Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

Sistema de control de accesos para el personal de la ENM

Grado en Ingeniería Mecánica

ALUMNO: Alejandro Sánchez Cervera-Mercadillo

DIRECTORES: José María Núñez Ortuño

CURSO ACADÉMICO: 2018-2019

Universidade Vigo

Centro Universitario de la Defensa
en la Escuela Naval Militar

TRABAJO FIN DE GRADO

Sistema de control de accesos para el personal de la ENM

Grado en Ingeniería Mecánica

Intensificación en Tecnología Naval

Cuerpo General

Universidade Vigo

RESUMEN

La Escuela Naval Militar (ENM) es un centro de formación que cuenta con un amplio número de personal, tanto civil como militar, que accede a diario al recinto o hace uso de sus instalaciones. En la puerta principal de acceso a la ENM, los usuarios están obligados identificarse mediante la presentación de una documentación que les acredite para poder acceder al recinto. Sin embargo, en la actualidad no se lleva un registro del personal que entra o sale de la ENM, excepto en el caso de los alumnos, que previamente deben marcar unas casillas en un cuaderno denominado “Libro de Francos” al efectuar su entrada o salida. El presente trabajo detalla la propuesta e implementación de una solución mixta de hardware y software, que permita identificar, mediante diversos métodos alternativos, a todo el personal que entra y sale del recinto de la ENM a fin de llevar un mejor registro y control del personal, mejorando así la seguridad de la dependencia.

PALABRAS CLAVE

Tarjeta magnética, DNI, DNI electrónico, control de acceso, identificación

AGRADECIMIENTOS

A mi tutor, D. José María Núñez Ortuño, por su ayuda constante a lo largo de todo este proyecto y por ayudarme a conseguir sacar este trabajo adelante.

A mis compañeros de promoción, en concreto a Jesús Fernández y muy en especial a Julio Albaladejo, cuyos conocimientos y carácter sosegado, consiguen aminorar cualquier dificultad y dar ánimos para continuar.

Y por último, a mi familia, a Valeria y a todas aquellas personas que me han ayudado a lo largo de estos cinco años en la Escuela Naval Militar.

CONTENIDO

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN Y OBJETIVOS

1.1 Motivación

El control de acceso de la Escuela Naval Militar varía ligeramente en función del tipo de personal que pretenda acceder al recinto:

- Por un lado, en la puerta Carlos I se ha de enseñar la documentación correspondiente para acceder, y ésta es diferente atendiendo al empleo de la persona en la Academia. En el caso de los militares, han de enseñar la TIM, y en de los civiles, una tarjeta específica para dicho propósito que incluye ciertos datos, como el número de DNI, el nombre, la firma y una fotografía personal.

Figura 1-1 Acreditación del personal civil.

- Por otro lado, en el caso de los alumnos, además de enseñar la TIM en la puerta principal, tienen que dejar cuenta de sus movimientos en el libro de francos, a fin de llevar un cierto control de la situación de los alumnos. Este libro custodiado por el Brigadier de guardia, puede estar en manos del Subalterno o del Profesor de Servicio dependiendo de la hora o las circunstancias que se den en la Guardia.

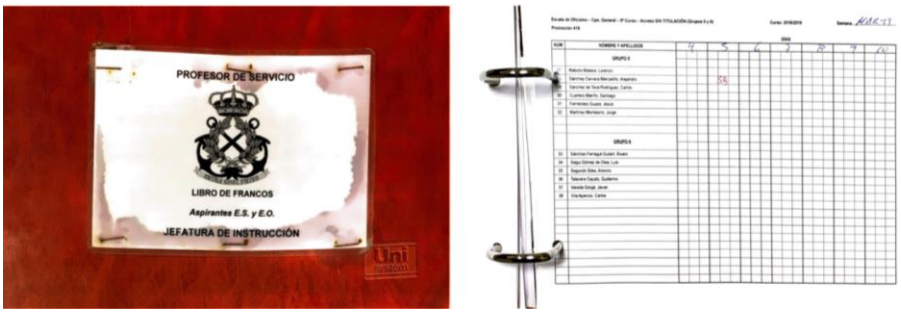

Figura 1-2 Libros de Francos.

El sistema actual, por tanto, presenta una serie de aspectos significativos: las tarjetas entregadas al personal civil no gozan de la seguridad adecuada, ya que son fácilmente reemplazables y de pronunciado desgaste, debido al material del que se componen. El hecho de implementar un sistema común de identificación para todo el mundo podría mejorar la organización y facilitar las entradas y salidas. Por otro lado, sería conveniente poder saber en todo momento, mirando en más lugares aparte de en el libro de francos, qué personas están dentro y quiénes fuera, con el objetivo de llevar un registro más exhaustivo y mejorar la seguridad de la academia.

1.2 Objetivo

El objetivo del proyecto es la implementación de mejoras mediante el planteo de un sistema de control de accesos alternativo al actual, atendiendo a aspectos como la unificación de criterios de identificación de todo el personal que quiera acceder, la posibilidad de llevar un control de salidas y entradas para comprobar a tiempo real y poder tener una situación clara, y agilizar procesos como la incorporación y baja de nuevo personal que vaya a entrar, trabajadores de empresas ajenas a la academia, o incluso pescadores.

1.3 Requisitos

En este trabajo se valoran los distintos sistemas de control de acceso, que serán de aplicación a todo el personal, y que deberá cumplir distintos requisitos:

1.3.1 Rapidez

La persona que desee entrar o salir, en la puerta de Carlos I sólo tendrá que sacar la documentación pertinente, pasarla por un lector, y automáticamente, esa información se almacenará en una base de datos, que podrá consultarse en tiempo real desde diversas ubicaciones.

1.3.2 Fiabilidad

Se contempla que se puedan emplear varios identificadores por persona, para asegurar, en el caso de que falle alguno, y para obtener más datos, si cabe, que corroboren esta identidad. Igualmente, hablando de fiabilidad, se tratará de utilizar tarjetas y marcadores que haya que portar consigo siempre, como por ejemplo la huella dactilar y el DNI.

1.3.3 Universalidad

Uno de los objetivos que se plantea es que, no solo pueda ser para todo el mundo que trabaja o vive dentro de la academia, sino que también se pueda incluir todo aquel personal cuya entrada sea solo temporal, como por ejemplo, en el caso de los pescadores, visitas, o los trabajadores de empresas de fuera que tengan que acudir para realizar alguna reparación.

1.3.4 Redundante y a tiempo real

Como se ha comentado, se intentará que el sistema tenga que pedir varios identificadores para confirmar la entrada y esto quedará registrado en una base de datos, que se actualizará en el momento

en el que se realice una entrada o salida, y que podrá consultarse desde varias plataformas. De esta manera, si el Profesor de Servicio o el Comandante de la Guardia precisan de saber cuál es la situación del personal dentro de la Escuela (si han salido todos los pescadores, si terminaron las visitas, o si todos los alumnos regresaron), solo tendrá que consultarlo desde su ordenador, en vez de tener que llamar a Carlos I para que le confirmen, buscar al alumno que trajo visitas para corroborar que finalizaron o tener que localizar el libro de francos.

1.4 Estructura de la memoria

La memoria del presente trabajo estará divida en cinco partes:

La primera es una revisión del estado del arte, en el que se detallarán aspectos relacionados con los sistemas de control de accesos, como pueden ser sus tipos, las distintas tecnologías electrónicas que utilizan para identificar a los individuos y esencialmente se centrará en las que van a ser utilizadas en el desarrollo del proyecto, como son el DNI electrónico y la tarjeta magnética. Se presentarán los distintos sistemas de accesos implantados en el ámbito del Ministerio de Defensa, así como sus características más reseñables y se hará mención a la ley de protección de datos vigente en nuestro país y en concreto la autorización que posee la escuela para manejar datos personales.

En la segunda y tercera parte se detalla la evaluación de distintas alternativas y desarrollo del sistema elegido. Se revisará tanto el Hardware como el Software, resaltando aquellos aspectos importantes concernientes a su diseño, programación y utilización.

La cuarta y quinta parte tendrán como objetivo presentar las distintas pruebas que se han realizado, así como las conclusiones y líneas futuras que se extraen de ellas.

2 ESTADO DEL ARTE

2.1 Sistemas de control de accesos

2.1.1 Definición

Un sistema de control de accesos es un conjunto de procedimientos, dispositivos o sistemas empleados en tareas de control, registro y verificación del personal que se dispone a hacer uso de la instalación.

2.1.2 Clasificaciones

Mediante estos sistemas, por tanto, se restringe principalmente el acceso a datos o recursos. Se habla de un sistema de control de acceso a datos si este es controlado principalmente por software y se pretende limitar la incursión de usuarios a, por ejemplo, cierta información, o a algún sitio web en concreto. Por otro lado, si lo que se quiere es restringir el acceso a una instalación física, el sistema de control de acceso estará enfocado al recurso, y se tendrán que implementar un sistema de apertura de puertas o tornos.

Atendiendo al grado de automatización, los sistemas pueden clasificarse en:

- **Manuales**, si están regulados por personas encargadas de gestionar las entradas y las salidas.
- **Semimanuales**, si además de contar con vigilantes, se apoyan en un soporte mecánico como puede ser una puerta que se abre y se cierra.
- **Automáticos**, en los cuales se descarta el uso de humanos durante la realización de la verificación y autenticación de la entrada y salida, por medio de sistemas electrónicos, que toman las decisiones en función de su programación. En el presente trabajo se tratará de diseñar un sistema automático, pues cuenta con un programa capaz de verificar y autenticar, pero se necesitará personal tanto para acercar el dispositivo al usuario, como para facilitar la apertura de la puerta. [1]

Si se examina la manera en la que la terminal es capaz de almacenar los datos, se distinguen tres tipos: sistemas de control de accesos autónomos, online y mixtos.

- Los sistemas de control de accesos autónomos. No necesitan estar conectados a un ordenador, ya que los datos se almacenan directamente en el terminal. Gracias a esta característica, resultan ser más flexibles precisos y baratos que los online, pero presentan incomodidad a la hora de introducir datos, ya que sólo se podrá realizar en la ubicación física del terminal.
- Los sistemas de control de accesos online. Se puede interactuar remotamente con ellos mediante un software, así como configurar datos de usuarios de manera sencilla y envío de

mensajes instantáneos, de aviso o emergencia. Se trata de sistemas útiles cuando se desea llevar un seguimiento de todas las acciones realizadas.

- Sistemas de control mixtos. Su comportamiento sigue una línea que es mezcla de los dos anteriores. Funciona de manera autónoma, pero realiza conexiones de manera online. La puesta en marcha de estos sistemas utiliza conexión constante con un servidor, pero en el caso de que ocurra algún imprevisto, ha conseguido guardar esos datos y puede funcionar de manera autónoma. [1] [2]

En referencia a qué se quiere controlar cuando se instalan estos sistemas, se pueden diferenciar entre controles de accesos peatonales, personales y de acceso vehicular.

- Los controles de acceso peatonal simplemente restringen o llevan el control de entrada en instalaciones de personas no autorizadas, como puede ser en un gimnasio o un portal.
- Los controles de acceso personal tienen como objetivo la monitorización de personal que ya se encuentra dentro de una instalación, así como llevar un control más exhaustivo de las horas a las que se puede entrar en una habitación concreta.

Figura 2-1 Controles de acceso personal. [1]

- Los controles de acceso vehicular son los más utilizados en grandes recintos, así como en los aparcamientos de pago en los aeropuertos. Su función se extiende no sólo a registrar el acceso, sino a gestionar el uso de las plazas y pagos. [3]

El control de accesos de la Escuela Naval militar es de tipo personal y vehicular, debido al hecho de que se controla el acceso de ambos al recinto, pero el registro se encuentra limitado, ya que no se lleva cuenta de cuántos vehículos se encuentran dentro, sólo si poseen el pase autorizado.

2.1.3 Componentes de un sistema de control de acceso de personal

Los sistemas de control de accesos destinados a ejercer el control del personal, suelen contar con una serie de elementos comunes, que se describen a continuación:

Lector/terminal: Encargado de transmitir la información entregada por el identificador o credencial (huella, tarjeta o chip), que deberá ser contrastada enviándola a un controlador, de manera remota en el caso de los sistemas online, o local en los sistemas autónomos.

Credencial: Es la información que posee la persona y que es capaz de proporcionarle acceso. Pueden ser códigos de seguridad, tarjetas con un chip o incluso su propio iris.

Servidor: Almacena la información después de cada registro y es capaz de llevar un seguimiento. Por otro lado, es también donde se ejecutan las instrucciones de los programas, por tanto en los sistemas autónomos no se presentan.

Controlador: Elemento encargado de decidir el acceso en función de parámetros como la propia identificación, la hora o la zona. Es consultado en cada intento, dado que ha de comunicarse con el servidor para tomar esta decisión. [4]

Mecanismo de apertura: Mediante la activación de contactos magnéticos o pulsadores eléctricos, previa verificación de la identidad, se dará paso al mecanismo de apertura para la entrada del usuario en concreto.

Elementos de alimentación: Como se ha comentado, los sistemas de control de acceso estarán provistos de diferentes elementos de alimentación dependiendo de su autonomía y su efectividad ante fallos.

2.2 Tecnologías electrónicas de identificación

2.2.1 Lectores biométricos

Se entiende por sistema biométrico aquel que utiliza algún rasgo del físico del usuario para establecer una inequívoca verificación de su identidad. Se puede realizar mediante la lectura de voz, iris, huella dactilar o reconocimiento facial.

Es ampliamente utilizado en sistemas que requieran alta seguridad y comodidad, ya que de manera eficaz se discriminan los rasgos diferentes de cada persona que hace uso de ellos.

El funcionamiento de los sistemas de acceso biométricos se resume en la captura y digitalización de alguno de los rasgos físicos individuo en cuestión y se compara esta información que finalmente resultará en un conjunto de parámetros con una base de datos. Tras esta comparación, se es capaz de decidir si el usuario cumple o no con estos datos guardados de antemano. Los dos errores que pueden dar estos sistemas son, por un lado, el falso rechazo, cuando se le niega la entrada a un usuario que realmente está registrado, debido por ejemplo a un cambio en sus rasgos físicos; por otro, la falsa aceptación, en la que se da un individuo que posee un número de rasgos característicos tales que puede llegar a hacerse pasar por otro del que sí se han registrado previamente en la base de datos.

Existen dos maneras de realizar estas verificaciones. Por un lado, se puede dar la directa, en la que directamente se comparan los datos biométricos con todos los almacenados en la base de datos. Por otro lado, el usuario primero se identifica mediante otra credencial, como puede ser una tarjeta, y luego, cuando se realice la fotografía, el sistema sólo tendrá que establecer si coinciden con los del usuario de la tarjeta.

Ventajas:

- Este tipo de información se lleva siempre consigo, por lo que no puede ser extraviada, garantizando así un acceso constante.
- Se trata de un sistema único que no puede ser sujeto de robo y muy difícilmente a la copia.

Inconvenientes:

- Son sistemas caros, dado que comparan varios rasgos que luego serán convertidos en parámetros, resulta ser un sistema lento.
- Aunque seguros, pueden estar sujetos a imprecisiones, como por ejemplo personas de una misma familia con rasgos muy similares. [1]

2.2.1.1 Huella dactilar

Primer sistema biométrico que se desarrolló y el más utilizado, pudiéndose comprobar en los múltiples dispositivos en los que se integra (móviles, pendrives, tablets). Al ser inequívocas y tener distintos rasgos, las huellas resultan difíciles de falsificar, de hecho, sin ir más lejos, el dedo que se utilice para registrarse tiene que ser uno en concreto, ya que tenemos los diez con rasgos diferentes. Se suelen dar de alta varios en un registro por si uno de ellos fallara debido a imperfecciones en la superficie dactilar.

En cuanto a su modo de funcionamiento, una vez el lector analiza los rasgos de la yema de uno de los dedos éste da una codificación o patrón de identificación único para ese dedo, basado en puntos sobre las líneas que componen la apariencia característica de una yema, pero que son irrepetibles dedo a dedo y persona a persona.

Ventajas:

- Fáciles de instalar.
- Lectores de tamaño reducido.
- Los individuos poseen varias huellas, luego pueden registrar varias en el caso de que alguna dé error.

Desventajas:

- Se utiliza un rasgo susceptible de ser modificado por factores como heridas, quemaduras, que invalidarían la huella.
- Necesidad de contacto físico con el lector. [5]

2.2.1.2 Reconocimiento facial

Se examinan diversos rasgos faciales como la boca, pómulos, distancia entre los ojos y la nariz para obtener una plantilla única a partir de una fotografía tomada. Ampliamente utilizada debido a que no utiliza contacto, pero posee desventajas tales como ser susceptible al cambio de apariencia, envejecimiento del rostro o la diferente exposición a la luz.

2.2.1.3 Iris

Al poseer cerca de 200 rasgos diferentes, el iris se establece como un elemento único que puede ser utilizado para proporcionar identificación. Se utiliza una cámara para captar una imagen, de la que se obtendrá un código único. Una de las ventajas es que, además de no necesitar contacto, no es susceptible de grandes modificaciones. Su desventaja principal es la necesidad de un lector más preciso que el resto.

2.2.1.4 Retina

Los sistemas que utilizan la retina analizan los vasos sanguíneos situados en la parte posterior del globo ocular mediante el escáner proporcionado por radiación infrarroja. A pesar de ser un sistema que cuenta con alta seguridad, resulta ser demasiado costoso y suele reservarse para determinadas instalaciones militares.

2.2.1.5 Geometría de la mano

Analizando aspectos como las medidas y colocación de los dedos, así como sus áreas, además de otros parámetros, realiza una imagen en tres dimensiones de la mano completa que puede ser comparada con otras del mismo tipo almacenadas en una base de datos. Cuenta con un lector de un espacio considerable, pero que capta fácilmente los parámetros que necesita.

2.2.1.6 Firma

Para introducir un registro de una firma en una base de datos, se solicita que se realice una serie de veces y de esta manera queda constancia, no sólo de la geometría, sino también del tiempo requerido y la velocidad de ejecución. Estos parámetros serán utilizados más tarde para comparar con otros previamente introducidos y validar o no la identificación.

2.2.1.7 Voz

Los espectros de la frecuencia de la voz son tan distintivos como los rasgos de una huella dactilar, pero no tan distintos como para obtener un sistema fiable del todo. El hecho de que se recojan sonidos concretos para realizar la comparación, hace igualmente posible su copia mediante la reproducción de una cinta grabada. Sin embargo, los recientes avances en esta técnica, le están proporcionando más fiabilidad. [1] [5] [6]

2.2.1.8 Venas de la mano

La captación de los patrones que forman las venas mediante el escaneo con rayos infrarrojos proporciona el reconocimiento con este sistema, y al estar los vasos sanguíneos bajo la piel, dificulta la suplantación.

En la Figura 2-2 se establece una comparativa entre los distintos sistemas biométricos.

SISTEMAS BIOMÉTRICOS	HUELLA DACTILAR	RECON. FACIAL	IRIS	RETINA	GEOMETRÍA DE LA MANO	VOZ/FIRMA	VENAS DE LA MANO
Facilidad de Uso	Alta	Media	Media	BAJA	Alta	Alta	Alta
Fiabilidad	Alta	Alta	Muy alta	Muy alta	Alta	Alta	Media
Seguridad	Media	Alta	Alta	Muy alta	Alta	Media	Media
Precio	Bajo	Alto	Alto	Alto	Medio	Bajo	Medio

Figura 2-2 Resumen de sistemas biométricos.

2.2.2 Tarjetas magnéticas

Las tarjetas con banda magnética poseen una banda oscura, compuesta de partículas ferromagnéticas envueltas en una matriz de resina, capaz de almacenar cierta cantidad de información mediante una codificación que polariza estas partículas. La banda magnética es leída mediante contacto físico gracias al fenómeno de la inducción magnética.

La información en estas bandas se puede organizar mediante diferentes pistas, reguladas por estándares internacionales: ISO-7813 (para las pistas 1 y 2) e ISO-4909 (para la pista 3). En función del nivel de seguridad deseado (fuerza electromagnética requerida para su codificación), se establecen tarjetas de baja y alta coercitividad. Un ejemplo de bandas magnéticas de alta coercitividad son las que se establecen de fábrica en las Tarjetas de Identificación Militar, aunque no estén grabadas.

Es muy común el uso de tarjetas de policloruro de vinilo (PVC) con bandas magnéticas en sistemas de control de accesos debido a su bajo costes. [7]

2.2.3 Códigos de lectura óptica

Código de barras: Aunque de disposición similar a la de banda magnética, en su lugar posee un código de barras, que es un conjunto de líneas blancas y negras verticales de distinto grosor, que ordenadas a lo largo de una secuencia, crean un patrón único que puede ser leído por una luz. No existe desgaste al no haber contacto con el lector en cuestión. El hecho de que no se deterioren excesivamente por el uso y que su grabación sea muy económica, constituyen sus mayores ventajas. Por otro lado, su falsificación es relativamente sencilla. [1]

QR (Quick Response code): Se trata de un código bidimensional cuya información se haya codificada y configurada dentro de una figura cuadrada. Se pueden diferenciar a simple vista gracias a sus característicos cuadrados situados en las esquinas superiores e inferior izquierda. Posee las siguientes particularidades:

- Maneja datos de tipo alfanumérico, símbolos, códigos binarios, códigos de control, Kanji, Hiragana y Hiragana.
- Es capaz de almacenar hasta 7.000 números y cerca de 4300 caracteres (100 veces más que un código de barras).
- Ocupa poco espacio, y como mínimo, ha de abarcar 10 mm cuadrados.
- Posee patrones de posicionamiento y son de lectura omnidireccional, factores que contribuyen a que su lectura sea rápida.
- Poseen correctores que proporcionan que aunque el código se haya dañado hasta un 30 %, no se pierda nada de información.
- Su área se puede dividir en distintas porciones de datos independientes, hasta un número de 16.
- Pueden ser leídos por la mayoría de dispositivos móviles que incluyen cámara y de manera gratuita. [8]

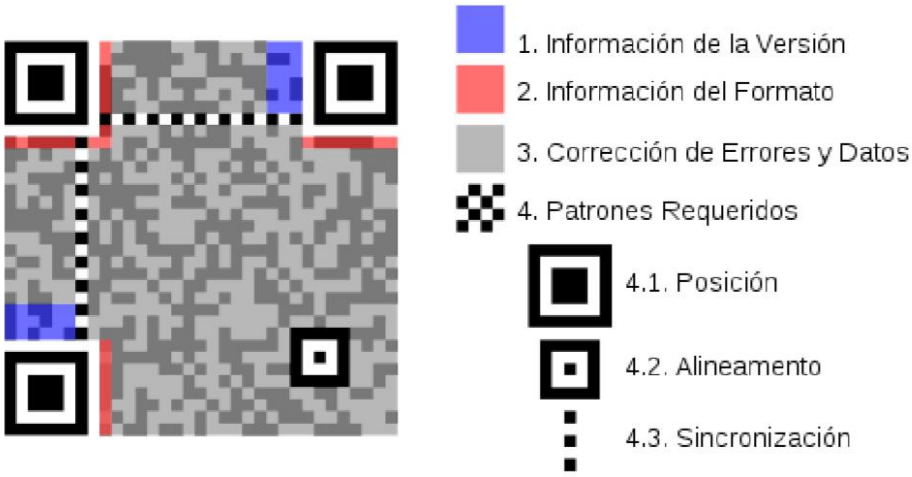

Figura 2-3 Distribución de código QR. [9]

BIDI: Son similares, pero carecen de las casillas cuadradas en las esquinas. Otra diferencia es que, frente a los QR, los códigos BIDI no son gratuitos, su uso suele ser comercial y para su generación es necesario obtener una aplicación de pago. En España, sólo tienen licencia para utilizarlos las compañías Vodafone, Movistar y Orange. [9]

Figura 2-4 Código BIDI. [9]

2.2.4 Tarjetas inteligentes

Este tipo de tarjetas encapsulan un chip capaz de almacenar información con datos de todo tipo, desde sanitarios y personales a bancarios. Se comunica con los lectores mediante unos contactos exteriores y tiene mayor capacidad que la banda magnética, además de no estar sujeta a un desgaste tan pronunciado.

Las tarjetas inteligentes de contacto disponen de unos contactos metálicos visibles que han de ser insertadas en el lector correspondiente para poder operar con ellas. El lector alimenta eléctricamente a la tarjeta mediante estos contactos metálicos y es capaz de transmitir los datos oportunos. Los lectores transmitirán esos datos a un ordenador, donde podrán ser procesados.

En función de las capacidades del chip, se clasificarán en tarjetas de memoria, que sólo almacenan datos, microprocesadas, que además serán capaces de ejecutar aplicaciones, y criptográficas, que a las anteriores funciones se le suma la capacidad de ejecutar algoritmos de cifrado.

Si atendemos a su interfaz, diferenciamos entre:

Con contactos: disponen de 8 contactos metálicos (Figura 2-5) visibles que han de ser insertadas en el lector correspondiente para poder operar con ellas. El lector alimenta eléctricamente a la tarjeta mediante estos contactos metálicos y es capaz de transmitir los datos oportunos. Los lectores transmitirán esos datos a un ordenador, donde podrán ser procesados. [10]

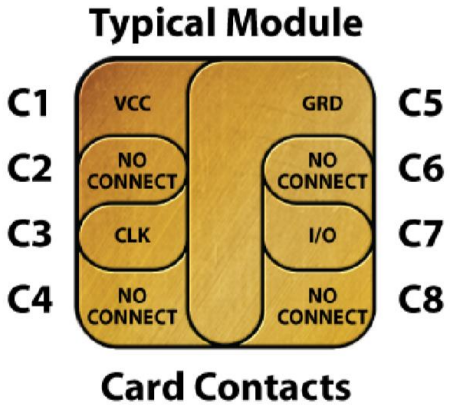

Figura 2-5 Tarjeta Inteligente con contacto [11]

Sin contactos: utiliza los tags RFID. Estos, son pequeños dispositivos que se incorporan en la tarjeta y le permiten transmitir y recibir peticiones por radiofrecuencia. Están compuestos por una antena, un transductor radio y un chip. Mediante la antena, el chip puede transmitir la información de identificación que ha sido almacenada en su memoria, que puede ser:

- De lectura, si el código se establece durante su fabricación y no puede ser modificado a posteriori.
- De lectura y escritura, en la que la información podría ser modificada por el lector.

- De anticolisión, donde estos adhesivos permiten que el lector identifique varios a la vez, y depende de la cobertura de éste.

Las etiquetas RFID [12] pueden ser activas, si requieren de alguna fuente de alimentación interna, o pasivas, si se sirven de la energía que le suministra el lector.

Ventajas principales de este sistema:

- Sus componentes no sufren desgaste ni interfiere en gran medida la suciedad, luego permite reducir el coste de su mantenimiento.
- Fácil utilización.
- Rapidez en la transferencia de datos.
- No se ve afectado por otras tecnologías, por lo que se podría hacer una tarjeta multifunción: banda magnética, chip de contacto, etc. [13]

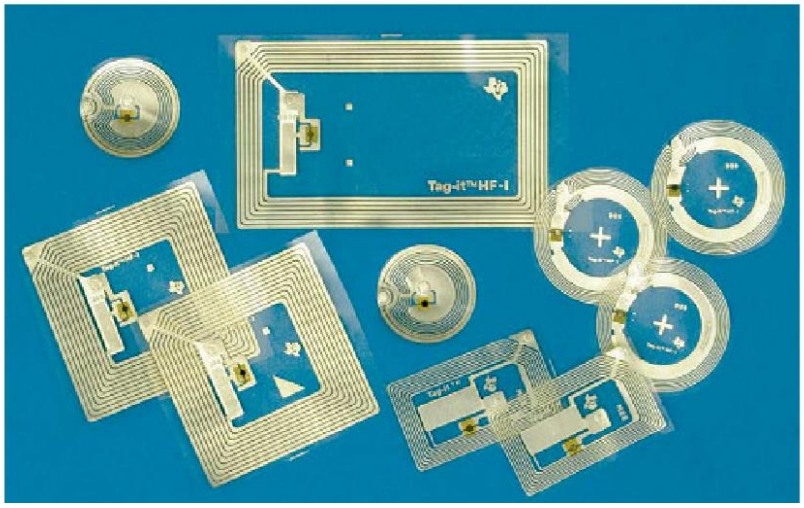

Figura 2-6 Tags RFID [14]

2.2.5 Clave personal

Mediante la introducción manual de un código (numérico o alfanumérico) que posee cada persona se da la identificación para la entrada a una instalación. Se trata del más económico, e igualmente el más inseguro ya que no sólo puede ser olvidada la clave, sino recordada por cualquier persona ajena que llegue a tener acceso a ella. Para la instrucción de la clave suelen emplearse teclados numéricos específicos.

2.2.6 Sistemas combinados

Dependiendo del grado de seguridad requerido, si se quisiese aumentar, se podrían buscar combinaciones entre los lectores anteriormente expuestos en este apartado. De hecho, uno de los objetivos del presente trabajo es la redundancia en la petición de información a un usuario a la hora de identificarse. Esto es, pedirle que introduzca la huella dactilar, además del DNI. De esta manera logramos una identificación segura y difícilmente engañable. [1]

2.3 Software de Control de accesos

El software es el equipamiento lógico e inmaterial de un ordenador y se define como el conjunto de eventos, procedimientos y normas asociados con la operación de un sistema de computación. Dentro de los tipos de software, los más trascendentes son los de base, donde se permite que el consumidor tenga control sobre el hardware, que son los componentes físicos. El software incluye

todas las aplicaciones informáticas y su desarrollo depende de un conjunto de símbolos y reglas sintácticas y semánticas llamado lenguaje de programación. [15]

El software se puede configurar acorde con las necesidades y requerimientos de la empresa en cuestión. De esta manera, pueden tenerse en cuenta factores como horarios de entrada y salida, número de empleados, gestión de visitas o distintos niveles de usuarios. Esto hace que exista la tendencia por parte de las empresas de diseñar sus propios softwares para el control de accesos. Estos, suelen organizarse de tres maneras:

- Arquitectura *Stand-Alone*: Todos los programas y servidores se encuentran en la misma máquina.
- Arquitectura *Cliente-Servidor*: Las bases de datos centrales se sitúan en el servidor y las interfaces de usuario (que pueden ejecutarse más de uno a la vez) se hallan en máquinas separadas, relacionándose con el servidor mediante la misma red.
- Arquitectura *Servidor Web*: Similar a la anterior, pero el servidor se encuentra en una red para la cual hay que acceder a internet para obtener la información de las BBDD.

2.3.1 Arquitectura Stand-Alone

El conjunto de servidores y programas está implementado en la misma máquina. Bases de datos, la aplicación y conexiones de los lectores y controladores se operan desde el mismo operador. Se suelen implementar en empresas de reducido tamaño donde se requieren usuarios de la misma categoría y no se necesita que se ejecute más de un proceso a la vez. Tienen como ventajas la simplicidad de su implementación y operación.

2.3.2 Arquitectura Cliente-Servidor

Las BBDD se instalan con el software en un servidor y la interfaz de usuarios donde interactúan los clientes se ejecuta desde una máquina separada. Por este motivo, se tendrán que instalar tantas máquinas como operaciones se quieran realizar al mismo tiempo y ahí reside su principal ventaja frente a la anterior arquitectura, sin que las distintas aplicaciones puedan interferir entre ellas. Por ejemplo, puede ser instalada en una oficina con dos porterías en la que se quieran llevar al mismo tiempo, por un lado los trabajadores, y por otro, las visitas. Por ello, a expensas de ser más flexibles, resulta más compleja su instalación.

2.3.3 Arquitectura del Servidor-Web

La arquitectura Servidor-Web es similar a la anterior en algunos aspectos, pero la comunicación entre el cliente y el servidor se realiza mediante un protocolo HTTP o HTTPS (HTTP seguro), luego es necesario que se acceda a la información mediante un navegador y no hace falta que los clientes estén instalados directamente en el sistema. [4]

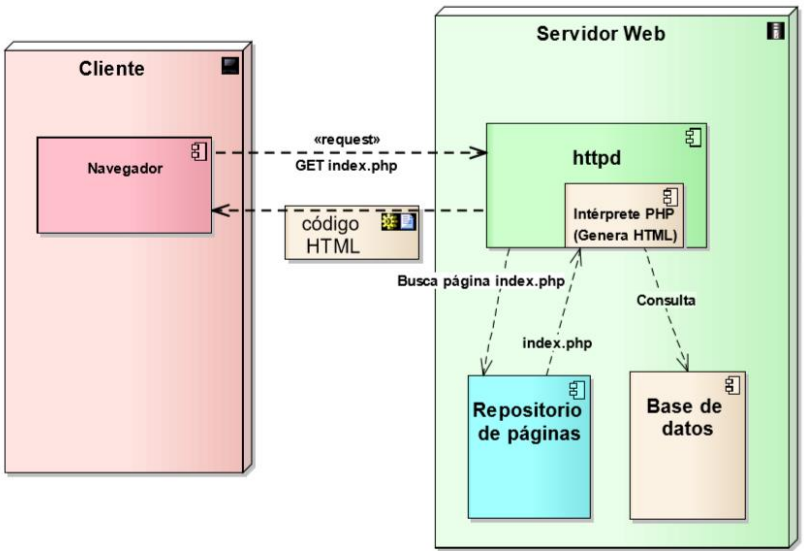

Figura 2-7 Arquitectura Servidor-Web. [16]

2.4 Sistemas de Control de accesos en las dependencias de la Armada

Como es lógico, los sistemas de control de accesos no sólo están presentes en el entorno civil, y resulta imprescindible, por necesidad en materia de control y seguridad de la información y acceso, que se implementen en instalaciones militares.

El sistema de control de accesos implementado en el Cuartel General de la Armada fue implementado en 2013, tras barajarse varias opciones. Por un lado, la Armada estudió la posibilidad de utilizar como credencial la Tarjeta de Identificación Militar (TIM), ya que simplemente con grabar un código particular en su banda magnética (que viene de fábrica) podría servir para validar identidad y dar acceso a esta dependencia. Se terminó descartando por propia normativa [17], ya que la TIM, aparte de tener el propósito de identificar, no posee chip interno, y se estableció que para entrar en dichas instalaciones la banda magnética no cumplía con los requisitos de seguridad apropiados.

El proyecto que se estableció fue el propuesto por la empresa *Desarroya*, que implementaba un conjunto de servidores unidos por fibra óptica y que incluía un programa de control de accesos en los que se monitorizaban las entradas y salidas, así como los horarios. Las tarjetas que permiten este acceso son tarjetas inteligentes que poseen un chip interno con varias celdas, en las que se pueden programar, por ejemplo, quién tiene derecho o no a un descuento en la cafetería del edificio.

Por otro lado, en el Estado Mayor de la Armada (EMA), además de utilizarse unas tarjetas similares a las del CGA, se utilizan otras para dar destino a determinados servicios y accesos. El jefe del destino o superior a cargo posee una tarjeta llamada PKI (*Public Key Infrastructure*), con otro chip que se introduce en un lector, generalmente integrado en un teclado y ofrece tanto autentificación como validación de credenciales y acceso a datos además de proporcionar una firma digital.

2.4.1 Control de accesos en los BAM

El sistema de control de accesos más moderno que se está desarrollando en la Armada es el que está implantado en los BAM, llamado Sistema de Control Personal y Material (SCPyM). Se trata de un sistema Stand-Alone basado en un ordenador personal que durante la estancia en puerto se ubicará en el puerto de guardia. Este ordenador estará conectado a la red Administrativa del Buque, que será el vínculo de unión entre la aplicación de Control y el Servidor de Base de Datos, así como con el Sistema Integrado de Control de Plataforma (SICP), pero éste último estará aislado del resto de equipos conectados a la red administrativa por un Firewall.

El SCPyM proporcionará las herramientas necesarias para controlar la entrada y salida del buque de personal y material, e informará al SICP para la gestión del Sistema de Localización de Personal (SLP).

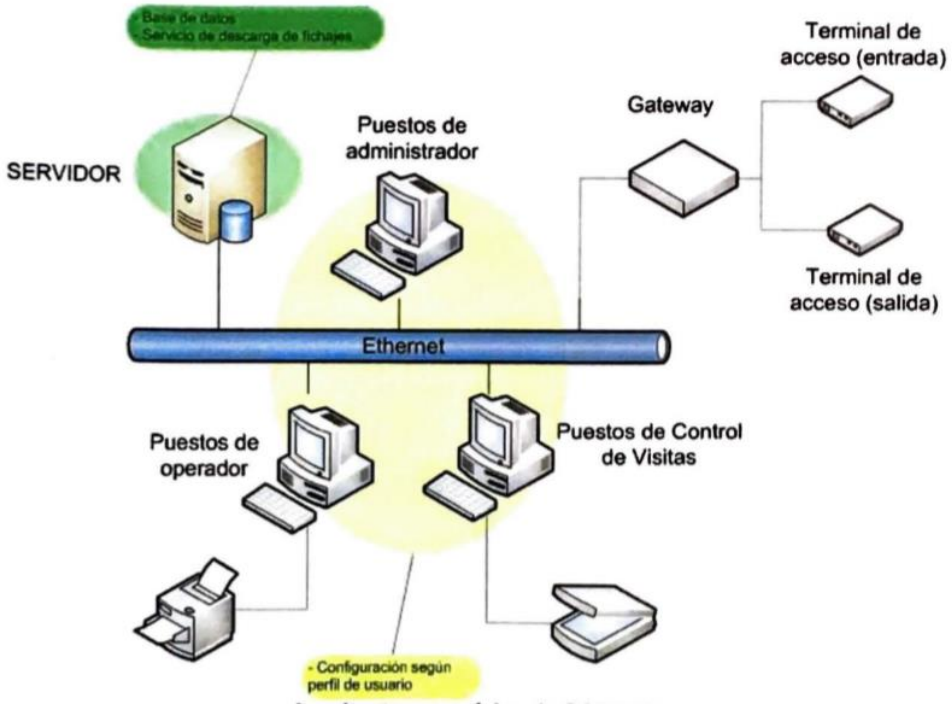

Figura 2-8 Arquitectura genérica de SCPyM. [18]

La infraestructura hardware está formada por terminales para control de accesos de personas, terminales de sobremesa, puestos de ordenador, lector de código de barras e impresora de código de barras.

El software permite gestionar el control de accesos individualizado de personas, visitas y material. Se trata de una aplicación desarrollada en Microsoft.NET, llamada SOCRATES [18], que admite la definición de diferentes roles de usuario, con permisos específicos, que configuran de forma dinámica las funciones accesibles al operador y que son requeridas para el sistema. Estas utilidades son las siguientes:

- Configuración del Sistema: permite al usuario configurar los diferentes aspectos de la aplicación, y su uso corresponde al perfil de administrador del sistema.
- Gestión de Personas: permite el alta, baja y modificación del personal y de sus datos asociados, tales como tarjetas o avisos de seguridad.
- Gestión de Visitas: maneja toda la funcionalidad del proceso de visitas, incluyendo la programación de visitas y la gestión del historial de visitas.
- Gestión de Tarjetas: permite la configuración, impresión y grabación física de los soportes necesarios para ser usadas en las lectoras de acceso, por parte de personal y visitantes.
- Gestión de Accesos: gestión de los diferentes elementos que intervienen en el sistema de control de acceso.
- Gestión de Llaveros: gestiona el registro de llaves del sistema. También se permite llevar un control de las entregas, devoluciones y recogidas de las mismas.
- Gestión de Material: permite la gestión de entrada y salida de material.
- Control de Presencia: permite llevar un control sobre las personas que se encuentran dentro o fuera del recinto y ofrece una visualización en tiempo real de los pasos de tarjeta realizados en la instalación.

- Gestión de Logs: tiene como objetivo gestionar y analizar el log generado por las acciones de los usuarios durante la ejecución de la aplicación. [18]

2.4.2 Sistemas de control de acceso en otras academias

Por último, en las academias militares de oficiales del ejército de Tierra, del Aire y en la Academia Central de la Defensa también existe un sistema de control de accesos, consistente en un lector de bandas magnéticas y una puerta cuya apertura se activa al obtener la validación. Esta validación es fruto de una consulta a una base de datos en la que están introducidos los datos de los alumnos y al compararse y autorizar la entrada, ésta se registrará para cambiar su estado para el sistema como "dentro" o "fuera". En cuanto a la credencial utilizada, se trata de una tarjeta de banda magnética, exclusiva para este propósito. Se ha de llevar siempre que se sale de la academia y sin la cual necesitaría la autorización de un centinela de guardia para acceder.

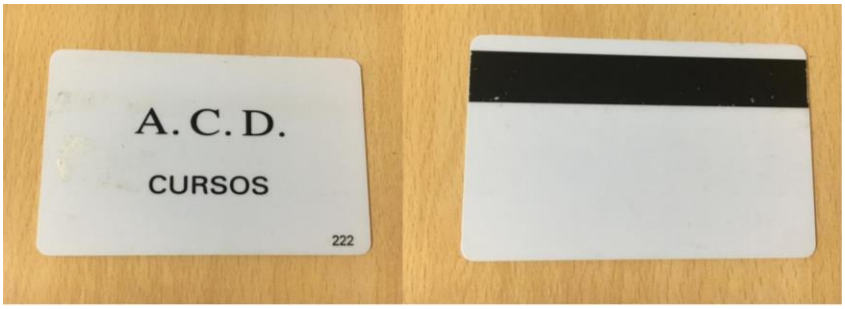

Figura 2-9 Credencial empleada en la Academia Central de la Defensa.

2.4.1 Control de accesos en la Escuela Naval Militar

En cuanto al sistema de control de accesos en la academia, se ha de comentar que no se encuentra automatizado. Está compuesto por una puerta con mecanismo hidráulico de apertura, en la que, una vez se para un transeúnte o un coche en las inmediaciones, sale el centinela, valida su identidad, y permite el acceso al individuo. En el caso de la salida, no se realiza ninguna validación salvo si el centinela detecta que alguien está saliendo con una tarjeta temporal.

2.4.1.1 Validación

La validación de la identidad se realiza de distintas maneras, dependiendo de la ocupación y de la manera en que vayan a entrar.

Personas que entran a pie:

Por un lado, en el caso del personal militar, para acceder a la academia necesitan enseñar al centinela la TIM (Tarjeta de Identificación Militar). Todo el personal que posea una TIM podrá entrar en la Escuela Naval Militar.

En el caso del personal civil con trabajo en las dependencias de la academia, la validación de identidad se realiza mediante una tarjeta de cartón con varios datos inscrita en ella, como el DNI, el nombre completo, una foto y, recientemente, un holograma. Es la única credencial que se les suministra y no hay otro documento que tengan que enseñar para acceder. No obstante, tal y como indica el reverso de la tarjeta, el hecho de portar la tarjeta no exime de la obligatoriedad de identificarse con el DNI para su comprobación cuando sea requerido.

Personas que entran en coche:

En el caso de entrar en automóvil, todos los ocupantes del vehículo tendrán que enseñar alguna de las credenciales anteriormente mencionadas, además de poseer en un lugar visible un pase para vehículo que proporciona la Oficina de Seguridad de la academia. En el caso de no disponer del citado pase, los centinelas tendrán que proporcionarle uno temporal, si la entrada ocurre antes del ocaso, y si no, esa noche no tendrán autorización de aparcar dentro de la academia un vehículo sin pase.

2.4.1.2 Registro

En relación al registro llevado del personal que accede a la academia, se ha de especificar que sólo se lleva en dos casos únicos.

Por un lado, en el caso de los alumnos, que con el fin de tener el control de quienes están dentro y quienes fuera, una vez accedan a la academia, se han de apuntar en un cuaderno llamado "Libro de Francos", donde se les actualiza su estado dependiendo de si salen o entran. Este libro, que está compuesto por dos tomos, uno para aspirantes y otro para Guardiamarinas y Alféreces, se halla bajo custodia del Brigadier de Guardia, y cuando regresa el último aspirante, pasa a estar bajo custodia del alumno Subalterno Comandante de la Guardia hasta que termine el punto nocturno de guardia del día siguiente.

Por otro lado, en el caso del personal civil que no trabaja con regularidad en la escuela y que por tanto no dispone de la tarjeta mencionada en el apartado anterior, este es registrado en la garita de la puerta de Carlos I. Tendrán que apuntar su DNI y nombre y se les facilitará un pase personal individual, que tendrá que ser devuelto cuando finalicen los asuntos que les hicieran acceder al recinto, como pueden ser visitas o pesca en el muelle de Cruceros.

El sistema de control de acceso planteado en este TFG estaría destinado a mejorar el control de accesos de la academia, mediante el uso de credenciales que todas las personas deben portar como son la huella dactilar y el DNI Electrónico. El sistema permitiría llevar un registro acerca de la situación de los usuarios (dentro/fuera) que quisiesen acceder a la Escuela Naval Militar, desde alumnos y profesores, hasta las entradas esporádicas, ofreciendo una situación clara y más segura de esta instalación militar.

2.5 Ley de protección de datos

Puesto que se van a manejar ciertos datos de los usuarios, se decidió establecer un apartado de este TFG para estudiar cuáles son y cómo han de tratarse los datos personales de los distintos individuos antes de ser almacenados en la base de datos.

2.5.1 Datos personales

Los datos de carácter personal incluyen cualquier información referente a personas que puedan ser físicamente identificables gracias a una credencial (por ejemplo localización, DNI o nombre) o a un rasgo físico, genético, económico, o cultural del individuo.

Dependiendo del tipo, podrán hacer alusión a la identificación propia, estado laboral y financiero, o incluso de salud, y además, existen otras categorías de datos denominadas especiales, que serán explicadas en el apartado 2.5.4.

2.5.2 Principios relativos al tratamiento de datos personales

- Principio de licitud, lealtad y transparencia: Han de ser tomados acorde al RGPD (Reglamento Europeo de Protección de Datos), que excluye que sean tratados de manera desleal y ello obliga que se pueda exigir completa transparencia acerca del manejar de estos datos.
- Principio de limitación de la finalidad: El objetivo o finalidad para la que se recogen los datos debe estar claramente expresada, e igualmente debe ser legítima y aprobada por el reglamento.
- Principio de minimización de datos: Los datos serán los necesarios para cumplir estrictamente con la finalidad, no se pueden coger por supuestos futuros o porque se consideren útiles.

- Principio de exactitud: Los datos tienen que ajustarse a la realidad y si cabe, deben estar actualizados, adoptándose medidas para su rectificación en el caso de que se requiriese.
- Principio del plazo de conservación: Con relación al principio de limitación de la finalidad, ello supone que una vez se cumpla esta finalidad, los datos no han de ser retenidos más tiempo del necesario, así como se debe informar de este periodo.
- Principio de integridad y seguridad Como se explicará en el siguiente subapartado, una vez se compilen los datos, se tiene que poder garantizar la seguridad en los mismos y protegerlos ante cualquier riesgo.
- Principio de responsabilidad proactiva: Los tomadores de datos han de cumplir todos los principios anteriormente expuestos conforme al RGPD. [19] [20]

Figura 2-10 Factores relativos a la protección de datos. [19]

Deben tenerse en cuenta los siguientes aspectos si atendemos a la legitimación en el trato de los datos personales que se facilitan:

- Consentimiento en el marco de un contrato:
  -La autorización en para el tratamiento de datos personales se hace mediante un contrato, que declarará la aceptación del individuo que firma.
  -No se pueden utilizar para otros fines que no estén expresamente firmados de antemano, por ejemplo, enviar publicidad no deseada.
- Se requiere una declaración afirmativa, ya que el Reglamento General de Protección de Datos no permite el "consentimiento tácito".
- Se requiere autorización de los tutores a menores de 14 años. [20]

2.5.3 Seguridad

Se han de adoptar una serie de medidas al recabar datos personales, con el fin de protegerlos. Algunas de las citadas son:

- Evitar accesos no autorizados.
- Realizar copias de seguridad.
- Cifrado de datos.
- Realizar un control del almacenamiento de los usuarios, de los soportes y del acceso a los datos.

Igualmente, todo aquel que trate con datos personales, ha de cumplir el deber de secreto.

En aplicación del RGPD, aquel que esté en disposición de tratar datos personales ha de evaluar si puede cumplir con el requisito de seguridad, sin ninguna clase de fuga, y en el caso de que tuviese conciencia que efectivamente existe, tendrá que comunicarlo al dueño de los datos. [21]

2.5.4 Categorías especiales de datos

Entre las categorías especiales de datos, se encuentran:

- Aquellos que denotan el origen étnico o racial, las opiniones políticas, las convicciones religiosas o filosóficas.
- Los datos biométricos dirigidos a identificar de manera unívoca a una persona.
- Los datos relativos a la salud.
- Los datos relativos a la vida sexual o la orientación sexual de una persona. [19]

2.5.5 Protección de datos en la Escuela Naval Militar

La academia, como cualquier otra administración pública, está sujeta a esta ley. Según la Orden DEF/1342/2015, se autoriza la creación de los ficheros de datos de carácter personal en el ámbito de la Dirección de Enseñanza Naval de la Jefatura de Personal de la Armada. Como se puede apreciar en el anexo de esta orden, en el apartado 14, se concreta el "fichero de Control de Acceso de escuelas y centros de la DIENA [22]. En este, deberán figurar los siguientes tipos de datos:

- 1º Identificativos: nombre y apellidos, NIF/DNI, dirección, teléfono, imagen, correo electrónico profesional y particular, datos del vehículo: matrícula, modelo, color.
- 2º Circunstancias sociales: situación militar.
- 3º Detalles del empleo: cuerpo/escala, categoría/grado y puesto de trabajo. Sistema de tratamiento: mixto.

3 SOLUCIÓN PROPUESTA

3.1 Evaluación de alternativas

Considerando todos los puntos anteriormente descritos a lo largo del trabajo se ha realizado un análisis para determinar qué tipo de sistema podría ser el más adecuado en la academia.

3.1.1 Manual/semimanual/automático

Por un lado, debido a las necesidades expresadas en la motivación, sería conveniente establecer un sistema semimanual, en el que, además de existir la autorización de la entrada por parte de un centinela que ha de estar permanentemente vigilando en la puerta, exista un software de sistema de control de accesos con el que se aseguren las identificaciones y los registros de las entradas y salidas. Por tanto, no se busca modificar la apertura de la puerta mediante un sistema automático, ya que se considera más seguro que sea el propio centinela el que tenga la decisión de abrirla tras validar la identidad.

3.1.2 Cableada/inalámbrica

La manera en que se debe alimentar el terminal debe ser cableada, principalmente por comodidad y porque se establece que en el caso de que hubiese algún inconveniente para su utilización, como un apagón o cualquier fallo en el suministro eléctrico, tendría que estar igualmente un centinela comprobando las entradas, y no habría un gran entorpecimiento en el flujo de entradas y salidas.

3.1.3 Autónomo/online/mixto

Puesto que uno de los objetivos que se estableció al principio del proyecto era la presentación de los registros y datos y en tiempo real, se elige la opción de que sea online. El sistema que se pretende realizar necesita un ordenador para que se puedan manejar los datos de los usuarios de manera sencilla y que se puedan dar de alta, borrar o incluso comprobar la situación de ese usuario.

3.1.4 Tipo de tecnología de identificación

Para atender a las necesidades y objetivos del sistema de control de acceso deseado, se barajaron las distintas tecnologías distintas de identificación existentes en el mercado. Como requerimientos principales se resaltan:

- Efectuar una implementación asequible económicamente.
- La inclusión de identificadores o credenciales comunes para todos los usuarios y que además se lleven encima con normalidad.
- Rapidez a la vez de precisión y seguridad.
- De fácil montaje, sin ser complejo ni aparatoso, ya que se tiene que acercar al usuario en el caso de que esté subido a un coche.

De esta manera se van descartando por su alto coste algunos sistemas biométricos como el de retina, iris, reconocimiento facial o venas de la mano, que a pesar de requerir credenciales portables en todo momento, sobresalen del precio deseado. Esto nos deja con los lectores de huella dactilar como elemento sencillo, portátil, relativamente fiable y seguro, y económico.

Dado que como se comentó anteriormente en el punto 2.2.1.1, es un sistema sujeto a imperfecciones en las yemas de los dedos, y que en determinadas ocasiones, debido a la posición y al estado de la credencial, puede dar falsa negación a la hora de validar una identificación. Por otro lado, el motivo fundamental por el que se deshecha esta tecnología biométrica es por la falta de autorización. El hecho de registrar y tratar datos biométricos (datos de carácter especial) no está contemplado en la permisión disponible comentada en el punto 2.5.5. Por tanto, entrañaría dificultades legales conforme a la Ley Orgánica de Protección de datos, que se escapan del ámbito de este trabajo. Por tanto, se deciden implementar tecnologías de identificación que no sugieran el trato de datos de carácter especial, aun suponiendo una reducción de fiabilidad.

En el marco de esta decisión, y habiendo desechado el lector de huellas dactilares, se estudia qué otra tecnología formará parte del sistema a implementar. Se examinan sistemas como el de lectura de banda magnética, que aunque sujetas a un desgaste superior que la de las tarjetas inteligentes, se trata de un sistema sencillo y rápido que podría dar solución a la inclusión de nuevos usuarios en el registro. En cuanto a las tarjetas con chip, se determina que quizá era costoso realizar una tarjeta con un chip para cada individuo. Por ejemplo, hay individuos que no suelen entrar frecuentemente en la academia, como en el caso de pescadores y visitas y a los que no se les debería hacer una tarjeta, porque no tienen por qué ser los mismos. Uno de los requerimientos es que se trate de aplicar las mismas credenciales para todo el mundo, por tanto se llega a una conclusión: como el Documento Nacional de Identidad es de necesaria obtención a partir de los 14 años y obligatorio exhibirlo cuando sea requerido por un Agente de la Autoridad según el Real Decreto 1553/2005, se vio oportuno utilizarlo como método de identificación, ya que al igual que la huella dactilar anteriormente descartada, se lleva siempre consigo.

3.2 Sistema propuesto

El sistema ideado, tendría varios métodos posibles de identificación, mediante la petición de diferentes credenciales:

- El DNI de manera visual e introducción a mano para comprobar si se halla registrado.
- Mediante la introducción del DNI en un lector de DNIE.
- Por comodidad, si la entrada en el recinto va a ser transitoria, está la opción de utilizar una tarjeta de banda magnética para registrarse de manera inmediata en la BBDD.

Para implementar el sistema de control de accesos anteriormente propuesto se emplearan varios lectores conectados a un ordenador, que no solo servirán para validar las credenciales mediante la comparación de las mismas con unos ya introducidos previamente en una base de datos, sino también para registrar todos esos accesos con el fin de ser consultados en cualquier momento. El sistema será aplicable a todo el personal que quiera acceder a la academia. En la Figura 3-1 se detalla el esquema de funcionamiento del sistema propuesto.

El usuario, tras exhibir sus credenciales ante alguno de los lectores, sería validado contra una BBDD. Si la validación es positiva, se le permitiría el acceso y se realizaría en la BBDD una anotación a modo de registro. La aplicación desarrollada también permitiría labores de gestión como al alta, modificación y baja de usuarios, así como diversas posibilidades de consulta.

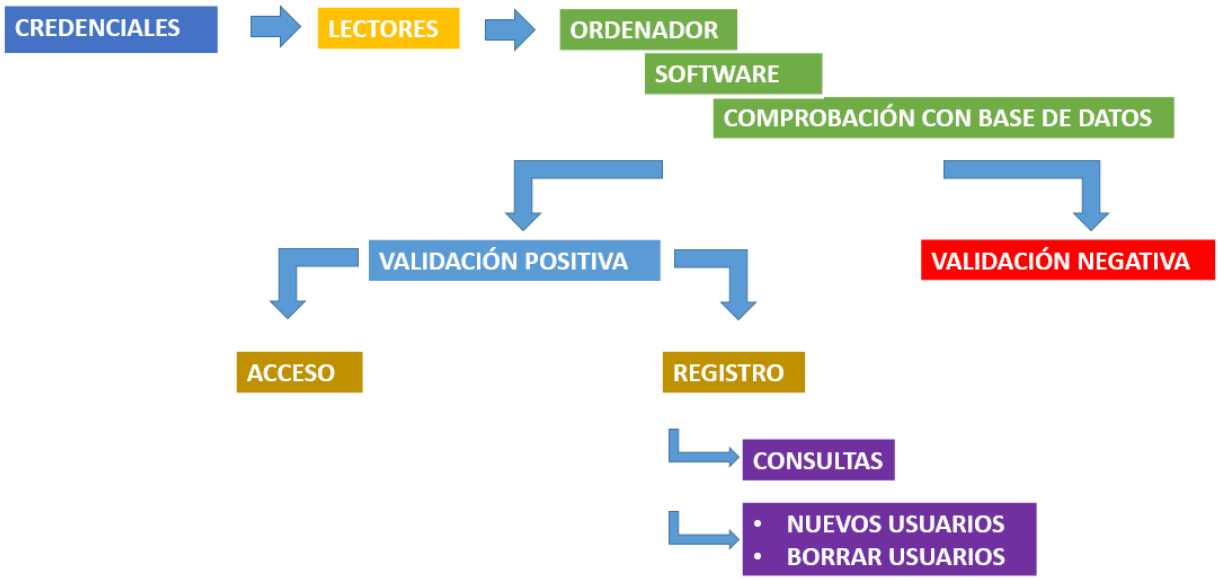

Figura 3-1 Sistema de control de accesos propuesto.

4 DESARROLLO DEL HARDWARE/SOFTWARE

4.1 Hardware

4.1.1 Componentes

Los componentes del sistema propuesto son:

- **Lector de bandas magnéticas de la marca *Yosoo*, modelo MSR90 con entrada USB.**

Figura 4-1 Lector de tarjetas magnéticas.

- **Lector de tarjetas con chip *Zoweetek*, modelo ZW-12026-5 con entrada USB.** Será el lector utilizado para la lectura del DNI.

Figura 4-2 Lector de tarjetas con chip.

- **Ordenador *Acer* con procesador Intel Core i3 a 1.90GHz, 4GB de RAM** con la distribución Ubuntu instalada.

Dichos componentes se han obtenido mediante la compra directa y no han sido modificados externamente para su instalación en el proyecto, aunque sí se han modificado ciertos parámetros para personalizar los datos que devuelven. Un ejemplo es el caso del lector de DNIE, que modificando el número de caracteres visibles que se desean observar, se pueden extraer de un gran conjunto de caracteres, información útil como por ejemplo el número del DNI.

4.2 Medios empleados

4.2.1 Sistema operativo

El sistema operativo instalado para la realización del proyecto ha sido Linux, en concreto la distribución Ubuntu. Se escogió aprovechando que venía con Python instalado y con la herramienta SQLite Manager para el tratamiento de bases de datos.

4.2.2 Lenguaje de programación

- El sistema utiliza el lenguaje de programación interpretado Python 3.
- Asimismo, puesto que se iba a trabajar con bases de datos, se escoge el sistema de gestión de bases de datos SQLite 3, de dominio público.
- SQL (*Structured Query Language*) es un lenguaje utilizado para gestionar y obtener información de sistemas de gestión de bases de datos.

4.2.3 Herramientas empleadas

Las herramientas empleadas para desarrollar y probar el código han sido el entorno de programación Spyder y los gestores y exploradores de bases de datos DB Browser y el SQLite Manager.

- Spyder es un entorno de desarrollo interactivo para múltiples lenguajes de programación (denominados “kernels”), de código abierto, que facilita la edición y prueba del código que se está diseñando. Posee funciones avanzadas de depuración e interactivas. En nuestro caso ha sido utilizado con un “kernel” de Python 3.
- DB Browser es un una herramienta de código libre pensada para crear, diseñar y editar bases de datos compatibles con SQLite. Muy útil a la hora de introducir los datos directamente en una tabla de una base de datos, al igual de disponer de una interfaz dinámica y cómoda.
- SQLite Manager es otro sistema de gestión de bases de datos cuyas funciones se asemejan a las del DB Browser, pero éstas son de introducción manual, luego a pesar de tener una interfaz menos intuitiva, posee una lista completa de todas órdenes, que puede ser consultada en caso de alguna duda.

4.3 Diseño del Software

4.3.1 Base de datos

SQLite es una biblioteca de C que implementa un motor de base de datos SQL. Al ser un software libre, existen una gran cantidad de componentes y librerías que pueden interactuar con él mediante una gran variedad de lenguajes de programación como Python, Java, Perl o C#. Sus particularidades hacen posible que sea empleado en tareas como la integración de una BBDD dentro de una aplicación, debido a su facilidad de configuración, aunque posee limitaciones de escalabilidad por motivos de capacidad (del orden de Terabytes).

Para la creación de la base de datos utilizada primero se instaló *sqlite3* y mediante los comandos estándar que proporciona esta herramienta se añadió una base de datos, llamada *accesosENM* y dentro se crearon dos tablas: *personal* y *registro*, relacionadas entre sí por el campo *ident.*

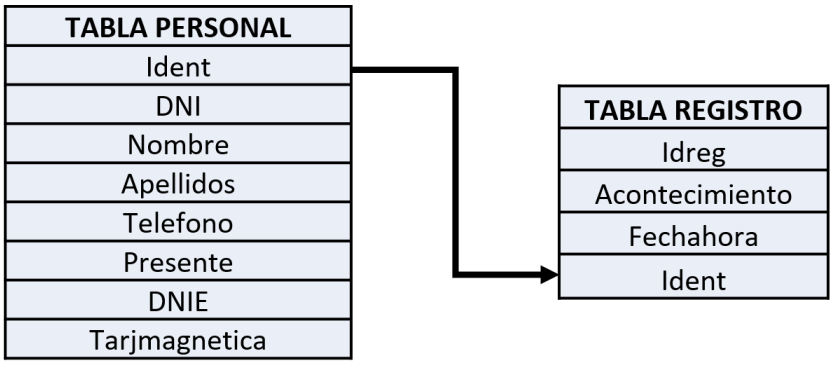

Figura 4-3 Tablas *personal* y *registro*.

4.3.1.1 Tabla Personal

Es la tabla donde se almacenan los datos de los usuarios, la cual será consultada cada vez que se vaya a realizar un tránsito, y en la que se insertarán y borrarán registros.

La estructura de esta tabla consta de tantos registros como usuarios haya dados de alta y una serie de campos, que serán:

- **ident**, campo configurado como clave primaria y cuyos datos introducidos serán un número entero e incremental. Un ejemplo sería:

Ident
1
2
3
4

Figura 4-4 Campo Ident.

- **DNI**, un campo configurado como único, puesto que los DNIs no se van a repetir, de tipo VARCHAR, con un máximo de 9 caracteres, correspondientes a 8 números y una letra en mayúscula. Un ejemplo sería:

DNI
53587014G
54534545Y
23342351U

Figura 4-5 Ejemplo campo DNI.

- **Nombre**, un campo de tipo VARCHAR, de máximo 30 caracteres. Un ejemplo sería:

Nombre
Alejandro
Víctor
Javier

Figura 4-6 Ejemplo campo Nombre.

- **Apellidos**, un campo de tipo VARCHAR, de máximo 60 caracteres. Un ejemplo sería:

Apellido
Sánchez Cervera-Mercadillo
García Lourido
Palma Rodríguez

Figura 4-7 Ejemplo campo Apellidos.

- **Telefono**, campo de tipo entero, con un máximo de 9 números. Un ejemplo sería:

Telefono
630345304
983258433
234543454

Figura 4-8 Ejemplo campo Apellidos.

- **Presente**, un campo de tipo BOOLEAN, con valores de 1 o 0, dependiendo, respectivamente, si el usuario está fuera o dentro del recinto.

Presente
1
0
0

Figura 4-9 Ejemplo campo Presente.

- **DNIE**, un campo de tipo TEXT, que puede devolver distintos datos que especifiquemos, relativos a la información genérica contenida en el chip del DNI. De entre muchos caracteres devueltos, configurando la función ***Update,*** se puede elegir, qué es lo que queremos extraer, como por ejemplo el número de serie del DNI, el IDESP o el número de DNI. Por tanto, este campo recoge un máximo de 255 caracteres, proporcionados por el lector de DNIE. Un ejemplo sería:

DNIE
53587014G1110010120302030102ALEJANDROSANCHEZCERVERA

Figura 4-10 Ejemplo campo DNIE.

- **Tarjmagnetica**, un campo de tipo TEXT, que al igual que el campo **DNIE**, puede devolver distintos datos que especifiquemos, relativos a la información contenida en la banda magnética de la tarjeta en cuestión. Al carecer de tarjetas similares para todo el mundo, se recogerán en este campo un número determinado de caracteres diferentes propios de cada tarjeta, con los cuales se identificará esa credencial en cuestión. Tendrá un máximo de 255 caracteres, proporcionados por el lector de tarjetas magnéticas. Un ejemplo sería:

Tarjmagnetica
10000000230501000003
4440304503020520376025
ER23406020000450406

Figura 4-11 Ejemplo campo Tarjemagnetica.

Un registro entero, tendría la siguiente información:

Ident	DNI	Nombre	Apellidos	Telefono	Presente	DNIE	Tarjmagnetica
1	52587014G	Gonzalo	Ruiz Ballesteros	657364565	1	3587014G1A…	10000000230501000003…

Figura 4-12 Ejemplo registro tabla *personal*.

4.3.1.2 Tabla Registro

Es la tabla donde se lleva el control de los tránsitos realizados durante la utilización del sistema, a fin de llevar un control indiscutible sobre los cambios que se han realizado en la tabla *personal*. Presentará los siguientes campos:

- **Idreg**, campo configurado como clave primaria y cuyos datos introducidos serán un número entero e incremental. Un ejemplo sería:

Idreg
1
2
3
4

Figura 4-13 Ejemplo campo Idreg.

- **Acontecimiento,** de tipo VARCHAR de máximo 60 caracteres en el que se recoge, textualmente, el hecho que ha originado el cambio en la tabla *personal*.

  Los valores posibles que puede tomar este campo son:

  - Entrada: “Ha entrado en el recinto.”
  - Salida: “Ha salido del recinto.”
  - Alta: “Se ha añadido al sistema.”
  - Baja: “Se ha eliminado del sistema.”

  Un ejemplo sería:

Acontecimiento
Ha salido del recinto.
Ha entrado en el recinto.
Se ha añadido al sistema.
Se ha eliminado del sistema.

Figura 4-14 Ejemplo campo Acontecimiento.

- **Fechahora**, un campo de tipo TEXT, en el que irá insertada la fecha y hora a la que se efectúa el acontecimiento. El formato de este campo es de la siguiente manera: YY-MM-DD HH:MM. Un ejemplo sería:

Fechahora
19-02-08 16:45
19-02-09 17:23
19-02-09 18:23

Figura 4-15 Ejemplo campo Fechahora.

- **Ident**, un campo de tipo entero e incremental, implementado como clave foránea o extranjera, cuyos datos tienen su correspondencia en el campo **Ident** de la tabla *personal*. Un ejemplo este campo sería:

Ident
1
2
3
4

Figura 4-16 Ejemplo campo Ident.

Un ejemplo de registro sería el siguiente:

Idreg	Acontecimiento	Fechahora	Ident
1	Ha salido del recinto.	19-02-08 16:45	3

Figura 4-17 Ejemplo registro de tabla registro.

	idreg	acontecimiento	hora	ident
	Filtro	Filtro	Filtro	Filtro
1	23	se ha añadido al registro	2019-03-02 20:34:31	36
2	22	se ha eliminado del registro	2019-03-02 20:33:42	35
3	21	Ha entrado en el recinto	2019-03-02 20:33:31	35
4	20	Ha salido del recinto	2019-03-02 20:33:15	35

Figura 4-18 Tabla *registro*.

4.3.1.3 Consulta mediante la conexión a la red

La base de datos con la que se trabaja es del tipo orientada a fichero (*file-oriented database*). En este tipo de BBDD la información reside en un uno varios ficheros que residen en la máquina. Las transacciones con la BBDD se realizan mediante un motor que accede al fichero/s para devolver, insertar o modificar los datos requeridos. La empresa desarrolladora, SQLite Consortium, recomienda que no se compartan en red esta clase de ficheros, ya que, por ejemplo, aparecen problemas de bloqueo cuando se consulta la base de datos desde dos lugares distintos a la vez, y no deja modificar simultáneamente.

En definitiva, el archivo de SQLite con el que se está trabajando sí que puede compartirse a través de una red, pero no es un método particularmente eficiente, puede dar fallos anteriormente mencionados por lo que su uso en red está desaconsejado. Se escogió ese tipo de base de datos por su sencillez y porque en principio, por la arquitectura del sistema, no se esperan modificaciones del sistema desde varios puntos a la vez (sólo va a estar instalado en un punto de acceso).

Sin embargo, se han propuesto sistemas para paliar esta clase de problemas al compartir ficheros SQLite por una red, como por ejemplo el **SQLiteDbServer** [23], que es un servidor de base de datos basado en SQLite TCP/IP multiproceso, cuya comunicación entre clientes y servidor está cifrada y comprimida.

4.4 Arquitectura general del programa

El programa o código que utilizará el sistema de control de accesos estará compuesto por un programa principal desde el que se lanzan tres hilos de ejecución simultánea:

- **ThreadDNI**
- **ThreadTIM**
- **ThreadDNie**

Cada hilo se corresponde con una tecnología de identificación, que serán la validación mediante la introducción manual del DNI, la validación mediante la lectura de la tarjeta magnética y la validación mediante la introducción del DNI en un lector de DNIE.

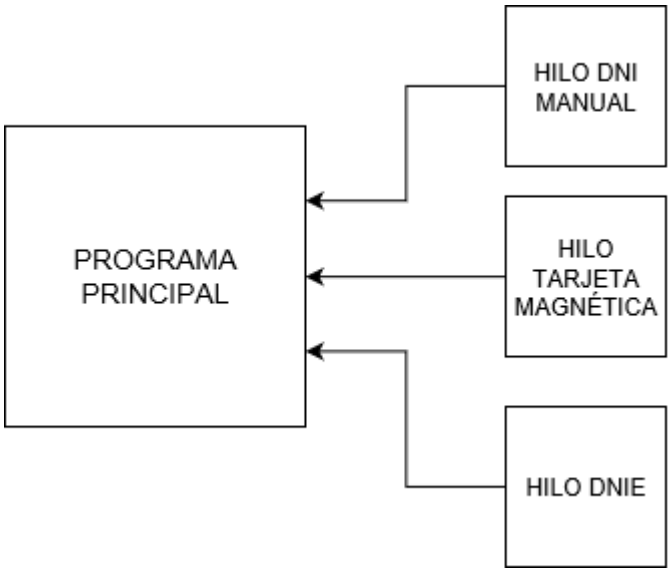

Figura 4-19 Arquitectura general del programa.

Cada hilo dispondrá de unas funciones propias. Mientras que el hilo correspondiente va a disponer de 6 distintas, los otros dos hilos dispondrán únicamente de la función ***transito***. Esta función, como quedará reflejado en los siguientes apartados solo cambiará entre hilos en la manera de obtener los datos de la credencial utilizada.

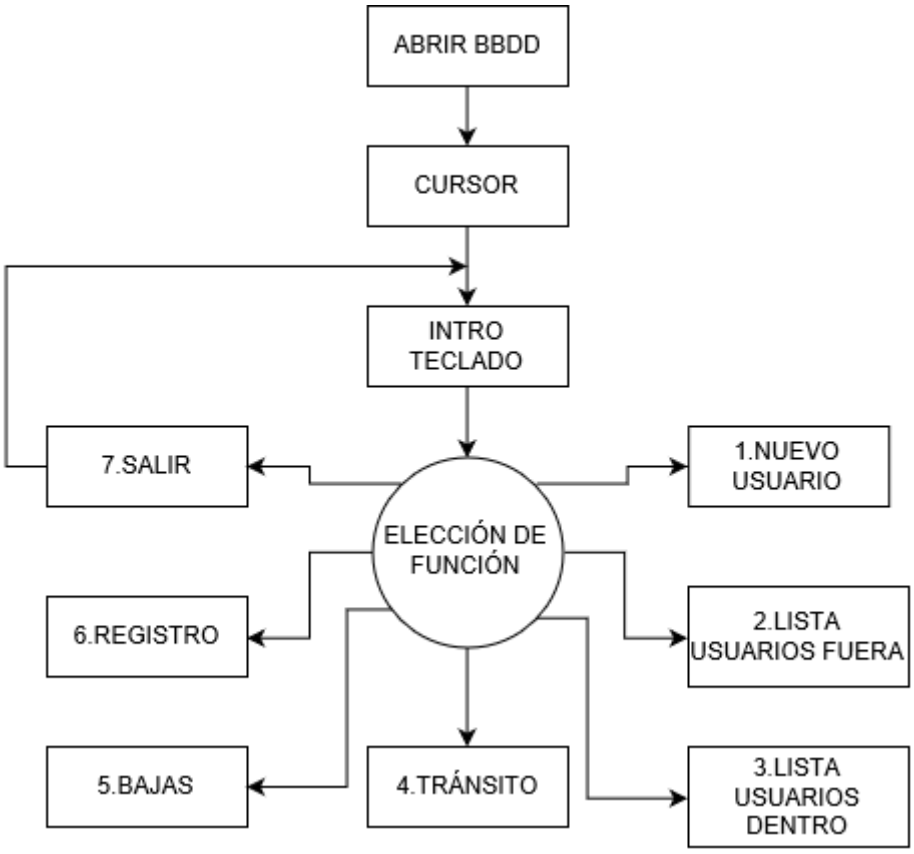

Figura 4-20 Funciones implementadas en el hilo ThreadDNI.

Los hilos que parten del programa principal comparten gran parte de la sección del código, y al ser independientes, permite que se puedan ejecutar de manera paralela, sin interferir unos con otros y de manera que si alguno tiene un fallo, no afectará ni a los otros thread, ni a la ejecución del programa principal.

Otro aspecto a comentar, es la utilización de una única base de datos por parte de los dos hilos. Esta se ha de abrir en la inicialización de cada uno de ellos. Estos hilos se pondrán en marcha a la vez cuando se inicialice el sistema y ser cerrarán al mismo tiempo. Dentro de cada hilo, se definen una serie de funciones y cuando el último hilo concluye su función, termina el proceso, libreándose todos los recursos empleados y cerrándose la base de datos. Estos se definen primero creando una subclase de thread en la que se reescribe el método *run()* y el constructor *_init_()*. A continuación, cuando todos están definidos, se ejecutan invocándolos todos a la vez.

Una de las ventajas de haber implementado el sistema mediante la inclusión de threads es que en el caso de que no funcione alguno de los lectores, se podrá continuar probando con otro, y además, en un futuro, si se quisiesen implantar otras tecnologías como el lector de huella dactilar o el lector de iris, sólo habría que adjuntar un hilo nuevo, sin la necesidad de modificar el resto del programa.

4.5 Acceso manual mediante DNI

La codificación de este método de validación se programará no en un hilo, sino dentro del programa principal. En él se van a definir una serie de funciones, que serán comunes entre los distintos threads. Éstos, pasan por una petición de usuario para escoger la función que se quiere ejecutar.

4.5.1 Nuevo usuario

La siguiente función tiene como objetivo introducir nuevos usuarios en la base de datos, para que puedan ser dados de alta en el del sistema de control. Para lo cual, se introducirán los datos requeridos

por el sistema y desde entonces ese usuario será utilizado en el resto de funciones del programa, quedando registrada esta acción por el programa.

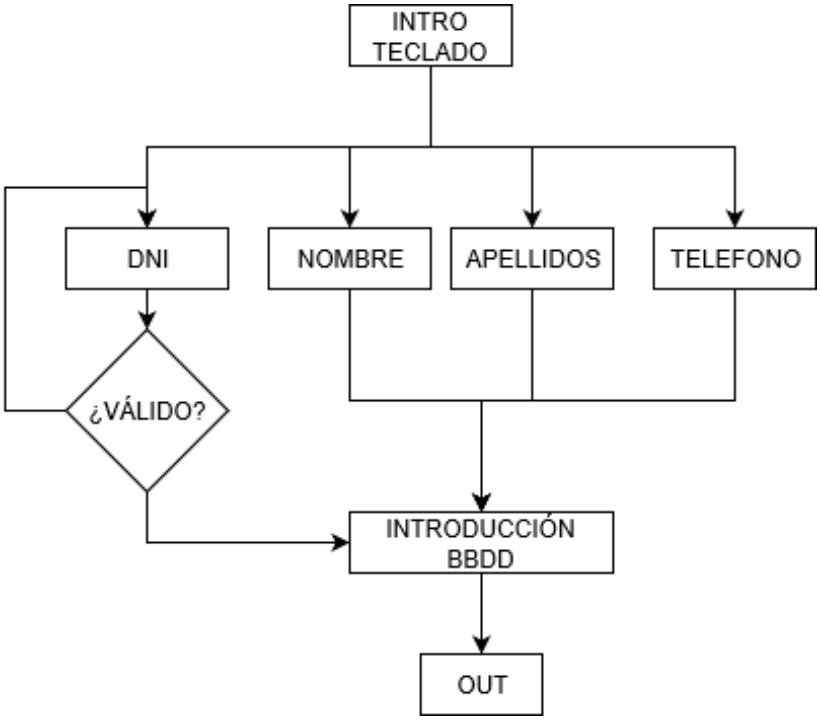

Figura 4-21 Función *Nuevo Usuario*.

Esta función se ha implementado con la misión de incluir un nuevo individuo en la BBDD. Para ello, primero se realiza una petición por la que se espera que se introduzcan el DNI, el nombre, los apellidos y el teléfono. El DNI se valida mediante una función llamada ***DNIvalido*** por la cual se determina que efectivamente ese DNI es válido, es decir, que corresponde a una secuencia de números y letra que cumplen la fórmula matemática recogida en el Articulo 11 del Real Decreto 1553/2005.

A continuación, se realizará una comprobación conforme ese DNI introducido no se encuentra ya registrado en la tabla *personal*. Para ello, se realiza una consulta a la BBDD ejecutando una secuencia SQL de tipo *SELECT* que localiza el campo de esta tabla en el que se encuentra el DNI introducido. Si ya está registrado, el programa informará y realizará la petición de nuevo. Si no, a esta petición le seguirán la del nombre, apellidos y teléfono.

Un ejemplo de la secuencia SQL mencionada sería:

"SELECT "DNI" FROM "personal" WHERE DNI="53587014G""

Tras la recopilación de los anteriores datos, le seguirá la introducción de los mismos en la tabla *personal* de la BBDD, mediante una secuencia SQL de tipo *INSERT*, en la que se indican los campos de interés, seguida de una orden de ejecución (*execute)* y otra de confirmación de cambios (*commit*).

Un ejemplo de la aplicación de la secuencia *INSERT* sería:

"INSERT INTO "personal" ("nombre, "apellidos", ...) Values ("Alejandro","Sánchez Cervera-Mercadillo", ...)"

Por último, se insertará información relativa a esta modificación de la tabla *personal* en la tabla *registro*, con ayuda nuevamente de una secuencia de tipo *INSERT*, que rellenará un campo llamado "acontecimiento" explicando que el usuario ha sido añadido, acompañado de la fecha y hora a la que se ha realizado. Será seguido de una orden de ejecución (*execute)* y otra de confirmación de cambios (*commit*). A continuación, la función finaliza y se devuelve el control al bucle principal del programa.

4.5.2 Tránsito

La siguiente función tiene como objetivo comprobar que efectivamente el individuo que está intentando acceder en el registro está dado de alta en la base de datos, y a continuación, una vez haya obtenido esta validación, que se actualice su situación en una tabla y que registre su entrada o salida en otra.

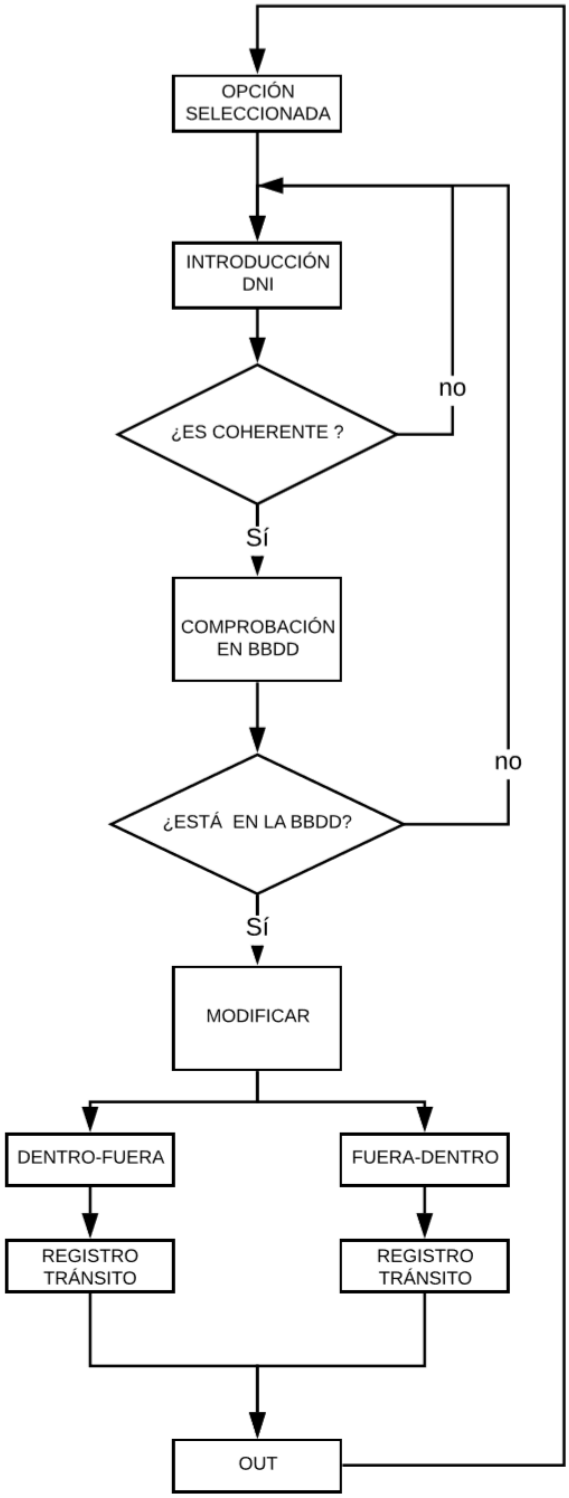

Figura 4-22 Función *Tránsito*.

Esta función se ha implementado con el objetivo de modificar y registrar el estado de un usuario en la BBDD. Para ello, primero se realiza una petición por la que el individuo introduce por teclado el DNI. Seguidamente, se comprueba si éste es válido mediante la función ***DNIvalido.***

Si el DNI que se ha introducido no es válido, se informará de que no es válido y la función finalizará.

Si por el contrario sí fuese válido, se realiza una consulta a la BBDD ejecutando una secuencia SQL que localiza el campo de la tabla de la BBDD en el que se encuentran los DNIs, con el objeto de localizar a cuál de todos los registros corresponde el DNI introducido.

Un ejemplo de la secuencia SQL mencionada sería:

"SELECT "DNI" FROM "personal" WHERE "DNI=53587014G""

Si no coincide con ninguno, se informará de que no existe y realizará la petición de nuevo. En el caso de que sí existiera, se llegaría a otra decisión:

Si el usuario figura en el sistema como que está dentro del recinto, es decir, si el valor del campo "presente" es un 1, está se modificará con un 0 y se informará de que el usuario está saliendo. De esta manera habríamos actualizado su estado cada vez que pasa por el programa. En el caso de que estuviese dentro, sería similar pero cambiando 0 por 1 e informando de que ha entrado. Esta modificación en la BBDD se realiza mediante una secuencia SQL de tipo *UPDATE*, en el que habrá que indicarle el campo que queremos modificar, seguida de una orden de ejecución (*execute)* y otra de confirmación de cambios (*commit*).

Un ejemplo de la aplicación de la secuencia citada sería:

"UPDATE "personal" SET presente = 1 WHERE "DNI=53587014G""

La función mostrará por pantalla el registro entero con objeto de que el operador compruebe que efectivamente se han realizado estos cambios. Esto se efectúa mediante otra consulta a la BBDD y la presentación del registro actualizado.

Por último, se insertará información relativa al tipo de tránsito que se ha realizado mediante su introducción en la tabla *registro*, con ayuda de una secuencia SQL de tipo *INSERT*, anteriormente explicada, que rellenará un campo llamado "acontecimiento" explicando si el usuario ha salido o ha entrado en el recinto, acompañado de la hora a la que se ha realizado. Deberá ser seguirse de una orden de ejecución (*execute)* y otra de confirmación de cambios (*commit*). Seguidamente, la función finaliza y se devuelve el control al bucle principal del threat.

4.5.3 Bajas

La función ***Bajas*** tiene como objetivo eliminar de manera rápida aquellos usuarios que ya no se necesiten en la BBDD, para que sean dados de baja en el sistema de control. Para lo cual, se ha de introducir el DNI del usuario que se desea borrar, y tras una confirmación, ese usuario ya no figurará en la base de datos, y se registrará este acontecimiento en la tabla de *registro.*

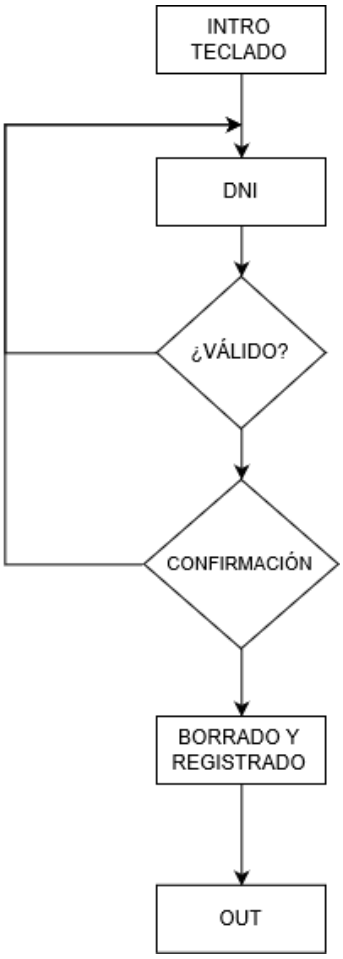

Figura 4-23 Función *Bajas*.

Primero se realiza una petición por la que se espera que se introduzcan el DNI del usuario que se desea eliminar. Este se valida mediante la función ***DNIvalido.*** Si los datos introducidos no son válidos, la petición se repetirá y si es correcto, se realiza una consulta a la BBDD ejecutando una secuencia *SELECT* explicada en el apartado 4.5.2, que localiza el campo de la tabla *personal* en el que se encuentra ese DNI, para saber qué registro se desea borrar. Si no coincide con ninguno, la función informará de que no existe.

En el caso de que existiera, se mostrará ese registro concreto por pantalla para que el usuario sepa qué registro está eliminando y a continuación, se realizará otra petición para solicitar confirmación de si se quiere borrar. Si la respuesta es negativa, la función finalizará. Si por el contrario la respuesta es positiva, le seguirá la eliminación del registro concreto que hemos seleccionado de la tabla *personal*, mediante una secuencia SQL de tipo *DELETE*, en la que se indica primero la tabla y luego el registro que se quiere eliminar, seguido de una orden de ejecución (*execute)* y otra de confirmación de cambios (*commit*).

Un ejemplo de la aplicación de la secuencia *DELETE* sería:

"DELETE from "personal" WHERE DNI="53587014G"

Por último, se insertará información relativa a esta modificación de la tabla personal en la tabla *registro*, con ayuda de una secuencia SQL de tipo *INSERT*, anteriormente explicada, que rellenará un campo llamado "acontecimiento" explicando que el usuario ha sido borrado, acompañado de la fecha y hora a la que se ha realizado. Será seguido de una orden de ejecución (*execute)* y otra de confirmación de cambios (*commit*). A continuación, la función finaliza y se devuelve el control al bucle principal del bloque.

4.6 Acceso con lector de tarjeta magnética

La codificación de este método de validación se programará en un hilo aparte, que se ejecutará al mismo momento que el hilo relacionado con el DNIE y en simultáneo con el programa principal. En él se va a definir una única función, que tendrá el mismo empleo que la función ***transito*** previamente mencionada, pero diferirán en la manera en la que se introducen los datos de la credencial correspondiente dentro del programa, ya que estos serán transmitidos por el lector de tarjeta magnética, en vez de por teclado.

4.6.1 Tránsito

La siguiente función tiene como objetivo comprobar que el individuo que está intentando acceder al recinto está dado de alta en la base de datos, y a continuación, una vez haya obtenido esta validación, que se actualice su situación en una tabla y que registre su entrada o salida en otra.

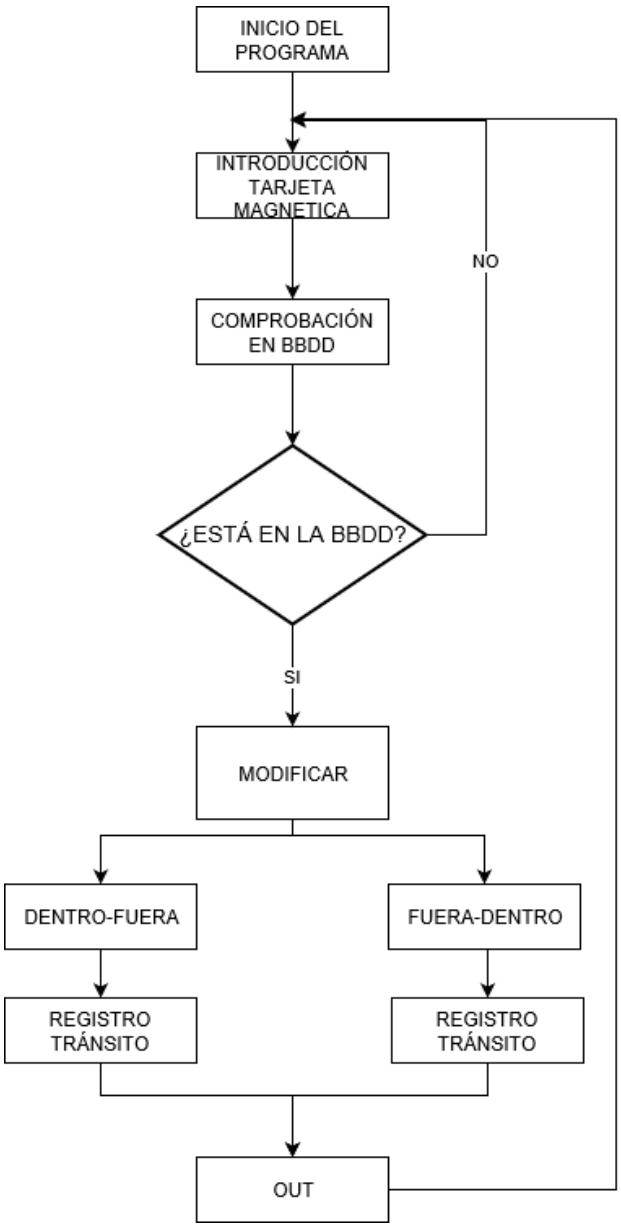

Figura 4-24 Función *Tránsito* con lector T.magnética.

Nada más inicializarse el programa, se realiza una petición por la que el usuario debe o escoger una opción de las propuestas o introducir la tarjeta magnética que registró en el lector correspondiente. El lector devuelve al programa una serie de caracteres, que serán los que habrá que comparar con la

BBDD para comprobar si está registrado. Por tanto, se realiza una consulta a la base de datos ejecutando una secuencia *SELECT* que localiza el campo de la tabla de la BBDD en el que se encuentran los datos de las tarjetas magnéticas de cada usuario, con el objeto de localizar qué registro contiene los datos introducidos.

Un ejemplo de la secuencia SQL mencionada sería:

"SELECT "Tarjmagnetia" FROM "personal" WHERE "Tarjmagnetica=100000000201020330"

Si no coincide con ninguno, la función informará de que no existe y realizará la petición de nuevo. En el caso de que si existiera, se llegaría a otra decisión:

Si el usuario figura en el sistema como que está dentro del recinto, es decir, si el valor del campo "presente" es un 1, está se modificará con un 0 y se informará de que el usuario está saliendo. De esta manera habríamos actualizado su estado cada vez que pasa por el programa. En el caso de que estuviese dentro, sería similar pero cambiando 0 por 1 e informando de que ha entrado. Esta modificación en la BBDD se realiza mediante una secuencia SQL de tipo *UPDATE*, en el que habrá que indicarle el campo que queremos modificar, seguida de una orden de ejecución (*execute)* y otra de confirmación de cambios (*commit*).

Un ejemplo de la aplicación de la secuencia citada sería:

"UPDATE "personal" SET presente = 1 WHERE "Tarjmagnetica=100000000201020330"

La función mostrará por pantalla el registro entero con objeto de que el operador compruebe que efectivamente se han realizado estos cambios. Esto se efectúa mediante otra consulta a la BBDD y la presentación del registro actualizado.

Por último, se insertará información relativa al tipo de tránsito que se ha realizado mediante su introducción en la tabla *registro*, con ayuda de una secuencia SQL de tipo *INSERT*, anteriormente explicada, que rellenará un campo llamado "acontecimiento" explicando si el usuario ha salido o ha entrado en el recinto, acompañado de la hora a la que se ha realizado. Deberá ser seguirse de una orden de ejecución (*execute)* y otra de confirmación de cambios (*commit*). Seguidamente, la función finaliza y se devuelve el control al bucle principal del thread.

4.7 Acceso por DNIE

Las funciones implementadas en la codificación de este método de validación se programarán en un hilo aparte, que se ejecutará al mismo tiempo que el hilo relacionado con la tarjeta magnética y el del DNI manual, todo ello en simultáneo con el programa principal. En él se va a definir una única función, que tendrá el mismo empleo que la función ***transito*** del hilo **ThreadDNI**, pero diferirá en la manera en la que se introducen los datos del DNI, que serán transmitidos por el lector de DNIE cuando insertemos el DNI, en vez de por teclado.

4.7.1.1.1 Tránsito

La siguiente función tiene como objetivo comprobar que el DNI introducido en el lector está dado de alta en la base de datos, y a continuación, una vez haya obtenido esta validación, que se actualice su situación en una tabla y que registre su entrada o salida en otra.

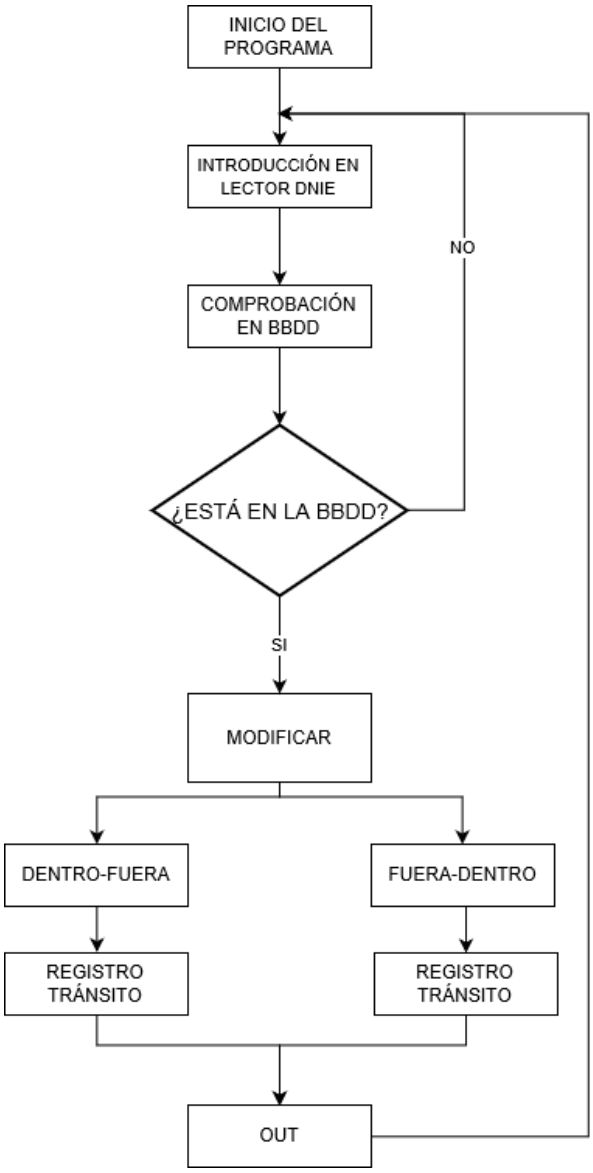

Figura 4-25 Función tránsito con DNIE.

Para ello, nada más iniciarse el programa se realiza una petición por la que el usuario introduce el DNI en el lector de DNIE. El lector devuelve una serie de caracteres, que pueden ser organizados para obtener los que interesen, en este caso el número del DNI con la letra. Estos serán los que habrá que comparar con la BBDD para comprobar si está registrado. Realizando con distintos DNIs, se llega a la conclusión de que dependiendo del punto en el que fueron tramitados, esos caracteres se encontrarán en distintas disposiciones, estando desfasados por lo general de dos a tres puestos. Por tanto, se decide insertar todos los caracteres que devuelva el lector en un mismo campo, y de esta manera, no se repetirán nunca entre usuarios.

Si no coincide con ninguno, la función informará de que no existe y realizará la petición de nuevo. En el caso de que si existiera, se llegaría a otra decisión:

Si el usuario figura en el sistema como que está dentro del recinto, es decir, si el valor del campo "presente" es un 1, está se modificará con un 0 y se informará de que el usuario está saliendo. De esta manera habríamos actualizado su estado cada vez que pasa por el programa. En el caso de que estuviese dentro, sería similar pero cambiando 0 por 1 e informando de que ha entrado. Esta modificación en la BBDD se realiza mediante una secuencia SQL de tipo *UPDATE*, en el que habrá que indicarle el campo que queremos modificar, seguida de una orden de ejecución (*execute)* y otra de confirmación de cambios (*commit*).

Un ejemplo de la aplicación de la secuencia citada sería:

"UPDATE "personal" SET presente=1 WHERE "DNIE=111111025358701G15234…"

La función mostrará por pantalla el registro entero con objeto de que el operador compruebe que efectivamente se han realizado estos cambios. Esto se efectúa mediante otra consulta a la BBDD y la presentación del registro actualizado.

Por último, se insertará información relativa al tipo de tránsito que se ha realizado mediante su introducción en la tabla *registro*, con ayuda de una secuencia SQL de tipo *INSERT*, anteriormente explicada, que rellenará un campo llamado "acontecimiento" explicando si el usuario ha salido o ha entrado en el recinto, acompañado de la hora a la que se ha realizado. Deberá ser seguirse de una orden de ejecución (*execute)* y otra de confirmación de cambios (*commit*). Seguidamente, la función finaliza y se devuelve el control al bucle principal del thread.

4.8 Interfaz de usuario

Al iniciar el programa principal se pondrán en marcha los 3 hilos a la vez, y el menú mostrado tendrá una serie de opciones correspondientes a cada función, que serán seleccionadas mediante las teclas numéricas del teclado.

```
Starting Main...
DNIe: Starting thread
DNI: Starting thread
DNI: BBDD abierta con éxito
FP: Starting thread
TIM: Starting thread

 *** CONTROL DE ACCESOS ***

1) NUEVO USUARIO
2) LISTADO DE PERSONAL AUSENTE
3) LISTADO DE PERSONAL PRESENTE
4) TRÁNSITOS
5) BAJAS
6) REGISTRO
7) SALIR

Elija una opción:
```

Ilustración 4-1 Menú de inicio del programa.

A continuación, se detallan las funciones que tienen como objetivo presentar cierta información requerida al usuario que quiera obtener datos del estado de la BBDD. Se encuentran implementadas en el hilo de **ThreadDNI** son los listados de presentes y ausentes en el recinto, así como el registro de acontecimientos.

4.8.1 Lista de usuarios que están fuera

La siguiente función se implementa con el objetivo de mostrar qué usuarios están fuera del recinto de la Escuela Naval.

Una vez se accede a esa función, el sistema realizará una consulta mediante la secuencia *SELECT*, en la que se especificará el campo requerido, en este caso "*presente"* en aquellos registros en los que tiene el valor 1 y se le especifica que ordene esa consulta por orden alfabético con una sentencia *ORDER BY*. Será seguido de una orden de ejecución (*execute)* y otra para extraer la información almacenada en el cursor (*fetchall*). En el caso de que no hubiese usuarios fuera del recinto, (no hubiese ningún registro en el que el campo presente fuese igual a 1, el sistema informaría de que todos los usuarios se encuentran dentro del recinto y a continuación la función finalizaría. Si por el contrario se da el caso de que sí existan, la lista de los registros introducidos cuyo campo *presente* sea igual a 1 se

mostrará por pantalla. A continuación, la función finaliza y se devuelve el control al bucle principal del thread.

```
Elija una opción: 2
2

LISTA DE USUARIOS AUSENTES

 20 04632558J   Enrique         Salamanca       623536342       0
 21 53589313A   José            Díez    654534378       0
 22 32719208Y   Marta           López   654435324       0
```

Ilustración 4-2 Listado de ausentes.

4.8.2 Lista de usuarios que están dentro

De una manera similar a la función anterior, la siguiente función se implementa con el objetivo de mostrar qué usuarios están fuera de la academia. Lo único que variará respecto a la anterior es que se mostrarán aquellos en los cuales el campo *presente* tenga el valor 0, en el caso de que no haya, informará de que todos están fuera.

```
LISTA DE USUARIOS PRESENTES

  3 32726547P   Victor          Sánchez Cervera-Mercadillo      620750338       1
  5 36084170E   javier          Sánchez Cervera-Mercadillo      620750338       1
  6 44568965F   Darío           Barbudo                 654343235       1
 38 53587015M   Luis            Sánchez Cervera-Mercadillo      686338437       1
```

Ilustración 4-3 Listado de presentes.

4.8.3 Registro

La función ***registro*** tiene una funcionalidad similar a las dos funciones ***lista***, pero su propósito es mostrar la lista entera de acontecimientos de la tabla *registro*, para poder observar los sucesos respectivos a las entradas, salidas, introducción y eliminación de los usuarios registrados en la tabla *personal* de la base de datos. Para ello, de manera análoga, realizará la misma consulta, pero se referirá a la tabla *registro*, y se hará de una manera general al no especificar ningún campo en concreto, de esta manera se mostrará la lista entera de acontecimientos por pantalla, ordenados por fecha y hora.

A continuación, la función finaliza y se devuelve el control al buque principal del thread.

```
(61, 'Ha entrado en el recinto', '2019-03-05 18:40:29', None)
(62, 'Ha salido del recinto', '2019-03-06 19:23:04', None)
(63, 'Ha entrado en el recinto', '2019-03-06 19:23:08', None)
(64, 'Ha salido del recinto', '2019-03-06 19:23:40', None)
(65, 'se ha añadido al registro', '2019-03-06 19:42:16', 42)
(66, 'se ha añadido al registro', '2019-03-06 20:00:33', 43)
(67, 'Ha salido del recinto', '2019-03-06 20:10:35', None)
(68, 'Ha entrado en el recinto', '2019-03-06 20:10:45', None)
(69, 'Ha salido del recinto', '2019-03-06 20:34:13', None)
(70, 'Ha salido del recinto', '2019-03-06 20:34:13', None)
(71, 'Ha entrado en el recinto', '2019-03-06 20:35:09', None)
(72, 'Ha salido del recinto', '2019-03-06 20:27:26', None)
(73, 'Ha entrado en el recinto', '2019-03-06 20:30:58', 41)
(74, 'se ha añadido al registro', '2019-03-06 20:34:12', 44)
(75, 'Ha salido del recinto', '2019-03-06 20:34:22', 44)
(76, 'Ha entrado en el recinto', '2019-03-06 20:36:29', 44)
```

Ilustración 4-4 Registro de acontecimientos.

5 PRUEBAS/ OBSERVACIONES

5.1 Pruebas de identidad

5.1.1 Validación con DNI

En primer lugar se realizan pruebas con el método de validación con introducción manual del DNI. Este método se implementa con el objetivo de tener una solución rápida de verificar que una persona se encuentra en el registro. Para ello, se pondrán a prueba las distintas funciones que dependen de este método: ***Nuevo usuario***, ***Tránsito*** y ***Bajas***.

5.1.1.1 Nuevo usuario

En cuanto a la creación de un nuevo usuario, se tendrá que introducir un DNI por teclado con una letra en mayúscula o minúscula.

Figura 5-1 Menú principal.

Si la introducción no corresponde con ningún DNI válido (función ***DNIvalido)***, el programa informará y nos realizará la petición de nuevo.

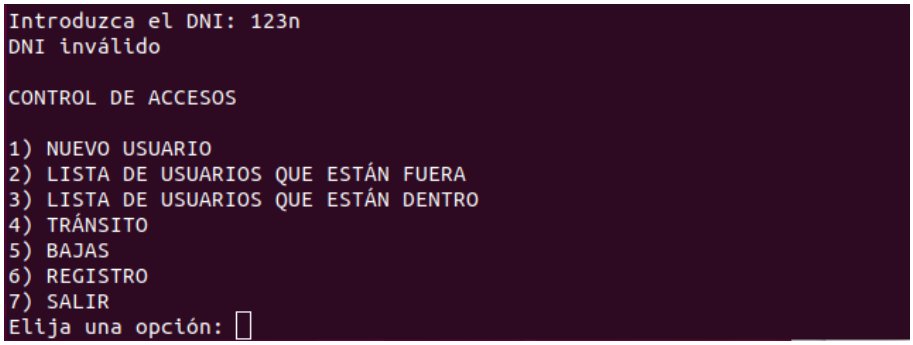

Figura 5-2 DNI inválido.

Si introducimos un DNI que sí que es válido, el programa comprobará si ese DNI ya consta en el registro de la tabla *personal*. En el caso de que ya exista, el programa informará y realizará de nuevo la petición. Si no existe, el programa pasará a la siguiente petición, que será la introducción del resto de datos.

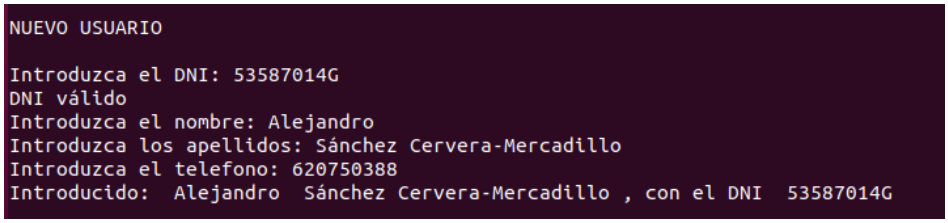

Figura 5-3 Creación de nuevo usuario.

Tras informar de que se ha creado este nuevo usuario, se introduce este acontecimiento en la tabla *registro*, acompañado de la fecha y hora.

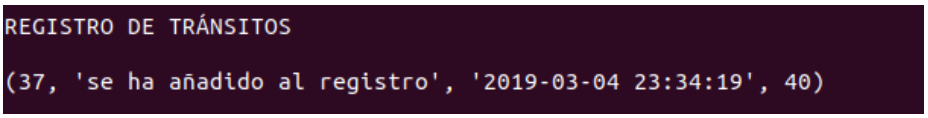

Figura 5-4 Inserción en tabla *registro*.

5.1.1.2 Tránsito

En cuanto a las salidas y entradas de los usuarios, el programa evaluará una serie de circunstancias.

Primero realizará una petición del DNI del usuario que está entrando o saliendo del recinto. Nuevamente evaluará si es o no válido este DNI.

Figura 5-5 DNI inválido en tránsito

Una vez se introduce un DNI válido se comprueba si existe o no en la tabla *personal*.

Figura 5-6 Usuario inexistente en tabla *personal*.

En función del campo *presente* de esta tabla, el usuario entrará o saldrá, modificándose este campo y mostrándolo por pantalla.

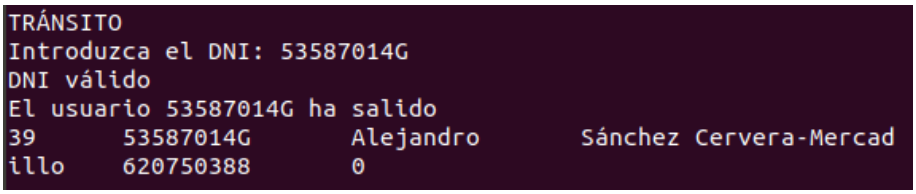

Figura 5-7 Comprobación de salida.

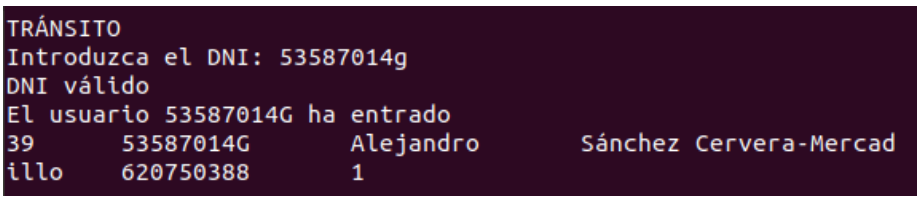

Figura 5-8 Comprobación de entrada.

Por último, este acontecimiento será recogido en la tabla *registro,* acompañado de la fecha y hora a la que ha tenido lugar.

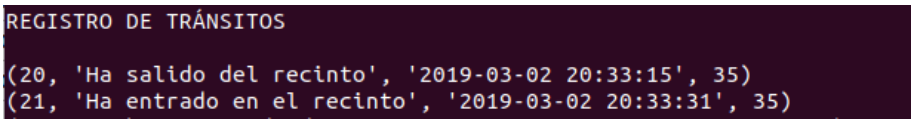

Figura 5-9 Inserciones de tránsitos en tabla *registro*.

5.1.1.3 Bajas

En cuanto a la eliminación de uno de los usuarios registrado en el sistema, se tendrá que introducir un DNI por teclado con una letra en mayúscula o minúscula.

Si la introducción no corresponde con ningún DNI válido (función ***DNIvalido)***, el programa informará y nos realizará la petición de nuevo.

Si introducimos un DNI que sí es válido, el programa comprobará si ese DNI ya consta en el registro de la tabla *personal.* Si ese usuario no existe, el programa informará y realizará la petición de nuevo. Si existe, el programa pasará a la siguiente petición, que será una confirmación de si efectivamente se desea eliminar ese registro.

Si la respuesta es negativa, la función finalizará y si es positiva, el registro será eliminado de la tabla *personal*, se informará de este hecho por pantalla y quedará registrado en la tabla *registro*, acompañado de la fecha y hora.

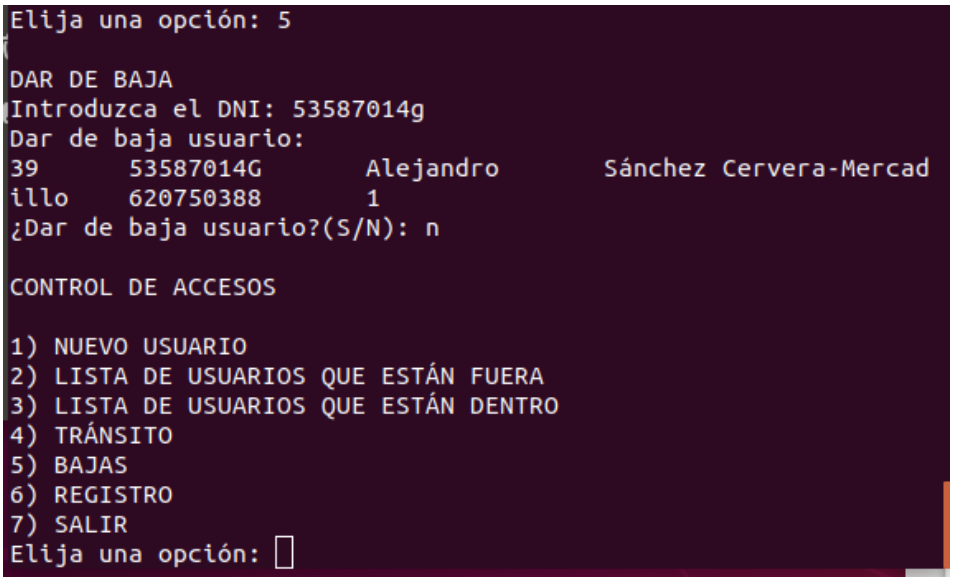

Figura 5-10 Negación de borrado.

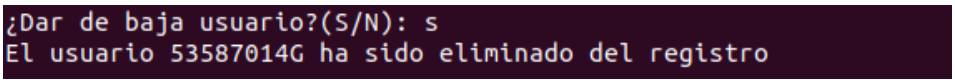

Figura 5-11 Confirmación de borrado.

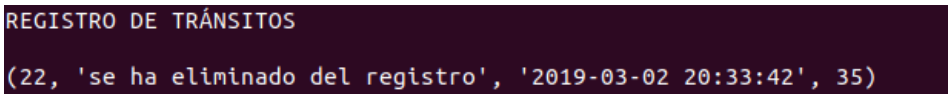

Figura 5-12 Inserción de eliminación en *registro*.

5.1.2 Validación con banda magnética

5.1.2.1 Transito

Esta función, cuyo empleo es el mismo que la función ***transito***, se pondrá en funcionamiento nada más inicialicemos el programa, y pretende ser un método rápido y efectivo para validar una identidad.

La función ***transito*** pretende ser un método rápido y efectivo para validar una identidad, se iniciará nada más inicializar el programa.

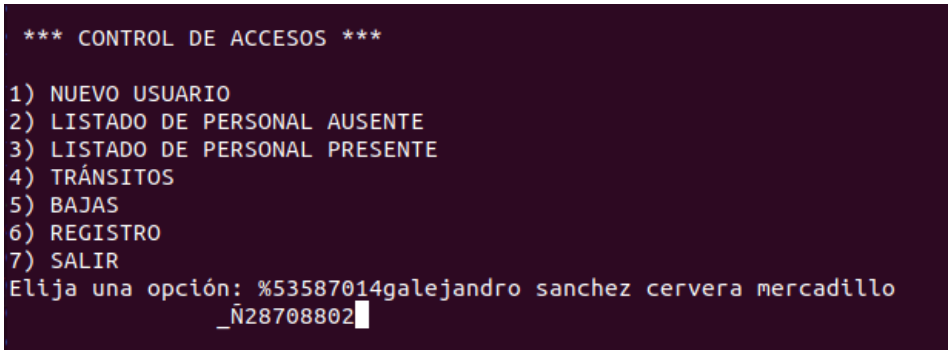

Figura 5-13 Tránsito con lector de tarjeta magnética.

5.1.3 Validación con DNIE

5.1.3.1 Update

Esta función, cuyo empleo es el mismo que la función ***transito***, se pondrá en funcionamiento nada más inicialicemos el programa, y pretende ser un método rápido y efectivo para validar una identidad.

No necesitará comprobar si el DNI es válido o no, porque la información llega directamente del lector de DNIE, pero sí que tendrá que establecer si ese usuario existe o no en la tabla *personal*.

```
TRÁNSITO
Introduzca el DNI: 50991502l
DNI válido
El usuario no existe
```

Figura 5-14 Usuario inexistente en tabla *personal*.

En función del campo *presente* de esta tabla, el usuario entrará o saldrá, modificándose este campo y mostrándolo por pantalla.

```
1) NUEVO USUARIO
2) LISTADO DE PERSONAL AUSENTE
3) LISTADO DE PERSONAL PRESENTE
4) TRÁNSITOS
5) BAJAS
6) REGISTRO
7) SALIR
Elija una opción:
El usuario Alejandro Sánchez Cervera-Mercadillo ha entrado
```

Figura 5-15 Tránsito de entrada con DNIE.

Por último, este acontecimiento será recogido en la tabla *registro*, acompañado de la fecha y hora a la que ha tenido lugar.

5.2 Pruebas a nivel usuario

5.2.1 Listados

En los siguientes apartados se muestran las distintas pruebas realizadas de los distintos listados que pueden mostrar a los individuos que se encuentran tanto dentro como fuera, y el registro total de acontecimientos.

5.2.1.1 Dentro del recinto

Pulsando la opción 2 se muestra el listado de usuarios que están dentro del recinto (los que tienen en su campo *presente* el valor 1 en la tabla *personal*.

```
Elija una opción: 3

LISTA DE USUARIOS PRESENTES

  6 44568965F   Dario          Barbudo                 654343235     1
  3 32726547P   Victor         Sánchez Cervera-Mercadillo   620750338        1
  5 36084170E   javier         Sánchez Cervera-Mercadillo   620750338        1
 38 53587015M   Luis           Sánchez Cervera-Mercadillo   686338437        1
```

Figura 5-16 Lista de presentes.

5.2.1.2 Fuera del recinto

Pulsando la opción 3 se muestra el listado de usuarios que están dentro del recinto (los que tienen en su campo *presente* el valor 0 en la tabla *personal*.

```
Elija una opción: 2

LISTA DE USUARIOS AUSENTES

 21 53589313A   José           Díez    654534378       0
 22 32719208Y   Marta          López   654435324       0
 20 04632558J   Enrique        Salamanca       623536342    0
```

Figura 5-17 Lista de ausentes.

5.2.1.3 Registro

Pulsando la opción 3 se muestra el listado de todos los acontecimientos registrados en la tabla *registro*.

```
REGISTRO DE TRÁNSITOS

(20, 'Ha salido del recinto', '2019-03-02 20:33:15', 35)
(21, 'Ha entrado en el recinto', '2019-03-02 20:33:31', 35)
(22, 'se ha eliminado del registro', '2019-03-02 20:33:42', 35)
(23, 'se ha añadido al registro', '2019-03-02 20:34:31', 36)
(24, 'Ha salido del recinto', '2019-03-02 20:40:29', 36)
(25, 'Ha entrado en el recinto', '2019-03-02 20:40:40', 36)
(26, 'se ha eliminado del registro', '2019-03-02 20:42:23', 36)
(27, 'se ha añadido al registro', '2019-03-02 20:45:47', 37)
(28, 'Ha salido del recinto', '2019-03-02 20:46:58', 37)
(29, 'se ha eliminado del registro', '2019-03-02 20:48:03', 37)
(30, 'se ha añadido al registro', '2019-03-03 15:47:10', 38)
(31, 'Ha salido del recinto', '2019-03-03 15:48:31', 38)
(32, 'Ha entrado en el recinto', '2019-03-03 15:48:51', 38)
(33, 'se ha añadido al registro', '2019-03-04 23:13:36', 39)
(34, 'Ha salido del recinto', '2019-03-04 23:20:35', 39)
(35, 'Ha entrado en el recinto', '2019-03-04 23:21:10', 39)
(36, 'se ha eliminado del registro', '2019-03-04 23:25:36', 39)
```

Figura 5-18 Registro.

6 CONCLUSIONES Y LÍNEAS FUTURAS

6.1 Conclusiones

Para finalizar el presente documento se realizará una serie de comentarios sobre el trabajo realizado, examinando las dificultades y problemas que han surgido a lo largo del mismo, para saldar con una serie de conclusiones.

Durante la realización del proyecto se comprueba, como se comentó en el apartado 2.4, que las Tarjetas TIM, incorporan una banda magnética que no está grabada, luego no pueden ofrecer ningún dato a no ser que éste se grabe con posterioridad. Por ello, a pesar de que una de las ideas iniciales era utilizar esta clase de tarjetas, se opta por otro medio quizá más efectivo, como puede ser la utilización del DNIE. Por otro lado, también se averigua que la academia no posee autorización para tratar datos biométricos, por ello, aun habiéndose adquirido el lector de huella dactilar, se decide no emplearlo, con vistas a su posible implantación en un futuro.

Una de las dificultades encontradas está relacionada con la base de datos seleccionada. Debido a que se trata de una BBDD sencilla y poco actualizada, se observó que existían problemas con el esquema multihilo (*multithreading*) planteado; no permitía abrir la BBDD al principio del programa y cerrarla al final y acceder a ella desde distintos hilos. Por ello, se decide abrir y cerrar la base de datos dentro de cada hilo. Otro de los problemas relacionados con esta base de datos son los problemas asociados con la modificación de esta desde distintos lugares remotos. Como se ha comentado en el apartado 4.3.1.3, como es una base de datos orientada al fichero, cuando se realizan cambios simultáneos en la BBDD puede dar fallos de bloqueo, por lo que no se recomienda utilizarlo con estos propósitos. Sin embargo, como en el caso propuesto en el trabajo solo se espera que se realicen cambios desde un punto concreto, la puerta del recinto, no ocurrirán esta clase de problemas, pudiéndose realizar consultas desde distintos lugares remotos.

Otra de las dificultades se ha presentado a la hora de extraer los datos del lector del DNIE. Según la bibliografía suministrada, los datos no se devuelven de la misma forma en todos los Documentos Nacionales de Identidad, ya que dependiendo de la comunidad en la que se ha expedido este, posee o no una serie de caracteres delante que hace sea imposible utilizar el mismo método para todos ellos. Tras examinar cerca de media centena de DNIs, se encontró que el mayor desfase posible eran dos caracteres hacia la derecha, luego la selección más aproximada sería seleccionar 7 caracteres centrados, de los 9 que posee el DNI. Aunque podría ser implementado como medida provisional, carece de utilidad en ocasiones como por ejemplo cuando se juntan dos familiares que obtuvieron este documento en fechas similares, un hecho común en la academia. Para remediar esto, se decidió almacenar los 255 primeros caracteres del fichero 6004 [24].

Como se ha expresado en anteriores apartados, el objetivo del proyecto era implementar mejoras en el sistema de control de accesos actual de la Escuela Naval Militar. El avance principal que se pretendía realizar era la puesta en marcha de una base de datos para el control y registro. Por lo tanto, desde el principio del proyecto se decide trabajar en un software para lograr este objetivo, con la ayuda de distintos lectores que en un principio se adjuntarían al software sin ser modificados.

Se ha de comentar que se han cumplido todos los objetivos, obteniéndose un sistema:

- Rápido y a tiempo real, que puede ser modificado con agilidad y consultado desde distintos lugares remotos.
- Fiable y redundante, dado que se han conseguido implantar distintos métodos para validar una identificación en el caso de que alguno falle.
- Universal, ya que se puede registrar y validar a todas personas, sin disponer de una credencial exclusiva de la propia academia, como es el DNI.

Por último, puesto que el objetivo era realizar un inicio a lo que pudiera ser en un futuro el sistema de control de accesos de la academia, tras las pruebas realizadas, se concluye que el sistema podría ser efectivamente implantado, trayendo beneficios como una mejora exponencial en el control y registro del personal que se encuentre dentro del recinto de la academia y una comprobación realista de los tránsitos realizados a lo largo del día, utilizando estos datos para el desempeño de la Guardia y pudiéndose consultar desde varios lugares.

6.2 Líneas futuras

6.2.1 Posibles implementaciones del software diseñado

Puesto que se pretendía que el siguiente trabajo fuera el inicio de una implementación moderna de un sistema de control de accesos en el marco del desarrollo tecnológico que está experimentando la Armada, existen ciertas mejoras que se podrían realizar en un futuro.

En cuanto a la organización del programa, se podrían implementar en un futuro distintas categorías en las tabla, en la que se separen por empleos o por profesiones, estableciendo así diferentes servicios en función del usuario. Un ejemplo sería que para ciertas personas no se les mostrase la hora de entrada y salida, o que el sistema impidiese accesos o salidas si el Comandante de la Guardia diese la orden. Del mismo modo, se podría implementar un buscador en los listados con el que se pudiese acceder rápidamente a un registro en concreto en el caso de estar buscando una irregularidad.

Otra de las implementaciones podría ser una función cuyo objetivo fuera efectuar cambios en los registros de los usuarios, para añadir o retirar información, o para añadir en un mismo registro otro método alternativo de identificación además del ya insertado.

Igualmente, otra de las posibles propuestas podría ser la realización de una interfaz gráfica para que no fuera necesario utilizar los números del teclado para desplazarse entre las distintas opciones del sistema.

6.2.2 Propuestas a nivel global para la mejora del sistema de control de acceso

Se debe mencionar que si se quisiese implementar un sistema de control de accesos en el futuro, la base de datos recomendada debería ser más potente, y actual, para que pudiese trabajar mejor en red, para que no hubiese problemas de bloqueo al modificarse desde distintos lugares.

Otra de las futuras implementaciones podría ser la grabación de la banda magnética que incorporan de serie las TIM. De esta manera, aunque no fuese el procedimiento más seguro, todos los militares podrían utilizar ese método de identificación y se aprovecharía un recurso que ya está en posesión de todos ellos.

Otra de las líneas futuras podría ser una investigación más exhaustiva acerca de cómo se organizan los datos en el chip del DNI, dado que según la documentación de la que se dispone, parece ser que en función de la localidad, se obtiene un distinto orden en la devolución de los datos, lo que hace imposible tratar todos los documentos de la misma forma y organizarlos en campos similares.

Además, con el fin de mejorar este sistema, se podría solicitar la autorización mencionada en el apartado 2.5.5, del tratamiento de datos biométricos. De esta manera, además de existir un método más de identificación en el caso de que otros fallaran, supondría un aumento de seguridad, al ser difícilmente reemplazable y no sería necesario llevar nada encima. En línea con este pensamiento, se debe comentar que una propuesta más interesante sería la introducción de la huella dactilar en el móvil como credencial. Al igual que está crecidamente implementado el uso de la huella dactilar para pagar con el teléfono móvil, lo mismo se podría hacer para la entrada en la academia. De esta manera, no se necesita ninguna autorización, ya que, aunque la huella se guarde en nuestro teléfono, la salida al lector es un simple número. De esta manera se realizaría de un modo más rápido y se excluiría el uso de tarjetas.

7 BIBLIOGRAFÍA

[1] Arantxa Mora Pérez, «Gestión de la prevención. Control de accesos,» Cartagena, 2016.

[2] Francisco Javier Navarro Amador, «El mundo de los controles de acceso.,» Universidad Oberta de Catalunya, 2014.

[3] Emilio Castro, «Proyecto de seguridad de control de accesos del edificio Villa de Madrid,» REF:170406, Madrid, 2017.

[4] Consentino, Luis, «Control de Accesos: Conceptos, historia y esquema básico».

[5] C. T. Borja y Á. G. Bueno, «Sistemas biométricos».

[6] *Tecnologías biométricas aplicadas a la ciberseguridad,* Instituto Nacional de Ciberseguridad, 2016.

[7] «Normativas ISO para las tarjetas con banda magnética,» FQ, 2013.

[8] Javier Gonzalez-Argote, Alexis Alejandro Garcia-Rivero, *Códigos QR y sus aplicaciones en las ciencias de la salud,* La Habana, 2016.

[9] Álvaro González Duarte, «Información para la planificación de la aplicación móvil para pagos con códigos QR.,» Universidad Militar Nueva Granada, Bogotá, 2013.

[10] José Rubén Ibáñez Sánchez, *Sistema lector de Tarjetas-Chip con acceso USB,* Madrid: Universidad Autónoma de Madrid, Enero 2013.

[11] Anna Melchor Pérez, *Las tarjetas inteligentes como herramienta innovadora en las ciudades,* Valencia: Universidad Politécnica de Valencia, Septiembre 2012.

[12] María Estefanía Casero, *Tecnología de identificación por radiofrecuencia. Lectura de pedidos RFID en un almacén.,* Universidad de la Rioja, 2013.

[13] J. M. G. Barceló, *Análisis y prueba de un sistema en tecnología de identificación por radiofrecuencia,* Cartagena: Universidad politécnica de Cartagena, 2016.

[14] Z. F. Llansola, «Sensores de identificación por Radio-Frecuencia,» Universidad Jaume I, Castellon, Octubre 2006.

[15] Miriam Cerón Brito, «Hardware y Software,» Universidad Autónoma del Estado de

Hidalgo, 2014.

[16] Juan Pavón Mestras, *Aplicaciones Web/Sistemas web,* Universidad complutense de madrid.

[17] «BOE-A-2002-5468,» 20 de marzo de 2002.

[18] G. A. a. D. S.A., «Manual de usuario: Sistema de Control de accesos de personal y material EXT.1ª Serie BAM 5-6,» Isaac Newton, 2018, pp. 2-7.

[19] «AEPD: Guía para el ciudadano,» Agencia Española de Protección de Datos.

[20] *Ley Orgánica 3/2018, de Protección de Datos Personales y garantía de los derechos digitales.,* 5 de diciembre de 2018.

[21] *REGLAMENTO (UE) 2016/679,* 27 de abril de 2016.

[22] *Orden DEF/1342/2015 Protección de datos de carácter personal. Ficheros de datos de carácter personal de la DIENA,* 20 de marzo de 2015.

[23] «Using SQLite on a Network,» SQLite, [En línea]. Available: https://www.sqlite.org/cvstrac. [Último acceso: 7 marzo 2019].

[24] «ATR de la tarjeta DNIe,» Cuerpo Nacional de Policía, Ministerio del Interior..

ANEXO I: CÓDIGO PROGRAMA PRINCIPAL

```
import threading
import DNImanual
import DNIe
import TIM
import evdev

#########################################################3

print ("Starting Main...")

# CREACIÓN DE LOS HILOS
HiloHuella  = ThreadHuella(1)
HiloDNIe    = DNIe.ThreadDNIe(2)
HiloTIM     = TIM.ThreadTIM(3)
HiloTeclado = DNImanual.ThreadTeclado(4)

# INICIALIZACIÓN DE LOS HILOS
HiloHuella.start()
HiloDNIe.start()
HiloTeclado.start()
HiloTIM.start()


try:
    while True:
        pass
except KeyboardInterrupt:
    print ("\nExiting main...")
    pass
```

ANEXO I I: CÓDIGO DE HILO DNI INTRODUCCIÓN MANUAL

```
import threading
import sqlite3
from time import gmtime, strftime
import os


def DNIvalido(dni):
    tabla = "TRWAGMYFPDXBNJZSQVHLCKE"
    dig_ext = "XYZ"
    reemp_dig_ext = {'X':'0', 'Y':'1', 'Z':'2'}
    numeros = "1234567890"
    dni = dni.upper()
    if len(dni) == 9:
        dig_control = dni[8]
        dni = dni[:8]
        if dni[0] in dig_ext:
            dni = dni.replace(dni[0], reemp_dig_ext[dni[0]])
        return len(dni) == len([n for n in dni if n in numeros]) \
            and tabla[int(dni)%23] == dig_control


class ThreadTeclado (threading.Thread):
   def __init__(self, threadID):
      threading.Thread.__init__(self)
      self.threadID = threadID
      print('DNI: Starting thread')
      self.con_bd = sqlite3.connect("accesos_ENM2.db", check_same_thread=False)
      self.cursor = self.con_bd.cursor()
      print('DNI: BBDD abierta con éxito')

   def DesconectarBBDD(self):
    try:
        self.cursor.close()
        self.con_bd.close()
        print('DNI: BBDD cerrada con éxito')
    except sqlite3.Error:
        print('DNI: Error al cerrar la BBDD')


   ###PARA LISTAR LOS USUARIOS QUE ESTÁN FUERA (LOS QUE ESTAN PRESENTE=0)
   def lista_ausentes(self):
    print("\nLISTA DE USUARIOS AUSENTES\n")
    sql = "SELECT * FROM personal WHERE presente=0"
    self.cursor.execute(sql)
    personas = self.cursor.fetchall()
    if personas:
      for persona in personas:
          print ("{:3} {:9}\t{:8}\t{:.20}\t{:9}\t{}".format(*persona))
    else:
          print("Todos los usuarios presentes")

   #### PARA LISTAR LOS USUARIOS QUE ESTÁN DENTRO (PRESENTE=1)
   def lista_presentes(self):
    print("\nLISTA DE USUARIOS PRESENTES\n")
    sql = "SELECT * FROM personal WHERE presente=1"
    self.cursor.execute(sql)
    personas = self.cursor.fetchall()
```

```
    if personas:
        for persona in personas:
            print ("{:3} {:9}\t{:8}\t{:20}\t{:9}\t{}".format(*persona))
    else:
        print("Todos los usuarios ausentes")
    return


   ### EXAMINA EL DNI QUE SE INTRODUZCA, MUESTRA EL REGISTRO, CAMBIA SU ESTADO E
INFORMA DEL TIPO DE TRÁNSITO REALIZADO
   def transitos(self):
    print( "\nTRÁNSITO")
    DNI = input("Introduzca el DNI: ").upper()
    #validar DNI
    if DNIvalido(DNI):
        print("DNI válido")
    else:
        print("DNI inválido")
    sql = "SELECT presente FROM personal WHERE DNI='"+DNI+"'"
    self.cursor.execute(sql)
    personas = self.cursor.fetchall()
    if personas:
        # El usuario existe
        if personas[0][0]==1:
            sql = "UPDATE personal SET presente=0 WHERE DNI='"+DNI+"'"
            self.cursor.execute(sql)
            self.con_bd.commit()
            fechahora=strftime("%Y-%m-%d %H:%M:%S")
            sql2 = "INSERT INTO registro VALUES (null,'Ha salido del
recinto','"+fechahora+"',(SELECT ident FROM personal WHERE DNI='"+DNI+"'))"
            self.cursor.execute(sql2)
            self.con_bd.commit()
            print("El usuario", DNI, "ha salido")
        else:
            sql = "UPDATE personal SET presente=1 WHERE DNI='"+DNI+"'"
            self.cursor.execute(sql)
            self.con_bd.commit()
            fechahora=strftime("%Y-%m-%d %H:%M:%S")
            sql2 = "INSERT INTO registro VALUES (null,'Ha entrado en el
recinto','"+fechahora+"',(SELECT ident FROM personal WHERE DNI='"+DNI+"'))"
            self.cursor.execute(sql2)
            self.con_bd.commit()
            print("El usuario", DNI, "ha entrado")
        # Comprobación
        sql = "SELECT * FROM personal WHERE DNI='"+DNI+"'"
        self.cursor.execute(sql)
        personas = self.cursor.fetchall()
        print ("{}\t{}\t{}\t{}\t{}\t{}".format(*personas[0]))
    else:
        print('El usuario no existe')
    return


   ##### AQUÍ MOSTRARÁ LOS TRÁNSITOS QUE HAN ACONTECIDO EN LA TABLA "registro"
   def registro(self):
    print("\nREGISTRO DE TRÁNSITOS\n")
    sql = "SELECT * FROM registro"
    self.cursor.execute(sql)
    registro = self.cursor.fetchall()
    if registro:
        for registr in registro:
            print (registr)
    else:
```

```
        print("NO HA HABIDO NINGÚN TRÁNSITO")
    return

   ###PARA CREAR UN NUEVO USUARIO
   def altas(self):
    print("\nNUEVO USUARIO\n")
    DNI = input("Introduzca el DNI: ").upper()
    sql = "SELECT presente FROM personal WHERE DNI='"+DNI+"'"
    self.cursor.execute(sql)
    personas = self.cursor.fetchall()
    if personas:
        print("El usuario ya existe.")
        return
    else:
    #validar DNI
      if DNIvalido(DNI):
        print("DNI válido")
      else:
        print("DNI inválido")
        return
      nombre = input("Introduzca el nombre: ")
      apellidos = input("Introduzca los apellidos: ")
      telefono = input("Introduzca el telefono: ")

      print( "Introducido: ", nombre, apellidos, ", con el DNI:", DNI)
      # Añadir registro a la tabla personal
      reg = (DNI, nombre, apellidos, telefono,"1")
      #Inserción del registro en tabla personal
      self.cursor.execute("INSERT INTO personal VALUES(null,?,?,?,?,?,null,null)",
reg)
      self.con_bd.commit()
      #Inserción de acontecimiento en tabla registro
      fechahora=strftime("%Y-%m-%d %H:%M:%S")
      sql2 = "INSERT INTO registro VALUES (null,'se ha añadido al
registro','"+fechahora+"',(SELECT ident FROM personal WHERE DNI='"+DNI+"'))"
      self.cursor.execute(sql2)
      self.con_bd.commit()
    return



   def bajas(self):
    print( "\nDAR DE BAJA")
    DNI = input("Introduzca el DNI: ").upper()
    # Comprueba si existe en la base de datos
    sql = "SELECT presente FROM personal WHERE DNI='"+DNI+"'"
    self.cursor.execute(sql)
    personas = self.cursor.fetchall()
    if personas:
    # No es necesario validar, ya que si está introducido, es que ha sido validado
de antemano.
        # El usuario existe
        print("Dar de baja usuario:")
        sql = "SELECT * FROM personal WHERE DNI='"+DNI+"'"
        self.cursor.execute(sql)
        personas = self.cursor.fetchall()
        print ("{}\t{}\t{}\t{}\t{}\t{}".format(*personas[0]))
        baja = input("¿Dar de baja usuario?(S/N): ")
        if baja.upper() == "S":
            sql1 = "SELECT * FROM personal WHERE DNI='"+DNI+"'"
            self.cursor.execute(sql1)
            datosborrados = self.cursor.fetchall()
            datosborrados = tuple(datosborrados[0])[1:6]
```

```
            fechahora=strftime("%Y-%m-%d %H:%M:%S")
            sql2 = "INSERT INTO registro VALUES (null,'se ha eliminado del
registro','"+fechahora+"',(SELECT ident FROM personal WHERE DNI='"+DNI+"'))"
            self.cursor.execute(sql2)
            self.con_bd.commit()
            # Se elimina de personal
            sql = "DELETE FROM PERSONAL WHERE DNI='"+DNI+"'"
            self.cursor.execute(sql)

            self.cursor.fetchall()
            sql2 = "INSERT INTO registros
(DNI,nombre,apellidos,telefono,presente,acontecimiento) FROM personal WHERE
DNI='"+DNI+"'"
            self.cursor.execute(sql2)
            self.con_bd.commit()

            sql = "DELETE FROM PERSONAL WHERE DNI='"+DNI+"'"
            self.cursor.execute(sql)
            self.con_bd.commit()
            print("El usuario", DNI, "ha sido eliminado del registro")
        else:
            return

    else:
        print('El usuario no existe')
        return
    return



   def run(self):
    while True:
        ###MENÚ DE ELECCIÓN
        os.system("clear")
        print("\n *** CONTROL DE ACCESOS ***\n")
        print("1) NUEVO USUARIO")
        print("2) LISTADO DE PERSONAL AUSENTE")
        print("3) LISTADO DE PERSONAL PRESENTE")
        print("4) TRÁNSITOS")
        print("5) BAJAS")
        print("6) REGISTRO")
        print("7) SALIR")
        opcion = input("Elija una opción: ")
        if   opcion == "1":
            print("1")
            self.altas()
        elif opcion == "2":
            print("2")
            self.lista_ausentes()
        elif opcion == "3":
            self.lista_presentes()
        elif opcion == "4":
            self.transitos()
        elif opcion == "5":
            self.bajas()
        elif opcion == "6":
            self.registro()
        elif opcion == "7":
           self.DesconectarBBDD()
           break
        else:
            print("\nOpción no válida")
```

ANEXO I I I: CÓDIGO DE HILO DNIE

```
import threading
import sqlite3
import os
from time import gmtime, strftime

from smartcard.CardType import AnyCardType
from smartcard.CardRequest import CardRequest
from smartcard.CardConnectionObserver import ConsoleCardConnectionObserver
from smartcard.CardMonitoring import CardMonitor, CardObserver
from smartcard.Exceptions import CardRequestTimeoutException
from smartcard.util import toHexString, toBytes
import re



class selectMyObserver(CardObserver):
    def __init__(self):
        self.observer = ConsoleCardConnectionObserver()

    def update(self, observable, actions):
        (addedcards, removedcards) = actions
        for card in addedcards:
            #print("+Inserted: ", toHexString(card.atr))
            card.connection = card.createConnection()
            card.connection.connect()
            card.connection.addObserver(self.observer)

            # apdu bytes
            CLA=0x90
            INS=0xB8
            P1=0x00
            P2=0x00
            LE=0x07
            # apdu chip info
            CHIP_INFO = [CLA, INS, P1, P2, LE]

            apdu = CHIP_INFO

            response, sw1, sw2 = card.connection.transmit(apdu)
            # HAY UN DNIE
            if sw1 == 0x90:
                # Vamos a por el PKCS15-CDF que se encuentra en 5015:6004
                # apdu bytes
                CLA=0x00
                INS=0xA4
                P1=0x00
                P2=0x00
                LC=0x02
                DATA = [0x50, 0x15] #Raiz

                # apdu Selección del fichero raiz
                SELECT_FILE_5015 = [CLA, INS, P1, P2] + [LC] + DATA
                apdu = SELECT_FILE_5015
                response, sw1, sw2 = card.connection.transmit(apdu)

                # apdu bytes
                CLA=0x00
```

```
                INS=0xA4
                P1=0x00
                P2=0x00
                LC=0x02
                DATA = [0x60, 0x04] #CDF (6004)

                # apdu Selección del fichero CDF (6004)
                SELECT_FILE_6040 = [CLA, INS, P1, P2] + [LC] + DATA
                apdu = SELECT_FILE_6040
                response, sw1, sw2 = card.connection.transmit(apdu)

                # apdu bytes
                CLA=0x00
                INS=0xC0
                P1=0x00
                P2=0x00
                LE=0x0E

                # apdu Get Response
                GET_RESPONSE = [CLA, INS, P1, P2, LE]
                apdu = GET_RESPONSE
                response, sw1, sw2 = card.connection.transmit(apdu)
                # FCI del fichero 6004. Longitud del fichero leído 0x0834 bytes

                # apdu bytes
                CLA=0x00
                INS=0xB0
                P1=0x00
                P2=0x00
                LE=0xff

                # apdu Read Binary
                READ_BINARY = [CLA, INS, P1, P2, LE]
                apdu = READ_BINARY
                response, sw1, sw2 = card.connection.transmit(apdu)
                DNIE = str(''.join(chr(i) for i in response if chr(i)<'~' and
chr(i)>' '))
                con_bd = sqlite3.connect("accesos_ENM2.db",
check_same_thread=False)
                cursor = con_bd.cursor()
                sql = "UPDATE personal SET DNIE=" + "'" + DNIE + "'" + " WHERE
DNI='53587014G'"
                cursor.execute(sql)
                con_bd.commit()

                sql1 = "SELECT presente FROM personal WHERE DNIE=" + "'" + DNIE +
"'"
                cursor.execute(sql1)
                personas = cursor.fetchall()

                if personas:
                    # El usuario existe
                    if personas[0][0]==1:
                        sql = "UPDATE personal SET presente=0 WHERE DNIE=" + "'" +
DNIE + "'"
                        cursor.execute(sql)
                        con_bd.commit()
                        fechahora=strftime("%Y-%m-%d %H:%M:%S")
                        sql2 = "INSERT INTO registro VALUES (null,'Ha salido del
recinto','"+fechahora+"',(SELECT ident FROM personal WHERE DNIE=" + "'" + DNIE +
"'))"
                        cursor.execute(sql2)
                        con_bd.commit()
```

```
                    else:
                        sql = "UPDATE personal SET presente=1 WHERE DNIE=" + "'" +
DNIE + "'"
                        cursor.execute(sql)
                        con_bd.commit()
                        fechahora=strftime("%Y-%m-%d %H:%M:%S")
                        sql2 = "INSERT INTO registro VALUES (null,'Ha entrado en
el recinto','"+fechahora+"',(SELECT ident FROM personal WHERE DNIE=" + "'" + DNIE
+ "'))"
                        cursor.execute(sql2)
                        con_bd.commit()
                    # Commpruebo que todo OK
                    sql = "SELECT * FROM personal WHERE DNIE=" + "'" + DNIE + "'"
                    cursor.execute(sql)
                    personas = cursor.fetchall()
                    if personas[0][5]==1:
                        print ("El usuario " + personas[0][2] + " ha entrado")
                    else:
                        print ("El usuario " + personas[0][2] + " ha salido")
                else:
                    print('El usuario no existe')
                #return
            else:
                print ('no DNIe')

class ThreadDNIe (threading.Thread):
   def __init__(self, threadID):
      threading.Thread.__init__(self)
      self.threadID = threadID
      print('DNIe: Starting thread')
      print('DNIe: BBDD abierta con éxito')


   def DesconectarBBDD(self):
    try:
        self.cursor.close()
        self.con_bd.close()
        print('DNIe: BBDD cerrada con éxito')
    except sqlite3.Error:
        print('DNIe: Error al cerrar la BBDD')


   def run(self):
       cardmonitor = CardMonitor()
       selectobserver = selectMyObserver()
       cardmonitor.addObserver(selectobserver)
```

ANEXO IV: CÓDIGO DE HILO TARJETA MAGNÉTICA

```
import threading
import sqlite3
import os

import sys
import evdev

vendor = 0xffff
product = 0x0035


CODE_MAP_CHAR = {
    'KEY_MINUS': "-",
    'KEY_SPACE': " ",
    'KEY_U': "U",
    'KEY_W': "W",
    'KEY_BACKSLASH': "\\",
    'KEY_GRAVE': "`",
    'KEY_NUMERIC_STAR': "*",
    'KEY_NUMERIC_3': "3",
    'KEY_NUMERIC_2': "2",
    'KEY_NUMERIC_5': "5",
    'KEY_NUMERIC_4': "4",
    'KEY_NUMERIC_7': "7",
    'KEY_NUMERIC_6': "6",
    'KEY_NUMERIC_9': "9",
    'KEY_NUMERIC_8': "8",
    'KEY_NUMERIC_1': "1",
    'KEY_NUMERIC_0': "0",
    'KEY_E': "E",
    'KEY_D': "D",
    'KEY_G': "G",
    'KEY_F': "F",
    'KEY_A': "A",
    'KEY_C': "C",
    'KEY_B': "B",
    'KEY_M': "M",
    'KEY_L': "L",
    'KEY_O': "O",
    'KEY_N': "N",
    'KEY_I': "I",
    'KEY_H': "H",
    'KEY_K': "K",
    'KEY_J': "J",
    'KEY_Q': "Q",
    'KEY_P': "P",
    'KEY_S': "S",
    'KEY_X': "X",
    'KEY_Z': "Z",
    'KEY_KP4': "4",
    'KEY_KP5': "5",
    'KEY_KP6': "6",
    'KEY_KP7': "7",
    'KEY_KP0': "0",
    'KEY_KP1': "1",
    'KEY_KP2': "2",
    'KEY_KP3': "3",
    'KEY_KP8': "8",
```

```
    'KEY_KP9': "9",
    'KEY_5': "5",
    'KEY_4': "4",
    'KEY_7': "7",
    'KEY_6': "6",
    'KEY_1': "1",
    'KEY_0': "0",
    'KEY_3': "3",
    'KEY_2': "2",
    'KEY_9': "9",
    'KEY_8': "8",
    'KEY_LEFTBRACE': "[",
    'KEY_RIGHTBRACE': "]",
    'KEY_COMMA': ",",
    'KEY_EQUAL': "=",
    'KEY_SEMICOLON': ";",
    'KEY_APOSTROPHE': "'",
    'KEY_T': "T",
    'KEY_V': "V",
    'KEY_R': "R",
    'KEY_Y': "Y",
    'KEY_TAB': "\t",
    'KEY_DOT': ".",
    'KEY_SLASH': "/",
}

def parse_key_to_char(val):
    return CODE_MAP_CHAR[val] if val in CODE_MAP_CHAR else ""

class ThreadTIM (threading.Thread):
   def __init__(self, threadID):
      threading.Thread.__init__(self)
      self.threadID = threadID
      print('TIM: Starting thread')
      self.con_bd = sqlite3.connect("accesos_ENM2.db", check_same_thread=False)
      self.cursor = self.con_bd.cursor()
      print('TIM: BBDD abierta con éxito')

      # find card reader from vendor and product ids
      try:
          devices = [evdev.InputDevice(device) for device in evdev.list_devices()]
          self.card_reader = next(device for device in devices
                      if device.info.vendor == vendor and device.info.product ==
product)
      except StopIteration:
          print('Device not found: productID: {0}, vendorID:
{1}'.format(hex(vendor), hex(product)))
          for device in devices:
              device.close()


      # take full control of reader (requires root)
      self.card_reader.grab()
      print('Connected to device: productID: {0}, vendorID:
{1}'.format(hex(vendor), hex(product)))


   def run(self):
       try:
           data = ""
           for event in self.card_reader.read_loop():
               # only get keypress events
               if event.type == evdev.ecodes.EV_KEY:
```

```
                e = evdev.categorize(event)
                # only get downpress
                if e.keystate == e.key_down:
                    # Esperamos por KEY_ENTER
                    if e.keycode == "KEY_ENTER":
                        data = ""
                    else:
                        #data += e.keycode
                        data += parse_key_to_char(e.keycode)
    except KeyboardInterrupt as e:
        print('Caught keyboard interrupt.  Freeing reader...')
        self.card_reader.ungrab()
```