TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

TFG - DASIoT

DASIoT: Desarrollo y Auditoría de Seguridad para prototipo de dispositivos IoT

Autor
Luis Aróstegui Ruiz

Directores
José Manuel Soto Hidalgo
Alberto Guillén Perales

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

Granada, 13 de abril de 2022

TFG - DASIoT: Desarrollo y Auditoría de Seguridad para prototipo de dispositivos IoT

Luis Aróstegui Ruiz

Palabras clave: Internet de las Cosas, Seguridad

Resumen

En el contexto de IoT, hay un ecosistema de entornos de desarrollo específicos. En este TFG se hará uso de alguno de ellos para llevar a cabo la implementación de un dispositivo IoT y analizar los potenciales problemas de seguridad que pueden aparecer durante la etapa de implementación.

Project Title: Project Subtitle

First name, Family name (student)

Keywords: Keyword1, Keyword2, Keyword3, ...

Abstract

Write here the abstract in English.

Yo, **Nombre Apellido1 Apellido2**, alumno de la titulación TITULACIÓN de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI XXXXXXXXX, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Nombre Apellido1 Apellido2

Granada a X de mes de 201 .

D. **Nombre Apellido1 Apellido2 (tutor1)**, Profesor del Área de XXXX del Departamento YYYY de la Universidad de Granada.

D. **Nombre Apellido1 Apellido2 (tutor2)**, Profesor del Área de XXXX del Departamento YYYY de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado ***Título del proyecto, Subtítulo del proyecto***, ha sido realizado bajo su supervisión por **Nombre Apellido1 Apellido2 (alumno)**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a X de mes de 201 .

Los directores:

Nombre Apellido1 Apellido2 (tutor1) **Nombre Apellido1 Apellido2 (tutor2)**

Agradecimientos

Poner aquí agradecimientos...

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1

Introducción

1.1. Descripción y contexto

El Internet de las Cosas, o IoT, es un sistema de dispositivos informáticos, máquinas mecánicas y digitales, objetos, animales o personas interrelacionados que cuentan con identificadores únicos (UID) y la capacidad de transferir datos a través de una red sin que sea necesaria la interacción entre personas o entre ordenadores. [?]

Una "cosa" en el Internet de las Cosas puede ser una persona con un implante de monitor cardíaco, un animal de granja con un chip, un automóvil que tiene sensores incorporados para alertar al conductor cuando la presión de los neumáticos es baja o cualquier otro objeto natural o artificial al que se le pueda asignar una dirección de Protocolo de Internet (IP) y que sea capaz de transferir datos a través de una red. Cada vez más, las organizaciones de diversos sectores utilizan el IoT para operar de forma más eficiente, comprender mejor a los clientes para ofrecerles un mejor servicio, mejorar la toma de decisiones y aumentar el valor del negocio.

El concepto de *Internet of Things* fue propuesto en 1999 por el laboratorio de identificación automática del Instituto Tecnológico de Massachusetts (MIT). La UIT lo dio a conocer en 2005, empezando por China. El IoT puede definirse como *"datos y dispositivos continuamente disponibles a través de Internet"*. La interconexión de "cosas" (objetos) que pueden dirigirse de forma inequívoca y las redes heterogéneas constituyen el IoT. La identificación por radiofrecuencia (RFID), los sensores, las tecnologías inteligentes y las nanotecnologías son los principales contribuyentes a al IoT para una variedad de servicios. Con la drástica

reducción del coste de sensores y con la evolución de tecnologías como el ancho de banda, el procesamiento, los teléfonos inteligentes, la migración hacia el IPv6 y el 5G se está facilitando la adopción del IoT.

El IoT también ve todo como lo mismo, sin discriminar entre humanos y máquinas. Las "cosas" incluyen a los usuarios finales, los centros de datos (DC), las unidades de procesamiento, los teléfonos inteligentes, las tabletas, el Bluetooth, el ZigBee, la Asociación de Datos por Infrarrojos (IrDA), la banda ultraancha (UWB), las redes celulares, las redes Wi-Fi, los DC de comunicación de campo cercano (NFC), la RFID y sus etiquetas, los sensores y los chips, los equipos domésticos, los relojes de pulsera, los vehículos y las puertas de las casas. [?]

1.2. Motivación

Las personas de todo el mundo estan ya preparadas para disfrutar de las ventajas del Internet de las cosas (IoT). El IoT lo incorpora todo, desde el sensor corporal hasta la computación en la nube. Comprende los principales tipos de redes, como la distribuida, la de red, la ubicua y la vehicular, que han conquistado el mundo de la informática durante una década. Desde de estacionamiento de vehículos a su seguimiento, de la introducción de datos de pacientes a la observación de los pacientes a la observación de los pacientes, de la atención a los niños a la atención a los ancianos, de las tarjetas inteligentes a las tarjetas de campo cercano, los sensores están haciendo sentir su presencia. Los sensores desempeñan un papel fundamental en el IoT.

El IoT funciona en redes y estándares heterogéneos. Excepcionalmente, ninguna red está libre de amenazas y vulnerabilidades de seguridad. Cada una de las capas del IoT está expuesta a diferentes tipos de amenazas. Este proyecto se centra en las posibles amenazas que hay que abordar y mitigar para conseguir una comunicación segura en el IoT. [?]

En otras palabras, el IoT combina "lo real y lo virtual" en cualquier lugar y en cualquier momento, atrayendo la atención tanto de desarrolladores como de ciberdelicuentes. Inevitablemente, dejar los dispositivos sin intervención humana durante un largo periodo podría dar lugar a robos. La seguridad ha sido finalmente reconocida como un requisito esencial para todo tipo de sistemas informáticos, incluidos los de IoT. Sin embargo, muchos sistemas IoT son mucho menos seguros que los típicos sistemas Windows/Mac/Linux.

Los problemas de seguridad de IoT se derivan de seguridad de la IO provienen de una serie de causas: características de seguridad inadecuadas en el hardware, software mal diseñado con una serie de vulnerabilidades, contraseñas por defecto

y otros errores de de seguridad. Los nodos IoT inseguros crean problemas para la seguridad de todo el sistema IoT. Dado que los nodos suelen tener una vida útil de varios años, la gran base instalada de de dispositivos inseguros creará problemas de seguridad durante algún tiempo. Los nodos IoT inseguros son ideales para los ataques de denegación de servicio. El ataque Dyn es un ejemplo de ataque basado en el IoT contra la infraestructura tradicional de Internet. Los sistemas IoT inseguros también causan problemas de seguridad al resto de Internet.

La privacidad está relacionada con la seguridad, pero requiere medidas específicas a nivel de aplicación la red y los dispositivos. No sólo hay que proteger los datos de los usuarios contra el robo, sino que la red debe diseñarse de forma que los datos menos privados no puedan utilizarse fácilmente para inferir datos más privados. [?]

1.3. Objetivos

El principal objetivo de este proyecto es hacer uso de un entorno de desarrollo específico para llevar a cabo una implementación de un dispositivo IoT y analizar los potenciales problemas de seguridad que pueden aparecer durante la etapa de implementación.

1.3.1. Objetivos específicos

- **OE.1:** Estudiar los distintos entornos de desarrollo para IoT y analizar funcionalidades y propiedades que se ajusten al proyecto.
- **OE.2:** Desarrollar un prototipo de aplicación para IoT utilizando un framework de desarrollo.
- **OE.3:** Explotación de una vulnerabilidad a nivel de dispositivo, protocolo, SO, aplicación o HW.

CAPÍTULO 2

Estado del arte

Antes de empezar a desarrollar los objetivos del proyecto es necesario conocer primero los fundamentos del Internet de las Cosas, esto engloba desde su funcionamiento, tipos de arquitecturas hasta los distintos estándares que existen.

2.1. Internet de las Cosas

El término "Internet de las Cosas" (IoT) se conoce desde hace unos años. En los últimos tiempos, está recibiendo más atención debido al avance de la tecnología inalámbrica. La idea básica se debe a la variedad de objetos, como RFID, NFC, sensores, actuadores, teléfonos móviles, que pueden interactuar entre sí teniendo una dirección distinta. El IoT permite a los objetos ver, oír, pensar y realizar trabajos haciendo que 'hablen' entre sí, para compartir y sincronizar información. El IoT transforma estos objetos de convencionales a inteligentes mediante la manipulación de sus tecnologías subyacentes, como los dispositivos integrados, las tecnologías de comunicación, las redes de sensores, los protocolos y las aplicaciones. [?]

La premisa básica y el objetivo de IoT es "conectar lo que no está conectado". Esto significa que los objetos que no están actualmente unidos a una red informática, es decir, a Internet, se conectarán para que puedan comunicarse e interactuar con personas y otros objetos. IoT es una transición tecnológica en la que los dispositivos nos permitirán sentir y controlar el mundo físico haciendo que los objetos sean más inteligentes y conectándolos a través de una red inteligente. Cuando los objetos y las máquinas pueden ser detectados y controlados a distancia a través de una red, se consigue una mayor integración entre el mundo físico y los ordenadores. Esto permite mejoras en las áreas de eficiencia, preci-

sión, automatización y habilitación de aplicaciones avanzadas.

El mundo del IoT es amplio y puede resultar algo complicado al principio debido a la abundancia de componentes y protocolos que engloba. En lugar de considerar IoT como un único tecnología, es bueno verlo como un paraguas de varios conceptos, protocolos y tecnologías, todos ellos dependientes a veces de un sector concreto. Aunque la amplia gama de elementos de IoT está diseñado para crear numerosos beneficios en las áreas de productividad y automatización, al mismo al mismo tiempo, introduce nuevos retos, como la ampliación del gran número de dispositivos y de la cantidad de datos que deben procesarse. [?]

2.1.1. Arquitecturas

En esta sección se muestran distintas arquitecturas para el IoT, que suele describirse como un proceso de cuatro etapas en el que los datos fluyen desde los sensores conectados a las çosas.ª través de una red y, finalmente, a un centro de datos corporativo o a la nube para su procesamiento, análisis y almacenamiento.

2.1.1.1. Arquitectura de tres capas

Normalmente, la arquitectura de IoT se divide en tres capas básicas: capa de aplicación, capa de red y capa de percepción, que se describen con más detalle a continuación.

- **Capa de percepción.** También conocida como capa de sensores, se implementa como la capa inferior en la arquitectura de IoT. La capa de percepción interactúa con los dispositivos y componentes físicos a través de dispositivos inteligentes (RFID, sensores, actuadores, etc.). Sus principales objetivos son conectar las cosas a la red IoT, y medir, recoger y procesar la información de estado asociada a estas cosas a través de los dispositivos inteligentes desplegados, transmitiendo la información procesada a la capa superior a través de las interfaces de la capa.

- **Capa de red.** También es conocida como la capa de transmisión, se implementa como la capa intermedia en la arquitectura de IoT. La capa de red se utiliza para recibir la información procesada proporcionada por la capa de percepción y determinar las rutas para transmitir los datos y la información al centro del IoT, los dispositivos y las aplicaciones a través de redes integradas. La capa de red es la más importante en la arquitectura de IoT, ya que varios dispositivos (hub, switching, gateway, cloud computing perform, etc.), y varias tecnologías de comunicación (Bluetooth o Wi-Fi) se integran en esta capa. La capa de red debe transmitir datos hacia o desde diferentes cosas o aplicaciones, a través de interfaces o pasarelas entre redes heterogéneas, y utilizando diversas tecnologías y protocolos de comunicación.

- **Capa de aplicación.** También conocida como la capa de negocio, se implementa como la capa superior en la arquitectura de IoT. La capa de aplicación recibe los datos transmitidos desde la capa de red y utiliza los datos para proporcionar los servicios u operaciones requeridos. Por ejemplo, la capa de aplicación puede proporcionar el servicio de almacenamiento para respaldar los datos recibidos en una base de datos, o proporcionar el servicio de análisis para evaluar los datos recibidos para predecir el estado futuro de los dispositivos físicos. Existen varias aplicaciones en esta capa, cada una con requisitos diferentes.

La arquitectura de tres capas es básica para el IoT y se ha diseñado y realizado en varios sistemas. Sin embargo, a pesar de la simplicidad de la arquitectura multicapa de IoT, las funciones y operaciones en las capas de red y aplicación son diversas y complejas. Por ejemplo, la capa de red no sólo necesita determinar rutas y transmitir datos, sino también proporcionar servicios de datos como agregación de datos, computación, etc. La capa de aplicación no sólo necesita proporcionar servicios a los clientes y dispositivos, sino que también debe proporcionar servicios de datos tales como minería de datos, análisis de datos, por ejemplo. Por lo tanto, para establecer una arquitectura multicapa genérica y flexible para la IoT, debe desarrollarse una capa de servicio entre la capa de red y la capa de aplicación para proporcionar los servicios de datos. Basándose en este concepto, recientemente se han desarrollado arquitecturas orientadas a los servicios (SoAs). [?]

2.1.1.2. Arquitectura basada en SoA

En general, SoA (Arquitectura Orientada a Servicios) es un modelo basado en componentes, que puede ser diseñado para conectar diferentes unidades funcionales, también conocidas como servicios, de una aplicación a través de interfaces y protocolos. SoA puede centrarse en el diseño del flujo de trabajo de los servicios coordinados, y permitir la reutilización de los componentes de software y hardware, mejorando la viabilidad de SoA para su uso en el diseño de la arquitectura IoT. Así, SoA puede integrarse fácilmente en la arquitectura de IoT, en la que servicios de datos proporcionados por la capa de red y la capa de aplicación en la arquitectura tradicional de tres capas pueden ser de la arquitectura tradicional de tres capas pueden extraerse y formar una nueva capa, la capa de servicios, también conocida como capa de interfaz o capa de middleware. Así, en una arquitectura de IoT basada en SoA, existen cuatro capas que interactúan entre sí, siendo éstas la capa de percepción, la capa de red red, la capa de servicio y la capa de aplicación.

En la arquitectura de IoT basada en cuatro capas, la capa de percepción desempeña la función de la capa inferior de la arquitectura, y se utiliza para medir, recoger y extraer los datos asociados a los dispositivos físicos. La capa de

red se utiliza para determinar las rutas y proporcionar soporte de transmisión de datos a través de redes heterogéneas integradas. La capa de servicio se sitúa entre la capa de red y la capa de aplicación, proporcionando servicios para apoyar la capa de aplicación. La capa de servicio consiste en el descubrimiento de servicios, la composición de servicios, la gestión de servicios y las interfaces de servicios. La capa de aplicación se utiliza para dar soporte a las solicitudes de servicio de los usuarios. [?]

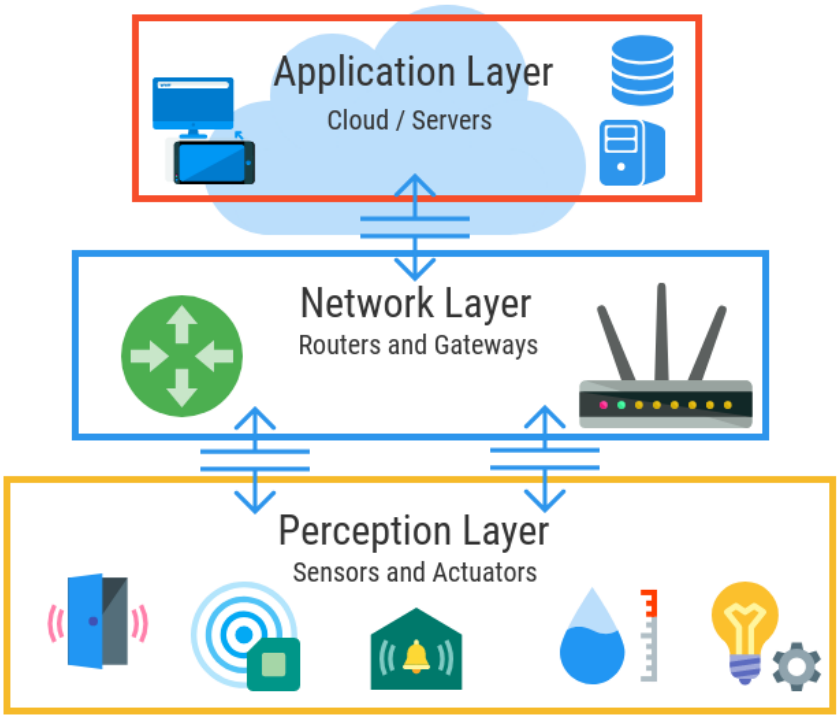

Figura 2.1: Arquitectura de 3 capas. [?]

2.1.2. Tecnologías asociadas

Teniendo en cuenta las arquitecturas mencionadas anteriormente, el IoT cuenta con varias tecnologías que hacen posible la cumplir el objetivo de cada capa de la arquitectura. A continuación se comentan sobre las tecnologías que nos encontramos en la Arquitectura basada en SoA. [?]- [?]

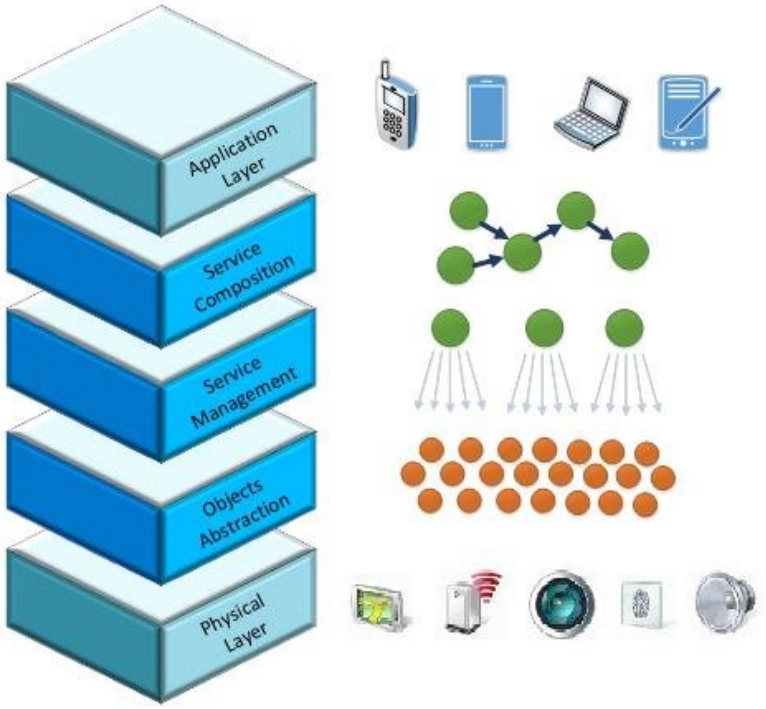

Figura 2.2: Arquitectura de 4 capas. [?]

2.1.2.1. Capa de percepción

En la capa de percepción, la función principal es identificar y rastrear objetos. Para lograr esta función, se pueden implementar las siguientes tecnologías pueden ser implementadas.

- **Radio frequency identification (RFID)**. El sistema RFID se compone de uno o varios lectores y varias etiquetas RFID. Utiliza campos electromagnéticos de radiofrecuencia para enviar los datos adjuntos. Las etiquetas que se adjuntan a ella, almacenan datos electrónicamente que pueden ser leídos por RFID cuando se encuentra en la proximidad del lector. [?]

- **Redes de sensores inalámbricos (WSN)**. Pueden desempeñar un papel muy importante en el IoT. Las WSN pueden monitorizar y rastrear el estado de los dispositivos, y transmitir los datos de estado al centro de control o a los nodos de enlace a través de múltiples saltos. Por lo tanto, las WSN pueden considerarse como un puente más entre el mundo real y el mundo cibernético. [?]

2.1.2.2. Capa de red

La capa de red se utiliza para determinar el enrutamiento y proporcionar soporte de transmisión de datos a través de redes heterogéneas integradas. A continuación, se presentan algunos protocolos que pueden permitir la comunicación fiable y segura en el IoT. [?]

- **IEEE 802.15.4**. Se lanzó a principios de 2003 y ha satisfecho la necesidad de dispositivos de bajo consumo. Es un protocolo diseñado para la capa física y la capa MAC en redes inalámbricas de área personal (WPAN). El objetivo de IEEE 802.15.4 es centrarse en las WPAN de baja velocidad, proporcionando las conexiones de baja velocidad de todas las cosas en un área personal con bajo consumo de energía, baja tasa de transmisión, y bajo coste. Una de las tecnologías que rompen los esquemas es el estándar de radio IEEE 802.15.4. [?]

- **6LoWPAN**. Las WPAN de baja potencia (LoWPAN) están organizadas por un gran número de dispositivos de bajo coste conectados mediante comunicaciones inalámbricas. En comparación con otros tipos de redes, las LoWPAN presentan una serie de ventajas, como el tamaño reducido de los paquetes, baja potencia, poco ancho de banda, etc. Como mejora, el protocolo 6LoWPAN fue diseñado combinando IPv6 y LoWPAN. En 6LoWPAN, los paquetes IPv6 pueden transmitirse a través de redes IEEE 802.15.4. Debido a que bajo coste y el bajo consumo de energía, 6LoWPAN es adecuado para el IoT, en el que se incluye un gran número de dispositivos de bajo coste.

- **ZigBee**. Es una tecnología de red inalámbrica, diseñada para la comunicación a corto plazo con bajo consumo de energía. En el protocolo ZigBee se incluyen cinco capas la capa física, la capa MAC, la capa de transmisión, la capa de capa de red y la capa de aplicación. Las ventajas de las redes ZigBee son el bajo consumo de energía, el bajo coste baja tasa de datos, baja complejidad, fiabilidad y seguridad.

- **Message Queue Telemetry Transport (MQTT)**. Usa la técnica de publicación/suscripción, es un protocolo de mensajería, que se utiliza para recoger datos medidos en sensores remotos y transmitir los datos a los servidores. MQTT es un protocolo simple y ligero, y soporta la red con bajo ancho de banda y alta latencia. MQTT puede implementarse en varias plataformas para conectar en Internet, y por lo tanto MQTT puede ser utilizado como un protocolo de mensajería entre sensores/actuadores y servidores, lo que hace que MQTT desempeñe un papel importante en el IoT.

- **Constrained Application Protocol (COAP)**. Es un protocolo de mensajería basado en la arquitectura de transferencia de estado representativa

(REST). Dado que la mayoría de los dispositivos del IoT tienen recursos limitados (es decir, poco almacenamiento y poca capacidad de cálculo), HTTP no puede utilizarse en el IoT, debido a su complejidad. Para superar este problema, se propuso CoAP para modificar algunas funciones de HTTP con el fin de satisfacer los requisitos de el IoT. En términos generales, CoAP es el protocolo de la capa de aplicación en la pila de protocolos 6LoWPAN, y tiene como objetivo permitir que los dispositivos con recursos limitados logren interacciones RESTful. La comunicación de grupo y la notificación push son compatibles con CoAP, pero la difusión no lo es. La observación de recursos, el transporte de recursos en bloque descubrimiento de recursos, interacción con HTTP, y seguridad son todas las características importantes proporcionadas por CoAP.

- **Advanced Message Queuing Protocol (AMQP)**. Es un protocolo de colas de mensajes de estándar abierto que se utiliza para proporcionar servicios de mensajes (colas, enrutamiento, seguridad, fiabilidad, etc.) en la capa de aplicación. AMQP se centra en los entornos orientados a mensajes y puede considerarse como un protocolo middleware orientado a mensajes. Usando AMQP, los clientes pueden lograr una comunicación estable con los middlewares de mensajes, incluso si estos clientes y middlewares son producidos por diferentes lenguajes de programación. [?]

2.1.2.3. Capa de servicio

Como se ha descrito anteriormente, la capa de servicio se encuentra entre la capa de red y la capa de aplicación, y proporciona servicios eficientes y seguros a los objetos o aplicaciones. En la capa de servicio, deben incluirse las siguientes tecnologías habilitadoras para garantizar que el servicio pueda prestarse de forma eficiente: tecnología de interfaz, tecnología de gestión de servicios, tecnología de middleware y tecnología de gestión y compartición de recursos.

- **Interfaz**. Debe diseñarse en la capa de servicio para garantizar el intercambio de información eficiente y seguro para las comunicaciones entre dispositivos y aplicaciones. Además, la interfaz debe gestionar eficientemente los dispositivos interconectados, incluyendo la conexión de dispositivos, la desconexión de dispositivos, la comunicación de dispositivos y el funcionamiento de dispositivos. Para dar soporte a las aplicaciones en el IoT, un perfil de interfaz (IFP) puede ser considerado como un estándar de servicio, que puede ser utilizado para facilitar las interacciones entre los servicios proporcionados por varios dispositivos o aplicaciones. Para lograr un IFP eficiente, debe implementarse el plug and play universal. Con el desarrollo del IoT, se han realizado varios esfuerzos sobre la interfaz. Por ejemplo, como el desarrollo de SoA-IoT, el proceso de provisión de servicios tiene la funcionalidad de proporcionar interacciones con aplicaciones y servicios. Aunque se han desarrollado varias tecnologías de interfaz para

el IoT, la implementación de más eficaces, seguras y escalables con bajo coste para apoyar al IoT.

- **Gestión de servicios**. Puede descubrir eficazmente los dispositivos y aplicaciones, y programar servicios eficientes y fiables para satisfacer las solicitudes. Un servicio puede considerarse como un comportamiento, que incluye la recogida, el intercambio y el almacenamiento de datos, o una asociación de estos comportamientos para lograr un objetivo especial. En el IoT, algunos requisitos pueden ser satisfechos por un solo servicio, mientras que otros deben ser satisfechos mediante la integración de múltiples servicios. Así, el servicio puede dividirse en dos categorías: servicio primario y servicio secundario. El servicio primario, también conocido como servicio básico, puede exponer las funcionalidades primarias en dispositivos o aplicaciones. En cambio, el servicio secundario servicio secundario puede lograr las funcionalidades auxiliares basadas en el servicio primario u otro servicio secundario. Para ocultar el detalle de la implementación de los servicios y hacer que estos servicios se implementen de forma compatible en dispositivos y aplicaciones heterogéneas, se ha utilizado SoA para integrar los servicios.

- **Middleware**. Es un software o servicio de programación que puede proporcionar una abstracción interpuesta entre las tecnologías de IoT y las aplicaciones. En el middleware se ocultan los detalles de las diferentes tecnologías y se proporcionan las interfaces estándar para que los desarrolladores puedan centrarse en el desarrollo de aplicaciones sin tener en cuenta la compatibilidad entre las aplicaciones y las infraestructuras. Así pues, mediante el uso de middleware, los dispositivos y las aplicaciones con diferentes interfaces pueden intercambiar información y compartir recursos entre sí. [?]

2.1.3. Aplicaciones

El IoT es una tecnología emergente que reclama la tercera posición después del ordenador e Internet. En esta tecnología, las cosas se conectan a Internet de una forma u otra, lo que da lugar a una enorme cantidad de datos que deben ser procesados, almacenados y representados de forma eficiente. Las aplicaciones de el IoT varían en diferentes campos e incluyen aplicaciones personales, industriales y nacionales. En los últimos años, estas aplicaciones han aumentado constantemente y algunas de ellas ya están desplegadas y se utilizan a diferentes niveles. Las aplicaciones de IoT requieren hardware, middleware y presentación. Estas aplicaciones tienen características como la comunicación de dispositivos con el mundo real, interacción con el entorno interacción entre personas y dispositivos, tareas rutinarias automáticas con menos supervisión, infraestructura organizada y seguridad en la comunicación. Son muchas las aplicaciones que se están desarrollando hoy en día y en los últimos años las aplicaciones de IoT han ido a más.

Las aplicaciones requieren un avance RFID y tecnologías de direccionamiento, con mejor visualización y capacidad de almacenamiento. Teniendo en cuenta los requisitos de IoT se ha desarrollado una arquitectura de aplicaciones para un menor consumo de energía. Algunos ejemplos que nos encontramos: [?]- [?]

- **Casas inteligentes**. En este caso nos encontramos nuevos inventos que se conectan y controlan nuestros hogares desde nuestros teléfonos.
- **Wearable Technology**. Se han diseñado e implementado varios productos diseñados y aplicados en este ámbito, desde anillos a calzado, y desde relojes hasta mochilas. Esta tecnología puede verse en todas partes.
- **Ciudades inteligentes**. Es un área de tecnología al servicio de las personas, y esta tecnología se construye esencialmente en torno al usuario. Una ciudad no es más que una red diseñada para optimizar los recursos.
- **Red inteligente**. Desde que las instalaciones del IoT se han añadido a la red anterior de datos, se han vuelto más inteligentes, lo que significa más y más información de la red. Desde que el IoT ha ayudado a convertirse en el sistema de suministro de energía de dos maneras la información se enviará al centro principal sobre el uso y el consumo.
- **Atención sanitaria inteligente**. Es un elemento vital para nuestra salud y bienestar, ya sea para el hospital, la odontología o la residencia de ancianos. Este apunta a todos los grupos de edad de la población al mismo tiempo. Para ello se requiere eficacia y menos costes.

2.1.4. Retos

Muchas empresas se apresuran a crear aplicaciones para el IoT y gastan enormes sumas de dinero porque es la próxima gran oportunidad. Se han introducido lavadoras, sistemas de calefacción y frigoríficos inteligentes. El IoT ha traído muchos cambios y dimensiones positivas. Sin embargo, tienen una serie de riesgos y desafíos. [?]- [?]

- **Seguridad y privacidad**. Las aplicaciones del IoT y las áreas inteligentes manejan muchos datos a diario y estos datos se comparten entre dispositivos. La información sobre casas, edificios y coches se comparte entre dispositivos todo el tiempo. Toda esta información exige un mejor sistema de gestión. Los sistemas de seguridad existentes no son adecuados en muchos aspectos y pueden causar graves problemas al usuario. Un error puede hacer que la seguridad del hogar, los coches, los edificios y los datos guardados sean vulnerables.
- **Comunicación fiable**. El IoT se compone de tecnologías fijas y móviles, y se intenta conseguir una comunicación bidireccional sin pérdida de datos y con total fiabilidad.

- **Consumo de energía**. Los dispositivos deben ser capaces de comunicarse con un menor uso de la batería porque estos dispositivos se despliegan lejos y necesitan funcionar con baterías.
- **Interoperabilidad**. Este requiere que los dispositivos se comuniquen entre sí. A nivel de usuario, el usuario puede tener múltiples dispositivos que funcionen en múltiples plataformas. El usuario no querrá quedarse con un solo dispositivo. En cambio, un usuario puede querer diversidad en caso de elegir un dispositivo.

2.2. Middleware

Generalmente, un middleware abstrae las complejidades del sistema o del hardware, permitiendo al desarrollador de la aplicación centrar todo su esfuerzo en la tarea a resolver, sin la distracción de preocupaciones a nivel del sistema o del hardware. Dichas complejidades pueden estar relacionadas con problemas de comunicación o de computación más generales. Un middleware proporciona una capa de software entre las aplicaciones, el sistema operativo y las capas de comunicación de la red, que facilita y coordina que facilita y coordina algún aspecto del procesamiento cooperativo. Desde el punto de vista perspectiva informática, un middleware proporciona una capa entre el software de aplicación y el software de sistema.

En el IoT, es probable que haya una considerable heterogeneidad tanto en las tecnologías de comunicación en uso, como en las tecnologías a nivel de sistema, y un middleware debe soportar ambas perspectivas según sea necesario. Sobre la base de las características descritas anteriormente de la infraestructura del IoT y de las aplicaciones que dependen de ella, se ha establecido un conjunto de requisitos para que un middleware soporte el IoT. A continuación, estos requisitos se agrupan en dos conjuntos: los servicios que debe proporcionar dicho middleware y la arquitectura del sistema [?]- [?].

2.2.1. Requisitos del servicio de middleware

Los requisitos de los servicios de middleware para el IoT pueden clasificarse en funcionales y no funcionales. Los requisitos funcionales recogen los servicios o funciones, como por ejemplo abstracciones y gestión de recursos un middleware, y los requisitos no funcionales, por ejemplo fiabilidad, seguridad y disponibilidad, captan el soporte de la calidad de servicio o del rendimiento.

En los requisitos funcionales nos encontramos:

- **Descubrimiento de recursos**. Ya que la infraestructura y el entorno del IoT son dinámicos, las suposiciones relacionadas con el conocimiento global y determinista de la disponibilidad de estos recursos es inviable. Pasa lo

mismo con la intervención humana, es inviable que este descubra recursos. Por tanto, es importante destacar que el descubrimiento de recursos este automatizado. Cuando no hay infraestructura, el propio dispositivo debe de anunciar su presencia y los recursos que ofrece.

- **Gestión de recursos**. Se espera una QoS aceptable para todas las aplicaciones, y en un entorno en el que los recursos que influyen en la QoS son limitados, como el IoT, es importante que las aplicaciones cuenten con un servicio que gestione esos recursos.
- **Gestión de datos**. Los datos son fundamentales en las aplicaciones de IoT. En el IoT, los datos se refieren principalmente a los datos detectados o a cualquier información de infraestructura de red de interés para las aplicaciones.
- **Gestión de eventos**. En las aplicaciones de IoT se genera potencialmente un número masivo de eventos, que deberían gestionarse como parte integral de un middleware de IoT.
- **Gestión del código**. El despliegue de código en un entorno de IoT es un reto, y debe ser soportado directamente por el middleware. En particular, se necesitan servicios de asignación y de código.

En los requisitos no funcionales nos encontramos:

- **Escalabilidad**. Un middleware de IoT debe ser escalable para adaptarse al crecimiento de la red y las aplicaciones/servicios de IoT.
- **Actuación en tiempo real**. Un middleware debe proporcionar servicios en tiempo real cuando la corrección de una operación que soporta depende no sólo de su corrección lógica sino sino también del tiempo en que se realiza.
- **Fiabilidad**. Un middleware debe permanecer operativo durante la duración de una misión, incluso en presencia de fallos.
- **Disponibilidad**. Un middleware que soporte las aplicaciones de un IoT, especialmente las de misión crítica, debe estar disponible en todo momento.
- **Seguridad y privacidad**. La seguridad es fundamental para el funcionamiento de IoT. En el middleware de IoT, la seguridad debe tenerse en cuenta en todos los bloques funcionales y no funcionales, incluyendo la aplicación a nivel de usuario.
- **Facilidad de despliegue**. Hay que evitar los procedimientos complicados de instalación y configuración.

- **Popularidad**. Un middleware de IoT, como cualquier otra solución de software, debe recibir un apoyo y una ampliación continua.

2.2.2. Requisitos de arquitectura en el middleware

En esta sección se muestran los requisitos para las abstracciones de programación y otros aspectos relacionados con la implementación.

- **Abstracción de la programación**. Para el desarrollador de aplicaciones o servicios, las interfaces de programación de alto nivel deben aislar el desarrollo de las aplicaciones o los servicios de las operaciones proporcionadas por las infraestructuras subyacentes y heterogéneas del IoT.
- **Interoperabilidad**. Un middleware debe funcionar con dispositivos, tecnologías y aplicaciones heterogéneas, sin esfuerzo adicional por parte del desarrollador de aplicaciones o servicios.
- **Basado en el servicio**. Una arquitectura de middleware debe estar basada en servicios para ofrecer una gran flexibilidad cuando sea necesario añadir funciones nuevas y avanzadas al middleware de un IoT.
- **Adaptable**. Un middleware debe ser adaptativo para poder evolucionar y adaptarse a los cambios de su entorno.
- **Consciente del contexto**. El conocimiento del contexto es un requisito clave en la construcción de sistemas adaptativos y también en el establecimiento de valor de los datos recogidos.
- **Autónomo**. Los dispositivos, tecnologías y aplicaciones son participantes activos en los procesos del IoT y deben estar habilitados para interactuar y comunicarse entre sí sin la intervención intervención humana.
- **Distribuido**. Las aplicaciones de un sistema IoT a gran escala de un sistema IoT intercambian información y colaboran entre sí.

2.3. Seguridad

2.3.1. Aspectos legales y éticos

2.3.2. Privacidad

2.3.3. Auditoría de seguridad

2.3.4. Seguridad de datos

CAPÍTULO 3

Planificación

3.1. Planificación a priori

Se trata de planificar como se espera desarrollar el proyecto en el tiempo, para ello se va a hacer uso de un diagrama de Gantt donde se va a proporcionar una vista general de las tareas programadas, estas tareas tendrán que completarse en unas fechas estipuladas.

3.1.1. Diagrama de Gantt

Un diagrama de Gantt es una herramienta útil para planificar proyectos. Proporciona una vista general de las tareas programadas, indicando el periodo de tiempo que tienen para completarse.

El diagrama se mostrará:

- Fecha de inicio y finalización del proyecto.
- Las tareas del proyecto.
- Fecha de programada de cada tarea, tanto la de inicio como la de final.
- Como se superponen las tareas y si hay relación entre ellas.

Con esta planificación se conseguirá una mayor claridad en las tareas a realizar y una mejor gestión del tiempo.

3.1.2. Etapas de desarrollo

- **1ª etapa**: Revisar entornos de desarrollo para IoT.

- **2ª etapa**: Desarrollar aplicación para IoT.
- **3ª etapa**: Explotación de una vulnerabilidad.
- **4ª etapa**: Documentación.

3.1.3. Temporización

De los 3 meses y medios a que se van a dedicar al proyecto, en todos ellos se va a desarrollar las etapas indicadas anteriormente. Se muestra la fecha de inicio y de fin de cada etapa.

Etapas del desarrollo	Fecha de comienzo	Fecha de finalización
Documentación del proyecto	9 de Marzo	23 de Junio
Revisar entornos de desaroollo IoT	21 de Marzo	6 de Abril
Desarrollar aplicación para IoT	7 de Abril	12 de Mayo
Explotar vulnerabilidad	13 de Mayo	22 de Junio

Tabla 3.1: Tabla con la organización temporal del proyecto.

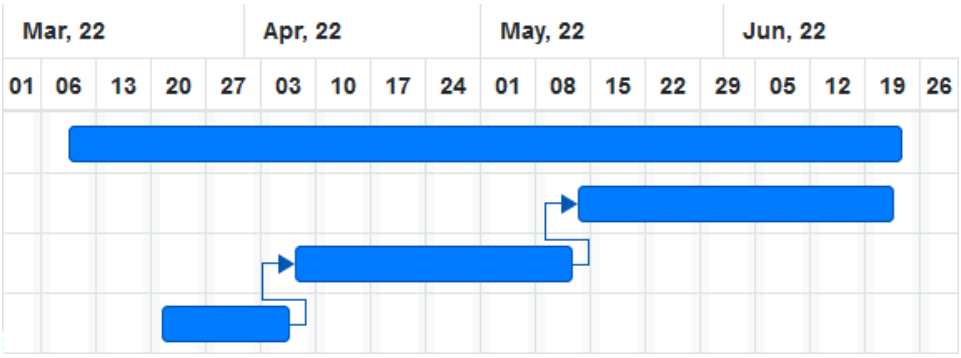

Figura 3.1: Diagrama de Gantt. Gráfico.

Este gráfico corresponde con la duración de cada etapa, comenzando con la documentación. También se indica que hay etapas que para que estas se puedan empezar a desarrollar necesitan que la anterior este terminada, es el caso de la etapa de *Desarrollo de la aplicación* y de *Explotación de una vulnerabilidad*.

3.1.4. Seguimiento del desarrollo

Con el fin de facilitar la visualización del progreso del proyecto se usan distintas herramientas.

3.1.4.1. Trello

Permite gestionar proyectos de forma sencilla y muy visual. Por cada proyecto se crea un tablero donde podemos crear listas y cada lista almacenará *tarjetas* donde se describirá una tarea u objetivo a cumplir. En nuestro caso el tablero se divide en 7 listas:

- **Dudas**, se dejan ancladas preguntas a los tutores.
- **Objetivos**, para recordar los objetivos que se tienen que completar para el proyecto.
- **Pendientes**, tareas que se tienen que realizar pero no se están desarrollando.
- **En proceso**, tareas que se están trabajando.
- **Pendientes de revisión**, tareas finalizadas que requieren de revisión por parte de los tutores.
- **Terminadas**, tareas que han sido aceptadas en la revisión.
- **Hitos**, se agrupan tareas que forman parte de una misma etapa.

3.1.4.2. Github

Se ha creado un repositorio [1] donde se van añadiendo *issues* por cada tarea que se tenga que realizar. Esta tarea es la misma que nos encontramos en el tablero de Trello, por tanto, cuando se crea una nueva tarea en Trello, creamos un nuevo issue con el mismo nombre y descripción para seguir un progreso coherente.

También se crean *Milestones* que se corresponde con el nombre y descripción de la lista de *Hitos* del tablero de Trello.

Cada vez que terminemos de trabajar en un *Milestone* se creará un *Pull Request* para que los tutores puedan revisar los cambios del proyecto y aceptarlos o rechazarlos. Cada acción que se realice tanto en Trello como en Github se verá reflejado en ambos, de esta manera conseguimos un desarrollo del proyecto realista de principio a final.

3.1.4.3. Clockify

Esta herramienta nos permite seguir el tiempo que estamos trabajando. Cada vez que nos pongamos a trabajar, iniciaremos el contador y nombraremos a ese contador con el nombre de la tarea que estemos realizando en ese momento.

[1] Enlace al repositorio: https://github.com/LuisArostegui/TFG

CAPÍTULO 4

Análisis del problema

CAPÍTULO 5

Diseño

CAPÍTULO 6

Implementación

CAPÍTULO 7

Pruebas

CAPÍTULO 8

Conclusiones y trabajos futuros