Universidad de Valladolid

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería de Organización Industrial

La tecnología RFID y el Hardware de bajo coste para el control de procesos: aplicación para el control de la jornada laboral.

Autor:

Velasco Almendro, Marco

Tutor:

De Benito Martín, Juan José
Pascual Ruano, José Antonio

Departamento de Organización de Empresas y Comercialización e Investigación de Mercados

Valladolid, julio de 2018.

AGRADECIMIENTOS

En primer lugar, agradecer a mis tutores, Juanjo y José, por aceptar la realización de este trabajo y por la ayuda y el interés mostrados a lo largo de todo el desarrollo del mismo.

Agradecer también a mi familia, por estar siempre ahí y por el apoyo recibido en todo momento.

RESUMEN:

El presente proyecto tiene como objetivo la aplicación de la tecnología de identificación por radiofrecuencia, RFID, para el control de procesos industriales, concretamente se empleará para el registro de la jornada laboral. Para ello, el desarrollo del proyecto requerirá la utilización de hardware low-cost, en concreto Arduino, y la programación mediante software de carácter libre, siendo el lenguaje Python el elegido.

Mediante el uso y la investigación de estas tecnologías se pretende también la familiarización con el concepto de Industria 4.0 así como dejar en claro la necesidad de adaptación de las fábricas actuales a esta nueva era de automatización industrial.

PALABRAS CLAVE:

Arduino, Industria 4.0, Python, RFID, Seguridad.

ABSTRACT:

The main objective of this project is to apply the RFID technology to process control. In this case, the process that needs to be controlled is the record of the workday of the employees. In order to get that, the project will require the use of low-cost hardware (Arduino) and free software (Python lenguage).

Another goal to be achieved during the development of this project is to become familiar with the concept of Industry 4.0 and with the need for adaptation to the new models of automated industries that are rising nowadays.

KEYWORDS:

Arduino, Industry 4.0, Python, RFID, Security.

ÍNDICE DE IMÁGENES

ÍNDICE DE TABLAS

ÍNDICE DE GRÁFICAS

INTRODUCCIÓN

En la actualidad, las nuevas tecnologías están experimentando un crecimiento sin precedentes, siendo ya parte indispensable de nuestro día a día, estando presentes en todos los campos de nuestra vida, seamos conscientes o no de ello, lo que hace indispensable la adaptación a los nuevos tiempos.

Esta evolución tecnológica y la necesidad de adaptación existirán también, por supuesto, en la Industria.

Nos encontramos en plena revolución industrial, la conocida como Industria 4.0, en la que las fábricas han de adaptarse a las nuevas tecnologías y evolucionar hacia una industria más automatizada e inteligente, que posea y maneje mayores cantidades de información, y con altos niveles de comunicación e interactuación entre todos sus componentes, lo que desembocará en la optimización de productos y procesos, así como en una mejor experiencia para el cliente.

Dentro de esta evolución tecnológica, está cobrando cada vez más importancia la conocida como cultura “Do it yourself”, o “hágalo usted mismo”, donde cada vez un mayor número de usuarios buscan crear sus propia soluciones a los problemas que puedan tener.

Esta cultura DIY está favorecida, en parte, por la existencia de software y lenguajes de programación de carácter libre, cada vez más importantes, como lo son Arduino y Python, lenguajes empleados en el desarrollo de este proyecto.

Otra tecnología que está creciendo de manera exponencial en la actualidad es la tecnología de identificación por radiofrecuencia, o RFID. En este proyecto se pretende diseñar e implementar un dispositivo que integre la aplicación de la tecnología RFID con la cultura DIY, mediante prototipado y programación en hardware de bajo coste y software de carácter libre, concretamente en Arduino y Python.

OBJETIVOS Y COMPETENCIAS.

Como se explicó en la introducción, el objetivo principal de este proyecto es el desarrollo de un dispositivo basado en Arduino y Python que aplique la tecnología RFID en procesos industriales.

Se pretende que a lo largo de la generación de este proyecto, se adquieran conocimientos y habilidades generales de programación y conocimientos específicos de los lenguajes Arduino y Python, además de adquirir un cierto grado de familiarización con el amplio abanico de

posibilidades que ofrecen los hardware low-cost como Arduino y su catálogo de módulos, que permiten el montaje de un sinfín de proyectos.

Se desea también, además, la profundización en los conceptos que refieren a la tecnología de identificación por radiofrecuencia, RFID.

Si bien las posibilidades de esta tecnología son numerosas (la tecnología RFID se puede aplicar a casi cualquier proceso, por ejemplo, se podría llevar a cabo un control de productos a lo largo de la cadena de fabricación), en este preciso caso se pretende desarrollar un dispositivo de control de acceso que, además de otorgar o denegar los permisos pertinentes, sea capaz de llevar a cabo un registro de la jornada laboral.

Con la aplicación de este dispositivo se pretende lograr una mejora tanto en materia de seguridad como en obtención de información, la cual puede ser aprovechada a tiempo real o a posteriori para ser estudiada y localizar debilidades con el fin de buscar oportunidades e intentar aprovecharlas.

Además, en el caso concreto de la jornada laboral, la obtención de un registro no solo proporcionará información que puede ser analizada, sino que también permite cumplir la legislación vigente en materia de registro de las horas trabajadas, registro que es obligatorio para ciertos tipos de trabajadores, y estar en posición de adaptarse a posibles cambios venideros en la normativa.

Este TFG también pretende contribuir al desarrollo de las competencias generales de la asignatura dentro del título de Ingeniería en Organización Industrial.

Se pretende, finalmente, desarrollar todas las competencias específicas del título, así como integrar los conocimientos y capacidades adquiridos a lo largo de la titulación y adquirir madurez.

ORGANIZACIÓN DE LA MEMORIA

El presente documento sigue una línea argumental muy similar a la que siguió el desarrollo del proyecto a efectos prácticos.

La memoria está organizada en cinco capítulos principales, un apartado introductorio, un apartado de conclusiones, un apartado de bibliografía y unos anexos.

En este presente apartado introductorio se sitúa el proyecto, se describen su motivación y sus objetivos, y se detalla la organización de la memoria.

En el primer capítulo se profundiza en el marco de trabajo del proyecto y en el estado del arte, situando al proyecto en un contexto, tanto técnico como temporal.

En el segundo capítulo de procede a la presentación del concepto hardware low-cost y los principales tipos que existen en la actualidad, aclarando, además, el porqué de la elección de Arduino. A continuación se detallan todos los componentes de hardware empleados en el prototipado del proyecto, sus diferentes conexiones y finalmente se muestra el aspecto físico final del dispositivo.

El tercer capítulo corresponde al primer apartado de software que se emplea en el desarrollo de este trabajo. Se presentan el IDE de programación de Arduino y el código empleado en el proyecto perteneciente a este lenguaje de programación, enumerando cada función empleada y su rol en el proyecto.

El cuarto capítulo corresponde al segundo apartado de software empleado, en este caso, el lenguaje de programación Python. Se dan ciertas nociones básicas sobre este lenguaje, sus características más relevantes y su historia, y se concluye con el porqué de su elección tras compararlo con otros lenguajes de programación importantes con los que comparte ciertos rasgos.

Se presenta también el entorno empleado para la programación en Python (Eclipse), y la función del código escrito en Python, reflejando finalmente los resultados que arrojaría una prueba de funcionamiento de todo el proyecto.

En el quinto y último capítulo se lleva a cabo un estudio económico sobre el desarrollo de todo el proyecto a lo largo de todas sus etapas, incluyendo su duración y sus costes.

Seguidamente se sitúa el apartado de conclusiones y líneas de mejora, que incluye las conclusiones extraídas a lo largo de todo este trabajo, así como los objetivos cumplidos y los que quedan por cumplir, y la propuesta de ciertas líneas de trabajo futuras que se han de seguir con el fin de optimizar el funcionamiento del dispositivo desarrollado.

CAPÍTULO 1:

MARCO DE TRABAJO Y ESTADO DEL ARTE

Este primer capítulo tiene como objeto poner en situación el proyecto desarrollado, pretendiendo centrar tanto su campo de aplicación (marco de trabajo) como señalar algunos antecedentes históricos, así como explicar el contexto temporal en el que se encuentra (estado del arte).

1.1. MARCO DE TRABAJO

En primer lugar, se ubica el proyecto dentro del espectro científico, siendo la automática y la electrónica los principales campos que abarcará.

Se profundiza, además, en la tecnología principal en la que se basa este trabajo, que es la tecnología RFID.

1.1.1. AUTOMÁTICA

Según la *Gran Enciclopedia Larousse*, el término "ingeniería" se puede definir como "conjunto de conocimientos y de técnicas que permiten aplicar el saber científico a la utilización de la materia y de las fuentes de energía, mediante invenciones o construcciones útiles para el hombre".

Herber Simon añadía, además, que la ingeniería "no se ocupa de cómo son las cosas, sino de cómo deberían ser".

Por su parte, Wiener acuñó al término "cibernética" la definición que sigue: "ciencia dedicada al estudio de los métodos de comunicación, control y auto-organización comunes a máquinas y organismos vivos".

Con todo lo anterior en mente, en Cuenca y Salt (2005), aparece la siguiente definición del concepto Automática: "ciencia que estudia a los sistemas cibernéticos en los que ha sido introducido conscientemente algún elemento que realiza funciones de control".

Otra definición acertada es la ofrecida por la Real Academia de las Ciencias (2018), y es la que sigue: "disciplina que trata de métodos y procedimientos cuya finalidad es la sustitución del operador humano por un operador artificial en la ejecución de una tarea física y mental previamente programada".

Esta segunda definición es, tal vez, la que más se adecúa al contenido de este proyecto, ya que su objeto es recabar y transmitir información de una manera automática, sin necesidad de intervención humana directa más allá de registrar su dispositivo de identificación.

Se entiende por "automatización" el proceso de implantación de la Automática en la Industria, y por "autómata", instrumento, aparato o

estructura que encierra dentro de sí los mecanismos necesarios para imprimir ciertas acciones programadas.

Ésta es, precisamente, la finalidad del proyecto. Automatizar tareas e intercambio de información mediante un dispositivo que bien podría ser considerado un autómata industrial.

1.1.2. ELECTRÓNICA:

Según la RAE, se entiende por electrónica "el estudio y aplicación del comportamiento de los electrones (u otras partículas cargadas) en diversos medios, como el vacío, los gases y los semiconductores, sometidos a la acción de campos eléctricos y magnéticos".

Los sistemas electrónicos trabajan a partir de estos conceptos, obteniendo señales, normalmente del exterior mediante dispositivos denominados sensores, transformándolas posteriormente en señales de tipo eléctrico para ser procesadas por los distintos circuitos o componentes del sistema, que interaccionan entre ellos, para acabar transformando de nuevo estas señales eléctricas en señales físicamente interpretables, vía actuadores u otros dispositivos similares.

El diagrama de bloques simplificado (en realidad entrarían otros elementos como perturbaciones externas o fuentes de alimentación) de un sistema electrónico se muestra en la Imagen 1.1:

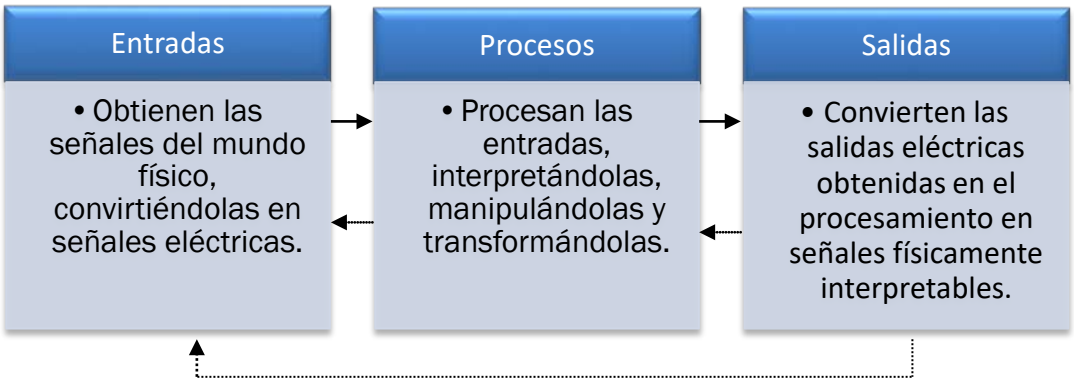


Imagen 1.1 *Diagrama de bloques simplificado de un sistema electrónico*

Fuente: elaboración propia.

1.1.3. TECNOLOGÍA RFID:

En el libro Technology Integration to Business, de John T. Yee y Seog-Chan Oh, se puede encontrar una breve introducción a la tecnología utilizada en este proyecto, la tecnología de identificación por radiofrecuencia o RFID.

La tecnología de identificación por radiofrecuencia, o, de aquí en adelante, RFID (Radio Frequency Identification), es una radiotecnología de corto alcance y baja potencia, utilizada para obtener datos en tiempo real. Esta tecnología puede considerarse un sustituto más potente de los códigos de barras, ya que los identificadores RFID pueden ser leídos a mayor distancia y sin necesidad de contacto directo.

La tecnología RFID emplea los denominados "tags" de radiofrecuencia, dispositivos transpondedores, normalmente provistos de baterías de larga duración, que emiten señales de radio que reciben los transceptores o lectores, que pueden ser tanto de sólo lectura como de lectura y escritura. Tanto los tags como los lectores están provistos de una antena.

Existen dos tipos de tags, pasivos y activos: los tags pasivos reciben radioenergía emitida por los lectores como interrogativa, y emplean la energía recogida para enviar una señal de vuelta al lector identificándose. Por su parte, los tags activos contienen su propia fuente de energía.

La comunicación por radiofrecuencia puede darse en distintas bandas frecuenciales. Las utilizadas en la actualidad son las mostradas en la Tabla 1.1:

Banda de frecuencias	Denominación	Rango habitual
125 kHz	Baja Frecuencia (LF)	< 50 cm
13,56 MHz	Alta Frecuencia (HF)	≈8 cm
400-1000 MHz	Ultra Alta Frecuencia (UHF)	3-10 m
2,45-5,4 GHz	Microondas	> 10 m

Tabla 1.1: *Frecuencias funcionamiento tecnología RFID,*

Fuente: Blog de By (2010).

Mientras que la banda de frecuencia de 125 kHz era utilizada en los albores de esta tecnología, debido sobre todo a su simplicidad, aunque ofrecían bajos niveles de seguridad, en la actualidad la frecuencia utilizada es la de 13,56 Mhz, la cual ofrece una mayor seguridad y una gran capacidad para almacenar información en las tarjetas o dispositivos correspondientes.

La tecnología RFID se está expandiendo rápidamente en la industria en los últimos años, en gran parte debido a sus grandes reducciones en los precios, y se prevé que este auge continúe en los años venideros.

En la tabla 1.2 se muestra el número de publicaciones académicas relacionadas con la tecnología RFID en los últimos años y la gráfica 1.1 marca la evolución correspondiente, pudiéndose observar con claridad el auge anteriormente mencionado.

Año	Nº de publicaciones	Ratio (%)
2000	10	0,4
2001	13	0,5
2002	16	0,7
2003	41	1,7
2004	84	3,5
2005	192	7,9
2006	335	13,8
2007	328	13,5
2008	364	15
2009	498	20.5
2010	545	22,5
Total	2426	100

Tabla 1.2: *Publicaciones sobre RFID anuales*

Fuente: Yee & Oh, 2012.

Gráfica 1.1: *Total de publicaciones sobre RFID acumuladas por año*

Fuente: Yee & Oh, 2012.

De todas estas publicaciones, el 35% se encontraban dentro del campo de la ingeniería, y aproximadamente otro 30% dentro de las telecomunicaciones y la informática.

PROS Y CONTRAS DE LA TECNOLOGÍA RFID

Como en cualquier aspecto de la vida, existen diferentes pros y contra respecto a la aplicación de la tecnología RFID. Enunciaremos algunos de ellos (Techspirited – Pros and cons of RFID Technology (2018)).

Pros

- Los sistemas RFID son robustos y funcionan correctamente incluso en condiciones adversas.
- No se requiere contacto físico entre tag y lector, como ya se aclaró con anterioridad.
- Se pueden leer varios tags al mismo tiempo.
- El sistema RFID es fiable y seguro.
- Poseen una gran capacidad para almacenar información.

Contras

- Alto coste (aunque cada vez esta tecnología es más barata y sus sistemas de menor tamaño).
- Algunas señales se envían de manera errónea en presencia de ciertos metales o líquidos.
- El rango de alcance puede ser insuficiente en algunos casos.
- Existen ciertas amenazas en lo respectivo a su seguridad, reflejadas en el apartado siguiente.

PROBLEMAS DE SEGURIDAD DE LA TECNOLOGÍA RFID.

En el año 2010, el Instituto Nacional de Tecnologías de la Comunicación (INTECO) publicó una guía sobre la seguridad y la privacidad en la tecnología RFID, la cual incluía riesgos del uso de esta tecnología, divididos en riesgos para la seguridad y riesgos para la privacidad, así como un manual de recomendaciones y buenas prácticas.

- **Riesgos para la seguridad:** según esta guía, son "aquellos riesgos derivados de acciones encaminadas a deteriorar, interrumpir o aprovecharse del servicio de forma maliciosa", buscando bien el beneficio económico o deteriorar el servicio prestado.

La amenaza más básica sobre un sistema RFID es evitar directamente la comunicación entre el lector y la etiqueta, aunque existen muchos otros tipos de ataques. Estos son solo algunos de ellos:

- Aislamiento de etiquetas: es el ataque más sencillo. Consiste en impedir la comunicación entre el lector y la etiqueta.
- Suplantación: consiste en el envío de información falsa que pretende hacerse pasar por información válida
- Repetición: consiste en la captación de la señal original válida de un tag y posteriormente reproducirla, tratando de suplantar la identidad de la etiqueta original.
- Denegación de servicio (DoS): se basa en saturar el sistema enviándole más información de la que puede procesar, invalidando así el funcionamiento del sistema de detección de etiquetas.
- Destrucción de etiquetas: trata de inutilizar el sistema mediante la aplicación de un campo electromagnético que destruye o inhabilita las etiquetas RFID.
- Clonación de la tarjeta RFID: es una de las amenazas más extendidas, y por tanto también una de las más temidas y conocidas. A partir de una comunicación tag-lector, se copian los datos emitidos y se replican en otra tarjeta que suplantará la identidad de la original.

- Riesgos para la privacidad: otro riesgo incipiente de la tecnología RFID es el que deriva de la privacidad de los usuarios, que consiste en el acceso no autorizado a información privada o personal del usuario, información que puede estar almacenada en la etiqueta o bien asociada a ella y almacenada en un sistema al que se puede acceder para sustraerla.

  Las amenazas más importantes son los accesos no permitidos a las etiquetas, que pueden contener datos personales de cualquier tipo, el rastreo de las personas y sus comportamientos, gustos, etc., y el uso de dichos datos para crear perfiles sobre los comportamientos individuales de cada usuario.

Si bien es cierto que existen numerosas amenazas en lo respectivo a esta tecnología y que han existido casos, por ejemplo, de suplantación de pasaportes o sustracción de dinero a distancia mediante RFID, también lo es que esta guía fue publicada en el año 2010 y que la seguridad de esta tecnología es mayor ahora de lo que era entonces, y continúa aumentando cada año, siendo cada vez más una tecnología más fiable.

Por otra parte, esto no debería ser razón para evitar que la tecnología RFID siga evolucionando en los años venideros, evolución en la cual el aspecto de la seguridad deberá jugar un rol importante de cara a que existan ciertas garantías en un futuro no muy lejano.

APLICACIONES DE LA TECNOLOGÍA RFID

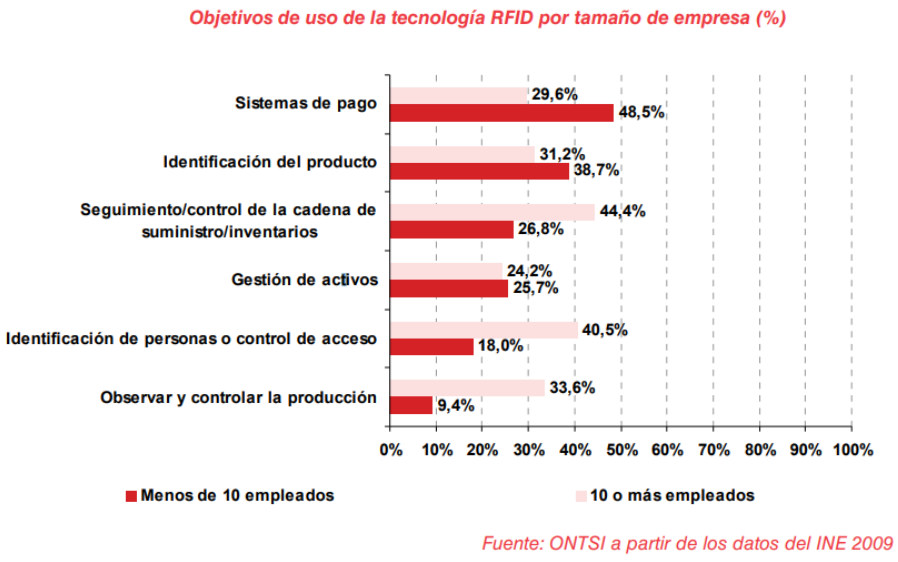
Las aplicaciones de la tecnología RFID en nuestros días son múltiples y variadas, y cada vez más. De hecho, en ThingMagic recientemente se propusieron enumerar cien usos de esta tecnología, consiguiendo cerrar finalmente la lista, que se adjunta como anexo a este proyecto.

Si bien dicha lista va a aplicaciones más específicas, se pueden enumerar varias de las más genéricas y relevantes, algunas de las cuales podemos encontrar en Universidad VIU (2017) y en la propia guía publicada por el INTECO mencionada en el apartado anterior:

- Tarjetas de identificación: pueden incluir desde controles de acceso a planta a tarjetas de transporte público o tarjetas con información de carácter médico en centros hospitalarios. En algunos países existen también pasaportes electrónicos, que almacenan la información del titular en un chip RFID.
- Autenticación de documentos.
- Localización y rastreo: puede ir desde lo más evidente, como puede ser un chip portado por una mascota, a su incorporación en centros escolares o, por ejemplo, chips localizados en vehículos.
- Identificación de problemas en la cadena de suministros: la aplicación de esta tecnología dentro del marco de la Logística está entre las más importantes a día de hoy. Otro ejemplo es el inventariado de productos. La industria de la automoción fue la primera en incluir esta tecnología, en los años 80.
- Aplicaciones de uso militar.

- Cuidados médicos: por ejemplo, controles de posición a ciertos enfermos que no puedan deambular por diversas zonas.
- Aplicaciones deportivas: monitorización de pasos por puntos de control, línea de meta, etc.
- Pago sin contacto: a día de hoy, ya son muchas las tarjetas de crédito que incorporan la capacidad de realizar pagos a distancia sin necesidad siquiera de sacar la tarjeta de la cartera, tan solo acercándola al lector correspondiente. También existen, por ejemplo, sistemas de pago automático en peajes o supermercados, y sistemas de pagos electrónicos con teléfonos móviles.
- Controles de acceso.
- Dispositivos antirrobo.
- Control de libros y usuarios en bibliotecas.

En la gráfica 1.2 se muestran las distintas aplicaciones de la tecnología RFID por tamaño de empresa, en tanto por ciento:



Gráfica 1.2: *aplicaciones de la tecnología RFID por tamaño de empresa*

Como se puede observar con estos ejemplos, las aplicaciones de la tecnología RFID son innumerables tanto en nuestra vida cotidiana como específicas de la industria u otros campos, y se puede esperar aún un mayor crecimiento y la aparición de nuevas funciones. Además, si se combina con otras tecnologías como la WLAN sus ventajas y su potencial son aún mayores.

1.2. ESTADO DEL ARTE

A continuación se enmarca este proyecto en un contexto histórico, presentando algunos de los antecedentes más relevantes pero haciendo hincapié en la situación contemporánea.

1.2.1. CUARTA REVOLUCIÓN INDUSTRIAL

En la actualidad, son cada vez más las voces que reconocen que estamos sumidos en la denominada Cuarta Revolución Industrial, o Industria 4.0, término acuñado en la Feria de Hannover de 2011. Pero, pongamos esta revolución en contexto histórico, entendiendo por "revolución" un cambio radical, y recordando de manera superficial las tres revoluciones industriales hasta la fecha.

- Primera revolución industrial: se puede ubicar en el siglo XXVIII, en torno al año 1786, tras la invención de la máquina de vapor de James Watt. Lo que era una economía fundamentalmente agrícola y artesanal evolucionó hacia una economía industrial y mecanizada. Destacaron, sobre todo, las industrias textil, siderúrgica y de transporte, siendo claves el ferrocarril y el barco a vapor.

- Segunda revolución industrial: se inicia en la segunda mitad del siglo XIX con avances técnicos tales como la electricidad, evolución en el uso del petróleo o la creación del automóvil de combustión y de los primeros aviones. Cabe destacar también que es en dentro de esta revolución cuando se inicia la producción en masa en la industria, gracias en gran medida a Henry Ford.

- Tercera revolución industrial: data del siglo XX y se basa principalmente en la aparición de las nuevas tecnologías de la información y comunicación, incluidas tecnologías que supusieron un cambio radical en la vida cotidiana de las personas (radio, televisión, o más adelante internet, favoreciendo la globalización) o los propios medios de transporte. También destaca la aparición de las energías renovables. Es también conocida a veces como la revolución digital o la revolución informática.

En su libro "La Cuarta Revolución Industrial", Klaus Schwab (2005) arroja algo de luz sobre el paradigma de esta nueva revolución.

Schwab considera que esta revolución es algo completamente nuevo, de dimensiones nunca antes vistas, y que está cambiando por completo a la humanidad tanto en su forma de vivir, como de trabajar o de relacionarse los unos con los otros. Esto se aplica también, por supuesto, a la industria, donde están apareciendo nuevos modelos de negocio y remodelándose todos los sistemas de producción, consumo, transporte, y entrega.

Schwab destaca tres razones por las que considera que la cuarta revolución está ya en marcha y se distingue de todas las anteriores. Estas son la velocidad - en este caso la evolución es exponencial, mientras que en las anteriores era lineal-, la amplitud y profundidad –las nuevas tecnologías no sólo cambian el "qué" y el "cómo", sino el "quiénes somos"-, y el impacto de los sistemas.

El autor considera también que la tecnología no es una fuerza exógena, sino que tecnología y sociedad coexisten, y que esta nueva revolución está basada en la revolución digital, caracterizada por un internet móvil, por dispositivos como por ejemplo sensores cada vez más potentes y de menor coste, y por la inteligencia artificial (IA) y el aprendizaje de las máquinas. De hecho, Erik Brynjolfsson y Andrew McAfee (MIT) denominaron este período como "la segunda era de las máquinas", en el libro que publicaron en el año 2014.

Para Klaus, uno de los pilares que diferencia a esta revolución de todo lo conocido anteriormente es la alta interacción entre las diversas tecnologías y la rápida difusión de las tecnologías e innovaciones emergentes, y cree además que será un suceso histórico clave, pero a su vez su dimensión será fuertemente dependiente de cómo sea recibida y aceptada por la sociedad.

En el año 2015, varios autores publicaron para el Boston Consulting Group (BCG) un artículo titulado "Industria 4.0: el futuro de la productividad y el crecimiento en las industrias manufactureras" (en inglés original, "Industry 4.0: The Future of Productivity and Growth in Manufacturing Industries"), en el que profundizaban sobre esta cuarta revolución (Gerbert-Harnisch et al., 2015).

En dicho artículo los autores señalaban que tanto sensores como maquinaria, objetos de trabajo y sistemas informáticos estarán conectados a lo largo de toda la cadena de valor. Estos sistemas conectados, además, podrán interactuar unos con otros gracias a protocolos basados en internet, analizar datos para prevenir errores, autoconfigurarse y adaptarse a cambios. Todo esto dará lugar, finalmente, a procesos más flexibles y eficientes en los que se obtendrán mayores calidades a menor costo, transformando la competitividad de las industrias actuales.

En esta publicación se basa la Industra 4.0 en nueve pilares fundamentales (según qué publicación se pueden encontrar otros pilares diferentes, pero la mayoría siempre son comunes o uniones o desgloses unos de otros, por lo que me ha parecido conveniente destacar estos en concreto, ya que parecen ser los más relevantes), que se muestran a continuación en la Imagen 1.2.

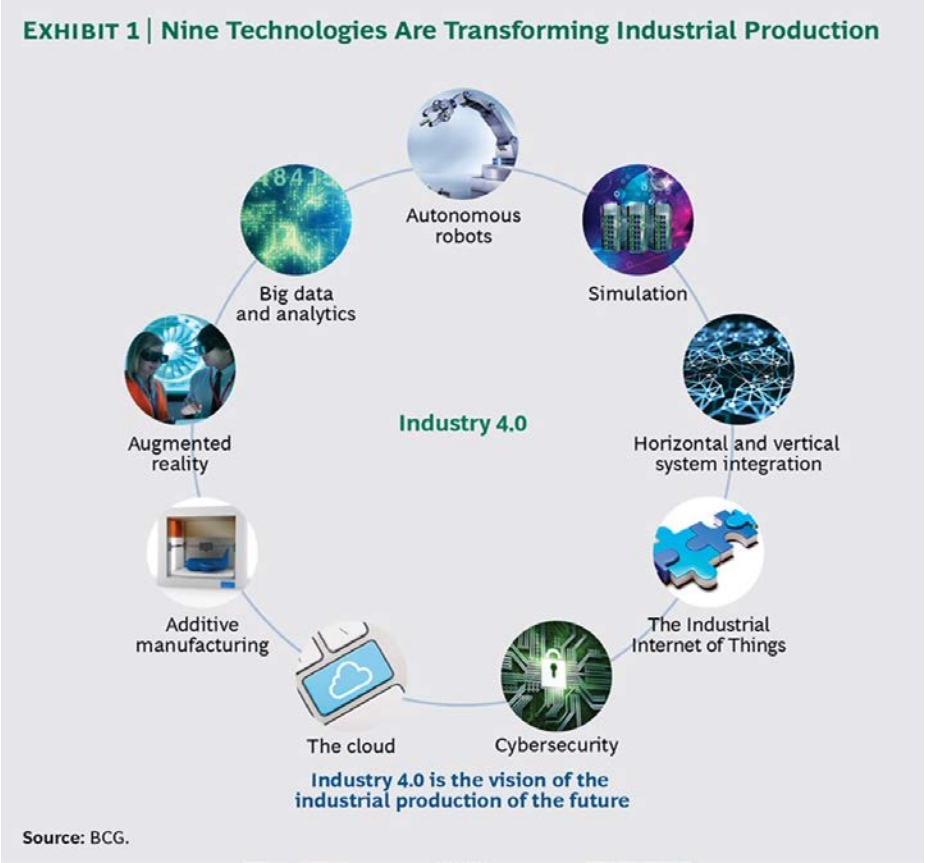


Imagen 1.2: *Nueve pilares de la Industria 4.0*

Fuente: Gerbert-Harnisch et al., 2015.

Estos pilares son: robos autónomos, simulación, intregración vertical-horizontal, internet de las cosas (IoT), ciberseguridad, la nube, fabricación aditiva, realidad aumentada, y análisis Big Data, y, mientras que algunos ya se utilizan a día de hoy, la verdadera revolución llegará cuando la integración de todos ellos sea completa.

El proyecto que nos ocupa se relaciona especialmente con cuatro de estos nueve pilares, concretamente su relación será más estrecha con la integración horizontal-vertical, el análisis Big Data, la ciberseguridad y el internet de las cosas (IoT).

- Integración horizontal-vertical: en la actualidad, la mayoría de sistemas informáticos no se encuentran completamente

integrados. No existe una integración total entre empresa-cliente y ni siquiera entre departamentos de una misma empresa, o funciones de un mismo departamento. La Industria 4.0 buscará la cohesión de todos los participantes, asegurando una gestión integral, fomentando la creación de cadenas de valor realmente automatizadas.

- Análisis Big Data: esta técnica, consistente en el análisis de grandes cantidades de datos (muchos provenientes del propio internet de las cosas), es relativamente reciente. Su correcto manejo ofrece muchas ventajas, tales como la optimización de la producción, el ahorro de energía o la mejora del equipamiento, así como la representación de predicciones de futuras tendencias obtenidas a través de ciertos patrones de predicción.

  En la Industria 4.0 habrá de jugar un papel como herramienta para la toma de decisiones en tiempo real, basándose en información proveniente de diferentes fuentes, como los propios sistemas de producción, clientes, etc., lo cual ofrecerá posibilidades aún mayores de optimización de procesos y calidad.

- Ciberseguridad: la Industria 4.0 requiere una gran conectividad así como un alto nivel de estandarización, lo que incluye también protocolos de comunicación estándar. Esto redunda en una necesidad de protección de las fábricas frente a las amenazas de ciberseguridad, por lo que comunicaciones seguras y fiables, así como sistemas de identificación y control de acceso de alta sofisticación serán indispensables.

- El internet de las cosas, IoT: como su propio nombre indica, este concepto, nacido en el MIT (Massachusetts Institute of Technology) hace referencia a la conectividad con internet de un número cada vez mayor de elementos (por ejemplo, objetos físicos, vehículos o máquinas), tanto dentro de la industria como en la vida cotidiana, y por tanto la conexión entre ellos de igual forma mediante protocolos de internet.

  Aplicando esta conexión mediante protocolos estándar a distintas tecnologías dentro de la Industria, los diferentes dispositivos podrán comunicarse entre ellos e interactuar, favoreciendo lo que se conoce como “fábrica inteligente”,

favoreciendo la descentralización del análisis y la toma de decisiones en tiempo real.

Los dispositivos IoT generarán una gran cantidad de datos, que deberán ser analizados y aprovechados mediante técnicas Big Data, preferentemente en tiempo real.

Es aquí donde la tecnología RFID puede jugar un papel muy importante, ya que, además de sus posibles aplicaciones en materia de seguridad, con la inclusión de un chip RFID de tamaño reducido se pueden almacenar y transmitir grandes cantidades de información de manera rápida y sencilla, que podrán ser posteriormente aprovechadas por las nuevas técnicas de análisis Big Data.

En el sitio web de SAP, empresa especializada en software para empresas (Soluciones de software de gestión empresarial para pymes | SAP - ¿Qué es el internet de las cosas? (2016)) reflejan algunas de las ventajas de la aplicación del internet de las cosas, entre las que destacan nuevos modelos de negocio, la eficiencia operativa, especialmente optimizando procesos de fabricación y cadenas de suministro, la productividad de la fuerza laboral, ayudando a los trabajadores a tomar mejores decisiones y automatizar tareas rutinarias, y una mejor experiencia de cliente, con servicios más personalizados.

APLICACIONES DE LA TECNOLOGÍA RFID EN LA INDUSTRIA 4.0

La tecnología de identificación por radiofrecuencia es una tecnología al alza, cuyo uso en la industria es cada vez mayor, ya que tiene numerosas y diversas aplicaciones dentro del entorno de la cuarta revolución industrial, y supone una herramienta destinada a recabar información útil del proceso productivo así como de adaptación a las nuevas formas de comercialización.

En El papel de la tecnología RFID en la fábrica inteligente (2010) encontramos algunas ideas sobre el papel que desempeña la tecnología RFID en la Industria 4.0:

El autor considera que los datos almacenados en las etiquetas RFID (que pueden almacenar kilobytes de información transmitidos en cuestión de milisegundos) se pueden utilizar de forma cada vez más sofisticada y eficaz, con el fin último de lograr mayor eficiencia y flexibilidad y menor coste en el proceso de fabricación, ayudando a automatizarlo y estandarizarlo.

Además, en la actualidad, la información que almacena la etiqueta puede utilizarse y estudiarse mediante técnicas Big Data, ya que dicha etiqueta recabará datos desde el inicio hasta el final de su ciclo de vida.

Un aspecto clave de la tecnología RFID en la fábrica inteligente es que permite el seguimiento del producto de forma inalámbrica y además durante toda la cadena de suministro.

Por tanto, estas serán algunas de las posibles aplicaciones de la tecnología RFID en la industria actual:

- Fabricación y venta: El uso de esta tecnología favorece la interacción entre los componentes, productos, maquinaria y empleados, posibilitando mayores oportunidades de individualización a lo largo de la cadena, ya que máquina o producto pueden ir almacenando información en cada etapa de ésta y ser accedida en tiempo real.

  Esta tecnología puede servir también como prueba de autenticidad, calidad o seguridad, tanto a lo largo del proceso productivo, como a la hora de que el producto sea comprobado por los potenciales clientes.

- Cadena de suministro: el seguimiento mediante RFID permite rastrear y mejorar la eficiencia a lo largo de toda la cadena de suministros, ya que otorga una identificación, trazabilidad y localización completas a lo largo de toda la cadena de suministros.

La utilización de esta tecnología en la industria supone una ventaja competitiva debido especialmente a la información aportada, si bien estas son algunas de las aplicaciones más importantes (la principal es la logística y aplicada a productos), en este proyecto se le da una vuelta de tuerca a la aplicación de esta tecnología con tres objetivos principales: controlar y permitir o restringir el acceso del personal a la vez que se facilita su identificación, recabar información sobre los accesos y horarios de cada empleado en tiempo real y, finalmente, el cumplimiento la normativa vigente, expuesta en el siguiente punto y, en su caso, adaptarse a potenciales modificaciones de ésta.

1.2.2. NORMATIVA ACTUAL

En la actualidad, el Estatuto de los Trabajadores (2015) recoge distintas obligaciones en cuanto al registro de la jornada laboral de los empleados, que son las que siguen:

El Art. 12.4 recoge la obligación del registro de la jornada de los trabajadores a tiempo parcial, la cual ha de totalizarse mensualmente y cuyos registros deben ser conservados por el empresario durante un período mínimo de cuatro años. En caso de incumplimiento de lo anterior, el contrato se presumirá a jornada completa.

El Art. 35.5., por su parte, recoge que la jornada de cada trabajador ha de ser registrada día a día con el fin de computar las horas extraordinarias, si bien la interpretación de dicho artículo ha generado cierta controversia.

El 4 de diciembre de 2015, la Audiencia Nacional emitió una sentencia que consideraba que las empresas debían mantener un registro diario de las horas trabajadas por sus empleados con el fin de establecer la existencia o no de horas extras, frente a la empresa Bankia (Audiencia Nacional (27 de enero, 2016)).

Si bien el Tribunal Supremo, a 23 de marzo de 2017, interpretó el Estatuto de manera contraria, eximiendo a las empresas de dicha obligación y considerando que únicamente deberían registrarse las horas extra, aunque admitiendo también que convendría la existencia de una ley que clarificase el asunto.

En cualquier caso, la falta del registro de la jornada laboral o su incorrecto manejo suponen una falta leve en la Ley sobre Infracciones y Sanciones en el Orden Social (8 de agosto, 2000).

Si bien a día de hoy se está debatiendo la posibilidad de modificar la ley para que el registro de la jornada laboral sea obligatorio, así como su mantenimiento durante mínimo cuatro años, aún no hay nada en claro, pese a que el PSOE propuso una ley para modificar el Estatuto (Munera, 2017) y la intención es que este registro sea obligatorio en un futuro, aún no se puede considerar oficial.

En cualquier caso, con una herramienta para el registro de empleados una empresa se aseguraría evitar conflictos a la par que prepararse ante posibles cambios legislativos venideros, además de ser compatible con el registro de las jornadas de los trabajadores a tiempo parcial así como de las horas extra, lo cual ya es obligatorio en la actualidad.

Si, además, dicha herramienta incluye tecnologías presentes en la Industria 4.0, la ganancia es doble, ya que la empresa se adapta tanto a la normativa como a los tiempos que vienen, ganando en fiabilidad y competitividad y, dicho registro, puede ser fácil y cómodo mediante el uso de la tecnología RFID, ya que por ejemplo el empleado puede llevar una tarjeta de identificación en la cartera y no preocuparse de pasarla por un lector, simplemente la antena detectará cuando el personal entra y sale del lugar, registrándolo sin que el usuario tenga que preocuparse por ello y reduciendo la probabilidad de olvidos y despistes, aumentando la fiabilidad de los datos recogidos.

CAPÍTULO 2:

HARDWARE Y PROTOTIPADO DEL PROYECTO

Han pasado ya ocho años desde que en el año 2010 Chris Anderson acuñase el término DIY ("Do It Yourself", o, en castellano, "Hágalo usted mismo") en su artículo "In the next industrial revolution atoms are the new bits" ("En la próxima revolución industrial, los átomos serán los nuevos bits") publicado en la revista Wired.

Desde entonces la conciencia DIY ha estado fuertemente instalada en nuestra sociedad, experimentado un auge en el que sigue inmersa, lo cual ha ido de la mano con la creación de numerosos tipos de Hardwares libres y low-cost, con el fin de facilitar la autosuficiencia a la hora de que el usuario pueda cubrir algunas de sus necesidades por sí mismo.

Antes de nada, destacar que la definición de "hardware" es la siguiente: "conjunto de elementos físicos o materiales que constituyen una computadora o un sistema informático".

En nuestro caso, el tipo de hardware principal en el que nos centraremos inicialmente serán las placas destinadas a llevar a cabo proyectos electrónicos, entre las que cabe destacar algunas como por ejemplo BeagleBone, Nanode, LittleBIts o Waspmote, pero por encima de todas las demás emergen dos marcas: Raspberry Pi y Arduino.

2.1. RASPBERRY PI

De acuerdo a propia web oficial de Raspberry Pi (Raspberry Pi – Teach, Learnd and Make with Raspberry Pi, 2018), la respuesta a la pregunta "¿qué es una Raspberry Pi?" es la siguiente:

"La Raspberry Pi es una computadora de bajo coste y dimensiones similares a las de una tarjeta de crédito, la cual se conecta a una pantalla de ordenador o a una televisión y utiliza un ratón y un teclado estándar. Es un dispositivo pequeño pero potente, apto para todas las edades, y que sirve para explorar la informática así como para aprender a programar en diversos lenguajes como Scratch o Python. Puede hacer todo lo que hace un ordenador de mesa, ya sea navegar por internet, reproducir vídeos en alta definición o ejecutar juegos.

También tiene la capacidad de interactuar con el exterior y se ha utilizado en un amplio rango de proyectos digitales, desde máquinas musicales a aplicaciones de control de presencia dirigidas a los padres, por ejemplo, situadas en colegios.

Podemos encontrar más información en What is a Raspberry Pi (n.d.).

"El objetivo del creador de Raspberry Pi, Eben Upton, era crear un dispositivo de bajo coste que mejorase las habilidades programadoras y el

entendimiento del hardware de usuarios a un nivel pre-universitario. Pero gracias a sus reducidas dimensiones y su precio tan accesible, rápidamente fue adoptada por usuarios de mayor nivel de conocimientos electrónicos, para proyectos que requerían más que un microcontrolador básico (que es lo que precisamente ofrecen los dispositivos Arduino).

La Raspberry Pi es más lenta que un ordenador portátil o de escritorio, pero aún así sigue siendo un ordenador completo con sistema operativo Linux, y puede proveer todos los usos que esto implica a bajo consumo.

Además, es un dispositivo de hardware libre, a excepción de su chip principal, el Broadcomm SoC, y también son de carácter libre y bien documentado una gran cantidad de proyectos llevados a cabo con ella".

Entre las especificaciones de la Raspberry destacan conexiones tales como la USB, SPI, HDMI o DPI, así como pines GPIO (General Purpose Input/Output). Éstas y otras funciones la habilitan para ser algo más que un ordenador, puede ser empleada en proyectos de electrónica o robótica, e interactuar con el exterior mediante sensores y actuadores.

Como ya se ha indicado anteriormente, sus principales ventajas son su carácter de código libre y su reducido precio.

Existen numerosos modelos, tales como: Raspberry Pi Zero W, 1 Model A+, 1 Model B+, 2 Model B, 3 Model B ó 3 Model B+, cuyo rango de precios varía entre los 5 y los 40 euros aproximadamente por solo la placa, existiendo también kits más completos por precios no muy superiores y generalmente por debajo de los 100 euros.

2.2. ARDUINO

En primer lugar, habrá que dar respuesta a una pregunta básica y fundamental para iniciar este proyecto: "¿qué es un Arduino?"

De acuerdo a su página oficial (Arduino – Home, 2018), Arduino "es una plataforma electrónica de código abierto basada en hardware y software de uso fácil, cuyas placas pueden leer entradas, por ejemplo mediante sensores de luz, pulsadores, etc., y devolver salidas, tales como el encendido de un LED o la activación de un motor".

Todo lo anterior se le ordenaría a la placa Arduino correspondiente mediante un conjunto de instrucciones enviadas al microcontrolador de dicha placa, utilizando el IDE de Arduino.

Este IDE está basado en "Processing", un lenguaje nacido en 2001 dirigido al aprendizaje a escribir código dentro del contexto de las artes

visuales (Processing, 2018), así como su lenguaje específico de programación, basado en la tecnología "Wiring", un marco de programación de código abierto dirigido a microcontroladores, el cual permite el uso de software multiplataforma para controlar diversos dispositivos conectados a una amplia gama de placas de microcontroladores, otorgando la posibilidad de crear una amplia variedad de proyectos (Wiring, 2018).

Arduino nace en el Instituto de Diseño Interactivo de Ivrea, localizado en Italia. Inicialmente surge como una herramienta de fácil uso para realizar prototipos de manera rápida, enfocada especialmente a estudiantes sin apenas formación en electrónica y programación.

Tan pronto como se extendió su uso hacia una comunidad mayor, Arduino comenzó a cambiar y adaptarse las nuevas necesidades que surgían, tanto las generales (internet de las cosas, impresión 3D…) como las necesidades particulares de cada usuario, en gran parte debido a su carácter de código abierto, carácter que comparte su Software, que crece día a día gracias a la contribución de todos los usuarios alrededor del mundo.

Ahora, una vez introducido, la segunda pregunta que se nos presenta sería: "¿por qué Arduino?"

Como ya se introdujo anteriormente, una de las principales características de las placas Arduino es su condición de código abierto, tanto en su parte hardware como en su parte software, lo cual favorece la existencia de gran cantidad de proyectos y, en consecuencia, también de una amplia cantidad de información, la cual está a disposición de cualquier usuario, es fácil de encontrar y además es de carácter gratuito.

También existe una extensa comunidad de usuarios, de distintos niveles de conocimiento, incluyendo expertos, que desarrollan, extienden y mejoran la experiencia Arduino, y de cuyo conocimiento cualquier usuario sin experiencia podrá obtener ayuda.

Tanto software como hardware, son, por tanto, ampliables. El software se puede ampliar mediante librerías de C++ y, de forma más detallada, se podría acceder también a la programación en lenguaje C en el que está basado. En lo que respecta al hardware, los planos de los microcontroladores están publicados, por lo que cualquiera puede hacer su propia versión del módulo, ampliándolo o mejorándolo.

Otra de las principales ventajas es, por supuesto, la económica.

Las placas Arduino son por lo general bastante baratas, en especial en comparación con otras plataformas de microcontroladores. Por lo general el

precio de casi todos los modelos es inferior a los 50€, existiendo además kits de iniciación más completos y a precios también muy asequibles, ya que normalmente oscilan entre los 60 y los 80€.

Una característica reseñable de Arduino es su condición de software multiplataforma, ya que funciona en todos los principales sistemas operativos, véase Windows, Linux y Mac.

Por último, cabría destacar su entorno de desarrollo integrado (IDE), simple e intuitivo, fácil de usar para principiantes y aún así también flexible para los más expertos. Además, al estar basado en el entorno de Processing, es adecuado para actividades de enseñanza así como para los usuarios ya iniciados en dicho entorno.

Si bien existen también ciertas desventajas, entre las que podríamos destacar la ralentización del programa al utilizar librerías, que no suelen estar optimizadas, el alto uso de su memoria o la necesidad de módulos extra para ciertas funciones básicas como puede ser la propia conexión a internet.

Existe un amplio catálogo de productos Arduino, incluyendo desde las propias placas a diversos módulos. El catálogo existente en la actualidad es el mostrado en la Imagen 2.1:

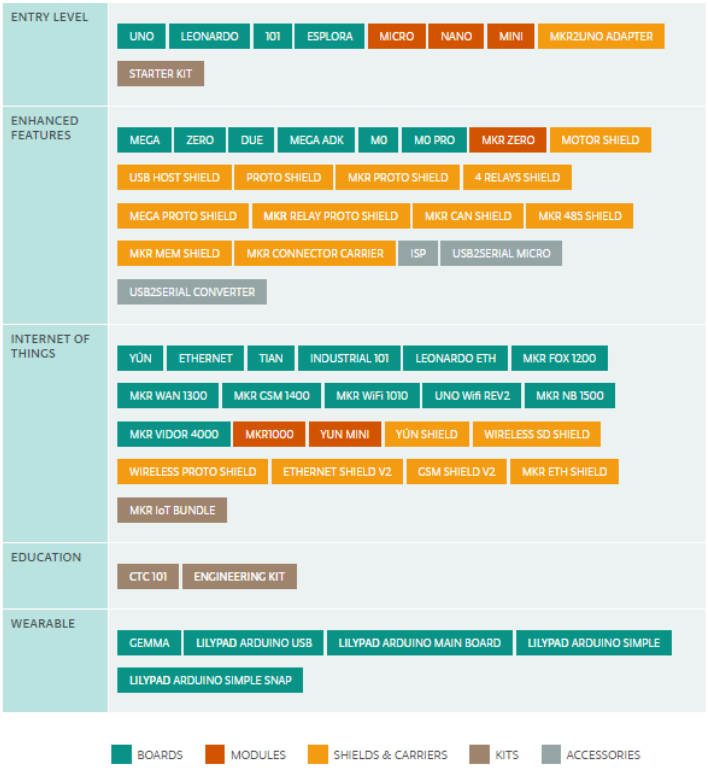


Imagen 2.1: *Catálogo de productos Arduino.*

Fuente: Arduino - Home, 2018.

2.3. ARDUINO VS RASPBERRY PI

En este subcapítulo se pondrán de manifiesto las principales diferencias entre las dos placas líderes del mercado, que son Arduino y Raspberry Pi.

La diferencia principal es que, mientras que la Raspberry Pi es un microordenador, Arduino es únicamente un microcontrolador, un circuito lógico programable, lo que podría ser una parte concreta de un microordenador.

Debido a esto, parece obvio que la Raspberry Pi está más enfocada a la informática, mientras que el uso de Arduino tiene un enfoque dirigido a la electrónica.

Arduino es además fácilmente programable, en gran parte gracias a sus librerías, sus entradas y salidas están mejor protegidas y enfocar un programa a la realización de una determinada acción simple es mucho más fácil de lo que sería llevar a cabo la misma acción en una placa Raspberry, además de ser mayor la posibilidad de quemar alguna de sus entradas o salidas si no se utiliza la resistencia correcta.

A favor de la Raspberry Pi cabe destacar su mayor facilidad de conexión, ya sea a internet (la cual se puede realizar de forma directa, mientras que en Arduino sería necesario un módulo extra) o mediante su puerto USB integrado, por ejemplo.

Raspberry es también más rápida, y a su vez más cara, además de requerir ciertos conocimientos de su sistema operativo.

Las principales fortalezas de Arduino, por tanto, son su enfoque dirigido a la electrónica, su menor consumo, la posibilidad de conectarlo y desconectarlo directamente sin causar daños a la placa, y, sobre todo, su mayor simpleza a la hora de llevar a cabo ciertos proyectos, especialmente los dirigidos a hacer cosas concretas, no excesivamente complejas, y especialmente si van a ser desarrollados por usuarios novatos o sin cierto nivel de experiencia.

Para ambas placas existe una grandísima cantidad de información online así como una extensa comunidad de usuarios, pero quizá por su mayor simpleza y su enfoque a la electrónica, para el proyecto que nos ocupa se escogió trabajar con una placa Arduino.

2.4. HARDWARE EMPLEADO

En este apartado se presentan todos los componentes de hardware empleados en la construcción del dispositivo final, que serán: placa Arduino Uno, módulo lector MiFare RC-522, módulo RTC DS3231, pulsador, diodos leds, resistencias y relé.

2.4.1. ARDUINO UNO

Concretamente, en este proyecto se ha utilizado el modelo Arduino UNO, el modelo más básico y a su vez el más adecuado para iniciarse en este tipo de placas, además de ser el más utilizado y el que está dotado de una mayor cantidad de información en la red. Es, por así decirlo, el modelo estándar.

Arduino Uno es un microcontrolador basado en el ATmega328P. Posee, entre otras cosas, catorce pines de entrada/salida digitales, seis entradas analógicas, una conexión USB, un botón de reseteo o un power jack.

El nombre de "Uno" proviene del lanzamiento del software IDE 1.0 de Arduino, siendo "Uno" el significado de "1" en italiano, donde se creó Arduino. La placa Uno y la versión del IDE de Arduino 1.0 fueron las versiones de referencia de Arduino.

Las especificaciones técnicas de la placa se muestran a continuación en la Imagen 2.2:

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Imagen 2.2: *Especificaciones técnicas del Arduino Uno*

Fuente: Arduino – Home, 2018.

A continuación se ofrece una imagen de la perspectiva superior de la placa Arduino Uno (Imagen 2.3), con sus partes más relevantes señalizadas.

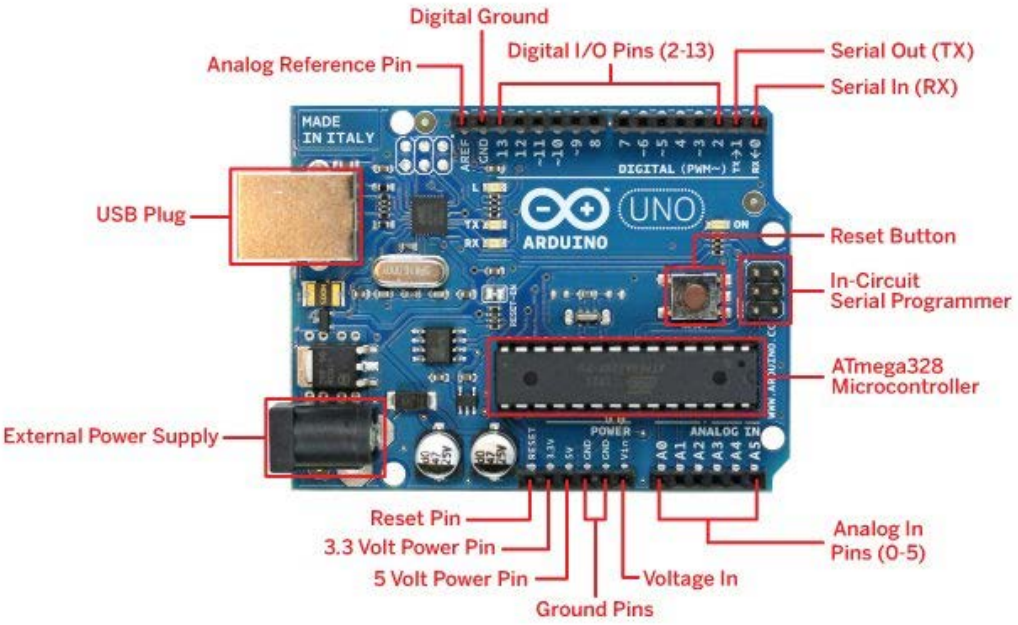


Imagen 2.3: *diagrama de pines de la placa Arduino Uno*

Fuente: Aprendiendo Arduino, 2016.

2.4.2. MÓDULO LECTOR RFID-RC522

Para llevar a cabo el proyecto, el módulo principal que se utiliza es un lector de tarjetas y otros dispositivos, normalmente llaveros o etiquetas adhesivas, denominados TAGs, basado en tecnología de radiofrecuencia, RFID (Radio Frequency IDentification). Concretamente, se utiliza el módulo RFID-RC522, el cual se muestra en la Imagen 2.4.

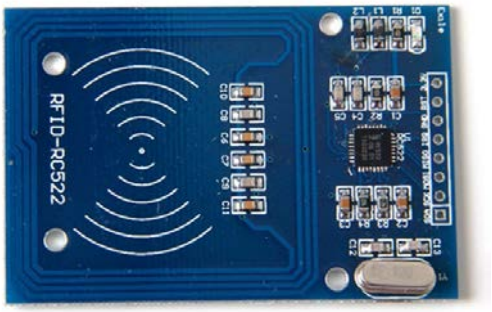


Imagen 2.4: *Módulo RFID-RC522.*

Fuente: Amazon, 2018.

Algunas de las características más reseñables de este módulo lector (que también puede funcionar como escritor) son:

- Frecuencia de comunicación: 13.56 MHz.
- Comunicación SPI.
- Alimentación: entre 2.5 y 3.3 V

Toda la información del módulo se encuentra disponible en The Arduino Playground (2018).

También se puede acceder a la librería principal encargada del manejo del módulo RFID desde Arduino, creada por miguelbalboa (GitHub, 2018).

Dicha librería es de código abierto y maneja la lectura y escritura de tarjetas RFID mediante el uso del módulo RC522, conectado a través de SPI (Serial Peripheral Interface).

2.4.3. MÓDULO RTC DS3231

Una de las desventajas de Arduino, ya señalada con anterioridad, era la necesidad de módulos externos para llevar a cabo ciertas funciones de carácter básico. Una de dichas funciones sería el control de la fecha y la hora, que Arduino no posee internamente, necesitando un módulo RTC para dotarle de ellas con fiabilidad.

RTC son las siglas en inglés de “Real Time Clock”, lo que en castellano quiere decir “reloj en tiempo real”, que es básicamente la función de este módulo, ya que es un dispositivo electrónico encargado de obtener mediciones temporales reales.

Existen otros modelos con sus correspondientes librerías, como por ejemplo el RTC DS1307, pero en este proyecto se escogió el modelo DS3231, dado que posee una mayor precisión, ya que incorpora medición y compensación de la temperatura, “garantizando una precisión de al menos 2ppm, lo que equivale a un desfase máximo 172ms/día o un segundo cada 6 días” (Llamas, 2018), que además añade que “en el mundo real normalmente consiguen precisiones superiores, equivalente a desfases de 1-2 segundos al mes”.

Los módulos RTC no suelen venir incluidos en los kit de iniciación de Arduino, pero se pueden conseguir a precios bastante asequibles. En concreto, éste costó alrededor de los 7 euros, adquirido en Amazon, aunque ciertos distribuidores lo ofrecen a precios incluso inferiores, alrededor de un euro.

Entre sus características más destacadas está que su comunicación se realiza a través del bus I2C, o que requiere una alimentación entorno a los

2,3 y los 5 V, estando dotado también de una batería de litio de 3V con el fin de que sea capaz de mantener la hora establecida aún estando desconectado de la corriente o de la propia placa Arduino.

La Imagen 2.5 muestra el aspecto físico de un módulo RTC.

Imagen 2.5: *RTC DS3231.*

Fuente: Fábrica Digital, n.d.

2.4.4. PULSADOR

Los pines digitales de Arduino pueden funcionar tanto de entradas como de salidas. En el caso de un pulsador, el pin correspondiente funcionará como entrada digital, pudiendo leer valores de entrada de HIGH o LOW.

Un pulsador es un botón propiamente dicho, que se conecta a la placa Arduino a un pin digital, ayudado de una resistencia, a través del cual leerá el valor del botón, si éste está pulsado o no. Dicho valor podrá ser HIGH o LOW en cada caso según el método de montaje en el que se disponga.

Los métodos de conexión a destacar son PULL UP y PULL DOWN, y ambos incluyen la presencia de resistencias, pero con distinta disposición.

En el método PULL UP, el valor que fuerza cuando el pulsador está abierto es HIGH, es decir, cuando no se pulsa el botón, mientras que al pulsarlo el estado será LOW.

En el método PULL DOWN es exactamente al contrario, el estado inicial del botón es de LOW, y tras pulsarlo se modificará hacia un valor de HIGH.

El hecho de que esta conexión proporcione un valor de LOW (0) en estado de reposo suele hacerla más deseable, ya que evita ciertas lecturas erróneas y además ahorra consumo. Es, por tanto, el tipo de conexión más empleado y también el que se empleará en este proyecto.

La resistencia que se utiliza es una resistencia de 10kΩ, cuyo código de color es marrón-negro-naranja, con el fin de evitar que tenga ninguna influencia sobre el circuito.

Las funciones para las que se empleará el pulsador se explicarán en el capítulo correspondiente.

2.4.5. LEDS

Un elemento muy común en cualquier proyecto electrónico es la existencia de señales luminosas, con diversos fines.

Una de las funciones principales de los elementos luminosos es la seguridad, señales lumínicas para advertir el correcto funcionamiento o no de ciertos dispositivos, indicaciones de peligro, etc.

Como introducción, cabe reseñar que un diodo es un dispositivo formado por dos electrodos que únicamente permite el paso de la corriente en un sentido.

En este proyecto se utilizan diodos LEDs (diodos que emiten luz al ser atravesados por una corriente) de tres colores (rojo, amarillo y verde), y diámetro 5mm, incluidos en cualquier kit de iniciación de Arduino, con el fin de mostrar el estado de ejecución en el que se encuentra el programa, así como el correcto o incorrecto funcionamiento de sus diversas partes.

Los LEDs, como cualquier diodo, tienen polaridad. Su patilla más larga es la positiva, y la más corta la negativa, por lo que han de ser conectados correctamente para que funcionen y también para que no se fundan, lo cual también se logra añadiendo una resistencia en el circuito correspondiente.

El valor nominal de la resistencia se calcula aplicando la ley de Ohm de la siguiente manera, a partir de los valores ofrecidos por el fabricante:

I = 20mA: Intensidad máxima que puede atravesar por el LED.

El LED se alimenta con una fuente de 5V, y la caída de tensión en el es de aproximadamente 2,2 V.

$V_{fuente} - V_{led} = 5V - 2{,}2V = 2{,}8\ V$

$V = I \times R \Rightarrow R = V / I$

$R = 2.8\ V / 0.02\ A = 140\ \Omega$

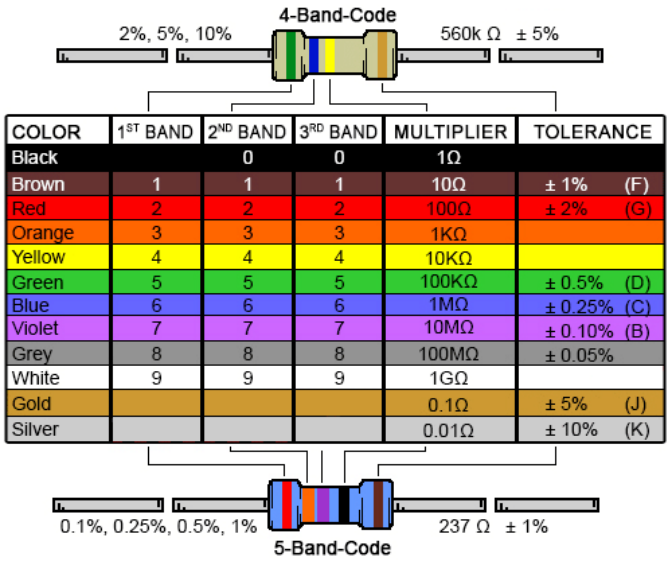
Por aproximación, la primera resistencia comercial que existe con un valor cercano a los 140 Ω de la resistencia nominal es la de 220 Ω, resistencias que se utilizarán en el desarrollo de este proyecto.

2.4.6. RESISTENCIAS

Se entiende por resistencias o resistores los componentes electrónicos que se intercalan en un circuito para modificar el paso de la corriente, oponiéndose a su paso mediante la generación de una resistencia eléctrica.

La aplicación de resistencias se utiliza generalmente con el fin de limitar la corriente eléctrica que atraviesa un componente, para evitar peligros o reducir un posible deterioro o destrozo del componente.

El valor de una resistencia puede ser fácilmente identificado mediante un código de colores, que se adjunta a continuación en la imagen 2.6:



COLOR	1ST BAND	2ND BAND	3RD BAND	MULTIPLIER	TOLERANCE
Black		0	0	1Ω	
Brown	1	1	1	10Ω	± 1% (F)
Red	2	2	2	100Ω	± 2% (G)
Orange	3	3	3	1KΩ	
Yellow	4	4	4	10KΩ	
Green	5	5	5	100KΩ	± 0.5% (D)
Blue	6	6	6	1MΩ	± 0.25% (C)
Violet	7	7	7	10MΩ	± 0.10% (B)
Grey	8	8	8	100MΩ	± 0.05%
White	9	9	9	1GΩ	
Gold				0.1Ω	± 5% (J)
Silver				0.01Ω	± 10% (K)

Imagen 2.6: *Código de colores de las resistencias.*

Fuente: Digikey, 2018.

En concreto, para la construcción de este proyecto se han utilizado tres resistencias de valor 220 Ω para la conexión de los leds, y una resistencia de 10 kΩ para el montaje del pulsador.

2.4.7. RELÉ

Rudolf Graf, en su libro del año 1984, “Diccionario de Electrónica”, definió el relé de la siguiente manera:

“Dispositivo electromecánico cuyos contactos están abiertos y/o cerrados debido a variaciones en las condiciones de un circuito eléctrico, lo que afecta operativamente a otros dispositivos en el mismo circuito o en otro”.

A efectos prácticos, se puede decir que el relé es un interruptor que tiene dos estados, “NO” (normalmente abierto) y “NC” (normalmente cerrado). Es a estas salidas (NO, NC y C, común) donde se conecta el circuito secundario que se quiere manejar de manera independiente al circuito principal, teniendo en cuenta las variaciones de NO a NC y viceversa que puede provocar el circuito principal al secundario, asegurándose de que se obtienen los resultados deseados en este nuevo circuito.

En el caso de este proyecto, se podría conectar al relé la puerta de acceso que se desee controlar, originalmente cerrada (a las salidas NO y C) y permitiendo su apertura tras escanear un tag con accesos permitidos (salidas NC y C). Esto se consigue programando cambios de valor en el relé de LOW a HIGH y viceversa dependiendo del estado de ejecución del programa y del tag escaneado, y con una conexión del circuito secundario en consecuencia a lo que se programa.

La utilización de relés en la industria está muy extendida desde hace décadas, sin haber quedado aún obsoletos.

Existen diferentes tipos de relé. Podemos encontrar algunos de estos tipos en el portal educativo ElProCus (2015), que son los siguientes:

- Relés electromecánicos.
- Relés de estado sólido.
- Relés de láminas.
- Relés híbridos.
- Relés térmicos.

El modelo de relé que se emplea en este proyecto es módulo de relé simple de 10 A, incluido en el kit de iniciación de Arduino (Imagen 2.7), mientras que su conexión con la placa se explica posteriormente.

Las especificaciones del fabricante de este relé se pueden encontrar en Single Relay Board #27115 (2013).

Imagen 2.7: Relé.

Fuente: Amazon, 2018.

2.5. PROTOTIPADO DEL PROYECTO

Para el montaje físico de este proyecto, una vez seleccionado el hardware que se va a emplear, mencionado en el apartado anterior, se necesitarán elementos de conexión tales como cables machos y hembras, y se conectarán todos los elementos mediante el uso de una placa de pruebas o protoboard.

Primeramente, se da paso al montaje del botón pulsador, mediante un montaje de resistencia PULL DOWN, por los motivos explicados anteriormente.

El esquema teórico de montaje se muestra en la Imagen 2.8, mientras que el montaje a efectos prácticos en nuestro proyecto queda reflejado en la Imagen 2.9.

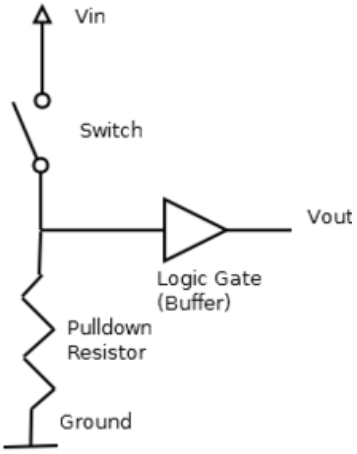


Imagen 2.8: Montaje pulsador PULL DOWN teórico.

Fuente: The Arduino Playground, 2018.

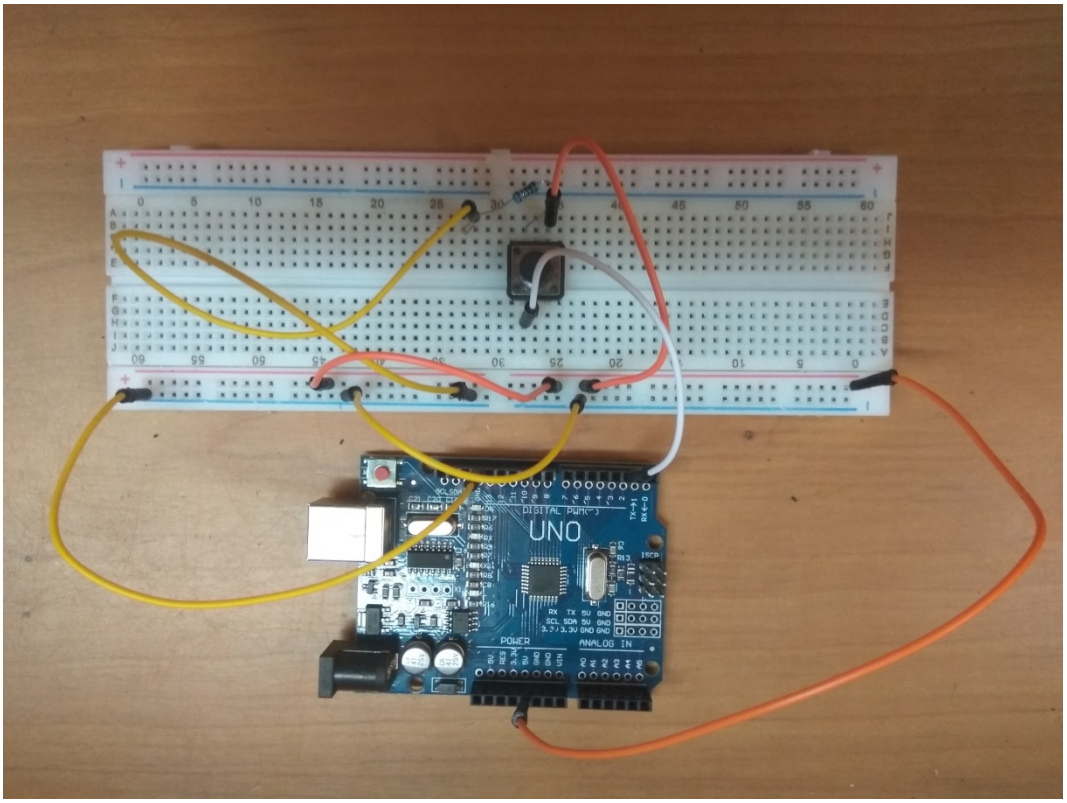


Imagen 2.9: montaje pulsador PULL DOWN práctico.

Como se observa en la imagen, el pulsador va conectado por una parte a la alimentación de 5V (aplicada en la protoboard mediante conexión directa de la línea + con una fuente de alimentación de 5V en la placa Arduino), por otra a la toma de tierra (también provista por un pin GND de Arduino que se lleva a la línea – de la protoboard) y, finalmente, y con la ayuda de una resistencia de 10 kΩ, estará conectado al pin digital número 2 (el número de pin es de libre elección, siempre y cuando se programe acorde al montaje realizado).

A continuación, se procede al montaje del módulo lector RFID RC-522. Para ello, el primer paso es soldar la tira de pines curvos incluida con el kit, que servirá para conectar el lector a la placa.

Una vez se ubica el lector en la protoboard, se procede a su conexión con la tarjeta Arduino. Las conexiones habituales entre el RC522 y el Arduino Uno son las especificadas en la Tabla 2.1:

RC522	ARDUINO UNO
SDA	PIN 10
SCK	PIN 13
MOSI	PIN 11
MISO	PIN 12
IRQ	NO CONECTADO
GND	GND
RST	PIN 9
3.3V	TOMA DE 3.3V

Tabla 2.1: *Conexiones RC522 - Arduino Uno*

Fuente: elaboración propia a partir de Arduino – Home, 2018.

La imagen 2.10 muestra el resultado de la conexión real entre Arduino y el módulo lector en nuestro proyecto:

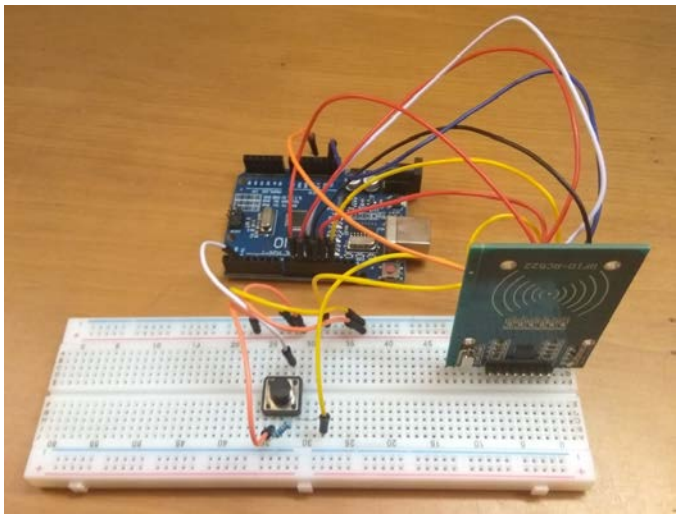


Imagen 2.10: *Montaje del módulo lector RFID.*

En tercer lugar se procede a la conexión del relé, que solo dispone de tres pines, toma de alimentación (5V), toma de tierra (GND) y el pin que se conectará al pin digital de Arduino correspondiente, en nuestro caso, el pin número 4. El montaje se puede observar en la Imagen 2.11:

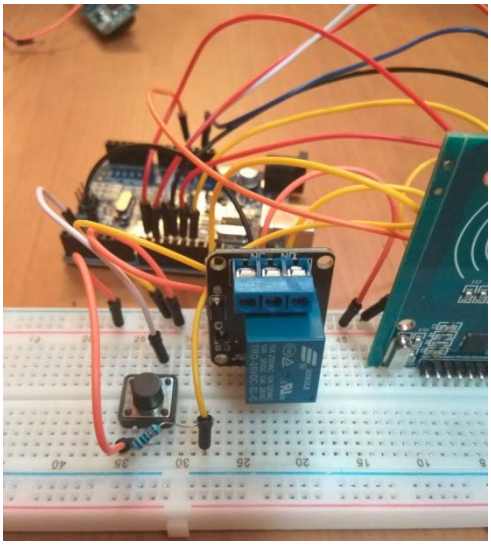


Imagen 2.11: montaje del relé.

Para continuar con la disposición de los elementos en la otra mitad de la protoboard, se emplean dos cables jumper que hagan las veces de puente, conectando ambas mitades de las líneas + y – de la placa de pruebas, facilitando así que exista conexión a la toma de alimentación de 5V de Arduino y a la toma de tierra a lo largo de toda la placa.

Tras ésto, se conectarán a la placa los tres leds (verde, amarillo y rojo). A continuación se presentan algunos ejemplos de esquemas de conexión de los LEDs a Arduino (Imagen 2.12):

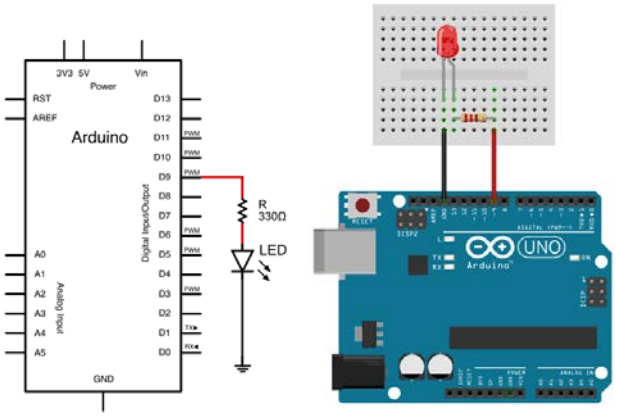


Imagen 2.12: esquema de conexión de un LED.

Fuente: Llamas, 2018.

La conexión a efectos prácticos de los tres leds sobre la protoboard se puede observar en la imagen 2.13:

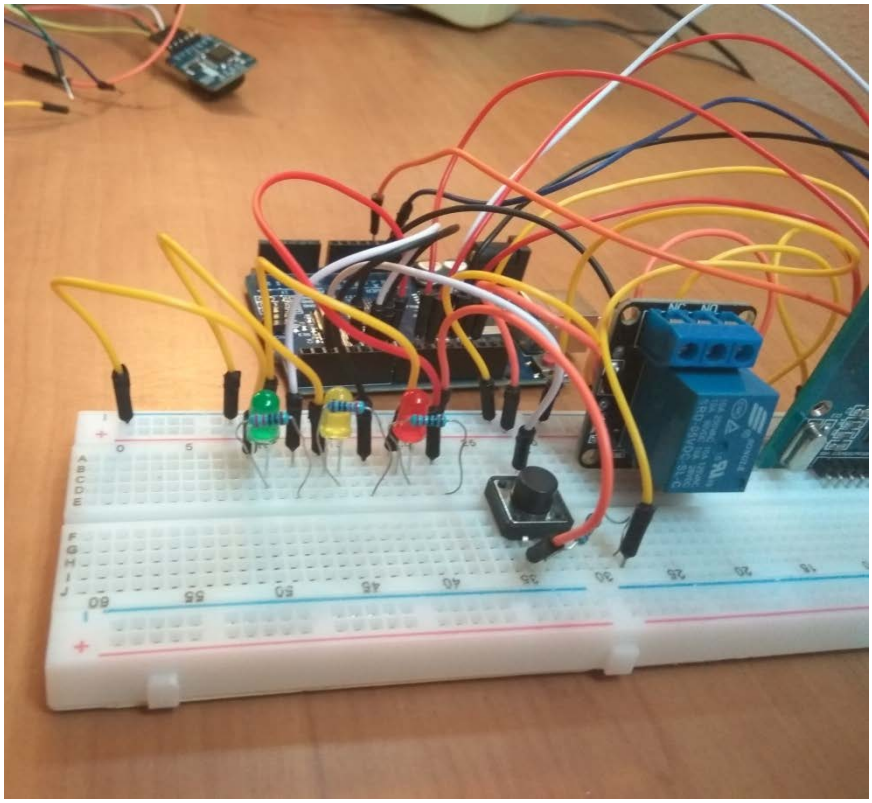


Imagen 2.13: conexión de los tres leds.

Finalmente, se procede a la conexión del módulo RTC, la cual se realizará aparte por motivos de seguridad, ya que a lo largo de las diversas pruebas realizadas sobre el funcionamiento de este proyecto, ha sido el componente que más problemas ha provocado, si bien sobre el papel nada impide de manera teórica que pudiese compartir alimentación y toma de tierra con el resto de componentes.

El esquema de conexiones del módulo con la placa Arduino es el siguiente:

RTC DS3231	ARDUINO UNO
SCL	A5
SDA	A4
Vcc	TOMA DE 5V
GND	GND

Tabla 2.2: conexiones RTC – Arduino.

Fuente: elaboración propia a partir de Arduino – Home, 2018.

Y en la Imagen 2.14 se muestra la conexión real del módulo a la placa Arduino.

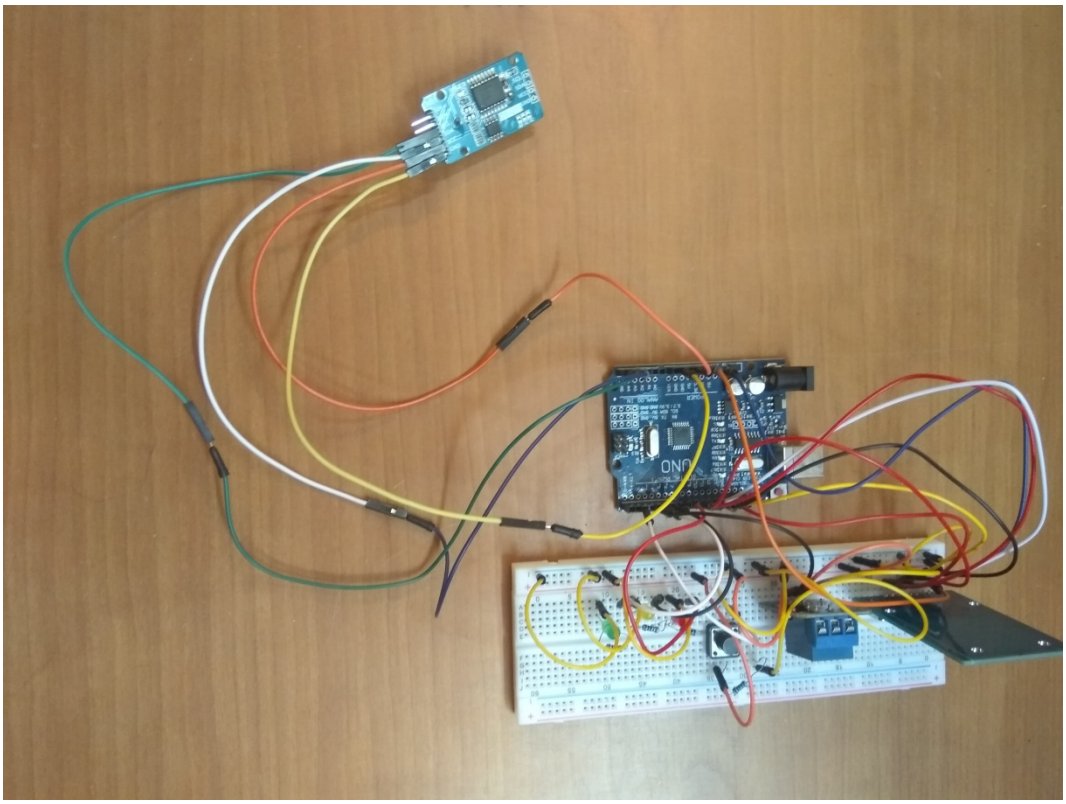


Imagen 2.14: *conexión del módulo RTC DS3231.*

Finalmente, en las imágenes 2.15 y 2.16 se muestra el prototipo final del proyecto así como una imagen de los tags y el cable USB de alimentación empleados.

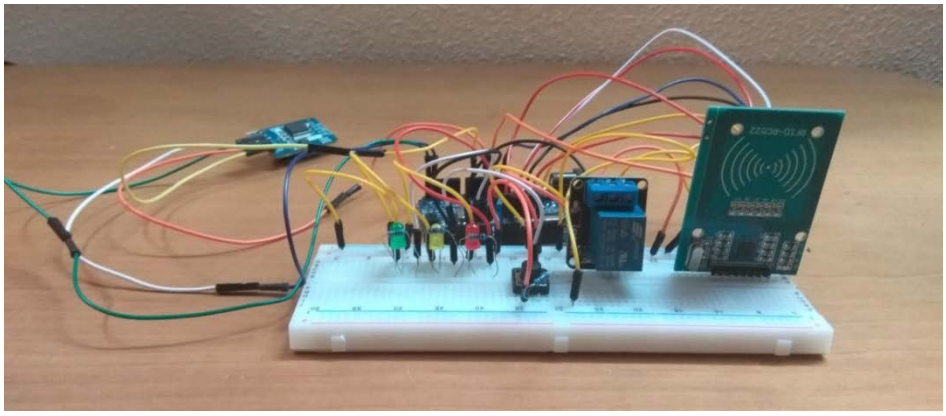


Imagen 2.15: *prototipado final del proyecto.*

Nota: todas las conexiones a tierra han sido realizadas mediante cables de color amarillo, y todas las conexiones a alimentación mediante cables de color naranja con el fin de facilitar su identificación visual.

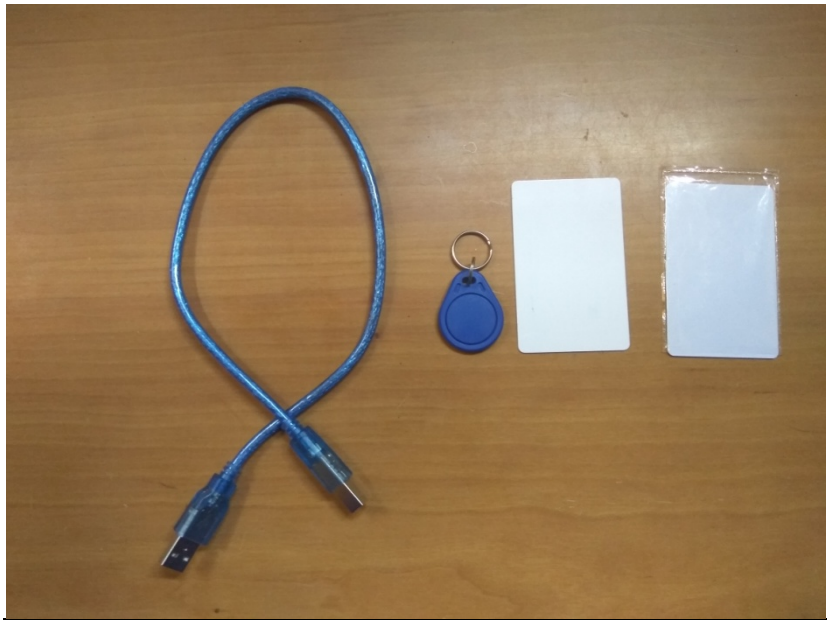


Imagen 2.16: tags y cable USB empleados en el proyecto.

Una vez construido el prototipo, será necesario escribir el código o códigos que logren llevar a cabo los objetivos marcados inicialmente para este proyecto. La programación representa la fase de empleo de software del proyecto.

En este caso, los lenguajes de programación que se utilizan son el lenguaje Arduino y el lenguaje Python, introducidos en los capítulos 3 y 4 respectivamente, así como todo el software necesario para cumplir este propósito.

CAPÍTULO 3:

SOFTWARE EMPLEADO Y PROGRAMACIÓN DEL PROYECTO: ARDUINO

3.1. IDE ARDUINO

Como ya se introdujo en el apartado 2.2, Arduino, al igual que poseía un lenguaje propio de programación basado en C, tiene también su propio entorno de desarrollo, de carácter libre y multiplataforma, escrito en Java y basado en la tecnología Processing, que funciona en cualquier modelo de placa Arduino.

Para su instalación no hay más que ir al apartado Download de su web oficial y proceder a su descarga y posterior instalación como con cualquier otro programa (seleccionando los paquetes que se desean instalar así como el directorio de instalación), si bien es cierto que también existe un IDE online.

Una vez instalado, el IDE requiere una configuración básica para su funcionamiento. En Arduino – Home (2018) se puede encontrar una guía completa sobre el entorno de Arduino, pero para el usuario que se dispone a dar sus primeros pasos, esta configuración se podría resumir en tres aspectos fundamentales:

En el bloque de Herramientas, dentro de Placa se ha de seleccionar el modelo de placa Arduino con el que se va a trabajar, en nuestro caso, Arduino/Genuino Uno.

Dentro también del bloque de Herramientas, en Programador se debe escoger “AVRISP mkII”.

Por último, y en este mismo bloque, se debe seleccionar el “Puerto” USB mediante el que se conecta la placa a la computadora. Esto puede variar dependiendo del sistema operativo que se utilice, pero por lo habitual, en Windows aparece directamente la opción “COM” y el número de puerto al que se encuentra conectado la placa. En este TFG se ha utilizado el puerto de entrada “COM3”, puerto que habrá que especificar en los parámetros de conexión serial tanto en Arduino como en Python.

Una vez realizados estos pasos, se puede proceder a la escritura de los programas, que en Arduino se suelen llamar “sketchs”, y cuya extensión de archivo será “.ino”.

3.1.1. LIBRERÍAS

Para la programación de sketchs, es muy habitual la utilización de las denominadas “librerías”, colecciones de código ya existentes que facilitan multitud de tareas de la programación. Estas librerías suelen estar basadas en C y, aunque ralentizan la ejecución del programa, son muy útiles.

Originalmente, el IDE de Arduino viene con diversas librerías instaladas, pero es la gran colección de librerías presentes en internet lo que hace de Arduino un lenguaje aún más potente.

Un listado de las librerías más importantes se puede consultar en el apartado correspondiente de la web oficial de Arduino.

Para la instalación de librerías descargadas en internet (en formato .zip), únicamente habrá que dirigirse al apartado "Programa" en la barra de herramientas del IDE, dentro de éste, habría que seleccionar "Incluir Librería", y finalmente seleccionar la opción "Añadir Librería .ZIP…", donde se permite cargar el archivo .zip deseado. Tras reiniciar el IDE, ya se podrá trabajar con la librería deseada.

Las librerías empleadas en la programación de este proyecto son:

- Librería EEPROM: EEPROM son las siglas de "Electrically Erasable Programmable Read-Only Memory", o memoria ROM programable y borrable eléctricamente. Es decir, es una memoria ROM de Arduino, no volátil, cuyos datos permanecerán almacenados tras desconectar la placa al menos que se proceda a su borrado.

  Para poder manejar esta memoria de la placa, se ha de utilizar esta librería, permitiéndonos grabar y borrar datos en dicha memoria, así como leer información ya almacenada.

- Librería SPI: permite la comunicación serial con otros dispositivos valiéndose del bus SPI.

- Librería Wire: permite la comunicación en ambos sentidos con otros dispositivos.

Estas tres primeras librerías vienen incluidas por defecto en el IDE de Arduino, mientras que las dos que siguen requieren instalación:

- Librería MFRC522 (miguelbalboa): librería que permite y facilita el manejo del módulo lector RFID RC-522 mediante la inclusión de funciones que contienen la programación de las tareas que se requiere que desempeñe el lector, tales como la lectura de tags o la comunicación con el puerto serie.

- Librería RTClib (adafruit): librería para el manejo del módulo RTC DS3231, que nos permite entre otras cosas configurar su fecha y hora, así como recibir posteriormente datos provistos

por el módulo, ya sean fechas, horas o temperaturas, para poder procesarlos en el programa.

Ambas librerías, como prácticamente todas, las podemos encontrar en el sitio web dedicado a código Github (2018).

3.2. CÓDIGO EN ARDUINO

Como se señaló inicialmente, Arduino es un software de carácter libre y con una gran comunidad de usuarios, lo cual era una de sus principales ventajas, y como ejemplo de ello cabe destacar que el código utilizado en Arduino para este proyecto está basado en gran medida en el código del programa localizado en RFID RC-522 (2016), cuyo esqueleto se mantiene en gran parte.

3.2.1. Primer bloque

El primer bloque de código corresponde al representado en las imágenes 3.1 y 3.2.

```
#include <EEPROM.h>   // Lectura y escritura de la EEPROM
#include <SPI.h>      // Protocolo SPI
#include <MFRC522.h>  // Librería para el manejo del módulo lector RC522
#include <RTClib.h>   // Manejo del módulo RTC
#include <Wire.h>

// Definición de los Pines
#define pulsador 2      // Define el pin para el pulsador
#define relay 4      // Define el pin para el relé

//Pines de los LEDs
#define amarillo 6
#define verde 5
#define rojo 7

// Pines para la conexión del módulo RFID RC522
#define RST_PIN 9
#define SS_PIN 10
```

Imagen 3.1: *Primer bloque de código (1/2)*

En primer lugar, se incluyen todas las librerías necesarias para el correcto funcionamiento del programa, ya mencionadas en el apartado 3.1.1.

```
MFRC522 mfrc522(SS_PIN, RST_PIN);   // Crea una instancia MFRC522
RTC_DS3231 rtc;                     // Crea una instancia RTC

int successRead;                    // Variable para verificar una correcta lectura

bool estado = false;                // Arduino encendido o apagado
boolean match = false;              // CARD inicialmente en estado "falso"
boolean programMode = false;        // Modo programa inicialmente en estado "falso"


char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday",
"Wednesday", "Thursday", "Friday", "Saturday"};

byte storedCard[4];                 // Almacena una UID leída de la EEPROM
byte readCard[4];                   // Almacena una UID leída del módulo lector
byte masterCard[4];                 // Almacena la UID de la MasterCard leída de la EEPROM
```

Imagen 3.2: Primer bloque de código (2/2)

Las primera línea crea un objeto o instancia RC522 (lector) con la que poder trabajar en el programa y repercutir en el módulo real, mientras que la segunda línea cumple la misma función pero con una instancia para el módulo RTC, con la misma finalidad.

Las siguientes líneas definen las siguientes variables globales:

- Una variable de carácter entero, “successRead”, cuya función es verificar que el RC522 ha obtenido una lectura correcta.
- Tres variables booleanas, “estado”, “match” y “programMode” con un valor inicial por defecto de falso, cuyas funciones son, respectivamente, verificar si el Arduino está en funcionamiento o no, si las UIDs comparadas coinciden o no, y si el sketch se encuentra en ejecución dentro del “modo programa” o no.
- Una variable char denominada “daysOfTheWeek” en la que se incluyen los días de la semana, para su posterior tratamiento con el módulo RTC.
- Tres variables byte de tamaño 4, denominadas “storedCard”, “readCard” y “masterCard”, cuyo objetivo es almacenar UIDs, respectivamente una UID leída de la EEPROM, una UID leída del módulo lector o la UID de la MasterCard, también leída desde la EEPROM.

3.2.2. Setup

El código perteneciente a la función setUp corresponde al representado por las imágenes desde la 3.3 hasta la 3.12.

```
//----------------------------------------Setup---------------------------
void setup(){
  // Configuración de los pines

  // Establece los pines de los LEDs y el relé como salida
  pinMode(verde, OUTPUT);
  pinMode(amarillo, OUTPUT);
  pinMode(rojo, OUTPUT);
  pinMode(relay, OUTPUT);

  // Establece el pin del pulsador como entrada
  pinMode(pulsador, INPUT);

  // LEDs inicialmente apagados
  digitalWrite(verde, LOW);
  digitalWrite(amarillo, LOW);
  digitalWrite(rojo, LOW);

  digitalWrite(relay, LOW); // Inicializa el relé con valor "LOW",
  //es decir, tras realizar las conexiones pertinentes, puerta cerrada
```

Imagen 3.3: *Función setup.*

En primer lugar (Imagen 3.3), se establecen los pines antes adjudicados a los leds y al relé como pines de salida, OUTPUT, y el pin asociado al pulsador se establece como salida, INPUT.

Además, se les da un valor inicial, que en el caso de los LEDs es de LOW, o lo que es lo mismo, apagados. También al relé se le da un valor de LOW, permitiendo que una vez se conecte el circuito a secundario a controlar al relé este se abra al pasar el relé a tener un valor de HIGH, ya que de esta manera, consumirá potencia solo al activarse, es decir, al "abrirse", y nunca en estado de reposo.

Por su parte, el pulsador no requiere el establecimiento de un valor inicial, ya que por defecto no estará pulsado (LOW) y al pulsarse pasará a un valor "HIGH" debido a su conexión PULL DOWN, explicada en el capítulo anterior.

```
//Protocol Configuration
Serial.begin(9600);    // Inicializa la comunicación serial con el PC a una velocidad de 9600 baudios
SPI.begin();           // Inicializa el protocolo SPI, utilizado por el módulo MFRC522 para comunicarse
mfrc522.PCD_Init();    // Inicializa el módulo lector MFRC522.
delay(500);
```

Imagen 3.4: *Función setup.*

A continuación (Imagen 3.4) se establecen los parámetros de comunicación siguientes: se inicializan la comunicación serian con el ordenador, a una velocidad de 9600 baudios y el protocolo de comunicación SPI (Serial Peripheral Interface). Se inicializa también el módulo lector real y se añade, además, una espera de medio segundo con el fin de dar tiempo a la correcta inicialización de los protocolos antes de continuar con la ejecución del programa.

```
  if (! rtc.begin()) {
    Serial.println("Error al iniciar el módulo RTC");
    while (1);
  }

if (rtc.lostPower()) // Si el RTC ha dejado de estar alimentado, lo pone en hora
  {
    Serial.println("El RTC ha perdido la alimentación, reprogramando la hora...");
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__))); // Le otorga la hora a la que este sketch fue compilado
    // Para establecer una hora y fecha concreta, por ejemplo:
    // 21 de enero de 2014, 3 AM::
    // rtc.adjust(DateTime(2018, 6, 12, 14, 15, 0));
  }
```

Imagen 3.5: Función setup.

Esta función (Imagen 3.5) pertenece a la librería RTClib. Inicia el módulo y comprueba tanto que éste ha iniciado de manera correcta, como que el no haya sufrido una pérdida de conexión y se hayan reseteado sus datos de fecha y hora. En caso de que así hubiese sido, lo vuelve a poner en fecha y hora con los datos obtenidos del momento de compilación del sketch (también se podría poner en una hora específica con la misma función rtc.adjust).

```
// SI SE INICIA EL PROGRAMA CON EL BOTON PULSADO,
// SE INICIA EL PROCESO DE BORRADO DE LA EEPROM. SI SE SUELTA ANTES DE 10s, CANCELA
 if (digitalRead(pulsador) == HIGH)
 {
   digitalWrite(rojo, LOW);
   digitalWrite(amarillo, LOW);
   delay(100);
   digitalWrite(rojo, HIGH);
   digitalWrite(amarillo, HIGH);
   Serial.println(F("Botón de borrado pulsado..."));
   Serial.println(F("Tiene 10 segundos para cancelar..."));
   Serial.println(F("Se borrarán todos los registros!"));
   delay(10000); // Otorga al usuario 10 segundos para soltar el pulsador y cancelar
```

Imagen 3.6: Función setup.

En la parte del código correspondiente a la Imagen 3.6 se actualiza la lectura del estado inicial del pulsador. Si está pulsado (valor HIGH) entra dentro de la función destinada a borrar todos los registros de la EEPROM del Arduino. Las cinco primeras líneas encienden los LEDs amarillo y rojo (tras asegurarse de que están apagados inicialmente) para avisar de que se está ejecutando esta parte del código, mientras que las tres siguientes muestran mensajes de aviso por el puerto serie de que el botón está pulsado y si no se

cancela (soltando el botón) en los próximos diez segundos, se iniciará el proceso de borrado de la EEPROM. La última línea otorga al usuario dicha espera de 10 segundos para soltar el botón y cancelar o mantenerlo pulsado y seguir adelante con el proceso de borrado.

```
if (digitalRead(pulsador) == HIGH) // Si el botón sigue pulsado, inicia el borrado
{
  Serial.println(F("BORRANDO DATOS...\n"));
  digitalWrite(verde, HIGH); // Parpadeo del LED verde
  delay(200);
  digitalWrite(verde, LOW);
  delay(200);
  digitalWrite(verde, HIGH);
  delay(200);
  digitalWrite(verde, LOW);
  for (int x = 0; x < EEPROM.length(); x = x + 1) //Itera a lo largo de toda la EEPROM
  {
    if (EEPROM.read(x) == 0) //Si la dirección x está vacía, continúa
    {
    }
    else
    {
      EEPROM.write(x, 0); // En caso contrario, escribe "0" en dicha dirección
    }
  }
  Serial.println(F("EEPROM borrada correctamente"));
  digitalWrite(verde, LOW); // Apaga los 3 LEDS
  digitalWrite(rojo, LOW);
  digitalWrite(amarillo, LOW);
  delay(200);
}
```

Imagen 3.7: Función setup.

Si el botón sigue pulsado tras la espera de 10 segundos, se entra en este bloque de código, encargado de borrar la EEPROM (Imagen 3.7). En primer lugar se muestra por el puerto serie el mensaje “BORRANDO DATOS…” y a continuación las siete siguientes líneas se encargan de hacer parpadear al led verde dos veces durante 0,2 segundos mientras los leds amarillo y rojo se mantienen encendidos desde el bloque anterior. A continuación, se ejecuta un bucle for que itera a lo largo de todas las direcciones de memoria de la EEPROM (EEPROM.length marca el tamaño de ésta), escribiendo 0 en cada dirección de memoria (dentro del else) salvo que ya contenga un 0 inicialmente (primer if).

Tras abandonar el bucle for, se muestra por el puerto serie el mensaje de que la EEPROM ha sido borrada correctamente, y las últimas 4 líneas se aseguran de apagar todos los LEDs, añadiendo una espera de 0,2 segundos para corroborarlo.

```
  else
  {
    Serial.println(F("Borrado cancelado"));
    delay(200);
    digitalWrite(rojo, LOW); //Parpadeo de los LEDS rojo y amarillo
    digitalWrite(amarillo, LOW);
    delay(200);
    digitalWrite(rojo, HIGH);
    digitalWrite(amarillo, HIGH);
    delay(200);
    digitalWrite(rojo, LOW);
    digitalWrite(amarillo, LOW);
    delay(200);
  }
}
```

Imagen 3.8: Función setup.

En caso de que el pulsador haya sido soltado y, por tanto, el proceso de borrado cancelado, se entra en el bloque else (Imagen 3.8) que muestra por el puerto serie el mensaje “borrado cancelado” y hace parpadear durante 0,2 segundos los LEDs rojo y amarillo que ya estaban encendidos, dejándolos finalmente apagados, y se cierra así este bloque de código, aunque no el setup.

```
// Comprueba si hay una MasterCard definida, y en su caso, deja definirla
// También sirve para redefinir la MasterCard
// Se utiliza 143 como número "mágico", que ha de estar en la dirección
// número 1 de la EEPROM para informar de que existe una MasterCard
```

Imagen 3.9: Función setup.

El siguiente bloque (Imagen 3.10) se encarga de comprobar si hay o no una MasterCard definida, permitiendo definir un tag como MasterCard en caso negativo, o redefinir la MasterCard si se desea.

Para comprobar si existe o no una MasterCard definida, se utiliza la lectura y escritura de un número mágico, en este caso 143, en la dirección número 1 de la EEPROM. Si en dicha dirección no está contenido el número 143, no existirá una MasterCard y se entrará dentro del bloque if.

En primer lugar, se encenderán los leds rojo y verde para indicar que estamos en esta parte del programa, y se mostrará por el puerto serie que no existe una mastercard y que se escanee un tag para definirla.

El bucle while se ejecuta repetidamente hasta que se obtiene una lectura correcta desde el módulo RC522 mediante la función get_ID, que se explicará en su momento.

```
if (EEPROM.read(1) != 143)
{
  Serial.println(F("No existe una MasterCard."));
  digitalWrite(rojo, HIGH); //Enciende los leds rojo y verde
  digitalWrite(verde, HIGH);
  Serial.println(F("Escanear TAG para definirlo como MasterCard..."));
  do
    {
      successRead = getID();      // Es 1 al obtener una lectura válida
    }
    while (!successRead);         // El bucle se repite si el valor sigue siendo 0
    for ( int j = 0; j < 4; j++ ) // Itera 4 veces
    {
      EEPROM.write( 2 + j, readCard[j] );
      // Graba la UID escaneada en la EEPROM, empezando en la dirección número 3
      // Las direcciones 1 y 2 (0 y 1 realmente) están ocupadas
    }
  EEPROM.write(1, 143);           // Escribe el número mágico en la dirección número 1
                                  // Esto informa de que ya existe una MasterCard
  delay(200);
  Serial.println(F("Master Card establecida."));
  digitalWrite(verde, LOW); //Parpadea el led verde
  delay(200);
  digitalWrite(verde, HIGH);
  delay(200);
  digitalWrite(verde, LOW);
  digitalWrite(rojo, LOW);  // Se apagan ambos leds
  delay(200);
}
```

Imagen 3.10: *Función setup.*

Una vez obtenida esta lectura, la UID se almacena provisionalmente en la variable readCard, y mediante un bucle for que itera cuatro veces (cuatro es el número de bloques de la UID a grabar, con dos dígitos hexadecimales por bloque), se graba dicha UID en la EEPROM, a través del comando EEPROM.write, empezando en la tercera dirección de memoria (número 2, ya que la primera posición es la 0), ya que en la dirección 0 se incluirá más adelante un recuento del número de UIDs que hay almacenadas en la EEPROM, y en la dirección número 1 está contenido el número mágico.

Una vez grabada esta UID, se graba el número mágico en la segunda dirección de memoria de la EEPROM para indicar que ya existe una Mastercard establecida, mostrando a continuación esto mismo por el puerto serie, mientras que se hace parpadear el led verde durante 0,2 segundos mientras se mantiene encendido el led rojo, antes de finalmente apagar ambos.

```
  Serial.println(F("-------------------"));
  Serial.println(F("Master Card's UID = "));
  for ( int i = 0; i < 4; i++ ) // Lee la UID de la mastercard de la EEPROM
  {
    masterCard[i] = EEPROM.read(2 + i); // Escribe dicha UID en la variable masterCard
    Serial.print(masterCard[i], HEX);   // Muestra la UID por pantalla
  }

  Serial.println("");
  Serial.println(F("-------------------"));
  Serial.println(F("Todo listo. Esperando el escaneo de tags..."));
  digitalWrite(rojo, LOW);
  digitalWrite(amarillo, HIGH);
  digitalWrite(verde, LOW);
}
```

Imagen 3.11: Función setup.

Seguidamente (Imagen 3.11), se muestra la UID de la Mastercard por el puerto serie, leída mediante un bucle for que iterará la lectura de la EEPROM mediante la función EEPROM.read cuatro veces a partir de la tercera dirección de memoria (posición 2), almacenando cada bloque de información en la variable masterCard, que será la que se muestre por el puerto serie con los datos hexadecimales ya transformados.

Finalmente, se muestra el mensaje “Todo listo. Esperando el escaneo de tags…” y se deja encendido únicamente el led amarillo, para mostrar que el programa se encuentra en estado de espera.

```
// Comprobar si esta programado el encendido
bool isScheduledON(DateTime date)
{
  int weekDay = date.dayOfTheWeek();
  float hours = date.hour() + date.minute() / 60.0;

  // De 09:00 a 21:00
  bool hourCondition = (hours > 9 && houras < 21);

  // De lunes a viernes
  bool dayCondition = (weekDay == 1 || weekDay == 2 || weekDay == 3 || weekDay == 4 || weekDay == 5);
  if (hourCondition && dayCondition)
  {
    return true;
  }
    return false;
}
```

Imagen 3.12: Función setup.

Este bloque de código (Imagen 3.12) es fundamental, ya que es el que se encarga de controlar si el programa debe ejecutarse o no según el día y la hora que sea.

Se define una función de tipo booleano denominada “isScheduleON”, a la que se le pasará un dato en formato fecha (DateTime).

Dentro de la propia función se crea un parámetro de tipo entero llamado weekDay (día de la semana) en el que se obtiene desde el parámetro fecha en qué día nos encontramos, y una variable de tipo float que incluya la hora y los minutos del momento correspondiente.

Se añaden dos condiciones de tipo bool, una condición horaria (hourCondition) y una condición diaria (dayCondition).

Para la condición horaria se ha establecido un rango de funcionamiento desde las 9 hasta las 21 horas, mientras que como condición diaria se ha fijado que el programa funcione de lunes a viernes (weekDay de 1 a 5). En caso de que se cumplan ambas condiciones, la función devolverá un valor de "true", mientras que en caso de no cumplirse alguna o ninguna de ellas, devolverá un valor "false".

3.2.3. MAIN LOOP

La función void loop es una función por defecto en Arduino que representa la función principal, la que se ejecuta repetidamente de manera indefinida.

El código de esta función se muestra en las imágenes de la 3.13 a la 3.19.

```
/////////////////////////////////////// Main Loop ///////////////////////////////////
void loop () {
  do{
    successRead = getID();  // 1 si se obtiene una lectura correcta
    }
  while (!successRead);     // Se repite hasta que haya una lectura
  if (programMode) {        // Si ya se está en el modo programa
    if ( isMaster(readCard) ) { // Si se escanea la Mastercard se sale del modo Programa
      Serial.println(F("Master Card escaneada."));
      Serial.println(F("Saliendo del modo programa..."));
      Serial.println(F("-----------------------------"));
      programMode = false;
      digitalWrite(rojo, LOW);
      digitalWrite(verde,LOW);
      digitalWrite(amarillo, HIGH); // Enciende el led amarillo de espera
      return;
    }
```

Imagen 3.13: void loop.

El programa se mantiene en espera hasta que se escanea algún tag (Imagen 3.13). Si la ejecución se encuentra dentro del denominado modo programa (programMode, para lo cual hay que escanear la MasterCard) se podrá manipular la información contenida en la EEPROM, añadiendo nuevos tags o borrando los ya existentes para así controlar sus permisos de acceso o no.

Para salir del modo programa se debe volver a escanear la MasterCard. Esto delverá un valor de “false” para programMode y encenderá el led amarillo que representa el estado de espera de nuevas tarjetas.

```
else {
  if ( findID(readCard) ) { // Si la UID es conocida, la borra
    digitalWrite(verde, HIGH);  // Led verde encendido
    digitalWrite(rojo, HIGH);
    delay(200);
    Serial.println(F("UID conocida. Borrando..."));
    digitalWrite(rojo, LOW);    // Parpadea el led rojo
    delay(200);
    digitalWrite(rojo, HIGH);
    delay(200);
    digitalWrite(rojo, LOW);
    delay(200);
    digitalWrite(verde, LOW);
    deleteID(readCard); // Borra la tarjeta de la EEPROM
    Serial.println("-----------------------------");
  }
```

Imagen 3.14: void loop.

```
else {                    // Si la UID no es conocida, la añade
  digitalWrite(verde, HIGH);
  digitalWrite(rojo, HIGH);  // Led rojo encendido
  delay(200);
  Serial.println(F("UID nueva, añadiendo..."));
  digitalWrite(verde, LOW);  // Parpadea el led verde
  delay(200);
  digitalWrite(verde, HIGH);
  delay(200);
  digitalWrite(verde, LOW);
  delay(200);
  digitalWrite(rojo, LOW);
  writeID(readCard); // Añade la UID escaneada a la EEPROM
  Serial.println(F("-----------------------------"));
}}}
```

Imagen 3.15: void loop.

Mientras la ejecución del programa se mantenga dentro del modo programa, al escanear un tag que no sea la MasterCard, se comprueba si su UID está contenida en la EEPROM (bloque if) o no (bloque else) mediante la función findID (se explicará posteriormente como el resto de funciones).

SI la UID ya estaba contenida en la EEPROM, el programa la borra mediante la función deleteID (Imagen 3.14), representado este proceso por

un breve parpadeo del led rojo mientras el verde se encuentra encendido y el amarillo apagado, lo que señaliza un borrado exitoso.

Si la UID no se encontraba en la EEPROM, la añade mediante la función writeID (Imagen 3.15), señalizado esta vez por un parpadeo del led verde mientras es ahora el rojo el que se mantiene fijo, con el amarillo igualmente apagado.

```
else {
  if ( isMaster(readCard) ) {    // Si se escanea la MasterCard...
    programMode = true;          // Entra en el modo programa
    Serial.println(F("MasterCard reconocida. Entrando en el modo programa..."));
    digitalWrite(rojo, HIGH);
    digitalWrite(verde, HIGH);
    digitalWrite(amarillo, LOW);
    int count = EEPROM.read(0);   // Lee el primer byte de la EEPROM donde se
    Serial.print(F("Existen "));   // encuentra el número de UIDs que contiene la EEPROM
    Serial.print(count);
    Serial.print(F(" UIDs en la memoria"));
    Serial.println("");
    Serial.println(F("Escanear un tag para añadirlo o borrarlo..."));
    Serial.println(F("-----------------------------"));
  }}
```

Imagen 3.16: void loop.

La parte del código detallada en la Imagen 3.16 ejecuta lo explicado anteriormente, el registro de la MasterCard para acceder al modo programa si la ejecución no se encuentra ya en él, dejando encendidos los leds rojo y verde para hacer saber que se está en este modo.

Muestra, además, el número de UIDs diferentes que la EEPROM tiene almacenadas. Dicho número se encuentra en la posición 0 de memoria de la EEPROM, que es el valor que se muestra por el puerto serie.

A partir de aquí se pueden escanear tags para añadir a la EEPROM o borrarlos de ella como se ha explicado anteriormente.

```
DateTime now = rtc.now(); // Fecha actual

if (estado == false && isScheduledON(now))      // Apagado y debería estar encendido
{
  estado = true;
}
else if (estado == true && !isScheduledON(now))  // Encendido y deberia estar apagado
 {
    estado = false;
    Serial.println("Saliendo...");
    delay(200);
    digitalWrite(amarillo, LOW);
    digitalWrite(verde, LOW);
    digitalWrite(rojo, LOW);
    delay(200);
 }
```

Imagen 3.17: void loop.

En la Imagen 3.17 aparece el código destinado a comprobar la programación horaria del sketch, haciéndolo funcionar si debe estar funcionando y no lo está ya y viceversa, a partir de los valores de hora y fecha actuales obtenidos con la línea rtc.now().

Si el programa no está funcionando, otorga el valor “false” a la variable “estado” (la que almacena si el programa ha de estar funcionando, true, o no, false), imprime “Saliendo…” por el puerto serie y apaga todos los leds.

```
if (estado){ // Si Arduino está funcionando
  digitalWrite(rojo, LOW);
  digitalWrite(verde, LOW);
  digitalWrite(amarillo, HIGH); //En espera
  delay(200);
  else
  {
  Serial.print(F("CARD's UID: "));
  for (int i = 0; i < 4; i++) {  // Itera a lo alrgo del tamaño de la UID
  readCard[i] = mfrc522.uid.uidByte[i];
  Serial.print(readCard[i], HEX);// La imprime por pantalla
  }
  Serial.println("");
```

Imagen 3.18: void loop.

En caso de que el programa de control de acceso deba estar funcionando (estado = true), se entra en el bloque if correspondiente (Imagen 3.18), se enciende el led amarillo de espera y se muestra la UID del tag correspondiente que se escanee.

```
if(findID(readCard)){ // Si la tarjeta leída actualmente coincide con una almacenada en la EEPROM
  Serial.println("Acceso concedido");
  //Enciende el LED verde mientras esté abierto el relé, se asegura de apagar los demás y al final apaga los tres
  delay(200);
  digitalWrite(rojo, LOW);
  digitalWrite(amarillo, LOW);
  digitalWrite(verde, HIGH);
  // delay(200);
  digitalWrite(relay, HIGH);     // Cambia el valor del relé y abre la puerta
  delay(5000);                   // Deja la puerta abierta 5 segundos
  digitalWrite(relay, LOW);      // Vuelve a cerrar la puerta
  // delay(200);
  digitalWrite(verde, LOW);
  //delay(2000);
  digitalWrite(amarillo, HIGH);  // En espera
  // delay(2000);
}
else{
  Serial.println("Acceso denegado"); /* Si no se permite el acceso a la tarjeta actual,
                                    el valor del relé no cambia ni se abre la puerta
                                    Enciende el LED rojo de acceso denegado durante
                                    2,5 segundos y después lo apaga.
                                    También se asegura de que los demás estén apagados */
  delay(200);
  digitalWrite(rojo, HIGH);
  digitalWrite(verde, LOW);
  digitalWrite(amarillo, LOW);
  delay(2500);
  digitalWrite(rojo, LOW);
  delay(200);
  digitalWrite(amarillo, HIGH);     // En espera
}}} // Fin del Loop
```

Imagen 3.19: void loop.

Esta es la parte que se encarga del control de acceso como tal (Imagen 3.19). Compara la UID escaneada con las que se encuentran almacenadas en la EEPROM mediante la función findID, y en caso de coincidencia, permite el acceso, mientras que si la UID no se encuentra en la EEPROM, lo restringe.

En caso de que el acceso esté permitido, muestra dicho mensaje por el puerto serie a la vez que enciende únicamente el led verde y abre el relé, que a su vez se encargará de abrir el circuito externo, por ejemplo, una puerta, durante 5 segundos, luego lo vuelve a cerrar, apaga el led verde y vuelve a dejar encendido el led amarillo de espera.

Si el acceso no está permitido, muestra por el puerto serie el mensaje "Acceso denegado" y enciende la luz roja durante 2,5 segundos, dejando finalmente también encendido únicamente el led amarillo de espera, y finalizando así la función main loop.

3.2.4. GetID

La función denominada getID() es la encargada de procesar las lecturas del módulo RC522, y su código se muestra en la Imagen 3.20.

```
////////////////////////////////////// Obtención de la UID ////////////////////.
int getID() {
  // A la espera de nuevas tarjetas
  if ( ! mfrc522.PICC_IsNewCardPresent()) { // Detecta si existe un tag
    return 0;
  }
  if ( ! mfrc522.PICC_ReadCardSerial()) {   // Lee el tag por el puerto serie
    return 0;
  }
  // Las UIDs tienen 4 bytes, se itera a lo largo de ellos
  for (int i = 0; i < 4; i++) {
    readCard[i] = mfrc522.uid.uidByte[i]; // almacena la UID leida
  }
  mfrc522.PICC_HaltA(); // Termina la lectura
  return 1; // Lectura correcta
}
```

Imagen 3.20: *función getID.*

Esta función trabaja a su vez con funciones incluidas en la librería MFRC522, las cuales son PICC_IsNewCardPresent, que detecta si existe un tag cerca del módulo lector, PICC_ReadCardSerial, que lee la UID de dicho tag y la manda al puerto serie y PICC_HaltA, que termina la lectura. La UID leída se almacena en la variable readCard y si la lectura ha sido correcta la función devuelve un 1.

3.2.5. ReadID

```
//////////////////////////////////////// Lee una UID de la EEPROM ///////////////
void readID( int number ) {
  int start = (number * 4 ) + 2;    // Lugar de comienzo
  for ( int i = 0; i < 4; i++ ) {     // Itera 4 veces para obtener los 4 Bytes
    storedCard[i] = EEPROM.read(start + i);   // Asigna valores a la variable
  }
}
```

Imagen 3.21: función readID.

La función readID (Imagen 3.21) se emplea para leer UIDs almacenadas en la EEPROM, calculando el lugar desde que debe comenzar a leer y asignando dicha UID a la variable storedCard.

3.2.6. writeID

La función writeID (Imagen 3.22) se encarga de agregar nuevas UIDs a la memoria EEPROM.

```
//////////////////////////// Añade una UID a la EEPROM   ///////////////////////////////////
void writeID( byte a[] ) {
  if ( !findID( a ) ) {     // Comprueba si la UID ya existe
    int num = EEPROM.read(0);     // La posición 0 almacena el número de UIDs existentes
    int start = ( num * 4 ) + 6;  // Calcula donde comenzar
    /*Empieza en la posicion (num + 6) porque 0 y 1 están ocupadas,
    y en las cuatro siguientes estará almacenada la UID de la MasterCard*/
    num++;
    EEPROM.write( 0, num );           // Actualiza el total de UIDs
    for ( int j = 0; j < 4; j++ ) {     // Itera 4 veces
      EEPROM.write( start + j, a[j] );  // Escribe la UID a la EEPROM
    }
  Serial.println(F("LA UID se añadió correctamente."));
  }
  else {
  Serial.println(F("Hubo un error al añadir la UID!"));
  }
}
```

Imagen 3.22: función writeID.

En primer lugar, comprobará si la UID escaneada ya existe en la EEPROM mediante la función findID, en caso negativo, procede a escribirla, calculando en primer lugar donde debe empezar a escribirla (ayudándose de la posición 0 de memoria que almacena cuantas UIDs existen ya en la EEPROM), procediendo posteriormente a la propia escritura y actualizando el valor de UIDs totales que existen en la EEPROM.

Además, muestra un mensaje por el puerto serie sobre si el procedimiento ha sido exitoso o si ha habido un error.

3.2.7. DeleteID

La función deleteID (Imagen 3.23) se encarga de borrar una UID ya existente en la EEPROM.

```
////////////////////////////// Borrar UID de la EEPROM   ///////////////////////////////////
void deleteID( byte a[] ) {
  if ( !findID( a ) ) {     // Comprueba si la UID ya existe
      // En caso contrario
  Serial.println(F("Hubo un error al borrar la UID!"));
  }
  else { // Si ya existe
    int num = EEPROM.read(0);   // Número de espacios utilizados
    int slot;       // Espacio de la UID
    int start;      // Posición de comienzo del siguiente espacio
    int looping;    // Iteraciones
    int j;
    int count = EEPROM.read(0); // Número de UIDs existentes
    slot = findIDSLOT( a );     // Espacio de la UID a borrar
    start = (slot * 4) + 2;     // Comienzo
    looping = ((num - slot) * 4);
    num--;      //
    EEPROM.write( 0, num );   // Escribe 0 en la posición debida
    for ( j = 0; j < looping; j++ ) { // Itera
      EEPROM.write( start + j, EEPROM.read(start + 4 + j)); // Para no dejar espacios vacíos
      // tras el borrado, adelanta 4 posiciones las UIDs almacenadas en la EEPROM
    }
    for ( int k = 0; k < 4; k++ ) {         // Deja las últimas posiciones vacías
      EEPROM.write( start + j + k, 0);
    }

  Serial.println(F("La UID se borró correctamente."));
  }
}
```

Imagen 3.23: función deleteID.

En primer lugar, la función comprueba si la UID existe actualmente en la EEPROM, en caso contrario muestra un mensaje de error, mientras que en caso de coincidencia sigue su ejecución.

La función averigua la posición de la tarjeta a borrar, e itera el proceso de borrado (escritura del valor '0' en la EEPROM) a lo largo del tamaño de la UID, procediendo posteriormente a adelantar las UIDs que existían a continuación de la UID borrada, y muestra un mensaje por el puerto serie de que el proceso ha funcionado exitosamente.

3.2.8. checkTwo

La función checkTwo (Imagen 4.23) compara dos cadenas de bytes (dos UIDs) para ver si existe coincidencia entre ellas o no.

```
//////////////////////////////////// Comparativa //////////////////////
boolean checkTwo ( byte a[], byte b[] ) {
  if ( a[0] != NULL ) // Comprueba que la cadena no está vacía
    match = true;     // Asume coincidencia al principio
  for ( int k = 0; k < 4; k++ ) {   // Itera
    if ( a[k] != b[k] )   // Si algún elemento no coincide...
      match = false;      // la coincidencia será falsa
  }
  if ( match ) {    // Si aún coinciden...
    return true;    // La coincidencia devuelve true
  }
  else  {
    return false;   // En caso contrario, false
  }
}
```

Imagen 3.24: función checkTwo.

En primer lugar se asegura de que la cadena no esté vacía y asume que ambas cadenas coinciden, posteriormente procede a iterar a lo largo de las cadenas para ver si coinciden, en caso positivo, devuelve “true”, mientras que si no coincidiesen, devuelve un valor “false”.

3.2.9. findIDslot

```
//////////////////////////////////// Encontrar Slot   ////////////////////////////////////
int findIDSLOT( byte find[] ) {
  int count = EEPROM.read(0);       // Número de UIDs existentes
  for ( int i = 1; i <= count; i++ ) {    // Itera
    readID(i);                      //Lee UID de la EEPROM (storedCard[4])
    if ( checkTwo( find, storedCard ) ) {   /* Compara si la UID leída de la EEPROM,
    storedCard, es la misma que la UID escaneada */
      return i;       // Posición de la UID
      break;
    }
  }
}
```

Imagen 3.25 función findIDslot.

La función findIDslot (Imagen 3.25) devuelve la posición de la UID buscada, comparando la UID que se escanea con las existentes de la EEPROM, y tras conseguirlo se detiene.

3.2.10. FindID

La función findID (Imagen 3.26) se encarga de verificar si la UID de la tarjeta que se escanea existía ya previamente en la EEPROM.

```
///////////////////////////Busca UID en la EEPROM   ////////////////////////////////
boolean findID( byte find[] ) {
  int count = EEPROM.read(0);   // Número de UIDs
  for ( int i = 1; i <= count; i++ ) { // Itera
    readID(i);          // Lee UID de la EEPROM, almacenada en storedCARD
    if ( checkTwo( find, storedCard ) ) {   // Busca la UID escaneada en la EEPROM
      return true; // True si la encuentra y se detiene
      break;
    }
    else {
    }
  }
  return false; // False si no existe tal UID en la EEPROM
}
```

Imagen 3.26: función findID.

Para ello, compara la UID que se le pasa a la función con las existentes en la EEPROM, almacenando cada UID leída de la EEPROM en la variable storedCARD y comparándola mediante la función checkTwo con la UID escaneada, devolviendo "true" si esta ya estaba almacenada en la memoria o "false" si no se encuentra.

3.2.11. isMaster

Finalmente, la última función necesaria para el correcto funcionamiento del programa es la llamada "isMaster" (Imagen 3.27), encargada de comprobar si la UID escaneada es la UID de la MasterCard.

```
////////////////////// Comprobar si es la MasterCard   ////////////////////////////////
boolean isMaster( byte test[] ) { // Comprueba si la UID pasada coincide con la MasterCard
  if ( checkTwo( test, masterCard ) )
    return true;
  else
    return false;
}
```

Imagen 3.27: función isMaster.

Para ello, la función recibe como argumento una UID, y ayudándose de la función comparativa checkTwo, verifica si dicha UID coincide con la almacenada en la variable masterCard o no, devolviendo "true" en caso de coincidencia o "false" en caso contrario.

CAPÍTULO 4:

SOFTWARE EMPLEADO Y PROGRAMACIÓN DEL PROYECTO: PYTHON

En la actualidad existe una gran cantidad de lenguajes de programación de diversos tipos (lenguajes compilados, lenguajes interpretados…), entre los que actualmente destacan, entre otros, lenguajes como C/C++, Java, Ruby o PHP. El lenguaje escogido para el desarrollo del presente proyecto es Python, sobre el cual se profundizará en este capítulo, así como en las funciones que llevará a cabo esta parte del proyecto.

4.1. LENGUAJE PYTHON

En primer lugar habrá que dar respuesta a la pregunta inicial: "¿qué es Python?".

De acuerdo a la documentación de su página web oficial (Python 2.7.15 Documentation, 2018), Python es un lenguaje interpretado, interactivo, dinámico y orientado a objetos, que combina potencia y una sintaxis muy clara, intentando favorecer la legibilidad del código. Es además extensible a C o C++ y multiplataforma, funciona en Mac, Windows y Linux.

Guido van Rossum es el principal creador e impulsor de Python, y el mismo explica por qué tuvo la necesidad de crear este lenguaje en un principio. Se necesitaba un sustituto del lenguaje ABC que funcionase en el proyecto Amoeba mientras trabajaba en el CWI (Centrum Wiskunde & Informatica, Holanda).

Guido desarrolló gran parte de este lenguaje entre 1989 y 1990, publicándolo finalmente en 1991.

Como dato curioso, señalar que el nombre de este lenguaje se debe a la admiración del autor, van Rossum, por los Monty Python y su serie de televisión "Monty Python's Flying Circus".

Algunas de las ventajas de Python son que es un lenguaje de alto nivel y propósito general, que puede ser aplicado a diversas clases de problemas y que incluye un gran número de librerías estándar aplicables a una gran variedad de áreas.

Es, además, un buen lenguaje para usuarios que se estén iniciando en la programación, en gran parte debido a su sintaxis, más simple e intuitiva que la de la mayoría de los lenguajes de programación más utilizados, favoreciendo así que el aprendizaje se centre en otras habilidades de programación más relevantes como aprender a "pensar como un programador", la descomposición de problemas o el uso de estructuras típicas como bucles. También influyen la anteriormente mencionada existencia de gran variedad de librerías estándar, y el hecho de que sea un lenguaje interpretado.

Se puede reseñar también la existencia de una guía de estilo para la programación en lenguaje Python, denominada PEP 8, y que podemos encontrar en Python Software Fundation (2018).

Por otra parte, para instalar Python no hay más que descargarlo, de manera gratuita, desde su sitio web oficial y proceder a los pasos recomendados para la instalación en nuestro ordenador.

Respecto a la comparativa con otros lenguajes de programación, Python suele compararse con otros como Java, Javascript, o C++. Guido van Rossom hizo una comparativa con varios de estos lenguajes, centrándose únicamente en sus características, destacando las siguientes:

4.1.1. PYTHON VS JAVA:

Mientras que los programas en Java se ejecutan a mayor velocidad que los programas desarrollados en Python, estos últimos tardan mucho menos tiempo en desarrollarse (entre 3 y 5 veces menos), ya que es un lenguaje de alto nivel más compacto y dinámico y, por ejemplo, sus variables no necesitan ser declaradas (se puede hacer sobre la marcha). Esto, a la hora de la simulación, ralentiza el programa ya que necesitará inspeccionar primero cada variable, su contenido y su tipo, dado que en la compilación inicial no se definen.

Java es mejor, por tanto, como lenguaje de implementación a bajo nivel, aunque la combinación de ambos suele arrojar buenos resultados, véase como ejemplo la utilización de componentes desarrollados en Java combinados para formar aplicaciones en Python. Se puede considerar a Python como lo que en la jerga programadora se conoce comúnmente como “lenguaje pegamento”, destinado a unir componentes de software.

Existe también, de hecho, un lenguaje denominado “Jython”, basado en Python e implementado directamente en Java, lo que ayuda a hacerse a la idea de la buena interacción de ambos lenguajes.

4.1.2. PYTHON VS JAVASCRIPT

Tanto el lenguaje Python como el lenguaje Javascript están basados en objetos. Otra similitud entre ambos es que, a diferencia de Java, ambos tienen un estilo de programación que permite utilizar funciones simples y variables sin necesidad de definir clases. Pero, a diferencia de JavaScript, que no va más allá, Python soporta también programas mucho más largos y reutilizables con un estilo verdaderamente orientado a objetos, donde clases y herencias ya intervienen con un papel importante.

4.1.3. PYTHON VS C++

Las similitudes enunciadas en la comparativa con Java son muy similares a las existentes entre Python y C++. La diferencia entre el tiempo de desarrollo de los programas en Python respecto a los programas en C++ es incluso mayor a la que existía respecto a los programas en Java ,tardándose mucho más en desarrollar un programa equivalente en lenguaje C++. De igual manera, Python también funciona como un excelente "lenguaje pegamento" utilizado para combinar componentes escritos en C++, mientras que C++, por su parte, no es un lenguaje interpretado sino compilado.

4.1.4. PYTHON VS C

En Python vs C (2018) podemos encontrar las principales diferencias entre ambos lenguajes, que se relatan a continuación.

La diferencia principal entre estos dos lenguajes es que, mientras que Python es un lenguaje multiparadigma, C es un lenguaje estructurado.

El lenguaje C es un lenguaje de alto nivel base para otros lenguajes como Java o el propio Python. Es, al igual que C++, un lenguaje compilado, trasformándose todo el código fuente a lenguaje máquina, lo cual facilita su entendimiento por parte del equipo, y por ello es más rápido, mientras que Python, como ya se ha señalado anteriormente, es un lenguaje intérprete.

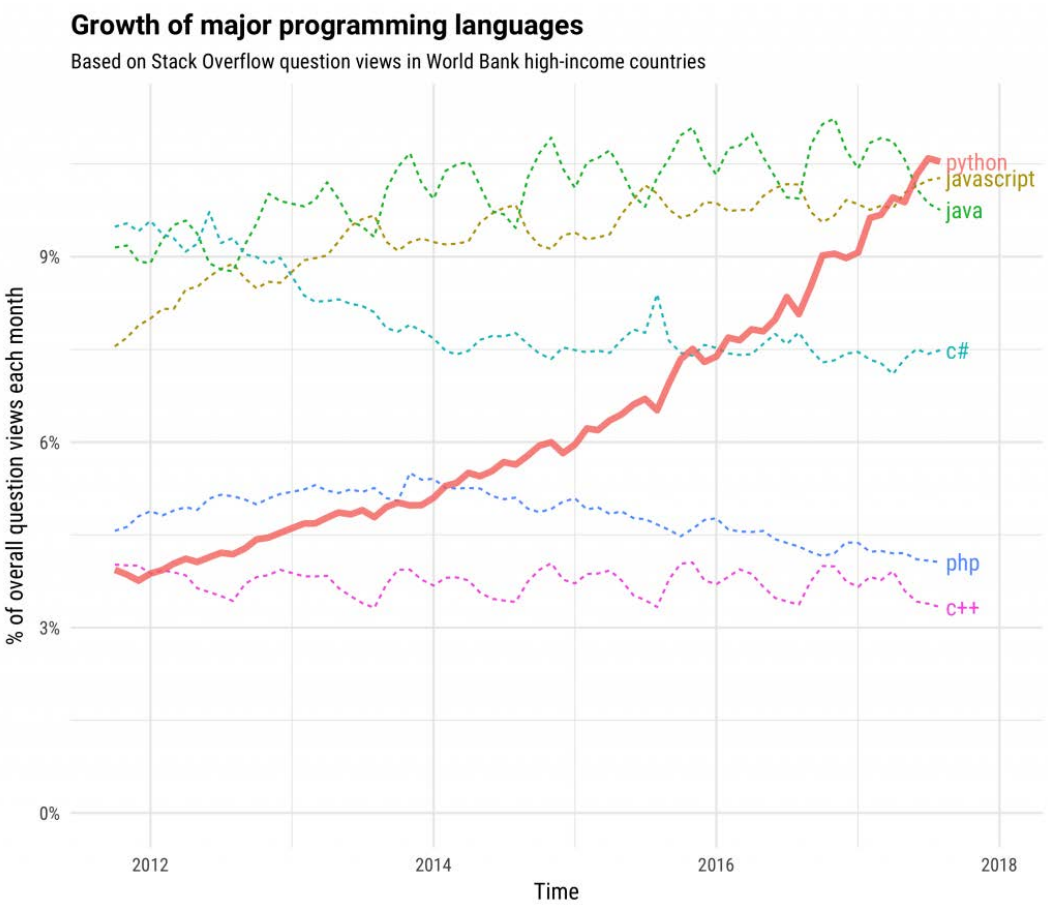
Otra de las características a señalar es la sencillez, siendo Python mucho más simple e intuitivo que C.

4.1.5. EL CRECIMIENTO DE PYTHON EN LA ACTUALIDAD

En los últimos cinco años el lenguaje Python ha experimentado un gran crecimiento, siendo clasificado como el lenguaje de programación con un crecimiento más rápido en la actualidad por parte del sitio web de programación stackoverflow, uno de los lugares de encuentro entre desarrolladores más importantes que existen, con una enorme comunidad que intercambia grandes cantidades de información sobre programación cada día.

En Robinson (2017) podemos encontrar un artículo completo relacionado con el tema, en el que destaca el crecimiento exponencial de las visitas a preguntas relacionadas con Python a lo largo de los últimos años provenientes de los países con mejor situación económica (que suponen un 64% del tráfico dentro de la web stackoverflow), culminando en junio de 2017, cuando Python fue por primera vez el lenguaje más visitado dentro del portal.

En la gráfica 4.1, obtenida del propio artículo, se representan los datos de visitas por lenguaje y año:

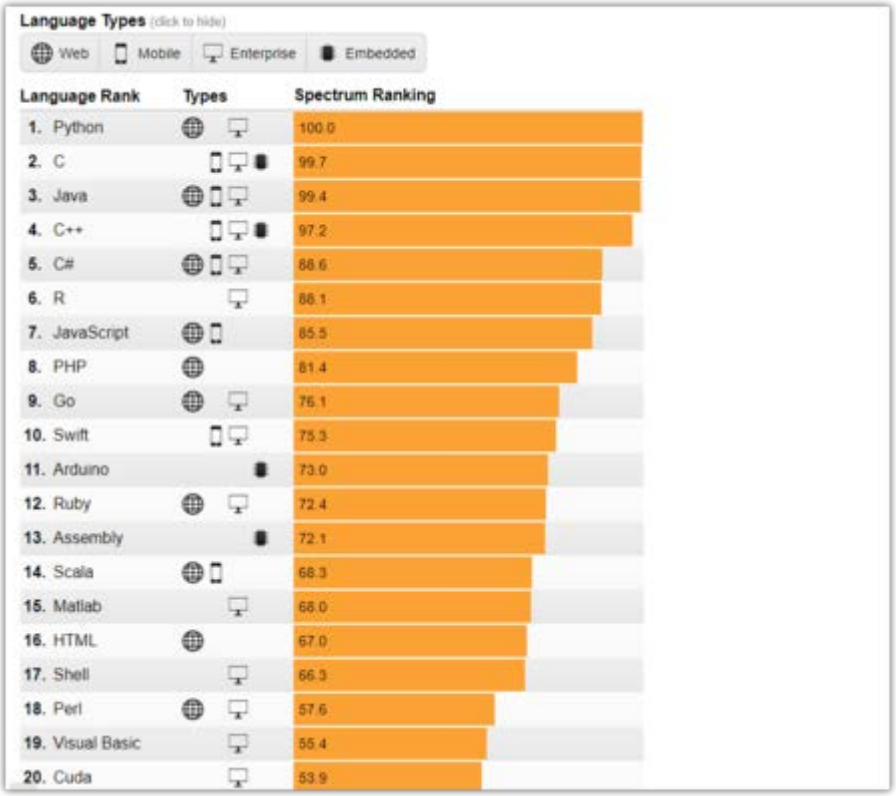


Gráfica 4.1: *Visitas por lenguaje y año en stackoverflow.*

Fuente: Robinson, 2017.

En cuanto al resto de países, el crecimiento de Python también es significativo, únicamente comenzó algo más tarde que en los denominados "high-income countries", o países de altos ingresos.

Para finalizar, cabe señalar que la revista especializada en ingeniería IEEE Spectrum, recogiendo múltiple información de GitHub, Google y diversos sitios web dedicados a la programación como el propio stackoverflow o Reddit, también consideró a Python como el lenguaje de programación más importante en el año 2017 (Gráficas 4.2).



Gráfica 4.2: *Lenguajes de programación más importantes para IEEE Spectrum.*

Fuente: Diakopoulos & Cass, 2017.

4.2. IDE ECLIPSE JAVA OXYGEN

Para la programación de la parte escrita en lenguaje Python se utiliza un entorno de desarrollo integrado, IDE (Integrated Development Environment), en concreto el IDE Eclipse.

Un entorno de desarrollo integrado es una herramienta informática que proporciona servicios integrales para facilitar el desarrollo de software, o, entendido de otra manera, un conjunto de programas que usan una interfaz única para todos ellos (Sistemas – Definición de IDE (n.d.).). En otras palabras y sin entrar en detalles, es principalmente un entorno de programación.

El entorno de desarrollo integrado Eclipse es de código abierto, multiplataforma y posee una gran versatilidad.

Sus orígenes se sitúan en el año 2001, en noviembre, donde fue creado por IBM. En 2004 nace la Fundación Eclipse, organización independiente y sin ánimo de lucro, encargada de manejar Eclipse, cuya intención es fomentar una comunidad abierta y transparente alrededor de sus productos (The Eclipse Foundation, n.d.).

Una de las principales características de este IDE es la posibilidad de instalar distintos plug-ins (por ejemplo, para desarrollo en Java o en C/C++) y poder trabajar interaccionando con todos ellos.

En este caso, para poder trabajar con Python, además de descargar inicialmente el IDE Eclipse de manera gratuita desde su sitio web oficial, así como la última versión de Java, es necesario instalar un complemento denominado PyDev, siguiendo las instrucciones explicadas paso a paso en PyDev (2018). Una vez instalado, Eclipse nos dará la opción de crear proyectos e instalar módulos asociados a Python.

4.3. CÓDIGO EN PYTHON

En este apartado se explicarán las funciones del proyecto correspondientes a la parte de programación en lenguaje Python.

Esta será la parte visible de la programación del proyecto, la que se ejecute en la computadora correspondiente y a través de la cual se interactuará con el prototipo creado en la placa Arduino.

La parte de programación perteneciente a Python se encarga de hacer de “emisario” o “intérprete” entre las funciones de la parte programada en Arduino y sistema informático correspondiente.

El programa, básicamente, lee la información que envía Arduino por el puerto serie y trabaja con ella. Además, está programada su ejecución para que se comience diariamente a las 8:55 A.M.

La información que recibe el programa es principalmente la de la parte del control de acceso de Arduino, constando de la UID correspondiente a cada tag así como sus permisos de acceso. El programa Python trabaja con dicha información y la escribe en un archivo Excel determinado (la ruta de dicho archivo ha de especificarse dentro del código) para su registro (objetivo que se marcaba al inicio de este proyecto), así como para poder trabajar con dichos datos y analizarlos, en su caso.

El programa crea un archivo Excel mensual, cuyas hojas son los diversos días del mes, y en las que no sólo registra la UID y si su acceso está permitido o no, sino que también registra la fecha y hora exactas en la que el tag correspondiente es escaneado, así como si está entrando o saliendo del sistema (o si tiene el acceso denegado, en su caso).

Estos archivos Excel hacen la función de base de datos que queda como resultado final del proyecto, pudiendo ser de gran valor, no sólo en cuanto al cumplimiento de la normativa en vigor o la posible normativa entrante, sino en cuanto al análisis de gran cantidad de información,

recopilada además en tiempo real, para detectar debilidades e implementar diferentes mejoras, suponiendo un paso más hacia la automatización de la industria.

Parece importante aclarar el origen de ciertas librerías de las cuales depende el correcto funcionamiento de todo el programa. Estas son: datetime, openpyxl, schedule, serial, time. Además, para la instalación de estas librerías se empleó el sistema de gestión de paquetes denominado "pip", incluido en Python, y gracias al cual para gran cantidad de paquetes de software como en concreto estas librerías, es suficiente con escribir en la consola del ordenador correspondiente, dentro del directorio Python, las líneas "pip install" seguidas del nombre de la librería deseada.

- Librerías time y datetime: estas librerías incluyen funciones encargadas del manejo del tiempo en Python, ya sean fechas u horas. En el programa se emplean no sólo para registrar el momento en el que una UID es escaneada, si no para darle nombre a los archivos Excel así como para la programación horaria de la ejecución del programa. La información perteneciente a la librería datetime así como sus funciones más destacadas se pueden encontrar dentro de la propia web de Python. La librería time, además, suele venir instalada por defecto dentro de Python.

- Librería openpyxl: es la librería encargada de trabajar con archivos Excel desde Python. Toda la información relativa a esta librería se puede encontrar en openpyxl - A Python library to read/write Excel 2010 xlsx/xlsm files (2018).

  Esta librería se emplea dentro del código para crear, abrir, manipular (escritura, estilo…) y finalmente guardar los archivos Excel correspondientes a la ejecución del programa.

- Librería schedule: es la encargada de controlar la programación horaria del proyecto. Toda la información relativa a esta librería se puede encontrar en The Python Package Index – Schedule (2018). Básicamente, se define una función principal y valiéndose de esta librería así como de las encargadas de controlar el tiempo, se establece un plazo de repetición para la ejecución de dicha función, en nuestro caso de lunes a viernes todos los días a las 8:55 A.M.

  Sin embargo, existe un aspecto de esta función que necesita corrección, y es que si se ejecuta el programa en un momento en el que ya debiese de estar funcionando, esta librería no lo

detecta por sí misma hasta la programación perteneciente al día siguiente, por lo que se ha establecido una función que si ocurre este caso concreto, y el programa ha de estar funcionando en el momento de ejecutar el código, programe su arranque justo para el minuto siguiente, ya que no es posible programarlo exactamente para el momento en el que se comienza a ejecutar.

- Librería serial: es la encargada de establecer y controlar la comunicación por el puerto serie entre Arduino y Python, leyendo y enviando datos en su caso. Para iniciar la comunicación habrá que establecer como mínimo el puerto utilizado así como la velocidad en baudios, que habrá de ser la misma en ambas interfaces (Llamas, 2018).

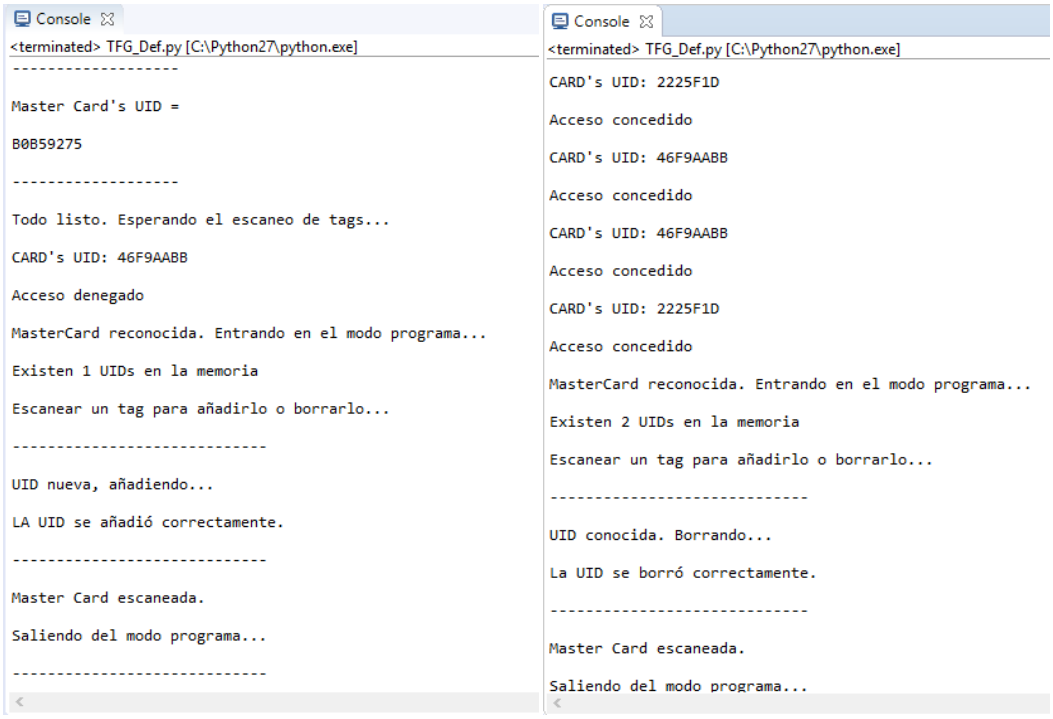
4.4. PRESENTACIÓN DE RESULTADOS

Para comprobar el correcto funcionamiento del programa, se realiza una prueba escaneando diversos tags con diferentes permisos, incluida la MasterCard.

En las imágenes 4.1 y 4.2 se muestra el resultado que arroja la consola en la parte de programación en Python dentro del IDE Eclipse al llevar a cabo dicha prueba.

Esta prueba genera a su vez un archivo Excel con los datos del registro correspondiente, incluyendo UID, fecha y hora de registro, permisos y tipo de acceso, llamado, en este caso, al haberse realizado dicha prueba en el mes de julio, “July.xlsx”. Este fichero se guardará en la ruta especificada dentro del código en Python.

Las imágenes 4.3 y 4.4 corresponden el archivo Excel generado durante la prueba.



Imágenes 4.1y 4.2: *resultados del ejemplo en la consola de Python.*

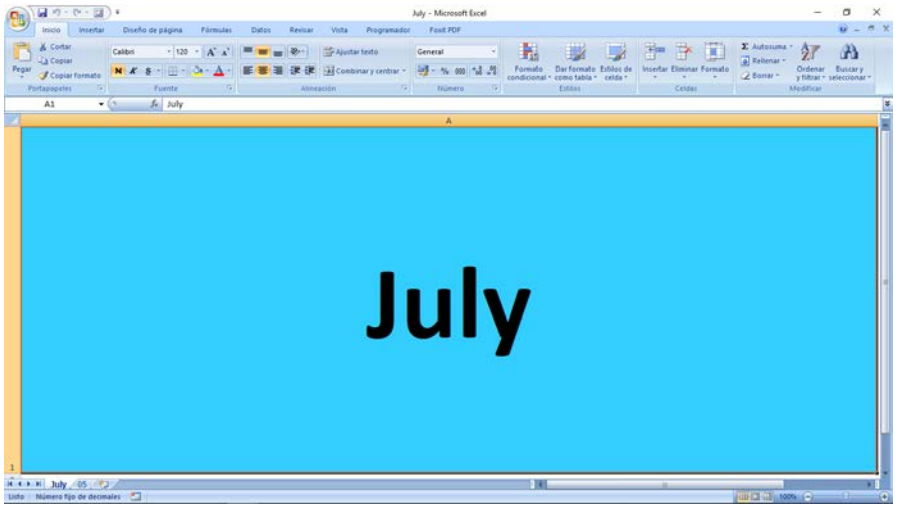
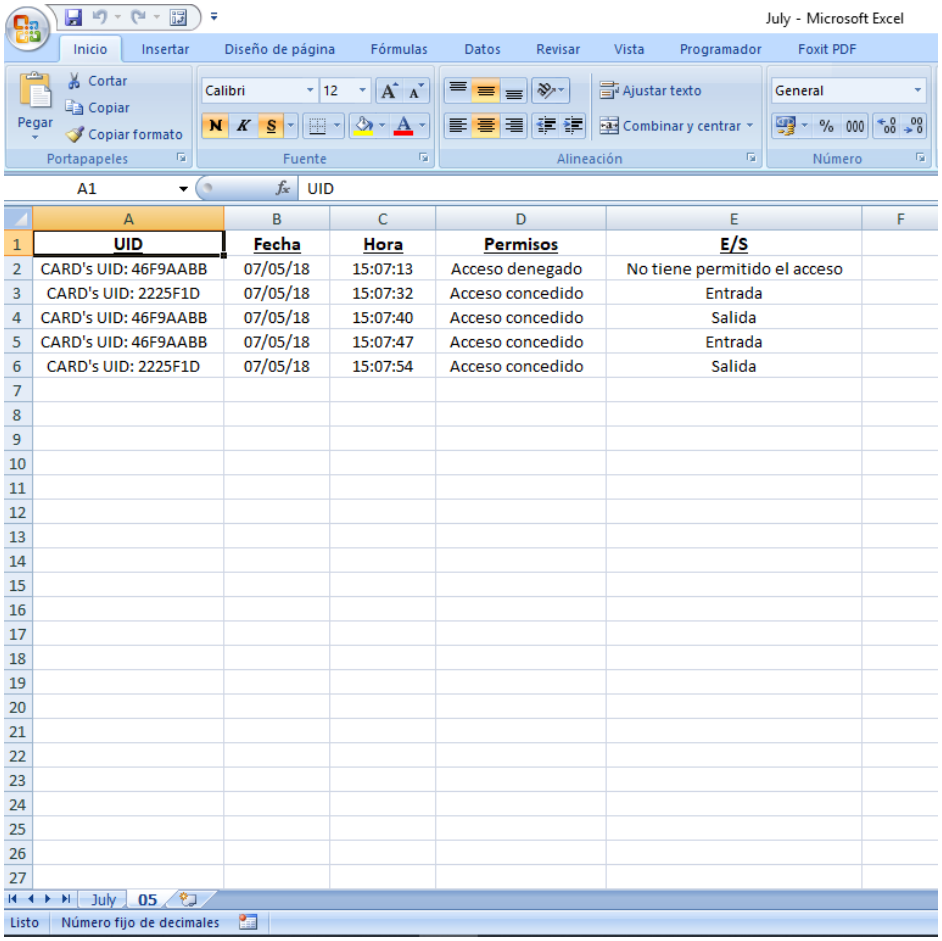


Imagen 4.3: Primera hoja del archivo Excel generado (portada).



July - Microsoft Excel

Inicio | Insertar | Diseño de página | Fórmulas | Datos | Revisar | Vista | Programador | Foxit PDF

Pegar | Cortar | Copiar | Copiar formato | Portapapeles | Calibri | 12 | Fuente | Ajustar texto | Combinar y centrar | Alineación | General | Número

A1 | UID

	A	B	C	D	E	F
1	**UID**	**Fecha**	**Hora**	**Permisos**	**E/S**	
2	CARD's UID: 46F9AABB	07/05/18	15:07:13	Acceso denegado	No tiene permitido el acceso	
3	CARD's UID: 2225F1D	07/05/18	15:07:32	Acceso concedido	Entrada	
4	CARD's UID: 46F9AABB	07/05/18	15:07:40	Acceso concedido	Salida	
5	CARD's UID: 46F9AABB	07/05/18	15:07:47	Acceso concedido	Entrada	
6	CARD's UID: 2225F1D	07/05/18	15:07:54	Acceso concedido	Salida	
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						
21						
22						
23						
24						
25						
26						
27						

July | 05

Listo | Número fijo de decimales

Imagen 4.4: *Segunda hoja del archivo Excel generado (registros).*

CAPÍTULO 5:

ESTUDIO ECONÓMICO

5.1. INTRODUCCIÓN

El estudio económico tiene por objeto la evaluación aproximada de los costes necesarios para el desarrollo del Trabajo de Fin de Grado. Para ello, se desglosa la elaboración del TFG en cada una de las diferentes etapas de que se compone, desde su planteamiento inicial hasta su terminación y presentación de los resultados.

Para la valoración de los costes, se tienen en cuenta elementos reales como las horas dedicadas por cada profesional en su elaboración y desarrollo y los recursos materiales empleados en la realización de las diferentes etapas.

5.2. PROFESIONALES PARTÍCIPES DEL PROYECTO

En la elaboración de cualquier proyecto intervienen una serie de personas con roles y cargos definidos. A la cabeza se encuentra el Director de proyecto, responsable máximo del mismo, que se encarga de proporcionar las ideas generales, coordinar, aconsejar y guiar al resto del equipo en todas las fases del desarrollo y ejecución del proyecto, hasta su aprobación y validación final.

En este preciso caso, además del director de proyecto, existe otro único profesional, un Ingeniero en Organización Industrial, responsable de las tareas de documentación, diseño y confección del proyecto. A partir de sus conocimientos, lleva a cabo el trabajo más complejo y extenso de exploración y desarrollo para cumplir con las especificaciones marcadas y los objetivos finales. Además, realiza la búsqueda de información, estructuración, redacción y corrección de los documentos pertinentes. El ingeniero está en contacto, de forma habitual, con el director del proyecto para realizar consultas técnicas e informar del progreso y los avances realizados en el trabajo.

Además, en este caso, el Ingeniero hará también las veces de administrativo, encargándose de las funciones burocráticas. Esta figura tiene como función la gestión de documentos de la empresa, el alquiler de los servicios requeridos, la tramitación de permisos y licencias, la compra de materiales, etc.

5.3. DEFINICIÓN DE LAS FASES

El presente TFG, que está orientado a la aplicación de la tecnología RFID para el control de la jornada laboral, mediante programación en Python y hardware de bajo coste, se ha estructurado, para su elaboración en una serie de etapas, representadas en la Imagen 5.1. El punto de partida es una

planificación inicial a partir de la idea del proyecto, definiendo los componentes que van a intervenir, los recursos necesarios, las tareas a realizar, y se establecen además plazos y tiempos de entrega antes de poner en marcha la metodología de trabajo en el TFG.

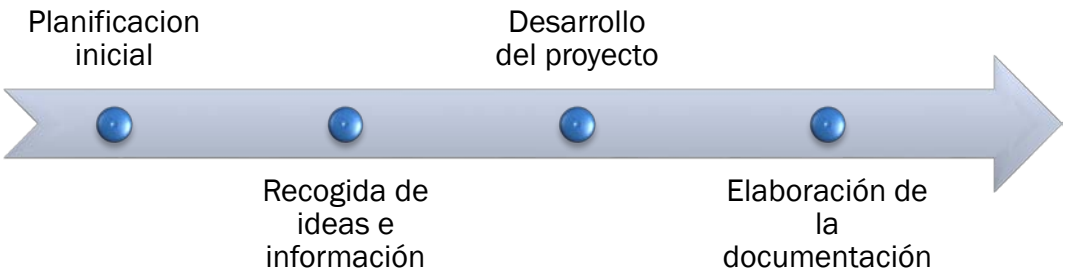


Imagen 5.1. *Fases del proyecto*

Fuente: elaboración propia.

A continuación tiene lugar la recopilación de información, la búsqueda de datos, referencias, herramientas existentes y material acerca del tema. Se recogen también todos los planes e ideas aportados por los interesados en el proyecto.

La etapa que sigue corresponde al desarrollo del cuerpo principal del proyecto, es decir, el montaje y programación de un control de acceso RFID mediante Python y un hardware de bajo coste, concretamente Arduino, aplicando todas las ideas y conceptos aprendidos.

Para ello, se realiza el montaje inicial a la vez que se programan los primeros códigos, que servirán de punto de partida para realizar pruebas e implementar mejoras hasta alcanzar los objetivos fijados.

Por último, todo el desarrollo práctico del TFG es descrito y documentado en una memoria teórico-práctica para entregar al cliente, que comprende la información necesaria para el total entendimiento del proyecto.

5.4. ANÁLISIS DEL PROYECTO

En este apartado se lleva a cabo la valoración económica del presente TFG a partir de todos los recursos necesarios para su desarrollo.

En la evaluación económica se tienen en cuenta una relación de los costes asociados a los siguientes apartados: personal, amortizaciones de los equipos informáticos, materiales consumibles y servicios indirectos del proyecto. Se analiza cada una de esas partes de forma individual con el objetivo de conocer la influencia que tiene cada una de ellas sobre el valor final del estudio.

5.4.1. HORAS EFECTIVAS Y TASAS HORARIAS DEL PERSONAL

En esta sección se obtienen las tasas por hora y por semana de cada uno de los profesionales que participan en la realización de este proyecto, para la valoración de los costes asociados al personal.

Inicialmente, se determina la cantidad de horas efectivas por trabajador para la realización de un TFG de esta envergadura, a lo largo de un período de cuatro meses, que concretamente abarca desde mediados de marzo hasta mediados de julio. Se calcula también el número de semanas efectivas en ese período (Tabla 5.1).

Concepto	Valor
Día inicio período	17/03/2018
Día fin período	10/07/2018
Período (días)	117
Sábados y domingos	34
Días festivos en Valladolid	4
Días de vacaciones	0
Días de baja médica	7
Días de cursos formativos	7
TOTAL DÍAS HÁBILES	79
TOTAL HORAS EFECTIVAS	632
TOTAL SEMANAS HÁBILES	16,4

Tabla 5.1. Cálculo de horas/días/semanas hábiles en el período.

Fuente: elaboración propia.

Los empleados que forman parte del proyecto son los definidos en el apartado 5.2 Profesionales partícipes del proyecto. A partir de los sueldos estándar asociados a cada uno de ellos para un período de aproximadamente cuatro meses (para facilitar el proceso, de ahora en adelante consideraremos que el período consta de cuatro meses) y de las horas efectivas de trabajo determinadas se calculan las tasas de cada empleado por hora y por semana (Tabla 5.2).

Concepto	Director del Proyecto	Ingeniero Industrial
Salario mensual (€)	3254	1993
Salario 4 meses (€)	13016	7972
Seguridad Social (35%)	4555,6	270,9
Coste total (€)	17571,6	8242,9
Coste (€/hora)	28,53	13,38
Coste (€/semana)	1098,23	515,18

Tabla 5.2. *Cálculo de coste del personal por hora y semana.*

Fuente: elaboración propia.

5.4.2. AMORTIZACIONES DEL EQUIPO INFORMÁTICO

En la Tabla 5.3 se recogen los costes asociados a los equipos informáticos, hardware y software, necesarios para la elaboración del TFG. El período de amortización considerado para los de equipos informáticos es de cinco años, con una cuota lineal, por lo que los costes totales se reparten equitativamente entre cinco períodos. No obstante, la amortización de los equipos en este proyecto será la proporcional a los cuatro meses de duración respecto del total de un año.

Concepto	Coste (€/ud.)	Unidades	Coste Total (€)
	Hardware		
Ordenador Hacer Aspire E15	399,00	1	399
Ratón VicTsing Inalámbrico Mini	7,99	1	7,99
	Software		
Licencia Windows 10	145,00	1	145
Licencia Microsoft Office	69,00	1	69
TOTAL A AMORTIZAR			620,99

Tipo	Número	Amortización
Anual	5 años	124,98
Mensual	4 meses	41,66

Tabla 5.3. *Cálculo de la amortización de los equipos informáticos.*

Fuente: elaboración propia.

La cantidad total a amortizar, durante el período de cuatro meses de elaboración del TFG correspondiente a los equipos informáticos es de 41,66€.

5.4.3. COSTE DEL MATERIAL CONSUMIBLE

Los materiales consumibles utilizados en la elaboración de este proyecto constan básicamente del kit Arduino Uno empleado (49,99 €), del módulo RTC que hubo que adquirir a mayores (7,09 €) y de material de papelería y reprografía, tasándose estos últimos gastos en unos 50€ en total. A partir del consumo medio por persona de los mismos, en la Tabla 5.4 se determina el coste del material consumible de 0,04 € por persona y hora de trabajo.

Concepto	Coste (€)
Kit Arduino UNO	49,99
Módulo RTC DS3231	7,09
Papelería y Reprografía	50
TOTAL	107,08
Coste total (€/Persona)	53,54
Coste (€/Persona y hora)	0,08

Tabla 5.4. Cálculo de costes de material consumible por trabajador y hora.

Fuente: elaboración propia.

5.4.4. COSTES INDIRECTOS

Los costes indirectos asociados a la elaboración del proyecto se refieren a los consumos de servicios básicos, como electricidad, agua, calefacción, el alquiler del local amueblado para establecer la oficina, el contrato de servicios de teléfono e internet, entre otros. En la Tabla 5.5 se determina el coste total, de 2040 €, correspondiente a los servicios necesarios durante los cuatro meses de elaboración del TFG.

Concepto	Coste (€/mes)	Coste 4 meses (€)
Alquiler local	300	1200
Teléfono e Internet	60	240
Electricidad, agua, calefacción…	150	600
TOTAL		2040

Tabla 5.5. Cálculo de costes indirectos relativos a servicios.

Fuente: elaboración propia.

5.4.5. TIEMPOS ASOCIADOS A CADA FASE DEL PROYECTO

En la Tabla 5.6 se especifican los tiempos de dedicación, de cada uno de los empleados, necesarios para la elaboración de las diferentes fases del proyecto, descritas en el apartado 5.3. Definición de las fases.

Esta segregación de los tiempos dedicados a cada una de las fases permite determinar el coste derivado de las mismas sobre el total del proyecto. Por ello, se calcula la relación de horas que representa cada fase sobre el total del TFG.

Personal	Fase I (h)	Fase II (h)	Fase III (h)	Fase IV (h)
Director de Proyecto	12	15	30	35
Ingeniero	15	112	300	105
TOTAL (h/fase)	**27**	**127**	**330**	**140**
TOTAL (h)		**624**		
Relación (%fase/total)	4,33%	20,35%	52,88%	22,44%

Tabla 5.6. *Cálculo de las horas de trabajo necesarias por persona para la elaboración del TFG.*

Fuente: elaboración propia.

5.5. COSTES ASIGNADOS A CADA FASE DEL PROYECTO

Los costes de recursos asignados a cada fase del proyecto se calculan a partir de los tiempos de dedicación correspondientes a cada uno de los empleados y de los costes establecidos en el apartado 5.4 Análisis Económico.

El coste estimado total referente a cada una de las etapas está compuesto por el coste de personal, el coste de material consumible, la amortización de los equipos informáticos y el coste de servicios indirectos de la organización. Los dos últimos costes de servicios indirectos y amortización, al ser una cantidad mensual fija, se asignan en cada una de las fases del proyecto de forma proporcional al número de horas dedicadas a cada etapa respecto del total de horas de proyecto (véase Tabla 5.6).

5.5.1. PLANIFICACIÓN INICIAL

En la etapa de planificación inicial intervienen el Director del proyecto y el Ingeniero Industrial para establecer la metodología de trabajo a seguir, definir los objetivos, organizar los recursos y estructurar el proyecto, además, comienza la gestión burocrática del proyecto. Los costes parciales derivados de esta etapa se recogen en la Tabla 5.7.

Concepto		Horas	Coste (€/h)	Coste (€)
Personal	Director Proyecto	12	28,53	342,36
Personal	Ingeniero	15	13,38	200,7
Material consumible	Varios	27	0,08	2,16
		Cantidad	**Porcentaje (%)**	**Coste (€)**
Amortización		41,66	4,33	1,80
Costes indirectos		2040	4,33	88,33
Coste total Fase I				**635,35**

Tabla 5.7. *Cálculo del coste total asociado a la Fase I.*

Fuente: elaboración propia.

5.5.2. RECOGIDA DE IDEAS E INFORMACIÓN

En esta segunda fase participan todos los trabajadores, en especial el Ingeniero, que se encarga de todo el proceso de investigación y búsqueda de documentación en multitud de fuentes. El director interviene de forma eventual, ayudando en algún asunto o resolviendo posibles inconvenientes. Esta etapa implica los costes parciales mostrados en la Tabla 5.8.

Concepto		Horas	Coste (€/h)	Coste (€)
Personal	Director Proyecto	15	28,53	427,95
Personal	Ingeniero	112	13,38	1498,56
Material consumible	Varios	127	0,08	10,16
		Cantidad	**Porcentaje (%)**	**Coste (€)**
Amortización		41,66	20,35	8,48
Costes indirectos		2040	20,35	415,14
Coste total Fase II				**2360,29**

Tabla 5.8. *Cálculo del coste total asociado a la Fase II.*

Fuente: elaboración propia.

5.5.3. DESARROLLO DEL PROYECTO

Es la principal etapa del proyecto y conlleva el aporte y la participación de todos los empleados de la organización. Es la fase más larga y complicada del proyecto, desarrollada, mayoritariamente por parte de los Ingeniero/s. Sin embargo, regularmente se requiere la coordinación con el Director del Proyecto (y con el administrativo, si lo hubiere). Los costes asociados al diseño inicial y construcción práctica del objetivo del proyecto se incluyen en la Tabla 5.9.

Concepto		Horas	Coste (€/h)	Coste (€)
Personal	Director Proyecto	30	28,53	855,9
Personal	Ingeniero	300	13,38	4014
Material consumible	Varios	330	0,08	26,40
		Cantidad	**Porcentaje (%)**	**Coste (€)**
Amortización		41,66	52,88	22,03
Costes indirectos		2040	52,88	1078,75
Coste total Fase III				**5997,08**

Tabla 5.9. Cálculo de coste total asociado a la FASE III.

Fuente: elaboración propia.

5.5.4. ELABORACIÓN DE LA DOCUMENTACIÓN

En la última fase, para la escritura, revisión y aprobación de los documentos necesarios del proyecto se requiere una alta intervención en el trabajo por parte de todos los empleados. En la Tabla 5.10 se exponen los costes derivados de la documentación y registro del TFG.

Concepto		Horas	Coste (€/h)	Coste (€)
Personal	Director Proyecto	35	28,53	998,55
Personal	Ingeniero	105	13,38	1404,9
Material consumible	Varios	140	0,08	11,2
		Cantidad	**Porcentaje (%)**	**Coste (€)**
Amortización		41,66	22,44	9,35
Costes indirectos		2040	22,44	457,78
Coste total Fase IV				**2881,78**

Tabla 5.10. Cálculo de coste total asociado a la FASE IV.

Fuente: elaboración propia.

5.6. RESULTADOS FINALES

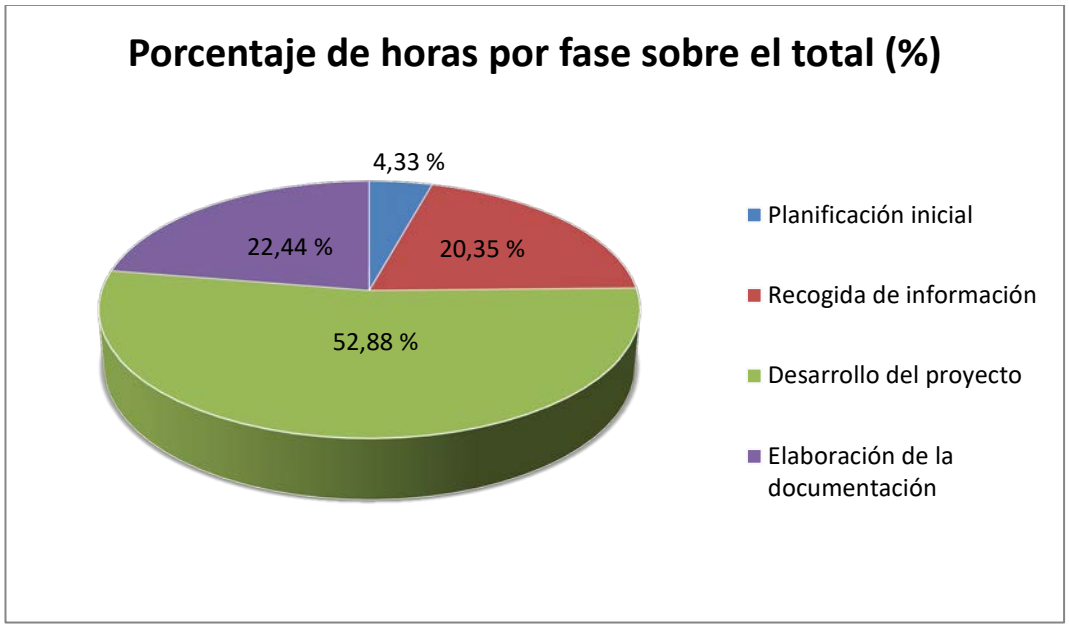
En este punto se evalúan los resultados obtenidos del análisis económico, refiriéndose a tiempo de dedicación y costes asociados. En la Tabla 5.11 se observa el resumen de ambos parámetros para cada una de las etapas del proyecto y los valores finales obtenidos.

RESULTADOS FINALES			
	Fases	**Tiempo (h)**	**Costes (€)**
I)	Planificación inicial	27	635,35
II)	Recogida de información	127	2360,29
III)	Desarrollo del proyecto	330	5997,08
IV)	Elaboración de la documentación	140	2881,78
	TOTAL	**624**	**11874,45**

Tabla 5.11. *Resumen de resultados finales en horas y costes.*

Fuente: elaboración propia.

El tiempo total necesario para la elaboración del presente TFG se obtiene como resultado de la suma de tiempos de todas las etapas. En la Gráfica 5.1 se muestra el desglose de tiempo dedicado a cada una de las cuatro fases sobre el total de 624 horas.



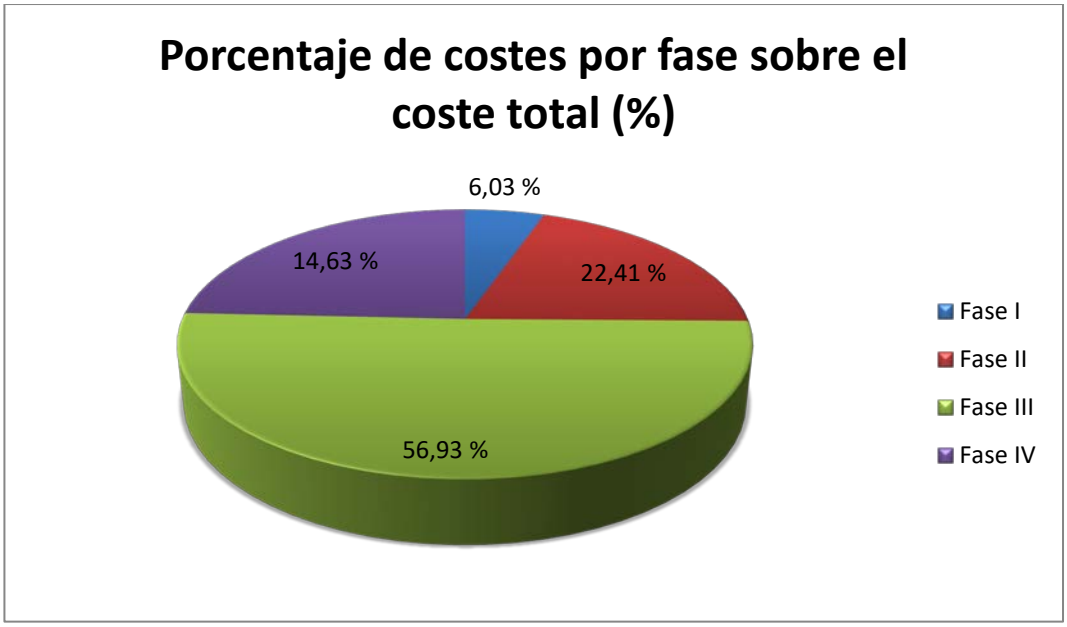
Gráfica 5.1. *Representación de tiempos de cada fase sobre el total del TFG.*

Fuente: elaboración propia.

Como era de esperar, algo más de la mitad del tiempo necesario ha sido empleado en la etapa principal de Desarrollo del Proyecto, y cerca de una cuarta parte en la elaboración de la memoria y el resto de documentación. No obstante, las otras dos etapas, también fundamentales para el desarrollo del TFG, requieren un 24,68% de dedicación del tiempo, lo que supone la cuarta parte restante.

El cálculo del coste total se obtiene como consecuencia de sumar los costes parciales de cada una de las fases. En la Gráfica 5.2 se observa la

proporción que supone el coste de cada etapa respecto del total del TFG de 11.874,45 €.



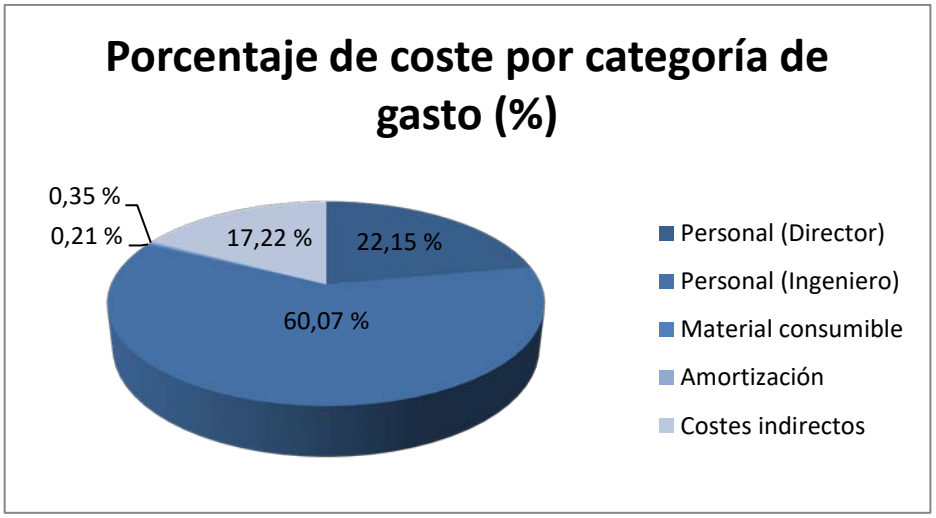
Gráfica 5.2. *Representación de costes de cada fase sobre el total del TFG.*

Fuente: elaboración propia.

A partir del análisis de la Gráfica 5.2 se desprende que el reparto de costes del TFG en función de las distintas etapas es muy similar al obtenido con las horas de dedicación, con un 71,56% para las etapas principales de Desarrollo del proyecto y Elaboración de la Documentación. El porcentaje restante corresponde a la Planificación Inicial y a la Recogida de Información, que en este proyecto supone un 22,41% de los costes.

Además, en la Gráfica 5.3 se visualiza la distribución del coste total necesario entre los diferentes gastos contemplados en el punto 5.4 Análisis Económico.

En la Gráfica 5.3 se comprueba que los costes se deben principalmente a los salarios del Ingeniero y del Director, que van a desarrollar la totalidad del TFG, y abarcan más de un 80% del coste total.



Gráfica 5.3. *Representación de los diferentes tipos de coste sobre el total.*

Fuente: elaboración propia.

El coste final de realización del TFG, Tabla 5.12, es resultado de aplicar al coste total anterior un porcentaje de beneficio del 30% para la organización y un 21% de impuestos (I.V.A.). Así, el coste final del TFG es de 17692,80 €.

TOTAL		11.874,45 €
Beneficio	30 %	3562,34 €
I.V.A.	21 %	2493,63 €
COSTE TOTAL DEL PROYECTO		17930,42 €

Tabla 5.12. *Coste final del TFG tras beneficios e impuestos.*

Fuente: elaboración propia.

CONCLUSIONES Y LÍNEAS DE MEJORA

A lo largo de todas las etapas que han formado parte del desarrollo de este proyecto, se han ido extrayendo diversas conclusiones, algunas de carácter general y otras relacionadas con los objetivos específicos que se marcaron al inicio del trabajo.

Conclusiones de carácter general: a grandes rasgos, durante el desarrollo de este TFG se ha podido concluir en la vital importancia de la necesidad de adaptación de la Industria a los tiempos que corren, es decir, a la Industria 4.0, siendo imperante una evolución de todas las fábricas hacia las denominadas fábricas inteligentes, adquiriendo un mayor grado de automatización. De lo contrario, las fábricas que no se adapten a esta imperante necesidad correrán el riesgo de quedar obsoletas y de perder rentabilidad y competitividad. Como se suele decir, "renovarse o morir".

Conclusiones de carácter específico: en cuanto a los objetivos fijados inicialmente relativos a la adquisición de conocimientos técnicos, se ha logrado la adquisición de conocimientos específicos sobre los lenguajes de programación de Arduino y Python, lo cual redunda a su vez en un desarrollo de las habilidades generales de programación.

También se han conseguido, como se marcó como objetivo en un principio, la familiarización con el hardware low-cost y sus inacabables posibilidades y con los conceptos básicos de la tecnología RFID.

En lo respectivo a los aspectos funcionales del proyecto, los requerimientos iniciales eran que, mediante identificación por tecnología RFID, el dispositivo controlase los permisos de acceso del tag identificado y que almacenase y enviase dicha información en tiempo real.

La identificación mediante RFID se ha logrado de manera satisfactoria mediante la programación y la utilización del módulo lector RC 522. También se ha logrado desarrollar un control de acceso efectivo, valiéndose de un módulo relé y programando la ejecución de dicho control durante los días y horas deseados. Todo lo anterior supone una evolución en materia de seguridad, así como el sistema de iluminación mediante leds empleado en el prototipado del proyecto.

En lo que a la transmisión de información se refiere, se ha conseguido enviar los datos de todos los accesos (o intentos de acceso) que se producen a lo largo de cada día en un archivo Excel, que abarcará un período de un mes, para su posterior análisis o almacenamiento.

Si bien esta transmisión de información se realiza en tiempo real, siendo la fecha y la hora de cada acceso parte de los datos transmitidos, no se podrá hacer uso de la misma en tiempo real, ya que en primer lugar ha de

almacenarse el archivo Excel correspondiente, y si dicho guardado se produce en mitad de un día con el control de acceso aún activo, al finalizar de trabajar con el archivo Excel y reactivar el control se sobrescribirían los datos de dicho día. En resumen, cada archivo Excel puede ser abierto, manipulado y estudiado al finalizar la jornada diaria.

Una línea de mejora para trabajos futuros basada en este último aspecto así como otras mejoras futuribles se presentan en el siguiente sub-apartado.

LÍNEAS DE MEJORA.

Como sugerencias para trabajos futuros, se señalan ciertos aspectos del proyecto que podrían mejorarse con el fin de optimizar su funcionamiento. Para no destacar erróneamente la importancia de unos sobre la de otros, se presentan en orden alfabético:

- Ejecución del programa: Mientras que el sketch de Arduino permanecerá cargado en la placa de manera ininterrumpida, para el funcionamiento del programa Python éste ha de estar en ejecución dentro del entorno correspondiente, en este caso, Eclipse. Sería conveniente en un futuro generar un archivo ejecutable a partir del archivo en formato .py con el fin de programar una tarea automática en la computadora donde se utilice el dispositivo, pudiendo prescindir de la apertura de un intérprete para su puesta en marcha.

- Estética: visualmente, el dispositivo presenta un aspecto ciertamente desordenado, con demasiados cables a la vista. Convendría encapsular los circuitos, dejando únicamente a la vista los elementos necesarios, como los leds.

- Rango de alcance: al ser este un proyecto a baja escala, el rango de alcance del módulo lector es de solo unos centímetros, siendo conveniente a efectos prácticos en el mundo real un mayor rango de alcance. El aumento de la distancia de lectura se puede conseguir fácilmente mediante el empleo de una antena.

- RTC: La dependencia del módulo RTC puede suponer un problema en materia de seguridad, ya que, a pesar de que la programación correspondiente está destinada a evitar estas cuestiones, aún se podrían producir errores por falta de sincronización, reinicio de los datos del RTC, agotamiento de las baterías, etc. Sería conveniente una mayor fiabilidad horaria del

programa, conectándose tal vez a un servidor horario de internet o al propio del sistema de computadoras donde se emplee.

- Seguridad: para aumentar la seguridad, sería deseable añadir cierta información confidencial a los tags que se fuesen a emplear como identificación mediante un módulo de lectura/escritura, siendo estos datos los utilizados posteriormente por el programa de identificación correspondiente, evitando emplear como identificación únicamente la UID del tag, ya que la tecnología RFID presenta la oportunidad de almacenar y transmitir grandes cantidades de información en poco tiempo y convendría aprovecharlo.

- Tamaño de la memoria: la cantidad de UIDs que se puede almacenar es limitada debido al tamaño de la EEPROM de la placa Arduino (1 kB). Existen placas con más capacidad de almacenamiento que permitirían el registro de un mayor número de UIDs con permisos de acceso si fuese necesario.

- Transmisión de información: como ya se señaló al inicio de este apartado, mientras que la información se transmite en tiempo real, no se puede manipular a la vez, ya que se sobrescribirían los datos correspondientes a la jornada en cuestión. Este defecto habría de ser mejorado con el fin de facilitar el empleo de las técnicas de análisis de datos y aprovechar al máximo la información obtenida y las oportunidades que se pudieran presentar.

Las directrices anteriores presentan ciertos aspectos del proyecto que se pueden mejorar, aunque probablemente existan más, difíciles de detectar inicialmente pero que aparecerían al hacer un uso prolongado del dispositivo.

BIBLIOGRAFÍA

REFERENCIAS BIBLIOGRÁFICAS

Cuenca Lacruz, A.M. y Salt Llobregat, J.J. (2005). *Automática Industrial y Control.* Universidad Politécnica de Valencia: Valencia.

Galindo Neira, L.E. y Ortiz Jiménez, J.G. (2005). *Economía y Política 2.* Santillana: Bogotá.

Schwab, K. (2016). *La Cuarta Revolución Industrial.* DEBATE.

Yee, J.T.; Oh, S. (2012). *Technology Integration to Business: Focusing on RFID, Interoperability, and Sustainability for Manufacturing, Logistics, and Supply Chain Management.* Springer: London.

REFERENCIAS WEB

Amazon (2018). Recuperado de: https://www.amazon.es/

Anderson, C., (25 de enero, 2010). "In the next industrial revolution, atoms are the new bits". *Wired*. Recuperado de: https://www.wired.com/2010/01/ff_newrevolution/

Aprendiendo Arduino (27 de junio, 2016). Recuperado de: https://aprendiendoarduino.wordpress.com/2016/06/27/arduino-uno-a-fondo-mapa-de-pines-2/

Arduino – Home (2018). Recuperado de: https://www.arduino.cc/

Audiencia Nacional, Sala de lo Social, Sentencia 207/2015 de 4 Dic. 2015, Rec. 301/2015 (27 de enero, 2016). *Diario La Ley, Nº 8690, Sección La Sentencia del día.* Recuperado de: http://diariolaley.laley.es/content/Documento.aspx?params=H4sIAAAAAAAEAMtMSbH1CjUwMDA1NTA1NDJRK0stKs7Mz7Mty0xPzStJBfEz0ypd8pNDKgtSbdMSc4pT1RKTivNzSktSQ4sybUOKSIMB-0yV80UAAAA=WKE

Blog de By: Sistemas de control de accesos y control de puertas – ¿Qué es RFID? (2010). Recuperado de: https://www.by.com.es/blog/que-es-rfid/

Cano, J.L. (19 de enero, 2014). Pybonacci – Leer datos de Arduino desde Python. Recuperado de: https://www.pybonacci.org/2014/01/19/leer-datos-de-arduino-desde-python/

Curso de Python – codigofacilito (2014). Recuperado de: https://www.youtube.com/playlist?list=PLE549A038CF82905F

Diakopoulos, N., Cass, S., (24 de julio, 2017). "Interactive: The Top Programming Languages 2017". *IEEE Spectrum.* Recuperado de: https://spectrum.ieee.org/static/interactive-the-top-programming-languages-2017

DigiKey Electronics (2018). Recuperado de: www.digikey.com

El papel de la tecnología RFID en la fábrica inteligente (2010). Recuperado de: http://www.clase10.com/el-papel-de-la-tecnologia-rfid-en-la-fabrica-inteligente/

ElProCus – Electronic Projects for Engineering Students (2015). Recuperado de: https://www.elprocus.com/different-types-of-relays-used-in-protection-system-and-their-workings/

Estatuto de los Trabajadores (24 de octubre, 2015). *Boletín Oficial del Estado núm. 255.* Recuperado de: https://www.boe.es/buscar/act.php?id=BOE-A-2015-11430

Fábrica Digital (n.d.). Recuperado de: https://fabricadigital.org/tienda/rtc-ds3231-reloj-en-tiempo-real/

Gerbert, P., Lorenz, M., Rüßmann, M., Waldner, M., Justus, J., Engel, P., Harnisch, M., (9 de abril, 2015). "Industry 4.0: The Future of Productivity and Growth in Manufacturing Industries". *BCG.* Recuperado de: https://www.bcg.com/publications/2015/engineered_products_project_business_industry_4_future_productivity_growth_manufacturing_industries.aspx

GitHub - The world's leading software development platform (2018). Recuperado de: https://github.com/

Guía sobre seguridad y privacidad de la tecnología RFID (2010). Publicada por el Instituto Nacional de Tecnologías de la Comunicación (INTECO) y la Agencia Española de Protección de Datos (AEPD). Recuperado de: https://rmd.jcyl.es/web/jcyl/MunicipiosDigitales/es/Plantilla100Detalle/1274785511218/481/1279887869260/Redaccion

Ley sobre Infracciones y Sanciones en el Orden Social (8 de agosto, 2000). *BOE núm. 189.* Recuperado de: https://www.boe.es/buscar/act.php?id=BOE-A-2000-15060

Llamas, L. (2018). Ingeniería, Informática y Diseño. Recuperado de: https://www.luisllamas.es/

Munera, I., (17 de octubre, 2017). “El Congreso tramitará la ley que obliga a las empresas a registrar las horas trabajadas”. *Diario El Mundo.* Recuperado de: http://www.elmundo.es/economia/2017/10/17/59e6250dca4741b2278b46f9.html

Naylamp Mechatronics (n.d.). Recuperado de:https://naylampmechatronics.com

openpyxl - A Python library to read/write Excel 2010 xlsx/xlsm files (2018). Recuperado de: http://openpyxl.readthedocs.io/en/stable/

Processing (2018). Recuperado de: https://processing.org/

PyDev (2018). Recuperado de: http://www.pydev.org/

Python 2.7.15 Documentation (2018). Recuperado de: https://docs.python.org/2.7/

Python Mania: control de bucles, break, continue y pass (2013). Recuperado de: https://www.pythonmania.net/es/2013/04/05/control-de-bucles-break-continue-y-pass/

Python Software Fundation (2018). Recuperado de: https://www.python.org/

Python vs C (2018). Recuperado de: https://difentre.com/difference-between-python-and-vs-c-language/

Raspberry Pi – Teach, Learnd and Make with Raspberry Pi (2018). Recuperado de: www.raspberrypi.org

Real Academia de Ciencias Exactas, Físicas y Naturales (2018). Recuperado de: www.rac.es

Real Academia Española (2018). Recuperado de: www.rae.es

RFID RC-522 (2016). Recuperado de: https://forum.arduino.cc/index.php?topic=429296.0

Robinson, D. (6 de septiembre, 2017). The Incredible Growth of Python. Recuperado de: https://stackoverflow.blog/2017/09/06/incredible-growth-python/

Single Relay Board #27115 (2013). Recuperado de: https://docs-emea.rs-online.com/webdocs/13b5/0900766b813b5d9e.pdf

Sistemas – Definición de IDE (n.d.). Recuperado de: https://sistemas.com/ide.php

Soluciones de software de gestión empresarial para pymes | SAP - ¿Qué es el internet de las cosas? (2016). Recuperado de: https://www.sap.com/spain/trends/internet-of-things.html

Stack Overflow – Where developers learn, share & build careers (2018). Recuperado de: https://stackoverflow.com/

Techspirited – Pros and cons of RFID Technology (2018). Recuperado de: https://techspirited.com/pros-cons-of-rfid-technology

The Arduino Playground (2018). Recuperado de: https://playground.arduino.cc/

The Eclipse Foundation (n.d.). Recuperado de: https://www.eclipse.org/

The Python Package Index – Schedule (2018). Recuperado de: https://pypi.org/project/schedule/

Universidad VIU (2017). Recuperado de: www.universidadviu.es/tecnologia-rfid-mejores-aplicaciones/

Virtual Pro – Portal especializado en procesos industriales (2016). Recuperado de: https://www.revistavirtualpro.com/noticias/reles-pieza-fundamental-en-la-industria-para-el-control-de-circuitos-electricos

What is a Raspberry Pi? (n.d.) Recuperado de: https://opensource.com/resources/raspberry-pi

Wiring (2018). Recuperado de: wiring.org.co

ANEXO

Asset Management / Smart Displays /UHF / Tool Tracking / Business Intelligence / ZigBee / Race Timing / Supply Chain / Retail Inventory Management / Pharma Pedigree / Work in Process / In-Transit Visibility / Access Control / NFC / Tolling / Border Control / Real Time Visibility / Staff Tracking / Patient Tracking / High Frequency / Security / Worker Safety / Smart Shelves / Embedded RFID / Reality Search Engines / IT Asset Tracking / Loss Prevention / Mobility / Low Frequency / Vehicle Tracking / RTLS / Container Tracking / Aerospace / Internet of Things / Active RFID / Smart Objects

The broad adoption of all types of RFID and sensing (RFIDS) solutions shows that the much-heralded promise of the technology goes far beyond the supply chain focus that generated so much hype over the last decade.

In fact, new innovative solutions where users naturally interact with RFIDS, and where the technology is so integrated and transparent that it disappears into its environment, are becoming commonplace. These innovations point to the fact that the vision of the ***Internet of Things*** and ***Reality Search Engines*** embraced by ThingMagic is coming to reality.

Just a few possibilities:

1. The Next Revolution in Wireless and Mobility
How RFID and Sensing Is Automating Identification, Data Collection, and Location Systems

2. The Batteryless RFID Imperative in HealthcarePatient-Centric Applications That Are Changing the Healthcare Landscape

3. Enhancing the Patient Experience with RFID
"It's Like Angels Singing"

4. Hospital Inventory Control with UHF RFID
What if Your Goods Could Talk?

5. Race Timing with RFID
How Rosie Ruiz Changed an Industry

6. RFID for Event & Hospitality Management
Creating Interactive and Personalized Customer Experiences

7. RFID-Enabled Smart Displays
RFID is the Missing Link Between Online Information and the Real World

8. Automating Order Fulfillment with a New Generation of RFID
Back to the Future

9. No Greenwashing Here: How RFID Helps the Environment
3 Powerful Examples: EV Power Station Billing, Pedal Power, and Recycling

10. No Greenwashing Here: How RFID Helps the Environment (Part 2)
Encouraging Healthy, Green Modes of Transportation with RFID

11. No Greenwashing Here: How RFID Helps the Environment (Part 3)
RFIDS for Recycling Incentive Programs, Waste Hauling Automation & Smart Packaging

12. RFID – The New Future of Retail
Everything Old is New Again

13. Specialty Retail Inventory Management with RFID
From Diamond Dealers to Pawn Shops, RFID Allows Small Retailers to Compete with Big Box Stores

14. RFID Enabled Smart Shelves
Out of Stock? I'm Out of Here!

15. RFID Takes on Container Fraud
Horticulture Partnership Takes a Giant Step for its Industry

16. Designer RFID
Enhancing the Shopping Experience While Boosting the Bottom Line

17. Improving Farming with RFID
Making Hay with RFI

18. Construction Management with RFID
Tool Tracking, Heavy Equipment Management, and Next-Generation 4-Dimensional Building Information Modeling

19. Process Manufacturing with RFID
WIP it Good - Vaillant Group Uses RFID forAutomation within One-Man Boiler Manufacturing Process

20. RFID for Document Management
Giving Valuable Documents a Digital Identity

21. The House Bets on RFID
Giving New Meaning to the 'Eye in the Sky'

22. E is for Agriculture?
How RFID Enables Sustainable Forrest Management, Accurate Apricot Harvesting, and Food Safety

23. RFID and Social Networks
RFID Brings the Digital 'Like' to the Physical World

24. The Appliance Approach to RFID
Combining Read Points and Auditing For Asset Tracking

25. Oncology Solution Uses RFID to Improve Patient Experience
Because it Takes More Than Courage to Beat Cancer

26. RFID and the Beverage Marketplace (part 1)
Thank You Sir! May I Have Another?

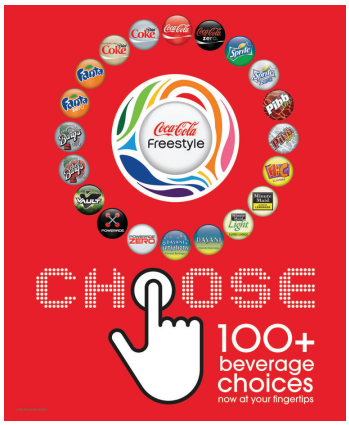
27. RFID and the Beverage Market (part 2)
RFID for Wine Production and Anti-Counterfeiting

28. RFID and the Beverage Market (part 3)
Beverage Dispensing Managing the Perfect Pour

29. RFID and the Beverage Market (part 4)
Combating Underage Drinking and Improving the Bottom Line with Patron ID and Point-of-Sale Wristbands

30. RFID and the Beverage Market (part 5)
Serving Up Soft Drinks, Freestyle

31. Zebra Earns its RFID Stripes Again
Low Cost, Item-Level Tagging for Package Verification, Work in Process, Product Authentication, and More



32. RFID Helps Deaf Children Learn Sign Language
Computer Program and RFID-Enabled Toys Designed to Help Preschoolers Learn Faster

33. Planes, Trains, Automobiles & RFID
Transportation manufacturing industry relies more and more on RFID

34. Mission Accomplished
RFID Lends Precision to Fighter Jet Refueling

35. RFID on School Busses
Schools Out, Do You Know Where Your Child Is?

36. How to Share Your Car with a Stranger
RelayRides Provides Access to Your Neighbor's Car with RFID

37. RFID Keeps its Cool
Temperature Tracking in Real-Time for Sensitive Shipments

38. RFID for Automobile Production
From Ford to Fahrvergnügen, RFID Expands the Opportunity to Streamline Production and Lower Manufacturing Costs

39. RFID for Border Security
U.S. Customs and Border Protection Programs Simplify Passage for Pre-Approved Travelers

40. Check it Out - RFID for Library
Media Tracking Significant Growth in RFID for Library Operations Reported

41. Cleaning Up Hazardous Materials with RFID
Manhattan Project Site Uses RFID for Waste Management

42. DC-Area Software Firm Partners with National Industries for the Blind
RFID Technology Aids People Who Are Blind in the Workplace

43. Live From Fargo
RFID for Real Time Visibility in Healthcare – You Betcha

44. RFID Predictions
Embedded Intelligence Drives Epic Innovation

45. Smart Buildings
Passive UHF RFID and the Smart Building Nervous System

46. Happy Birthday ThingMagic!
10th Anniversary Celebrates Key Milestones, Innovative Applications and Offers Predictions for the Future

47. RFID for Airport Operations
Can RFID Help Make the Skies More Friendly?

48. RFID Helps Estée Lauder Innovate
Interactive Kiosks Provide Unique User Experience & New Level of Business Intelligence

49. Shredding It with Sensors
Nokia and Burton Boards Combine the Misty Flip with Mobile Apps

50. Don't Lose Your Taser Bro
Weapons Tracking and Management with RFID

51. Traffic Management with RFID
You Can't Get There From Here (or can you?)

52. Automating Point-of-Sale Processes with RFID
An End to the Supermarket Checkout Line?

53. Urban Planning with Building Blocks and RFID Paint Brushes
Next-Generation Building & Cityscape Modeling

54. RFID Puts Las Vegas Under Lock & Key
Taking Luxury Guest Rooms to the Next Level

55. RFID Let's Cars Park Themselves
The Next Generation Parking Garage

56. Cactus Chips
RFID Helps Protect a Desert Symbol

57. Smart Ball
RFID Will Help Deliver the Fan Experience of the Future

58. Milkin' It with RFID
Robotic Milking Systems Use RFID and Sensors to Increase Production and Ensure Quality

59. RFID Gives New Meaning to Train Spotting
Single View of trains to Improve Railway Safety

60. Music Branding with RFID
RFID to Match the Mood of Clothing with Music

61. There's Something in the Air
RFID Detects Airborne Toxins

62. Cup o' Joe to Go
RFID and Coffee Drinkers Unite to Cut Waste and Time

63. RFID for Disaster Management
Auto ID Technologies Help Deliver Supplies and Locate People During 72 Hour 'Golden' Rescue Period

64. Are RFID License Plates Coming Down the Road
Or Will Google Cars Solve the World's Traffic Problems?

65. The New Junkyard Dog
RFID Helps Sniff Out Hiding Vehicles in Salvage Yards

66. Don't be a Hothead
RFID-Enabled Helmet Designed to Reduce Cases of Heatstroke

67. Are RFID Tags on the Ropes?
RFID Used to Improve Mussel Cultivation

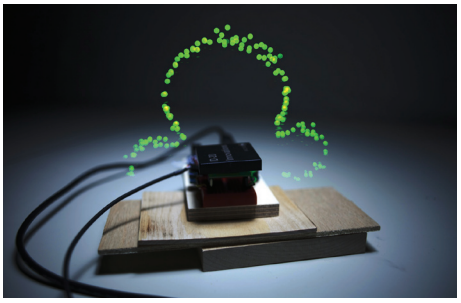
68. RFID Hits Center Stage
Following Opera Singers to Capture the Experience for the Audience

69. ThingMagic, a Division of Trimble
ThingMagic Continues Lead Position as Developer of RFID Technology from Within Trimble Navigation

70. RFID and Mud Motors
A Natural Fit for Natural Resource Exploration

71. Automated Video Stores
A New Way to Rent Movies – Powered by RFID

72. Ghost in the Machine
RFID and the 'Magic' of Nearness



73. RFID Replaces Ouija Boards, Witchcraft, and Séances
Lets the Dead Speak from Beyond the Grave

74. Billions of Identities
Use of RFID Enabled National eID Cards and ePassports is Growing

75. RFID Gets Waterproofed
Helps Control Crowds at the Pool and More

76. Man Down Monitoring
RFID Provides a Safety Net for Workers in Hazardous Conditions

77. RFID Put Behind Bars
Helps Keep the Peace and Cut Costs in Prisons

78. RFID for Wander Prevention
Providing Added Peace of Mind for Elder Care

79. RFID Has its Finger on the Pulse
Tracking Body Movements for a Picture of Health

80. India's National ID Card Program
RFID and Biometrics to Deliver Access to Social Services

81. RFID Lets Theme Parks Be Fun for All
Takes Special Care of Special Needs Guests and Children

82. Tons of Pig Iron + 2300 °C Blasts + Big Trucks: Are U-Safe?
RFID Enhances Safety and More for Steel Manufacturing Plant

83. RFID Delivers
Keeps Your Bundle of Joy Safe and Sound

84. RFID for Student Tracking
Let Us Know What Grade You Think it Deserves

85. RFID Left Out In The Cold
Solution Provides Tracking, Temperature Monitoring & Scheduling for Iditarod Participants

86. Can You See Mi Now?
RFID Addresses Danger of the 'Right Turn' for Urban Bicyclists

87. Rings, Spiders, Horsetails, Fish and RFID
Auto-ID Technology Used to Secure Nation's Largest July 4th Celebration

88. RFID and the Miracle of Life
Delivers Insight into Fertility Cycles and Prevents Mistakes at IVF Labs

89. RFID and the Bank Branch of the Future
Solution Enables Personalized Banking Services and Process Automation

90. RFID Gives Surgeons Second Set of Eyes
Helps Locate Breast Tumors During Surgery with Precision, Without Risk of Infection

91. IT Asset Tracking with RFID is not Rocket Science
NASA Ditches Paper & Pen in Favor of RFID to Locate Valuable Equipment

92. RFID for High-Value, Critical-Dose Medication Inventory
Are You Prepared For The Future?

93. Personal Robotics Advance with UHF RFID
The Future is Closer Than We May Think

94. Messages on the Move
RFID Helps Cheer Marathoners Along New York City Route

95. Find It, Play It - with RFID
Will This Golf Ball Location System be A Holiday Hit?

96. RFID for Counting Bees. Really?
Beekeepers, Farmers, Buyers and Consumers Benefit from Hive Monitoring

97. The Next Big Step Toward a Multi-Scale Wireless World
Will Consumer Use of NFC Drive Widespread RFID Adoption?

98. RFID and The CSI Effect
Efficiently Tracking Evidence from Collection to Storage

99. RFID Migrates Upstream
Could This Mean the End of Human-Injected Hormones and Antibiotics in our Food?

100. 100 OF 100 Uses of RFID
Use Your Imagination. Just Add RFID.

ThingMagic
A DIVISION OF TRIMBLE

ThingMagic is the Engine in RFID®
www.thingmagic.com