DISEÑO DE PROTOTIPO PARA CONTROL DE ACCESO AL PARQUEADERO DEL CAMPUS DE LA UNIVERSIDAD SANTO TOMÁS SECCIONAL TUNJA

DANIEL YESID MOJICA CRUZ

JORGE ARTURO RODRIGUEZ OCHOA

UNIVERSIDAD SANTO TOMÀS
INGENIERIA ELECTRONICA
TUNJA
2017

DISEÑO DE PROTOTIPO PARA CONTROL DE ACCESO AL PARQUEADERO DEL CAMPUS DE LA UNIVERSIDAD SANTO TOMÁS SECCIONAL TUNJA

DANIEL YESID MOJICA CRUZ

JORGE ARTURO RODRIGUEZ OCHOA

Trabajo de monografía que presenta los resultados del proyecto realizado para optar el título en Ingeniería Electrónica

Directores
Msc. CARLOS ALBERTO CARDONA COY
Msc. ANGELICA MARIA SALAZAR MADRIGAL

UNIVERSIDAD SANTO TOMÁS
INGENIERÍA ELECTRÓNICA
TUNJA
2017

Exoneración de responsabilidades

Las ideas expresadas en este trabajo de grado
Son responsabilidad exclusiva de los autores,
no es la opinión de la Universidad Santo Tomás
o la Facultad de Ingeniería Electrónica

Nota de aceptación

Observaciones

_____
_____
_____
_____
_____

_____
Firma Decano

_____
Firma Primer jurado

_____
Firma Segundo jurado

_____
Firma Director

_____
Firma Director

Tunja, febrero 2017

Mi agradecimiento está dirigido a quienes han forjado mi camino y me han guiado por el sendero correcto, a Dios que en todo momento a estado conmigo, me ha dado la sabiduría y la fortaleza para enfrentar todas las dificultades que se me han presentado a lo largo de la vida, a mis padres y hermanos quienes me han dado todo, me han dado el cariño y la confianza que necesito para mi formación tanto personal como profesional, a mis demás familiares, amigos y docentes que contribuyeron en la terminación de mis estudios profesionales.

Daniel Mojica

Dedico este trabajo a Dios que ha llenado mi camino de oportunidades, y ha dispuesto personas maravillosas que me acompañaron a lo largo de mi formación y me han aportado grandes cosas, a mi familia, que son mi principal apoyo y sin desfallecer me han dado el empujoncito que siempre me ha faltado, sin ellos nada de esto sería posible, a mis amigos que han estado presente y aportaron con su compañía a mi aprendizaje y por ultimo a mi compañero de proyecto por su dedicación, su amistad y su compromiso conmigo.

Jorge Rodriguez

AGRADECIMIENTOS

Agradezco primero a Dios, por la vida, una vida llena de salud y bienestar acompañado de una hermosa familia, gracias Dios por la sabiduría y fortaleza que en momentos de dificultad me guiaron por el lugar correcto y que me llevaron hacia el cumplimiento de este sueño profesional.

A mis Padres, gracias por enseñarme que los sueños hay que perseguirlos a pesar de las dificultades que se presenten y que la distancia no es más que un obstáculo el cual debemos superar para lograr nuestros sueños, por llenar cada momento de mi vida de alegría y de cariño, por ser los mejores maestros en mi vida , dándome fortaleza en cada momento de mi formación, guiándome y ensenándome a conseguir las cosas que verdaderamente han hecho de mí una buena persona y un profesional lleno de muchas enseñanzas y muchos conocimientos que he adquirido y que sé que faltan muchos por obtener. A mis hermanos por ser cada uno de ellos un ejemplo de vida y personas maravillosas que junto con mis padres son el motor para seguir adelante, en fin, a toda mi familia por su incondicional apoyo y sus palabras de aliento y apoyo en cada momento.

A mis amigos y compañeros que a lo largo de mis estudios de pregrado estuvieron siguiendo de cerca el proceso de mi formación y que aun con mi familia lejos hicieron que cada día de soledad y de angustia se tornara más agradable logrando así que no desistiera en búsqueda de mis sueños.

A la Universidad Santo Tomás, en especial a la facultad de Ingeniería Electrónica que tanto mis tutores siguiendo muy de cerca el desarrollo del proyecto de grado, hicieron que se llevara a feliz término la terminación de este proyecto y a los demás docentes quienes aportaron grandes conocimiento y enseñanzas y me fueron formando despacio para mis procesos de investigación y de formación profesional.

Daniel Mojica

AGRADECIMIENTOS

El proyecto ha sido la etapa de aprendizaje y trabajo más importante en mi formación hasta ahora, por eso quiero agradecer a Dios por darme la oportunidad y llenar de bendiciones mi vida, por darme Salud y cuidar a las personas importantes para mí, por darme las herramientas para cumplir mis objetivos.

A mi madre por la batalla que en el silencio afronta, una lucha incansable por ver a su familia de la mejor manera y apoyarme a lo largo de los años con lo que es mejor para mí, gracias por su cariño, su amor, su comprensión y sus consejos; a mi padre porque en su momento supo darme una mano para seguir en este camino y poder estar en esta etapa de mi formación profesional, que sin sus consejos, y su cariño nada de esto sería posible, a mis hermanos que son mis compañeros de vida y que han sido compañía en la formación más importante, y me enseñan cada día cosas nuevas, que son mi ejemplo a seguir. A toda mi familia por compartir, días buenos y malos conmigo, a quien, sin pensar, se ganó un puesto en ella y en mi vida.

A mis amigos y compañeros que han estado siempre, que se han ido, que han vuelto y que, sin estar, con sus buenos deseos y amistad han querido siempre lo mejor para mí, agradezco a las personas con las que compartí 5 años de formación, porque me ayudaron a lo largo del camino y se forjaron lazos de amistad motivados por una meta en común. Por el apoyo de las personas que han hecho parte de este proceso por que han aportado mucho para mi aprendizaje.

Agradezco a la Universidad Santo Tomás en especial al grupo de docentes que me acompaño a lo largo de la etapa de formación profesional, a nuestro director de proyecto, por el acompañamiento en el proceso, y a todo el personal administrativo, de planta física por brindarme la mejor experiencia y calidad.

Jorge Rodríguez

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

GLOSARIO

EPC: El código electrónico de producto, usa tecnología de identificación por radio frecuencia para identificar de manera única a los productos de una cadena de suministro clasificándolos por sus características agregando un número de serie a la información sobre su tipo y fabricante.

Identificación automática: Se refiere a un grupo amplio de tecnologías usadas para ayudar a que sistemas y maquinas puedan adquirir automáticamente datos, permitiendo identificar variedad de objetos o productos para generar mayor eficacia y eficiencia a la hora de ejecutar una tarea y mejorar el manejo de información.

Interfaz: Conjunto de dispositivos, canales y enlaces que permiten conectar unidades distintas entre sí, es la conexión física y funcional que se establece entre dos sistemas.

LF: Término usado para referirse al espectro de radiofrecuencia que opera entre 30 y 300KHz; tiene aplicaciones en RFID, Radiodifusión en AM, y análisis de señales de tiempo. Significa Low Frecuency

Microondas: Se denominan microondas a las ondas electromagnéticas que operan entre 300 MHz y 30GHz ; entre sus características se encuentra que poseen una longitud de onda muy corta y va desde 1mm hasta 30 cm; tienen aplicaciones en radio y televisión, radares, meteorología, etc.

Middleware: Es un subsistema de procesamiento de datos en el que reside un servidor y se comporta como intermediario entre el lector y las aplicaciones, es el encargado del manejo de información, agrupación y filtrado. El Middleware está encargado principalmente de cuatro funciones que son; adquisición de datos, encadenamiento de los datos, gestión de procesos y gestión de dispositivos.

Radio frecuencia: se emplea para denominar las frecuencias del espectro electromagnético.

RFID: Es una técnica de almacenamiento e identificación de datos que soporta la comunicación por radiofrecuencia, se denomina así por sus iniciales en inglés que denotan Radio Frecuency IDentification.

Transponder: Se denomina transponders debido a una combinación de las palabras transmitter (transmisor) y responder (contestador), permite almacenar y enviar datos a un lector a través de ondas de radio, está equipada con una antena encargada de enviar la información y son utilizados para etiquetar productos u objetos.

HF: Término usado para referirse al espectro de ondas que operan entre 3 y30 MHz, significa alta frecuencia y tiene aplicaciones en comunicaciones aéreas, radares, telefonía móvil, etc.

UHF: Término usado para referirse al espectro de ondas que operan entre 30 y 300MHz significa muy alta frecuencia y tiene aplicaciones en FM, Televisión, comunicaciones aéreas, telefonía móvil, etc.

.NET Framework: El Microsoft .NET Framework, es un componente de software que puede ser o es incluido en los sistemas operativos Microsoft Windows. Provee soluciones pre-codificadas para requerimientos comunes de los programas y gestiona la ejecución de programas escritos.

SQL server: Microsoft® SQL Server™ es un sistema de administración y análisis de bases de datos relacionales de Microsoft para soluciones de comercio electrónico, línea de negocio y almacenamiento de datos.

1. INTRODUCCIÓN.

El hombre con el pasar del tiempo ha tenido nuevos retos para mejorar su calidad de vida, generando investigación y por ende nuevos conocimientos que han permitido que surjan nuevos campos multidisciplinares y un sinfín de técnicas, teorías, postulados y tecnologías para dar solución a sus problemas, con ello, del avance tecnológico surge la necesidad de crear ramas de conocimiento especializadas para poder llevar a cabo tareas que requieren de un conocimiento específico.

Con el surgimiento de la economía, el bien capital y el avance de la sociedad humana, las distintas instituciones han tenido la necesidad de generar sistemas de control para llevar un registro detallado de su patrimonio al igual que de diferentes variables para poder optimizar el uso de recursos y monitorear distintos factores; por este motivo en el área de la computación, la ingeniería de Sistemas junto con ingeniería Electrónica se ha venido desarrollando investigación y nuevos avances sobre las bases de datos, que son herramientas capaces de almacenar información y clasificarla según características de los datos que se requieran guardar.

Junto a nuevas tecnologías de bases de datos se ha venido trabajando con sistemas de identificación, capaces de llevar procesos ya sistematizados con bases de datos a un nivel de automatización muy superior, dentro de estas tecnologías emergentes se encuentra el código de barras, identificación óptica e identificación por radio frecuencia, permitiendo una mayor eficiencia a la hora de almacenamiento, clasificación y análisis de información en una base de datos permitiendo dar soluciones eficientes a múltiples situaciones; es por esto que surge la necesidad de fomentar el uso de estas tecnologías para mejorar situaciones cotidianas tal como se presenta en El campus de la Universidad Santo Tomás seccional Tunja.

La universidad Santo Tomás seccional Tunja presta el servicio de parqueadero en la cede campus a los funcionarios, profesores y estudiantes que hayan diligenciado el permiso correspondiente. El protocolo para el ingreso que se usa actualmente en el campus de la universidad pretende brindar a los usuarios la oportunidad de aprovechar el espacio físico que la institución ofrece, sin embargo, el modo de acceso debe optimizarse para la eficiencia en el uso del tiempo y la comodidad del usuario y del vigilante, permitiendo un control restringido para el ingreso a la planta física de la universidad. Es por esto que se decide diseñar un prototipo de sistema electrónico capaz de mejorar la demora de ingreso y la falta de registro para los usuarios a este servicio.

Dentro del objetivo se pretende determinar si la implementación del dispositivo puede presentar mejoras a la situación actual del parqueadero del campus y si presenta un beneficio económico para la institución, debido a que se debe evaluar el factor de consumo del sistema final con respecto a la utilidad que se obtiene actualmente.

2. JUSTIFICACIÓN

En la actualidad el uso de elementos tecnológicos como identificadores por radiofrecuencia han tenido gran auge debido a que presentan grandes ventajas con respecto a otros sistemas como el código de barras, ya que presenta una mayor rapidez, precisión y rango de lectura, por esto se encontró que mediante el uso de un lector RFID, se podría controlar de una manera eficiente el acceso al parqueadero de la Universidad debido a que este tipo de tecnología proporciona mayor alcance evitando todo tipo de contacto de los tranponders con el dispositivo lector, además los Tags de RFID no se pueden falsificar de manera fácil, puesto que en su interior la información está protegida, es 25 veces más rápida que los códigos de barras y poseen varios rangos de alcances con los cuales se pretende dar flexibilidad al sistema para que el usuario pueda acceder al lector, no es necesario tener línea de visión directa para poder leer un Tag de manera que se puede tener almacenado en cualquier parte dentro del rango de lectura; los Tags pueden ser escritos y se puede actualizar la información almacenada en ellos, dando aún más versatilidad a la aplicación deseada.

La información que el sistema adquiere es muy importante para que sea almacenada para su posterior análisis, la fecha y hora tanto como el reporte de históricos hacen que una base de datos sea esencial dentro del sistema que se pretende desarrollar, debe ser capaz de recolectar información acerca del servicio que la institución ofrece y que el dispositivo sea estricto y preciso con el registro de hora de entrada y de salida, junto con un histórico para cada usuario y así ya de forma independiente al sistema se puedan generar estadísticos para toma de decisiones administrativas que puedan mejorar el servicio del parqueadero del campus de la Universidad santo Tomás, el software de desarrollo debe ser capaz de vincular el modulo lector RFID PT-3LA01Z junto con la base de datos y debe permitir el desarrollo de aplicaciones ejecutables para que sea eficaz y que su licencia sea de libre desarrollo.

El protocolo para el ingreso que se usa actualmente no permite a los usuarios el acceso de forma eficiente (tanto en la entrada como en la salida) debido a que la apertura del acceso vehicular se hace de forma manual por el personal de vigilancia que también está encargado del flujo peatonal, lo cual ocasiona demora en el ingreso y no se tiene la información necesaria de la cantidad y disponibilidad de cupos en el parqueadero, por esto lo que se busca con el sistema a desarrollar es brindar a las personas encargadas del control de acceso, la posibilidad de tener información a partir del lector RFID; para obtener datos importantes como tiempo dentro del parqueadero y frecuencia de uso de cada usuario frente a este servicio, también haciendo que el personal encargado esté dispuesto a operar el sistema propuesto y el ingreso peatonal, mediante el sistema se logra evitar que problemas de accesibilidad tomen más tiempo del necesario y que el control de ingreso tanto vehicular como peatonal se haga de la mejor manera.

Posteriormente del desarrollo del prototipo se podrá analizar si en realidad el sistema puede beneficiar a la comunidad estudiantil y si realmente se justifica hacer la inversión en consumo energético, tarjetas, tags y dispositivos adicionales para poder gestionar junto con la persona encargada un complemento que apoye decisiones administrativas para el uso del espacio institucional destinado a la población Tomasina.

3. DESCRIPCION DEL PROBLEMA

3.1 FORMULACIÓN DE PREGUNTAS

- ¿Qué beneficio tendrá la implementación del prototipo de control de acceso al campus de la Universidad Santo Tomás seccional Tunja para los funcionarios, profesores y estudiantes?

- ¿Qué costo puede llegar a tener un sistema que optimice el acceso al parqueadero de la Universidad Santo Tomás sede campus?

3.2 DEFINICION DEL PROBLEMA

La Universidad Santo Tomás seccional Tunja presta el servicio de parqueadero en la sede campus a los funcionarios, profesores y estudiantes que hayan diligenciado el permiso correspondiente. Para el acceso a este servicio, el usuario debe permanecer con el vehículo estacionado en frente de la puerta de la institución mientras el vigilante cerciora que el usuario tenga el permiso, sin embargo, como el encargado del ingreso vehicular también tiene delegada la función de controlar el ingreso peatonal a la universidad, el vehículo debe esperar estacionado en la puerta mientras el vigilante restringe el acceso peatonal y se dedica a controlar el vehicular.
El proceso causa demora tanto para el ingreso vehicular como para el peatonal y no hay un control de acceso preciso y restringido a la planta física de la universidad.

Actualmente no se tiene información de que número de usuarios se encuentran dentro del parqueadero y que numero de cupos están disponibles, también es importante saber quiénes están haciendo un uso frecuente del servicio debido a que los espacios institucionales que la Universidad dispone deben ser aprovechados de la mejor manera para poder brindar la mejor experiencia, es de gran ayuda el poder contar con reportes que puedan dar soporte a decisiones administrativas para un mejor aprovechamiento del parqueadero y un acceso de forma eficaz, seguro y restringido para las personas a las cuales se les haya dado el permiso correspondiente

3.3 DELIMITACIÓN DEL PROBLEMA

La institución cuenta con 83 cupos disponibles para sus funcionarios, profesores y estudiantes en la sede campus. Normalmente la universidad otorga en el semestre 249 permisos según la información recibida por parte del departamento de planta física, debido a que en el manejo de estos espacios los usuarios no estarán al mismo tiempo dentro de las instalaciones y no todos

los usuarios dispondrán del servicio el mismo día, además de brindarle el servicio a una gran parte de la comunidad y no solo a un grupo reducido de personas.

El acceso al campus de la Universidad es controlado por el personal de vigilancia y este no puede tener un registro detallado de cuantos usuarios están dentro de las instalaciones, por esta razón el personal de vigilancia tiene que estar pendiente de la capacidad del recinto y debe desplazar a cerciorarse de los cupos disponibles, Teniendo en cuenta que el proceso de ingreso causa demora tanto para el peatón como para los automóviles, se puede presentar un problema de flujo vehicular dentro del parqueadero tanto en el momento de ingreso, como en el de salida complicando la labor del personal encargado.

4. OBJETVOS

4.1 OBJETIVO GENERAL

- Diseñar prototipo para el control de acceso al parqueadero del campus de la Universidad Santo Tomás Seccional Tunja para optimizar el acceso y llevar un registro detallado por cada usuario.

4.2 OBJETIVOS ESPECIFICOS

- Diseñar un software de control de acceso que permita configurar el modulo lector de RFID PT-3LA01Z para asignar un ID de ingreso a cada usuario para verificar el acceso a las instalaciones.

- Complementar el sistema con dos sensores dentro y fuera de la puerta para que permita la detección del automóvil.

- Diseñar una base de datos en donde se guarden reportes para 250 usuarios en tiempo real de hora y fecha de entrada y salida junto con el número de automóviles que estén dentro del parqueadero de la Universidad Santo Tomás seccional Tunja sede campus.

- Diseñar una interfaz de usuario máquina que permita visualizar fecha, hora de ingreso y hora de salida, nombre, código y facultad para generar un histórico para cada usuario.

5. ALCANCES Y LIMITACIONES

El proyecto ofrece una herramienta que optimiza el proceso de acceso al campus de la Universidad Santo Tomás seccional Tunja, es importante tener registro de uso para el parqueadero, por esta razón el prototipo brinda la posibilidad de visualizar históricos para cada usuario y llevar un registro detallado de entradas y salidas, además su implementación ofrece un acceso rápido y eficiente.

La limitación actual del proyecto consiste en la posibilidad de adquirir una barrera vehicular en el desarrollo del prototipo, entonces las pruebas de funcionamiento se ven limitadas a observar valores eléctricos que sean capaces de ser compatibles con sistemas que existan en el mercado.

6. MARCO TEORICO

6.1 IDENTIFICACIÓN AUTOMÁTICA

Se refiere a un grupo amplio de tecnologías usadas para ayudar a que sistemas y maquinas puedan adquirir automáticamente datos, permitiendo identificar variedad de objetos o productos para generar mayor eficacia y eficiencia a la hora de ejecutar una tarea y mejorar el manejo de información.

Es un conjunto de tecnologías que usan diferentes técnicas pero la forma de operación mantiene una estructura similar y se trata de distintas maneras de codificación de información almacenada sobre un dispositivo específico para que posteriormente pueda ser extraída por un sistema de lectura y enviada a un sistema capaz de analizar dicha información.

Con el avance tecnológico de los últimos años la creación masiva de información junto con las necesidades de llevar controles, analizar bancos y grupos de datos organizados por distintas categorías han surgido varias formas de identificación automática brindándole soluciones a problemáticas en la industria, el comercio, la medicina y un sinfín de campos surgiendo tecnologías de Auto-ID dentro de las técnicas más destacadas como se muestra en la siguiente imagen y de manera relacionada:

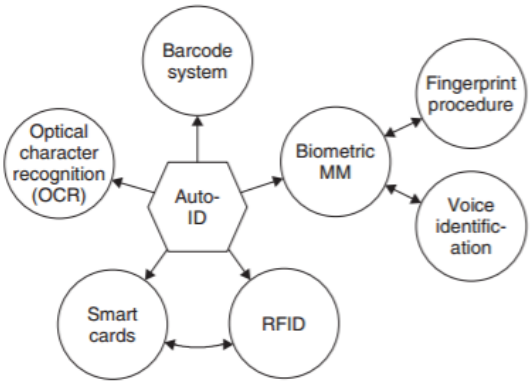

Figura 1. Procedimientos para auto-ID y su relación
Fuentes: Klaus F. (2010) RFID Handbook: Fundamentals and applications in contactless Smart cards, Radio Frequency identification and Neard.

6.1.1 Código de barras

El código de barras es un código binario que comprende un campo de barras y espacios dispuestos en una configuración paralela, están dispuestos de acuerdo con un patrón predeterminado y representan elementos de datos que hacen referencia a un símbolo asociado de tal manera que está compuesta de barras anchas, estrechas y

espacios que se pueden interpretar numéricamente y alfanuméricamente. Se lee por barrido laser óptico, es decir por la diferente reflexión de un haz de laser a partir de las barras negras y blancas. (Finkenzeller, 2010).

6.1.2 Reconocimiento óptico de caracteres

Es un sistema que identifica patrones de líneas, curvas y bordes que compone cada carácter en un conjunto de símbolos definidos previamente procesados y almacenados en la memoria del computador.

Tiene aplicaciones por lo general en el mundo de las finanzas y es muy usado en el procesamiento de documentos, cobro de cheques bancarios, giros, etc también suele combinarse con la tecnología *Magnetic Ink Character Recognition* que garantiza que los caracteres aun puedan ser detectados incluso cuando han sido cubiertos. (Carro & Gonzalez Daniel, 2012)

6.1.3 Procedimientos biométricos

La identificación por procedimientos biométricos se basa en las características biológicas únicas de cada individuo que mediante herramientas electrónicas que son capaces de detectar rasgos, patrones, y formas de las características personales de los usuarios, haciendo más seguro el sistema, para luego guardar esta (Carro & Gonzalez Daniel, 2012)información para su posterior análisis y aplicación.

Existen diferentes técnicas para reconocer patrones biométricos, así mismo también está la posibilidad de reconocer distinta información de una misma persona basados en sistemas independientes según su necesidad o área biológica de trabajo.

6.1.3.1 identificación de voz.

El objetivo de la identificación por voz es adquirir esta señal mediante un micrófono conectado a un ordenador que la descompone y convierte en señal digital para luego compararla con un patrón existente y decidir si corresponde o no para ejecutar una acción o completar un proceso.

6.1.3.2 Identificación por huellas dactilares

Mediante un transductor óptico se digitaliza un patrón de la huella dactilar del usuario y se compara con una referencia existente.

6.1.4 tarjetas inteligentes

Las tarjetas inteligentes son dispositivos que tienen la capacidad de almacenamiento de datos y que en su interior poseen un microprocesador, cuando existe un contacto con el lector, este suministra un pulso de energía y de reloj, permitiendo una comunicación bidireccional. Una de las principales ventajas de la tarjeta inteligente es el hecho de que ella pueda protegerse contra el acceso y la manipulación no deseados. (Finkenzeller, 2010) ((ORSI), 2007)

6.1.5 Sistemas RFID

Consiste básicamente en que una etiqueta o transponder genera una señal de radio a determinada frecuencia transmitiendo información al lector para que posteriormente un ordenador analice la información y ejecute una acción.

En la siguiente tabla se muestra una comparación entre todas las tecnologías de identificación previamente descritas.

System parameters	Barcode	OCR	Voice recognition	Biometry	Smart card	RFID systems
Typical data quantity (bytes)	1–100	1–100	–	–	16–64 k	16–64 k
Data density	Low	Low	High	High	Very high	Very high
Machine readability	Good	Good	Expensive	Expensive	Good	Good
Readability by people	Limited	Simple	Simple	Difficult	Impossible	Impossible
Influence of dirt/damp	Very high	Very high	–	–	Possible (contacts)	No influence
Influence of (optical) covering	Total failure	Total failure	–	Possible	–	No influence
Influence of direction and position	Low	Low	–	–	Unidirectional	No influence
Degradation/wear	Limited	Limited	–	–	Contacts	No influence
Purchase cost/reading electronics	Very low	Medium	Very high	Very high	Low	Medium
Operating costs (e.g. printer)	Low	Low	None	None	Medium (contacts)	None
Unauthorised copying/modification	Slight	Slight	Possible* (audio tape)	Impossible	Impossible	Impossible
Reading speed (including handling of data carrier)	Low ~4 s	Low ~3 s	Very low >5 s	Very low >5–10 s	Low ~4 s	Very fast ~0.5 s
Maximum distance between data carrier and reader	0–50 cm	<1 cm Scanner	0–50 cm	Direct contact**	Direct contact	0–5 m, microwave

Tabla1. Comparación entre las tecnologías de identificación automática.
Fuente: Klaus F. (2010) RFID Handbook: Fundamentals and applications in contactless Smart cards, Radio Frequency identification an Neard.

6.2 DEFINICIÓN DETALLADA DE RFID

Es una técnica de almacenamiento e identificación de datos que soporta la comunicación por radiofrecuencia, se denomina así por sus iniciales en inglés que denotan Radio Frecuency IDentification; se trata de una tecnología que puede transmitir de forma automática datos usando ondas electromagnéticas empleando una etiqueta y un dispositivo lector .

Dentro de su eventualidad la tecnología RFID ofrece la auto-identificación de un objeto que contiene un dispositivo emisor de radio también llamados transponders, con la característica que se pueda reescribir, asignando la información necesaria para reconocer el objeto o producto; el lector realiza una solicitud a la etiqueta para que ella suministre la información almacenada de tal manera que por medio de radiofrecuencia, el transponder, brinde los datos contenidos para que un sistema o dispositivo sea el encargado de analizar dicha información, RFID está desarrollado en un concepto muy cercano al sistema de código de barras, debido a que el principal objetivo de ambas tecnologías es la identificación mediante un código, la principal diferencia nace en que el código de barras usa una señal de tipo óptica para adquirir la información sobre una impresión láser, en cambio RFID utiliza radiofrecuencia en diferentes bandas de transmisión que determina una potencia y un alcance especifico dependiendo de las características de cada sistema.

6.2.1 HISTORIA Y GENERALIDADES

El origen de la identificación por radio frecuencia empieza con la segunda guerra mundial por varios países motivados por identificar las aeronaves, los militares emplearon tecnología de radar para detectar los aviones mientras se encontraban dentro del perímetro de detección, pero el problema es que solo podían averiguar si habían aviones o no, ellos no sabían de qué nacionalidad eran las aeronaves dentro del rango de medición, por ello el ejército Alemán descubrió que si balanceaban sus aviones al volver a la base, la señal de radio que se captaba en las instalaciones cambiaria y podrían distinguir a los aviones aliados, de tal manera que se desarrolló el primer dispositivo RFID de forma pasiva.

Los británicos encontraron una solución aún más eficiente y segura desarrollando el primer sistema activo de RFID debido a que colocaron un transmisor en cada aeronave aliada, de manera que al recibir señales de las estaciones de radar, se devolvía una señal que les permitía en tierra identificarlos.

Más tarde en los años 50 y 60 científicos de varios países avanzaron estudios e investigaciones para poder explicar cómo identificar objetos de forma remota y empezaron a fabricarse sistemas de seguridad anti robo empleando ondas de radio para determinar si un producto había sido pagado o no dentro de la tienda, para que luego en 1973 Mario W. Cardillo patentara por primera vez en la historia un tag RFID activo con memoria modificable y el mismo año Charles Walton recibió una patente para un sistema de cerradura que abría las puertas sin necesidad de llave . Dentro de esta misma década el gobierno delego varias investigaciones sobre el tema,

desarrollando nuevas aplicaciones para el seguimiento de materiales nucleares y ganado vacuno insertando bajo la piel de estos transponders para poder identificar que animal había sido vacunado y el control de edad individual.

En los años posteriores se desarrollaron por parte de distintas compañías los sistemas que operaban en diferentes bandas de radiofrecuencia, empezando por los de 125KHz, denominados como Low Frecuency, y más adelante avanzarían en el espectro a 13,56Mhz que se conoce como alta frecuencia, pero IBM trabajaría más a fondo y sus ingenieros desarrollaron y patentaron el sistema UHF que por sus siglas significa Ultra High Frequency.

Con el avance de la tecnología RFID a la banda UHF las empresas Uniform Code Council EAN International entre otras decidieron emprender un proyecto llamado el Auto-ID Center en el Instituto Tecnológico de Massachussets que pretendía investigar la posibilidad de colocar tags RFID de bajo costo a todos los productos de una línea de suministro, y poner el número de serie únicamente como dato de identificación para que las características asociadas al producto fueran almacenadas en una base de datos a la cual se podía entrar por internet; más tarde entre 1999 y 2003 el Auto-Id center con la ayuda de nuevas compañías desarrollo Electronic Product code (EPC) que mantiene un código que identifica de forma precisa un objeto o producto, dentro de los números de EPC, mantiene una serie de dígitos que identifican al fabricante y las características del producto. ((ORSI), 2007)

Actualmente esta tecnología presenta mejoras en la capacidad de emisión y recepción, así como en la distancia, lo cual ha llevado a extender su uso a lo largo del mundo con distintas aplicaciones industriales, empresariales, etc.

6.2.2 Principios de funcionamiento

Existe gran variedad de sistemas RFID cada uno con diferente aplicación y estructura, pueden tener diferencias significativas en cuanto al desarrollo tecnológico, pero es el mismo principio de funcionamiento, empezando por equipar con transponders todos los objetos o productos a identificar , seguidamente la antena del módulo lector emite un campo de radiofrecuencia que energiza los tags, luego cuando una etiqueta ingresa a dicho campo, envía la información contenida utilizando la energía temporal recibida por parte del lector , inmediatamente el lector recibe la información y los transmite a un ordenador para su procesamiento y análisis.

Un sistema RFID está basado principalmente en trasnponder, interrogador o lector, host u Ordenador y un servidor para contener la información, y están dispuestos como se muestra en la figura 3, se observa que existen tres tipos de interfaces: lector-sistema de información, lector-etiqueta y usuario-servidor.

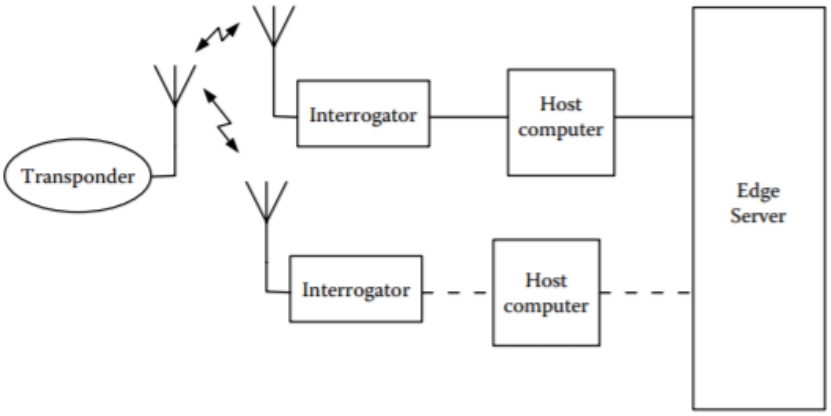

Figura 2. Estructura básica de un sistema RFID
Fuente: Albert L. (2011). RFID Design Fundamentals and Applications

Los dispositivos que almacenan y transportan la información se denominan transpondedores o etiquetas y se muestran en la Figura 4

Figura.3 Varios tipos de transpondedores utilizados en sistemas RFID.
Fuente: RFID - Fundamentos de diseño y aplicación.

6.2.2.1 Transponders o etiquetas RFID

Se denomina transponders debido a una combinación de las palabras

transmitter (transmisor) y responder (contestador), permite almacenar y enviar datos a un lector a través de ondas de radio, está equipada con una antena encargada de enviar la información y son utilizados para etiquetar productos u objetos.

Los transponders deben contener circuitos capaces de recolectar energía de los campos electromagnéticos generados por el interrogador o lector, en su interior deben existir elementos de memoria y circuitos de control. Existe una gama variada de etiquetas que varían, incluyendo desde memorias de solo lectura denominadas ROM, mientras que los de más altas especificaciones contienen memorias de acceso aleatorio y memorias de uso más específico como las PROM o EEPROM.

Por lo general la cadena de información está contenida en la memoria ROM, así mismo esta gama de etiquetas también ofrece la capacidad de almacenar un solo bit abaratando su costo, en se puede obtener solo dos estados lógicos, usada en aplicaciones de comercios minoristas y las que ofrecen la posibilidad de almacenar varios Kilobits, las gamas de 512 bits normalmente tienen la opción de ser reprogramables para que el número de identificación individual pueda tener información adicional para su aplicación. (Lozano, 2011)

Un transponer está básicamente compuesto principalmente por un microchip y una antena también puede incluir otro tipo de circuitos que añaden más funciones como registros; dentro del microchip se contiene una lógica de seguridad, una lógica de control y una lógica interna.

La información de la etiqueta se transmite modulada en amplitud (ASK, Amplitude Shift Keying), frecuencia (FSK, Frequency Shift Keying) o fase (PSK, Phase Shift Keying). Es decir, para realizar la transmisión se modifica la amplitud, frecuencia o fase de la señal del lector. Típicamente la modulación más utilizada es la ASK debido a su mayor sencillez a la hora de realizar la demodulación. La frecuencia utilizada por el transpondedor, en la gran mayoría de los casos, coincide con la emitida por el lector. Sin embargo, en ocasiones se trata de una frecuencia subarmónica (submúltiplo de la del lector) o incluso de una frecuencia totalmente diferente de la del lector (no armónica). La antena que incorporan las etiquetas para ser capaces de transmitir

los datos almacenados en el microchip puede ser de dos tipos: Un elemento inductivo (bobina) o un dipolo. (P., 2008)

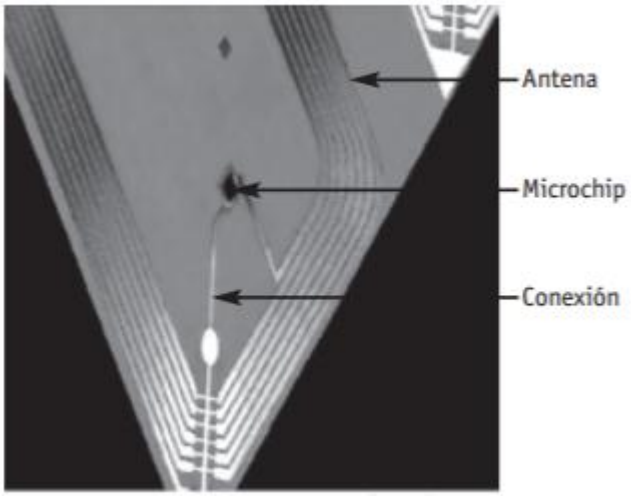

Figura 4.: Composición interna de un Transponder.
Fuente: Auto-ID center

6.2.2.1.1 Etiquetas Pasivas

Las etiquetas pasivas funcionan con la energía que el lector proporciona mediante un campo, ya que no poseen batería, les da la posibilidad de ser diseñados en una amplia gama de formas y tamaños haciéndolos más flexibles y duraderos, abaratando costos, pero requieren una mayor energía de emisión por parte del interrogador, presentan desventajas en cuanto a los tags activos, debido a que presentan interferencias electromagnéticas en ambientes con bastante flujo de emisiones radiales.

Poseen circuitos resonantes, y su antena debe estar diseñada para que pueda obtener la energía suficiente para poder transmitir información.

6.2.2.1.2 Etiquetas Activas

Poseen una batería interna, y su capacidad de almacenamiento es mejor, su implementación es más costosa, pero emite señales más potentes y precisas por tanto su alcance cubre más distancia, tiene mejor desempeño en ambientes hostiles de interferencia electromagnética. Su tamaño es mayor y el tiempo de respuesta es veloz, ofreciendo un ratio de lectura más fuerte en comparación con las etiquetas pasivas. Su periodo de vida es menor, porque dependen de la duración de su batería, haciendo que sea la mayor limitante

	Etiquetas Activas	*Etiquetas Pasivas*
Incorporan batería	Sí	No
Coste	Mayor	Menor
Tiempo de vida	Limitado	Casi ilimitado
Cobertura	Mayor	Menor
Capacidad datos	Mayor	Menor

Tabla 2. Cuadro comparativo entre etiquetas activas y pasivas
Fuente: Javier P. (2008), Tecnología de identificación por radiofrecuencia (RFID): Aplicaciones en el ámbito de la salud.

En la tabla número 2 se puede observar las diferencias entre las etiquetas activas y pasivas, sus ventajas comparativas.

También a continuación se muestra las diferentes bandas utilizadas en la tecnología RFID clasificadas por zonas de operación:

Frecuencia	Denominación	Rango
125 kHz – 134 kHz	LF (Baja Frecuencia)	Hasta 45 cm.
13,553 MHz – 13,567 MHz	HF (Alta Frecuencia)	De 1 a 3 m.
400 MHz – 1000 MHz	UHF (Ultra Alta Frecuencia)	De 3 a 10 m.
2,45 GHz – 5,4 GHz	Microondas	Más de 10 m.

Tabla 3. Clasificación de frecuencias por bandas según zona de operación
Fuente: red.es

6.2.2.2 Modos de transmisión

Existen dos mecanismos de propagación de la información de la tecnología RFID

6.2.2.2.1 Acoplamiento inductivo

Maneja cobertura baja debido a que trabaja en el campo cercano y se tiene que tener en cuenta la orientación de la antena para que se pueda extraer la información del transponder, es muy sensible a interferencias electromagnéticas y trabaja con antenas pasivas por que poseen un rango de lectura muy bajo.

Usualmente contiene un microchip y una bobina de gran área conductora que funciona como antena, para esta función la bobina de la antena del interrogador emite un campo electromagnético fuerte de alta frecuencia, que puede generar las corrientes suficientes para activar el lector. La eficiencia de la transferencia de potencia entre la antena del interrogador y el Transponder es proporcional a su funcionamiento.

6.2.2.2.2 Propagación por ondas EM

A diferencia del acoplamiento inductivo trabaja a altas frecuencias, suele usar etiquetas activas, pero también en este tipo de tecnologías funciona las pasivas pero con más limitaciones, además de trabajar en campo lejano.

Por lo general este tipo de propagación se usa en distancias superiores a 1m, y funcionan a frecuencias UHF de hasta 915 MHz, en algunos se casos se estudia la posibilidad de usar transponders de bajo consumo de tal manera que requiera la menor potencia posible para que se pueda leer la información a una distancia considerable debido a que la potencia de consumo que use la etiqueta será restada de la potencia de transmisión de la antena del lector.
También se pueden clasificar las etiquetas dependiendo de su frecuencia de operación, ya que puede llegar a determinar cualidades de transmisión velocidad y tiempo de lectura, etc.

Protocolo	Frecuencia	Tipo de etiqueta
Clase 0	UHF	solo lectura
Clase 0 plus	UHF	lectura-escritura
Clase 1	HF/UHF	una escritura múltiples lecturas
Clase 1 Gen2	UHF	una escritura múltiples lecturas
Clase 2	UHF	lectura y escritura
Clase 3	UHF	clase 2 más batería y sensores
Clase 4	UHF	etiquetas activas
Clase 5	UHF	clase 4+ capacidad de lectura

Tabla 4. Protocolos EPCGlobal para RFID
Fuente: Javier P. (2008), Tecnología de identificación por radiofrecuencia (RFID): Aplicaciones en el ámbito de la salud.

6.2.2.3 Lector de RFID

También llamados interrogadores, son capaces de leer la información almacenada dentro del transponder enviando una orden a la etiqueta para transmitir la información en su interior creando un campo magnético cuyo radio de acción cambia dependiendo las características de la antena , la potencia del

lector y de la frecuencia de emisión, principalmente tiene dos formas de operación y depende de cada fabricante y de las necesidades que el sistema requiera para trabajar con etiquetas pasivas o activas; si la etiqueta es pasiva el lector genera un campo de radiofrecuencia que energiza la etiqueta y el chip contenido en su interior para que la antena del transponder envié la información. Cuando el sistema utiliza etiquetas activas no es necesario que el interrogante o lector energice el tag, porque el mismo puede iniciar la comunicación. El lector decodifica y pone en contacto con el sistema de control a través de una aplicación denominada (Object Name Service).

Básicamente todos los lectores RFID contienen en su interior para un oscilador local, un modulador, amplificador de potencias y un circuito de sintonización para el sistema de emisión de ondas por radio, para el sistema de recepción, contiene un demodulador, un amplificador de señales y un filtro pasabanda; ambos sistemas están conectados a un microcontrolador que envía y recibe información de un host dispuestos como se muestra en el siguiente diagrama de bloques:

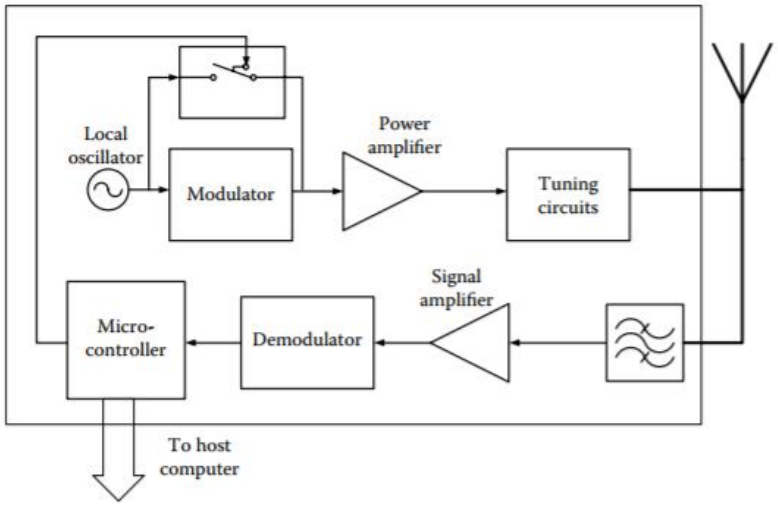

Figura 5. Diagrama d bloques de un transmisor RFID genérico
Fuente: Albert L. (2011). RFID Design Fundamentals and Applications

Un gran inconveniente de los trasmisores es la radiación espurea que se produce cuando una cantidad de energía cae fuera de la banda de transmisión requerida, ocurre cuando los interrogadores quieren trasmitir órdenes al transponder, es decir, que se pretende escribir nueva información en la memoria del tag, por lo que el interrogador debe ser los más eficiente y preciso para

enviar la potencia adecuada a la banda de transmisión sino que también debe tener en cuenta las pérdidas de potencia por emisión de calor, puesto que el calor excesivo puede reducir la vida útil del sistema.

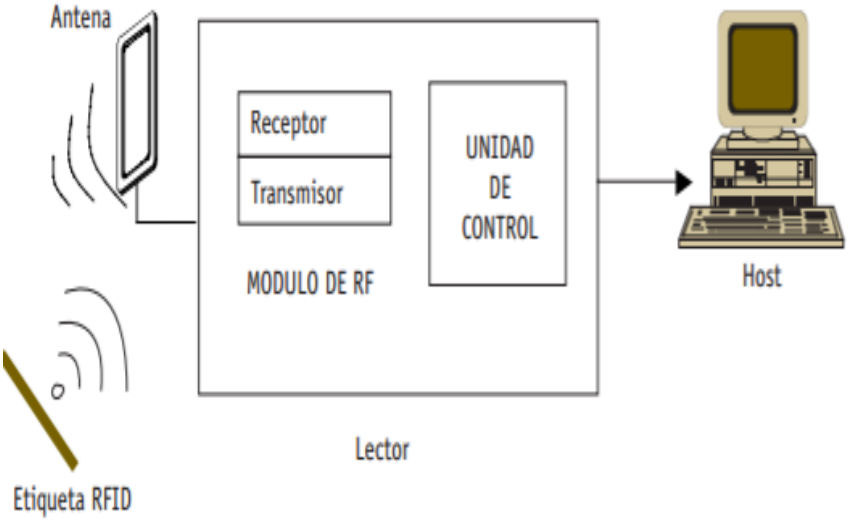

Figura 6. Esquema de un lector de RFID
Fuente: Javier P. (2008), Tecnología de identificación por radiofrecuencia (RFID): Aplicaciones en el ámbito de la salud.

Cuando los transmisores interactúan con tranponders de solo lectura, ellos emiten una señal ininterrumpida de radio y recibe la información de los tags, a este modo de trabajo se le conoce como *Tag talks,* pero cuando los tags son activos, el interrogador envía comandos para leerlos o escribirlos denominados (FRR) y bypass de lectura rápida (FRB) (Lozano, 2011)

6.2.2.4 Middleware

Es un subsistema de procesamiento de datos en el que reside un servidor y se comporta como intermediario entre el lector y las aplicaciones, es el encargado del manejo de información, agrupación y filtrado.

El Middleware está encargado principalmente de cuatro funciones que son; adquisición de datos, encadenamiento de los datos, gestión de procesos y gestión de dispositivos, es el responsable de la extracción, agrupación, además de facilitar la integración de las redes de elementos y sistemas RFID de la aplicación; es capaz de poner en marcha eventos en función de las normas del sistema.

6.2.2.5 SEGURIDAD

RFID es un blanco de ataques informáticos debido a que hay sistemas que pueden contener información valiosa, o al ser vulnerados puede afectar el funcionamiento de la aplicación y acarrear pérdidas importantes para la organización que hace uso de sus prestaciones, dentro de las formas de vulnerar un sistema RFID se considera evitar la comunicación entre el interrogador y la etiqueta, de esta manera se puede afectar obstruyendo la línea de vista del transponder y el lector con pantallas de metales.
Dentro de los posibles ataques se pueden considerar técnicas como suministrar información falsa que al pasar por el lector es aceptada por el sistema; se puede infiltrar en las bases de datos y se puede insertar datos, donde normalmente se espera. Se puede interceptar una seña RFID, con un dispositivo que la replique y la almacene, luego los datos pueden retransmitirse e ingresar al sistema que los acepta como correctos, además de existir otras técnicas de ataques también se puede colapsar el sistema , haciendo que se sature de datos.

Se pueden usar algoritmos adecuados para evitar que en los sistemas RFID hayan infiltraciones; los sistemas de alta seguridad, deben tener un plan de contingencia ante enfrentar amenazas comunes , como hallar un portador de datos extranjero en la zona de interrogación el lector con la intención de descifrar códigos y modificarlos; interceptar información y replicarla , así mismo existen aplicaciones que no requieren una función de seguridad, como por ejemplo , en la automatización industrial, se harían innecesariamente caras, además existen operaciones muy sencillas para proteger la información como la operación XOR donde se tiene una llave o key, de tal manera que a la información requeridas se le aplica la XOR y queda codificada, asi mismo para des encriptar se hace el mismo procedimiento con la llave para volver a la información original.

A continuación se muestran dos técnicas de proteger la información:

6.2.2.5.1 Autenticación mutua simétrica

Se basa de acuerdo con la norma ISO/IEC 9798-2 que dictamina que debe haber una autenticación mutua , en este procedimiento, todas las etiquetas y el receptor poseen un misma clave usada para encriptar denominada

K. cuando el transponder entra en la zona de lectura, el interrogador protege la aplicación de la manipulación, y la etiqueta protege la información contenida en su interior; para autenticar el tag, el lector envía un comando denominado GET_CHALLENGE, en la etiqueta se activa la generación de un código aleatorio denominado RA que luego envía al lector; de nuevo el lector genera un numero nuevo denominado RB , usando la clave k el lector calcula una trama de datos cifrada, que contiene los números aleatorios generados y los envía al transponder, de tal manera que cuando la etiqueta lo reciba, lo descifre y compare el numero RA ; si coincide con el enviado anteriormente calcula una nueva trama de datos con un nuevo valor para RA , ejecuta un bloque de control que asigna nuevos valores a los bloques de control y genera una nueva trama de datos, y los envía al lector que lo descifra y obtiene el dato RB que es nuevamente comparado con el primer dato que se había enviado y se corresponde se establece la comunicación entre ambas partes.

6.2.2.5.2 transferencia de datos encriptados

Se usa de nuevo una clave y un algoritmo secreto para generar un cifrado y un descifrado; puesto que la transmisión de datos cifrados se verá protegida en cuanto a que el atacante no conoce la clave ni el algoritmo, ahora se pueden desarrollar diferentes combinaciones posibles para obtener un tipo distinto de cifrado, por ejemplo si para cifrar y descifrar se usa la misma clave, se estaría hablando de un mecanismo de cifrado simétrico, pero si se usa una clave distinta para cifrar y descifrar, se estaría refiriendo a una técnica de encriptación asimétrica, de la misma manera también se usan técnicas en donde se cifra cada carácter antes de ser enviado entonces se denomina encriptación secuencial o se puede hacer un conjunto de caracteres para luego encriptarlos, se estaría hablando de cifrado por bloque. (Finkenzeller, 2010)

6.3 Estandarización y normativa RFID

Actualmente existen varias instituciones dedicadas a desarrollar normativa sobre este tipo de tecnologías según sus interese y aplicaciones; dentro de estas compañías se encuentran: La ISO (International Electrotechnical Organization for Standardization), la IEC (international Electrothecnical

commisio), ASTM (American Society for Testing and Materials) e incluso EPCglobal.

Algunos estándares para la normativa por parte de estas empresas son:

ASTM D7434- estandariza los métodos de test de rendimiento de los transponders pasivos en cargas.

ASTM D7435- Determina la normativa para realizar test a transponders pasivos que se usan en contenedores cargaos

ETSI EN 302 208-1 v1.4.1 (2011-07)- describe características técnicas y formas de medida para los equipos de radio; en este caso aplica para los interrogadores o lectores de RFID

EPC UHF Class1 Gen2- estandariza las etiquetas RFID pasivas para que puedan ser identificados en la cadena de suministro a nivel mundial.

ISO/IEC 11784-11785 – contienen la estructura del código de RFID para animales

ISO/IEC 10536- define las características físicas, la interfaz y la inicialización, además de definir los protocolos anticolisión de tarjetas inteligentes.

ISO/IEC 15961-15963- definen los estándares para la identificación automática y adquisición de datos para gestión de objetos.

ISO/IEC 18000 describe las diferentes tecnologías y ratifica la banda de frecuencias aceptadas a nivel internacional.

ISO/IEC 18047 describe los métodos de prueba y de rendimiento de dispositivos.

6.4 Aplicaciones de RFID

La identificación por radiofrecuencia es una tecnología muy flexible y útil para las necesidades de tipo informático que se presentan hoy en día; cada vez se complica el manejo de la información y la captura de datos de forma manual causa demora y es susceptible al error humano, es por esto que se están implementando técnicas de detección automática en procesos en donde se necesita llevar un control detallado de los productos. La tecnología RFID y sus avances permite hoy llevar información detallada acerca de cientos de productos u objetos, incluso procesos alimenticios y ganaderos de forma eficiente y sin error alguno es por esto que está incursionando en los sectores de la industria, la tecnología, economía y agrícola.

Algunas de las aplicaciones son:

- En tiendas, EPCglobal ha desarrollado un protocolo con una etiqueta pasiva con el cual se puede determinar las características del producto además de verificar si ya está pago el objeto para evitar robo dentro de los comercios.
- Se usa la tecnología RFID está incursionando en el mundo de la identificación de mascotas mediante la implantación subcutánea, y dentro del transponder se almacena informacio del propietario, y datos sanitarios del animal.
- Control de acceso a edificios inteligentes.
- Seguimiento de ganado bovino para ceba.
- Logística y gestión de almacene mayoristas.
- Sistemas de pago de peajes.
- Control de acceso a parqueaderos.

6.5 Reader Protocol

Se trata de una hoja técnica con un conjunto de técnicas y de manejo de trama de datos que se usan para acceder a los registros del lector PT-3LA01Z ofrecido por el fabricante donde la información de interés para el desarrollo del proyecto se centra en , leer, escribir, bloquear un transponder

6.5.1 Formato de comandos:

El cuadro de comandos de datos para uso operativos del módulo lector tiene dos formas de flujo: host reader y reader host es como se muestra a continuación:

Trama de datos host-reader

Packet Type	Length	Command Code	Device Number	Command Data	...	Command Data	Command Data	Checksum
0xA0	n +3	1 byte	1 byte	Byte 1		Byte n-1	Byte n	cc

Tabla 5. Trama de datos host-reader
Fuente: Plintec

Trama de datos Reader host

Packet Type	Length	Response Code	Device Number	Response Data	...	Response Data	Response Data	Checksum
0xE0	n +3	1 byte	1 byte	Byte 1		Byte n-1	Byte n	cc

Tabla 6. Trama de datos reader-host
Fuente: Plintec

En las figuras anteriores esta la trama de datos que se genera en una comunicación bidireccional, a continuación se presenta la identificación de un tag; la lectura y la escritura.

6.5.1.1 Identificación EPC de un Tag

Para la identificación del tag es necesario observar la trama de datos que se recibe tipo host-reader.

El host envía la siguiente cadena:

Reply	Data length	Command	Device Number	Checksum
Data0	Data1	Data2	Data3	Data5
A0	03	82	usercode	checksum

Tabla 7.Cadena de datos para la identificación de un Tag.
Fuente: Plintec

Por tanto el código recibido es: A0, 03, 82, 00, DB

Cuando el lector no reconoce el código, él envía una trama de datos con el código de valor de cabecera (E4 04 82 00 05)

Por el contrario cuando el lector reconoce la trama recibida el transmite un trama de datos con el código de valor de cabecera (E0 10 82 01) e inmediatamente reconoce el ID

6.5.1.2 Lectura de un tag

Cuando el tag entra dentro del campo de lectura del interrogador envía una trama de datos hacia el lector que tiene la siguiente estructura:

Reply	Data length	Command	Device Number	Memory location	Address	Read length (word)	Checksum
Data0	Data1	Data2	Data3	Data4	Data5	Data6	Data7
A0	06	80	usercode	MemBank	addr	Length	checksum

Tabla 8.trama de datos para la lectura de un tag.
Fuente: Plintec

El código enviado es: A0 06 80 00 01 02 01 D6

Inmediatamente el sistema envía el código E0 09 80 04 01 02 01 con Status cheksum 49, esto quiere decir que la información ha sido recibida correctamente.

La trama de datos recibida se puede interpretar de la siguiente manera:

E0: lee completamente la trama recibida.
08: tamaño del dato.
80: comando de lectura de etiqueta.
Código de usuario: numero
01: banco de memoria
02: dirección
12 34: datos de lectura
4E: Checksum

6.6 Visual Studio

Visual Studio es un conjunto completo de herramientas de desarrollo para la generación de aplicaciones web ASP.NET, Servicios Web XML, aplicaciones de escritorio y aplicaciones móviles. Visual Basic, Visual C# y Visual C++ utilizan todos el mismo entorno de desarrollo integrado (IDE), que habilita el uso compartido de herramientas y facilita la creación de soluciones en varios lenguajes. Asimismo, dichos lenguajes utilizan las funciones de .NET Framework, las cuales ofrecen acceso a tecnologías clave para simplificar el desarrollo de aplicaciones web ASP y Servicios Web XML.

6.6.1 C Sharp (C#)

C# es un lenguaje orientado a objetos elegante y con seguridad de tipos que permite a los desarrolladores compilar diversas aplicaciones sólidas y seguras que se ejecutan en .NET Framework. Puede utilizar C# para crear aplicaciones cliente de Windows, servicios Web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos, y mucho, mucho más. Visual C# proporciona un editor de código avanzado, cómodos diseñadores de interfaz de usuario, depurador integrado y numerosas herramientas más para facilitar el desarrollo de aplicaciones basadas el lenguaje C# y .NET Framework.

6.6.1.1 lenguaje C#

La sintaxis de C# es muy expresiva, pero también es sencilla y fácil de aprender. La sintaxis de C# basada en signos de llave podrá ser reconocida inmediatamente por cualquier persona familiarizada con C, C++ o Java. Los desarrolladores que conocen cualquiera de estos lenguajes pueden empezar a trabajar de forma productiva en C# en un

plazo muy breve. La sintaxis de C# simplifica muchas de las complejidades de C++ y proporciona características eficaces tales como tipos de valor que admiten valores NULL, enumeraciones, delegados, expresiones lambda y acceso directo a memoria, que no se encuentran en Java. C# admite métodos y tipos genéricos, que proporcionan mayor rendimiento y seguridad de tipos, e iteradores, que permiten a los implementadores de clases de colección definir comportamientos de iteración personalizados que el código cliente puede utilizar fácilmente. Las expresiones Language-Integrated Query (LINQ) convierten la consulta fuertemente equipada en una construcción de lenguaje de primera clase.

Como lenguaje orientado a objetos, C# admite los conceptos de encapsulación, herencia y polimorfismo. Todas las variables y métodos, incluido el método Main que es el punto de entrada de la aplicación, se encapsulan dentro de definiciones de clase. Una clase puede heredar directamente de una clase primaria, pero puede implementar cualquier número de interfaces. Los métodos que reemplazan a los métodos virtuales en una clase primaria requieren la palabra clave override como medio para evitar redefiniciones accidentales. En C#, una struct es como una clase sencilla; es un tipo asignado en la pila que puede implementar interfaces pero que no admite la herencia.

Además de estos principios básicos orientados a objetos, C# facilita el desarrollo de componentes de software a través de varias construcciones de lenguaje innovadoras, entre las que se incluyen las siguientes:
Firmas de métodos encapsulados denominadas delegados, que habilitan notificaciones de eventos con seguridad de tipos.
Propiedades, que actúan como descriptores de acceso para variables miembro privadas.
Atributos, que proporcionan metadatos declarativos sobre tipos en tiempo de ejecución.
Comentarios en línea de documentación XML.
Language-Integrated Query (LINQ) que proporciona funciones de consulta integradas en una gran variedad de orígenes de datos.

Si necesita interactuar con otro software de Windows, como objetos COM o archivos DLL nativos de Win32, podrá hacerlo en C# mediante un proceso denominado "interoperabilidad". La interoperabilidad habilita los programas de C# para que puedan realizar prácticamente las mismas tareas que una aplicación C++ nativa. C# admite incluso el uso de punteros y el concepto de código "no seguro" en los casos en que el acceso directo a la memoria es totalmente crítico.

El proceso de compilación de C# es simple en comparación con el de C y C++, y es más flexible que en Java. No hay archivos de encabezado independientes, ni se requiere que los métodos y los tipos se declaren en un orden determinado. Un archivo de código fuente de C# puede definir cualquier número de clases, structs, interfaces y eventos.

6.6.2 Plataforma .NET Framework.

Los programas de C# se ejecutan en .NET Framework, un componente que forma parte de Windows y que incluye un sistema de ejecución virtual denominado Common Language Runtime (CLR) y un conjunto unificado de bibliotecas de clases. CLR es la implementación comercial de Microsoft de CLI (Common Language Infrastructure), un estándar internacional que constituye la base para crear entornos de ejecución y desarrollo en los que los lenguajes y las bibliotecas trabajan juntos sin ningún problema.

El código fuente escrito en C# se compila en un lenguaje intermedio (IL) conforme con la especificación CLI. El código de lenguaje intermedio y recursos tales como mapas de bits y cadenas se almacenan en disco en un archivo ejecutable denominado ensamblado, cuya extensión es .exe o .dll generalmente. Un ensamblado contiene un manifiesto que proporciona información sobre los tipos, la versión, la referencia cultural y los requisitos de seguridad del ensamblado.

Cuando se ejecuta un programa de C#, el ensamblado se carga en CLR, con lo que se pueden realizar diversas acciones en función de la información del manifiesto. A continuación, si se cumplen los requisitos de seguridad, CLR realiza una compilación Just In Time (JIT) para convertir el código de lenguaje intermedio en instrucciones máquina nativa. CLR también proporciona otros servicios relacionados con la recolección de elementos no utilizados automática, el control de excepciones y la administración de recursos. El código ejecutado por CLR se denomina algunas veces "código administrado", en contraposición al "código no administrado" que se compila en lenguaje máquina nativo destinado a un sistema específico. En el diagrama siguiente se muestran las relaciones en tiempo de compilación y tiempo de ejecución de los archivos de código fuente de C#, las bibliotecas de clases de .NET Framework, los ensamblados y CLR. (MSDN, n.d.)

6.7 SQL server

Microsoft® SQL Server™ es un sistema de administración y análisis de bases de datos relacionales de Microsoft para soluciones de comercio electrónico, línea de negocio y almacenamiento de datos.

SQL server ofrece distintas tecnologías y servicios disponibles dentro del paquete de instalación, dentro de ellas se puede encontrar:

6.7.1 Motor de base de datos

El Motor de base de datos es el servicio principal para almacenar, procesar y proteger datos. El Motor de base de datos proporciona acceso controlado y procesamiento rápido de transacciones para cumplir los requisitos de las aplicaciones consumidoras de datos más exigentes de su empresa. El Motor de base de datos también proporciona una completa compatibilidad para mantener una gran disponibilidad.

6.7.2 R services

Microsoft R Services proporciona varias maneras de incorporar el popular lenguaje R en los flujos de trabajo de la empresa.

R Services (en bases de datos) integra el lenguaje R con SQL Server, facilitando así la compilación, el reciclaje y los modelos de puntuación llamando a los procedimientos almacenados de Transact-SQL.

Microsoft R Server proporciona compatibilidad multiplataforma y escalable para R en la empresa y admite orígenes de datos como Hadoop y Teradata.

6.7.3 Data Quality Services

SQL Server Data Quality Services (DQS) proporciona una solución de limpieza de datos controlada por conocimiento. DQS permite generar una base de conocimiento y usarla para realizar tareas de corrección de datos y eliminación de datos duplicados, usando medios asistidos por ordenador e interactivos. Puede usar servicios de consulta de datos basados en la nube y puede generar una solución de administración de datos que integra DQS con SQL Server Integration Services y Master Data Services.

6.7.4 Analysis Service

Analysis Services es una plataforma y un conjunto de herramientas de datos analíticos para Business Intelligence en un entorno personal, de equipo o empresa. Los servidores y los diseñadores de cliente admiten soluciones OLAP tradicionales, nuevas soluciones de modelado tabular y análisis y colaboración de autoservicio mediante Power Pivot, Excel y un entorno de SharePoint Server.

Analysis Services también incluye minería de datos para permitir descubrir las relaciones y los patrones ocultos en grandes volúmenes de datos.

6.7.5 Replicación

Analysis Services es una plataforma y un conjunto de herramientas de datos analíticos para Business Intelligence en un entorno personal, de equipo o empresa. Los servidores y los diseñadores de cliente admiten soluciones OLAP tradicionales, nuevas soluciones de modelado tabular y análisis y colaboración de autoservicio mediante Power Pivot, Excel y un entorno de SharePoint Server. Analysis Services también incluye minería de datos para permitir descubrir las relaciones y los patrones ocultos en grandes volúmenes de datos. (MSDN, n.d.)

6.8 Comunicación serie

La comunicación serie permite la transmisión de información por medio de dos líneas de operación denominadas RX y TX, existen varios puertos, pero los más conocidos son el RS-232 o el RS-485, además existe una cantidad de puertos serie como el I2C, SPI, Serial Ata, etc; hay que recordar que se denomina puerto serie a una interfaz física o virtual que genere una comunicación entre dispositivos.

Ciertos procesadores contienen una unidad de comunicación llamada UART que por sus siglas significa Universally asynchronous receiver/ transmitter que se encarga de convertir los datos a una secuencia de bits y enviarlos o recibirlos a una velocidad determinada.
Todas las placas de Arduino disponen de al menos un puerto serie que están unidos físicamente a distintos pines , todos los modelos de placas Arduino disponen de un conector USB o micro USB, conectados al puerto serie facilitando la conexión a un ordenador.

Para realizar la conexión mediante puerto serie se necesita establecer la conexión física y la conexión virtual, para establecer una conexión serial se necesita una estructura básicade programación y definir el puerto serie que se va a usar dentro de la ventana de herramientas del compilador de Arduino:

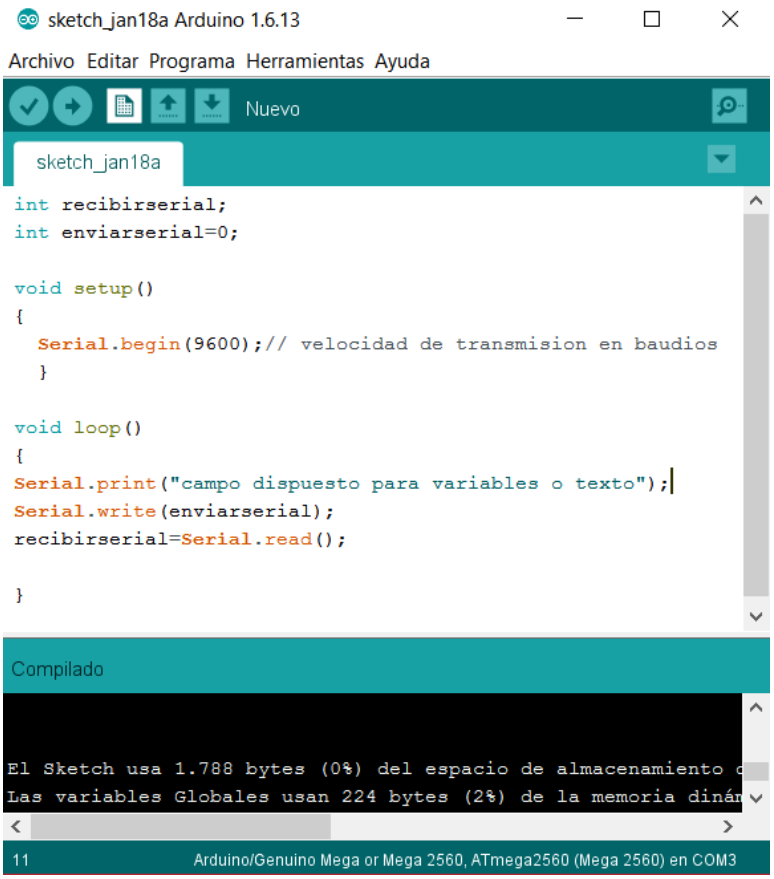

Figura 7. Comandos para la configuración virtual de un puerto serial
Fuente: Autores.

6.8.1 Norma RS 232

La norma define un miembro transmisor denominado DTE(Data terminal Equipment) y un dispositivo receptor denominado DCE (Data Communications Equipment) que no pueden estar ubicados a distancias mayores de 15 metros con una velocidad máxima de transmison de 20000 bps, la principal limitación que tiene este sistema con la tecnología actual es que no es compatible con sistemas TTL debido a que traba con rangos de -3V , -15V , 3V y 15V, usa dos líneas de transmisión totalmente independientes los dos cables de alimentación.

Figura 8. Terminales de un cable RS 232
Fuente: Autores.

Para hacer conexión con un sistema TTL se deben hacer conversiones de voltaje por lo que se usan circuitos que empleen el integrado MAX 232 , dentro del mercado existen cables conversores que ya traen implícita la función de convertir el voltaje.

Figura 9. Cable conversor de RS 232 a TTL
Fuente: Autores.

6.9 Sensor HC-SR04

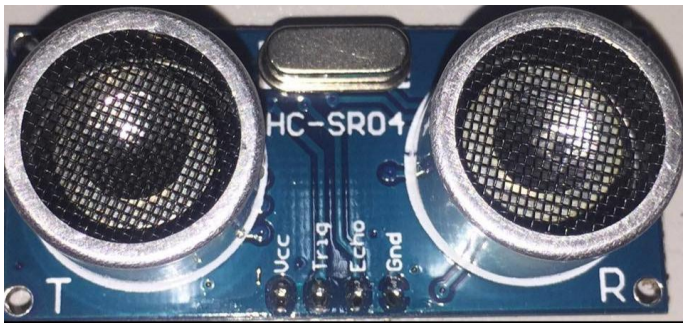

Figura 10. Sensor de ultrasonido HC-SR04
Fuente: Autores.

- Voltaje de trabajo: 5V DC
- Corriente de trabajo: 15mA

- Frecuencia de trabajo: 40Hz
- Rango máximo de medición: 4m
- Rango mínimo de medición: 2cm
- Angulo de medición: 15°
- Señal de disparo de entrada: 10us pulso TTL
- Dimensiones: 45*20*15mm

7. **DISEÑO METODOLÓGICO**

Se presenta un proceso de investigación aplicada, puesto que se plantea una situación en un entorno social el cual debe ser solucionado mediante un diseño de un sistema que es desarrollado en distintas etapas y que es capaz de dar solución a la aplicación deseada, se pretende que el proyecto sea realizado con modalidad documental-exploratorio, debido a que su desarrollo está basado en otras investigaciones, modelos técnicos y tecnológicos que ya están planteados. Se solucionó cada necesidad según la situación y un plan previamente establecido.

Recolección de información

Dentro de esta etapa del proceso se averiguo la forma de trabajo del módulo lector RFID PT-3LA01Z, además se realizaron consultas técnicas acerca de la construcción de una trama de datos para poder obtener información de los tags; luego se averiguo cómo escribir y asignar un ID a cada usuario con el fin de tener más claridad sobre el diseño.

Se tuvo en cuenta el número de usuarios que podrán acceder al servicio y la forma de acceso, además de recolectar toda la información necesaria por parte del departamento de planta física.

Adicionalmente se procede a investigar y elegir un programa en el cual se pueda desarrollar bases de datos, y a indagar sobre tipos de sensores de barrera que existen en el mercado.

Diseño de software compatible con la lectora

Se realiza un software que sea capaz de adquirir la información que genera el modulo lector y construir un ID con una trama de datos para cada usuario, fue importante tener en cuenta en esta etapa del proyecto que tipo de comunicación se usó entre estas dos partes.

Diseño de una base de datos.

Se escoge un software adecuado para diseñar una base de datos que permita un registro detallado de cada usuario y que genere un historial completo con información relevante como fecha, hora y tiempo dentro de las instalaciones.

Diseño de una interfaz usuario

En esta etapa del proceso se realiza una interfaz de usuario para que se puedan generar reportes según se requiera, es de fácil acceso para que el operario pueda ingresar a todo tipo de información de cada usuario según su interés de manera inmediata.

Medición de variables eléctricas

Esta sección dentro del desarrollo metodológico surge con la necesidad de mover una talanquera que tiene por consumo de energía la cantidad de 300W

con un voltaje de alimentación de 120 VAC, dentro de las especificaciones del fabricante se encontró que tiene un voltaje de control de 24 VDC por tanto el prototipo se diseñó para ser capaz de ofrecer una corriente mayor a 2.5 A; por lo tanto fue necesario realizar esta etapa de medición para verificar que se cumplan las condiciones requeridas.

Solución de preguntas

En este momento y según el prototipo obtenido se debe someter a juicio el resultado final, se deben hacer pruebas de funcionamiento y se debe dar solución a las preguntas que se surgieron cuando se planteó problema.

El diseño metodológico se resume en el siguiente diagrama de bloques:

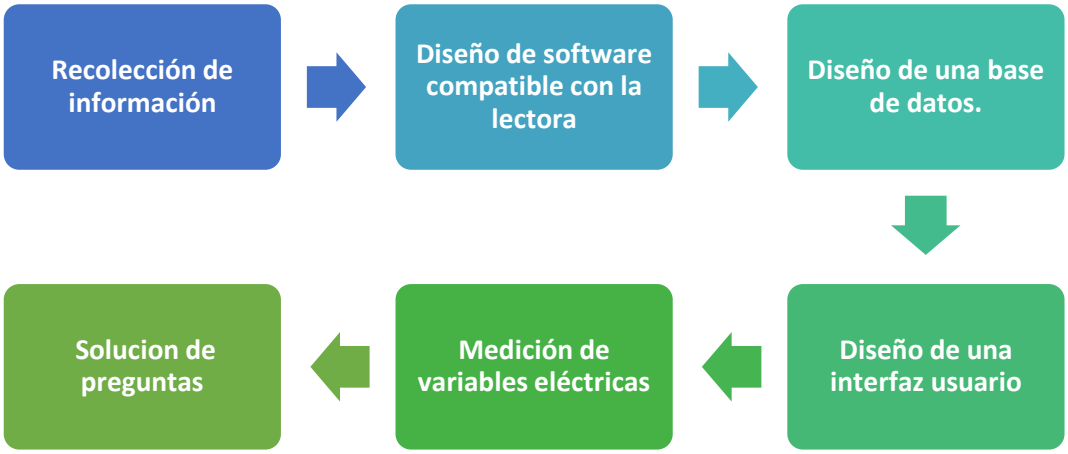

Figura 11. Diagrama del Diseño metodológico del proyecto
Fuente: Autores.

8. DESARROLLO DEL PROTOTIPO Y RESULTADOS

A partir de este apartado del documento se describe cada procedimiento de manera detalla que se usó para el desarrollo del prototipo para control de acceso al parqueadero del campus de la Universidad Santo Tomás seccional Tunja.

Se describe el desarrollo del Software de lectura y la forma de programación dentro del programa Visual Studio; se comprende el protocolo de comunicación que el fabricante nos facilita para ejecutar la aplicación.

Se evidencia el desarrollo de dos bases de datos en SQL server que vinculan con la programación en C# además muestra el manejo de la información de forma interna y la asignación de históricos para cada usuario.

Se realizó una aplicación de Windows forms para que pueda ser ejecutada desde el escritorio y cuenta con una interfaz de usuario sencilla y detallada expresando todo el potencial de la aplicación.

8.1 DESARROLLO DE SOFTWARE DE LECTURA RFID

Para dar inicio al desarrollo del proyecto se decide hacer uso del software Visual Studio porque es una plataforma multiservicios que se adecua a las necesidades requeridas por el proyecto; se necesitaba buscar una herramienta compatible con el modulo lector de RFID y dentro de las especificaciones técnicas del fabricante, se encuentra que facilita un complemento de programación dispuesto en el leguaje C#. Visual Studio integra el lenguaje de programación C Sharp, posee una sintaxis muy expresiva y practica para el desarrollo de aplicaciones dentro de la plataforma Windows, sus uso se manifiesta como muchos otros lenguajes de programación y usa signos de llave que fácilmente pueden ser reconocidos inmediatamente por cualquier persona familiarizada con C, C++ o Java.

C# es un lenguaje de programación orientado a objetos admite todo tipo de variables y métodos, dentro del programa base se puede elegir qué tipo de aplicación se desea ejecutar, ofreciendo distintas opciones, entre las cuales se elige el desarrollo de una aplicación de Windows forms, lo que permite que se desarrolle en conjunto la lectura y la aplicación.

Un elemento importante dentro del programa base para desarrollar es el cuadro de herramientas, debido a que dentro de este se contiene todos los elementos necesarios para empezar a trabajar sobre la aplicación requerida, se procede a acoplarlo al panel de trabajo para hacer más eficiente el uso de los recursos disponibles en esta ventana de herramientas.

Dentro del desarrollo es importante la interfaz gráfica que C# ofrece de Windows forms, debido a que proporciona distintas clases de botones y herramientas de visualización que están asociados a controles que generan eventos para determinar como el programa debe actuar frente a determinada acción. Cuando se genera un botón o un elemento de la interfaz, en la ventana de programación se crea un nuevo evento y este interactúa dependiendo de su controlador.

8.1.1 Usar archivos DLL

Un Archivo DLL es una biblioteca de vínculos dinámicos que Windows usa para ejecutar acciones con recursos incorporados del sistema, ayudando a mejorar la eficiencia en el uso de las capacidades del equipo, los programas se cargan más rápido y en este caso trae funciones incorporadas y facilitan el uso de herramientas y desarrollo de software

El fabricante facilita soportes técnicos entre los cuales se encuentra una biblioteca de vínculos dinámicos que contiene los métodos necesarios para desarrollar la aplicación y manejar el protocolo de comunicación de la lectora; de tal manera que se procede a agregar la biblioteca para empezar a hacer uso de comandos especiales para la aplicación y el manejo de la trama de datos.

Para agregar la biblioteca, dentro de Visual Studio , se abre el explorador de soluciones en donde se observan los diferentes archivos y herramientas que componen el proyecto dentro se encuentra una pestaña de referencias, sobre ella, se da click derecho y se selecciona la opción añadir referencia.

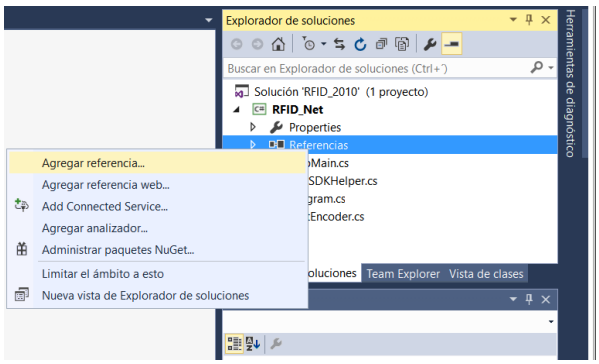

Figura 12. Agregar una referencia para una biblioteca de vínculos dinámicos
Fuente: Autores.

Se busca dentro de las opciones explorar y se agrega el archivo con extensión .DLL

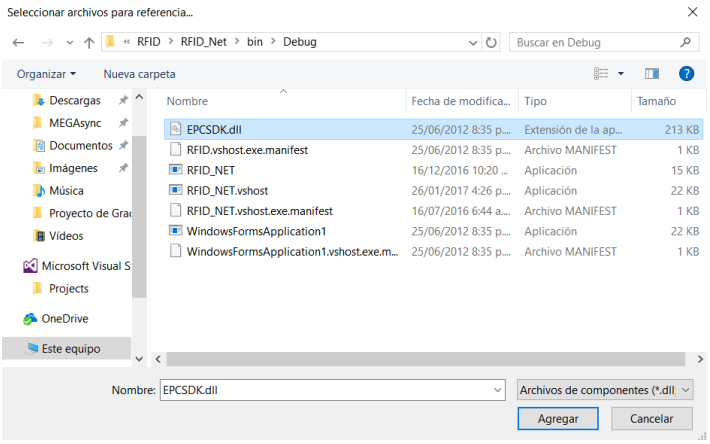

Figura 13. Archivo DLL agregado a el proyecto C#
Fuente: Autores.

Con la librería cargada y lista para usarse, se procede a revisar el documento para empezar a usar los distintos comandos e iniciar con el desarrollo del software.

8.1.2 Comandos para el uso de el modulo lector PT-3LA01Z.

En esta sección se presentan los comandos que se usaron dentro de la programación para usar el modulo que se explican en la guía de desarrollo SDK que el fabricante proporciona.

8.1.2.1 Abrir el puerto

Para abrir el puerto se usa el comando:

HANDLE OpenCom (int portNo);

Para usar esta función se usó la siguiente rutina:

HANDLE hCom= OpenComm (1) // se abre el puerto serial 1
If (hCom!= INVALID_HANDLE_VALUE)
AfxMessageBox (“serial abierto”);
else
AfxMessageBox (“fallo la conexión”)

Se define el puerto serie al cual se va a conectar el dispositivo lector, luego se hace la pregunta de que si se hizo la conexión, si es verdad, muestra en un mensaje que el puerto se abrió,

si no detecta ningún puerto, imprime el mensaje de conexión fallida.

8.1.2.2 Cerrar la comunicación

Para cerrar la comunicación su usa el comando:

voidCloseComm(HANDLE hCom, BYTE ReaderAddr);

Para usar la función se usa la siguiente rutina:

CloseComm (hCom,0);

Dentro de la rutina el único parámetro que se define es el puerto que se desea cerrar.

8.1.2.3 LEER UN TAG

Para leer un tag se tiene que tener en cuenta la función ReadTag que contiene la siguiente estructura:

*BOOL ReadTag (HANDLE hCom, BYTE memBank, BYTE address, BYTE length, BYTE*data, BYTE ReaderAddr);*

Cuando la función se completa retorna un estado de TRUE, por el contrario cuando ha fallado, retorna a un estado de False.

Los valores que se deben contener dentro de esta función son:

0x00: sección de la trama de datos reservada
0x01: Área para escribir la cabecera de la trama EPC
0x02: Área para escribir la cabecera de la trama para TID
0x03: Área de usuario.

Address: El área de dirección puede contener un rango de 0 a 7.

Length: Es el tamaño de la trama de datos que está dispuesto a leer, se tiene un rango de 1 a 8.

Data: Es la dirección para ser escrita.

ReaderAddr: Lee la dirección, por defecto es 0.

Para ejecutar la función se usa la siguiente rutina:

```
if (:: ReadTag (hCom, 0x01, 0x03, 1, data,0)) / /
 {
 CString strData;
 strData.Format ("% 02X% 02X", data [0], data [1]);
 AfxMessageBox ("lectura del tag completa!" + StrData);
 }
 else
 AfxMessageBox ("lectura de tgs fallida!");
```

cuando la lectura de las cabeceras de datos se completan y se realiza de manera completa , imprime la lectura de los caracteres, de lo contrario imprime la lectura de los tags ha fallado.

Para ejecutar la lectura del interrogador se usaron las rutinas previamente vistas, dentro de una estructura de Windows forms programando en C# y se obtuvo la siguiente interfaz básica del proyecto; que cumple con la necesidad de recibir los datos por parte de el modulo lector (figura 14).

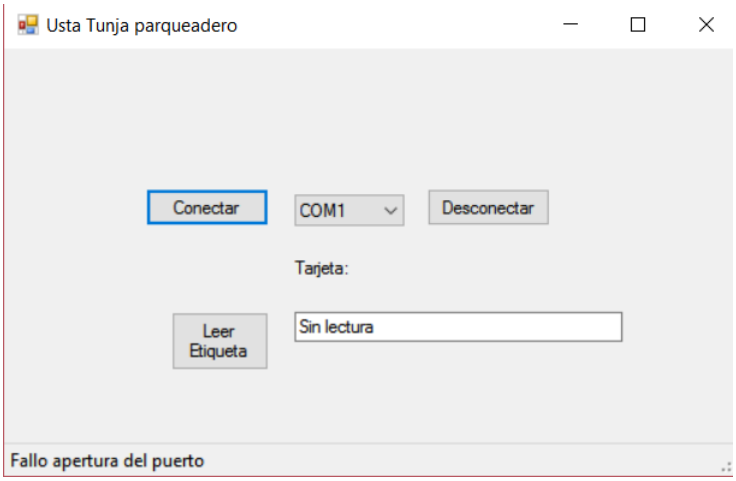

Figura 14. Interfaz del programa básico para leer los tags.
Fuente: Autores.

8.2 BASE DE DATOS

La base de datos debe contener la información que se adquiere por parte del interrogador de RFID, definida la información, la base de datos principal y según se muestra en este desarrollo contiene nombre, apellido, facultad y características también como la opción de guardar, modificar y borrar.

Se realizó el desarrollo de la interface relacionando cada parámetro dentro de la aplicación con el código de programación siguiendo el orden que se especifica a continuación:

8.2.1 Ingresar información del usuario:

Se generan las cajas de texto (textbox) y se les asigna una variable para que cada vez que se escriba un dato, inmediatamente quedara guardado dentro de ella respectivamente y se registra en la tabla de la base de datos; para desplegar este tipo de formas dentro de la interface se acude al cuadro de herramientas y se agregan al espacio de trabajo de la interface.

Figura 15. Espacios dispuestos para guardar cada usuario dentro de la bases de datos.
Fuente: Autores

8.2.2 Botón guardar :

Es el encargado de agregar a la tabla de la base de datos los valores anteriormente ingresados en las cajas de texto.

Figura 16. Botón guardar de la interfaz de usuario.
Fuente: Autores.

Se genera la función de respuesta que se ejecuta al oprimir guardar, donde se accede primero a la conexión con la base de datos creada, posteriormente se arma la consulta para la tabla con el nombre asignado con el orden correspondiente para cada columna dependiendo de las creadas en SQL Server, donde cada valor encontrado en las cajas de texto se ubica en la columna indicada, esto se logra con el uso de la instrucción ***insert into*** "***Nombre de la tabla de datos"*** que inserta la información a la tabla con los valores de ingreso manipulados como se muestra en el comentario 2 dentro

de la figura 19, luego se ejecuta la acción anterior y se accede a la información que se muestra en la interface en la cuadricula de datos (***DataGridView***) contenida en el comentario 4 de la figura 19; con la instrucción ***select*from "Nombre de la tabla de datos"*** que se muestra en el numeral 4 de la figura 19**,** si aparece un mensaje en donde se afirma que la información se agregó correctamente y se procede a la desconexión con la base de datos debido a que ya se ingresó la información y se liberan los espacios en las cajas de texto para un recibir el registro de un nuevo usuario.

```
//1. CONECTA CON LA BASE DE DATOS
ManipularDatos.Conectar();

//2. ARMAR A CONSULTA (Query)
String Query = "insert into datosrfid (CODIGO,NOMBRES,APELLIDOS,EDAD) values ('"
+ txtEtiqueta.Text + "','" + txtNombres.Text + "','" + txtApellidos.Text + "'," + txtFacultad.Text + ");";

//3. EJECUTA LA CONSULTA
ManipularDatos.EjecutarSql(Query);

//4. ACTUALIZAR LA CUADRICULA DE DATOS (DataGridView)
ManipularDatos.ActualizarGrid(this.dataGridView1, "select * from datosrfid;");

//5. DESCONECTA LA BASE DE DATOS
ManipularDatos.Desconectar();

//6. LIMPIA LAS CAJAS DE TEXTO
txtNombres.Text = "";
txtApellidos.Text = "";
txtFacultad.Text = "";
txtEtiqueta.Text = "";
```

Figura 17. Código para el ingreso de usuarios a la base de datos.
Fuente: Autores

8.2.3 Botón actualizar:

Permite renovar la información de los usuarios que están guardados en la base de datos, para lo cual se agrega el botón Actualizar y se crea la función que contendrá las instrucciones que realizan dicha acción, contiene el siguiente orden de programación:

Figura 18. Botón de actualizar de la interfaz de usuario.
Fuente: Autores.

Se debe seleccionar el usuario al que se le va a actualizar la información y dar Click al botón Actualizar el cual accede a la base de datos para tomar la información del usuario e imprimirlo o mostrarlo en la caja de texto asignado para cada valor dependiendo de la celda o columna designada a cada uno de estos, como se observa en la figura 18 cada caja de texto toma el dato escrito en las celdas y modifica la variable o tipo de dato con el de la selección actual.

```
private void btnActualizar_Click(object sender, EventArgs e)
{
    // CAPTURA LOS DATOS DEL LA CUADRICULA DE DATOS (DataGridView)...
    // Y LOS ASIGNA A CADA CAJA DE TEXTO
    Codigo = this.dataGridView1.CurrentRow.Cells[0].Value.ToString();
    txtNombres.Text = this.dataGridView1.CurrentRow.Cells[1].Value.ToString();
    txtApellidos.Text = this.dataGridView1.CurrentRow.Cells[2].Value.ToString();
    txtFacultad.Text = this.dataGridView1.CurrentRow.Cells[3].Value.ToString();
}
```

Figura 19. Código para captura de datos y asignación a cada caja de texto
Fuente: Autores.

Al momento de completar la actualización se oprime el botón guardar al cual cuenta con otra secuencia que contiene la instrucción de Actualizar, primero se habilita la conexión con la base de datos, posteriormente se arma la consulta donde cada valor de las cajas de texto se ubica actualizando dicha información seleccionada, esto se logró usando la instrucción ***update "Nombre de la tabla de la base de datos"*** contenido en el numeral 2 en la figura 20 seguido de la palabra set tomando los valores a actualizar, se ejecuta luego la consulta anterior tomando los valores actuales y se seleccionan para mostrarlos en la cuadricula de datos (***DataGridView***) con la instrucción ***select * from "Nombre de la tabla de la base de datos"*** que se muestra en comentario 4 de la figura 20, finalmente se desconecta con la base de datos y se ejecuta la acción de limpiar las cajas de texto.

```
//1. CONECTA CON LA BASE DE DATOS
ManipularDatos.Conectar();

//2. ARMAR A CONSULTA (Query)
String Query = "update datosrfid set NOMBRES = '" + txtNombres.Text + "', APELLIDOS = '"
    + txtApellidos.Text + "', EDAD =" + txtFacultad.Text + " where CODIGO = '" + Codigo + "';";

//3. EJECUTA LA CONSULTA
ManipularDatos.EjecutarSql(Query);

//4. ACTUALIZAR LA CUADRICULA DE DATOS (DataGridView)
ManipularDatos.ActualizarGrid(this.dataGridView1, "select * from datosrfid;");

//5. DESCONECTA LA BASE DE DATOS
ManipularDatos.Desconectar();

//6. LIMPIA LAS CAJAS DE TEXTO
txtNombres.Text = "";
txtApellidos.Text = "";
txtFacultad.Text = "";
txtEtiqueta.Text = "";
```

Figura 20. Código para el ingreso de usuarios a la base de datos.
Fuente: Autores.

8.2.4 Botón borrar:

Permite eliminar información de los usuarios de la tabla de la base de datos, borrando todos los datos que se han guardado en la base de datos.

Figura 21. Botón de eliminar información en la interfaz de usuario.
Fuente: Autores

Se crea entonces la función del botón y se agrega la secuencia de programación que contendrá las instrucciones de respuesta del botón, para esto primero se selecciona manualmente el usuario a eliminar el cual captura principalmente la columna que contiene el código de etiqueta, luego mediante un mensaje de aviso pregunta en la interface si desea eliminar esta información usando en programación el método ***MessageBox.Show*** que se muestra en el comentario 2 de la figura 22 y se evalúa la respuesta entregada por el usuario mediante la condición if, si es falsa el programa retorna y se sale de la función de borrado, si por el contrario la respuesta es verdadera se conecta con la base de datos creada posteriormente se arma la consulta que hará la tarea de borrado de la información mediante el uso de la instrucción ***delete from*** contenida en la figura 22 en el numeral 3.2, luego se actualiza la información en la base de datos y en la cuadricula de datos que ya no contendrá la información anteriormente eliminada usando la instrucción ***select * from*** que se muestra en el número 3.4 de la siguiente figura, finalmente se desconecta con la base de datos.

```
//1. CAPTURA E CODGIO DE LA TABLA datosrfid
Codigo = this.dataGridView1.CurrentRow.Cells[0].Value.ToString();

//2. PREGUNTA AL USUARIO SI REALMENTE DESEA ELIMINAR EL REGISTRO
var resultado = MessageBox.Show("¿Desear realmente eliminar este registro?", "Confirme el borrado",
    MessageBoxButtons.YesNo, MessageBoxIcon.Question);

//3. PROCEDE A EVALUAR EL RESULTADO DE LA RESPUESTA
if (resultado == DialogResult.Yes)  // SI LA RESPUESTA ES VERDADERA PROCEDE A BORRAR EL REGISTRO
{
    //1. CONECTA CON LA BASE DE DATOS
    ManipularDatos.Conectar();

    //2. ARMAR A CONSULTA (Query)
    String Query = "delete from datosrfid where CODIGO = '" + Codigo + "';";
    //3. EJECUTA LA CONSULTA
    ManipularDatos.EjecutarSql(Query);

    //4. ACTUALIZAR LA CUADRICULA DE DATOS (DataGridView)
    ManipularDatos.ActualizarGrid(this.dataGridView1, "select * from datosrfid;");

    //5. DESCONECTA LA BASE DE DATOS
    ManipularDatos.Desconectar();
```

Figura 22. Código para borrar información de la base de datos.
Fuente: Autores.

8.2.5 Desarrollo tabla de base de datos en SQL Server

Primero al abrir el software Microsoft SQL Server se conecta con el servidor, dando Click en Connect. Es importante observar el nombre del servidor debido a que dentro del comando C# que hace el llamado a la base de datos se debe colocar tal como aparece escrito.

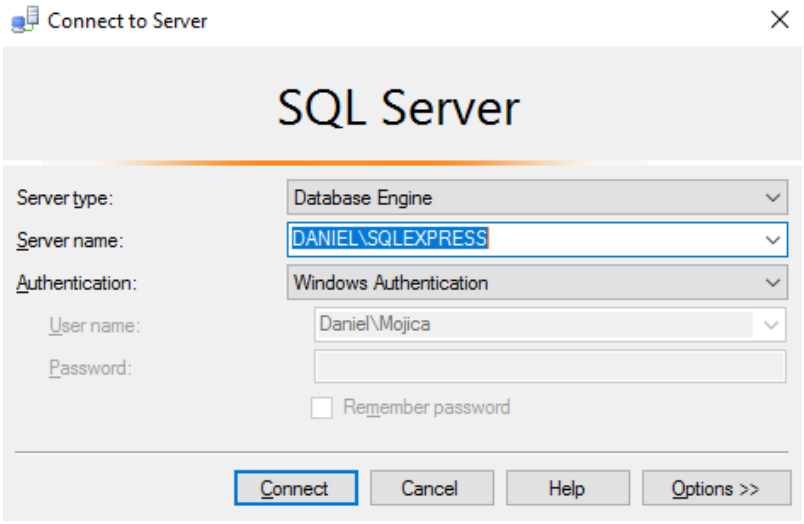

Figura 23. Cuadro de conexión de servidor de SQL
Fuente: Autores.

En la ventana de inicio del programa en el explorador de objetos se crea la base de datos, dando Click derecho en Databases y se selecciona New Database, como se muestra en la figura 24.

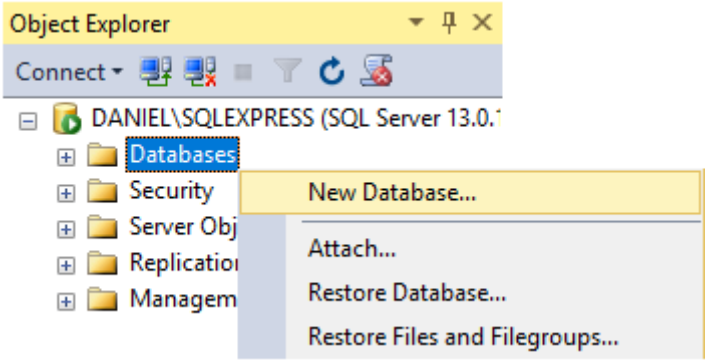

Figura 24. Crear una nueva base de datos en SQL server.
Fuente: Autores.

Luego se da nombre a la base de datos (figura 25)

Figura 25. Nombrar la base de datos.
Fuente: Autores.

Dentro de la base de datos, SQL contiene carpetas con distintas funciones y agrupaciones, en la base se necesita gestionar la creación de una tabla en donde se clasifiquen y se guarden los datos de cada usuario, por lo que se crea una tabla (figura 26).

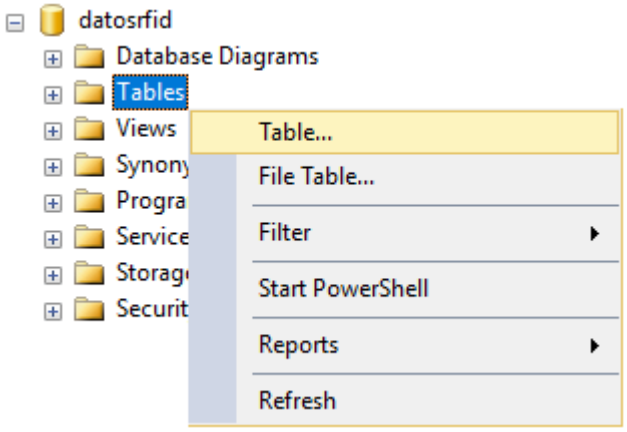

Figura 26. Crear una tabla dentro de la base de datos.
Fuente: Autores.

La tabla y la base de datos ya están exitosamente cargadas, de tal manera que se procede a generar las columnas con los requisitos para cada usuario, se nombra cada columna y se define que el tipo de datos que va a contener sea de tipo Char con un tamaño de 50 caracteres como máximo sin aceptar valores nulos (figura 27).

Column Name	Data Type	Allow Nulls
CODIGO	varchar(50)	☐
NOMBRES	varchar(50)	☑
APELLIDOS	varchar(50)	☑
FACULTAD	varchar(50)	☑

Figura 27. Asignación de columnas para una base datos
Fuente: Autores.

8.2.6 Conexión de C# con SQL Server

Conectar la aplicación del sistema realizado con la base de datos en SQL Server da la posibilidad para guardar los datos de los usuarios, en donde primero se crea la conexión, luego ejecuta los diferentes tipos de comandos para finalmente observar la interacción de los datos con las cuadricula de datos (***DataGridView***) que se muestra en la interface del sistema.

Para crear la conexión entre la aplicación y el servidor de base de datos que se encarga de gestionar los datos de SQL hacia la aplicación, se crea una nueva clase en C Sharp (Una clase es una construcción que le permite crear sus propios tipos personalizados

agrupando variables de otros tipos, métodos y eventos), que permite que se minimice código y mediante un llamado desde el programa principal se haga uso de las instrucciones que contiene dicha clase. Se agrega entonces a la clase dos directivas ***using*** que permiten utilizar las funciones de interface y manipular variables de conexión de datos de SQL como se muestra en la figura 28.

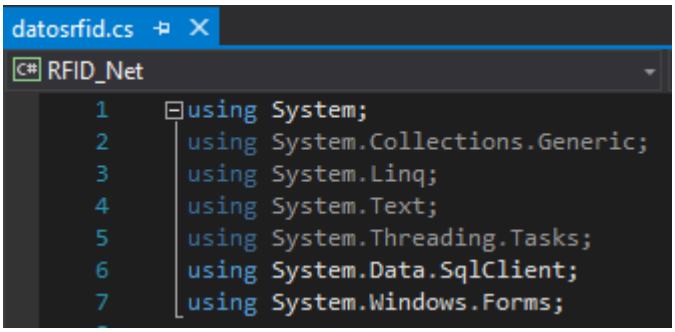

Figura 28. Agregar el método para el uso de SQL.
Fuente: Autores.

Se crean variables de conexión con SQL y funciones con instrucciones para conexión, desconexión, ejecución SQL y actualización de cuadricula de datos (***DataGridView***), las cuales serán llamadas desde el programa principal.

Es importante abrir la conexión hacia el servidor con el método ***Open()***; y agregarle a la variable ***SqlConnection*** el nombre de la cadena de conexión creada con SQL, como se observa en la figura 29.

```
SqlConnection MiConexion;
4 referencias
public void Conectar()
{
    //la cadena de conexión va a cambiar en sus computadoras...
    MiConexion = new SqlConnection("Data Source=DANIEL\\SQLEXPRESS;Initial " +
    "Catalog=datosrfid;Integrated Security=True");
    MiConexion.Open();
}
```

Figura 29. Código para abrir la comunicación con SQL.
Fuente: Autores.

Al finalizar las operaciones sobre la base de datos es importante cerrar la conexión usando el método ***Close(),*** usado como se ve en la figura 30.

```
public void Desconectar()
{
    MiConexion.Close();
}
```

Figura 30. Método para cerrar la comunicación con SQL.
Fuente: Autores.

Creando las instrucciones elementales para establecer una conexión, se procede a definir la conexión con el servidor, en donde se busca el origen de datos que habilita finalmente la conexión en ambas partes del sistema, para esto se da click en proyecto y en agregar nuevo origen de datos como se observa en la figura 31.

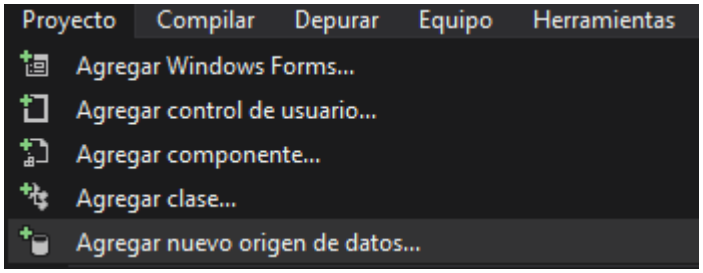

Figura 31. Establecer un nuevo origen de datos de SQL.
Fuente: Autores.

Se despliega una ventana de selección de tipo de origen de datos, se elige Base de datos, que permite conectarse a una base de datos y elegir los objetos de base para aplicación y poder hacer uso de la tabla de SQL previamente creada.

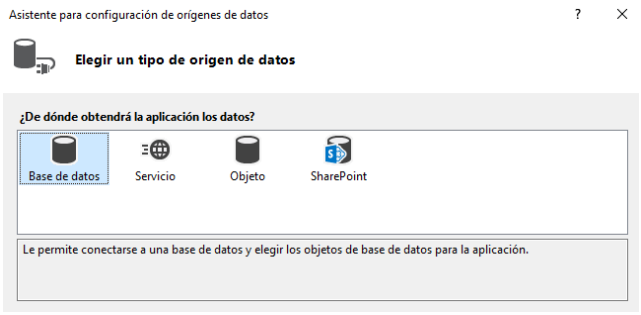

Figura 32. Establecer un nuevo origen de datos de SQL.
Fuente: Autores.

Se debe seleccionar ahora como modelo de base de datos; conjunto de datos, este basa su funcionamiento en la utilización de la aplicación para agregar un archivo de conjunto de datos al proyecto, seguidamente es decide la conexión de datos, se da click en nueva conexión, como se muestra en la figura 33.

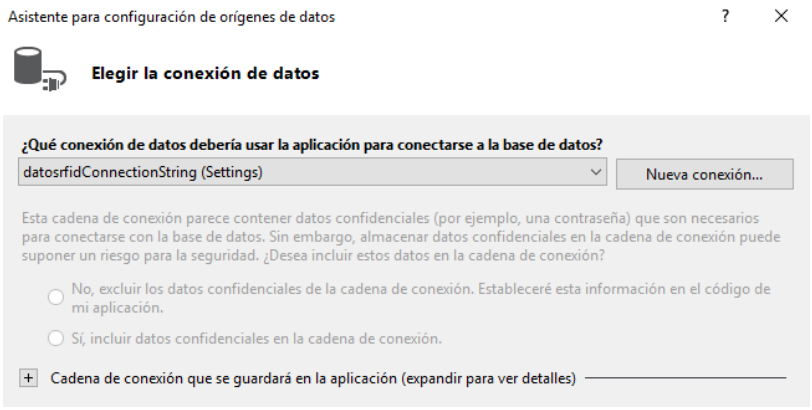

Figura 33. Elegir la conexión de datos.
Fuente: Autores.

Se abre una nueva ventana donde se agrega origen de los datos y nombre de servidor y finalmente se busca la base de datos creada en SQL como se ve en la figura 34.

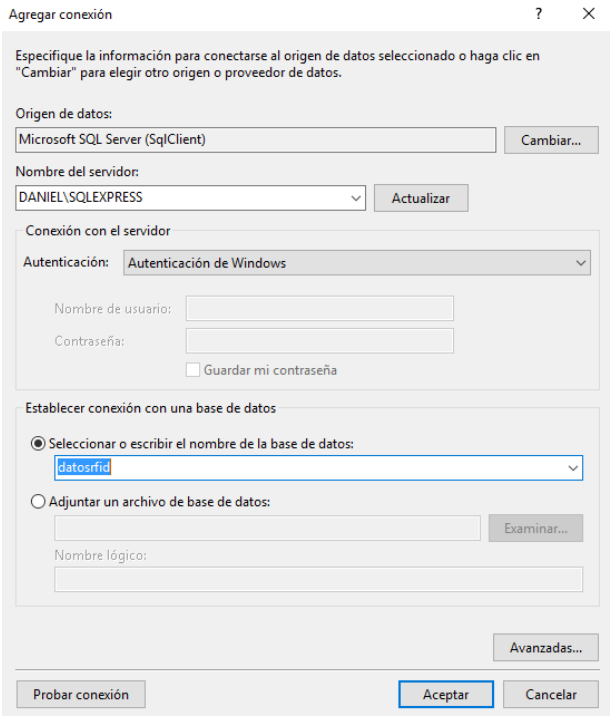

Figura 34. Agregar una nueva conexión.
Fuente: Autores.

Por último se eligen los objetos de la base de datos que se desean tener el conjunto de datos, seleccionando la tabla con el nombre especificado en SQL Server (figura 35).

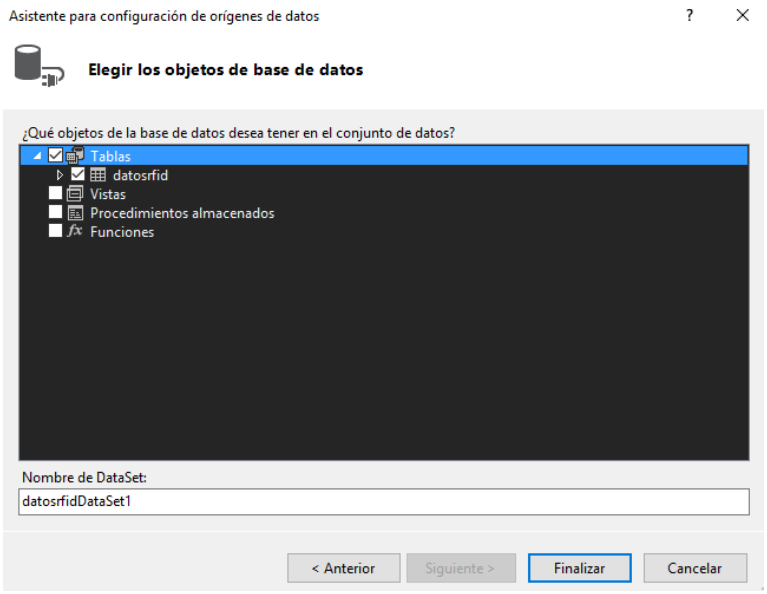

Figura 35. Configuración de orígenes de datos.
Fuente: Autores.

8.3 BASE DE DATOS PARA REGISTRO DE HISTÓRICO DE ENTRADAS Y SALIDAS DEL PARQUEADERO.

En esta base de datos se guarda información sobre reporte de entradas y salidas de cada usuario, estableciendo como parámetros o condiciones la lectura de cada uno de los códigos asignados anteriormente a los usuarios, teniendo como finalidad mostrar o llevar seguimiento a las personas que hacen uso del parqueadero.

8.3.1 Desarrollo de interface y programación en C#

En este caso primero se define la información que contiene la base de datos como código de lectura de etiquetas, hora entrada, hora salida y fecha de ingreso y se definen características de interface vinculado a la programación de cada uno de los botones y cuadriculas de datos de los que usaron en esta parte del desarrollo del sistema, para lo cual se siguieron los pasos nombrados a continuación:

8.3.1.1 INFORMACIÓN A INGRESAR A LA BASE DE DATOS

El primer paso es generar una estructura con el uso de la instrucción ***private DateTime*** que representa un instante en el tiempo, normalmente expresado como una fecha (DD/MM/AA) y hora (HH:MM:SS) del día, almacenada dicha información en una variable de tipo ***DateTime***.

```
private DateTime datetime;
private DateTime datetime1;
```

Figura 36. Configuración de registro de tiempo
Fuente: Autores.

Luego se crean botones para tomar los datos de entrada y salida los cuales harán uso de las anteriores variables y guardar el reporte en la base de datos que es mostrada finalmente en una cuadricula de datos (***DateGridView***), a continuación de explica cada uno de los botones que se usaron.

8.3.1.2 Botón Entrada

Se agregar el botón a la ventana de interface y se genera la función de respuesta al oprimir el botón.

Figura 37. Botón de entrada.
Fuente: Autores.

El cual ejecuta las siguientes instrucciones:

Se declaran las variables para la comparación de la condición If en donde se guardara el código ingresado y los códigos existentes en la base de datos general.

```
byte[] data = new byte[12];                              // PROGRAMACION
if (EPCSDKHelper.ReadTag(_handle, 0x01, 0x02, 6, data, 0))  // PARA LECTURA
{                                                        // DE LA ETIQUETA
    byte[] bTmp = new byte[12];                          // RFID
    Buffer.BlockCopy(data, 0, bTmp, 0, 12);              //
    this.txtBuscar.Text = TextEncoder.ByteArrayToHexString(bTmp);

    // VARIABLES PARA LA COMPARACION EN LA CONDICION IF
    string CodigosBaseDatos = this.dataGridView1.CurrentRow.Cells[0].Value.ToString();
    string LecturaCodigo = TextEncoder.ByteArrayToHexString(bTmp);   //hasta aca
```

Figura 38. Declaración de variables para comparación de existencia en la base de datos
Fuente: Autores.

Primero se hace lectura de la etiqueta al tener la información con la condición if se compara la lectura actual con la información guardada en la base de datos principal donde se encuentra el registro de los usuarios, si la comparación es falsa el programa retorna al inicio, si por el contrario es verdadera primero toma los datos de hora y fecha actuales que luego ingresan a la base de

datos, luego se activa la conexión con la base de datos, posteriormente se insertan los datos a la tabla usando la instrucción ***insert into "Nombre de la tabla de la base de datos ",*** se ejecuta la consulta para que acceda e inserte los valores anteriores y se toma esta información para que se muestre en la interface dentro de la cuadricula de datos (***DataGridView***) mediante el uso de la instrucción ***select * from "Nombre de la tabla de la base de datos"***, que se observa en la figura 39 se desconecta por ultimo con la base de datos y envía un dato por el puerto serial al arduino para completar la función del elevado para entrada de la talanquera.

```
if (LecturaCodigo == CodigosBaseDatos) // CONDICION PARA GUARDAR INFORMACION DE ENTRADA
{
    string Parq = Convert.ToString(Parqueaderos);   // PROGRAMACION
    txtParqueaderos.Text = Parq;                    // PARA CONTEO
    Parqueaderos = Parqueaderos + 1;                // DE PARQUEADEROS
    conteo = conteo + 1;
    ComArduino.Write("1");                          // ESCRIBE EN EL PUERTO SERIAL PARA ARDUINO
    // PROGRAMACION PARA DATOS DE HORA Y FECHA DE INGRESO
    datetime = DateTime.Now;
    txtEntrada.Text = datetime.Hour + " : " + datetime.Minute + " : " + datetime.Second;
    txtFecha.Text = datetime.Day + " / " + datetime.Month + " / " + datetime.Year;

   //1. CONECTA LA BASE DE DATOS
   ManipularDatos1.Conectar1();

   //2. ARMA CONSULTA (Query1)
   String Query1 = "insert into entradasalida (CODIGO,ENTRADA,SALIDA,FECHA,NUMERO) values ('" + txtBuscar.Text
       + "','" + txtEntrada.Text + "','"  +"ENTRADA" + "','" + txtFecha.Text + "'," + conteo + ");";

   //3.EJECUTA LA CONSULTA
   ManipularDatos1.EjecutarSql1(Query1);

   //4. ACTUALIZA LA CUADRICULA DE DATOS (DataGridView)
   ManipularDatos1.ActualizarGrid1(this.dataGridView2, "select * from entradasalida;");

   //5. DESCONECTA LA BASE DE DATOS
   ManipularDatos1.Desconectar1();

}
else
{
    ComArduino.Write("0");  //CUANDO NO ENTRA EN LA CONDICION ENVIA DATOS POR PUERTO SERIAL A ARDUINO
}
```

Figura 39. Programación para la consulta de históricos.
Fuente: Autores.

8.3.1.3 Botón Salida

Cuando se activa botón por medio de la lectura de salida de la etiqueta se busca la fila correspondiente al código del usuario que va de salida y ubica la casilla correspondiente a la hora de salida para completar el reporte, para esto se agrega el botón a la ventana de interface y se genera la función de respuesta al oprimirlo.

Figura 40. Botón de salida.
Fuente: Autores.

En la siguiente imagen se observa el código contenido dentro de la función del botón; inicialmente se guarda en la variable ***codigo1*** para buscarlo en la base de datos según el código de la lectura actual, después selecciona la celda correspondiente para el dato de salida y por ultimo actualiza la cuadricula.

```
// BUSCA EL CODIGO EN LA BASE DE DATOS SEGUN LA LECTURA DEL CODIGO ACTUAL
Codigo1 = this.dataGridView2.CurrentRow.Cells[0].Value.ToString();
// SELECCIONA LA CELDA CORRESPONDIENTE PARA EL DATO DE SALIDA
txtSalida.Text = this.dataGridView2.CurrentRow.Cells[2].Value.ToString();
// ACTUALIZA LA CUADRICULA DE DATOS PARA EL INGRESO POSTERIOR DE LOS DATOS
ManipularDatos1.ActualizarGrid1(this.dataGridView2, "select * from entradasalida where CODIGO like '"
    + txtBuscar.Text + "%';");
```

Figura 41. Programación para el Botón de salida
Fuente: Autores.

8.3.1.4 Botón Confirmar salida

Con la ubicación de los paramentos anteriores genera la lectura de la etiqueta y se hace uso de la condición if para comparar la lectura actual con la información guardada en la base de datos principal donde se encuentra el registro de usuarios, si la comparación es falsa el programa retorna al inicio, si por el contrario es verdadera lo primero que hace es tomar los datos de hora actuales que luego se usan para el ingreso de hora de salida a la base de datos, luego se crea conexión con la base de datos creada para activarla, como se muestra en la figura 42 posteriormente con la instrucción update set, se actualiza la base de datos en la fila y casilla correspondiente para el usuario en proceso, agregando el dato de hora de salida para completar la tabla, se usa luego la instrucción que selecciona los datos de la tabla y los imprime en la cuadricula de datos (***DataGridView***) que se muestran en a interface.

```
byte[] data = new byte[12];                                  // PROGRAMACION
if (EPCSDKHelper.ReadTag(_handle, 0x01, 0x02, 6, data, 0))   // PARA LECTURA
{                                                            // DE LA ETIQUETA
    byte[] bTmp = new byte[12];                              // RFID
    Buffer.BlockCopy(data, 0, bTmp, 0, 12);                  //
    this.txtBuscar.Text = TextEncoder.ByteArrayToHexString(bTmp);

    // VARIABLES PARA LA COMPARACION EN LA CONDICION IF
    string CodigosBaseDatos = this.dataGridView1.CurrentRow.Cells[0].Value.ToString();
    string LecturaCodigo = TextEncoder.ByteArrayToHexString(bTmp);   //hasta aca

    if (LecturaCodigo == CodigosBaseDatos) // CONDICION PARA GUARDAR INFORMACION DE ENTRADA
    {
        string Parq = Convert.ToString(Parqueaderos);   // PROGRAMACION
        txtParqueaderos.Text = Parq;                    // PARA CONTEO
        Parqueaderos = Parqueaderos - 1;                // DE PARQUEADEROS
        conteo = conteo + 1;
        ComArduino.Write("1");                          // ESCRIBE EN EL PUERTO SERIAL PARA ARDUINO
        // PROGRAMACION PARA DATOS DE HORA DE SALIDA
        datetime = DateTime.Now;
        txtSalida.Text = datetime.Hour + " : " + datetime.Minute + " : " + datetime.Second;
```

Figura 42. Programación para el Botón confirmar salida
Fuente: Autores.

Por ultimo desconecta la base de datos como se muestra en la figura 43 y envía un dato por el puerto serial al arduino para completar la función de salida para elevar la talanquera.

```
string entr = "ENTRADA";   // CREA VARIABLE

//1. CONECTA LA BASE DE DATOS
ManipularDatos1.Conectar1();

//2. ARMA CONSULTA (Query1)
String Query1 = "update entradasalida set SALIDA = '" + txtSalida.Text + "' where CODIGO = '"
    + Codigo1 + "' and SALIDA = '" + entr + "';";

//3. EJECUTA LA CONSULTA
ManipularDatos1.EjecutarSql1(Query1);

//4. ACTUALIZA LA CUADRICULA DE DATOS (DataGridView)
ManipularDatos1.ActualizarGrid1(this.dataGridView2, "select * from entradasalida;");

//5. DESCONECTA LA BASE DE DATOS
ManipularDatos1.Desconectar1();

ManipularDatos1.ActualizarGrid1(this.dataGridView2, "select * from entradasalida where NUMERO like '
    + conteo + "%';");
```

Figura 43. Programación para conectar y desconectar la base de datos cuando se actualiza
Fuente: Autores.

8.3.1.5 Botón Borrar Historial

Este botón permite eliminar el reporte de históricos de los usuarios de la tabla de la base de datos, se agrega entonces el botón y se crea la función que contiene la siguiente secuencia de programación, primero se selecciona manualmente el usuario al que se le va eliminar el reporte de históricos, luego se oprime el botón el cual cumple con las siguientes instrucciones, captura el usuario que se selecciona para eliminar al oprimir el botón el sistema muestra un mensaje de aviso para confirmación de eliminación con el método ***MessageBox.Show***, luego se evalúa la respuesta entregada mediante la instrucción if, si esta es falsa el sistema retorna al inicio si por el contrario la respuesta es

verdadera se conecta con la base de datos creada, posteriormente se arma la consulta para la tabla, en donde se hace uso de la instrucción ***delete from***, la cual elimina la información seleccionada, se actualiza la información luego de la eliminación y con la instrucción ***select * from*** se muestra la información de la tabla de la base de datos con la eliminación que se muestra luego en la tabla o cuadricula de datos que se encuentra en la interface, por último se desconecta base de datos (figura 44).

```
//1. CAPTURA EL CODIGO DE LA BASE DE DATOS entradasalida
Codigo1 = this.dataGridView2.CurrentRow.Cells[0].Value.ToString();

//2. PREGUNTA AL USUARIO SI REALMENTE DESEA ELIMINAR EL REGISTRO
var resultado1 = MessageBox.Show("¿Desear realmente eliminar este registro?",
    "Confirme el borrado", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

//3. PROCEDE A EVALUAR EL RESULTADO DE LA RESPUESTA
if (resultado1 == DialogResult.Yes) // SI LA RESPUESTA EL VERDADERA PROCEDE A ELIMINAR EL REGISTRO
{
    //1. CONECTA LA BADE DE DATOS
    ManipularDatos1.Conectar1();

    //2. ARMA LA CONSULTA (Query)
    String Query1 = "delete from entradasalida where CODIGO = '" + Codigo1 + "';";

    //3. EJECUTA LA CONSULTA
    ManipularDatos1.EjecutarSql1(Query1);

    //4. ACTUALIZA LA CUADRICULA DE DATOS (DataGridView)
    ManipularDatos1.ActualizarGrid1(this.dataGridView2, "select * from entradasalida;");

    //5. DESCONECTA A BASE DE DATOS
    ManipularDatos1.Desconectar1();
}
```

Figura 44. Programación para borrar el historial.
Fuente: Autores.

8.3.2 Conexión de C# con SQL server

Se crea una nueva conexión como la desarrolla para la conexión de la base de datos principal, para crear la conexión entre la aplicación y el servidor de base de datos se crea una nueva clase en C Sharp (Una clase es una construcción que le permite crear sus propios tipos personalizados agrupando variables de otros tipos, métodos y eventos), que ejecuta acciones de conexión, ejecución, desconexión entre otras permitiendo la interacción entre los dos elementos del sistema, activando funciones con las que cuenta la clase pero que son llamadas desde el programa principal. Se agrega entonces a la clase dos directivas ***using*** que permiten utilizar las funciones de interface y manipular variables de conexión de datos de SQL, asi mismo se repite el procedimiento para establecer y crear una base de datos desde el servidor en la sección 8.2.5

Luego se crean la funciones para la interacción y se abre la conexión hacia el servidor con el método ***Open();*** y se agrega a la variable SqlConnection el nombre de la cadena de conexión creada con SQL, como se observa en la figura 45.

```
SqlConnection MiConexion;
4 referencias
public void Conectar()
{
    //la cadena de conexión va a cambiar en sus computadoras...
    MiConexion = new SqlConnection("Data Source=DANIEL\\SQLEXPRESS;Initial " +
    "Catalog=datosrfid;Integrated Security=True");
    MiConexion.Open();
}
```

Figura 45. Abrir la conexión
Fuente: Autores.

Al finalizar las operaciones sobre la base de datos es importante cerrar la conexión usando el método Close(), usado como se ve en la figura 46:

```
public void Desconectar()
{
    MiConexion.Close();
}
```

Figura 46. Cerrar la conexión
Fuente: Autores.

Teniendo creada la clase, se procede a definir la conexión con el servidor, en donde se busca el origen de datos y se habilita la conexión en ambas partes de sistema, para esto se da en proyecto y en agregar nuevo origen de datos, como se ve en la figura 47.

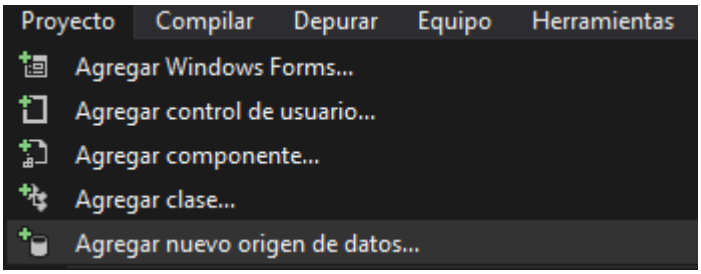

Figura 47. Agregar nuevo origen de datos
Fuente: Autores.

Donde se despliega una ventana de selección de tipo de origen de datos, se selecciona Base de datos, inmediatamente se abre una nueva ventana en donde se elige conjunto de datos.

Ahora la configuración ofrece la oportunidad de elegir la conexión de datos; se elige una nueva conexión y se configura el tipo de conexión como se muestra en la siguiente figura

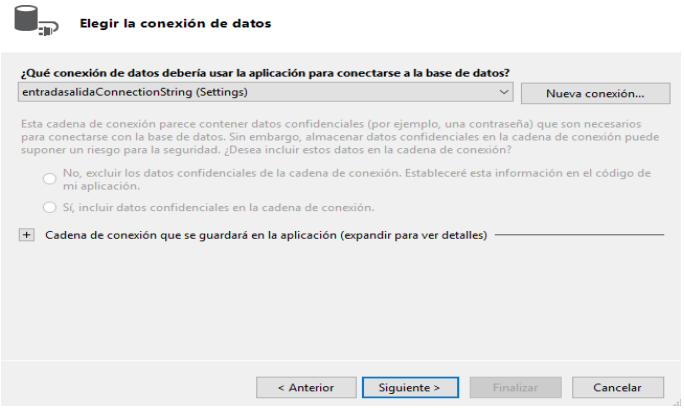

Figura 48. Abrir la conexión
Fuente: Autores.

De nuevo el asistente de configuración, muestra una ventana donde se agrega origen de datos y nombre de servidor, finalmente se busca la base de datos creada en SQL, luego se eligen los objetos de la base de datos(Figura 49) que se desean tener en el conjunto de datos, seleccionando la tabla con el nombre especificado en SQL server.

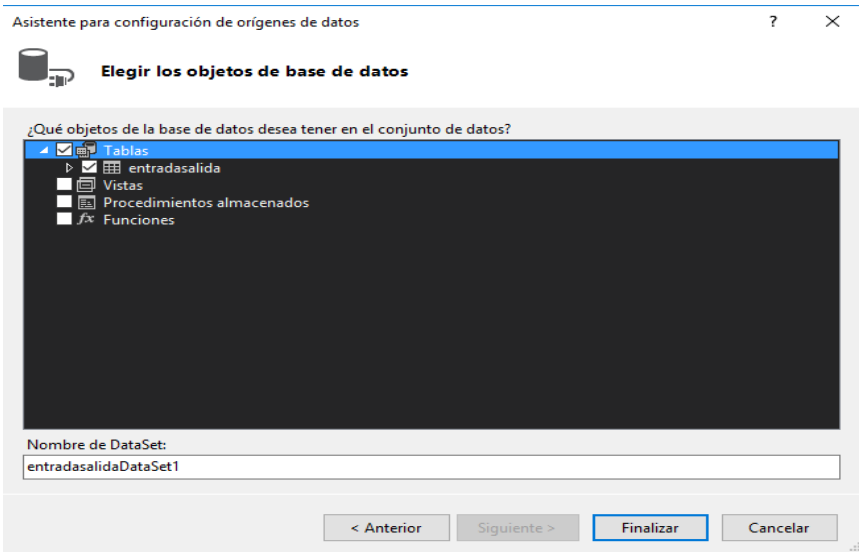

Figura 49. Elegir los objetos de la base de datos.
Fuente: Autores.

8.4 PROGRAMACION EN ARDUINO Y ETAPA DE POTENCIA

8.4.1 **Programación Arduino:**

El arduino fue usado para el sensado que indica la posición del vehículo por medio del módulo de ultrasonido, en este caso se usaron dos sensores, los cuales por medio de la programación en arduino indican si

un vehículo está por entrar o salir, activando el circuito que acciona el movimiento de la talanquera.

En el programa de arduino primero se determinaron las variables para cálculo de distancia y asigna los pines que luego se usaron para la lectura.

```
#define Pecho 8             //DEFINE PIN PARA ECHO DEL MODULO 1
#define Ptrigger 9          //DEFINE PIN PARA TRIGGER DEL MODULO 1
long duracion, distancia;   //VARIABLES PARA CALCULO DE DISTANCIA MODULO 1

#define Pecho1 11           //DEFINE PIN PARA ECHO DEL MODULO 2
#define Ptrigger1 10        //DEFINE PIN PARA TRIGGER DEL MODULO 2
long duracion1, distancia1; //VARIABLES PARA CALCULO DE DISTANCIA MODULO 2
```

Figura 50. Crear variables para cálculo de la distancia.
Fuente: Autores.

En el ***void setup***, se determina la velocidad de lectura serial y se abre la conexión del puerto, también se definieron los pines que se usaron como entradas o salidas.

```
void setup()
{
  Serial.begin(9600);          //ABRE EL PUERTO Y ESTABLE LA VELOCIDAD DE DATOS
  pinMode(Pecho,INPUT);        //DEFINE PIN COMO ENTRADA
  pinMode(Ptrigger,OUTPUT);    //DEFINE PIN COMO SALIDA
  pinMode(13,OUTPUT);          //DEFINE PIN COMO SALIDA
  pinMode(Pecho1,INPUT);       //DEFINE PIN COMO ENTRADA
  pinMode(Ptrigger1,OUTPUT);   //DEFINE PIN COMO SALIDA
}
```

Figura 51. Configuración de entradas y salidas, velocidad comunicación serial
Fuente: Autores.

Luego se agrega la función para activar la lectura por el puerto serial y se evalúa el dato de llegada, el cual indica si el vehículo esta entrado o saliendo, dependiendo de este dato entra a diferente condición y realiza la medición de distancia, si está dentro del rango donde se activa el sensor activa el pin usado para el acople con el circuito de potencia para el motor de la talanquera.

Si la lectura del puerto serial es el numero 1 pero en código Ascii (49), activa el sensor de entrada y se ejecuta el código que se muestra en la figura 52.

```
int leer = Serial.read();       //ASIGNA VARIABLE PARA LA LECTURA SERIAL
if(leer == 49)                  //COMPARA LA LECTURA
{                               //SI ES VERDADER ENTRA AL CICLO

  //ENVIO DE SENAL PARA CALCULO DE DISTANCIA
  digitalWrite(Ptrigger,LOW);
  delayMicroseconds(2);
  digitalWrite(Ptrigger,HIGH);
  delayMicroseconds(10);
  digitalWrite(Ptrigger,LOW);

  //OPERACIONES PARA CALCULO DE DISTANCIA
  duracion = pulseIn(Pecho,HIGH);
  distancia = (duracion/2)/29;

  // CONDICION PARA ACTIVAR EL PIN 13 Y ACITVACION
  // DE CIRCUITO PARA LA TALANQUERA
  if (distancia <= 50 && distancia >= 1)
  {
```

Figura 52. Calculo para hallar distancias.
Fuente: Autores.

Si la lectura del puerto serial es 2 pero en código Ascii (50), activa el sensor de salida y se ejecuta el código que se muestra en la figura 53.

```
if(leer == 50)
{
  //ENVIO DE SENAL PARA CALCULO DE DISTANCIA
  digitalWrite(Ptrigger1,LOW);
  delayMicroseconds(2);
  digitalWrite(Ptrigger1,HIGH);
  delayMicroseconds(10);
  digitalWrite(Ptrigger1,LOW);

  //OPERACIONES PARA CALCULO DE DISTANCIA
  duracion1 = pulseIn(Pecho1,HIGH);
  distancia1 = (duracion1/2)/29;

  // CONDICION PARA ACTIVAR EL PIN 13 Y ACITVACION
  // DE CIRCUITO PARA LA TALANQUERA
  if (distancia1 <= 50 && distancia1 >= 1)
  {
    digitalWrite(13,HIGH);
    delay(5000);
    digitalWrite(13,LOW);
    delay(500);
  }
  else
  {
    digitalWrite(13,LOW);
  }
delay(400);
}
```

Figura 53. Condiciones para activar el motor.
Fuente: Autores.

8.4.2 Etapa de potencia.

Esta sección está desarrollada con el fin de gestionar el funcionamiento de la talanquera, la etapa de potencia se diseña con el objetivo de ser compatible con la barrera automática MICHELANGELO 80 de la compañía BFT

TECHNICAL FEATURES		
	MICHELANGELO 60	MICHELANGELO 80
CONTROL PANEL	LIBRA C MV (incorporated)	LIBRA C MV (incorporated)
REVERSIBLE/IRREVERSIBLE OPERATOR	irreversible	irreversible
POWER SUPPLY	120V 60 Hz single-phase	120V 60 Hz single-phase
ABSORBED POWER	0.4 hp (300W)	0.4 hp (300W)
OPENING OR CLOSING TIME	6 s (19.6')	8 s (26.2')
IMPACT REACTION	electronic torque limiter integrated in the control panel	electronic torque limiter integrated in the control panel
TYPE OF LIMIT SWITCHES	electronic opening/closing	electronic opening/closing
MANUAL RELEASE	fast key release	fast key release
OPERATING CYCLE	very intensive use	very intensive use
ENVIRONMENTAL CONDITIONS	from 5°F to + 140°F	from 5°F to + 140°F
DEGREE OF PROTECTION	IP24	IP24
OPERATOR WEIGHT WITHOUT BOOM	680 N (150lb)	680 N (150lb)
DIMENSIONS	see drawing	see drawing
FUSE VALUES LIBRA CMV	1A T	1A T

Figura 54. Especificaciones eléctricas Michelangelo 80.
Fuente: Sistemas BFT.

Dentro de la especificación técnica se observa que la barrera automática se debe alimentar con 120V AC a 60 HZ, además el fabricante especifica que su sistema de control funciona con tecnología a 24 voltios; el sistema trae un sensor de detección y finales de carrera dentro de los accesorios; debido a esta información se debe garantizar que el sistema de acceso al parqueadero de la Universidad Santo Tomas Tunja sea capaz de cumplir con esta medida.

REFERENCE	DESCRIPTION	VOLTAGE	BOOM LENGTH
P941006 00002	MICHELANGELO 60 RED	24 V	6 m
P941002 00002	MICHELANGELO 60 BLUE	24 V	6 m
P941007 00002	MICHELANGELO 80 RED	24 V	8 m
P941004 00002	MICHELANGELO 80 BLUE	24 V	8 m

Figura 55. Voltaje de control Michelangelo 80.
Fuente: Sistemas BFT

Teniendo la información técnica de la barrera automática también se diseñó el prototipo que incluye el modulo lector y los dos sensores como están dispuestos en la figura 56

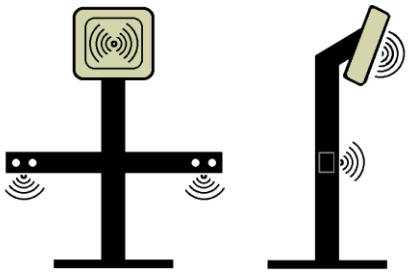

Figura 56. Diseño de prototipo.
Fuente: Autores.

8.4.2.1 Obtención de la señal

Las etapas de potencia son elementos intermedios entre el microcontrolador y aquellos aparatos que requieran cantidades de corriente mayores a los que pueden manejar el microcontrolador, debido a esto el acondicionamiento de la señal que recibe el motor es la parte más importante para la activación del mismo, pues primero se toma el pulso de control enviado por el arduino y se acopla a la etapa de potencia ya que la corriente y voltaje necesarios para la activación de motor debe ser acondicionada y adecuada para que el circuito del motor interprete de manera eficiente la información y por tanto realice la función que se le especifica dependiendo de la instrucción que se le envían desde el programa principal.

8.4.2.2 Etapa de control

Para esto, primero se tomó el pulso enviado por el arduino o pulso de control y se acopla a un circuito que contiene un transistor que trabaja en conmutación que pasa de corte a saturación según la entrada de control que reciba por su base.

La entrada a este circuito es una entrada digital (“0”,”1” lógicos), por tanto cuando la entrada por la base es “0”, no se activan sus otras terminales; se dice entonces que el transistor está en corteo como si se tratara de un interruptor abierto. El transistor está en saturación cuando la entrada lógica es”1”; en ese caso se permite la circulación de corriente entre el colector y el emisor y el transistor se comporta como si fuera un interruptor cerrado.

A la salida del circuito anterior se tiene un relay que se activa o desactiva dependiendo de la señal de control, en donde se busca aislar el circuito de control con el de potencia en caso de que se presente cualquier inconveniente y no se afecte la etapa de control del sistema pues este trabaja con voltajes y corrientes justas y la

etapa de potencia excede estos valores lo que haría que se dañara el controlador.

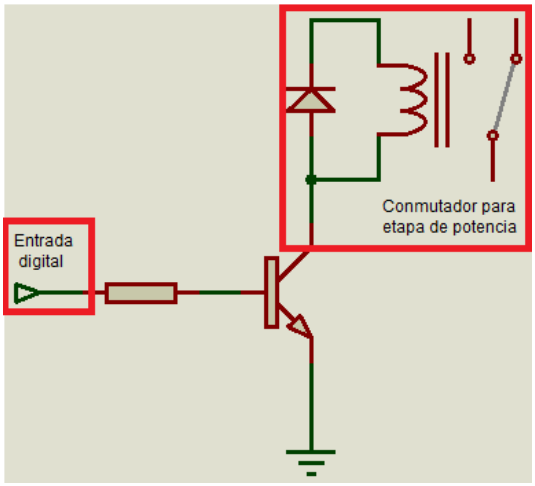

Figura 57. Etapa de control.
Fuente: Autores.

8.4.2.3 Etapa de potencia

A continuación se enuncian las etapas que contiene el circuito de potencia:

Etapa de transformación: Consta básicamente de un transformador el cual integra un bobina primario y uno o varios bobinados secundarios que convierte la energía eléctrica alterna de la red en niveles de energía eléctrica alterna más bajos por medio de la acción de un campo magnético, además provee una aislación galvánica entre la entrada y la salida

En este caso se hace uso de un transformador que convierte los 120 Voltios de energía eléctrica de la red a 24 Voltios en su salida con una corriente de salida de 3 Amperios.

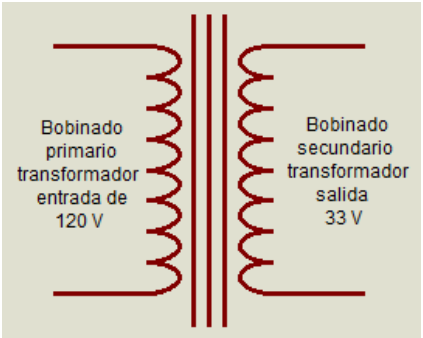

Figura 58. Devanado del transformador
Fuente: Autores.

Etapa de rectificación: La rectificación de onda completa consiste básicamente, en un par de rectificadores de media onda conectados a una resistencia de carga común, lo que hace que los diodos conduzcan durante ciclos alternos. Cuando la polaridad es positiva semi-ciclo positivo de la señal alterna, el extremo superior del transformador tiene un potencial positivo, pasa lo mismo con el semi-ciclo negativo en tanto que este pasa por el extremo inferior del transformador, convirtiendo en señal positiva, circulando en ambos casos en una sola dirección a través de la carga para producir la rectificación de onda completa, como se muestra en la figura 59.

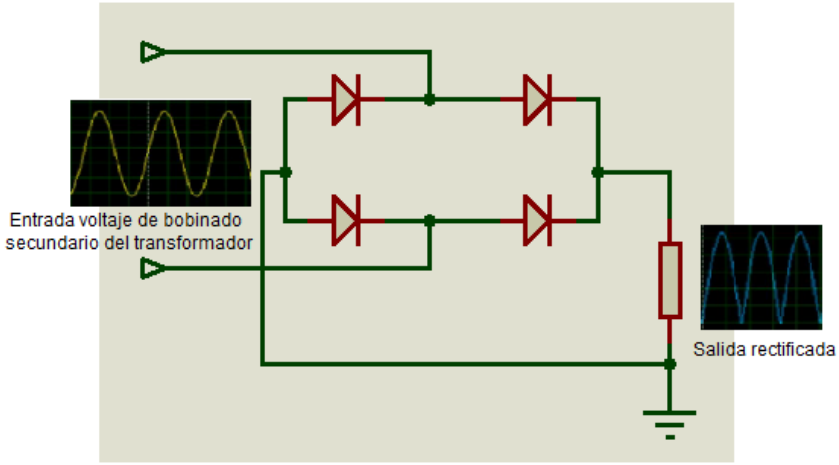

Figura 59. Etapa de rectificación.
Fuente: Autores.

Etapa de filtrado: Esta etapa está constituida varios condensadores en donde su función es eliminar los componentes de tensión alterna que proviene de la etapa de rectificación. Estos se cargan al valor máximo nivel de voltaje que llega desde el rectificador y se descargan lentamente cuando la señal lo cual permite lograr un nivel de tensión lo más continua posible.

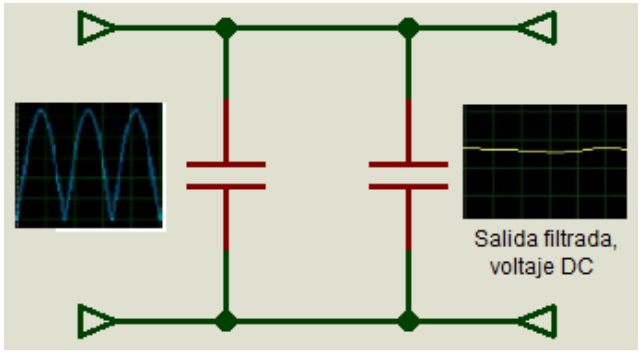

Figura 60. Etapa de filtrado.
Fuente: Autores.

Etapa de regulación: Esta etapa es la encargada de mantener determinado nivel de tensión constante teniendo la capacidad de mantener dicha tensión de salida sin importante el valor de entrada que se le esté entregando.

8.5 DISEÑO DE LA APLICACIÓN DE WINDOWS

Para crear la aplicación final, luego de completar y personalizar la interface de usuario se encuentra que Windows Forms es una de las dos tecnologías que se utiliza en Visual C# para crear aplicaciones inteligentes basadas en Windows. Windows Forms es especialmente adecuado para escenarios de desarrollo rápido de aplicaciones, el Diseñador de Windows Forms se utiliza para crear la interfaz de usuario, y permite obtener acceso a otras características de diseño y ejecución.

Entonces luego de tener el producto del sistema deseado se procede a iniciar el desarrollo de la aplicación de Windows, se sitúa en la solución del proyecto, se hace Click derecho, se despliega otra ventana y seleccionando agregar luego se selecciona en Nuevo proyecto como se ve en la figura 61.

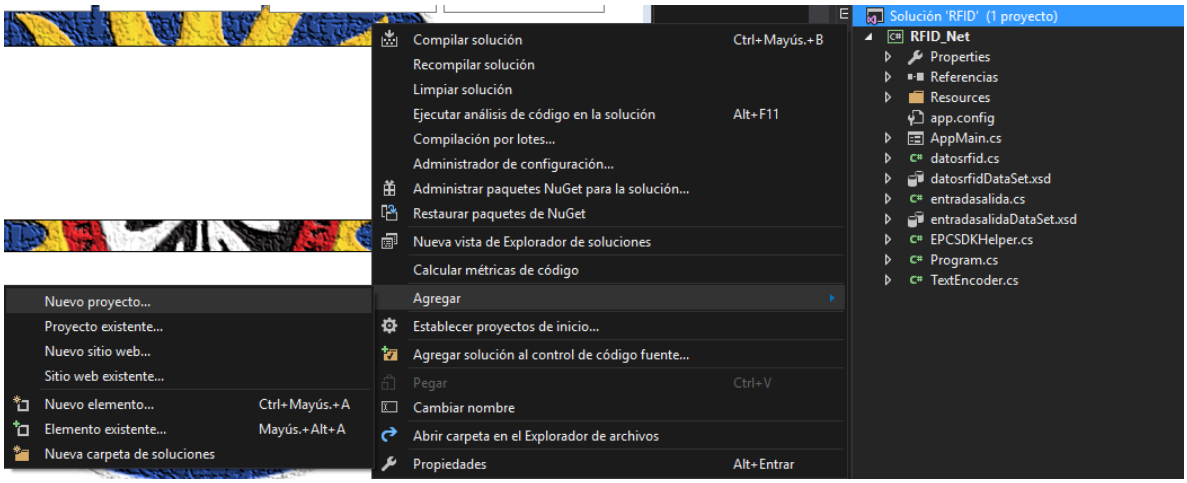

Figura 61. Ejecutando Windows installer
Fuente: Autores.

Se abre una nueva venta donde se selecciona Visual Studio Installer, posteriormente en Setup Project y se da nombre a la aplicación que se va a crear, como se evidencia en la figura 62.

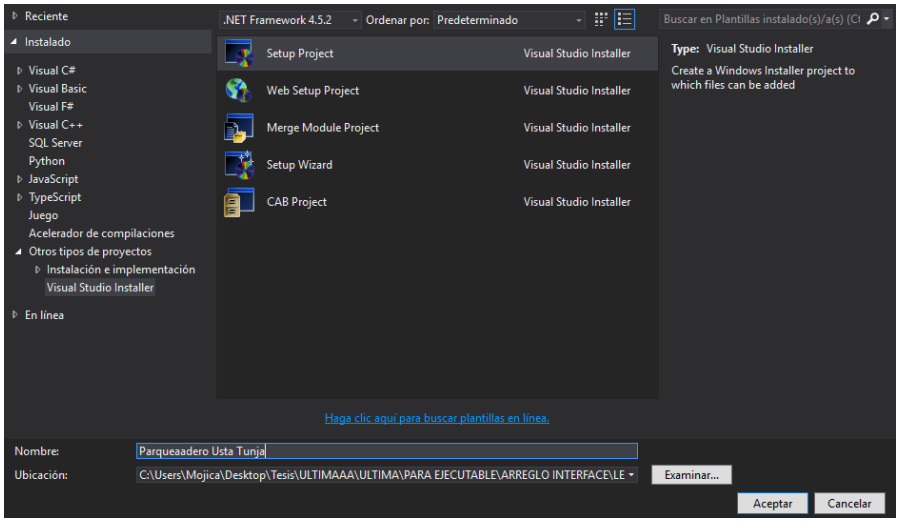

Figura 62. Configurar instalador del proyecto.
Fuente: Autores.

Se abre una nueva ventana en la interface de trabajo de software en donde se agrega cada archivo para generar la aplicación, para esto primero se dirige a carpeta de aplicación que es donde se crea la carpeta en el Disco Local C, dando Click derecho y agregar resultado de proyecto.

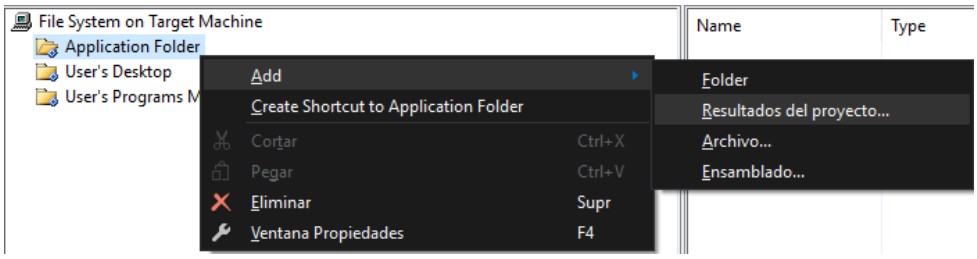

Figura 63. Agregar archivos para crear el instalador de la aplicación.
Fuente: Autores.

Ahora se abre una nueva ventana donde se selecciona Resultado principal y se da Click en Aceptar, esta adiciona todas las extensiones .dll y .exe que posee el sistema.

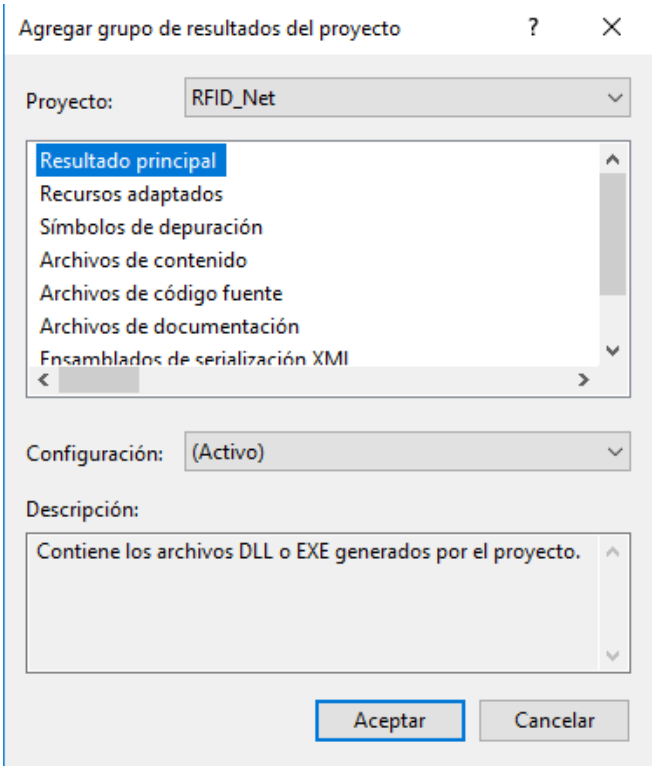

Figura 64. Selección de resultado principal.
Fuente: Autores.

Luego se selecciona escritorio de usuario que es el equivalente a generar el acceso directo en el escritorio, se da Click derecho y crear nuevo acceso directo, como se observa en la figura 65.

Figura 65. Generar Acceso directo
Fuente: Autores.

Se abre una nueva ventana en donde pregunta hacia donde se quiere redimensionar el acceso directo, entonces se da en carpeta de aplicación y a resultado principal.

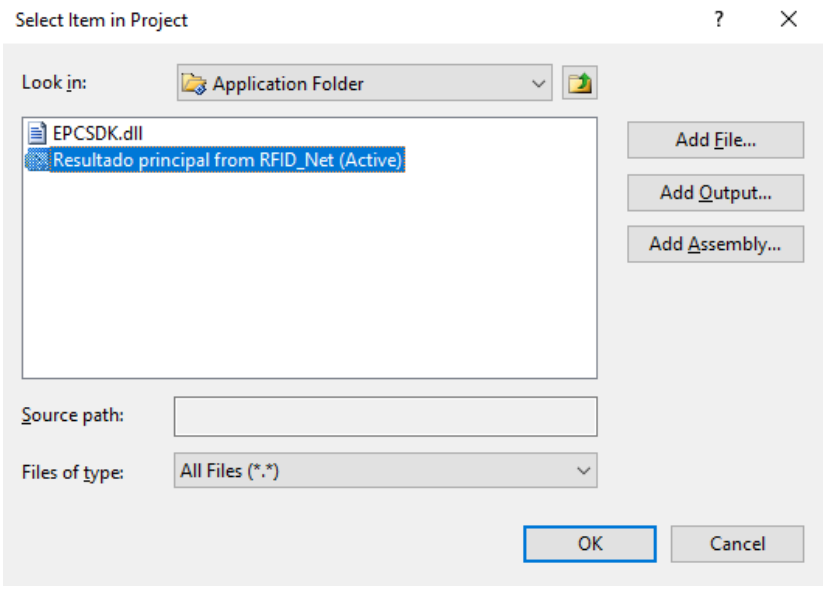

Figura 66. Redimensionar el acceso directo
Fuente: Autores.

Se crea ahora la carpeta que se ubica en el menú de inicio, para esto se hace Click derecho y agregar carpeta como se observa en la figura 67.

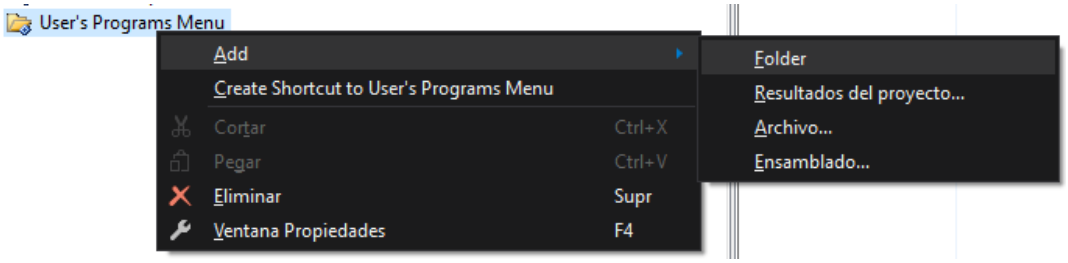

Figura 67. Agregar carpeta al menú de inicio.
Fuente: Autores.

Se abre una nueva ventana y pregunta hacia donde se quiere redimensionar el acceso directo, entonces se da en carpeta de aplicación y a resultado principal.

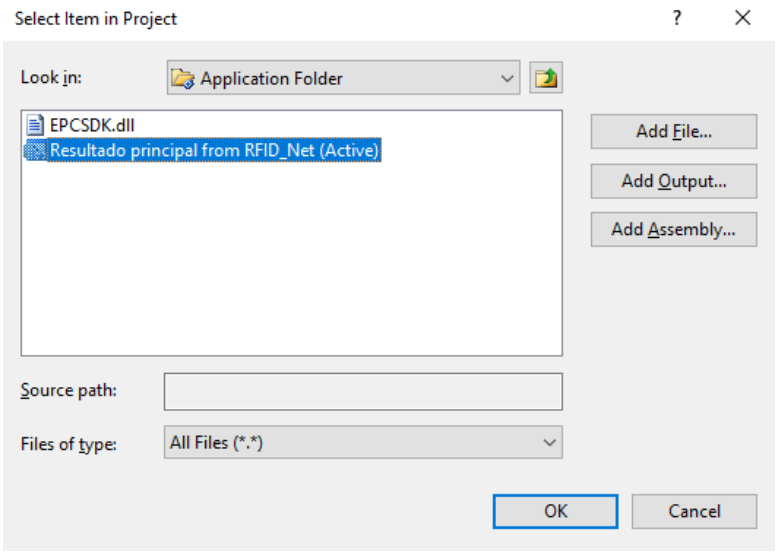

Figura 68. Redimensionar el acceso directo al escritorio
Fuente: Autores.

Adicionalmente a los pasos anteriores se pude personalizar el acceso directo, con el nombre un icono y demás herramientas que pueden hacer lograr una buena aplicación de Windows.

Finalmente para completar la aplicación se va a compilar el proyecto como se ve en la figura 69.

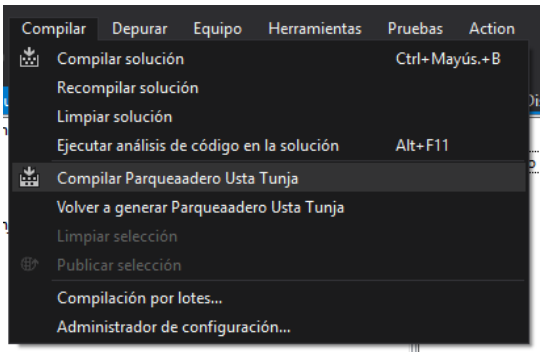

Figura 69. Elegir los objetos de la base de datos.
Fuente: Autores.

Se espera hasta que termine de compilar y se busca la carpeta en donde se creó el ejecutable de la aplicación y se instala normalmente, para luego obtener ya el programa con su acceso directo y la aplicación funcional, obteniendo como resultado final una interface amigable, dinámica y fácil de utilizar como la que se observa en la figura 70.

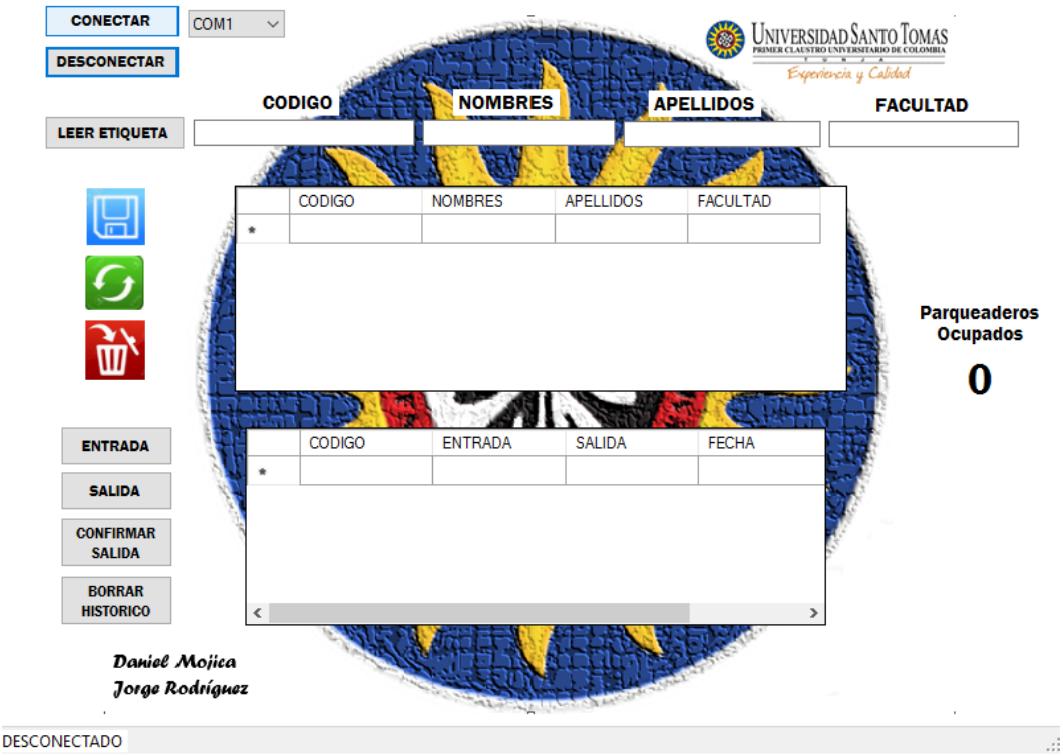

Figura 70. Interfaz de usuario.
Fuente: Autores.

8.6 FUNCIONAMIENTO.

Lo primero que se debe hacer en instalar la aplicación de Windows creada al final del desarrollo del sistema, ya instala se abre el ejecutable y se inicia con la aplicación, previo a esto las conexiones tanto del módulo RFID como del arduino ya deben estar conectados, pues luego se eligen los puertos de conexión para estos dos elementos, para conocer estos puertos en donde están conectados dichos elementos, se busca en la ventana de administrador de dispositivos y en la opción de puertos (COM y LPT) se encuentran los dispositivos conectados y los puertos a los cuales cada uno de estos está conectado, como se observa en la figura 71.

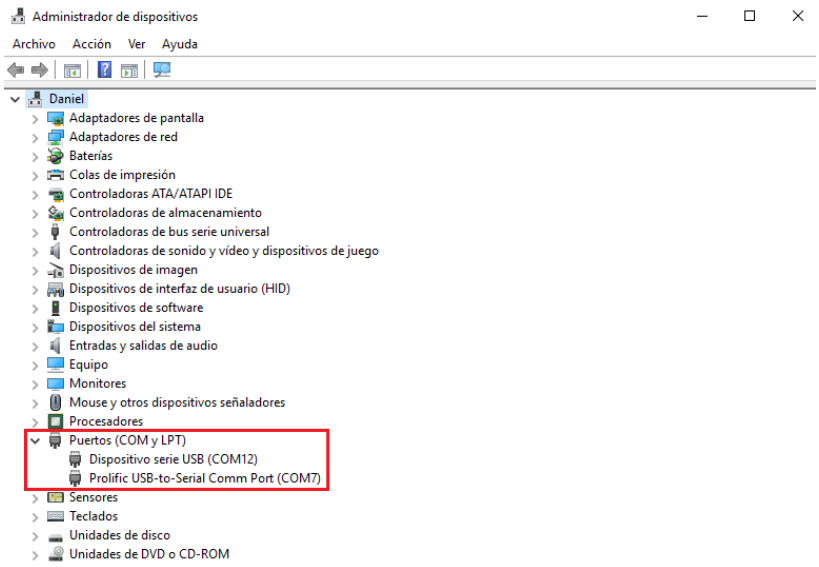

Figura 71. Visualización de puertos del computador.
Fuente: Autores.

Luego de conocer los puertos que están habilitados para cada elemento y con la aplicación ya en ejecución, se busca el puerto de conexión en el botón asignado para esto, se da Click en el botón Conectar y en la parte inferior izquierda se observa cuando se ha completado o activado la conexión con el puerto serial, como se observa en la imagen 72.

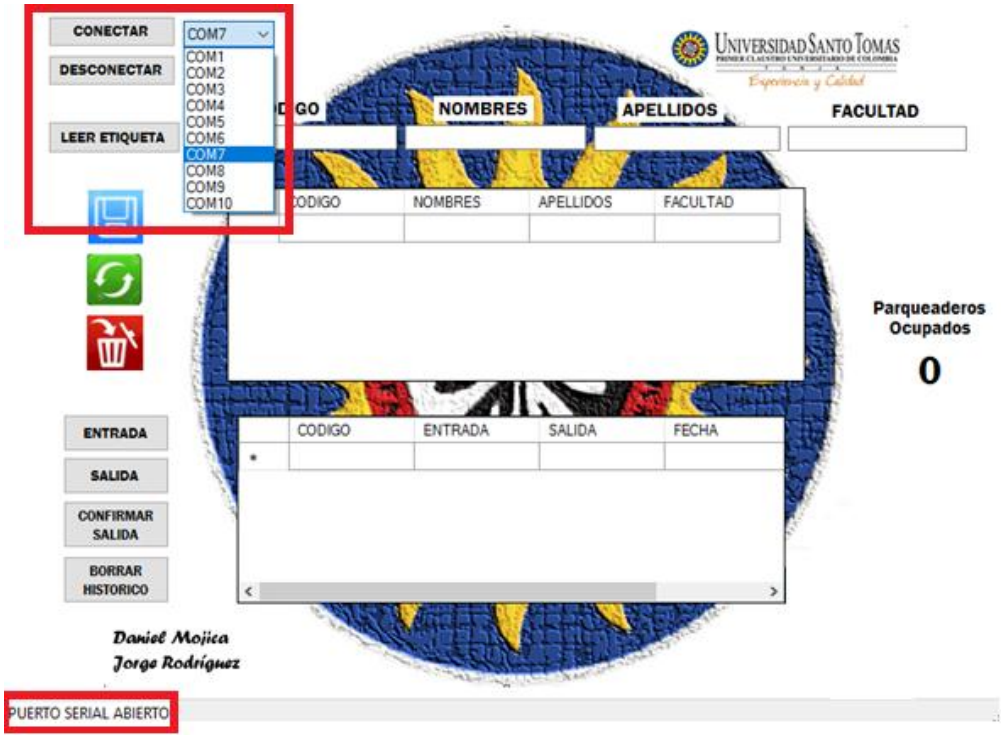

Figura 72. Conexión serial de la interfaz con el interrogador.
Fuente: Autores.

Al momento de tener habilitada la conexión el paso a seguir es asignar cada etiqueta a cada usuario que va siendo agregado a la base de datos, entonces primero se hace lectura de la etiqueta con el botón leer etiqueta, al tener ya el código en la caja de texto se procede a agregar los nombres, apellidos y la facultad a la que pertenece el usuario y se da en el botón guardar, este procedimiento se realiza para cada nuevo usuario que se desee agregar a la base de datos, al momento de oprimir el botón el sistema envía un mensaje que debe ser verificación para completar la acción del botón.

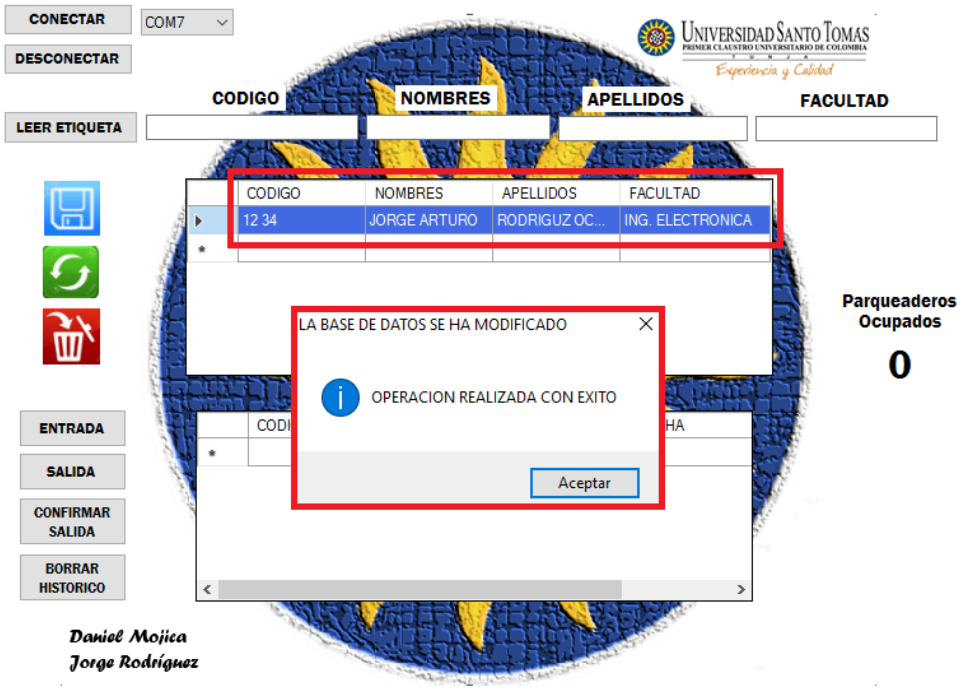

Figura 73. Ingreso de usuario a la base de datos.
Fuente: Autores.

En caso de que la información de algún usuario deba ser modificada o actualizada se selecciona el usuario y se da Click en el botón Actualizar, posteriormente se hacen la modificaciones y se da Cli (MSDN, n.d.)ck en el botón Guardar, mostrando también los mensajes de verificación para dicha acción.

Si en alguna oportunidad el usuario no va a seguir siento parte del sistema de hace uso de los botón Borrar, para esto se selecciona el usuario que se va a eliminar y se da en el botón, el sistema envía un mensaje de verificación y finalmente elimina la información de dicho usuario.

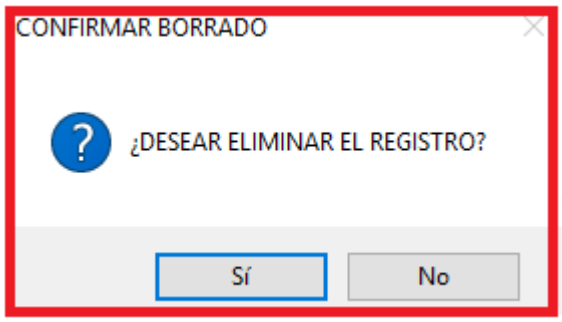

Figura 74. Ventana emergente para borrar registros de usuarios.
Fuente: Autores.

Cuando los usuarios ya están agregados a la base de datos se puede iniciar con el reporte y uso del sistema de interacción con el arduino y la manipulación del motor de la talanquera.

Para lo anterior el vehículo que va a ingresar al parqueadero debe estar en ra que le sensor tenga una lectura sobre este, siendo así en la aplicación se da Click en el botón Entrada, el cual hace lectura de la etiqueta con el módulo RFID y si este usuario está dentro de la base de datos envía un dato por puerto serial haciendo que se produzca la activación de un pin de arduino el cual hace que conmute el circuito de potencia para el motor de la talanquera, como se observa en la figura 75, si por algún caso hay un usuario que no esté registrado en la base de datos principal y quiere ingresar el sistema no habilita ni el reporte de histórico ni la apertura de la talanquera.

Al mismo tiempo en la base de datos de registro de históricos se agrega en una tabla los datos del usuario que ingresa, como el código de la etiqueta, hora de entrada y fecha dejando en la fila la casilla de hora de salida vacía o en espera para generar este valor al momento en el que el usuario realice la salida del parqueadero, Lo mismo se hace para cada uno de los usuarios que deseen ingresar al parqueadero.

En el momento de la salida se tiene otro sensor que detecta la presencia del vehículo, si este está en ubicación correcta, en la aplicación se oprime el botón de Salida el cual primero hace lectura de la etiqueta donde ubica la fila del usuario que va de salida y la celda donde debe ir situada la hora de salida, ahora se acciona el botón confirmar salida el cual genera o imprime finalmente en la base de datos en los espacios correspondiente como se indicó anteriormente la hora de salida del vehículo, completando así la fila o el reporte de entrada y salida para el usuario al mismo tiempo que envía un dato serial al arduino el cual interpreta y genera la activación de la talanquera, este procedimiento es idéntico para completar los demás usuarios que hacen uso del sistema. En la siguiente figura 75 se observa el reporte final para un usuario,

donde se tienen los datos de este y el registro de hora de entrada y salida y la fecha de ingreso.

Figura 75. Interfaz de usuario.
Fuente: Autores.

8.7 ANALISIS DE RESULTADOS

8.7.1 Ubicación de la etiqueta.

Dentro del proceso de desarrollo del prototipo se encontró que es fundamental para el correcto funcionamiento encontrar una ubicación estratégica para que el transponder pueda ser leído por el interrogador; el sistema está desarrollado para que el acceso y la salida vehicular sea por una misma entrada por tanto la identificación para las dos acciones será determinada por un solo modulo lector de RFID de tal manera que la ubicación de la etiqueta debe permitir la identificación en la salida como en la entrada, para ello se hizo un análisis estadístico con cada ubicación, se eligieron 6 puntos estratégicos dentro de la parte frontal del automóvil con 40 intentos por cada ubicación se tiene que tener en cuenta que el sensor debe detectar el automóvil que le permita a la tarjeta estar dentro del rango de lectura que por defecto es de 5m obteniendo los siguientes resultados:

8.7.1.1 Ubicación 1.

Se ubicó la etiqueta en la esquina superior derecha del vidrio panorámico.
Entrada:

Figura 76. Ubicación de la etiqueta en el automóvil
Fuente: Autores.

El automovil esta ubicado dentro del rango de deteccion del sensor de distancia, la longitud de separacion que se eligio es no mayor a un metro para que la distancia entre la etiqueta y el interrogador sea inferior a la especificada por el fabricante cuando se elige la ubicación de la etiqueta.

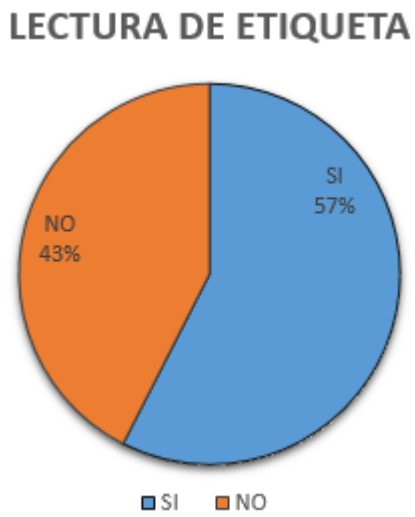

Figura 77. Porcentaje de lectura de etiqueta.
Fuente: Autores.

En los intentos de lectura con respecto a esta ubicación se obtuvo un acierto en lectura 23 veces y 17 veces fallidas, con un porcentaje de acierto del 57% cuando el automovil ingresa.

Salida:

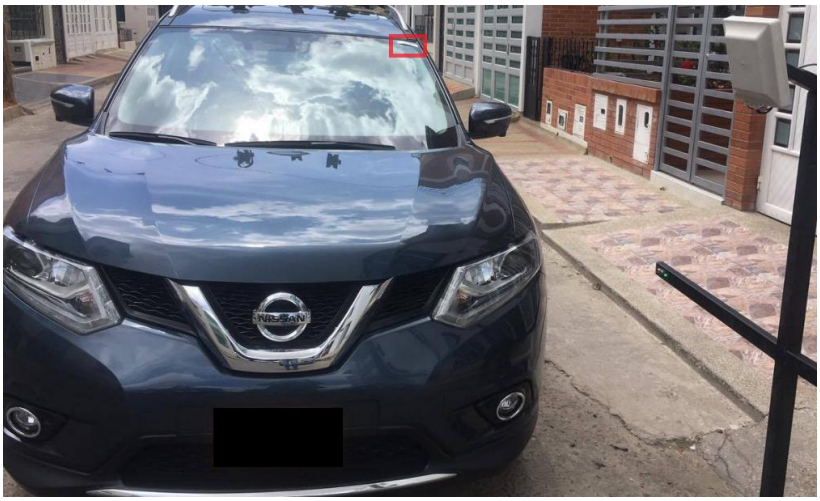

Figura 78. Ubicación de la etiqueta en el automóvil.
Fuente: Autores.

Se observa que el angulo entre la etiqueta y el interrogador disminuye, dificultando la lectura arrojando los siguientes resultados:

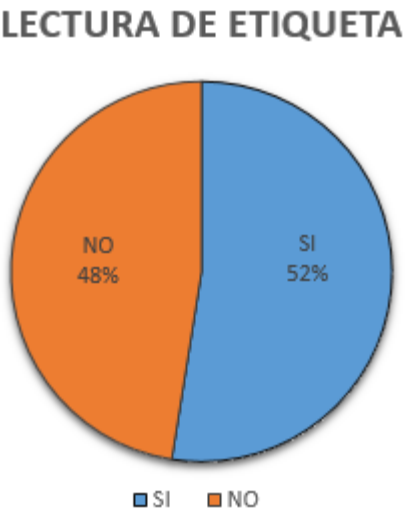

Figura 79. Porcentaje de lectura de etiqueta.
Fuente: Autores.

Se observa que hay 21 aciertos de lectura y 19 intentos fallidos, con un porcentaje de acierto del 52% en la salida del vehiculo.

8.7.1.2 Ubicación 2.

Se localizó la etiqueta en la esquina superior izquierda del vidrio panorámico del automóvil.

Entrada:

Figura 80. Ubicación de la etiqueta en el automóvil.
Fuente: Autores.

Nuevamente el automóvil está ubicado dentro del rango del sensor de proximidad y se procede a hacer las 40 mediciones de la etiqueta en donde se obtuvieron los siguientes resultados:

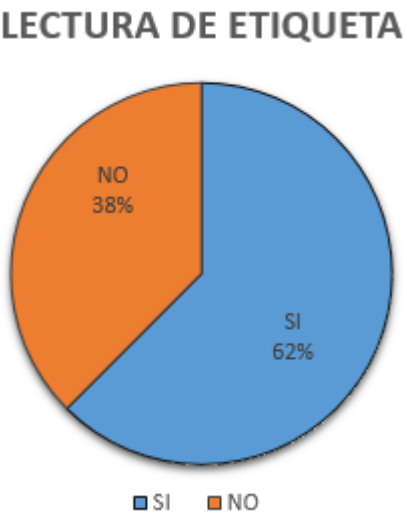

Figura 81. Porcentaje de lectura de la etiqueta.
Fuente: Autores.

Se observaron 25 aciertos en y 15 fallidos con un 62% de probabilidad de lectura.

Salida:

Figura 82. Ubicación de la etiqueta en el automóvil
Fuente: Autores.

El ángulo entre la etiqueta y el lector mejora considerablemente mostrando los siguientes resultados en la salida:

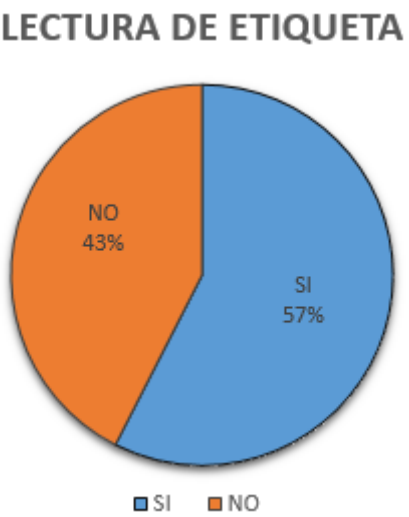

Figura 83. Porcentaje de lectura de la etiqueta.
Fuente: Autores.

En esta ocasión disminuye el porcentaje de acierto en cuanto a la entrada incluso mejorando el ángulo de vista de la etiqueta hacia el lector obteniendo un porcentaje de acierto del 57% con 23 lecturas y 17 intentos fallidos.

8.7.1.3 Ubicación 3.

En esta oportunidad se ubica la etiqueta en el centro de la parte superior del vidrio panorámico.

Entrada:

Figura 84. Ubicación de la etiqueta en el automóvil.
Fuente: Autores.

Se observa que la etiqueta tiene mejor ángulo para la lectura por parte del interrogador, el tag queda con vista total hacia el modulo y su antena queda en, mejor disposición para su lectura.

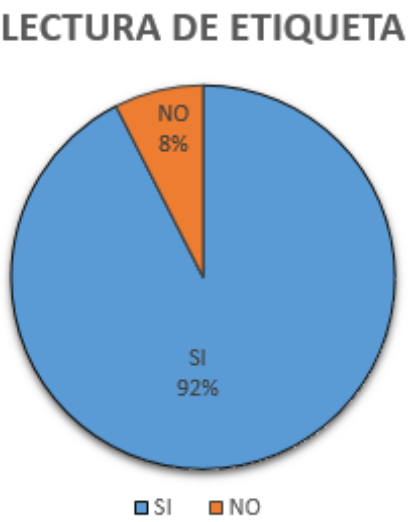

Figura 85. Porcentaje de lectura de la etiqueta.
Fuente: Autores.

Con esta ubicación del tag, hay un porcentaje de lectura del 92 % con un total de 37 aciertos y 3 intentos fallidos en el ingreso al sistema.

Salida:

Figura 86. Ubicación de la etiqueta en el automóvil.
Fuente: Autores.

Esta ubicación ofrece un ángulo de lectura en la salida amplio con excelente disposición de la antena del tag hacia el interrogador es por esto que en los 40 intentos se obtienen los siguientes resultados.

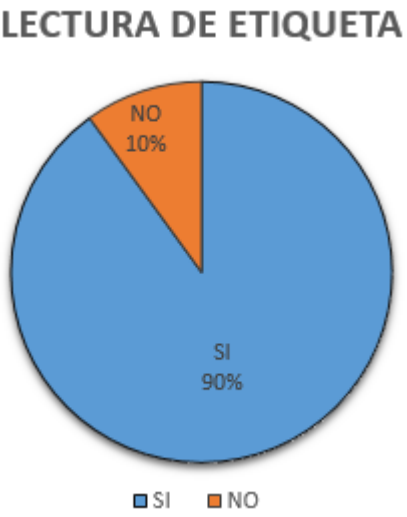

Figura 87. Porcentaje de lectura de etiqueta.
Fuente: Autores.

En la salida se obtiene un porcentaje de acierto del 90% con 36 intentos correctos de lectura y 4 intentos fallidos.

8.7.1.4 Ubicación 4.

Se localiza el tag en la parte inferior del capo

Entrada:

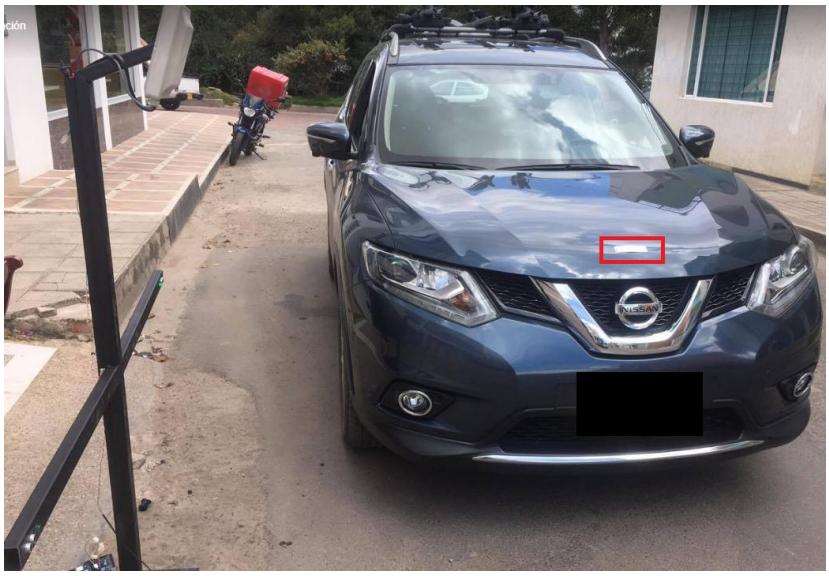

Figura 88. Ubicación de la etiqueta en el automóvil.
Fuente: Autores.

Es la primera ocasión en donde se ubica el tag en una marte metálica sobre la carrocería del automóvil, mantiene buena ubicación respecto al interrogador, y se obtienen los siguientes resultados:

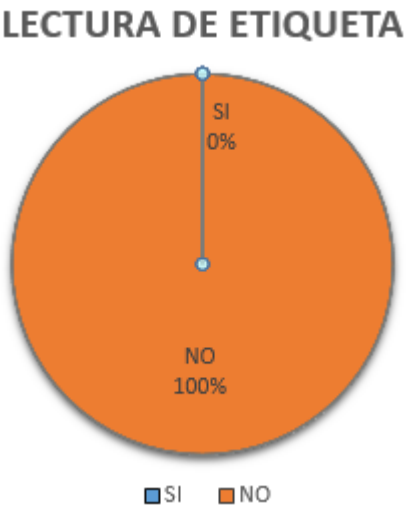

Figura 89. Porcentaje de lectura de la etiquetas.
Fuente: Autores.

En esta práctica, se observó que no hay ningún porcentaje de acierto en la lectura por tal razón se considera interferencia de la superficie metálica de la camioneta con la antena del tag.

Salida:

Figura 90. Ubicación de la etiqueta en el automóvil.
Fuente: Autores.

Se procede a hacer la prueba de salida del automóvil con la ubicación sobre el capo y nuevamente se observa resultados nulos con respecto a la lectura.

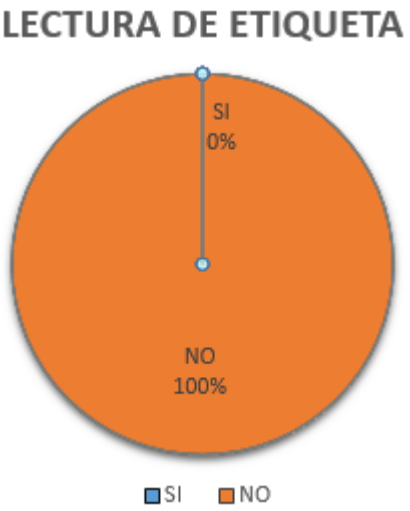

Figura 91. Porcentaje de lectura de etiquetas.
Fuente: Autores.

8.7.1.5 Ubicación 5.

Entrada:

Figura 92. Ubicación de la etiqueta en el automóvil.
Fuente: Autores.

Nuevamente se ubica la etiqueta sobre el capo, pero en esta ocasión se cambia la ubicación para tratar de mejorar los resultados con respecto a la prueba anterior o ratificar la interferencia de superficies metálicas con la antena del tag.

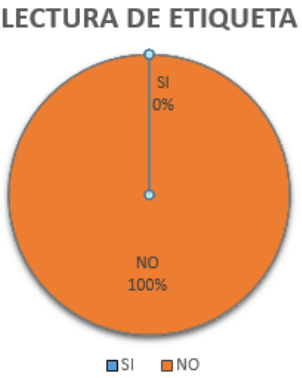

Figura 93. Porcentaje de lectura de la etiqueta.
Fuente: Autores.

La práctica arroja resultados nulos con ningún acierto sobre la lectura del tag evidenciando total interferencia la superficie metálica del automóvil con la lectura del trasnponder.

Salida:

Figura 94. Ubicación de la etiqueta en el automóvil.
Fuente: Autores.

Al completar el ejercicio de entrada-salida, nuevamente se obtienen resultados nulos para la lectura del tac, ratificando la interferencia de la carrocería del automóvil con el tag.

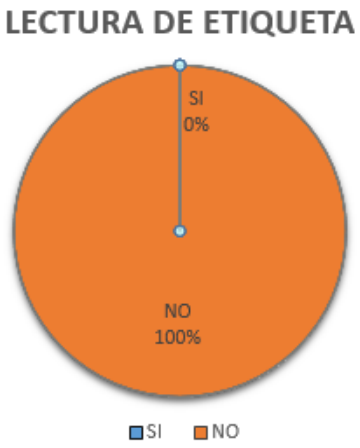

Figura 95. Porcentaje de lectura de etiqueta.
Fuente: Autores.

8.7.1.6 Ubicación 6.

Se coloca el tag en el medio de la parte inferior del vidrio panorámico

Entrada:

Figura 96. Ubicación de la etiqueta en el automóvil.
Fuente: Autores.

Se tiene en cuenta esta ubicación con el fin de no modificar la disposición del tag con el interrogador cambiando la superficie metálica por el vidrio, de esta manera se observara según los resultados el cambio de material.

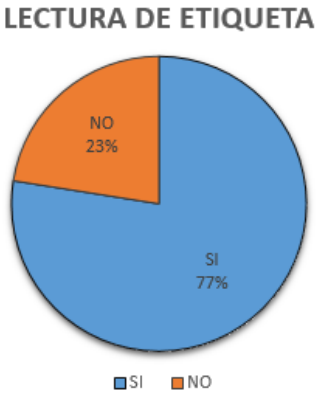

Figura 97. Porcentaje de lectura de la etiqueta.
Fuente: Autores.

Los resultados obtenidos son evidentes y se observa que el material sobre la superficie influye en la lectura del tag arrojando un 77% de acierto en la lectura de la entrada al sistema.

Salida:

Figura 98. Ubicación de la etiqueta en el automóvil.
Fuente: Autores.

En la salida del sistema se observa que hay un 70% de acierto en la lectura con 28 intentos correctos y 30% de error.

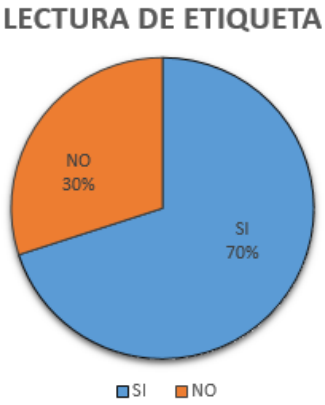

Figura 99. Porcentaje de lectura de la etiqueta.
Fuente: Autores.

En la siguiente tabla se muestra la comparación entre los resultados para decidir cuál es la mejor ubicación para el ingreso y salida del sistema propuesto:

Comparación de porcentajes de lectura en la entrada:

UBICACIÓN	PORCENTAJE DE LECTURA
1	57
2	62
3	92
4	0
5	0
6	77

Tabla 9. Porcentaje de acierto de cada ubicación para la entrada.
Fuente: Autores.

Comparación de porcentajes de lectura a la salida:

UBICACIÓN	PORCENTAJE DE LECTURA
1	52
2	57
3	90
4	0
5	0
6	70

Tabla 10. Porcentaje de acierto de cada ubicación para la salida.
Fuente: Autores.

Observando las tablas anteriores la ubicación con mejor porcentaje de lectura es la ubicación 3 en donde el tag está localizado en el medio de la parte superior del vidrio panorámico

8.7.2 Pruebas de funcionamiento en el campus de la Universidad Santo Tomás seccional Tunja.

Ya definida la ubicación mas eficiente para la lectura de la etiqueta tanto para la entrada como la salida, del prototipo del sistema del modulo RFID y de los sensores, se realizaron pruebas completas del funcionamiento, observando primero la correcta ubicación del automovil al activarse el sensor y posteriormente hacer la accion de entrada o de salida en la aplicación para el usuario a verificar y generar luego el reporte de historico de hora de entrada y salida de automovil.

A continuacion se observa en las figuras 100 y 101, la prueba de funcionamiento realizada durante el dia y la noche, donde se evidencia que la etiqueta esta en una ubicación y distancia para una lectura eficiente, asi como tambien la buena ubicación del automovil, pues se ha activado el sensor dispuesto para la deteccion.

El sistema tiene un correcto funcionamiento debido a que principalmente se ha encontrado la ubicación adecuada de la etiqueta y se han seguido los pasos debidos para la funcion que debe presentar el sistema, tanto de hardware como de software, es importante agregar que no se presenta alguna diferencia del funcionamiento en el sistema en diferentes ambientes y que finalmente tanto en el dia como en la noche el sistema trabaja correctamente aun si el sistema se ve afectado por algun ambiente externo o artificial (luz artificial), pues según la infraestructura presente en la universidad el sistema respondio satisfactoriamente.

Figura 100. Ubicación automóvil para entrada y salida del parqueadero durante el día.
Fuente: Autores.

Figura 101. Ubicación automóvil para entrada y salida del parqueadero durante la noche.
Fuente: Autores.

8.7.3 Consumo de potencia:

Elemento	**Potencia**
Arduino	100 mW
Modulo lector RFID PT-3LA01Z	36 W
Fuente control lógico barrera de acceso vehicular	72 W
Total prototipo	**98.1 W**

Tabla 11. Consumo de potencia de cada elemento del sistema.
Fuente: Autores.

9. APORTES DEL TRABAJO

Articulo redactado para presentar ponencia en SIMPOSIO DE INVESTIGACIÓN USTAMED 2016.

El prototipo está desarrollado para ser compatible con barreras vehiculares que en su etapa de control usen voltaje de 0 a 24 V, se recomienda la barrera que ofrece la compañía BFT Colombia de la serie MICHELANGELO 80 debido a sus características de tamaño y la ventaja del control; el producto contiene accesorios que hacen de este producto muy completo y adecuado para el prototipo.

En las pruebas de funcionamiento en el campus de la Universidad Santo Tomas Tunja se evidencio el correcto funcionamiento del prototipo, se realizaron pruebas en la entrada del parqueadero durante el día y la noche, mostrando el mismo comportamiento y desempeño en las dos situaciones demostrando que el sistema cumple los requisitos para mejorar la movilidad vehicular al ingreso del parqueadero.

El Proyecto no tiene un nivel completo de automatización, sin embargo cumple con los objetivos propuestos y ofrece el servicio de llevar un control de uso para cada usuario, busca mejorar el acceso para quienes utilicen este servicio, y limitar las funciones del personal de vigilancia debido al desplazamiento que deben hacer para abrir o cerrar la puerta de la entrada vehicular, el número de personal de vigilancia no se ve afectado por el prototipo pero si se implementa el sistema se obtendrá un beneficio en el acceso y la movilidad, se reducirá el tiempo de ingreso y facilitara al encargado sus funciones para prestar mayor atención al ingreso peatonal.

Con el estudio de tecnologías RFID es recomendado incluir contenidos sobre el tema de identificación por radiofrecuencia ya que posee características excepcionales y un contenido muy interesante para aplicar en semilleros de investigación.

10. CONCLUSIONES

- Se logró cumplir con los objetivos propuestos en el proyecto debido a que se diseñó un prototipo capaz de llevar un registro detallado para cada usuario con el fin de optimizar el acceso.

- Se diseñó el software que permite la comunicación con el lector RFID PT-3LA01Z para leer las etiquetas y asignar al usuario un código de identificación, en esta etapa de desarrollo se consultaron distintos manuales ofrecidos por el fabricante y es importante destacar el aprendizaje en el manejo de trama de datos en protocolos de comunicación y acceso a registros de memoria de los dispositivos para su correcto manejo.

- En el desarrollo del proyecto se adquieren nuevas herramientas para el desarrollo de software que optimice el uso de recursos y aporten funciones con utilidad flexible y potente como son el uso de bibliotecas de vinculo dinámicos (.DLL) que contienen recursos compartidos con Windows.

- El prototipo contiene el diseño de una interfaz de fácil acceso, porque se obtuvo la aplicación de escritorio para que el designado de administrar la base de datos se ubique rápidamente en un entorno amigable y pueda ofrecer un uso óptimo de la aplicación; Visual Studio ofrece la herramienta Windows installer que posibilita gestionar y fabricar instaladores de modo que se sincronicen los comandos y producto instalados permitiendo mantener características importantes como el llamado de bibliotecas dinámicas y sostener el uso compartido de recursos .

- C# es un lenguaje orientado a objetos, que contiene características para el desarrollo de software y de aplicaciones de Windows que utiliza todos los componentes del sistema, tiene total compatibilidad con los puertos seriales, y facilita la integración de software con hardware, un claro ejemplo es la implementación del programa de lectura en donde se adaptó el protocolo EPC ofrecido por el fabricante para ofrecer una aplicación eficiente.

- EPC (Electronic Product Code) es un concepto de identificación a nivel mundial que ofrece características inigualables con otros sistemas de identificación automática; este modo de identificación por radiofrecuencia ofrece la posibilidad de identificar mercancías en tiempo real asignándole una etiqueta que en el interior contiene una trama de datos única que almacena información sobre características del producto, tipo de empaque o incluso el fabricante, este tipo de tecnología es aplicable al prototipo de acceso al

parqueadero de la sede campus de la Universidad Santo Tomas, debido a que se pueden usar las diferentes cabeceras de información para identificar datos del usuario, como por ejemplo facultad, semestre, y código interno de la institución para permitirle una mayor eficiencia al sistema.

- El uso de tecnologías UHF representa un abaratamiento en la implementación de sistemas RFID debido a que facilita el uso de etiquetas pasivas que tienen características de flexibilidad y vida útil mayores que las activas, además este rango de frecuencias hacen que el diseño de la antena sea más sencillo y de menor tamaño, haciendo que el interrogador requiera la menor potencia posible para energizar el tag haciendo que la radiación espurea sea lo más baja posible evitando perdida de potencia en la señal de transmisión.

- La ubicación de las etiquetas RFID de propósito general que ofrece el fabricante Plintec debe ser sobre superficies no metálicas, debido interferencias con la antena del tag; se recomienda que el transponder sea ubicado en el medio de la parte superior del vidrio panorámico para optimizar su lectura.

11. ANEXOS

A. Articulo IEEE “Diseño de prototipo para control de acceso al parqueadero del campus de la Universidad Santo Tomas Tunja”
B. Manual técnico y de usuario
C. SDK Development Guide V3.5
D. Reader Communication Protocol 3.5

BIBLIOGRAFIA

(ORSI), O. R. (2007). *RFID Tecnologia de identificacion por radiodfrecuencia y sus principales aplicaciones.* Junta de Castilla y Leon: Consejeria de fomento.

Carro, R., & Gonzalez Daniel. (2012). *Identificación Automática.* Mar de Plata.

Finkenzeller, K. (2010). *RFID Handbook:Fundamentals and applications in contactless Smart cards, Radio Frecuency Identification and Neard.* Munich: John Wiley & Sons, Ltd.

Lozano, A. (2011). *RFID Design Fundamentals and Applications.* Boca Raton: Taylor & Francis Group.

MSDN. (n.d.). *Microsoft*. Retrieved from Microsoft: https//msdn.microsoft.com

P., J. (2008). *tecnolofia de identifiacion por radiofrecuencia (RFID): aplicaciones en el ambito de la salud.* Madrid: Fundacion madri+d para el Conocimiento.