaje de error en consola, podemos comenzar a hacer uso de nuestra aplicación.

Se ha adjuntado dentro del código en la ruta “*/postman*” una *colección postman* que permitirá hacer uso de la aplicación de forma sencilla y que tiene todos los *endpoints* de interés de nuestra aplicación. Se agrupan en cuatro carpetas principales, una por cada microservicio que describimos al principio de esta sección.

Para cada uno de los microservicios, se ha incorporado la carpeta *Healthcheck* en la colección *postman*. Esta carpeta contiene el *endpoint PING* que permitirá comprobar que el microservicio se ha iniciado correctamente y está listo para su uso. Si todo esta correcto, debe devolver “*pong*” (ver la Figura 3.25).

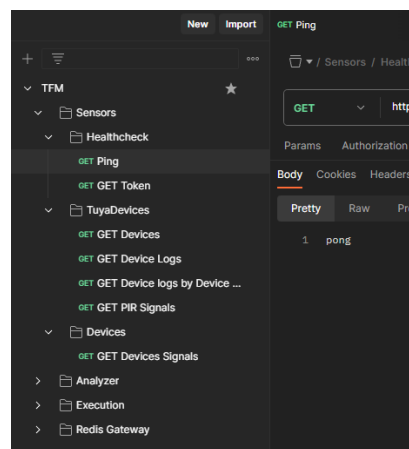


Figura 3.25 Healthcheck del microservicio Sensors

El recorrido básico que se debe seguir para usar esta primera prueba de concepto se ha realizado en la Figura 3.27. Por cada paso en el ciclo se le ha añadido un identificador a la captura, yendo desde el paso 1 al 7. Los pasos que se detallan son los siguientes:

1. **GET DEVICE SIGNALS** (*Sensors > Devices*). Este *endpoint* requiere el uso de cuatro parámetros como explicamos en el desarrollo los cuales son la fecha de inicio y la fecha de fin de la que queremos recuperar los datos de los sensores, el nombre de usuario al que están asociados los sensores, y de manera opcional si queremos activar el modo *debug* del reloj de Samsung junto a la parametrización deseada (este paso corresponde a la captura con el ID A en la Figura 3.27).
2. **GET SIGNALS BY USERNAME** (*Redis Gateway > Signal*). Debemos comprobar que una vez se han recopilado los datos de los sensores, se han almacenado en *Redis* para el usuario indicado antes. Ejecutando este *endpoint* utilizando como *Path Param* el nombre de usuario deseado, debería devolver inmediatamente los eventos de tipo *SIGNAL* asociados a este usuario (este paso corresponde a la captura con el ID B en la Figura 3.27).
3. **GET ANALYZER SIGNALS** (*Analyzer > Analyzer*). Este *endpoint* solo requiere que le pasemos como *Path Param* el nombre de usuario sobre el que recopilamos los eventos *signals* relacionados y construye los eventos *workbook* correspondientes. Adicionalmente, debemos comprobar dos cosas:
 - a. Al ejecutar de nuevo el *endpoint* **GET SIGNALS BY USERNAME** (*Redis Gateway > Signal*) para ese usuario, ya no debe aparecer nada, solo una lista vacía, ya que al crearse el nuevo evento *workbook* todos los eventos *signals* que han sido analizados se han debido borrar de *Redis* (este paso corresponde a la captura con el ID C en la Figura 3.27).
 - b. Al finalizar el análisis, y si se han encontrado datos de interés, este *endpoint* debe devolver un evento (o eventos) *workbook* (este paso corresponde a la captura con el ID D en la Figura 3.27).

Este *endpoint* corresponde al paso indicado con el ID C en la Figura 3.27)

4. **GET WORKBOOKS BY USERNAME** (*Redis > Workbook*). Al igual que en el caso anterior para el caso de las señales, los eventos generados del tipo *workbook* han debido ser almacenados y pasándole como *Path Param* se deben devolver todos los eventos de este tipo para el usuario indicado que están pendientes de ejecutar por el siguiente microservicio (este paso corresponde a la captura con el ID E en la Figura 3.27).
5. **GET EXECUTE ACTION** (*Execution > Execute*). Con este *endpoint* se genera el PDF de reporte los eventos del tipo *workbook* para el usuario indicado mediante *Path Param* (este paso corresponde a la captura con el ID F en la Figura 3.27). Debemos realizar dos comprobaciones tras ejecutar este *endpoint*:
 - a. Al igual que en la comprobación de los eventos de tipo *signal*, tras ejecutar este *endpoint* deben desaparecer de *Redis* los eventos de tipo *workbook* que han sido utilizados para generar el informe. Si volvemos a ejecutar **GET WORKBOOKS BY USERNAME** (*Redis > Workbook*) para ese usuario que estamos utilizando, se debe mostrar una lista vacía (este paso corresponde a la captura con el ID G en la Figura 3.27).

- b. Para las alertas que tengan el tipo *AlertType.ACTION*, si nos fijamos en los *logs* de la aplicación (ver la Figura 3.26), se muestra el mensaje que se le enviaría al usuario inmediatamente para alertarle de algún peligro. En Figura 3.26 se identifican dos alertas del tipo *AlertType.ACTION* en la que se alerta al usuario de la aplicación de peligro.

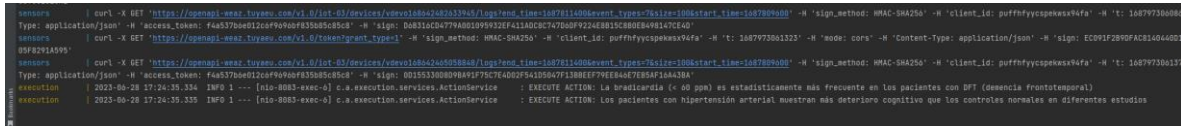


Figura 3.26 Logs para alertas de tipo *AletType.ACTION*

En la Figura 3.27 se puede observar el ciclo normal de la aplicación que acabamos de explicar.

En el conjunto de las capturas que hemos utilizado para ilustrar la prueba de concepto, existen un conjunto de *endpoints* para cada microservicio que no se han explicado. Los describiremos a continuación (no se tendrán en cuenta los tipos *Healthcheck*):

- **GET TOKEN** (*Sensors > Healthcheck*). Debido a que nuestra aplicación depende de que el servicio de Tuya Smart esté disponible, se ha diseñado este *endpoint* para comprobar justo esto. En el caso de que todo esté correcto, se deberá devolver como respuesta el *token* temporal de usuario generado a partir de su *accessId* y *accessKey*.
- **GET DEVICES** (*Sensors > TuyaDevices*). Este *endpoint* permite conocer todos los dispositivos asociados a la cuenta de un usuario.
- **GET DEVICE LOGS BY DEVICE ID** (*Sensors > TuyaDevice*). Para cada *deviceId*, se muestran todos los *logs* asociados para un periodo de tiempo. Los datos aparecen en bruto, sin traducción de qué significa cada valor.
- **GET DEVIC LOGS** (*Sensors > TuyaDevice*). Igual que en el caso anterior, pero se recopilan los *logs* de todos los dispositivos asociados a esa cuenta.
- **GET PIR SIGNALS** (*Sensors > TuyaDevice*). Los datos del *endpoint* anterior son filtrados a un formato comprensible.
- **POST PDF BY USERNAME** (*Execution > PDFExporter*). Con este *endpoint* podemos simular la creación de un reporte. Es una petición *POST* en la que en el cuerpo del mensaje le indicamos en formato *JSON* el conjunto de eventos del tipo **workbook**. Si la estructura de los eventos es correcta, el servicio tomará este cuerpo del mensaje y lo utilizará para crear el reporte en formato *PDF*.

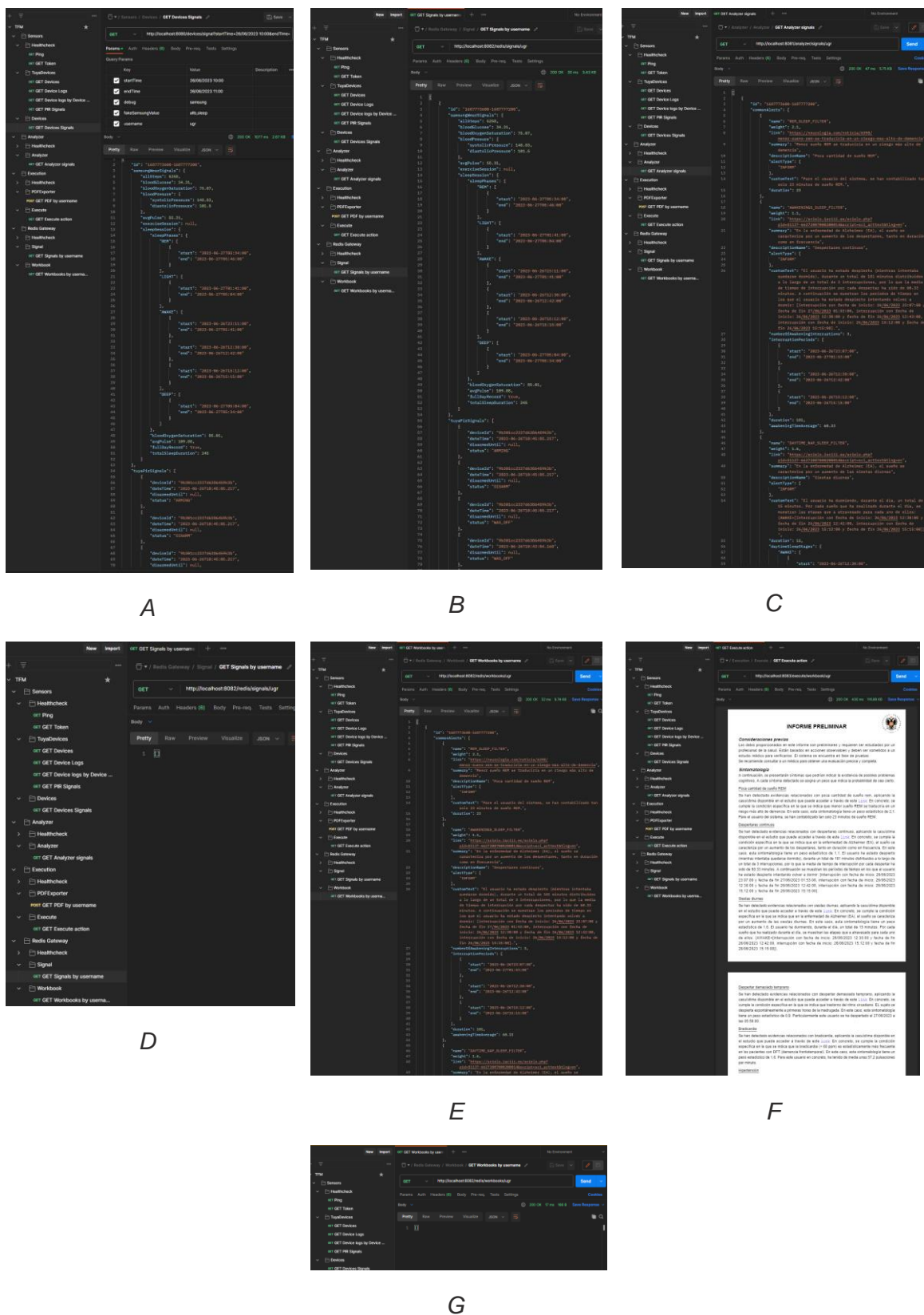


Figura 3.27 Ciclo normal de la aplicación usando Postman

3.4 Prototipado de la aplicación móvil

Pese a que no se ha hecho un desarrollo real de la aplicación para dispositivos móviles, en esta sección se mostrará un prototipado inicial de las principales funciones.

En Figura 3.28 se muestra el primer flujo de la aplicación. Nada más abrirla se mostrará la pantalla de login, donde se le mostrará al usuario tanto la posibilidad de iniciar sesión (*log in*), como la de registrarse (*sign up*). Las correspondientes capturas con el *ID A* y *B* de la Figura 3.28.

Cabe destacar de que en caso de que se registre por primera vez, será necesario conceder el acceso a su cuenta personal de Tuya Smart. Una manera con la que se podrían evitar errores y agilizar el registro del sistema es permitiendo el uso de la pasarela de autenticación de Google. Tuya Smart actualmente permite usarla y de esta forma se agilizaría el proceso de adquirir tanto el *accessToken* como el *accessId* de Tuya que requerimos para que funcione nuestro sistema. Para ello, tan solo se tendría que añadir una ventana adicional en la que la pasarela de Google requiere el uso de estos datos (como la captura con el *ID C* de la Figura 3.28).

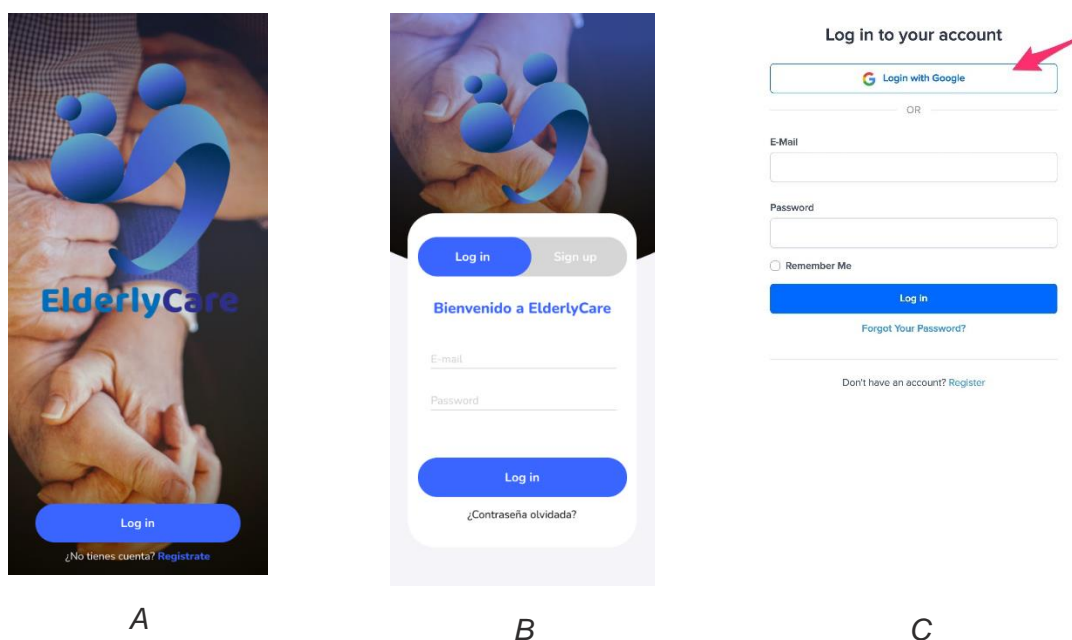


Figura 3.28 Flujo de inicio de la aplicación

Una vez hemos iniciado sesión, lo realmente interesante es el funcionamiento principal de la aplicación como puede verse en la Figura 3.29. Como bien indicamos, nuestra propuesta pretende crear alertas de problemas reales que sean captados por nuestros sensores, así como analizar la posibilidad de casos de deterioro cognitivo en etapas iniciales.

Para el primer caso, como se puede observar en la captura de la izquierda en la Figura 3.29, se muestran alertas de situaciones que el sistema considera peligrosas, al requerir de una acción lo más pronta posible, como posibles subidas de tensión repentinas o la ocurrencia de un accidente, como una caída en alguna estancia del hogar. En el caso de

que se lance alguna de estas alertas, la aplicación notificará a quien este cargo de la persona mayor, que podrá acceder a información detallada de los datos que ha recogido el sensor, con los que podrá decidir qué hacer consecuentemente (avisar a emergencias, algún vecino...). Si el usuario tutor estuviese interesado en una alerta en particular, podría utilizar el buscador para filtrarla de entre todas las alertas (captura con el *ID A* de la Figura 3.29).

Por otro lado, en la captura con *ID B* de Figura 3.29 se muestra el apartado de informes, al que se accederá simplemente pulsado la cabecera con el mismo nombre. El usuario podrá filtrar estos informes tanto por fecha como el peso final de los mismos (sumatorio de la gravedad de los problemas detectados). Por cada uno de ellos, tendrá la posibilidad de descargarlo, por si deseara tener una copia local y enviársela a algún profesional, o bien eliminarlo completamente del sistema.

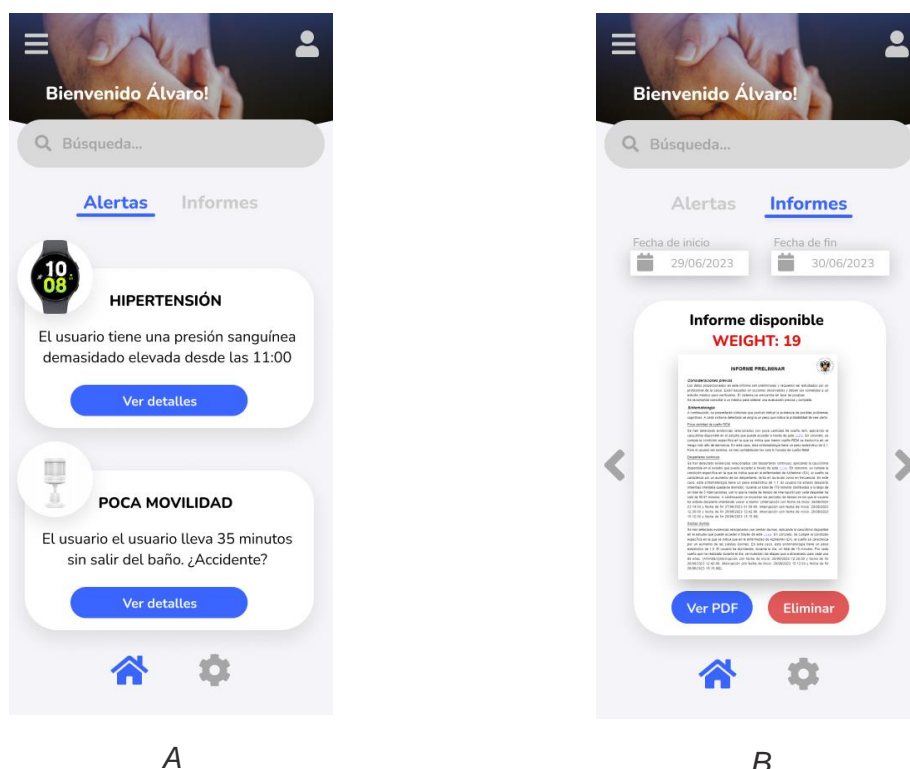


Figura 3.29 Funcionamiento principal de la aplicación

Además, como somos conscientes de que cada persona tiene unas características propias especiales que nuestro sistema debe tener en cuenta y adaptarse a ellas, también se dispondrá una sección especial de ajustes para configurar los sensores, pulsando en el menú con el icono de tuerca tal y como puede ver en la Figura 3.30.

En primer lugar, en la primera parte de la Figura 3.30 se mostrarán todos los sensores asociados a la cuenta que tiene la sesión iniciada en la aplicación, clasificados según su tipología. De esta forma se puede navegar por los diferentes tipos de sensores utilizando el

filtro horizontal de la parte superior, o buscar directamente el sensor por su nombre utilizando la barra de búsqueda.

Para cada uno de estos sensores, se indica su nombre, su estado actual (conectado o no al sistema), y además se habilita la opción de personalización, que llevará al usuario a una nueva ventana secundaria como puede observar en la parte derecha de Figura 3.30.

Cada sensor contará con una configuración personalizada según su tipología y las alertas que pueda detectar. Si bien como en el caso de ejemplo los sensores de movimiento tienen acciones muy limitadas desencadenantes de alertas, para otros tipos de sensores como por ejemplo los relojes contaríamos con muchas más opciones de personalización, como serían elegir una hora estándar para levantarse cada mañana, tensión media normal del usuario (para las alertas de hipertensión e hipotensión), etcétera.

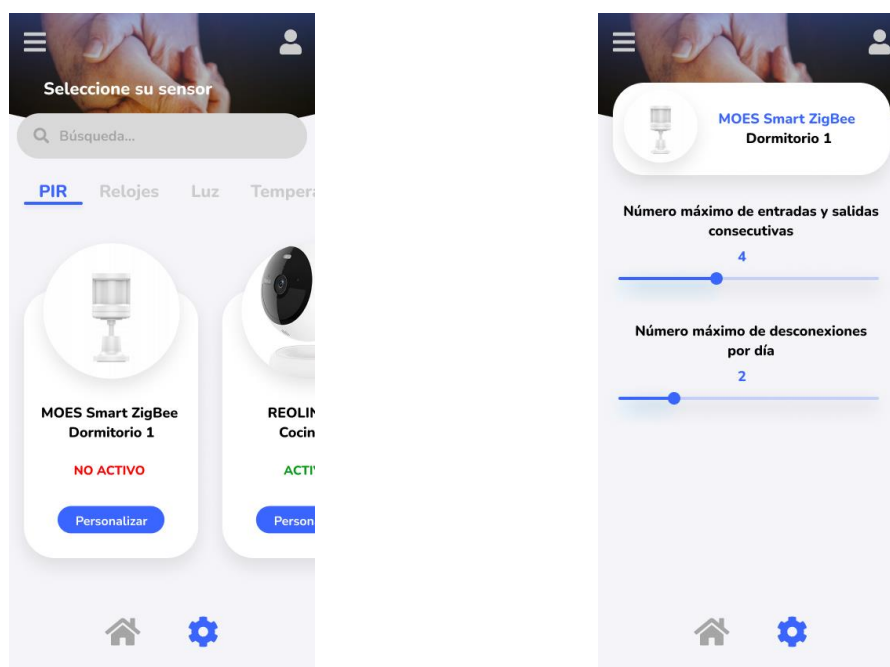


Figura 3.30 Personalización de los sensores

Como se ha podido ver en las capturas, en este primer prototipo no se permite modificar la frecuencia de actualización, ni de la recogida de datos de los sensores, ni de la generación de datos. Una posible comercialización del sistema, como bien se indicó antes, sería apostar por un sistema tipo *freemium* donde todos los usuarios tendrían la funcionalidad que hemos detallado hasta ahora, contando quizás con la posibilidad de mostrar anuncios publicitarios, y una funcionalidad *premium*, donde se podría ajustar estas frecuencias de actualización a gusto del usuario de la aplicación, entre otras funcionalidades adicionales, sin mostrar ningún contenido publicitario.

3.5 Planificación temporal

A continuación, mostraremos cuál ha sido la planificación del desarrollo de prueba de concepto, invirtiéndose unas 176 horas de trabajo. La cifra restante hasta completar las 300

horas (número de horas mínimas requeridas para un TFM) se han empleado en la elaboración de la memoria, búsqueda de información relativa a los problemas cognitivos y estudio de las tecnologías actuales que pudieran ser útiles para nuestra propuesta.

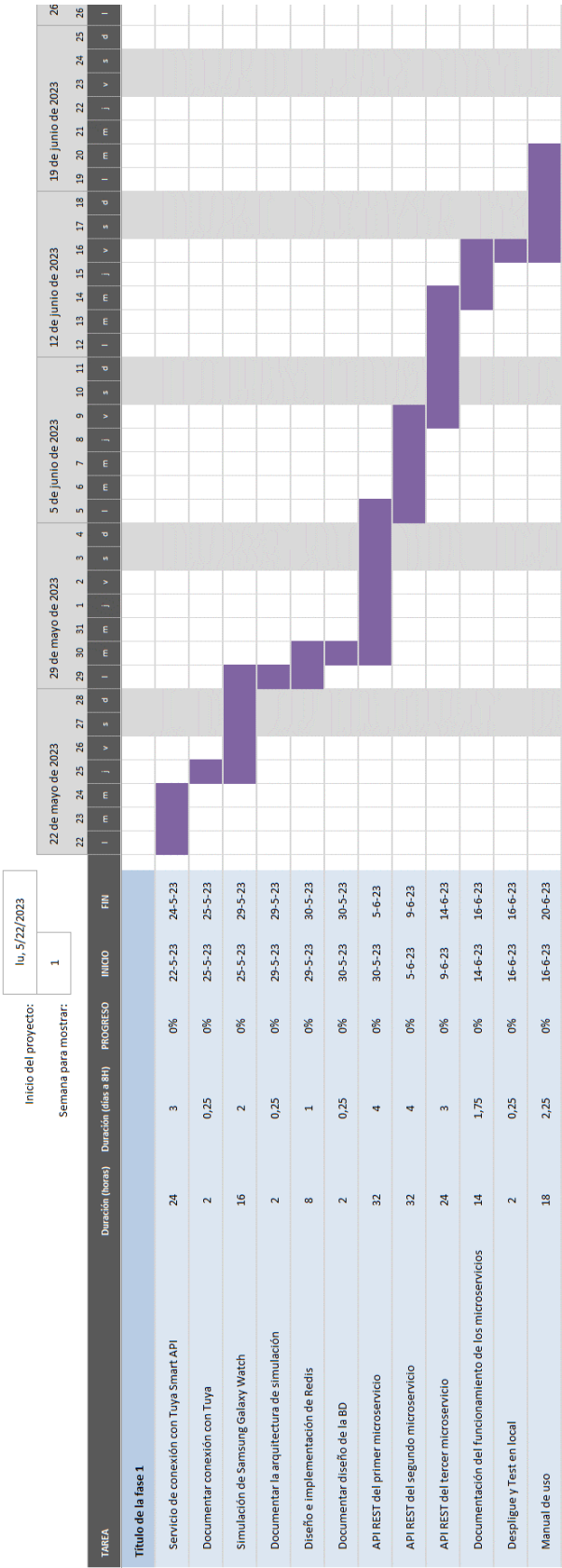


Figura 3.31 Diagrama de Gantt

En la Figura 3.31 se puede observar el diagrama de Gantt para las tareas que se han planificado para este proyecto. En total se han definido 12 tareas iniciales. Las tareas que se han planteado son:

1. Servicio de conexión con Tuya Smart API

El objetivo de esta tarea es establecer la conexión con la API de Tuya Smart y permitir la recogida de datos de los distintos sensores que estuviesen asociados a la cuenta del usuario.

Esta tarea tendrá una duración de 24 horas, lo que corresponderá a 3 días de trabajo de un desarrollador.

2. Documentar conexión con Tuya

Esta tarea consistió en detallar técnicamente en nuestra memoria el proceso anterior, haciendo mención también a los problemas a los que nos enfrentamos en el desarrollo.

Esta tarea tendrá una duración de 2 horas, lo que corresponde a 0,25 días de trabajo de un desarrollador.

3. Simulación de Samsung Galaxy Watch

Durante el desarrollo de esta tarea, a parte desarrollar la lógica de simular los datos, se utilizó toda la información que se había recopilado para la elaboración de la memoria para ponerla en común con signos de deficiencia cognitiva que se podían detectar mediante los datos recogidos por el reloj.

Esta tarea tendrá una duración de 16 horas, lo que corresponderá a 2 días de trabajo de un desarrollador.

4. Documentar la arquitectura de simulación

Esta tarea consistió en documentar en la memoria toda la lógica que permite usar el sistema, para simular tanto los datos que se mantienen dentro de los márgenes recomendados como válidos, como los que sobrepasan o quedan por debajo de los límites.

Esta tarea tendrá una duración de 2 horas, lo que corresponderá a 0,25 días de trabajo de un desarrollador.

5. Diseño e implementación de Redis

Diseño e implementación de la base de datos que almacenará las diferentes señales recogidas a los sensores que se almacenarán como un evento *Signal* y así como las señales analizarán que serán almacenadas como un evento *Workbook*. Además, se deberá construir también un servicio *gateway* que permita la comunicación abstracta entre los diferentes microservicios y la instancia de base de datos Redis.

Esta tarea tendrá una duración de 8 horas, lo que corresponderá a 1 día de trabajo de un desarrollador.

6. Documentar el diseño de la BD

Dejar constancia de la arquitectura usada finalmente del servicio *gateway* desarrollado para conectar la base de datos Redis con el resto de microservicio, su funcionamiento y como establecer la conexión con este.

Esta tarea tendrá una duración de 2 horas, lo que corresponderá a 0,25 días de trabajo de un desarrollador.

7. API REST del primer microservicio

Se continuará con el desarrollo de la primera actividad (la conexión con *Tuya Smart*) y la tercera (simulación de *Samsung Watch*) construyéndose un microservicio completo capaz de recopilar todos los registros de los distintos sensores (para cada usuario) y construir un evento (de tipo *signal*) para que estos datos sean analizados posteriormente. Además, todos los eventos deben ser almacenados de forma persistente utilizando el *gateway* de *Redis*.

Esta tarea tendrá una duración de 32 horas, lo que corresponderá a 4 días de trabajo de un desarrollador.

8. API REST del segundo microservicio

El microservicio construido deberá ser capaz de contactar con el *gateway* de *Redis* para recuperar todos los eventos *signal* de cada usuario. Por cada evento se deberán analizar las diferentes razones por las que se podría lanzar una alerta (se deberá estudiar posibles casuísticas). Una vez analizados los datos, se guardarán los análisis en el evento *workbook* que analizará el último microservicio.

Esta tarea tendrá una duración de 32 horas, lo que corresponderá a 4 días de trabajo de un desarrollador.

9. API REST del tercer microservicio

El desarrollador deberá ser capaz de crear el nuevo microservicio que analice los eventos *signal*, construya un informe en forma de documento PDF y ejecute acciones de alerta que consistirán en mensajes *log* en la consola de la aplicación.

Esta tarea tiene una duración de 24 horas, lo que corresponderá a 3 días de trabajo de un desarrollador.

10. Documentación del funcionamiento de los microservicios

Se deberá documentar la comunicación entre los distintos microservicios y el *gateway* de *Redis*, como acceder a ellos para realizar pruebas localmente sin

necesidad de la aplicación Android, para favorecer el futuro mantenimiento y desarrollo.

Esta tarea tendrá una duración de 14 horas, lo que corresponderá a 1,75 días de trabajo de un desarrollador.

11. Despliegue y Test en local

Se comprobará el correcto funcionamiento del sistema en diversos dispositivos utilizando el despliegue local mediante *Docker* realizando un ciclo completo de toda la funcionalidad.

Esta tarea tiene una duración de 2 horas, lo que corresponderá a 0,25 días de trabajo de un desarrollador.

12. Manual de uso de la aplicación Android

Esta tarea consiste en realizar un manual de usuario de la aplicación backend, explicando paso a paso el funcionamiento de casa *endpoint* y la puesta en contexto de cada funcionalidad dentro del microservicio.

Esta tarea tendrá una duración de 18 horas, lo que corresponderá a 2,25 días de trabajo de un desarrollador.

Téngase en cuenta que en esta planificación sólo se ha tenido en cuenta el desarrollo general de la aplicación. Quedaría por completar un plan de pruebas completo, en el que se testee la aplicación con usuarios reales y profesionales que ayuden a calibrar los filtros creados para cada una de las alertas diseñadas, así como también podrían añadirse nuevos tipos de alertas.

3.6 Presupuesto del desarrollo

El coste total del proyecto es de 4349,99 €. Este gasto se desglosa en coste de personal y coste de material.

3.6.1 Coste de personal

Según el portal Glassdoor (2023), el sueldo bruto para un ingeniero software junior a tiempo completo en España es de unos 25956 euros brutos al año. Teniendo en cuenta que en España se establecen 1.800 horas anuales máximas de trabajo según Convenio Colectivo Estatal de Empresas de Consultoría y Estudios de Mercado y de la Opinión Pública, Orden ESS/1451/2015 de 19 de junio, publicado en el Boletín Oficial del Estado, núm. 156, de 1 de julio de 2015, páginas 52095-52130, se tomará 14.42 € la hora como sueldo estándar para un desarrollador como el autor de este proyecto.

Según queda establecido en el programa docente de este máster, la asignatura *Trabajo Fin de Máster* tiene una asignación de 12 créditos. En el ámbito universitario, el Real Decreto 1393/2007, de 29 de octubre, establece la estructura de las enseñanzas universitarias y establece que un crédito ECTS (*European Credit Transfer and Accumulation System*)

equivale a 25 horas de trabajo del estudiante. Este valor de referencia se aplica a los estudios de grado, máster y doctorado. Por tanto, el número máximo de horas que debería implicar este proyecto es de 300 horas.

Si aproximamos las horas de desarrollo, así como de elaboración de documentación que requerían el desarrollo de esta propuesta al completo, se aproximaría a las 300 horas de trabajo, por lo que el **coste de personal** sería de alrededor de **4326 €**.

3.6.2 Costes de material

Para la primera fase inicial de proyecto, solo se compró un sensor de movimiento, aunque idealmente hubieran sido requeridos al menos 5 (puerta principal, salón, cocina, baño y dormitorio) además del reloj *Samsung Galaxy Watch 5*.

Al momento de desarrollar la memoria, el sensor que fue adquirido y utilizado físicamente ha sido “*MOES Smart ZigBee PIR*”, que tiene un precio en Amazon de 23,99€. En cuanto al *Samsung Galaxy Watch 5* tiene un valor de 236,57 € en su configuración más básica, aunque los valores de este dispositivo se han generado automáticamente mediante un simulador utilizado para nuestra prueba de concepto.

No se han incluido costes de despliegue, ya que, por el momento, solo se utilizó nuestra propuesta de forma local en nuestro propio equipo.

La suma total real de coste de materiales asciende a una cifra de 23,99 €. En el caso de que se hubiese adquirido el reloj, hubiese tenido un precio de 260,56 €.

4 CONCLUSIONES Y TRABAJOS FUTUROS

Para finalizar, expondremos las principales conclusiones del trabajo realizado, así como posibles mejoras que se podrían llevar a cabo en un futuro para extenderlo y/o mejorarlo.

4.1 Conclusiones

A nivel personal, nos sentimos profundamente convencidos de que la solución que hemos planteado tiene un potencial transformador y podría ser una fuente de beneficios para innumerables personas. Nuestra visión tiene la esperanza de marcar una diferencia significativa en la vida de aquellos que más lo necesitan. Sin embargo, en el transcurso hacia la realización de nuestra propuesta, nos hemos encontrado con desafíos inevitables. Lamentablemente, debido a restricciones de coste, no hemos podido incorporar todos los sensores que originalmente planeábamos utilizar. Esto significa que no podemos mostrar plenamente el potencial completo de nuestra propuesta en su forma actual.

Es crucial reconocer que, para llevar a cabo un verdadero impacto, es necesario reunir y compartir nuestros conocimientos e ideas con un equipo de expertos en el ámbito médico. Su experiencia y perspectiva serán fundamentales para validar y perfeccionar el funcionamiento de nuestra propuesta. Aunque sentimos cierta decepción por no poder desplegar todo el potencial de nuestra idea de manera inmediata, estamos comprometidos a colaborar con profesionales médicos para llevar a cabo pruebas reales y ajustar nuestra solución en beneficio de todos aquellos a quienes aspiramos ayudar.

Nuestra propuesta, si se confía en ella, se convierte en una garantía de simplicidad y eficacia en el mercado de sistemas de monitorización para personas mayores. Con la implementación de sensores compatibles con *Tuya*, hemos creado un enfoque revolucionario que transforma por completo la experiencia de la monitorización remota y de cuidado de personas mayores, especialmente de aquellas que viven solas y son más vulnerables. Con solo otorgar permiso de acceso de lectura a los diversos sensores, el usuario encargado de la aplicación puede despreocuparse por completo. No se requiere ninguna instalación engorrosa ni procesos complicados. Simplemente se sitúa en un estado de expectativa, esperando los resultados que nuestro sistema ofrece con garantías y de forma no intrusiva para las personas mayores.

Aunque pueda parecer complicado, estamos convencidos de que cada paso que demos nos acercará un poco más a hacer realidad la mejora en la calidad de vida y a brindar esperanza y tranquilidad a quienes más la necesitan.

En relación al cumplimiento de **los objetivos específicos que se plantearon**, estamos orgullosos de haber completado todos y cada uno de ellos. A continuación, se comenta cómo se ha logrado completar cada uno de ellos:

- **Analizar el estado de la domótica actual y su evolución a lo largo de los años:** Se ha realizado una revisión de la literatura al respecto y un resumen cronológico, que se muestra en los capítulos 2.2 y 2.3 de esta memoria.

- **Investigar el mercado actual de dispositivos domóticos:** Hemos investigado diferentes tipos de dispositivos para uso en el hogar, mostrando tablas comparativas de sus principales características. Esto puede verse también en el capítulo 2.9.
- **Desarrollar un sistema de procesamiento de datos:** Hemos sido capaces de diseñar un sistema de procesamiento de eventos, capaz de detectar situaciones inusuales a partir de los datos recopilados por los sensores. Esto puede verse en el capítulo 3.3.2, en el apartado del desarrollo de la propuesta.
- **Configurar los diferentes sensores:** Hemos logrado configurar un sensor que contábamos con el físicamente y el sistema construido es capaz de recopilar la información de este a partir de la cuenta de un usuario y almacenar de forma persistente todos los datos, de forma autónoma y abstracta, además de procesar estos datos y llevar a cabo la generación de informes a partir de ellos. Esto también puede verse en el capítulo 3.3.1.
- **Facilitar la interoperabilidad entre los sensores domóticos y el sistema que gestione los datos:** Como explicamos en el desarrollo de la propuesta del capítulo 3, optar por una arquitectura basada en microservicios y eventos contribuye al crecimiento de la aplicación. La modularidad que se proporciona favorece la interoperabilidad entre los distintos servicios y la rapidez de implementación de nuevas funcionalidades.
- **Diseñar una interfaz de usuario:** La interfaz de usuario diseñada, puede verse en la sección 3.4 *Prototipado de la aplicación móvil* del capítulo 3.
- **Organizar un entorno de pruebas:** Como la colección de consultas *Postman* que se han adjuntado a nuestro proyecto y permite evaluar toda la casuística que permite nuestra propuesta, tal y como se ha presentado en el capítulo 3, y más concretamente en la sección 3.3.4 Manual de uso.
- **Realizar una comparativa de los sistemas de monitorización existentes:** Los resultados de esta comparativa pueden verse en la valoración de los diferentes dispositivos de monitorización presentada en el capítulo 2.9.

4.2 Posibles futuras mejoras en funcionalidad y tecnologías

Después de llevar a cabo el desarrollo de nuestra propuesta, somos conscientes de futuras mejoras que añadirían un valor extra a nuestra propuesta. Existen varias maneras de extender y/o mejorar la funcionalidad que se propone. Básicamente, el conjunto de ellas no se ha integrado en el prototipado inicial, ya que es una prueba de concepto, pero podrían ser fácilmente añadidas en un futuro, en una evolución de la propuesta. A continuación, se presentan algunas posibles extensiones y mejoras:

- ***Integración con un sistema de reconocimiento facial:*** se pueden incluir cámaras con sistemas de reconocimiento facial para detectar la presencia y el movimiento de las personas, y así identificar a las personas que están en la habitación. Esto puede ayudar a garantizar la seguridad y la privacidad de las personas, al controlar o evitar el acceso de extraños a una determinada habitación o zona de la vivienda.
- ***Análisis de datos más avanzado:*** se pueden utilizar técnicas de aprendizaje automático para analizar los datos recopilados por los sensores y los actuadores. Por ejemplo, se pueden utilizar algoritmos de aprendizaje automático para detectar patrones de comportamiento y para predecir el comportamiento futuro. Esto puede ayudar a mejorar la precisión y la eficacia de las reglas establecidas y de las acciones automáticas.
- ***Integración con otros dispositivos domóticos:*** se pueden incluir otros dispositivos domóticos, como termostatos inteligentes o sistemas de audio, para mejorar la experiencia del usuario. Por ejemplo, se pueden utilizar altavoces inteligentes para reproducir una determinada música (del gusto del usuario) cuando se detecta su presencia en la habitación.
- ***Integración o combinación con tecnologías existentes***

La combinación de una monitorización una monitorización inteligente con los sistemas de teleasistencia, como el caso del sistema de teleasistencia andaluz, puede ser altamente beneficiosa para mejorar la calidad de vida de las personas mayores o con discapacidades.

Una posible integración podría ser la de utilizar la plataforma de teleasistencia (como la del sistema andaluz de asistencia) como un medio de comunicación adicional, en caso de que se detecte una situación de emergencia o de alerta. Si el sistema de monitorización que planteamos en esta propuesta detecta una situación de riesgo, como una caída o un cambio brusco de temperatura, se podría enviar automáticamente una señal de alerta a la plataforma de teleasistencia. A partir de ahí, el personal de la plataforma de teleasistencia podría tomar las medidas necesarias para atender la emergencia, como contactar con los familiares o enviar una ambulancia.

Estos profesionales tendrían además acceso a monitorizar en tiempo real los datos de los sensores, si nuestro sistema les alerta de que hay un problema, para poder así conocer qué está pasando realmente en una situación grave y con la inmediatez que requieren dichas situaciones.

5 REFERENCIAS

Alor-Hernández, J. A., Cruz Ramos, N. A., Alor-Hernández, G., Sánchez-Cervantes, J. L., & Rodríguez-Mazahua, L. (2022). Desarrollo de un Módulo para la Prevención de la Hipertensión usando el Paradigma IoT y Aprendizaje Automático. *Research in Computing Science*, 151(5), 91-104.

Amazon. (s. f.). What is the Alexa Skills Kit? Alexa Skills Kit. <https://developer.amazon.com/es-ES/docs/alexa/ask-overviews/what-is-the-alexa-skills-kit.html>

Alzheimer's Association. (s. f.). Las 10 Señales. ALZ. <https://www.alz.org/alzheimer-demencia/las-10-senales#:~:text=Desorientaci%C3%B3n%20de%20tiempo%20o%20lugar,est%C3%A1n%20y%20c%C3%B3mo%20llegaron%20all%C3%AD>

Bayón, A. R., Sampedro, F. G., & Quesada, J. T. (2014). Bradicardia en la Demencia Frontotemporal. *Neurología*, 29(2), 76-85.

Bernal Raspall, D. (2022, 1 febrero). El Apple Watch salva otra vida: un hombre mayor es rescatado tras sufrir una caída y perder el conocimiento a temperaturas bajo cero. Applesfera. <https://www.applesfera.com/apple-watch/apple-watch-salva-otra-vida-hombre-mayor-rescatado-sufrir-caida-perder-conocimiento-a-temperaturas-cero>

Castro Morales, W. E., García Criollo, I. E., & Muzzio Argüello, J. M. (2009). Diseño Domótico para Brindar Confort y Seguridad a un Asilo de Ancianos Mediante Comandos de Voz o Mandos a distancia. Escuela Superior Politécnica del Litoral.

Ceraso, D. (2001). Hipotensión Arterial y Shock. Hospital Juan A. Fernández, Buenos Aires, Argentina. Federación Argentina de Cardiología.

Chaparro, J. (2003). Domótica: la Mutación de la Vivienda. *Scripta Nova. Revista Electrónica de Geografía y Ciencias Sociales*, 7(146/136).

de la Cal, L. (2019, 4 diciembre). China te obliga a dar la cara. DPL News. <https://dplnews.com/china-te-obliga-a-dar-la-cara/>

Delgado Sanchis, A. J. (2016). Comunicación entre Módulos para la Construcción de Jardines Inteligentes (Doctoral Dissertation, Universitat Politècnica de València).

Echávarri, C., & Erro, M.E. (2007). Sleep Disorders in the Elderly and in Dementias. *Anales del Sistema Sanitario de Navarra*, 30 (Supl. 1), 155-161.

Europapress. (2020, 4 mayo). Apple Watch recoge síntomas de isquemia coronaria que las pruebas ECG del hospital no detectaron. Europapress Portal TIC. <https://www.europapress.es/portaltic/software/noticia-apple-watch-recoge-sintomas-isquemia-coronaria-pruebas-ecg-hospital-no-detectaron-20200504095812.html>

Ferradans Rodríguez, P., & Soto González, M. (2017). Terapia Cognitiva en Pacientes con Parkinson. *Ansiedad y Estrés*, 23(2-3), 104-109.

Fundación Pasqual Maragall. (2022, 11 noviembre). Hablemos del Alzheimer. El blog de la fundación Pascual Maragall. <https://blog.fpmaragall.org/aceptar-enfermedad-alzheimer>

Gálvez Díaz, N. del P. (2016). Alzheimer y Cuidados: Testimonios Reales de Personas con Familiares con Demencia (Trabajo Fin de Máster, Universitat Jaume I).

Gao, W., Cao, B., Shan, S., Chen, X., Zhou, D., Zhang, X., & Zhao, D. (2008). The CAS-PEAL Large-Scale Chinese Face Database and Baseline Evaluations. *IEEE Transactions on Systems, Man, and Cybernetics*, 38(1), 149-161. <https://doi.org/10.1109/tsmca.2007.909557>

Glassdoor. (2023). Sueldo Medio Software Developer en Granada. Recuperado 29 de junio de 2023, de https://www.glassdoor.es/Sueldos/granada-software-developer-sueldo-SRCH_IL.0,7_IC2614045_KO8,26.htm

Guerrero-Ulloa, G., Andrango-Catota, A., Abad-Alay, M., Hornos, M. J., & Rodríguez-Domínguez, C. (2023). Development and Assessment of an Indoor Air Quality Control IoT-Based System. *Electronics*, 12(3), 608. <https://doi.org/10.3390/electronics12030608>

Lacoma, T. (2022, 30 septiembre). The History of the Amazon Echo. DigitalTrends. <https://www.digitaltrends.com/home/history-of-amazon-echo/>

Niño Rivera, M. C. (2018). Desarrollo de un Prototipo de Aplicación Móvil para el Seguimiento de la Hipertensión Arterial Utilizando el Asistente Personal Alexa Voice Service (AVS) (Trabajo de Grado, Universidad Distrital Francisco José de Caldas).

Nuki (2022). Nuki for AirBnb. Nuki Shop. <https://nuki.io/es/airbnb/>

Olmedo-Aguirre, J. O., Reyes-Campos, J., Alor-Hernández, G., Machorro-Cano, I., Rodríguez-Mazahua, L., & Sánchez-Cervantes, J. L. (2022). Remote Healthcare for Elderly People Using Wearables: A Review. *Biosensors*, 12(2), 73. <https://doi.org/10.3390/bios12020073>

Quesada Moreno, J. F., García, F., Sena Pichardo, M. E., Bernal Bermejo, J. Á., & de Amores Carredano, J. G. (2001). Dialogue Management in a Home Machine Environment: Linguistic Components over an Agent Architecture. *Procesamiento del Lenguaje Natural*, 27, 89-96.

Samsung. (s. f.). Accessing Samsung Health Data through Health Connect. <https://developer.samsung.com/health/blog/en-us/2022/12/21/accessing-samsung-health-data-through-health-connect>

Samsung. (2022, 18 octubre). Measure Stress on Samsung Watch. About Samsung Galaxy Watch 5. <https://www.samsung.com/au/support/mobile-devices/measuring-stress-levels/>

Sánchez Blanco, L. (2022). Privación de Sueño y Neurodegeneración (Trabajo Fin de Grado, Universidad de Sevilla).

Smith, D. (2020, 10 abril). AoG ProTips: Test Suite for Smart Home. Medium. <https://medium.com/google-developers/aog-protips-test-suite-for-smart-home-929a181e91e9>

Sprenger, N., Shamloo, A. S., Schäfer, J., Burkhardt, S., Mouratis, K., Hindricks, G., Bollmann, A., & Arya, A. (2022). Feasibility and Reliability of Smartwatch to Obtain Precordial Lead Electrocardiogram Recordings. *Sensors*, 22(3), 1217. <https://doi.org/10.3390/s22031217>

Sarrais, F., & de Castro Manglano, P.. (2007). El insomnio. *Anales del Sistema Sanitario de Navarra*, 30(Supl. 1), 121-134. Recuperado en 28 de junio de 2023, de http://scielo.isciii.es/scielo.php?script=sci_arttext&pid=S1137-66272007000200011&lng=es&tlng=es

Pase, M. P., Himali, J. J., Grima, N. A., Beiser, A. S., Satizabal, C. L., Aparicio, H. J., Thomas, R. J., Gottlieb, D. J., Auerbach, S. H., & Seshadri, S. (2017). Sleep architecture and the Risk of Incident Dementia in the Community. *Neurology*, 89(12), 1244–1250. <https://doi.org/10.1212/WNL.0000000000004373>

Tai, K.-Y., Chiang, D.-L., Chen, T.-S., Shen, V. R. L., Lai, F., & Lin, F. Y.-S. (2020). Smart Fall Prediction for Elderly Care using iPhone and Apple Watch. *Wireless Personal Communications*, 114, 347-365.

Tuya Smart. (2023, 13 marzo). Cloud Services API Reference. Tuya Developer. <https://developer.tuya.com/en/docs/cloud/device-connection-service?id=Kb0b8geg6o761>

Vargas, D. C. Y., & Salvador, C. C. (2016). Smart IoT Gateway for Heterogeneous Devices Interoperability. *IEEE Latin America Transactions*, 14(8), 3900-3906. <https://doi.org/10.1109/tla.2016.7786378>

Vicario, A., Vainstein, N. E., Zilberman, J. M., del Sueldo, M., & Cerezo, G. H. (2010). Hipertensión Arterial: Otro Camino Hacia el Deterioro Cognitivo, la Demencia y las Alteraciones Conductuales. *Neurología Argentina*, 2(4), 226-233.