UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERIA EN TELECOMUNICACIONES

TRABAJO DE INTEGRACIÓN CURRICULAR PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TELECOMUNICACIONES

"DESPLIEGUE DE UN SISTEMA PARA SMARTPHONES QUE PERMITA LA LECTURA DEL CHIP IMPLANTE X3 MEDIANTE LA TECNOLOGÍA NFC PARA LA RESPUESTA TEMPRANA DEL TRIAJE DE EMERGENCIAS EN LA CLÍNICA DE SIMULACIÓN DE LA UNIVERSIDAD TÉCNICA DEL NORTE"

AUTOR: José Camilo Nogales Romero

DIRECTOR: Ing. Hernán Mauricio Domínguez Limaico Ms.C.

Ibarra-Ecuador

2024

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	1003858352		
APELLIDOS Y NOMBRES:	NOGALES ROMERO JOSE CAMILO		
DIRECCIÓN:	LOS CEREZOS A9 Y AV. MONS. LEÓNIDAS PROAÑO (ANILLO VIAL)		
EMAIL:	jcnogalesr@utn.edu.ec		
TELÉFONO FIJO:	----------	**TELÉFONO MÓVIL:**	0939992678

DATOS DE LA OBRA	
TÍTULO:	DESPLIEGUE DE UN SISTEMA PARA SMARTPHONES QUE PERMITA LA LECTURA DEL CHIP IMPLANTE X3 MEDIANTE LA TECNOLOGÍA NFC PARA LA RESPUESTA TEMPRANA DEL TRIAJE DE EMERGENCIAS EN LA CLÍNICA DE SIMULACIÓN DE LA UNIVERSIDAD TÉCNICA DEL NORTE
AUTOR (ES):	JOSE CAMILO NOGALES ROMERO
FECHA DE APROBACIÓN: DD/MM/AAAA	05/03/2024
PROGRAMA:	■ **PREGRADO** ☐ **OSGRADO**
TITULO POR EL QUE OPTA:	INGENIERO EN TELECOMUNICACIONES
ASESOR /DIRECTOR:	ING. MAURICIO DOMÍNGEZ Ms.C.

2. CONSTANCIAS

El autor (es) manifiesta (n) que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto la obra es original y que es (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 06 días del mes de marzo de 2024

EL AUTOR:

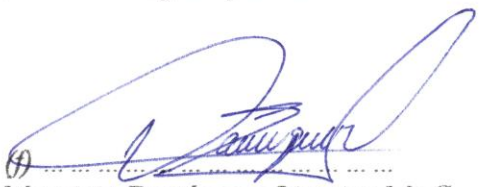
(Firma) .....
Nombre: JOSE CAMILO NOGALES ROMERO

CERTIFICACIÓN DEL DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Ibarra, 31 de Enero de 2024

Ing. Mauricio Domínguez Limaico Ms.C.
DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

CERTIFICA:
Haber revisado el presente informe final del trabajo de Integración Curricular, el mismo que se ajusta a las normas vigentes de la Universidad Técnica del Norte; en consecuencia, autorizo su presentación para los fines legales pertinentes.



Ing. Mauricio Domínguez Limaico Ms.C.
C.C.: 1002379301

APROBACIÓN DEL COMITÉ CALIFICADOR

El Comité Calificador del trabajo de Integración Curricular "DESPLIEGUE DE UN SISTEMA PARA SMARTPHONES QUE PERMITA LA LECTURA DEL CHIP IMPLANTE X3 MEDIANTE LA TECNOLOGÍA NFC PARA LA RESPUESTA TEMPRANA DEL TRIAJE DE EMERGENCIAS EN LA CLÍNICA DE SIMULACIÓN DE LA UNIVERSIDAD TÉCNICA DEL NORTE" elaborado por JOSE CAMILO NOGALES ROMERO previo a la obtención del título de INGENIERO EN TELECOMUNICACIONES aprueba el presente informe de investigación en nombre de la Universidad Técnica del Norte:

(f):.....

Ing. Mauricio Domínguez Ms.C

C.C.: 1002379301

(f):.....

Ing. Edgar Maya Ms.C.

C.C.: 1002702197

DEDICATORIA

Dedico este trabajo a mis padres, Luis y Sonia, que han sido un pilar fundamental durante toda mi vida estudiantil. Esto no hubiese sido posible sin su ayuda, cariño y comprensión.

Así también dedico este trabajo a la persona por la cual me decidí en estudiar ciencias exactas e ingeniería, a Don Luis Eduardo Nogales Herrera.

Desde aquí hacia la eternidad,

Tu eterno Daña-Teles

AGRADECIMIENTO

Es de mi agrado reconocer a todas las personas que hicieron posible este trabajo de titulación. En primer lugar, a mis padres por su acompañamiento y guía durante toda esta etapa.

A mis hermano Pável, por su ayuda técnica en el desarrollo de este proyecto de titulación.

A mi hermana Nikole, por su ayuda como IRM de la Carrera de Medicina de la UTN para la realización de pruebas.

A la Carrera de Medicina de la UTN, sus profesionales y sus estudiantes, por su acompañamiento y tutoría durante el desarrollo del Trabajo de Titulación.

A mis profesores, Ing. Mauricio Domínguez e Ing. Edgar Maya, por su guía y acompañamiento dentro de este proceso, las ideas y discusiones fortalecieron esta investigación aún más.

A mis amigos que tuve el agrado de conocer durante toda mi etapa estudiantil.

Y finalmente a ti, Milena Arellano, gracias y mil gracias siempre por ayudarme a no desfallecer con tu amor y cariño, *Te amo.*

RESUMEN EJECUTIVO

Este proyecto de titulación se fundamenta en la necesidad de mejorar los procesos de triaje en emergencias médicas mediante el uso de la tecnología NFC y el chip implante X3. El objetivo general es desarrollar una aplicación móvil que optimice la lectura de información médica almacenada en el chip para agilizar el proceso de atención médica. La metodología aplicada involucra la unión de la metodología XP y la de desarrollo de aplicaciones móviles, con pruebas exhaustivas tanto en entornos de emulación como en dispositivos reales.

Los resultados más relevantes se centran en la significativa reducción de tiempos de triaje, con un promedio del 38.70%, demostrando la eficiencia del sistema. Este logro se respalda con pruebas de detección de etiquetas NFC y de rendimiento, las cuales garantizan la fiabilidad y precisión de la aplicación. La conclusión más destacada es que la implementación de este sistema tiene el potencial de transformar los procedimientos de triaje, mejorando la eficacia y la contribución de la tecnología en pro de salvar vidas.

En resumen, este proyecto propone una solución tecnológica innovadora, respaldada por una metodología robusta y pruebas rigurosas, que podría tener un impacto significativo en la atención médica de emergencia.

Palabras clave: NFC, triaje, chip implante, aplicación móvil, etiqueta, emergencias

ABSTRACT

This degree project is based on the need to improve triage processes in medical emergencies through the use of NFC technology and the X3 implant chip. The overall objective is to develop a mobile application that optimizes the reading of medical information stored on the chip to streamline the medical care process. The applied methodology involves the union of the XP methodology and the mobile application development methodology, with exhaustive testing both in emulation environments and on real devices.

The most relevant results focus on the significant reduction of triage times, with an average of 38.70%, demonstrating the efficiency of the system. This achievement is supported by NFC tag detection and performance tests, which guarantee the reliability and accuracy of the application. The most salient conclusion is that the implementation of this system has the potential to transform triage procedures, improving efficiency and the contribution of technology to saving lives.

In summary, this project proposes an innovative technological solution, supported by a robust methodology and rigorous testing, which could have a significant impact on emergency medical care.

Keywords: NFC, triage, implant chip, mobile app, tag, emergency

LISTA DE SIGLAS

NFC. Near Field Communication (Comunicación de Campo Cercano)

NDEF. NFC Data Exchange Format (Formato de Intercambio de datos NFC)

ISO. International Standard Organization (Organización Internacional de Estándares)

IEC. Reglas Internacionales para el Análisis de las Semillas.

FDA. Federal

OSHA. Organización de las Naciones Unidas para la Agricultura y la Alimentación.

OMS. Organización Mundial de la Salud

MSP. Ministerio de Salud Pública – Ecuador.

IoMT. Internet of Medical Things (Internet de las Cosas Médicas)

IRM. Interno Rotativo de Medicina.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

En el primer capítulo del presente proyecto de titulación, se justificará el porqué del estudio e investigación del tema propuesto fundamentando el análisis desde la definición del problema y su alcance, hasta la justificación de este como se ha desarrollado en el anteproyecto de titulación.

1.1 Problema de investigación.

Un punto débil del sistema de salud pública ha sido el aprendizaje, uso y manejo de las tecnologías existentes para poder agilitar y facilitar los procesos de salud preventiva, consulta externa y triaje de emergencias. Varias técnicas y modelos de gestión de la información han estado desarrollándose e implementándose para conseguir un orden en estos procesos y de esta manera flexibilizarlos, teniendo un acceso más confiable a esta información. (Ministerio de Salud Pública, 2015).

Los procesos de triaje de emergencia en la región han estado siendo manejados a través del uso de la metodología SHORT como se lo describe en (Gómez & Rivera, 2019), siendo un proceso el cual ha presentado muchas inconformidades en la sociedad imbabureña, dado a que los métodos utilizados, si bien tienden a ser oportunos, la manera en que se los está ejecutando ya en las situaciones que presentan los pacientes en emergencias no son ágiles. Minutos, inclusive segundos serían valiosos al momento de agilizar este tipo de procedimientos sin cambiar la metodología, más bien utilizando las nuevas tecnologías las cuales permiten a los personeros de la salud, ganar tiempo en el cual se puede salvar una vida.

De acuerdo con (Ministerio de Salud Pública, 2015), el sistema de triaje Manchester:

"El proceso de asignación de la prioridad cuenta con las siguientes acciones que no deben llevar más de 5 minutos: Primero, Realizar una evaluación rápida de vía aérea, ventilación y circulación; Segundo, motivo de la urgencia, debe ser una anamnesis dirigida, específica que identifique de manera oportuna e inmediata la causa principal de solicitud de consulta, conforme a lo referido por el paciente y los hallazgos clínicos en el primer contacto, se asigna la prioridad del paciente; Tercero, Evaluar los signos vitales del paciente; Cuarto, asignación del área/box/sala de tratamiento dentro del servicio de urgencias de acuerdo a la prioridad del paciente y /o consultorio en caso de no ser urgencia/emergencia".

Acceder a la información personal del paciente, así como sus signos vitales, resulta en ciertos procesos un retardo al momento de registrar a los pacientes, tiempo en él cual puede ponerse en riesgo la vida de una persona. Las tecnologías actuales pueden agilizar de cierta manera el tiempo de demora de estos procesos y así poder reducir de manera notable el retardo en los mismos. La tecnología NFC, al ser de corto alcance, permite que la transmisión y recepción de datos que no demandan grandes capacidades de transmisión se la pueda hacer de manera fiable, (Marcus A. et al, 2009) y de esta manera el acceso a la información sea de una manera más ágil, confiable e íntegra.

Por esa razón, en la actualidad, existen diversos hospitales y clínicas internacionales que están adoptando este tipo de tecnologías para poder disminuir el retardo de respuesta en los procesos de triaje de emergencias, entre las cuales se encuentran, el uso de dispositivos subcutáneos, que no son invasivos tal como se muestra

en (Shafeie S. et al, 2022) e implantes microchip como se detalla en (Schiffman A. et al, 2020) con diferentes tecnologías con la información necesaria y sin redundancia que cada centro hospitalario necesita para cada uno de sus procesos. Esto abre un camino para que, en el Ecuador, se pueda implementar este tipo de tecnologías y reducir los retardos provocados por el aumento de demanda de la salud pública (MSP, 2020), en especial en la ciudad de Ibarra, para ser pionera en el uso de las de metodologías de acceso a la información.

El uso de la tecnología NFC en implantes subcutáneos ofrece una nueva alternativa para la respuesta temprana ante el triaje de emergencias en la localidad, dado a su rapidez y eficiencia en el análisis e interpretación de los datos y como éstos pueden ser mostrados al usuario. Un implante subcutáneo tiene la capacidad de permanecer dentro del cuerpo humano durante muchos años sin generar alguna afección en la piel o dentro del organismo. De esta manera se puede agilizar el acceso a la información personal de los pacientes.

Es así que surge la interrogante, una vez analizados los precedentes antes mencionados: ¿Cómo se puede disminuir el tiempo de demora en los procesos de triaje para pacientes que ingresan a la sala de emergencias, considerando la problemática existente en el retardo de registro de datos personales y toma de signos vitales?, una muy fiable solución es la que plantea la investigación propuesta, el uso de la tecnología NFC para la recolección de información a través de una aplicación móvil desarrollada con una metodología de análisis-prueba-error.

1.2 Justificación

Nuevas tecnologías están empezando a incursionar diferentes áreas de estudio como lo es la medicina. Este avance tecnológico permite a la carrera de Telecomunicaciones jugar un papel de mayor importancia para manejar, regular y controlar todas las características que ahora estos procesos están desarrollándose en ambientes tan delicados como lo es la salud, por eso la importancia de conocer, analizar y desarrollar aplicativos para usuarios finales debe tener la responsabilidad crucial que conlleva tener el proceso de triaje en manos de un aplicativo.

A su vez el Objetivo 6 del Plan de Creación de Oportunidades nos dictamina que: "Garantizar el derecho a la salud integral, gratuita y de calidad, impulsando el desarrollo permanente de la ciencia e investigación tecnológica", y la política 6.5 "Modernizar el sistema de salud pública para garantizar servicios de calidad con eficiencia y transparencia", motivo por el cual el desarrollo de ésta investigación impulsará el inicio del desarrollo y conocimiento de implantes tecnológicos para ser utilizado en la salud ecuatoriana, así también abrir campo al ámbito investigativo que este tipo de tecnología conlleva en cuanto al desarrollo tecnológico del norte del país.

Con el desarrollo de este proyecto se abren nuevos caminos en el diseño de dispositivos para de transferencia de datos como es la NFC y se ofrece una contribución sustancial de conocimientos en el área. (Horikoshi H., et al, 2018) Este aporte pretende dar a la ingeniería en telecomunicaciones un nuevo campo innovador y siga avanzando en el desarrollo de conocimientos sobre esta tecnología y sus aplicaciones en el mundo los implantes subcutáneos de comunicación de campo cercano (Vikram, 2016).

1.3. Objetivos

1.3.1. Objetivo General

Implementar un sistema basado en Android que permita la lectura de datos a través de la tecnología NFC del chip implante X3 NFC, para disminuir los tiempos de triaje en emergencias en la clínica de simulación de la Universidad Técnica del Norte.

1.3.2 Objetivos Específicos

- Estudiar a detalle la literatura disponible acerca de las tecnologías NFC para su análisis en el desarrollo de tecnologías médicas, así como la convergencia de esta con los elementos de medicina involucrados en los tiempos de demora de los procesos de triaje en dependencias públicas de salud.
- Definir parámetros y características del sistema que garanticen el acceso a la información evitando la redundancia en los datos para estructurar los procesos de envío y recolección de datos del sistema.
- Desplegar el sistema basándose en la unión de la metodología XP (eXtreme Programming) y la metodología de desarrollo de aplicaciones móviles, realizando pruebas de escritorio de cada una de las etapas.
- Ejecutar pruebas del aplicativo con las personas afines al campo de la salud. para contrastar los tiempos con el uso del aplicativo, creando ambientes controlados para la comprobación del correcto funcionamiento.

CAPÍTULO II

MARCO TEÓRICO

En este capítulo se abordarán todos los conceptos y fundamentación teórica acerca de la tecnología NFC, sus características, su historia y así también su funcionamiento basado en las normas ISO/IEC 18092 la cual rige la manera en la que la tecnología NFC realiza el intercambio de información entre dos dispositivos NFC o también llamados tags, así también se explica la manera en la que se puede tener una convergencia técnica con el chip implante NFC X3, como propuesta tecnológica para agilizar los tiempos de triaje de emergencias en las dependencias medicas de la localidad.

2.1. Tecnología NFC

RFID son las siglas de Radio-Frequency Identification, que en español significa Identificación por Radiofrecuencia. Es una tecnología de identificación automática que utiliza ondas de radio para transmitir información entre una etiqueta (tag) y un lector (reader).

El sistema RFID consta de una etiqueta (tag) que contiene un microchip y una antena, y un lector (reader) que emite una señal de radiofrecuencia que es recibida por la antena de la etiqueta. Esta señal de radiofrecuencia energiza el chip de la etiqueta, lo que permite que la etiqueta responda al lector y transmita información almacenada en el chip, como números de serie, códigos de barras, entre otros.

RFID (Identificación por Radiofrecuencia) y NFC (Comunicación de Campo Cercano) son tecnologías inalámbricas que se utilizan para comunicar información entre dispositivos. Ambas tecnologías utilizan ondas electromagnéticas para transmitir datos, pero existen diferencias clave en la forma en que se utilizan y en sus capacidades.

Una diferencia importante entre RFID y NFC es que RFID se refiere a un sistema de identificación de objetos que utiliza etiquetas RFID, que pueden ser activas o pasivas, mientras que NFC se refiere a una tecnología de comunicación inalámbrica de corto alcance que utiliza el estándar ISO/IEC 18092 (Anexo 1).

En el contexto de RFID, un elemento activo es una etiqueta que tiene su propia fuente de alimentación, como una batería, y puede transmitir datos sin necesidad de recibir energía de un lector. Por otro lado, una etiqueta pasiva no tiene su propia fuente de alimentación y depende de la energía suministrada por el lector RFID para transmitir datos.

En el caso de NFC, todos los dispositivos son pasivos, lo que significa que dependen de la energía suministrada por un lector NFC para poder funcionar. Además, la distancia de transmisión de NFC es mucho más corta que la de RFID, lo que hace que la tecnología sea ideal para aplicaciones en las que se requiere un alto nivel de seguridad y privacidad.

NFC utiliza un chip que puede actuar como un emisor y un receptor de información, y se puede integrar en una variedad de dispositivos, como teléfonos móviles, tarjetas inteligentes, etiquetas y otros dispositivos electrónicos. La tecnología NFC

permite la transferencia de datos, como textos, imágenes, audio y video, entre dos dispositivos que estén a una distancia de hasta 10 cm (Faudzi N., et al, 2020).

NFC es considerada segura, ya que utiliza métodos de autenticación y cifrado para proteger la privacidad de los usuarios y evitar la interferencia de dispositivos no autorizados. Esta tecnología utiliza un proceso de autenticación que implica la verificación de la identidad de los dispositivos y los usuarios involucrados antes de que se realice cualquier transferencia de datos.

2.1.1. Historia de la Tecnología NFC

La tecnología NFC ha experimentado un desarrollo significativo desde su origen en la década de 2000 hasta la actualidad. El desarrollo de la tecnología ha sido impulsado por una creciente demanda de soluciones de comunicación inalámbrica que sean seguras, rápidas y fáciles de usar. La tecnología NFC ha demostrado ser una respuesta efectiva a esta demanda, ofreciendo una solución inalámbrica segura y versátil que se adapta a una amplia variedad de aplicaciones.

La introducción de los primeros estándares de NFC por parte de Philips y Sony en 2002 fue un paso importante en el desarrollo de la tecnología. Estos estándares sentaron las bases para la creación de un sistema de comunicación inalámbrica estándar que permitiría la comunicación entre diferentes dispositivos NFC en todo el mundo. (Singh N., 2020)

La estandarización de la tecnología NFC por parte de la ISO en 2003 fue otro hito importante en el desarrollo de la tecnología. La estandarización permitió una mayor

compatibilidad entre diferentes dispositivos NFC, lo que ayudó a impulsar la adopción de la tecnología en una variedad de aplicaciones y mercados.

La norma ISO/IEC 18092 para la tecnología de comunicación de campo cercano (NFC) fue desarrollada por el comité conjunto de la ISO/IEC para estandarizar el uso de NFC en todo el mundo. El desarrollo de la norma fue motivado por la necesidad de un estándar común que permita la interoperabilidad entre diferentes dispositivos NFC y la facilidad de uso por parte de los usuarios finales. (Anaya-Cantellán A., et al, 2014)

El proceso de desarrollo de la norma ISO/IEC 18092 comenzó en el año 2002, cuando el comité ISO/IEC JTC 1/SC 17/WG 8 (conocido como "WG 8") fue creado para abordar la necesidad de una norma internacional para la tecnología NFC. La norma se publicó en 2004 bajo el título "Near Field Communication Interface and Protocol-1 (NFCIP-1)".

En 2006, se publicó una actualización de la norma, denominada "Near Field Communication Interface and Protocol-2 (NFCIP-2)", que incluía mejoras en la seguridad y el soporte para la transmisión de datos a una velocidad más alta.

En 2011, la norma ISO/IEC 18092 se actualizó y se incorporó en la norma ISO/IEC 14443 para formar la norma ISO/IEC 18000-3, que incluye los estándares para la comunicación inalámbrica de corto alcance. Esta norma define la comunicación por radiofrecuencia de 13,56 MHz para la identificación de objetos y sistemas similares a través de etiquetas y dispositivos sin contacto.

Otro estándar importante en el ecosistema NFC, es el NFC Fórum, una organización sin fines de lucro que promueve la tecnología NFC y desarrolla

especificaciones adicionales para mejorar la interoperabilidad entre diferentes dispositivos y aplicaciones NFC.

El NFC Fórum ha desarrollado varios estándares de datos adicionales, como el protocolo de intercambio de datos NFC (NDEF, por sus siglas en inglés), que define cómo se deben estructurar y codificar los datos en una etiqueta NFC o en un mensaje NFC entre dos dispositivos. NDEF también define tipos de registros para diversos tipos de datos, como texto, URL, contacto, etc.

Una de las principales tareas del NFC Forum es desarrollar y mantener especificaciones técnicas para el uso de la tecnología NFC en diferentes contextos, como pagos móviles, transporte público, identificación y autenticación, entre otros. Estas especificaciones se basan en las normas ISO/IEC existentes y se actualizan regularmente para garantizar la interoperabilidad y la seguridad en los sistemas NFC.

El NFC Forum ha creado varias especificaciones técnicas, entre las que se incluyen:

- NFC Data Exchange Format (NDEF): especificación para el intercambio de datos entre dispositivos NFC.
- NFC Forum Type Tag Operation: especificación para el uso de etiquetas NFC Forum Type 2 y Type 4.
- NFC Forum Type 1-4 Tag Specifications: especificaciones para el uso de etiquetas NFC en diferentes contextos.
- NFC Controller Interface (NCI): especificación para la interfaz entre el controlador NFC y el software de la aplicación.

2.1.2. Estructura de los Datos según la estandarización

La estructura de los estándares de datos NFC es esencial para garantizar la interoperabilidad y la seguridad de la comunicación inalámbrica a corta distancia mediante NFC. Cada una de las cuatro capas definidas por las normas ISO/IEC 18092 tiene una función específica en la comunicación NFC.

El modelo ISO/IEC 18092 define cuatro capas que conforman el protocolo NFC:

2.1.2.1. Capa de Interfaz de Radiofrecuencia (RF):

Esta capa es la que se encarga de la transmisión de la señal de radiofrecuencia entre los dispositivos NFC. Define los parámetros técnicos de la transmisión, como la frecuencia de operación y la modulación. La Capa de Interfaz de Radiofrecuencia (RF) es la primera capa del modelo ISO/IEC 18092 y se encarga de la transmisión de la señal de radiofrecuencia entre los dispositivos NFC. Esta capa define los parámetros técnicos de la transmisión, tales como la frecuencia de operación y la modulación. La transmisión se realiza mediante una señal de radio de alta frecuencia (13,56 MHz) que puede transmitir datos hasta una distancia de 10 cm. La Capa RF también incluye la configuración de los modos de operación (lector/etiqueta o peer-to-peer) y la negociación de parámetros de comunicación, también se encarga de controlar la potencia de transmisión, la tasa de bits, la codificación y la modulación para asegurar que los dispositivos NFC puedan comunicarse de manera efectiva. Además, también incluye la gestión de la seguridad y privacidad de los datos transmitidos, mediante el uso de técnicas de cifrado y autenticación para garantizar la integridad de los datos transmitidos. En

resumen, la capa de Interfaz de Radiofrecuencia es la encargada de garantizar que la comunicación entre dispositivos NFC se realice de manera fiable y segura.

2.1.2.2. Capa de Enlace de Datos (LLC):

Esta capa establece la conexión entre dos dispositivos NFC y controla el flujo de datos entre ellos. Define el formato y la estructura de los paquetes de datos y proporciona servicios de control de errores y retransmisión de paquetes.

La capa de Enlace de Datos (LLC) es la segunda capa del protocolo NFC y se encarga de establecer la conexión entre dos dispositivos NFC y controlar el flujo de datos entre ellos. Esta capa define el formato y la estructura de los paquetes de datos y proporciona servicios de control de errores y retransmisión de paquetes para garantizar la fiabilidad de la transmisión de datos.

En la capa LLC, los dispositivos NFC utilizan el protocolo de control de enlace lógico (LLCP) para establecer una conexión de comunicación y transferir datos. El LLCP se encarga de la autenticación, el cifrado y la compresión de datos, así como del control de flujo y la gestión de errores. También proporciona servicios de capa de sesión para garantizar la integridad de los datos y la continuidad de la conexión.

2.1.2.3. Formato de Trama

El formato de trama en la capa LLC se define mediante el protocolo de comunicación de datos NFC (NCDP), que establece la estructura y el contenido de los campos de datos, así como los mecanismos de control de errores y la retransmisión de paquetes. En general, la capa LLC proporciona una conexión de comunicación fiable y eficiente entre los dispositivos NFC, lo que permite la transmisión de datos sin errores y en tiempo real.

La estructura y formato de trama en esta capa consta de un encabezado, un cuerpo de datos y un campo de detección de errores. El encabezado contiene información como la dirección de origen y destino, y la longitud del trama. El cuerpo de datos contiene la información que se está transmitiendo, mientras que el campo de detección de errores permite la verificación de errores en los datos transmitidos. La capa LLC también proporciona servicios de control de errores y retransmisión de paquetes para garantizar la integridad y confiabilidad de los datos transmitidos.

Figura 1

Formato de Trama NFC

Según la norma ISO/IEC 18092, segunda edición, Information technology − Telecommunications and information exchange between systems − Near Field Communication − Interface and Protocol (NFCIP-1), 2013, el formato de trama constará de preámbulo, SYNC, longitud, carga útil y CRC, como en la figura 1. El preámbulo tiene un mínimo de 48 bits, todos ellos ceros lógicos. El SYNC tiene 2 bytes. El primer byte

del SYNC es "B2" y el segundo byte será "4D". La longitud será un campo de 8 bits y se ajustará al número de bytes a transmitir en la carga útil más 1. El rango de longitud será de 2 a 255, y otros ajustes son RFU. La carga útil constará de n bytes de datos de 8 bits, donde n viene indicado por el número de bytes de datos. CRC puede ser calculado de acuerdo con lo que detalla la norma.

2.1.2.4. Capa de Protocolo (NFCIP-1):

Esta capa proporciona un protocolo para la transferencia de datos entre dispositivos NFC, incluyendo la negociación de los parámetros de la conexión y la definición de las funciones y servicios disponibles.

Esta capa permite que dos dispositivos NFC establezcan una conexión mediante la detección mutua de la señal de radiofrecuencia. Una vez que se ha establecido la conexión, los dispositivos negocian los parámetros de la comunicación, como la velocidad de transferencia de datos y el tipo de modulación a utilizar.

NFCIP-1 define también cómo los dispositivos NFC pueden intercambiar datos. La capa proporciona servicios para la transferencia de datos en ambos sentidos, desde el iniciador del intercambio hacia el objetivo y viceversa. Para ello, utiliza diferentes modos de comunicación, como el modo de transferencia de datos activo y el modo de transferencia de datos pasivo.

2.1.2.4.1. Modo Activo de Transferencia de datos

En el modo activo de la norma ISO/IEC 18092, el dispositivo que inicia la transferencia actúa como el "lector" y el dispositivo objetivo actúa como la "tarjeta".

El lector genera un campo electromagnético que es detectado por la antena del dispositivo objetivo, lo que hace que este último se active y envíe su respuesta al lector.

Luego, el lector envía una solicitud de conexión al dispositivo objetivo, que responde con los parámetros de conexión, como la velocidad de transmisión y el número de bloques a transferir. El lector también puede enviar una solicitud para que el dispositivo objetivo envíe una lista de servicios que ofrece.

Una vez establecida la conexión, los dispositivos pueden intercambiar datos en forma de mensajes. Cada mensaje se compone de un encabezado y una carga útil, que puede ser un comando, una respuesta o un conjunto de datos. El encabezado incluye información como el tipo de mensaje, la longitud de la carga útil y los códigos de error de detección y corrección.

El lector controla el flujo de datos durante la transferencia y puede solicitar la retransmisión de datos perdidos o dañados. Cuando se completa la transferencia, el lector envía una solicitud de desconexión y el dispositivo objetivo responde con un mensaje de desconexión.

2.1.2.4.2. Modo Pasivo de Transferencia de datos

En el modo pasivo de la norma ISO/IEC 18092, el dispositivo pasivo NFC, como una etiqueta NFC, espera a ser activado por un dispositivo activo NFC, como un teléfono móvil. Cuando el dispositivo activo se acerca lo suficiente, la antena del dispositivo pasivo recibe la señal de radiofrecuencia y se induce una corriente eléctrica en la antena,

lo que alimenta el circuito del dispositivo pasivo. A continuación, el dispositivo pasivo comienza a modular la señal de radiofrecuencia y enviar datos al dispositivo activo.

El dispositivo pasivo no tiene su propio reloj interno y utiliza la frecuencia de la señal de radiofrecuencia recibida del dispositivo activo para sincronizar su transmisión de datos. El dispositivo pasivo también está limitado en la cantidad de energía que puede recibir, lo que significa que su capacidad de transmisión de datos es más limitada que la del dispositivo activo.

Además, NFCIP-1 define un conjunto de protocolos para el intercambio de datos entre dispositivos NFC. Estos protocolos incluyen el protocolo de inicialización de conexión, el protocolo de desconexión y el protocolo de intercambio de datos. Estos protocolos permiten que los dispositivos NFC se comuniquen de manera eficiente y establezcan una conexión segura y fiable.

2.1.2.5. Capa de Aplicación (NDEF):

La capa de aplicación en la norma ISO/IEC 18092 es la capa superior del protocolo NFC y es responsable de la definición del formato de los datos que se intercambian entre los dispositivos NFC, así como de proporcionar un marco para el desarrollo de aplicaciones NFC.

Esta capa se basa en el formato de datos llamado NDEF (NFC Data Exchange Format), que define una estructura para los mensajes intercambiados entre dispositivos NFC. Los mensajes NDEF pueden contener diferentes tipos de registros, como textos, URLs, imágenes, entre otros. Cada registro NDEF está compuesto por un encabezado que

indica el tipo de registro y su longitud, seguido de los datos del registro propiamente dicho.

Además de definir el formato de los datos, la capa de aplicación también proporciona un conjunto de servicios y comandos que permiten a las aplicaciones NFC intercambiar información de manera más eficiente. Algunos de los servicios más comunes incluyen la escritura y lectura de registros NDEF, la creación de mensajes NDEF y la negociación de los servicios y protocolos disponibles en los dispositivos NFC conectados.

2.1.2.5.1. NDEF (NFC Data Exchange Format)

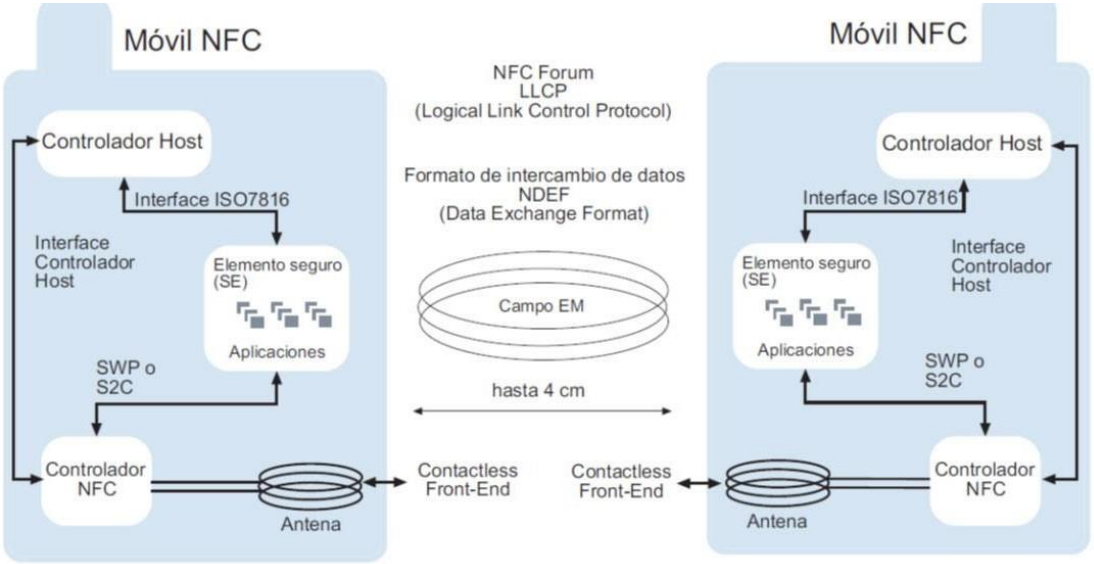
NDEF es un formato de intercambio de datos utilizado en el protocolo NFC. Se trata de un formato de datos binarios que define la estructura de los registros que se intercambian entre dispositivos NFC. Cada registro NDEF contiene información estructurada que puede incluir texto, URLs, datos de contacto, información de eventos, entre otros. Los registros NDEF se pueden codificar en diferentes tipos de mensajes, como mensajes de texto, mensajes URI, mensajes de conectividad, entre otros.

El formato NDEF es compatible con la mayoría de los dispositivos NFC y es ampliamente utilizado en aplicaciones NFC. La especificación de NDEF permite a los desarrolladores crear aplicaciones NFC que sean compatibles con una amplia variedad de dispositivos y plataformas. Además, la compatibilidad con el formato NDEF permite a los dispositivos NFC intercambiar información estructurada de manera estandarizada, lo que facilita la interoperabilidad entre dispositivos y aplicaciones NFC.

La estructura de los estándares de datos NFC es esencial para garantizar la interoperabilidad y la seguridad de la comunicación NFC entre diferentes dispositivos y aplicaciones. Cada una de las cuatro capas definidas por las normas ISO/IEC 18092 tiene una función específica en la comunicación NFC, desde la definición de las características técnicas de la comunicación hasta la estructuración y codificación de los datos transmitidos. (Hameed S., et al, 2015)

Figura 2

Arquitectura del intercambio de información en NFC



Nota: El gráfico representa cada una de las capas que la tecnología NFC utiliza para el intercambio de la información, al ser una tecnología de transmisión de datos tipo bidireccional, utiliza un formato de intercambio de datos NDEF. Tomado de *Conocimiento FQ Ingeniería Electrónica*, enero 2015 por FQIE.

2.1.3. Características Técnicas de la Tecnología NFC

La tecnología NFC, como ya se mencionó, utiliza una frecuencia de operación de 13,56 MHz y dos tipos de modulación, la modulación de amplitud en carga (ASK) y la modulación de fase en carga (PSK), lo que permite una comunicación inalámbrica de corto alcance entre dispositivos. Esta tecnología tiene una velocidad de transmisión de datos que varía de 106 kbps a 424 kbps, lo que la hace adecuada para la transferencia de datos de tamaño pequeño a mediano.

La distancia de operación de NFC es típicamente de menos de 10 cm, lo que la hace una tecnología de comunicación de corto alcance, pero suficiente para la mayoría de las aplicaciones que utiliza. Además, esta tecnología soporta diferentes tipos de datos, tales como texto, URL, imágenes, audio y vídeo, y se almacenan en un formato llamado Registro de datos NFC (NDEF).

En términos de seguridad, la tecnología NFC utiliza protocolos de autenticación y encriptación para proporcionar seguridad en la transferencia de datos. El protocolo de autenticación mutua (MIFARE) se utiliza para autenticar los dispositivos y prevenir el acceso no autorizado a los datos. También, es compatible con otros estándares ISO como ISO/IEC 14443A/B, ISO 15693 y ISO/IEC 18092, lo que permite su integración en diferentes dispositivos y sistemas operativos.

La tecnología NFC tiene un bajo consumo de energía, lo que la hace adecuada para dispositivos móviles. Además, utiliza una tecnología de detección de campos electromagnéticos para activar la comunicación, lo que reduce aún más el consumo de energía.

La tecnología NFC es una tecnología inalámbrica de corto alcance que ofrece seguridad y compatibilidad con otros estándares ISO, y su bajo consumo de energía y capacidad de soportar diferentes tipos de datos la hacen atractiva para su integración en dispositivos móviles y otros dispositivos electrónicos. Es por estas características técnicas que en la actualidad la compatibilidad de los tags NFC tiene convergencia con cualquier dispositivo que utilice esta tecnología, como es el caso del chip implante a ser analizado en esta investigación que permite mediante su bajo consumo de energía y capacidad de soportar diferentes tipos de datos.

2.2. El Chip Implante NFC X3

El chip implante X3 es un dispositivo subcutáneo de pequeñas dimensiones que se implanta bajo la piel de seres vivos con el objetivo de permitir su identificación sin necesidad de contacto físico. Este dispositivo hace uso de la tecnología NFC (Near Field Communication) para habilitar la lectura del chip de forma inalámbrica.

El chip implante X3 ha sido diseñado para ofrecer una alta seguridad y privacidad en la gestión de la información almacenada en su interior. En este sentido, cuenta con sistemas de encriptación y autenticación para garantizar que la información solo pueda ser accedida por usuarios autorizados y en momentos específicos. Además, el dispositivo cumple con la norma ISO/IEC 15693, ISO/IEC 18092 e ISO/IEC 14443 A/B, la cual establece los requisitos para la transferencia de datos a través de la tecnología NFC en aplicaciones de identificación y control de acceso.

Otra característica importante del chip implante X3 es su capacidad de resistir los efectos del entorno, como la humedad, los golpes y la temperatura. Esto permite que el

dispositivo sea utilizado en diversas condiciones y ambientes, sin que esto afecte su funcionamiento o su capacidad de almacenamiento.

El chip implante X3 se presenta en un tamaño aproximado de 2x12 mm y está recubierto de una cápsula de vidrio biocompatible, lo que le otorga resistencia a la corrosión y al desgaste. Asimismo, este dispositivo tiene una vida útil superior a 20 años y no requiere de batería para su funcionamiento, ya que se alimenta de la energía emitida por el lector NFC.

En cuanto a sus características técnicas, el chip implante X3 cumple con la norma ISO 14443A, una norma internacional que establece los requisitos para los sistemas de comunicación inalámbrica de corto alcance. Además, el dispositivo cuenta con una capacidad de almacenamiento de 880 bytes de datos, lo que permite almacenar información personal relevante, como el nombre, dirección, historial médico, entre otros datos.

El chip implante X3 cuenta con un número de identificación único y permanente, lo que lo convierte en una herramienta eficaz para la identificación de personas y animales. En este sentido, resulta ideal para aplicaciones de seguimiento y control, como en el caso de los sistemas de respuesta temprana previa del triaje de emergencias.

Cabe destacar que, aunque el chip implante X3 ha sido diseñado originalmente para ser utilizado en animales, su uso en humanos ha sido objeto de debate y controversia. A pesar de ello, se ha demostrado que este dispositivo puede ser utilizado de forma segura y efectiva en seres humanos, por ejemplo, en aplicaciones de identificación de pacientes en emergencias médicas.

El chip implante X3 es un dispositivo subcutáneo que ofrece una alta seguridad y privacidad en la gestión de información, resistencia a diversos entornos y cumplimiento de normas internacionales. Si bien su uso en humanos ha sido objeto de controversia, su aplicación en animales y en sistemas de identificación de pacientes en emergencias médicas puede resultar altamente efectiva y beneficiosa.

2.2.1. Características Técnicas Chip Implante X3

Tabla 1

Características técnicas generales del chip implante NFC X3

Característica	**Descripción**
Fabricante	I am Robot
Modelo	x3
Tipo de chip	NFC (Near Field Communication)
Frecuencia	13,56 MHz
Memoria	1 KB de memoria EEPROM
Velocidad de transmisión	106 kbps
Protocolos compatibles	ISO 14443A
Modos de operación	Lectura y escritura
Temperatura de operación	-40°C a +85°C

Vida útil	Más de 10 años
Dimensiones	2 x 12 mm
Material de encapsulación	Vidrio biocompatible

Nota: Esta tabla muestra los valores típicos técnicos que el chip implante X3 posee de

manera general, dado a las diferentes versiones del chip algunos parámetros pueden variar dependiendo de la necesidad del usuario y de la disponibilidad del dispositivo en I am Robot. Tomado de *S. Becker, I am Robot*, diciembre 2022.

2.3. Estado del Arte de NFC en el ámbito médico

La tecnología de comunicación de campo cercano (NFC, por sus siglas en inglés) es una tecnología inalámbrica de corto alcance que permite la transferencia de datos entre dispositivos cercanos. El uso de la tecnología NFC en el ámbito médico ha demostrado ser beneficioso en varias aplicaciones, incluyendo el triaje de emergencias. La literatura existente sobre el uso de la tecnología NFC en el ámbito médico y su relación con el triaje de emergencias muestra que su utilización puede mejorar la eficiencia y precisión del triaje, permitiendo una respuesta temprana y efectiva en situaciones de emergencia.

Uno de los principales beneficios de la utilización de dispositivos NFC en el ámbito médico es la capacidad de transferir información de manera rápida y precisa. En situaciones de emergencia, esto puede ser crucial para la identificación rápida y precisa de los pacientes y la asignación de recursos adecuados. Además, la tecnología NFC permite la transferencia de información en tiempo real, lo que facilita la colaboración entre médicos y personal de emergencias.

Un estudio llevado a cabo por (Li Y., et al.,2015) evaluó la efectividad de un sistema de triaje basado en NFC en un hospital de China. Los autores encontraron que la utilización de la tecnología NFC permitió una clasificación más precisa de los pacientes en comparación con el método tradicional de triaje. Además, el sistema de NFC permitió una transferencia de información más rápida y precisa entre los médicos y el personal de emergencias, lo que contribuyó a una atención médica más efectiva.

En otro estudio, (Kothari N., et al., 2012), se utilizó un sistema de identificación basado en NFC para mejorar la eficiencia del triaje en situaciones de emergencia. Los autores encontraron que la utilización de la tecnología NFC permitió una identificación más rápida y precisa de los pacientes, lo que permitió una respuesta temprana y efectiva en situaciones de emergencia.

Además de la transferencia de información, otro beneficio de la utilización de la tecnología NFC en el ámbito médico es la capacidad de acceder a información médica importante de forma rápida y sencilla. Por ejemplo, los pacientes pueden llevar consigo chips implantes NFC que contienen información médica importante, como alergias y enfermedades crónicas. En situaciones de emergencia, los médicos pueden acceder rápidamente a esta información, lo que puede ser crucial para tomar decisiones de tratamiento efectivas.

Sin embargo, el uso de chips implantes NFC en pacientes ha sido objeto de debate y controversia. Uno de los principales problemas es la seguridad de la información almacenada en los chips. La información almacenada en un chip implante NFC es accesible mediante la tecnología NFC, lo que plantea preocupaciones sobre la privacidad

y seguridad de la información médica. Además, el uso de chips implantes NFC puede plantear cuestiones éticas, como el consentimiento informado y el riesgo potencial de efectos secundarios.

Por su parte, el uso de chips implantes NFC en pacientes ha sido objeto de debate y controversia. Sin embargo, algunos estudios han demostrado su eficacia en situaciones de emergencia. En este sentido, (Carlberg M, et al., 2016) evaluó el uso de chips implantes NFC en pacientes con enfermedades crónicas en Finlandia. Los autores encontraron que la utilización de estos dispositivos permitió una identificación más rápida y precisa de los pacientes, lo que contribuyó a una atención médica más efectiva en situaciones de emergencia.

Además, el estudio de (Strömmer E., et al., 2006) examinó la aplicabilidad de la tecnología NFC en el monitoreo de pacientes con enfermedades crónicas. Los autores diseñaron una plataforma que utiliza etiquetas NFC para almacenar y transmitir información médica de los pacientes, como los resultados de pruebas de laboratorio y las citas médicas programadas. La plataforma permitió que los pacientes compartieran su información médica con sus proveedores de atención médica y, al mismo tiempo, les permitió mantener un mayor control sobre su propia información médica.

Por otro lado, el estudio de (Cruz Ángel, B., 2021). analizó la aplicabilidad de la tecnología NFC en el seguimiento de pacientes postoperatorios. Los autores utilizaron etiquetas NFC para recopilar información sobre la evolución de la cicatrización de las heridas quirúrgicas en pacientes que habían sido sometidos a cirugía. La información se transmitió a través de NFC a un sistema de seguimiento en línea que permitió a los

médicos monitorear la evolución de la cicatrización de las heridas y tomar medidas en caso de que se detectaran problemas.

En lo que respecta al triaje de emergencias, varios estudios han explorado el uso de la tecnología NFC para mejorar el proceso de triaje y reducir el tiempo necesario para identificar y tratar a los pacientes con mayor urgencia. Es así como (Hernández-Gota C, et al., 2014) propuso un sistema basado en NFC que permitía a los servicios de emergencia acceder a la información médica de los pacientes de forma rápida y eficiente. Los autores sugirieron que el sistema podría mejorar la eficiencia del proceso de triaje, lo que a su vez podría reducir el tiempo necesario para identificar y tratar a los pacientes con mayor urgencia.

Asimismo, (Sánchez-García, J. et al., 2021) evaluó el uso de la tecnología NFC en el triaje de emergencias para pacientes con enfermedades crónicas. Los autores propusieron un sistema de triaje basado en NFC que permitía a los pacientes con enfermedades crónicas llevar consigo su información médica en todo momento y compartirla con los servicios de emergencia en caso de una emergencia médica. Los autores sugirieron que el sistema podría mejorar la calidad de la atención médica y reducir el tiempo necesario para identificar y tratar a los pacientes con mayor urgencia.

La literatura existente sugiere que la tecnología NFC tiene un gran potencial para mejorar el proceso de atención médica, especialmente en el ámbito del monitoreo de pacientes crónicos y el triaje de emergencias. La tecnología NFC permite el almacenamiento y transmisión de información médica de forma rápida y eficiente, lo que

puede mejorar la eficiencia y calidad de la atención médica y reducir el tiempo necesario para identificar y tratar a los pacientes con mayor urgencia.

En particular, en el contexto de un sistema de triaje de emergencias, la tecnología NFC puede ayudar a los servicios de emergencia a acceder rápidamente a la información médica de los pacientes, lo que a su vez puede mejorar la eficiencia del proceso de triaje y reducir el tiempo necesario para identificar y tratar a los pacientes con mayor urgencia. Sin embargo, es importante tener en cuenta que la implementación de la tecnología NFC en el ámbito médico plantea desafíos importantes en términos de discrepancia en cuanto al debate existente en poder implantarse ciertos artefactos electrónicos o gadgets los cuales se puede llegar a la discusión acerca si son invasivos o no y como influyen en la salud de cada persona.

La tecnología NFC ha demostrado ser útil en el monitoreo de pacientes con enfermedades crónicas, permitiendo la transmisión inalámbrica de datos médicos y el seguimiento remoto de los pacientes. Por ejemplo, en un estudio realizado en China, se implementó un sistema de monitoreo remoto de pacientes con diabetes utilizando etiquetas NFC en las tarjetas de identificación de los pacientes. Los datos de glucemia y otros indicadores fueron leídos por los médicos a través de dispositivos móviles equipados con tecnología NFC, lo que permitió un mejor seguimiento y control de la enfermedad (Cao Z., et al., 2019).

En el ámbito de las emergencias médicas, la tecnología NFC se ha utilizado para la identificación y el seguimiento de pacientes en situaciones de desastre. En un estudio realizado en Japón, se implementó un sistema de triaje de emergencias utilizando

etiquetas NFC en las pulseras de los pacientes. La información de los pacientes se registró en un servidor central y los médicos pudieron acceder a ella en tiempo real a través de dispositivos móviles equipados con tecnología NFC. Este sistema mejoró la eficiencia del triaje de emergencias y la atención a los pacientes (Kim H., et al., 2017).

En otro estudio, se desarrolló un sistema de identificación y seguimiento de pacientes utilizando etiquetas NFC en las tarjetas de identificación de los pacientes. Los datos médicos de los pacientes se almacenaron en la etiqueta NFC, lo que permitió a los médicos acceder a la información en tiempo real y tomar decisiones clínicas informadas en situaciones de emergencia. Además, este sistema permitió el seguimiento de los pacientes después del alta hospitalaria y el monitoreo remoto de su salud (Aoki R., et al., 2015).

La tecnología NFC tiene un gran potencial en el ámbito médico y puede mejorar significativamente la eficiencia y la calidad de la atención médica. En el contexto de las emergencias médicas, la tecnología NFC puede mejorar el triaje y la atención de los pacientes al permitir una identificación rápida y precisa de los pacientes y el acceso a su información médica en tiempo real. Sin embargo, se necesitan más estudios y desarrollos tecnológicos para optimizar su uso en el ámbito médico y garantizar la seguridad y privacidad de los datos de los pacientes.

2.4. Historia y situación actual de los chips subcutáneos

La tecnología NFC (Near Field Communication) ha revolucionado diversos aspectos de la vida cotidiana, y una de sus aplicaciones más innovadoras es la integración en implantes subcutáneos. Este marco teórico rastrea la evolución de los

implantes NFC, desde sus primeros intentos en animales hasta las soluciones avanzadas y seguras disponibles en la actualidad, como el chip implante NFC X3 de I am Robot. (Sevilla V., et al., 2016)

- **Los Primeros Experimentos en Animales:** En las primeras etapas del desarrollo de los implantes NFC, los experimentos se centraron en animales para evaluar la viabilidad de la tecnología. Investigadores y científicos comenzaron a implantar chips NFC en mascotas, como perros y gatos, para rastrear su ubicación y proporcionar datos de salud. Estos experimentos iniciales establecieron las bases para futuras aplicaciones de implantes subcutáneos en humanos.
- **Avances en la Salud y el Monitoreo Médico:** A medida que la tecnología NFC evolucionó, los implantes subcutáneos comenzaron a desempeñar un papel crucial en el campo de la salud y el monitoreo médico. Los primeros intentos se centraron en la monitorización de signos vitales, como la temperatura y la frecuencia cardíaca. Con el tiempo, estos implantes avanzaron para medir niveles de glucosa, administrar medicamentos y monitorear enfermedades crónicas.
- **Seguridad y Privacidad en la Era Digital:** Con la creciente adopción de implantes NFC en aplicaciones médicas y de salud, surgió la preocupación por la seguridad y la privacidad. A medida que más personas consideraban la posibilidad de implantar estos dispositivos, se implementaron medidas de seguridad más sólidas, como la encriptación de datos y la autenticación, para proteger la información del usuario.
- **Implantes Subcutáneos en la Actualidad:** Hoy en día, los implantes subcutáneos con tecnología NFC han evolucionado considerablemente. El chip

implante NFC X3 de I am Robot representa un hito en esta evolución. Ofrece una amplia gama de aplicaciones, desde el acceso seguro a dispositivos y sistemas hasta el monitoreo avanzado de la salud. La miniaturización y la eficiencia energética permiten una vida útil prolongada, y los avances en la interfaz con aplicaciones móviles han mejorado la facilidad de uso y la conectividad.

- **Perspectivas Futuras:** El campo de los implantes subcutáneos con tecnología NFC continúa avanzando a un ritmo rápido. Se espera que estos dispositivos desempeñen un papel aún más importante en la salud, la seguridad y la comodidad en el futuro. Las perspectivas incluyen aplicaciones más amplias en la atención médica, la seguridad y la interacción con la tecnología. (Banafa A., 2021)

2.5. Softwares Existentes Para El Desarrollo De Aplicaciones Android

La creación de aplicaciones móviles ha tomado gran relevancia en la actualidad, especialmente en el ámbito de Android, ya que se trata del sistema operativo más utilizado en dispositivos móviles en todo el mundo. Por ello, es importante conocer los distintos lenguajes de programación y softwares existentes para el desarrollo de aplicaciones Android.

Uno de los softwares más conocidos para el desarrollo de aplicaciones Android es Android Studio. Este software es desarrollado por Google y es considerado el entorno de desarrollo integrado (IDE) oficial de Android. Android Studio se basa en el lenguaje de programación Java y ofrece una gran cantidad de herramientas para el desarrollo de aplicaciones móviles, como emuladores de dispositivos, diseño de interfaces gráficas y depuración de código.

Por otra parte, un software muy utilizado para el desarrollo de aplicaciones Android es Unity. Aunque originalmente fue creado para el desarrollo de videojuegos, Unity también se ha utilizado para crear aplicaciones móviles debido a su capacidad para crear interfaces 3D y 2D altamente interactivas. Unity utiliza el lenguaje de programación C# y ofrece una amplia variedad de herramientas de desarrollo.

En ese sentido el software que se ha popularizado en los últimos años para el desarrollo de aplicaciones móviles es Flutter, creado por Google. Flutter utiliza el lenguaje de programación Dart y permite crear aplicaciones móviles tanto para Android como para iOS con un solo código base. Además, Flutter se caracteriza por ser altamente personalizable y cuenta con una amplia variedad de widgets y herramientas de diseño.

Una alternativa es el software muy utilizado para el desarrollo de aplicaciones Android es Xamarin, que permite crear aplicaciones móviles para Android e iOS utilizando el lenguaje de programación C#. Xamarin se integra perfectamente con Visual Studio, lo que permite utilizar todas las herramientas de este entorno de desarrollo para el desarrollo de aplicaciones móviles.

En relación, el software para el desarrollo de aplicaciones Android es React Native, creado por Facebook es otra alternativa. React Native utiliza el lenguaje de programación JavaScript y permite crear aplicaciones móviles para Android e iOS con un solo código base. React Native cuenta con una gran cantidad de bibliotecas y herramientas para el desarrollo de aplicaciones móviles, lo que lo convierte en una opción popular para los desarrolladores.

También se encuentra PhoneGap, un software que utiliza tecnologías web como HTML, CSS y JavaScript para crear aplicaciones móviles para múltiples plataformas, incluyendo Android. PhoneGap se caracteriza por ser fácil de utilizar y cuenta con una amplia variedad de herramientas para el desarrollo de aplicaciones móviles, como emuladores de dispositivos y herramientas de diseño.

Otro software que se ha popularizado recientemente para el desarrollo de aplicaciones Android es Kotlin Multiplatform. Kotlin Multiplatform permite crear aplicaciones móviles para múltiples plataformas, incluyendo Android, utilizando el lenguaje de programación Kotlin. Kotlin Multiplatform permite compartir una gran cantidad de código entre diferentes plataformas, lo que reduce significativamente el tiempo y los costos de desarrollo.

Por último, existe Cordova, un software que utiliza tecnologías web para crear aplicaciones móviles para múltiples plataformas, incluyendo Android. Cordova se basa en tecnologías web como HTML, CSS y JavaScript, y ofrece una amplia variedad de herramientas para el desarrollo de aplicaciones móviles, como emuladores de dispositivos y herramientas de diseño.

En definitiva, la elección del software dependerá de las necesidades específicas del desarrollador y del proyecto en cuestión. Es importante evaluar las diferentes opciones y elegir la que mejor se adapte a las necesidades del proyecto.

2.5.1. Lenguajes de Programación de Aplicaciones Android

Existen varios lenguajes de programación que se pueden utilizar para desarrollar aplicaciones Android. Uno de los más populares es Java, un lenguaje de programación

orientado a objetos que se ha utilizado durante muchos años para el desarrollo de aplicaciones Android. Java es un lenguaje flexible y ampliamente utilizado en el mundo del desarrollo de software, lo que hace que haya muchos recursos y herramientas disponibles para los desarrolladores que lo utilizan.

Otro lenguaje de programación que se ha vuelto cada vez más popular en la comunidad de desarrolladores de Android es Kotlin. Kotlin es un lenguaje de programación moderno que se integra perfectamente con Java, lo que lo convierte en una buena opción para aquellos desarrolladores que buscan una alternativa a Java. Su sintaxis es más limpia y legible que la de Java, lo que puede hacer que sea más fácil de leer y escribir código.

Además de Java y Kotlin, también se pueden utilizar otros lenguajes de programación para desarrollar aplicaciones Android. C++ es uno de estos lenguajes, y puede ser utilizado en combinación con el kit de desarrollo de software (SDK) de Android NDK. Python y JavaScript también pueden ser utilizados a través de frameworks como Kivy y React Native, respectivamente.

Es importante tener en cuenta que cada lenguaje de programación tiene sus propias ventajas y desventajas. Por ejemplo, C++ puede ser más rápido que Java en algunas operaciones, pero también puede ser más complicado de programar y depurar. Por lo tanto, es importante considerar cuidadosamente las necesidades y requerimientos específicos de la aplicación antes de elegir un lenguaje de programación.

Kotlin es un lenguaje de programación multiplataforma que se ejecuta en la Máquina Virtual de Java (JVM) y que también puede ser compilado a código nativo. Este

lenguaje fue desarrollado por JetBrains y presentado al público en 2011. Desde entonces, ha ganado popularidad debido a su legibilidad, concisión y seguridad.

Kotlin es un lenguaje de programación que ha ido ganando popularidad en los últimos años, especialmente para el desarrollo de aplicaciones para Android. Fue creado en el año 2010 por JetBrains y su primera versión estable fue lanzada en 2016. Kotlin es un lenguaje de programación multiparadigma, es decir, que soporta diferentes estilos de programación, como la programación orientada a objetos, programación funcional y programación estructurada.

La sintaxis de Kotlin es concisa y fácil de leer, lo que la hace ideal para el desarrollo de aplicaciones móviles. Además, está diseñada para ser interoperable con Java, lo que permite a los desarrolladores utilizar librerías y código Java existente en sus proyectos de Kotlin.

En cuanto a la lectura de NFC, Kotlin tiene un soporte nativo para la tecnología NFC en Android, lo que facilita la programación de aplicaciones que hacen uso de esta tecnología. Por ejemplo, para leer datos de una etiqueta NFC, se puede utilizar la clase NfcAdapter, que permite detectar etiquetas NFC y recibir notificaciones cuando se detecta una nueva etiqueta.

Kotlin también permite el uso de anotaciones para reducir la cantidad de código necesario para realizar tareas repetitivas. Por ejemplo, se puede utilizar la anotación @Parcelize para generar automáticamente el código necesario para que una clase pueda ser parcelada, lo que es útil cuando se necesita pasar objetos de una actividad a otra.

Kotlin es un lenguaje de programación moderno y versátil que se adapta bien al desarrollo de aplicaciones móviles para Android, incluyendo aquellas que utilizan tecnología NFC. Su sintaxis concisa y fácil de leer, así como su interoperabilidad con Java, lo hacen una opción atractiva para desarrolladores que buscan un lenguaje de programación eficiente y fácil de utilizar.

2.5.2. Aplicaciones Android dentro del ámbito de la medicina

Las aplicaciones móviles Android son programas informáticos diseñados para ser utilizados en dispositivos móviles que usan el sistema operativo Android. Con la popularización de los smartphones, las aplicaciones Android se han convertido en una herramienta esencial para las personas en la actualidad, ya que ofrecen una gran cantidad de servicios y funcionalidades que mejoran su calidad de vida y les permiten acceder a información y servicios de manera rápida y sencilla.

Dentro del ámbito de la medicina, las aplicaciones móviles Android han demostrado ser una herramienta útil para el seguimiento y control de diversas enfermedades crónicas, como la diabetes, la hipertensión y las enfermedades cardiovasculares. Estas aplicaciones permiten a los pacientes monitorear su estado de salud, llevar un registro de sus medicamentos y sus niveles de glucemia o presión arterial, entre otros parámetros.

En cuanto al uso de NFC en aplicaciones médicas, se ha demostrado que esta tecnología es especialmente útil para la identificación de pacientes en situaciones de emergencia. Al incorporar un chip NFC en una pulsera o en un implante, se puede acceder rápidamente a información médica relevante del paciente, como alergias, enfermedades

crónicas, medicamentos y otros datos importantes para el personal médico. De esta manera, se puede proporcionar una atención más rápida y efectiva, lo que puede ser crucial en situaciones de emergencia.

Además, la tecnología NFC también se ha utilizado para el seguimiento y control de medicamentos, a través de la incorporación de chips NFC en los envases de los medicamentos. Esto permite a los pacientes tener un control más preciso de su medicación y evitar errores de dosificación.

Las aplicaciones móviles Android y la tecnología NFC tienen un gran potencial en el ámbito de la medicina, ya sea para el monitoreo de enfermedades crónicas, la identificación de pacientes en situaciones de emergencia o el seguimiento y control de medicamentos. Su uso adecuado puede mejorar significativamente la calidad de vida de los pacientes y facilitar la labor del personal médico.

2.5.3. Desarrollo de aplicaciones NFC en la actualidad

En la actualidad, existen varias aplicaciones móviles que utilizan la tecnología NFC para la lectura de chips implantes en el ámbito médico. Estas aplicaciones tienen el objetivo de facilitar la identificación de pacientes en emergencias y optimizar los tiempos de atención en situaciones críticas.

Una de las aplicaciones más utilizadas en el ámbito médico para la lectura de chips implantes es NFC TagInfo, disponible tanto en iOS como en Android. Esta aplicación permite la lectura y análisis de chips NFC, incluyendo los chips implantes como el X3. A través de la aplicación, es posible conocer la información almacenada en el chip, como el

número de identificación del paciente, información médica relevante y datos de contacto de emergencia.

Otra aplicación móvil para la lectura de chips implantes es Medical ID, disponible solo en iOS. Esta aplicación permite la creación de una tarjeta médica digital en la que se puede almacenar información relevante sobre el paciente, incluyendo su nombre, fecha de nacimiento, grupo sanguíneo, alergias, enfermedades crónicas y contactos de emergencia. La información almacenada en la tarjeta puede ser accedida mediante la lectura del chip NFC del paciente.

Además de estas aplicaciones específicas para la lectura de chips implantes, existen otras aplicaciones que utilizan la tecnología NFC en el ámbito médico, como las aplicaciones para la gestión de medicamentos y citas médicas. Estas aplicaciones permiten la identificación rápida y eficiente de pacientes en las consultas médicas, evitando posibles errores en la administración de medicamentos y facilitando la gestión de citas.

El uso de aplicaciones móviles para la lectura de chips implantes en el ámbito médico ha sido de gran utilidad para la identificación rápida y eficiente de pacientes en emergencias y la optimización de los tiempos de atención. Además, la tecnología NFC también ha sido aplicada en otras áreas de la medicina, como la gestión de medicamentos y citas médicas, mejorando la calidad de la atención médica en general.

CAPITULO III

DISEÑO

En este capítulo se analiza los requerimientos necesarios para realizar el diseño del aplicativo en base al estado del arte propuesto y también a la metodología de desarrollo de aplicaciones en conjunción con la metodología de programación extrema, teniendo varias etapas como son: planificación, diseño, programación, pruebas y lanzamiento de software; y a su vez analizar todos los requisitos y stakeholders necesarios para cumplir con las expectativas de los usuarios tanto en software como en hardware de acuerdo a la extrapolación de la información obtenida por los potenciales usuarios del sistema y también de la literatura estudiada en el capítulo anterior utilizando los conceptos y estándares propuestos de la tecnología NFC dentro del ámbito médico.

3.1. Requerimientos del Sistema

Durante el proceso de desarrollo de este proyecto, surgieron interrogantes que proporcionaron directrices para el diseño secuencial, en concordancia con la metodología propuesta. En cada fase de la investigación, se realiza un análisis del entorno, los requerimientos y, sobre todo, se consideran los actores sociales que intervienen en el proyecto.

3.1.1. Construcción de Atributos de los requerimientos

La construcción de atributos se elabora en base a tres requerimientos que son: Stakeholders, sistema y arquitectura; y cada requerimiento debe cumplir con que sea verificable, que se pueda cumplir y que sea medible y limitable.

3.1.1.1. Nomenclatura de los Requerimientos para usarse

Con el propósito de mejorar la gestión de los requerimientos, es necesario asignar una abreviatura única a cada uno de ellos. En la Tabla 2 se presentan las abreviaturas utilizadas para referirse a los requerimientos en este proyecto, lo cual favorece una mayor organización y comprensión en el contexto del mismo.

Tabla 2. Abreviatura de los requerimientos

Requerimiento	Abreviatura
Stakeholders	STSR
Sistema	SYSR
Arquitectura	SRSH

3.1.2. Requerimientos de Stakeholders y Análisis

El propósito de establecer estos requisitos es definir los criterios necesarios para el diseño del sistema NFC, teniendo en cuenta las necesidades específicas de los usuarios en el contexto del sistema.

Para iniciar el desarrollo del sistema NFC, son importantes los requerimientos de Stakeholders ya que muestran las necesidades de los involucrados. Como se observa en la Tabla 3, los involucrados estarán ligados con las soluciones que se puedan, de tal forma que se cumplan los objetivos propuestos. Además, es necesario considerar los requerimientos tanto de hardware como de software y arquitectura para la construcción del sistema NFC. Estos aspectos jugarán un papel fundamental en el diseño y

funcionamiento del sistema, asegurando su correcta operatividad y satisfaciendo las expectativas de los usuarios involucrados.

Tabla 3. Descripción de Stakeholders

Stakeholders	**Función**
Nogales Romero José Camilo	Desarrollador del Proyecto
Ing. Hernán Mauricio Domínguez Limaico, Ms.C.	Director del Proyecto
Ing. Edgar Alberto Maya Olalla, Ms.C.	Asesor del Proyecto
Personal de la Salud	Usuario Final App/Sistema NFC

Los requerimientos se fundamentan en entrevistas realizadas a profesionales de la salud y médicos investigadores de la carrera de Medicina de la UTN. Estas entrevistas se llevaron a cabo mediante charlas, mesas redondas y sesiones de lluvia de ideas. El objetivo de estas interacciones fue obtener información precisa y perspectivas expertas para identificar las necesidades y requisitos específicos del sistema NFC para el triaje de emergencias. (Anexo 2)

Es fundamental asegurar el cumplimiento de estos requisitos para satisfacer las expectativas de los stakeholders involucrados, garantizando una conexión efectiva entre las expectativas de los usuarios. En la Tabla 4 se procede con una evaluación de los requisitos operativos y las necesidades de los usuarios

Tabla 4. Requerimientos de Stakeholders

Nomenclatura	Requerimiento	Prioridad		
		Baja	Media	Alta
STRS1	El sistema debe tener un lector NFC activo y funcional.			X
STRS2	El chip NFC debe contar con memoria de almacenamiento para guardar la información			X
STRS3	El chip NFC debe contar con normas de sellado al vacío y no ser invasivo			X
STRS4	El sistema deberá tener la capacidad de procesar la información para mostrar en la aplicación			X
STRS5	La aplicación deberá tener detallada la información del paciente que ha solicitado la lectura			X
STRS6	La aplicación deberá tener información acerca del chip que se está accediendo		X	

De acuerdo con la Tabla 4. , el análisis de requerimientos revela que se han asignado prioridades a cada uno de ellos en relación con su importancia para el sistema NFC de triaje de emergencias. Los requerimientos de alta prioridad incluyen la necesidad de un lector NFC activo y funcional, así como un chip NFC con capacidad de almacenamiento y normas de sellado al vacío no invasivas. Estos aspectos son críticos para garantizar el correcto funcionamiento y la seguridad del sistema de acuerdo con la norma ISO/IEC 18092 e ISO 21534.

Además, se ha asignado una alta prioridad a la capacidad de procesamiento de la información y su presentación en la aplicación, así como a la disponibilidad de información detallada del paciente. Por otro lado, el requerimiento de prioridad media es la inclusión de información acerca del chip en la aplicación. Si bien sigue siendo

relevante, no se considera tan crítico en comparación con otros aspectos clave del sistema. (Anexo 2)

3.1.3. Requerimientos del Sistema y Análisis

Los requerimientos se los realiza en base a las funciones y limitaciones que va a desempeñar el sistema de despliegue del aplicativo. Entre estos se va a analizar requerimientos de uso y rendimiento, análisis de las 6M's, entre otros.

Según (Gasca-Mantilla, M., et. al, 2013): "La metodología propuesta para el desarrollo de aplicaciones para móviles se fundamenta en la experiencia de investigaciones previas en aplicaciones móviles, la evaluación del potencial de éxito para servicios de tercera generación denominada 6 M, la ingeniería de software educativo con modelado orientado por objetos (ISE-OO), y principalmente en los valores de las metodologías ágiles. … De la 6 M's se extrae la concepción de que las aplicaciones móviles deben garantizar el cumplimiento de las necesidades de los usuarios y al mismo tiempo generen ingresos. La 6 M's debe su nombre a los seis atributos que se miden para evaluar el éxito del servicio propuesto: Movement (Movimiento), Moment (Momento), Me (Yo), Multi-user (Multiusuario), Money (Dinero) y Machines (Máquinas) (Ahonen, Barret y Golding, 2002)."

De esta manera, en la Tabla 5 se presenta un resumen del significado de las 6M's para el desarrollo de aplicaciones móviles

Tabla 5: Resumen de las 6M's.

Atributo	Descripción
Movimiento	Capacidad de adaptarse y responder a los movimientos y desplazamientos del usuario, proporcionando una experiencia fluida.
Momento	Ofrecer valor relevante y oportuno en el momento adecuado, comprendiendo las necesidades y preferencias del usuario en diferentes contextos.
Yo	Personalización y adaptación de la aplicación a las preferencias y características individuales del usuario para brindar una experiencia personalizada.
Multiusuario	Admitir múltiples usuarios y permitir la interacción y colaboración entre ellos, especialmente en aplicaciones sociales o de trabajo en equipo.
Dinero	Generación de ingresos a través de la aplicación móvil, mediante modelos de negocio como publicidad, compras integradas, suscripciones, etc.
Máquinas	Interacción con otros dispositivos o sistemas, como sensores o dispositivos IoT, para ampliar las funcionalidades y la integración tecnológica de la aplicación.

Los requerimientos del sistema son los que establecen los límites y las funciones que el sistema debe realizar. Estos requerimientos se derivan del planteamiento del problema descrito anteriormente y están estrechamente relacionados con las necesidades de los usuarios. En la Tabla 6 se especifica las funciones básicas que tendrá el sistema NFC

Tabla 6. Requerimientos de Sistema

Nomenclatura	Requerimiento	Prioridad		
		Baja	Media	Alta
SYSR1	Conectividad NFC de acuerdo con el NDEF			X
SYSR2	La interfaz de lectura del chip implante X3 debe captar la información del mismo de acuerdo con los parámetros descritos en la norma ISO/IEC 18092			X
SYSR3	La información recolectada debe concordar con la información existente en la base de datos del hospital o clínica donde se implementará el sistema			X
SYSR4	Debe tener compatibilidad con todas las plataformas existentes		X	
SYSR5	El sistema debe cumplir con los estándares de calidad y superar la prueba de las 6Ms para aplicaciones móviles			X
SYSR6	La aplicación deberá tener información acerca del chip que se está accediendo	X		
SYSR7	El sistema no debe saturarse y debe tener un funcionamiento correcto en cada etapa de lectura		X	

Al analizar la Tabla 6, se puede observar que se han asignado prioridades a cada uno de ellos en función de su importancia en el sistema NFC de triaje de emergencias. SYSR1, SYSR2 y SYSR3 tienen prioridad alta debido a su relevancia crítica para el correcto funcionamiento del sistema. SYSR2 se enfoca en la interfaz de lectura del chip implante X3, garantizando que cumpla con los parámetros establecidos en la norma ISO/IEC 18092. SYSR3 asegura que la información recolectada concuerde con la base de datos del hospital o clínica. SYSR1, que se refiere a la conectividad NFC de acuerdo con el NDEF, también tiene prioridad alta, ya que es fundamental para establecer una comunicación efectiva con los dispositivos NFC. De esta manera se puede tener

estándares de calidad y rendimiento muy aceptables para poner en funcionamiento el sistema por lo que es de prioridad alta superar las pruebas de las 6M's que describen el requerimiento SYSR5.

SYSR4 tienen prioridad media, ya que se centran en la calidad del sistema y su compatibilidad con diferentes plataformas, respectivamente. SYSR6 se considera de prioridad baja, ya que se refiere a la información específica del chip que se está accediendo, mientras que SYSR7 tiene prioridad media, ya que se relaciona con el correcto funcionamiento del sistema en cada etapa de lectura. En general, estas prioridades se asignan en función de la importancia crítica de cada requerimiento para garantizar un sistema NFC confiable, preciso y compatible con los estándares de calidad y funcionalidad requeridos en el contexto de triaje de emergencias.

La atención detallada a estos requerimientos del sistema y el análisis exhaustivo de las 6M's (Movimiento, Método, Material, Máquina, Medio Ambiente y Medición) en el desarrollo de aplicaciones móviles reflejan el compromiso con la calidad y la precisión en las siguientes etapas del proyecto de titulación. La elección específica del formato NDEF para la presentación de datos subraya la cuidadosa planificación en la estructuración de la información. Además, el cumplimiento estricto de normativas como ISO/IEC 18092, ISO/IEC 14443-2 y ISO/IEC 14443-3 evidencian la adhesión a estándares internacionales para garantizar la interoperabilidad y seguridad del sistema basado en NFC.

3.1.4. Requerimientos de Arquitectura

Los requerimientos de arquitectura se refieren a los componentes y necesidades de hardware y software, así como a sus características, en relación con el funcionamiento del sistema NFC. Para su determinación, se tienen en cuenta los requerimientos lógicos, de diseño, de hardware y de software. Los requerimientos de arquitectura se detallan en la Tabla 7.

Tabla 7. Requerimientos de Arquitectura

Nomenclatura	Requerimiento	Prioridad		
		Baja	Media	Alta
SRSH1	El sistema de lectura del chip implante NFC deberá someterse a pruebas antes de su lanzamiento			X
SRSH2	Comunicación con el hardware debe ser a 13,56 MHz			X
SRSH3	La información recolectada debe concordar con la información existente en la base de datos del hospital o clínica donde se implementará el sistema			X
SRSH4	Debe tener una Aplicación para conectar el chip con el Smartphone Android.			X
SRSH5	El sistema deberá tener un chip implante NFC X3 de al menos 1Kbyte de almacenamiento			X
SRSH6	El sistema debe tener un lector NFC integrado.			X
SRSH7	El sistema no debe saturarse y debe tener un funcionamiento correcto en cada etapa de lectura			X
SRSH8	Se debe contar con un sistema de control para el seguimiento de las etapas de lectura			X
SRSH9	El sistema operativo de Desarrollo debe ser Open Source			X
SRSH10	La Plataforma del lenguaje de programación debe ser Open Source			X

SRSH11	La Plataforma de lectura del chip implante debe ser Open Source	X
SRSH12	El sistema alojará la información en bloques dentro del chip	X
SRSH13	Se debe generar un acceso para los administradores del sistema	X

El análisis de la Tabla 7, se basa en las prioridades asignadas a cada requerimiento. Los requerimientos de alta prioridad son aquellos que garantizan la fiabilidad y el correcto funcionamiento del sistema NFC de triaje de emergencias. Estos incluyen pruebas del sistema de lectura del chip NFC, comunicación precisa con el hardware, concordancia de la información recolectada con la base de datos, y disponibilidad de una aplicación para la conexión con el Smartphone Android. También se considera de alta prioridad contar con un chip NFC de almacenamiento adecuado, un lector NFC integrado y un sistema que no se sature en cada etapa de lectura.

3.2. Diseño del Sistema NFC

El diseño de un sistema NFC para los procesos de triaje de emergencias médicas constituye un paso crucial en el desarrollo de una solución tecnológica que permita una lectura eficiente y segura de los chips implantados en los pacientes. El objetivo principal de este sistema es facilitar la recopilación y el acceso rápido a la información médica relevante durante situaciones críticas, lo que puede marcar la diferencia en la atención y el tratamiento adecuados de los pacientes en momentos de emergencia.

Para lograrlo, se deben tener en cuenta aspectos como la funcionalidad del lector NFC, una interfaz de usuario intuitiva y amigable, la capacidad de almacenamiento y las normas de sellado del chip NFC, así como los requisitos de conectividad y comunicación con otros sistemas de información médica. Asimismo, se deben abordar temas relacionados con el rendimiento, la seguridad, la compatibilidad con plataformas existentes y el diseño general del sistema para asegurar un funcionamiento confiable y eficiente.

Durante el proceso de diseño del sistema NFC, se deben realizar evaluaciones de las mejores prácticas de desarrollo de software y hardware, así como llevar a cabo pruebas exhaustivas para garantizar la calidad y la robustez del sistema. Además, es imprescindible seguir estándares y normas establecidas en la industria, como ISO/IEC 18092 para la captación de información del chip NFC y NDEF para la conectividad NFC.

3.2.1. Evaluación de los Requerimientos

En base a los requerimientos levantados en el apartado 3.1, junto con sus respectivas prioridades, a continuación, se presentan las elecciones de hardware y software. Estas elecciones han sido evaluadas considerando las necesidades de los stakeholders involucrados en el proyecto.

3.2.1.1. Requerimientos de Stakeholders

Los Stakeholders involucrados desempeñan un papel fundamental en la definición de los procesos y elementos de la arquitectura propuesta en el desarrollo del proyecto. El componente central de esta propuesta es el chip implante NFC X3 fabricado por la empresa “I am Robot”, el cual cumple diversas funciones vitales, como almacenar,

escribir,

interactuar, transmitir y solicitar información crucial para los procesos de triaje de emergencia. Por otro lado, el aplicativo del smartphone también adquiere gran relevancia al encargarse de mostrar, leer, escribir, solicitar, interactuar, activar y poner en marcha todo el sistema NFC.

3.2.1.2. Requerimientos del Sistema

Por otra parte, debido a que el sistema se compone de múltiples etapas, la interacción y el correcto funcionamiento entre ellas resultan fundamentales para garantizar un desempeño óptimo. Por lo tanto, el sistema deberá trabajar bajo los protocolos NDEF e ISO/IEC 18092, lo que permitirá lograr la interoperabilidad del sistema con diferentes entornos de desarrollo y sistemas operativos de teléfonos inteligentes. De esta manera, se superará las pruebas de las 6M's, las cuales definen el desarrollo funcional y excelente de una aplicación móvil.

3.2.1.3. Requerimientos de Arquitectura

Debido a la complejidad y diversidad de los requerimientos de arquitectura y sus prioridades, es necesario abordar cada uno de ellos de manera individual. En los apartados siguientes, se explican las razones principales que sustentan la elección de los elementos de la arquitectura de acuerdo con los requerimientos. En primer lugar, se describe la metodología que se utilizará, ya que constituye la base fundamental del sistema. A continuación, se presenta la selección del ambiente de programación, y finalmente, se define la arquitectura NFC que se empleará en este proyecto.

Algunos de los parámetros incluidos en estos requerimientos de arquitectura son la frecuencia de operación de 13,56 MHz, que corresponde a la frecuencia estándar para las comunicaciones NFC (ISO/IEC 18092), y el chip implante NFC X3, el cual cuenta con 4KB de memoria (consultar Anexo 3).

3.2.2. Modelo de Diseño para Desarrollo Aplicaciones Móviles y eXtreme Programming

La metodología para el desarrollo de aplicaciones móviles se basa en el artículo "Methodology for Mobile Application Development", donde se introduce una metodología propuesta por (Gasca-Mantilla, et al, 2013) para el desarrollo de aplicaciones móviles. Esta metodología surge a partir de la evolución de los servicios de telefonía móvil en Latinoamérica y la conceptualización de tecnologías y metodologías ágiles para el desarrollo de software, con el objetivo de facilitar la creación de aplicaciones y servicios para dispositivos móviles.

La metodología se compone de cinco etapas: análisis, diseño, desarrollo, pruebas de funcionamiento y entrega.

La metodología se enfoca específicamente en el desarrollo de servicios móviles, tomando en cuenta las características únicas de los entornos móviles, como el canal de radio, la capacidad de los terminales, la portabilidad y la movilidad del usuario. Además, se basa en metodologías ágiles para la ejecución de proyectos en plazos cortos y para adaptarse a los cambios en el mercado de manera ágil y eficiente.

(Gasca-Mantilla, et al, 2013) también presenta resultados del desarrollo de un servicio m-salud (mHealth) destinado a dispositivos Android y J2ME, utilizando la metodología propuesta. Este servicio está diseñado para pacientes que requieren un control periódico de las medidas corporales de tensión arterial y glucosa, y ha demostrado un potencial de éxito entre los usuarios de prueba.

La metodología se destaca por su enfoque en la agilidad y la colaboración con el cliente, permitiendo así lograr el éxito en el desarrollo de aplicaciones móviles.

Por otra parte, la metodología eXtreme Programming (XP) es una reconocida metodología ágil de desarrollo de software creada por Kent Beck en los años 90. Su principal objetivo es entregar software de alta calidad y satisfacer las cambiantes necesidades del cliente de forma colaborativa. XP se fundamenta en 12 prácticas clave que buscan fomentar la comunicación y la adaptación rápida a los cambios, centrándose en la satisfacción del cliente y la entrega continua de software funcional.

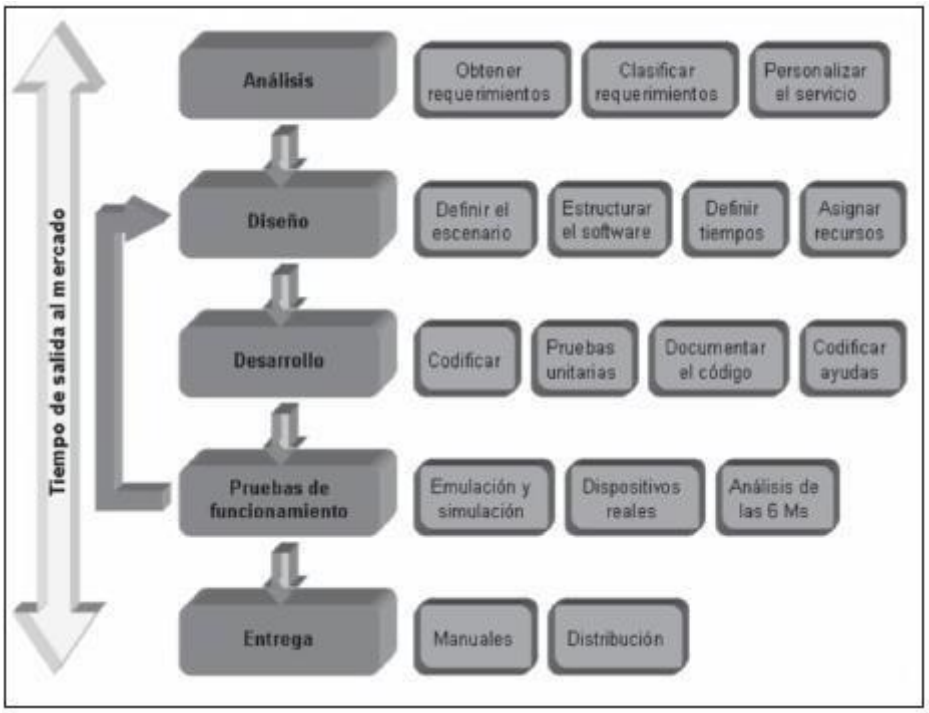
XP aboga por la simplicidad y el diseño limpio, evitando la complejidad innecesaria y favoreciendo la refactorización constante para mantener un código claro y fácil de mantener.

A continuación, se presentan los puntos más importantes de cada etapa de desarrollo de estas metodologías y cómo se han fusionado y asociado para crear la metodología propuesta en esta investigación.

3.2.2.1. Metodología para el Desarrollo de Aplicaciones Móviles

La metodología de desarrollo de aplicaciones móviles, como se muestra en la Figura 3, consta de varias etapas que se siguen en un flujo secuencial. La primera etapa es el análisis, donde se estudian los requerimientos y peticiones del usuario para definir las características del entorno de la aplicación. Se realizan tareas como la obtención y clasificación de estos requerimientos, así como la personalización del servicio.

Figura 3**:** Metodología para el desarrollo de aplicaciones móviles.
Fuente: (Gasca-Mantilla, M, et al, 2013)



A continuación, en la etapa de diseño, el objetivo es plasmar la solución mediante diagramas o esquemas, considerando aspectos técnicos, funcionales, sociales y económicos. Aquí se definen el escenario, la estructura del software, los tiempos y los recursos necesarios para el desarrollo.

Después de la etapa de diseño, se avanza a la fase de desarrollo, donde se implementa lo diseñado en un producto de software. Se lleva a cabo la codificación en el lenguaje de programación seleccionado, se realizan pruebas unitarias para verificar el funcionamiento de la aplicación y se documenta el código desarrollado. Además, se crean ayudas para los usuarios que explican el uso de la aplicación.

Posteriormente, se realiza la etapa de pruebas de funcionamiento, cuyo objetivo es verificar el correcto funcionamiento de la aplicación en diferentes escenarios y condiciones. Se realizan pruebas de emulación y simulación, así como pruebas con dispositivos reales. También se realiza un análisis de las 6 M's para evaluar el potencial de éxito del servicio.

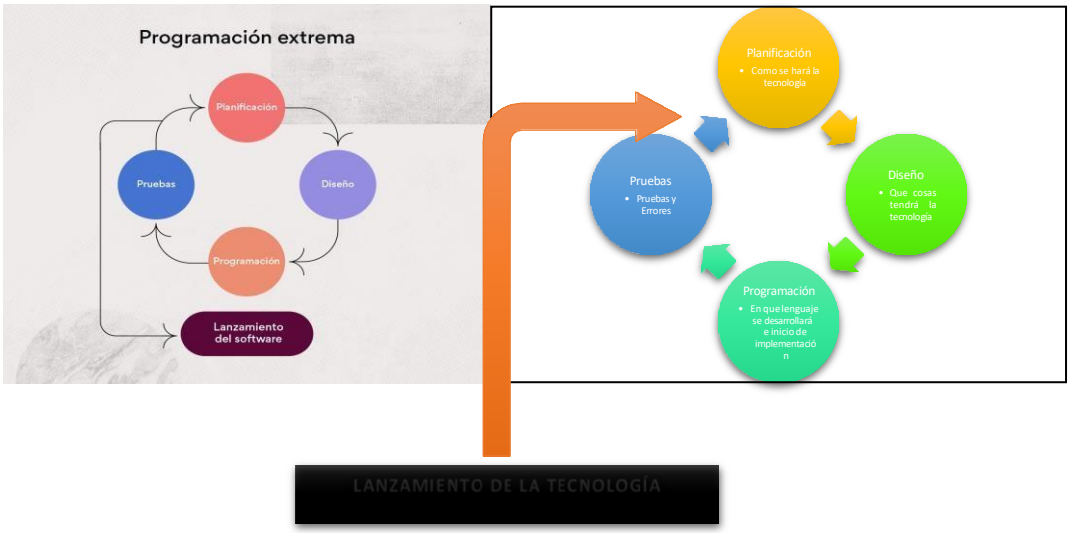
Finalmente, se procede a la etapa de entrega, donde una vez que la aplicación ha sido depurada y se han atendido los requerimientos finales del usuario, se procede a la entrega del ejecutable, código fuente, documentación y manuales del sistema. Además, se define el canal de comercialización de la aplicación.

Cabe mencionar que, si en alguna etapa se encuentran problemas o es necesario realizar ajustes, es posible retroceder a una etapa anterior para corregirlos antes de continuar con el desarrollo del proyecto.

3.2.2.2. eXtreme Programming

Para el desarrollo de software de manera eficiente y efectiva, la metodología eXtreme Programming (XP) se compone de etapas fundamentales (Planificación, Diseño, Programación, Pruebas y Lanzamiento) tal como se muestra en la Figura 4.

Figura 4: Metodología de eXtreme Programming (XP).



- **Planificación:** En esta etapa inicial, se lleva a cabo una planificación detallada del proyecto. Se establece el alcance del trabajo y se definen las prioridades de las funcionalidades a desarrollar. Se crea una lista de tareas y se determina el tiempo y los recursos necesarios para cada una. Además, se establecen los criterios de éxito y los objetivos a alcanzar.

- **Diseño**: Una vez finalizada la etapa de planificación, se procede al diseño del software. En esta fase, se identifican las entidades y componentes clave del sistema. Se definen las interacciones entre ellos y se establecen las interfaces necesarias. El diseño se enfoca en la simplicidad y en la

capacidad de adaptación a los posibles cambios que puedan surgir durante el desarrollo.

- **Programación**: En esta etapa, los programadores se encargan de implementar las funcionalidades del software según el diseño establecido previamente. Un aspecto destacado de XP es la práctica de programación en pareja, donde dos programadores trabajan en conjunto en la codificación, lo que ayuda a mejorar la calidad del código y a compartir conocimientos. Se busca mantener un código limpio y legible para facilitar su mantenimiento y futuras modificaciones.
- **Pruebas**: Las pruebas son una parte integral de XP y se realizan de forma continua a lo largo de todo el proceso de desarrollo. En esta etapa, se llevan a cabo pruebas unitarias para verificar que cada funcionalidad cumple con los requisitos y funciona correctamente. También se realizan pruebas de integración para garantizar el correcto funcionamiento del sistema como un todo. El objetivo es asegurar que el software cumpla con los estándares de calidad establecidos.
- **Lanzamiento del software:** En la última etapa, el software es entregado al usuario o puesto en producción. Se realiza la instalación, configuración y puesta a punto necesaria para que el software esté disponible y funcional para los usuarios finales. Se recopila la retroalimentación de los usuarios y se realizan las mejoras necesarias para futuras iteraciones del proyecto.

3.2.2.3. Desarrollo del Sistema NFC propuesto.

Para el desarrollo del sistema NFC para triaje de emergencias, en el presente trabajo de grado, se combina elementos de las metodologías XP (eXtreme Programming) y de la metodología de desarrollo de aplicaciones móviles. A continuación, se describe cómo se aborda cada etapa:

• **Planificación:** En esta etapa es necesario la reunión con los stakeholders y expertos en triaje de emergencias para definir los parámetros relevantes del proyecto y establecer los requisitos prioritarios. Se identificaron las funcionalidades necesarias para el sistema NFC, tales como la lectura de información de los pacientes, asignación de niveles de urgencia y envío de datos a los servicios médicos. Además, se establecieron los objetivos, prioridades, plazos y recursos disponibles del proyecto. Estas características se obtuvieron mediante diferentes técnicas de recolección de información tales como entrevistas (Anexo 2), y también del análisis y comprensión de la literatura encontrada y el estado del arte del proyecto.

• **Diseño:** En esta etapa, se realiza el diseño del sistema NFC para triaje de emergencias. Se identifican las entidades principales, como pacientes, médicos, dispositivos NFC y administradores. Se definen las interacciones entre ellos y se establecen las interfaces necesarias para una comunicación eficiente. También se diseña la interfaz gráfica de usuario (GUI) que permita a los profesionales de la salud utilizar el sistema de manera intuitiva y eficaz.

• **Programación:** En esta etapa, se implementan las funcionalidades del sistema NFC utilizando la metodología XP. Se trabaja mediante iteraciones para codificar las características identificadas en la etapa de diseño. Se utilizan buenas prácticas de programación, como pruebas unitarias continuas y refactorización del código para mantenerlo limpio y de calidad. Además, se realizan pruebas de escritorio con el aplicativo y el chip implante para visualizar el avance de la programación.

• **Pruebas:** Las pruebas son una parte fundamental del proceso de desarrollo. Se realizan pruebas unitarias para verificar que cada funcionalidad del sistema NFC cumpla con los requisitos establecidos. También se llevan a cabo pruebas de integración para asegurar el correcto funcionamiento del sistema en su conjunto. Se realizan pruebas con datos de muestra y se simulan diferentes situaciones de emergencia para validar la eficacia del sistema en tiempo real.

• **Lanzamiento del software:** En esta etapa final, se procede a la instalación y configuración del sistema NFC en los dispositivos móviles y en los puntos de acceso a la información médica. Se realiza una validación final del sistema antes de ponerlo en producción. Una vez lanzado, se recopila la retroalimentación de los usuarios y se realizan mejoras iterativas basadas en esa retroalimentación para seguir optimizando el sistema NFC para triaje de emergencias. También se lleva a cabo la evaluación de las 6M's (Misión, Modelo, Mecanismo, Medida, Método y Medio) como parte del proceso de lanzamiento final.

3.2.3. Selección de tecnologías de programación adecuadas

Para realizar la programación del aplicativo móvil se necesita una serie de softwares y lenguajes de programación los cuales serán el pilar fundamental de cada etapa de la metodología propuesta para ésta investigación. De ésta manera se seleccionarán las mejores herramientas de acuerdo con su comprensión teórica vista en el capítulo anterior y la capacidad de estas herramientas de aportar facilidad al momento de realizar el despliegue del sistema.

3.2.3.1. Selección del Lenguaje de programación

En el diseño del aplicativo Android para el sistema NFC propuesto destinado a los procesos de triaje de emergencias médicas, la elección del lenguaje de programación juega un papel fundamental que impacta directamente en la eficiencia, productividad y calidad del desarrollo.

En la Tabla 8, se muestra la calificación de varios lenguajes de programación donde Kotlin predomina como lenguaje para el diseño del aplicativo Android del sistema NFC. Se han asignado calificaciones de 1 a 3 para cada criterio de selección, donde 1 representa la calificación más baja y 3 la calificación más alta.

Tabla 8: Criterios de Selección del Lenguaje de Programación

Lenguaje de Programación	Comunidad	Aprendizaje	Rendimiento	Compatibilidad con Android	Herramientas y Bibliotecas	Desarrollo	Total
Kotlin	3	3	2	3	3	3	17
Java	2	2	3	3	3	2	15
Swift	2	2	2	1	3	3	13
Dart (Flutter)	2	3	2	3	3	2	15
JavaScript (React Native)	3	2	2	2	2	2	13

Cada uno de los criterios de selección están sustentados en las técnicas de calificación TQI (Tiobe, 2023), donde destacan los parámetros de la Tabla 8 como los necesarios para los criterios de selección del lenguaje de programación.

TQI (Indicador de Calidad Tiobe) es una métrica utilizada para evaluar la calidad de los lenguajes de programación y herramientas de programación, en función de varios factores relacionados con la seguridad, la portabilidad y la eficiencia del código. Este índice busca proporcionar una medida objetiva de cuán adecuado es un lenguaje de programación para desarrollar software confiable y de alta calidad, y que tan factible es utilizar o no una herramienta de programación para cada uno de los lenguajes.

Kotlin es un lenguaje de programación que ha ganado una gran popularidad en el desarrollo de aplicaciones móviles y más allá. Una de las razones clave para elegir Kotlin es su excelente interoperabilidad con Java. Esto significa que se puede utilizar Kotlin en proyectos que ya tienen una base de código Java existente, lo que facilita la transición y la adopción gradual.

La seguridad de tipo es otro aspecto fundamental de Kotlin. Este lenguaje es fuertemente tipado, lo que reduce los errores en tiempo de ejecución y aumenta la robustez del software. Además, Kotlin aborda de manera más segura el problema de las referencias nulas, lo que disminuye significativamente los errores de NullPointerException que son comunes en Java.

La sintaxis concisa de Kotlin es un punto a favor. Ofrece una forma más legible y expresiva de escribir código en comparación con Java. También cuenta con características avanzadas como las funciones de extensión, que permiten agregar funciones a clases existentes sin modificar su código, lo que mejora la modularidad del software.

Kotlin es un lenguaje que fomenta la programación funcional, lo que resulta en un código más limpio y expresivo. Además, ofrece soporte para la programación asincrónica mediante corrutinas, lo que facilita la escritura de código asincrónico legible y evita problemas comunes de concurrencia.

Otro factor importante es la amplia adopción de Kotlin en el desarrollo de aplicaciones Android, donde es el lenguaje oficial. Esto garantiza que Kotlin esté respaldado por una comunidad activa y que tenga un soporte sólido en la plataforma móvil.

3.2.3.2. Selección del Software de Programación

En el contexto del desarrollo de aplicaciones, es de suma importancia elegir la plataforma de programación adecuada. Para abordar esta cuestión, se ha llevado a cabo una comparación de las principales herramientas de programación disponibles en el mercado para aplicaciones Android. En este proceso de selección, se ha optado por enfocarse en aquellas soluciones que son tanto gratuitas como de código abierto, lo que permite un acceso libre y transparente al código fuente, lo que resulta especialmente beneficioso en términos de flexibilidad y personalización.

La evaluación de estas herramientas se basa en una serie de criterios esenciales para el desarrollo de aplicaciones móviles. Estos criterios abarcan aspectos como la plataforma de destino, lenguaje de programación, integración con Android, compatibilidad con sistemas operativos, la capacidad de integración con otras tecnologías y servicios, así como la comunidad de desarrolladores activa que respalda cada una de estas herramientas. Estos criterios son tomados del TQI (Tiobe, 2023), que detallan cada uno de los argumentos necesarios para la selección del software de programación.

En la Tabla 9, se detalla los criterios de selección entre los softwares de programación existentes.

Tabla 9 : Criterios de selección de software de programación

Criterios de Selección	Android Studio	Xamarin	Flutter	React Native
Plataforma de Destino	Android	Android, iOS	Android, iOS	Android, iOS
Lenguaje de Programación	Java, Kotlin	C#	Dart	JavaScript, TypeScript
Integración con Android	Totalmente integrado con Android SDK	Integración con Android SDK	Integración con Android SDK	Integración con Android SDK
Compatibilidad con Sistemas Operativos	Compatible con Windows, macOS y Linux	Compatible con Windows, macOS	Compatible con Windows, macOS y Linux	Compatible con macOS y Linux
Comunidad y Soporte	Gran comunidad de desarrolladores y amplia documentación	Comunidad activa	Comunidad creciente	Comunidad activa
Entorno de Desarrollo	Entorno de desarrollo completo y rico en características	Entorno de desarrollo completo	Entorno de desarrollo completo	Entorno de desarrollo completo
Interfaz de Usuario (UI)	Editor visual de interfaces de usuario (XML) y vista previa en tiempo real	Editor visual de interfaces de usuario	Widget-based UI	Component-based UI
Depuración y Perfilado	Herramientas avanzadas de depuración y análisis de rendimiento	Herramientas de depuración	Herramientas de depuración	Herramientas de depuración

Emuladores y Dispositivos Físicos	Amplia variedad de emuladores y soporte para dispositivos físicos	Emuladores disponibles	Emuladores disponibles	Emuladores disponibles
Integración de Control de Versiones	Soporte para Git y otras herramientas de control de versiones	Soporte para control de versiones	Soporte para control de versiones	Soporte para control de versiones
Actualizaciones y Mejoras	Actualizaciones frecuentes y mejoras constantes	Actualizaciones periódicas	Actualizaciones periódicas	Actualizaciones periódicas
Costo	Gratuito y de código abierto	Varias opciones de licencia	Gratuito	Gratuito

En consecuencia, se evalúan los distintos criterios de selección de software con el fin de escoger la herramienta más adecuada para llevar a cabo este trabajo de grado.

A continuación, en la Tabla 10, se refleja cómo se han asignado calificaciones de 1 a 3 para cada criterio de selección, donde 1 representa la calificación más baja y 3 la calificación más alta.

Tabla 10: Matriz de Selección de Software de Programación

Criterios de Selección	Android Studio	Xamarin	Flutter	React Native
Plataforma de Destino	3	2	2	2
Lenguaje de Programación	3	2	2	2
Integración con Android	3	2	2	2
Compatibilidad con SO	3	2	3	2
Comunidad y Soporte	3	2	2	2
Entorno de Desarrollo	3	2	2	2
Interfaz de Usuario (UI)	3	2	2	2
Depuración y Perfilado	3	2	2	2
Emuladores y Dispositivos Físicos	3	2	2	2

Integración de Control de Versiones	3	2	2	2
Actualizaciones y Mejoras	3	2	2	2
Costo	3	2	3	3
Total	36	24	26	25

Android Studio se destaca como la opción preferida en la matriz de selección de software debido a su alto rendimiento en todos los criterios de selección. Diseñado específicamente para el desarrollo de aplicaciones Android, ofrece una integración perfecta con la plataforma, utiliza lenguajes de programación ampliamente aceptados como Java y Kotlin, y proporciona un entorno de desarrollo amigable con herramientas de depuración y perfilado de alta calidad. Además, su gran comunidad de desarrolladores y el sólido soporte de Google garantizan una amplia base de conocimientos y soluciones rápidas. Su gratuidad y frecuentes actualizaciones lo hacen accesible y actualizado, lo que lo convierte en la elección óptima para el desarrollo de aplicaciones móviles en Android.

3.2.4. Definición de parámetros de la Arquitectura NFC

La arquitectura NFC es una solución tecnológica que permite la comunicación inalámbrica de corto alcance entre dispositivos compatibles. En este contexto, el diseño de la arquitectura NFC adquiere una importancia significativa, ya que establece las bases para el intercambio seguro de datos y la ejecución de diversas funcionalidades. Este diseño se basa en la estructuración de componentes y etapas clave que trabajan en conjunto para garantizar la eficiencia y confiabilidad del sistema NFC.

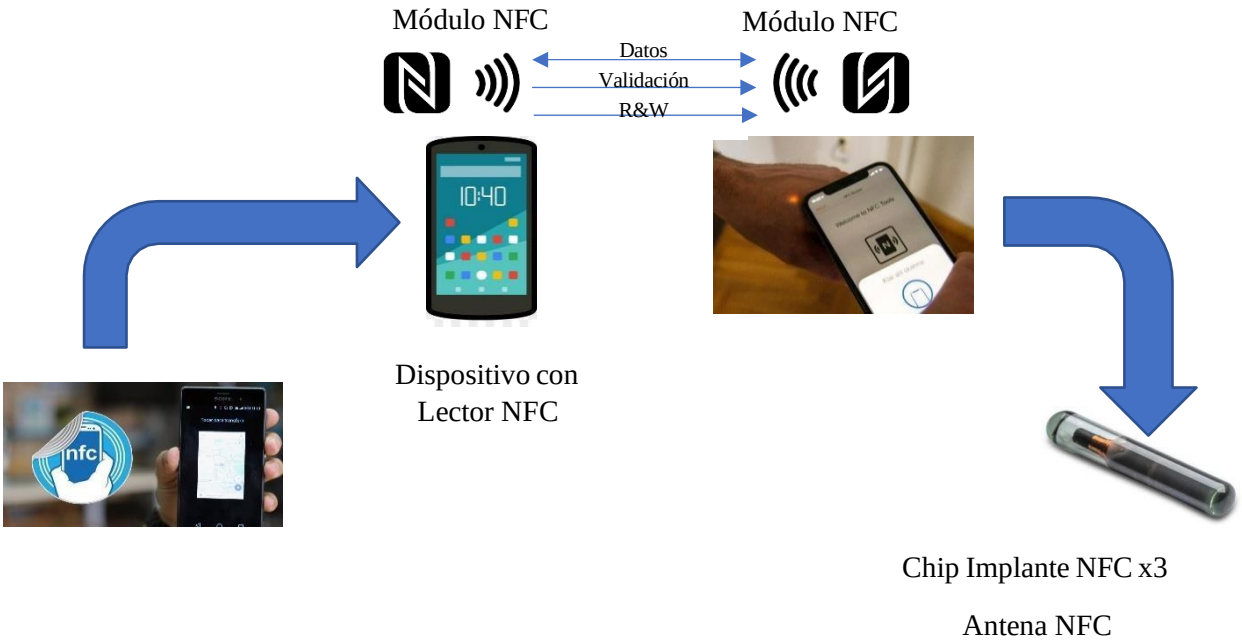
La arquitectura NFC se compone de varios elementos interrelacionados para permitir la comunicación inalámbrica de corto alcance y el intercambio de datos. En primer lugar, se encuentra el lector NFC, el cual está integrado en un teléfono inteligente.

Este lector NFC actúa como punto de partida para las transacciones NFC y es capaz de enviar y recibir datos, así como realizar validaciones y procesos de lectura y escritura.

Dentro del teléfono inteligente, se encuentra el módulo NFC, el cual despliega las funcionalidades específicas de NFC tales como el NDEF e ISO/IEC18092 descritos en el capítulo 2, para el correcto funcionamiento de la tecnología NFC. Este módulo es responsable de establecer y mantener la comunicación con otros dispositivos NFC, así como de gestionar la transferencia de datos entre ellos.

Adicionalmente, en el contexto de esta arquitectura, se hace referencia al chip NFC X3, el cual incluye un módulo NFC embebido. Este chip constituye un componente físico clave en el sistema, ya que proporciona la infraestructura necesaria para el funcionamiento del módulo NFC. Además, el chip NFC X3 está diseñado con su propia antena NFC, lo cual es fundamental para habilitar la comunicación NFC a nivel físico, tal como se muestra en la figura 5.

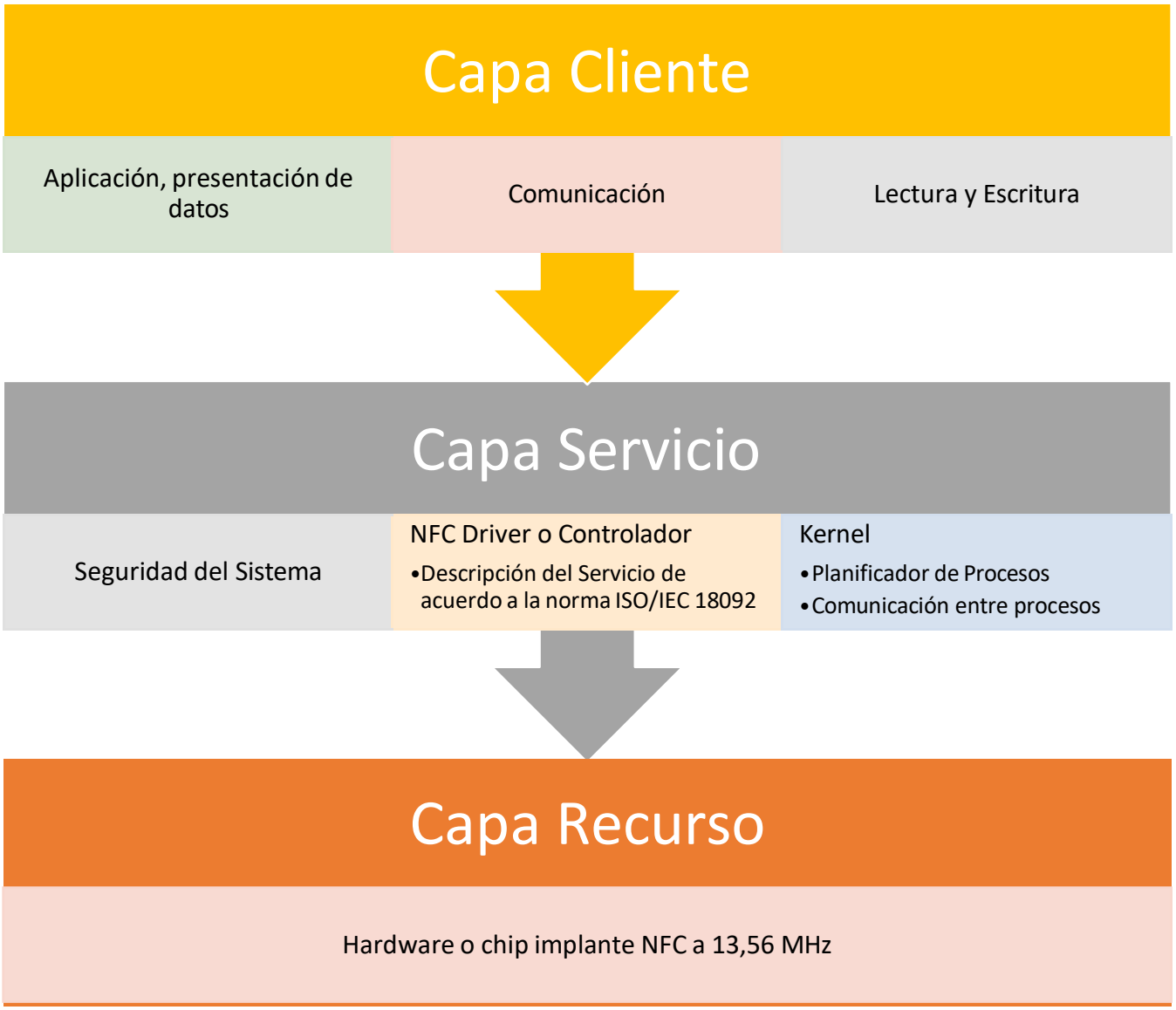
Figura 5: Diagrama de bloques de la Arquitectura NFC propuesta



Por consiguiente, se ha desarrollado un modelo de análisis de la tecnología NFC de 3 capas donde cada etapa estará interrelacionada con la superior para lograr una convergencia en toda la arquitectura NFC.

En la arquitectura NFC propuesta, se representa un diagrama de bloques, que proporciona una descripción detallada de su funcionamiento. La Figura 6 muestra el diagrama de capas general del sistema NFC, el cual se basa en 3 etapas principales. Cada etapa tiene requisitos específicos que garantizan su correcto funcionamiento y contribuyen al desarrollo integral del proyecto. Estas etapas se diseñan de acuerdo con los requerimientos correspondientes presentados en la Tabla 7, para asegurar una funcionalidad adecuada del sistema.

Figura 6: Diseño de 3 capas del sistema NFC



Con base en los requerimientos propuestos y analizados, el sistema NFC ha sido diseñado y estructurado para cumplir con dichos requerimientos. Ahora, se inicia la fase de implementación y despliegue del sistema, donde se pondrá en práctica todo el trabajo realizado durante la definición de parámetros del sistema. Esta etapa implica la configuración de los componentes NFC, la integración con los dispositivos y sistemas

correspondientes, así como la realización de pruebas exhaustivas para garantizar el correcto funcionamiento del sistema. Una vez completada esta fase, el sistema estará listo para su implementación y uso por parte de los usuarios finales.

CAPITULO IV

DESPLIEGUE DEL SISTEMA

En este capítulo se aborda el despliegue del sistema NFC, tomando en consideración los requerimientos descritos en el capítulo anterior, así como las consideraciones de diseño . El objetivo principal de este capítulo es presentar la implementación del sistema NFC, utilizando una combinación de las metodologías eXtreme Programming y Metodología de Desarrollo de Aplicaciones Móviles. Estas metodologías permiten un enfoque secuencial del proyecto, basado en las etapas de planificación que ya ha sido descrita en esta investigación.

Las etapas que se analizan de manera global en este capítulo son: planificación, diseño, programación, pruebas y lanzamiento del software. Cada una de estas etapas juega un papel fundamental en el despliegue exitoso del sistema NFC. Durante la etapa de planificación, se establecen los objetivos del proyecto, se definen los requisitos y se asignan los recursos necesarios. En la etapa de diseño, se plasman las soluciones mediante diagramas y esquemas que consideran aspectos técnicos, funcionales y médicos, así como también la elección de las herramientas a utilizarse para el desarrollo del sistema.

La etapa de programación implica la implementación del diseño en un producto de software, mediante la codificación y la realización de pruebas unitarias para verificar el correcto funcionamiento de la aplicación. La etapa de pruebas se enfoca en verificar el funcionamiento adecuado del sistema en diferentes escenarios y condiciones, utilizando tanto pruebas de emulación y simulación como pruebas con dispositivos reales. Por último, la etapa de lanzamiento del software implica la instalación y configuración del

sistema NFC, la validación final y la entrega del producto como tal a los usuarios finales.

4.1.Etapa de Planificación

Durante esta etapa, se llevaron a cabo diversas metodologías que se describieron en capítulos anteriores, para recopilar la información necesaria y satisfacer los requerimientos de los stakeholders del proyecto, con el fin de alcanzar los objetivos propuestos.

4.1.1. Recopilación de Información Previa

En esta etapa de investigación, se lleva a cabo un análisis del estado actual de la tecnología NFC en el contexto médico, así como la recopilación de datos a través de entrevistas con profesionales de la salud, mesas redondas con investigadores (Anexo 2) y revisiones de la literatura científica (ver Capítulo 2). Estas interacciones han revelado diversas perspectivas sobre el uso de NFC en situaciones de emergencia, destacando sus ventajas, como la identificación precisa de pacientes y la mejora en la toma de decisiones médicas, pero también las preocupaciones relacionadas con la seguridad de los datos y los desafíos técnicos. Esta información ha servido como base para la planificación del sistema NFC, garantizando que se aborden adecuadamente los desafíos y se maximicen los beneficios (ver Capítulo 3).

En este sentido, la revisión de la literatura científica existente relacionada con la aplicación de NFC en entornos médicos (ver Capítulo 2), permite tener una visión más amplia y fundamentada acerca de las tendencias y avances tecnológicos en este campo. En conjunto, estas actividades de investigación han respaldado un enfoque basado en evidencia y mejores prácticas en la implementación de la tecnología NFC en situaciones

de emergencia, contribuyendo así a un panorama más completo y realista de su viabilidad y aplicaciones en la atención médica de emergencia.

Es así como en la Tabla 13a, se resume cual es la información más relevante dentro de la etapa de planificación para el despliegue de la aplicación móvil y cuáles serían las posibles variables para utilizarse dentro del ambiente de programación, teniendo como base fundamental los requerimientos de stakeholders analizados en las tablas del capítulo anterior y cada una de las calificaciones que se otorgó a los diferentes softwares y lenguajes de programación.

Tabla 13a: Aspectos relevantes de programación con sus variables

Aspecto	**Impacto en el Aplicativo**	**Programación Global**
Historial médico	Permite acceder a la información médica	Historial médico de acuerdo con la BDD del hospital o clínica. Permitir al sistema la lectura de URLs
Identificación rápida	Agiliza el proceso de identificación	Desarrollar una función de lectura NFC en la aplicación móvil que permita identificar rápidamente al paciente y acceder a su información de manera instantánea.
Privacidad de datos	Protege la información personal y médica	Adaptar al entorno de programación la autenticación y encriptación provista por el chip.
Interoperabilidad	Facilita la integración con otros sistemas	Asegurar la compatibilidad de la aplicación móvil con diferentes dispositivos y sistemas de atención médica. Utilizar estándares de comunicación y protocolos adecuados para facilitar la integración con otros sistemas existentes.

Interfaz de usuario	Mejora la usabilidad y experiencia del usuario	Diseñar una interfaz de usuario intuitiva y fácil de usar para que los profesionales médicos puedan acceder y visualizar la información del paciente de manera clara y rápida. Utilizar principios de diseño centrados en el usuario.

Una vez definidos los aspectos más relevantes que se analizarán en el diseño del sistema, es posible establecer las variables globales en función de sus características específicas. Cada una de estas variables se convertirá en una entidad programada dentro de la aplicación. Cada entidad estará representada por una clase que reflejará las acciones y funcionalidades del aplicativo relacionadas con esa variable particular. (ver Tabla 13b)

Tabla 13b: Variables globales para el desarrollo de entidades (Pseudocódigo)

Variable	Función	Tipo de Variable
historialMedicoURL	Almacena la URL hacia el historial médico.	Variable global de tipo String.
chipNFCID	Almacena el ID del chip NFC para identificar la información del chip.	Variable global de tipo String.
ObtenDatos	Almacena el bloque de código para obtener información.	Variable global de tipo String que contiene un bloque de código.
protocoloComunicacion	Almacena el protocolo de comunicación utilizado.	Variable global de tipo String.
formatoDatos	Almacena el formato de datos utilizado en la comunicación. (NDEF, MIFARE)	Variable global de tipo String.
disenoInterfaz	Almacena información relacionada con el diseño de la interfaz.	Variable global de tipo Object para diseño de interfaz.
interaccionElementos	Almacena información sobre la interacción con elementos de la interfaz.	Variable global de tipo Object para interacción con elementos de la interfaz.

ndefMessage	Representa un mensaje NDEF para la comunicación NFC.	Variable global de tipo NdefMessage.

Estas variables globales se utilizarán en la programación de la aplicación móvil para cumplir con las funciones específicas de cada aspecto y para facilitar la comunicación a través del protocolo NDEF en el contexto de NFC.

4.1.2. Alcance de la Implementación

El alcance de la implementación del despliegue del sistema NFC en ambientes controlados se refiere a la aplicación de esta tecnología en entornos específicos y controlados, como hospitales, clínicas y laboratorios de investigación médica. En estos ambientes, se busca aprovechar las ventajas del sistema NFC para mejorar la eficiencia y precisión en la identificación de pacientes, el acceso a su información médica y la realización de procesos de lectura y escritura en los chips implantes NFC.

Al implementar el sistema NFC en estos ambientes controlados, se pueden establecer medidas de seguridad y protocolos para garantizar la privacidad y confidencialidad de los datos del paciente. Además, se tiene un mayor control sobre los dispositivos y equipos utilizados, así como sobre las condiciones ambientales que pueden afectar su funcionamiento. Esto permite realizar pruebas, validaciones y ajustes de manera más efectiva, asegurando que el sistema opere de manera adecuada y cumpla con los estándares de calidad requeridos.

Es importante tener en cuenta que el alcance de la implementación en ambientes controlados no abarca necesariamente la aplicación en entornos no controlados o de acceso público, donde pueden surgir desafíos adicionales en términos de seguridad, compatibilidad de dispositivos y gestión de riesgos. Por lo tanto, la implementación del sistema NFC se enfoca en brindar soluciones específicas y eficientes dentro de estos

ambientes controlados, donde se pueden garantizar condiciones óptimas para su funcionamiento.

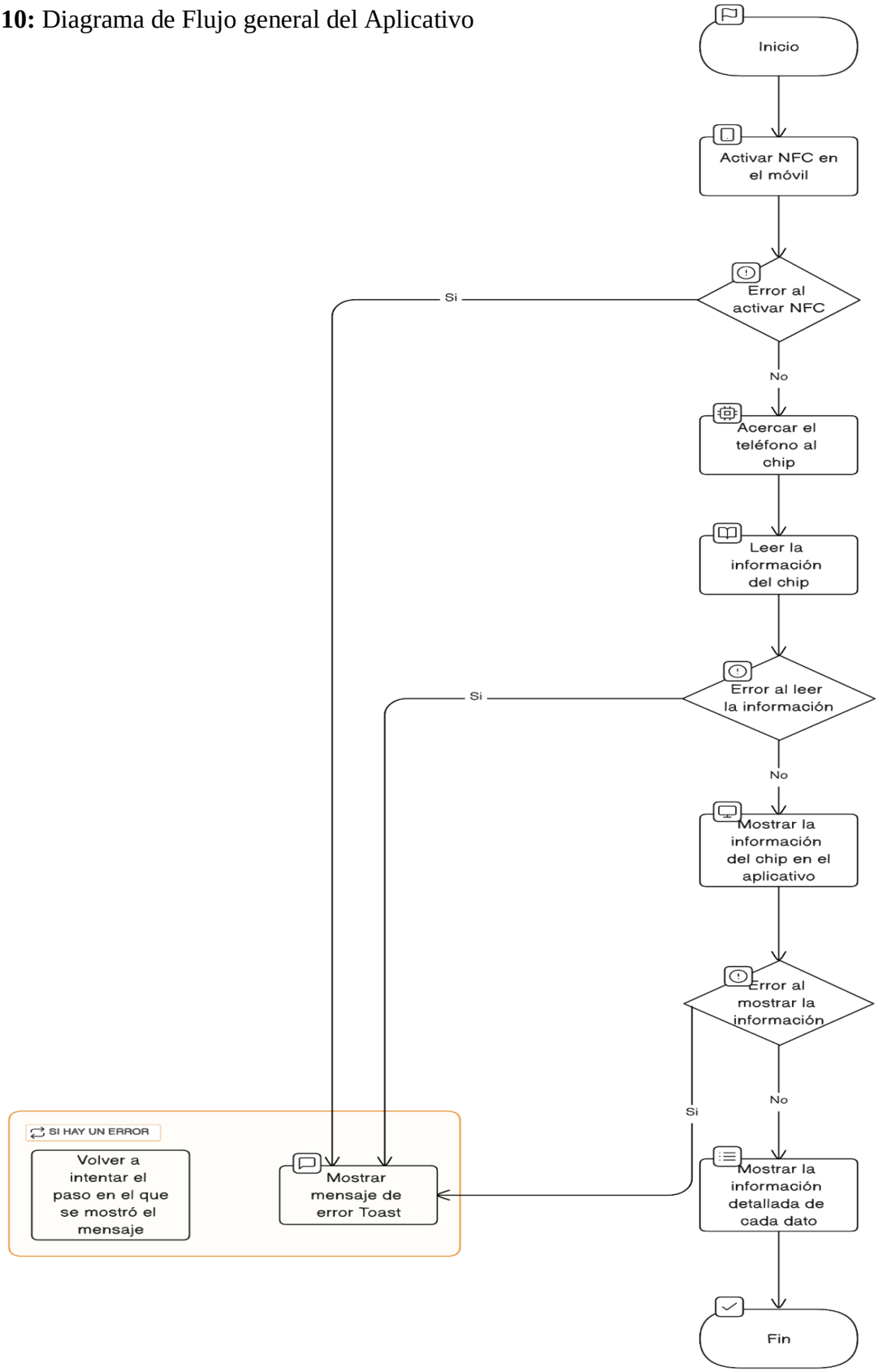
4.2. Etapa de Diseño

En esta etapa, se lleva a cabo el diseño del sistema NFC tomando en cuenta diversos aspectos fundamentales para su desarrollo. Se analiza la arquitectura del sistema, optando por un modelo de 3 capas, descrito en el capítulo anterior, que permite una convergencia integral en toda la estructura NFC. Asimismo, se selecciona el lenguaje de programación Kotlin para el diseño de la aplicación Android, basándose en su compatibilidad con Android, aumento de productividad y capacidad de interoperabilidad con Java, lo cual lo convierte en una opción de primer orden para desarrollar el aplicativo en este lenguaje de programación

Además, se eligen plataformas de desarrollo y lectura Open Source que ofrecen flexibilidad e integración para implementar las funcionalidades necesarias del sistema NFC. La estructura del software se organiza en módulos y componentes, como la interfaz gráfica de usuario (GUI), el sistema de lectura y escritura en el chip NFC, y el sistema de control para gestionar la comunicación, asegurando así la eficiencia y confiabilidad del sistema en conjunto.

Se toma en cuenta y se hace uso de variables detalladas en el apartado anterior para el funcionamiento del sistema NFC, como la identificación del paciente, datos médicos relevantes, afecciones y alergias, y datos que puedan ser usados para los procesos del triage de emergencias. Para facilitar la programación, en la Figura 10, se detalla el diagrama de flujo que representa la secuencia de acciones y procesos en la implementación del sistema.

Figura 10: Diagrama de Flujo general del Aplicativo



4.2.1. Diagrama de casos de uso y secuenciales

En el desarrollo de este proyecto de titulación, se identifican tres casos de uso fundamentales que resaltan las funciones principales de la aplicación móvil diseñada para gestionar información médica mediante la tecnología NFC. Estos casos de uso se centran en la escritura de datos en el chip NFC por parte del administrador del sistema, mediante la aplicación NFC Tools, la lectura de la información almacenada en el chip por parte del médico y la visualización de los datos médicos del paciente dentro de la aplicación. Cada caso de uso aborda una parte esencial de la interacción entre los diferentes actores involucrados y la aplicación, proporcionando una comprensión completa de cómo se emplea esta tecnología para mejorar la atención médica.

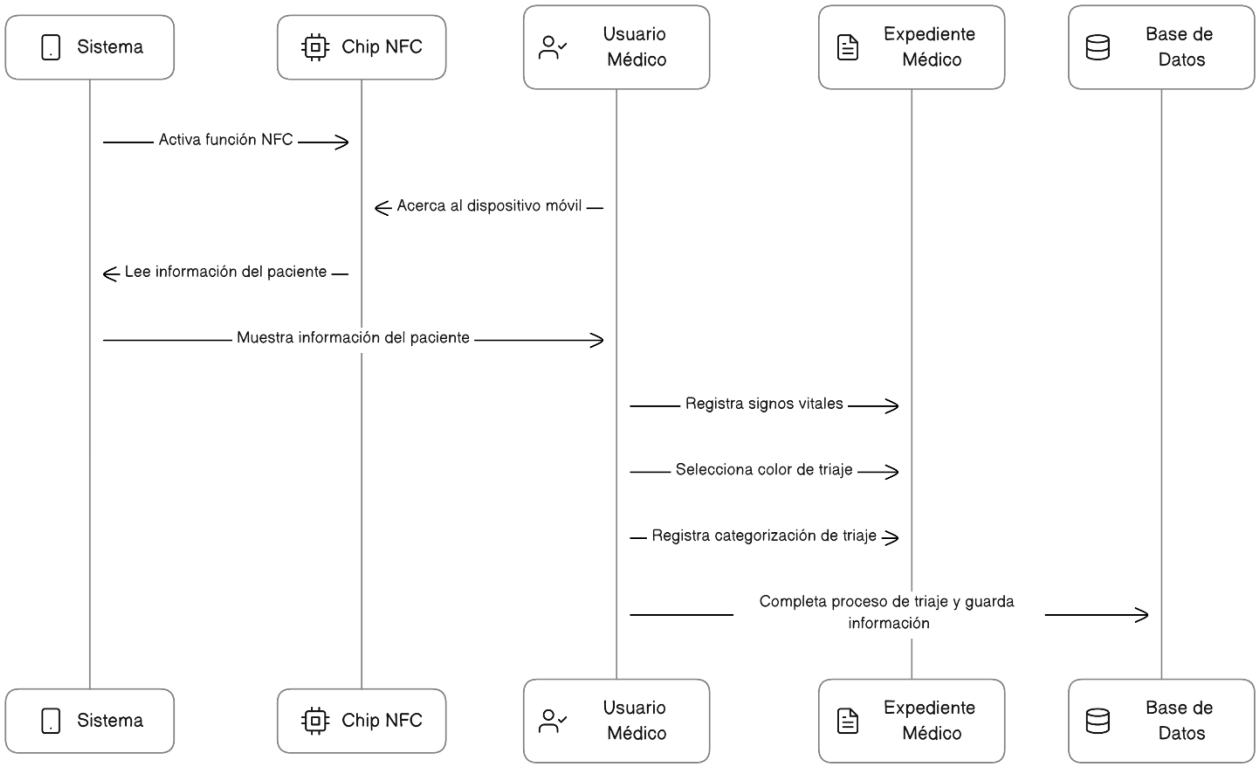
Tabla 14: Diseño de casos de Uso

Caso de Uso	Actores	Descripción	Escenario Principal	Escenario Alternativo	Precondiciones
Escritura de Información en el Chip (Administrador)	Administrador del Sistema	Permite al administrador del sistema utilizar la aplicación NFC Tools para escribir información en el chip NFC del paciente.	1. El administrador del sistema inicia la aplicación NFC Tools. 2. El administrador selecciona la opción de escritura de datos en el chip NFC. 3. El administrador ingresa la información médica relevante del paciente en la aplicación. 4. La aplicación NFC Tools escribe la información en el chip NFC del paciente.	- Si la escritura en el chip NFC no es exitosa, se muestra un mensaje de error y se permite volver a intentarlo.	El administrador del sistema tiene acceso a la aplicación NFC Tools y al chip NFC del paciente.
Lectura de Información del Chip (Médico)	Médico	Permite al médico utilizar la aplicación desarrollada en esta tesis para leer la información almacenada en el chip NFC del paciente.	1. El médico inicia la aplicación desarrollada. 2. El médico activa la función NFC en el dispositivo móvil. 3. El médico acerca el chip NFC al dispositivo móvil. 4. La aplicación desarrollada lee la información del paciente almacenada en el chip NFC. 5. La aplicación muestra la información médica del paciente en la pantalla del dispositivo.	- Si la lectura del chip NFC no es exitosa, se muestra un mensaje de error y se permite volver a intentarlo.	El médico tiene acceso a la aplicación desarrollada y al chip NFC del paciente.

Mostrar Información del Paciente en la Aplicación	Médico	Permite al médico utilizar la aplicación desarrollada para visualizar la información médica del paciente almacenada en la aplicación.	1. El médico inicia la aplicación desarrollada. 2. El médico selecciona la opción de "Mostrar Información del Paciente". 3. La aplicación muestra la información médica del paciente, incluyendo antecedentes médicos y alergias, en la pantalla del dispositivo.	- Si no hay información del paciente disponible, se muestra un mensaje indicando que la información está vacía.	El médico tiene acceso a la aplicación desarrollada y a la información del paciente almacenada en la aplicación.

Con el fin de mejorar el desarrollo de los casos de uso para su programación, en la Figura 11, se muestra el diagrama de secuencia de los casos de uso presentados anteriormente y como interactúan con las entidades entre sí, de esta manera se puede avanzar al siguiente apartado para realizar la estructuración del software, donde se trabajará con el modelo MVVM para desarrollar la codificación respectiva.

Figura 11: Diagrama de secuencia del sistema

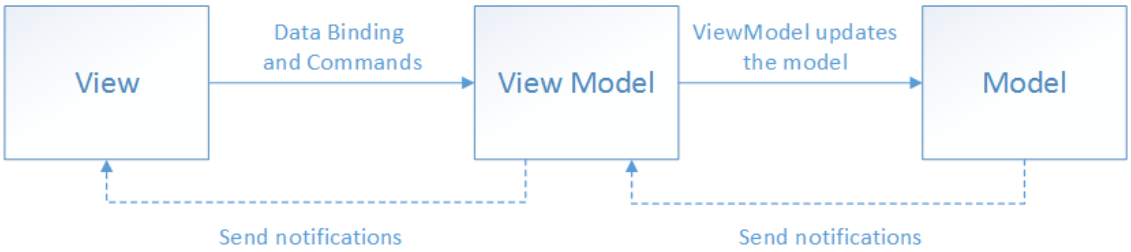


4.2.2. Estructura del Software

Para implementar el sistema NFC en el ambiente de programación Android Studio utilizando el lenguaje de programación Kotlin, se sigue la arquitectura de software sólida y modular propuesta en unión que se basa en las metodologías eXtreme Programming y Desarrollo de Aplicaciones móviles como se detalló en el capítulo anterior.

Para desarrollar la programación y codificación del aplicativo, es necesario aplicar una metodología específica para el desarrollo del software seleccionado. Un enfoque popular para estructurar aplicaciones Android es el patrón de arquitectura MVVM (Modelo-Vista-Vista-Modelo), que facilita la separación de responsabilidades y el mantenimiento del código.

Figura 12: Modelo MVVM y su interacción. Fuente: (Stonis M., et al, 2023)



Como se observa en la Figura 12., el primer componente es el Modelo (Model), donde se definen las clases y estructuras de datos que representarán la información de los chips implantes NFC X3 y otros datos necesarios para el funcionamiento del sistema. Aquí se incluirán clases para el historial médico, alergias, medicamentos y otros detalles relevantes del paciente. También se crean clases para interactuar con la base de datos centralizada y manejar la comunicación NFC con los chips implantes.

El siguiente componente es la Vista (View), encargada de la interfaz de usuario y la presentación visual de la información. Aquí se desarrolla actividades y fragmentos que compondrán la aplicación móvil utilizando la interfaz XML que nos ofrece la herramienta Android Studio. Estos elementos se comunicarán con el ViewModel (VistaModelo) para obtener y mostrar datos en la interfaz. Además, aquí implementaremos acciones del usuario, como la lectura y escritura de información en

los chips NFC, los cuales se definen en la arquitectura en unión con los estándares NDEF e ISO/IEC 18092.

El tercer componente es el VistaModelo (ViewModel), que actúa como intermediario entre el Modelo y la Vista. Su función principal es manejar la lógica de "negocio" y mantener los datos en un estado adecuado para la interfaz de usuario. Cuando la Vista necesita acceder a datos o realizar operaciones, lo hace a través del ViewModel.

Además de la estructura MVVM, es esencial seguir buenas prácticas de programación, como la modularidad y el uso de patrones de diseño para promover la reutilización del código y mejorar la legibilidad. Asimismo, se deben implementar técnicas de manejo de errores y pruebas unitarias para asegurar la calidad del software y prevenir fallos críticos en situaciones reales. Con esta estructura sólida y enfoque de desarrollo, el sistema NFC para el triage de emergencias podrá operar de manera confiable y brindar un acceso rápido y seguro a la información médica relevante para los pacientes y profesionales de la salud.

4.2.3. Interacciones entre las entidades

El sistema se apoya en la tecnología NFC, en particular en el chip implante NFC X3, para almacenar y gestionar la información médica crítica de los pacientes. Este chip se convierte en un repositorio seguro para datos médicos, como historiales médicos completos, alergias y registros de medicamentos. Los pacientes, en este contexto, pueden aprovechar la funcionalidad de sus dispositivos móviles equipados con NFC para acceder a estos datos almacenados en el chip implante NFC X3.

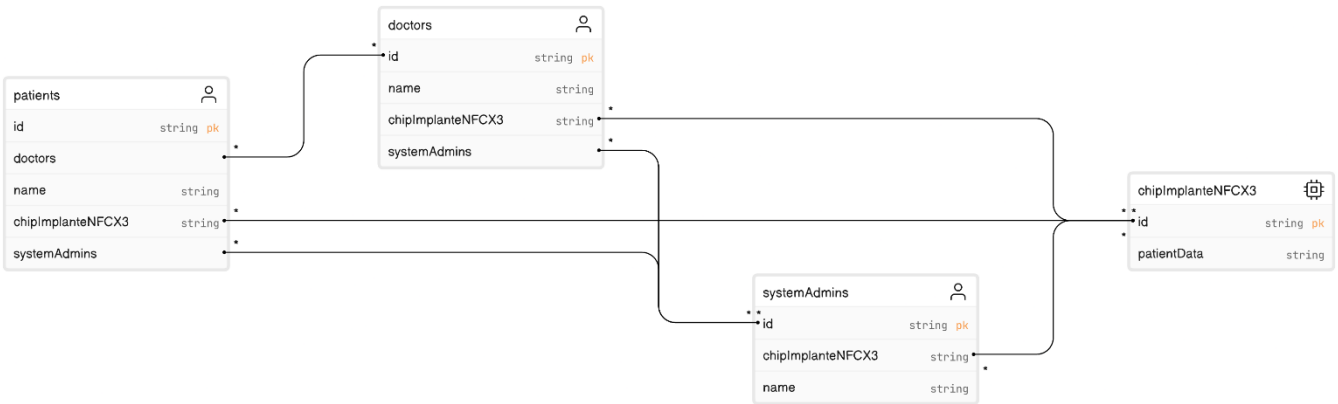
En lo que respecta a los profesionales de la salud, los médicos desempeñan un papel esencial en la utilización del sistema. Su acceso autorizado les permite consultar los

historiales médicos de los pacientes, conocer sus alergias y revisar sus tratamientos farmacológicos actuales. Además de la visualización de datos, los médicos también tienen la capacidad de registrar nuevos datos médicos en el sistema, siempre que cuenten con la debida autorización administrativa.

El componente físico clave en esta infraestructura es el chip implante NFC X3. Este dispositivo es responsable de la retención segura de los datos médicos de los pacientes y actúa como un punto de acceso tanto para pacientes como para médicos. Su utilización implica la lectura y escritura de datos médicos, lo que garantiza que la información relevante esté siempre al alcance de quienes la necesitan.

Para la gestión integral del sistema, se requieren administradores del sistema. Estos profesionales asumen la responsabilidad de supervisar y mantener el sistema en su totalidad. Esto incluye la gestión de datos de usuario, así como la facultad de registrar nuevos datos médicos en el sistema.

Figura 13: Diagrama de interacción de entidades del proyecto.



Como se observa en la Figura 13, en el contexto de este sistema de gestión médica, los pacientes desempeñan un papel fundamental al utilizar el chip implante NFC X3 para el almacenamiento y acceso a su información médica, que incluye historiales médicos, alergias y medicamentos. Para llevar a cabo esta operación, los pacientes requieren un dispositivo móvil habilitado con tecnología NFC, que les permite la lectura de los datos almacenados en dicho chip. Además de esta función, los pacientes colaboran activamente en la actualización de la información proporcionando datos relevantes a los administradores del sistema. Estos últimos son los únicos con la autorización para agregar o modificar datos en el chip, lo que garantiza la integridad y seguridad de la información médica alojada en el mismo.

Por otra parte, los médicos, en calidad de profesionales de la salud, utilizan el sistema como herramienta central para acceder a la información médica de los pacientes. A través de esta plataforma, tienen la capacidad de consultar el historial médico completo, así como detalles sobre alergias y medicamentos de los pacientes. Esta funcionalidad les permite tomar decisiones médicas fundamentadas en situaciones clínicas. En este sentido, los médicos desempeñan un papel crucial al determinar el grado de urgencia de una emergencia médica, basándose en la información disponible en el chip implante NFC X3 y el sistema en general.

Los administradores del sistema tienen la responsabilidad de gestionar y supervisar el funcionamiento integral del sistema. Esto comprende la administración de datos de usuario, tales como médicos y pacientes, con el objetivo de asegurar que las actualizaciones de datos médicos se realicen de manera precisa y segura. Además, llevan a cabo una supervisión constante del sistema para garantizar su operatividad sin inconvenientes, así como la integridad de la información médica almacenada en el chip implante NFC X3.

En el epicentro de estas interacciones se encuentra el chip implante NFC X3, actuando como un almacén central para la información médica crítica de los pacientes. Facilita la lectura de datos médicos por parte de médicos y pacientes autorizados, asegurando un acceso rápido y seguro a esta información vital. En resumen, estas interacciones que involucran a pacientes, médicos, administradores y el chip implante NFC X3 forman un sistema de gestión médica integral que garantiza la seguridad y la toma de decisiones fundamentadas en el ámbito médico de emergencias.

4.2.4. Diseño de la interfaz gráfica

En el contexto de las aplicaciones desarrolladas para la plataforma Android mediante la utilización de Android Studio, la interfaz gráfica se concibe y configura mediante el uso de lenguaje XML y herramientas de diseño visual.

La interfaz gráfica de una aplicación no solo se refiere a su apariencia visual, sino también a la manera en que los usuarios interactúan con ella. Esta debe ser diseñada de manera que resulte intuitiva, atractiva y eficaz, con el propósito de asegurar una experiencia positiva para el usuario. Esto implica la configuración de elementos como botones, campos de entrada, menús, pantallas de inicio y otros componentes que los usuarios visualizan y utilizan.

El proceso de diseño de la interfaz gráfica en Android Studio permite la creación de diseños personalizados que se ajustan a las necesidades particulares de la aplicación en desarrollo. Este proceso se lleva a cabo mediante la combinación de elementos previamente definidos en la Tabla 6 de requerimientos del capítulo anterior, y elementos personalizados de acuerdo a los estándares NDEF e ISO/IEC 18092, los cuales se organizan en archivos XML y se gestionan a través del editor visual de Android Studio.

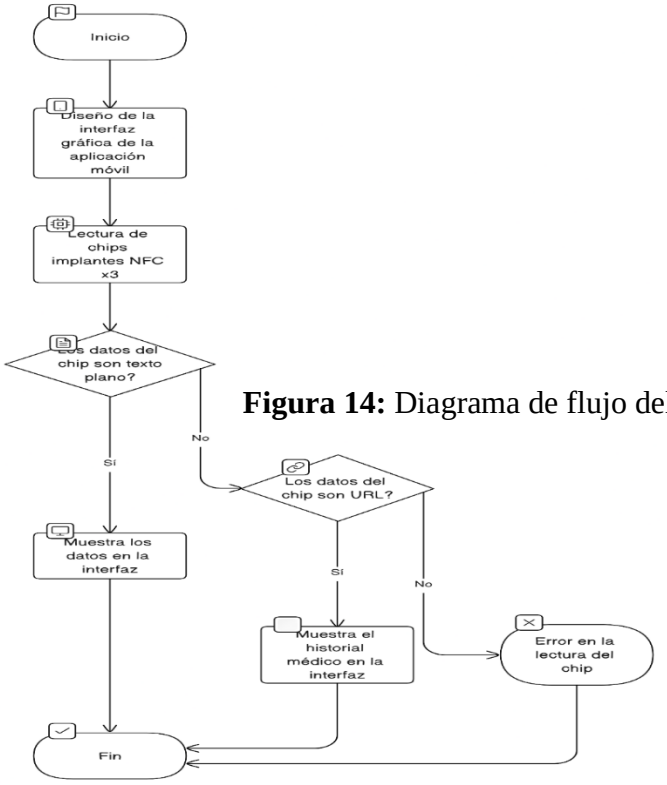


Figura 14: Diagrama de flujo del diseño de la interfaz gráfica.

En esta etapa del proceso de desarrollo, se definen los parámetros, tanto para el aspecto visual como el comportamiento de la aplicación. Esto involucra la configuración de elementos como el diseño de la interfaz, la tipografía, la paleta de colores, la disposición de elementos en pantalla y las transiciones entre distintas pantallas. La interfaz gráfica debe ser coherente con la identidad visual de la aplicación y cumplir con las directrices de diseño específicas del proyecto (ver figura 14), asegurando así la uniformidad en el conjunto de aplicaciones dentro del ecosistema Android.

4.3.Etapa de Programación

La etapa de programación se posiciona como uno de los cimientos fundamentales en el proceso de desarrollo de software, desempeñando un papel crucial en la materialización de la aplicación en construcción. Esta fase sucede a la etapa de diseño y

precede a las pruebas y la implementación final del producto, y se compone principalmente de dos aspectos centrales: la codificación y las pruebas unitarias.

La codificación constituye el proceso mediante el cual los desarrolladores traducen las previas planificaciones de diseño y arquitectura de software en código fuente ejecutable. Aquí se plasman las instrucciones y se definen las funciones que permitirán que la aplicación cumpla con sus tareas y funciones preestablecidas.

Las pruebas unitarias, por otro lado, emergen como una faceta esencial en esta fase con el propósito de salvaguardar la calidad del código. Estas pruebas implican la evaluación de componentes individuales o "unidades" de código para certificar su adecuado funcionamiento. Cada función, método o clase escrita en el código es sometida a un riguroso proceso de pruebas para identificar y subsanar posibles errores o comportamientos inesperados. Las pruebas unitarias permiten a los desarrolladores descubrir y remediar problemas antes de que se propaguen a otras áreas del software, lo que conlleva ahorro de tiempo y recursos en etapas posteriores del desarrollo.

4.3.1. Codificación

En la etapa de codificación, se traducen los conceptos y diseños previamente planificados en código fuente ejecutable. Esta fase implica escribir líneas de código en Kotlin, el lenguaje de programación escogido por sus características positivas en el capítulo anterior.

Durante la codificación en Kotlin, se crean clases, funciones, métodos y estructuras de datos que conformarán la base de la aplicación, esto debido a tener un mejor desarrollo del aplicativo, se crean archivos de clases separados según MVVM, razón por la cual se modifican las variables mencionadas en la Tabla 13. Además, implementan patrones de diseño y arquitectura MVVM.

La codificación no solo se trata de escribir código, sino también de mantener prácticas de programación limpias y eficientes, siguiendo estándares y buenas prácticas que faciliten la colaboración y el mantenimiento a largo plazo del código.

4.3.1.1.Codificación Archivos Main

La codificación en archivos Main en Kotlin se refiere a la parte central de una aplicación o programa Kotlin, donde comienza la ejecución del código. Este archivo Main actúa como el punto de entrada de la aplicación y típicamente contiene la función principal (main) que se ejecuta al iniciar el programa. Desde esta función principal se llaman a otras funciones y se inicia el flujo de la aplicación.

Durante la codificación en archivos Main en Kotlin, es común llevar a cabo tareas como la inicialización de variables, la configuración de entornos, la carga de configuraciones iniciales y la preparación general para el correcto funcionamiento de la aplicación.

Un elemento fundamental en la codificación de archivos Main es asegurarse de que la función main o el punto de entrada principal sean claros y eficientes. Esto implica organizar el código de manera que sea fácil de comprender y mantener. Además, es crucial gestionar excepciones y errores de forma adecuada para prevenir fallos inesperados en la aplicación.

Para el desarrollo del proyecto se crean tres archivos Main que van a contener diferentes aspectos relevantes del aplicativo, el primero el MainActivity, el cual es la parte principal de la aplicación, aquí se programan todos los elementos del aplicativo, el MainFragment, que se encarga de la interacción entre la interfaz gráfica y las interacciones con el chip NFC X3, y el archivo MainViewModel, que proporciona métodos y flujos para interactuar con estos aspectos desde otras partes de la aplicación.

4.3.1.1.1. MainActivity

El código para este apartado (ver Anexo 4) es la parte primordial del aplicativo a desarrollarse.

En primer lugar, se inician las importaciones de las bibliotecas necesarias para el funcionamiento de NFC, junto con otras bibliotecas estándar de Android. Este paso es necesario para garantizar el acceso a las funcionalidades requeridas en la aplicación.

La clase principal que rige la actividad principal de la aplicación se denomina "MainActivity". Este componente es el punto de partida de la aplicación cuando se abre y es donde se realiza la configuración inicial y se maneja la interacción del usuario. En su construcción, se establece un objeto compañero (companion object) que contiene una constante llamada "TAG". Esta constante se utiliza para etiquetar los mensajes de registro (logs) en el proceso de depuración.

Dentro de la clase MainActivity se definen varias variables de instancia, como "binder", relacionada con la interfaz de usuario, y "viewModel", una instancia de MainViewModel. Esta última se utiliza para gestionar los datos y la lógica de la aplicación siguiendo el patrón arquitectónico MVVM (Modelo-Vista-ViewModel).

El método "onCreate" se encarga de la inicialización de la actividad. Aquí se configura la interfaz de usuario utilizando DataBindingUtil para elevarla desde el archivo XML "activity_main.xml". Se establece el ViewModel y se determina el "dueño del ciclo de vida", que en este caso es la propia actividad. Asimismo, se configura un "OnCheckedChangeListener" en el botón denominado "toggleButton" para supervisar los cambios en su estado, tal y como se muestra en la Figura 15.

Figura 15: Métodos creados para la configuración de la interfaz gráfica

```
// Establece el ViewModel y el ciclo de vida
binder?.setViewModel(viewModel)
binder?.setLifecycleOwner(this@MainActivity)
super.onCreate(savedInstanceState)
// Configura el listener para el botón de alternancia
binder?.toggleButton?.setOnCheckedChangeListener(this@MainActivity)
// Inicia corutinas para realizar operaciones asíncronas
Coroutines.main( activity: this@MainActivity, { scope : CoroutineScope ->
```

La gestión de eventos y otros procesos iniciales, como la preparación de la interfaz de usuario, también se llevan a cabo en este método. Es esencial destacar que se utilizan corutinas de la biblioteca kotlinx.coroutines para realizar operaciones asíncronas y mantener una interfaz de usuario receptiva. Se ejecutan tres corutinas para observar y manejar diferentes aspectos de la aplicación, como el estado NFC, los mensajes de notificación emergente y las etiquetas NFC detectadas.

El método "onCheckedChanged" (ver Figura 16) se activa cuando se produce un cambio en el estado del botón "toggleButton". Si el cambio proviene de este botón, se ejecuta el método "onCheckNFC" en el ViewModel, lo que indica que el usuario ha habilitado o deshabilitado la funcionalidad NFC.

Figura 16: Método onCheckedChanged que almacena cambios en el hardware NFC

```
override fun onCheckedChanged(buttonView : CompoundButton?, isChecked : Boolean) {
    if (buttonView == binder?.toggleButton)
    // Actualiza el estado NFC en el ViewModel
        viewModel.onCheckNFC(isChecked)
}
```

Por último, el método "onTagDiscovered" (ver Figura 17)se desencadena cuando se detecta una etiqueta NFC. En este punto, la etiqueta se envía al ViewModel para su procesamiento.

Figura 17: Método para detectar el chip implante NFC X3

```
override fun onTagDiscovered(tag : Tag?) {
    // Lee los datos de la etiqueta y actualiza el ViewModel
    binder?.getViewModel()?.readTag(tag)
}
```

El código constituye la base de la aplicación Android que aprovecha la tecnología NFC para interactuar con etiquetas NFC que para el caso del proyecto de titulación es el chip implante NFC X3. La colaboración entre las clases MainActivity, MainViewModel y NFCManager permite habilitar la funcionalidad NFC del dispositivo y la lectura de etiquetas NFC. Además, el uso de patrones de diseño y corutinas contribuye a mantener un código organizado, de está manera se cumple con la metodología de refactorización de código que se propuso en el capítulo anterior.

4.3.1.1.2. MainFragment

Para este archivo Main, la codificación representa una parte crucial de la aplicación Android dedicada a la gestión de la interfaz de usuario y las interacciones relacionadas con la lectura de etiquetas NFC. (ver Anexo 5)

La clase "MainFragment" se define como un fragmento de la aplicación Android que, además, implementa la interfaz "CompoundButton.OnCheckedChangeListener". Esta implementación tiene el propósito de manejar eventos asociados al botón y los cambios que existan al habilitar o inhabilitar el hardware NFC del dispositivo.

Dentro de la clase, se crea un "objeto compañero" (companion object) que facilita la creación eficiente de instancias de "MainFragment."

La función "newInstance()" se presenta como una función estática que desencadena la creación y retorno de una nueva instancia de "MainFragment."

Se declara una variable denominada "binder," la cual se utiliza para vincular la interfaz de usuario y los datos presentes en el fragmento.

Además, se define la variable "viewModel" mediante el uso de "lazy", lo que permite acceder al ViewModel asociado al fragmento a través de "ViewModelProvider". Este ViewModel es esencial para gestionar los datos y la lógica de la aplicación de manera efectiva.

El método "onCreateView()" se ejecuta al crear la vista del fragmento. Aquí, se carga la interfaz de usuario desde el archivo "fragment_main.xml", se configura el ViewModel y se establece el ciclo de vida del fragmento antes de devolver la vista resultante (ver Figura 18).

Figura 18: Método para ejecutar la vista del fragmento dentro de la GUI.

```
override fun onCreateView(inflater : LayoutInflater, container : ViewGroup?, savedInstanceState : Bundle?) : View? {
    binder = DataBindingUtil.inflate(inflater, R.layout.fragment_main,container, attachToParent: false)
    binder?.setViewModel(viewModel)
    binder?.setLifecycleOwner(this@MainFragment)
    return binder?.root ?: super.onCreateView(inflater, container, savedInstanceState)
}
```

El método "onViewCreated()" (ver Figura 19) entra en acción una vez que la vista se ha creado por completo. En este punto, se configuran las corutinas para llevar a cabo operaciones asíncronas, como la presentación de mensajes de Toast (Notificaciones Breves de Android) y la actualización de la interfaz de usuario con datos provenientes de etiquetas NFC.

Figura 19: Método de creación de vistas para configurar corutinas.

```
override fun onViewCreated(view : View, savedInstanceState : Bundle?) {
    Coroutines.main( fragment: this@MainFragment, { scope : CoroutineScope ->
        scope.launch( block = { binder?.getViewModel()?.observeToast()?.collectLatest ( action = { message : String? ->
            Toast.makeText(requireContext(), message, Toast.LENGTH_LONG).show()
        }) })
        scope.launch( block = { binder?.getViewModel()?.observeTag()?.collectLatest ( action = { tag : String? ->
            binder?.textViewExplanation?.setText(tag)
        }) })
    })
    super.onViewCreated(view, savedInstanceState)
}
```

Finalmente, el método "onCheckedChanged()" (ver Figura 20) se activa en respuesta a un cambio en el estado del botón de alternancia ("toggleButton"). Si dicho cambio proviene de este botón, se llama a "viewModel.onCheckNFC()" para actualizar el estado NFC en el ViewModel.

Figura 20: Método para verificar cambios en el hardware NFC

```
override fun onCheckedChanged(buttonView : CompoundButton?, isChecked : Boolean) {
    if (buttonView == binder?.toggleButton)
        viewModel.onCheckNFC(isChecked)
}
```

El código de este archivo Main, despliega una función crítica al gestionar la interfaz de usuario y las interacciones relacionadas con la lectura de etiquetas NFC. Su labor incluye elevar la interfaz, mostrar mensajes de Toast y actualizar la interfaz con datos

NFC cuando sea necesario, además de supervisar los cambios en el estado del botón de alternancia.

4.3.1.1.3. MainViewModel

El código del archivo MainViewModel (ver Anexo 6) desempeña un papel importante en la aplicación Android, al ser responsable de la comunicación con dispositivos NFC. Su función principal radica en abstractizar la complejidad inherente a la interacción NFC, al tiempo que ofrece una interfaz unificada para la gestión de esta tecnología. Esto simplifica el desarrollo de funciones relacionadas con NFC, como la lectura de etiquetas y el seguimiento del estado NFC, al tiempo que mejora la experiencia global del usuario.

Este ViewModel realiza un seguimiento constante del estado NFC en el dispositivo, verificando que esté habilitado y sea compatible. Proporciona métodos para la lectura de etiquetas NFC, interpretando los datos contenidos en estas etiquetas, como su identificación y tipo. Además, se encarga de manejar la presentación de mensajes Toast, lo que facilita la retroalimentación efectiva al usuario. Para su comunicación interna, hace uso de flujos de datos estructurados y administra excepciones relacionadas con NFC, como la falta de soporte en el dispositivo.

El MainViewModel simplifica y centraliza la gestión de NFC en una aplicación Android, al ofrecer una capa de abstracción eficaz para interactuar con esta tecnología. Es así como en la Tabla 14, se describen cada uno de los elementos/clases que tiene este archivo y cuál es su función dentro del aplicativo.

Tabla 15: Descripción de los elementos/clases del archivo MainViewModel

Elemento / Clase	Descripción
Importaciones	Importa las clases y bibliotecas necesarias para trabajar con NFC, así como bibliotecas estándar de Android.
Clase *MainViewModel*	Clase que extiende *AndroidViewModel* y administra la lógica relacionada con etiquetas NFC en la aplicación.
Variables estáticas	Define dos variables estáticas: TAG y prefix.
Inicialización de flujos	Inicializa tres flujos (*liveNFC*, *liveToast*, *liveTag*) usando *MutableStateFlow* y *MutableSharedFlow* para transmitir información.
Constructor	Constructor que toma una instancia de Application como argumento. Inicializa flujos y registra mensajes en el Log.
Métodos para mensajes Toast	- *updateToast*(message: String): Actualiza y emite un mensaje Toast. - *postToast*(message: String): Emite un mensaje Toast. - *observeToast*(): Proporciona un flujo compartido (SharedFlow) para observar mensajes Toast.
Métodos para configuraciones NFC	- *getNFCFlags*(): Devuelve combinación de banderas NFC. - *getExtras*(): Recupera opciones extras NFC en forma de Bundle.
Métodos relacionados con NFC	- *onCheckNFC*(isChecked: Boolean): Maneja cambios en el estado NFC y muestra mensajes Toast. - *readTag*(tag: Tag?): Lee datos de etiquetas NFC y los procesa. - *updateNFCStatus*(status: NFCStatus): Actualiza estado NFC y muestra mensajes si es necesario. - *postNFCStatus*(status: NFCStatus): Publica estado NFC y muestra mensajes si no está habilitado o compatible. - *observeNFCStatus*(): Proporciona flujo de estado NFC (StateFlow) para observar el estado NFC. - *getDateTimeNow*(): Obtiene fecha y hora actual en formato específico.
Métodos para información de etiquetas	- *getHex*(bytes: ByteArray): Convierte bytes en representación hexadecimal. - *getDec*(bytes: ByteArray): Convierte bytes en número decimal. - *getReversed*(bytes: ByteArray): Convierte bytes invertidos en número decimal. - *observeTag*(): Proporciona flujo de información de etiqueta (StateFlow) para observar datos de etiquetas.

4.3.1.2.Codificación Corutinas y NFC

En el contexto del desarrollo de aplicaciones Android, las Corutinas desempeñan un papel destacado al simplificar la ejecución de tareas asincrónicas. En contraste con enfoques anteriores que requerían el uso de callbacks complicados o clases como

AsyncTask, las Corutinas permiten escribir código de manera secuencial, lo que mejora significativamente la legibilidad y facilita la administración de tareas concurrentes. En el código proporcionado, estas Corutinas se emplean para llevar a cabo acciones esenciales, como la presentación de mensajes Toast y la actualización del estado NFC de manera asincrónica, lo que resulta fundamental para proporcionar una experiencia de usuario fluida y eficiente.

Por otro lado, el NFC Manager representa una pieza clave en las aplicaciones Android que requieren interactuar con la tecnología NFC. Este componente proporciona métodos y herramientas destinados a verificar la habilitación del NFC en el dispositivo, gestionar la lectura de etiquetas NFC y controlar el estado general de la tecnología NFC. Esta funcionalidad es crítica para garantizar el funcionamiento correcto y seguro de la aplicación en lo que respecta a las interacciones NFC.

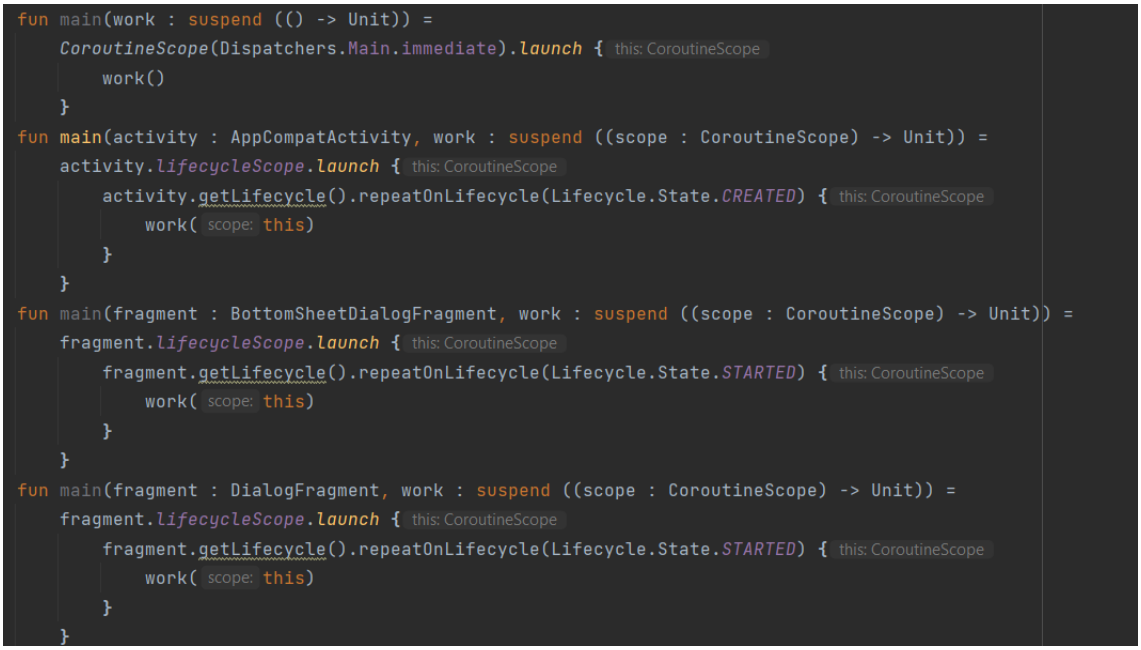
El concepto de NFC Status se refiere a la representación del estado actual de la tecnología NFC en un dispositivo Android. Este estado puede asumir diversas categorías, como "Habilitado", "Deshabilitado" o "Lectura en Proceso", y se utiliza como indicador de la situación del NFC en un momento específico. Por ejemplo, este estado es fundamental para determinar si NFC está habilitado o no, lo que puede resultar en la toma de decisiones, como mostrar un mensaje en caso de que NFC esté desactivado o iniciar la lectura de etiquetas cuando está activado.

4.3.1.2.1. Corutinas

En la programación de corutinas (ver Anexo 7), se define un objeto llamado "coroutines", el cual ofrece funciones diseñadas para administrar de manera efectiva tareas asincrónicas en el contexto de una aplicación Android.

El código se compone de una serie de funciones esenciales, siendo "main" (ver Figura 21) y sus variantes las más destacadas. Estas funciones tienen la finalidad de ejecutar tareas en el hilo principal de la interfaz de usuario (UI thread), un ámbito crítico donde se realizan las actualizaciones de la interfaz de usuario. "main" se adapta a diversos componentes de la interfaz de usuario, como AppCompatActivity, BottomSheetDialogFragment, DialogFragment y Fragment, garantizando una gestión adecuada del ciclo de vida para cada uno.

Figura 21: Funciones Main del código Coroutines donde se ejecutan las tareas principales.

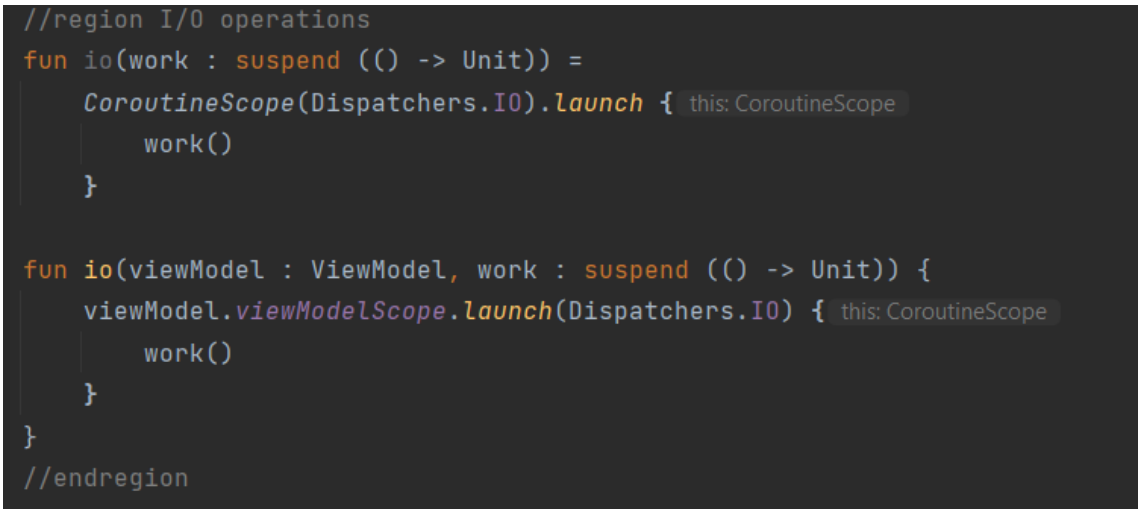


La función "io" (ver Figura 22) se utiliza para ejecutar tareas relacionadas con operaciones de entrada/salida (I/O) en un hilo de fondo[1], evitando bloqueos que puedan comprometer la capacidad de respuesta de la aplicación. Esto resulta fundamental en

[1] En el contexto de la programación, un hilo de fondo es un hilo separado del hilo principal de una aplicación. Está diseñado para llevar a cabo tareas en segundo plano sin bloquear la interfaz de usuario (UI). Estos hilos se utilizan para ejecutar operaciones de larga duración, como operaciones de entrada/salida (I/O) o procesamiento intensivo, al mismo tiempo que se mantiene la capacidad de respuesta de la aplicación.

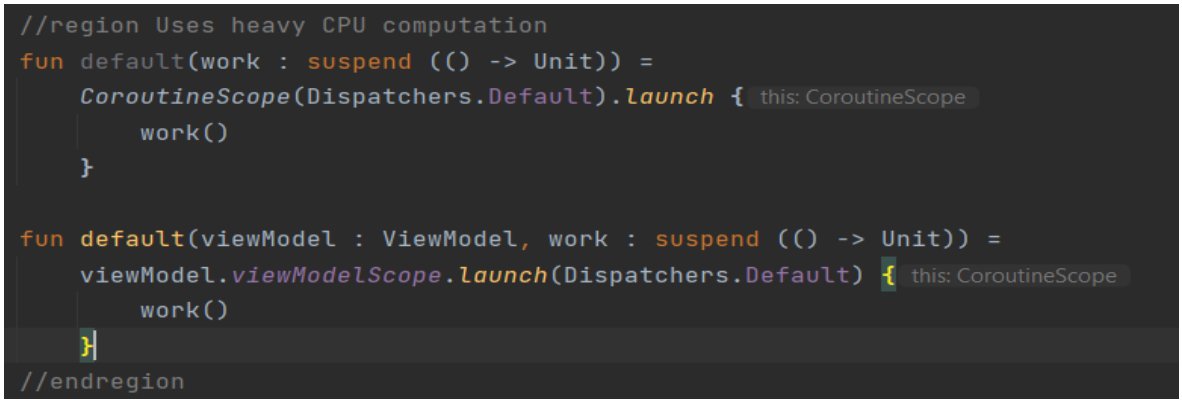
operaciones lentas y bloqueantes, como lectura o escritura de archivos y solicitudes de conexión con el hardware NFC.

Figura 22: Métodos de entrada y salida para evitar bloqueos



Asimismo, la función "default" (ver Figura 23) desempeña un papel similar a "io", pero está orientada a tareas que involucran una intensa computación en la unidad central de procesamiento (CPU). Su implementación en un hilo de fondo independiente evita que tales operaciones afecten el hilo principal y, por ende, la experiencia del usuario.

Figura 23: Función default donde se hace el trabajo de procesamiento en la CPU.



Por último, "unconfined" se utiliza en situaciones donde no es necesario un contexto de hilo específico para ejecutar una tarea, brindando flexibilidad en cuanto a la asignación de hilos.

4.3.1.2.2. NFC Manager y NFC Status

El Archivo NFC Manager (ver Anexo 8), tiene un código que en cuestión introduce un objeto denominado NFCManager, cuya finalidad es ofrecer funciones destinadas a supervisar tanto el estado como la funcionalidad de la tecnología en una aplicación Android. En este contexto, el mencionado objeto cumple diversas funciones, cada una de las cuales desempeña un papel específico en la gestión de NFC.

La función "enableReaderMode" se emplea para activar el denominado "modo lector" NFC en la aplicación. Entre sus parámetros se incluyen el contexto de la aplicación, la actividad actual, un objeto NfcAdapter.ReaderCallback encargado de gestionar las lecturas NFC, así como un conjunto de banderas (flags) y extras que configuran el mencionado modo lector.

La función "disableReaderMode", por su parte, desempeña la tarea opuesta, desactivando el "modo lector" NFC en la aplicación. Requiere como parámetros el contexto de la aplicación y la actividad actual.

Para determinar si el dispositivo es compatible con NFC, se recurre a la función "isSupported". Esta devuelve "true" en caso afirmativo y "false" en caso negativo. Asimismo, la función "isNotSupported" complementa a la anterior, proporcionando "true" en caso de incompatibilidad y "false" en caso de compatibilidad.

La funcionalidad de habilitación de NFC se encuentra en las funciones "isEnabled" (que devuelve "true" si NFC está habilitado) y "isNotEnabled" (que, al igual que su homóloga, complementa la función "isEnabled").

Finalmente, la función "isSupportedAndEnabled" combina las verificaciones de compatibilidad y habilitación de NFC, devolviendo "true" si el dispositivo es compatible con NFC y NFC se encuentra habilitado.

El archivo NFC Status (ver Anexo 9) en cambio, nos retorna los diferentes estados y banderas que puede tener un lector NFC. El código en cuestión introduce una numeración (enum) denominado "NFCStatus" que cumple con la función de representar varios estados asociados con la funcionalidad NFC dentro del contexto de una aplicación Android. Cada uno de los valores incluidos en este enum tiene una interpretación específica.

En primer lugar, se tiene el estado "NoOperation", que refleja que en el momento actual no se está llevando a cabo ninguna operación relacionada con NFC, lo que puede considerarse como una especie de estado inactivo o predeterminado. Seguidamente, "Tap" indica que se detecta una etiqueta NFC y se inicia una operación, ya sea de

lectura o escritura. Este estado puede considerarse el punto de partida de una operación NFC.

Por otro lado, "Process" representa el estado en el que la aplicación se encuentra procesando la información obtenida de una etiqueta NFC. Esto puede abarcar tareas como la interpretación de datos o la ejecución de acciones basadas en la información de la etiqueta.

Luego existe el estado "Confirmation", que se utiliza para indicar que la aplicación está a la espera de la confirmación por parte del usuario antes de ejecutar una acción basada en la etiqueta NFC. Un ejemplo sería una aplicación de pago NFC que requiere la confirmación antes de completar una transacción.

El estado "Read" indica que la información almacenada en la etiqueta NFC ha sido leída exitosamente. La información puede encontrarse en un estado listo para ser utilizada o puede requerir procesamiento adicional. En cuanto a "Write", este estado se utiliza cuando se está escribiendo información en una etiqueta NFC, como en el caso de programar una etiqueta NFC con datos específicos.

Finalmente, "NotSupported" indica que la funcionalidad NFC no es compatible con el dispositivo en el cual se está ejecutando la aplicación, una situación que es común en dispositivos que carecen de hardware NFC. Por último, "NotEnabled" refleja que NFC se encuentra desactivado en el dispositivo. Ante esta situación, la aplicación puede optar por mostrar un mensaje que invite al usuario a activar NFC para poder continuar.

4.3.2. Pruebas Unitarias

Las pruebas unitarias, también conocidas como "unit testing," constituyen un método esencial en el desarrollo del software, donde se examinan minuciosamente unidades individuales de código, como funciones o métodos, de forma aislada para

garantizar su funcionamiento conforme a las especificaciones definidas. Estas pruebas automatizadas, esenciales en la metodología de Desarrollo Basado en Pruebas (TDD), son fundamentales para asegurar la calidad del código, identificar errores tempranamente y facilitar la depuración, contribuyendo así a mantener la integridad del software durante su evolución y actualización.

Dentro de ese contexto, se llevan a cabo pruebas integrales en los códigos implementados con el objetivo de garantizar su funcionamiento sin errores. En el caso de NFCManager, se evalúa la eficacia de las funciones relacionadas con la habilitación y deshabilitación del modo lector NFC, así como la gestión adecuada de excepciones. Se verifica la detección de la compatibilidad del dispositivo y el estado de habilitación de NFC, todas ellas a través de su interacción con el hardware NFC y el análisis de su estado.

Para el código relacionado con Coroutines, se ejecuta pruebas en cada contexto de corutina disponible. Se confirma que las corutinas en el hilo principal se ejecutan de manera apropiada y pueden gestionar funciones suspendidas de manera efectiva intercambiando corutinas del tipo main al realizar la lectura de cada etiqueta NFC. Se realizan pruebas en el hilo predeterminado para evaluar la capacidad de las corutinas para manejar operaciones que requieren una intensidad significativa de CPU añadiendo fragmentos de código que son utilizados por el archivo MainFragment para realizar un estrés a nivel de CPU. Además, se verifica el funcionamiento adecuado de las corutinas sin contexto específico mediante adición de Toast, los cuales no están presentes en ningún contexto especifico del código.

En relación con el código NFCStatus, se efectúa una prueba para confirmar que la enumeración NFCStatus se declara con precisión, de acuerdo con los valores

previamente definidos, mediante los mensajes de Toast y los fragmentos de códigos ejecutados, en donde son visibles cada uno de estos estados.

Por último, en los códigos de MainActivity y MainFragment, se someten a pruebas detalladas con el fin de garantizar una inicialización apropiada del ViewModel en ambas clases. Esto es de vital importancia para lograr una conexión efectiva entre la lógica de programación y la interfaz de usuario. Se verifica que las vistas se elevan de manera correcta desde sus respectivos archivos de diseño. Asimismo, se evalúan las observaciones de mensajes Toast y de datos de etiquetas para asegurarse de su funcionamiento sin inconvenientes. Se garantiza que los cambios en el estado del botón de alternancia se reflejen de manera precisa en el ViewModel y se gestionen de forma adecuada en ambas clases.

4.4.Etapa de Pruebas

La fase de pruebas se erige como un componente importante en el ciclo de despliegue del sistema. En este proceso, se evalúa y valida el software para garantizar su conformidad con las especificaciones y requisitos establecidos. Esta fase se divide en tres etapas fundamentales: la evaluación, la emulación y la ejecución en dispositivos reales.

En la etapa de evaluación inicial, se efectúa un escrutinio del software para identificar posibles deficiencias o fallos. Este proceso puede abarcar la revisión manual del código, la realización de pruebas estáticas para analizar la calidad del código y la evaluación de documentos de diseño y requisitos. El propósito subyacente es la detección temprana de errores o discrepancias antes de que el software avance a las fases subsiguientes de pruebas.

En la fase de emulación, se crea un entorno de prueba que replica las condiciones en las cuales el software operará, si bien sin involucrar hardware físico real. Esto se realiza para evaluar el rendimiento y la compatibilidad del software en escenarios específicos. Por ejemplo, es factible emular diversos dispositivos, sistemas operativos o configuraciones de red para evaluar el comportamiento del software en tales condiciones simuladas.

La ejecución en dispositivos reales constituye la etapa en la cual se ejecutan pruebas concretas en hardware físico o dispositivos finales. Esta fase es vital para validar el funcionamiento del software en un entorno real y para detectar posibles problemas que podrían no haber sido evidentes en las fases previas.

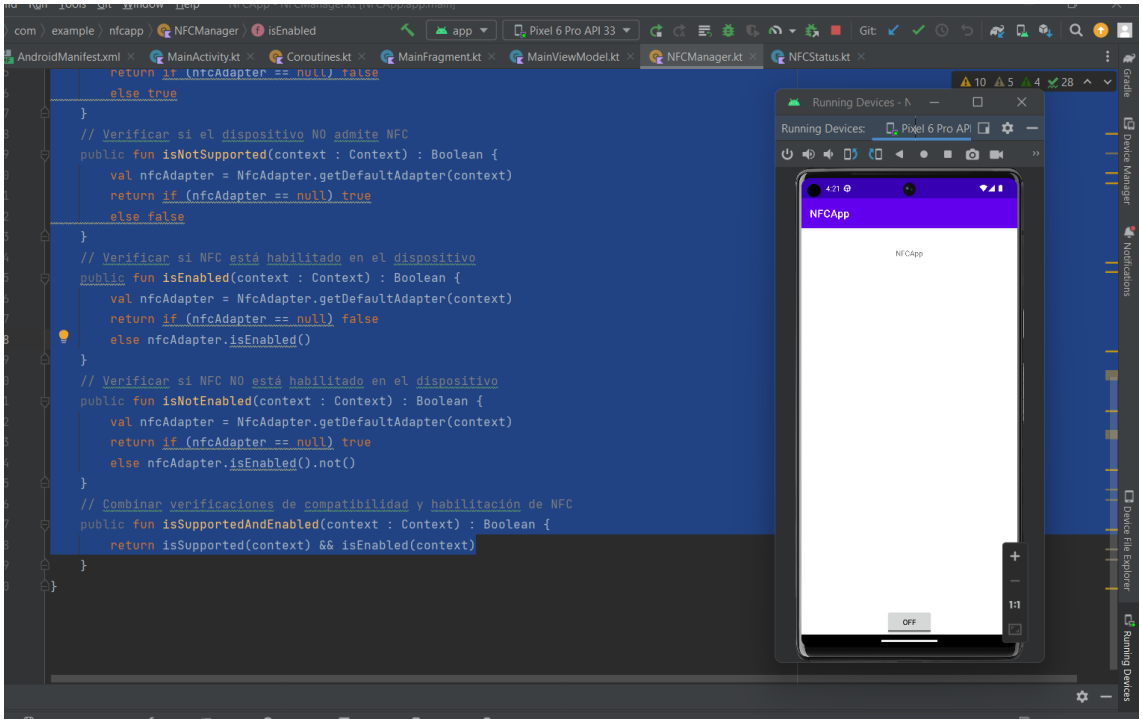
4.4.1. Evaluación del Software

La evaluación concreta de los códigos realizados, relacionados con la gestión de NFC, requiere una serie de pasos destinados a garantizar la correcta implementación de la funcionalidad NFC y su alineación con los requisitos establecidos. A continuación, se presenta una descripción pormenorizada de cómo se lleva a cabo esta evaluación:

- **Prueba de Detección de Etiquetas NFC**: se realiza una evaluación de la capacidad de la aplicación para identificar etiquetas NFC y garantizar un procesamiento y activación adecuados, tal como se ilustra en la Figura 24. Este proceso implica una ejecución parcial del paquete de software en Android Studio, utilizando el modo de depuración. En el lenguaje de programación Kotlin, se establece un punto de interrupción en la línea de código que se desea examinar. Posteriormente, se inicia la aplicación en el modo de depuración al seleccionar la opción “Debug” en lugar de “Run”. Al llegar al punto de interrupción, se utilizan las opciones “Step Over”, “Step Into” y “Step Out” para

controlar la ejecución del código. Durante este proceso, es posible inspeccionar los valores de las variables en tiempo real. Los resultados obtenidos a través de esta evaluación se manifiestan como mensajes Toast en la pantalla del simulador de aplicaciones integrado en Android Studio, específicamente en el dispositivo virtual Pixel 6.

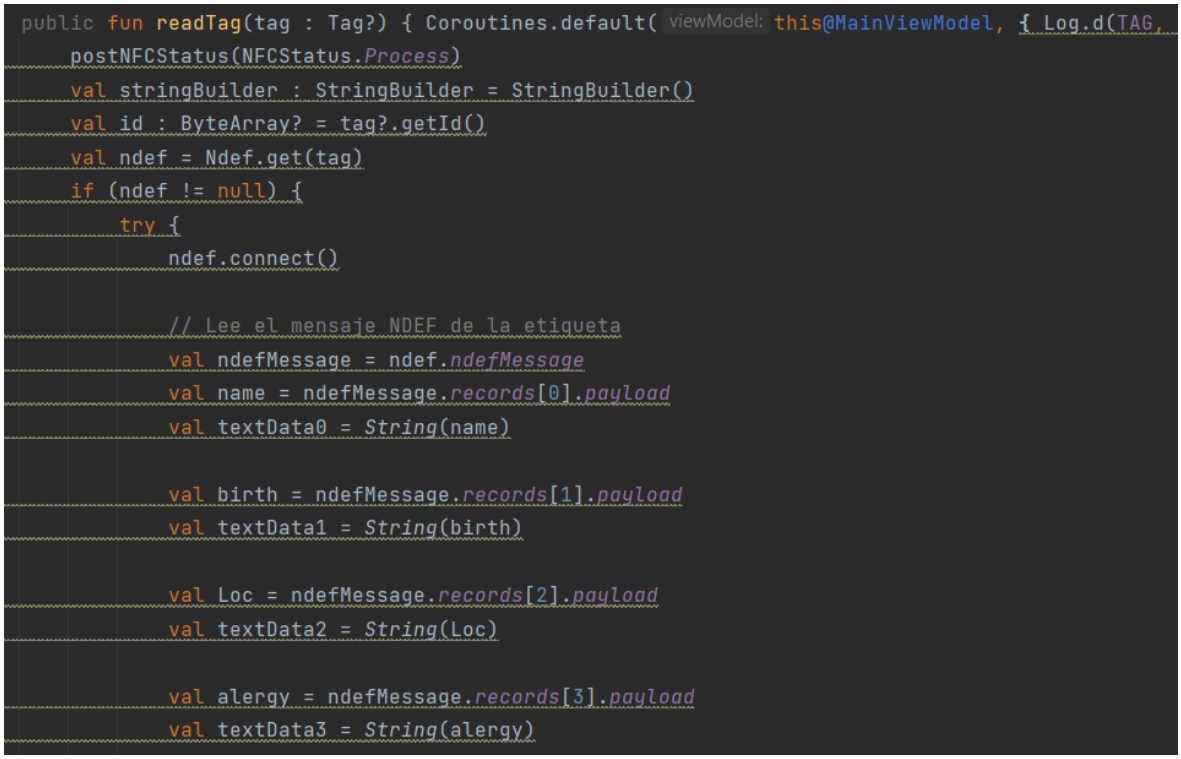
Figura 24: Prueba de Detección de Etiquetas NFC



- **Prueba de Lectura de Datos NFC:** Se procede a la validación de la lectura de datos de las etiquetas NFC detectadas, abarcando una variedad de formatos de datos NDEF y garantizando la correcta interpretación y presentación de los datos, esto verificando que la refactorización del código[2] de lectura de formatos NDEF se está ejecutando de manera correcta y que su escritura no tenga errores (ver Figura 25).

[2] Una práctica esencial en la refactorización del código es el proceso de renombrar variables, se realiza en Android Studio con Kotlin para mejorar la legibilidad del código utilizando la función Refactor.

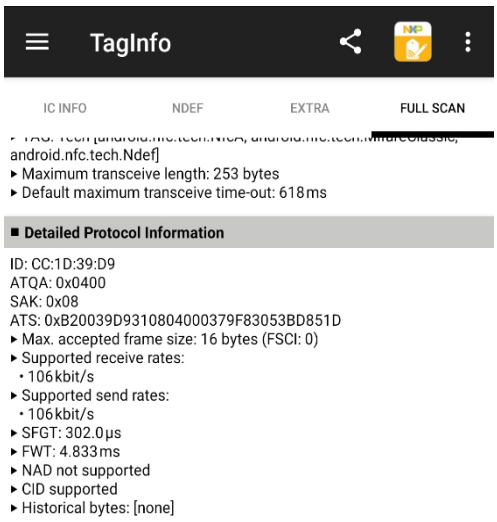
Figura 25: Verificación del funcionamiento del protocolo NDEF.



```
public fun readTag(tag : Tag?) { Coroutines.default( viewModel: this@MainViewModel, { Log.d(TAG,
    postNFCStatus(NFCStatus.Process)
    val stringBuilder : StringBuilder = StringBuilder()
    val id : ByteArray? = tag?.getId()
    val ndef = Ndef.get(tag)
    if (ndef != null) {
        try {
            ndef.connect()

            // Lee el mensaje NDEF de la etiqueta
            val ndefMessage = ndef.ndefMessage
            val name = ndefMessage.records[0].payload
            val textData0 = String(name)

            val birth = ndefMessage.records[1].payload
            val textData1 = String(birth)

            val Loc = ndefMessage.records[2].payload
            val textData2 = String(Loc)

            val alergy = ndefMessage.records[3].payload
            val textData3 = String(alergy)
```

- **Prueba de Rendimiento:** El rendimiento de la aplicación se somete a una evaluación, centrándose en aspectos como la frecuencia, modulación, codificación, la velocidad de detección de etiquetas y el procesamiento de datos, con el objetivo de garantizar una experiencia de usuario receptiva incluso en situaciones de alta demanda. (ver Figura 26-a)

Figura 26-a: Pruebas de Rendimiento del aplicativo en términos de velocidad de detección de datos y procesamiento.



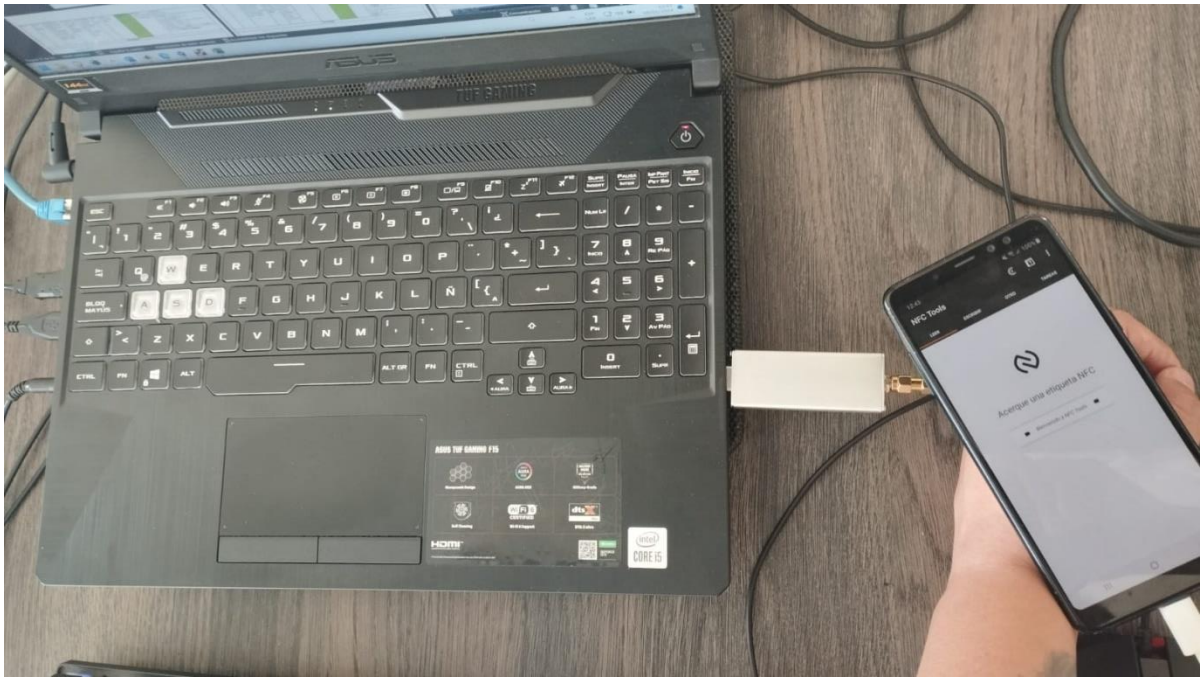
De esta manera se crean los escenarios respectivos para realizar las pruebas globales, la cuales son: análisis y comprobación de capa física de acuerdo con las normas: ISO/IEC 18092, ISO/IEC 14443-2, NFCIP-1, NFC Forum, NDEF, basándose en la norma IEEE/ANSI C63.10-2020, la cual se basa en las pruebas de dispositivos inalámbricos no licenciados, análisis y comprobación de los parámetros de RF y radiación de acuerdo con los valores aceptados por las entidades medicas de control como son: la Administración de alimentos y medicamentos (FDA), Organización Mundial de la Salud (OMS) y Administración de Salud y Seguridad Ocupacional (OSHA); con el objetivo de brindar a la Carrera de Medicina de la Universidad Técnica del Norte los insumos de información necesarios, tanto en aspectos técnicos como en aspectos médicos, para formular la normativa y código de ética relacionado al uso de chips implantes subcutáneos NFC. Finalmente, el análisis y comprobación de capas superiores de acuerdo con las normativas NDEF mediante el uso de un sniffer NFC. (Campos J., 2023)

Para realizar estas pruebas, se detalla previamente el ambiente y escenario global el cual se compone del uso del RTL-SDR con el software AirSpy, una antena

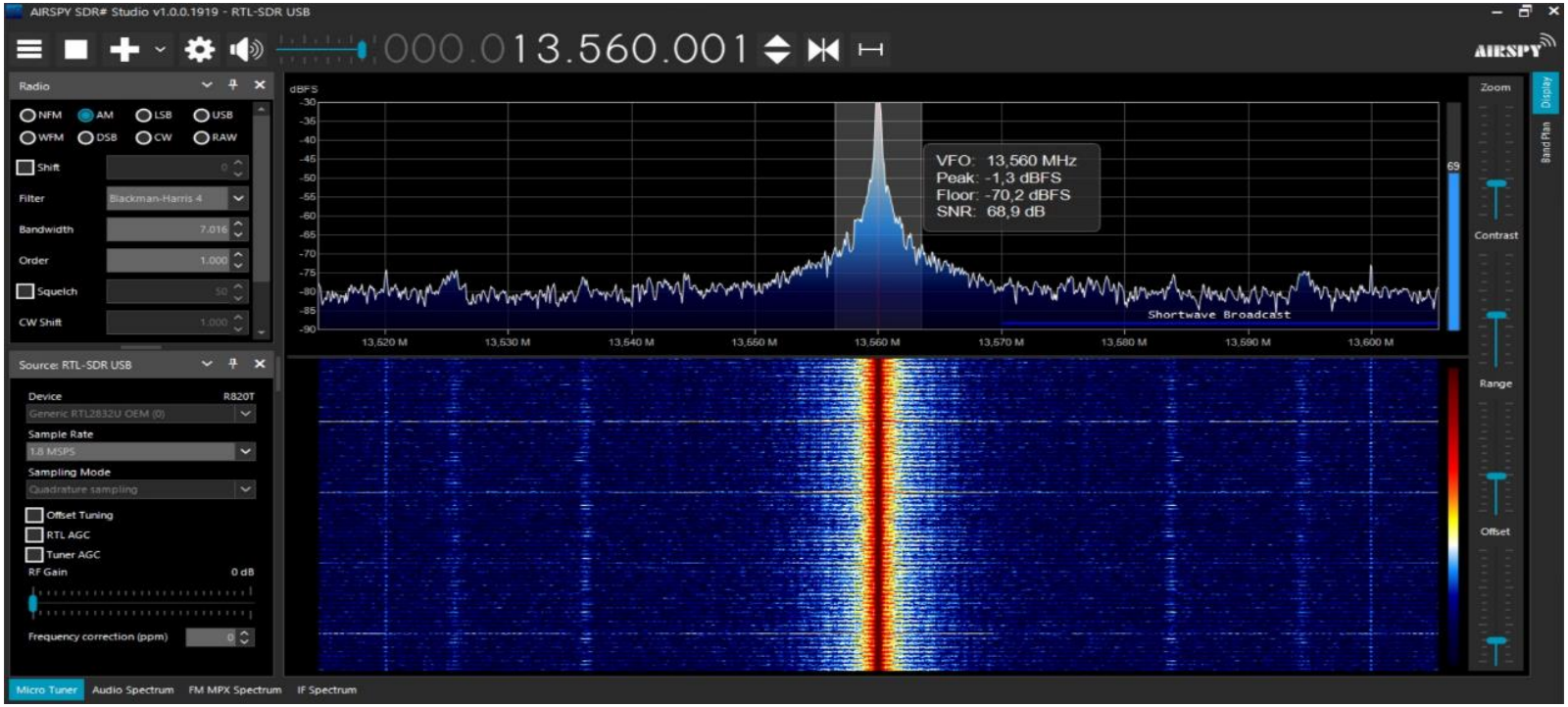
NFC a 13.56 MHz acoplada a 50 Ω con un conector SMA doble para mantener una Relación de Onda Estacionaria (SWR) de 1.1:1 teniendo un acople del 99.7%. El Testbed necesario para realizar dichas pruebas, es un área física en la cual exista una reducción de los fenómenos de alteración de ondas, como lo son: interferencia, ruido y distracciones mínimas; para poder obtener las señales de NFC más cercanas a las transmitidas y recibidas en la comunicación.

El escenario físico global es una antena tipo parche la cual se va a ubicar en el dorso de la mano de una persona entre el chip implante y el dispositivo móvil que hará la lectura NFC, como se muestra en la Figura 26-b.

Figura 26-b: Escenario físico global de pruebas

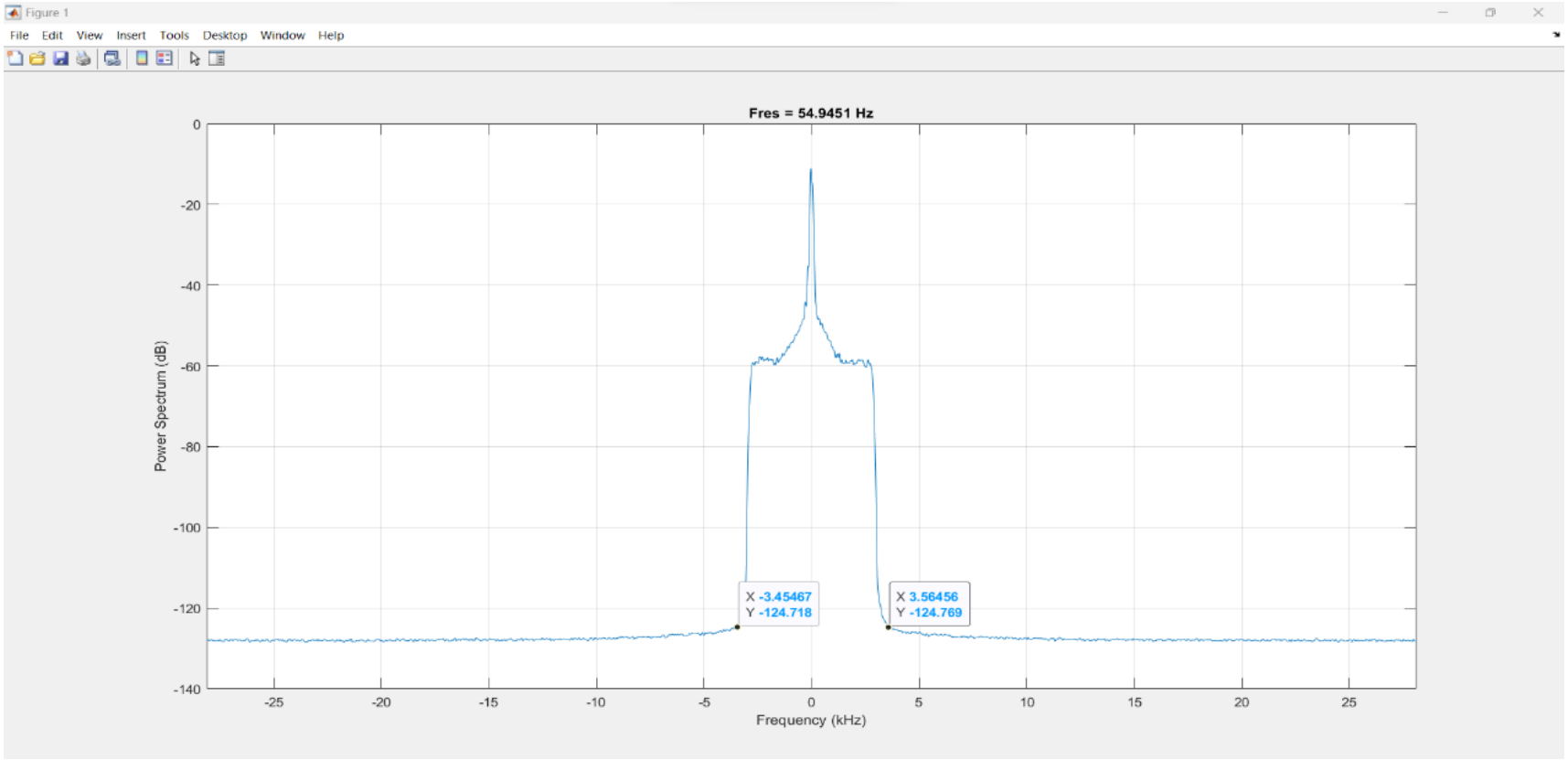


Dentro de este análisis se comprobará diferentes parámetros de radiofrecuencia que se encuentran en las diferentes normas mencionadas anteriormente. En la tabla 15-a se detalla la matriz de pruebas de los parámetros por analizar y comprobar, que son: frecuencia operacional (RF-01), ancho de banda (RF-02), relación señal-ruido (RF-03) y tiempo de transmisión de paquetes (RF-04).

ID	Caso de Prueba	Descripción	Escenario	Objetivo	Resultado Esperado	Resultado Obtenido	Cumplimiento (CUMPLE/NO CUMPLE)	Referencia en la norma	Código/Software Utilizado
RF-01	Frecuencia de operación	La frecuencia de operación teórica de NFC es 13.56 MHz	Recepción de la onda a través del RTL-SDR y verificación de la frecuencia con el software AirSpy	Comprobar que la frecuencia de operación de NFC es 13.56 MHz	Frecuencia central: 13560000 Hz	Frecuencia: 13560001 Hz	CUMPLE	ISO/IEC 14443-2, literal 6.1. Frequency	AirSpy/SDR Sharp
Resultado									
									

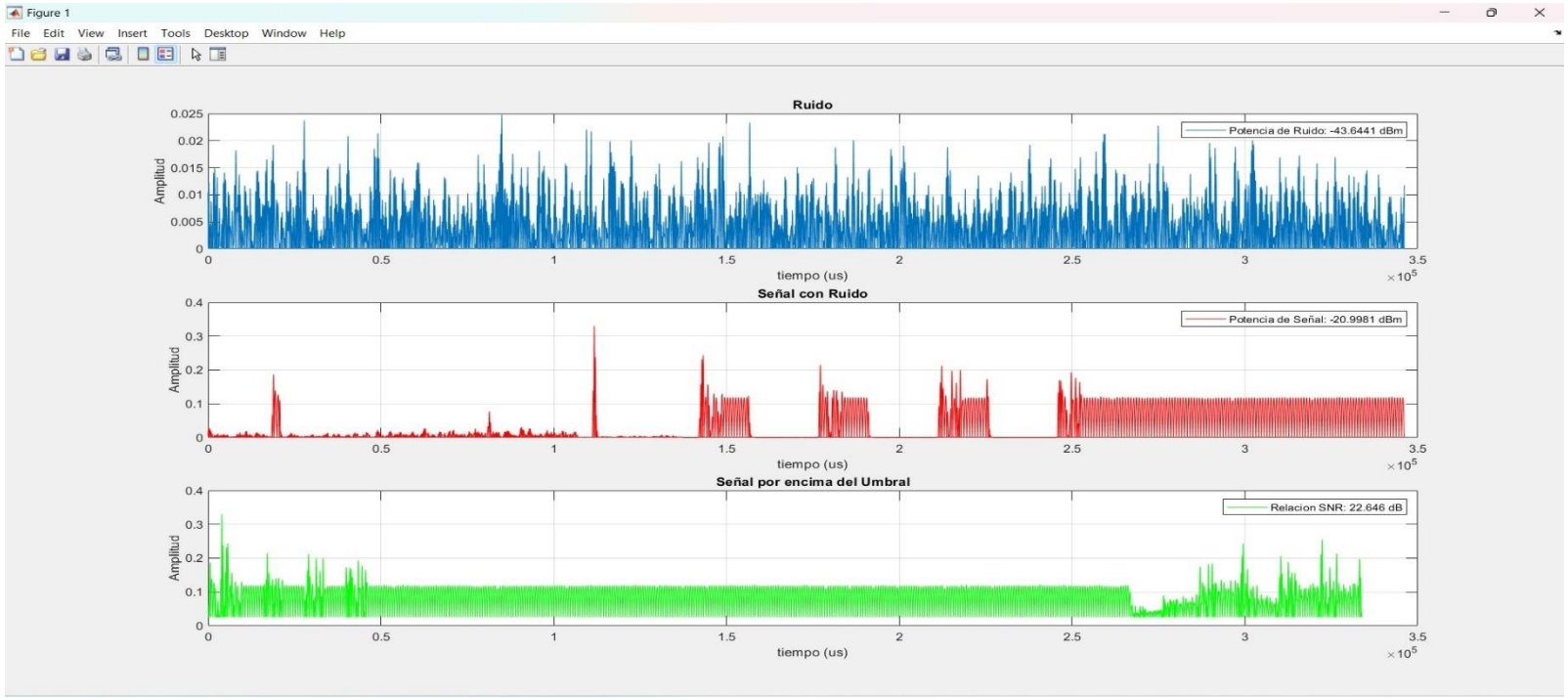
RF-02	Ancho de Banda	El ancho de banda de NFC depende del modo y velocidad de transmisión. Para NFC-A se especifica un AB= 7 KHz	Análisis de la onda grabada con AirSpy en Matlab	Comprobar y verificar el AB al que trabaja NFC-A a una velocidad de 106 Kbps	AB: 7 KHz	AB: 7,01923 KHz	CUMPLE	ISO/IEC 14443-2, literal 6.1. Frequency	MatLab:
									`[data,fs] = audioread('SDRSharp_20231211_163919Z_13560000Hz_IQ.wav');`
									`t = 1:1:size(data(:,1));`
									`N=length(data);`
									`% Seperating the I and Q channels`
									`q_sample = data(:,1) ;`
									`i_sample = data(:,2) ;`
									`IQ_samples = i_sample + 1i.*q_sample ;`
									`pspectrum(IQ_samples,fs,'FrequencyLimits',[-13560000 13560000])`

Resultado



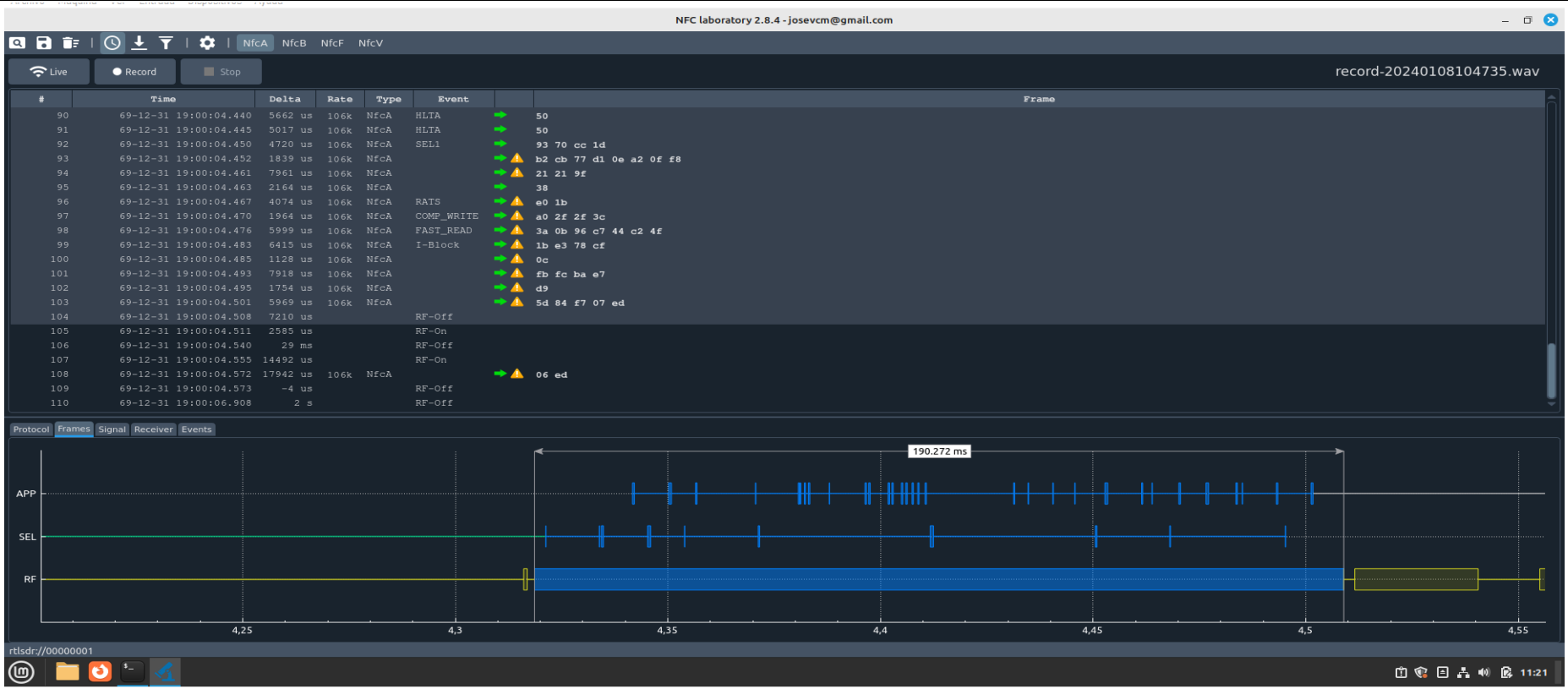
RF-03	Relación Señal a Ruido	El ruido de manera teórica está definido por valores negativos y muy bajos, alteran la señal de este sin influir en la transmisión efectiva del sistema.	Obtención del ruido con el uso de SDR# y comparar con el obtenido en las muestran IQ. Cálculo del campo EM.	Comprobar el valor obtenido del ruido dentro del escenario de análisis, tanto con el uso del SDR como con el procesamiento de las muestras IQ	Amplitud del Ruido: 0.025 Potencia del Ruido: -49dBm SNR: 20 dB Señal: 2.12 a 53.03 mW/m2	Amplitud del Ruido: 0.0246 Potencia del Ruido: -43.64dBm Relación SNR: 22.646dB Señal: 7.95mW/m2	CUMPLE	ISO/IEC 14443-2, literal 6 Power Transfer ISO/IEC 10373-6, literal 4.7 Test Conditions Table 4 y 5	MatLab: https://github.com/CamiloNogalesR/calculossnrnfc/blob/main/CalculoSDR.m

Resultado



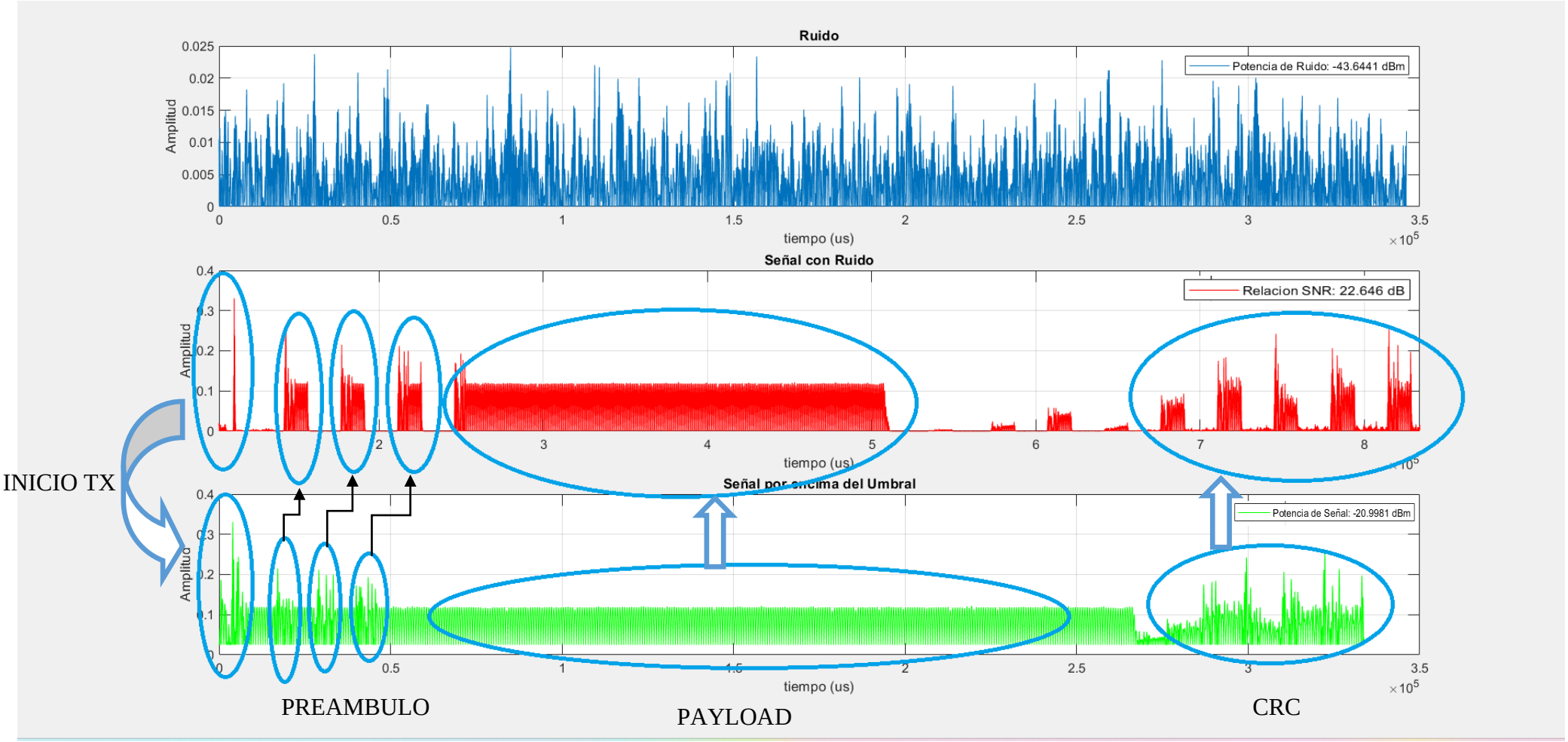
RF-04	Tiempo de Tx de Paquetes	Tiempo de transmisión de la información NFC	Tiempo de trasmisión de información	Saturar el canal con la mayor cantidad de información para determinar el tiempo de transmisión	Tamaño máximo del paquete: 253B * 10 sectores = 2530 Bytes. Vtx=106Kbps. Ttx=190.94 ms	Ttx=190.272 ms	CUMPLE	MIFARE Classic EV1 MF1S50 YYX_V1, literal 8.5 al 9	NFC Sniffer (Campos J., 2023)

Resultado



✓ **Análisis de Relación Señal a Ruido**

Figura 26-c: Análisis del Ruido y la SNR dentro de Matlab



En la Figura 26-c, se evidencia el proceso de obtención de la señal de ruido a partir del dispositivo RTL-SDR Dongle en forma de muestras IQ, las cuales se definen como señales complejas en forma de I+jQ, representando fase y cuadratura respectivamente. Estas componentes I y Q constituyen señales senoidales y cosenoidales, cuya combinación forma la señal capturada por el Dongle.

La aplicación de conceptos matemáticos relacionados con números complejos facilita la determinación del valor máximo de amplitud del ruido, expresado mediante la magnitud (valor absoluto) de la muestra . En este trabajo de titulación, el valor máximo obtenido mediante el software Matlab es 0.0246. Este valor máximo actúa como umbral para distinguir entre los valores de la señal y el ruido. Empleando el software Matlab, se procesa la señal NFC, generando vectores que cumplen la condición esencial: "los valores de amplitud por encima del umbral del ruido pertenecen a la señal".

Concluido el procesamiento, se verifica que la condición establecida mantiene la naturaleza de la señal NFC sin distorsiones significativas, conservando su formato de trama original. El análisis del ruido revela valores de amplitud en el rango de 0 a 0.025, considerados aceptables en los sistemas de telecomunicaciones. A partir de estos valores de amplitud, se determinan los valores de potencia del ruido calculando el promedio de los cuadrados de las amplitudes. Este proceso asegura la precisión y calidad en la identificación y separación de la señal y el ruido en el contexto de la captura de datos NFC, (Soria E., 2010) plantea la Ec. 1 para el cálculo de la potencia promedio de la señal; donde N es el número de muestras, I es la componente en fase y Q la componente en cuadratura:

$$\textbf{Ec. 1} \quad P = \frac{1}{N}\sum_{1}^{N}(|I_N + jQ_N|^2)$$

En el ámbito de las telecomunicaciones, la calidad de la transmisión es de mucha importancia. En este contexto, se verifica que los valores de potencia del ruido se encuentran en el orden de -43.64 dB. De acuerdo con el análisis realizado, tanto con el software SDR# y con el procesamiento de las señales IQ en Matlab, la potencia de ruido es aceptable para este trabajo de titulación, teniendo en cuenta que este valor pertenece a la consideración de ruido térmico, ruido ambiental y ruido impulsivo presente en todas las comunicaciones inalámbricas de corto, mediano y largo alcance.

Una vez obtenido el ruido, se identifica las muestras de amplitud IQ pertenecientes a la señal NFC al capturar con el SDR#. En la figura 26-d, se ilustra 10 muestras de la amplitud para el procesamiento de la señal sin el ruido.

Figura 26-d: Valores de Amplitud de la Señal Obtenida en el SDR#

946176x1 double

	1	2
403	0.0236	
404	0.0237	
405	0.0239	
406	0.0240	
407	0.0241	
408	0.0243	
409	0.0243	
410	0.0245	
411	0.0246	
412	0.0247	
413	0.0248	
414	0.0249	
415	0.0250	

Estos valores tendrán que ser clasificados de acuerdo con el umbral del ruido (0.0246), donde los datos que estén por encima del umbral del ruido pertenecen únicamente a la señal. Una vez clasificados, utilizando la Ec. 1, se procede a calcular el valor de la potencia de la señal, que da como resultado -20.99 dB (7.95mW/m2).

Así, (Stallings, W., 2011) establece la Ec. 2, del cálculo de la SNR, donde S (dBm) es la potencia de la señal, N(dBm) es la potencia del ruido de la señal:

Ec. 2. $SNR(dB) = S(dBm) - N(dBm)$

En el presente trabajo de titulación se obtiene un valor de Relación Señal a Ruido (SNR) de 22.64 dB, el cual constituye un parámetro crítico para evaluar la calidad de la transmisión en sistemas de comunicación NFC. La SNR, medida en decibelios, refleja la proporción entre la potencia de la señal deseada y la potencia del ruido presente en el entorno.

Un valor de SNR de 22.64 dB se considera aceptable en este contexto, indicando que la señal NFC, , situado en el orden de -20.99 dB, es lo suficientemente robusta en comparación con el nivel de ruido de fondo. Este resultado favorable sugiere que la transmisión de datos puede llevarse a cabo de manera eficiente, respaldando así la integridad y confiabilidad de la comunicación inalámbrica en el sistema evaluado.

Es importante recalcar que los valores obtenidos por el software SDR# son valores normalizados. Estos valores se ajustan de acuerdo con las escalas de visualización del dispositivo RTL-SDR Dongle. La normalización es un proceso que se realiza por varias razones. En primer lugar, facilita el manejo de los valores, ya que estos siempre están en el mismo rango. Esto simplifica el procesamiento posterior de los datos, ya que se pueden aplicar los mismos algoritmos y técnicas de análisis a todos los valores.

En segundo lugar, la normalización mantiene la precisión de los valores de amplitud de la señal. Esto es importante para la correcta demodulación de las señales, ya que cualquier error en la amplitud puede resultar en una interpretación incorrecta de los datos transmitidos.

El proceso de normalización de una señal obtenida en SDR# depende de varios factores tales como: ganancia de RF, características del Dongle, características de la

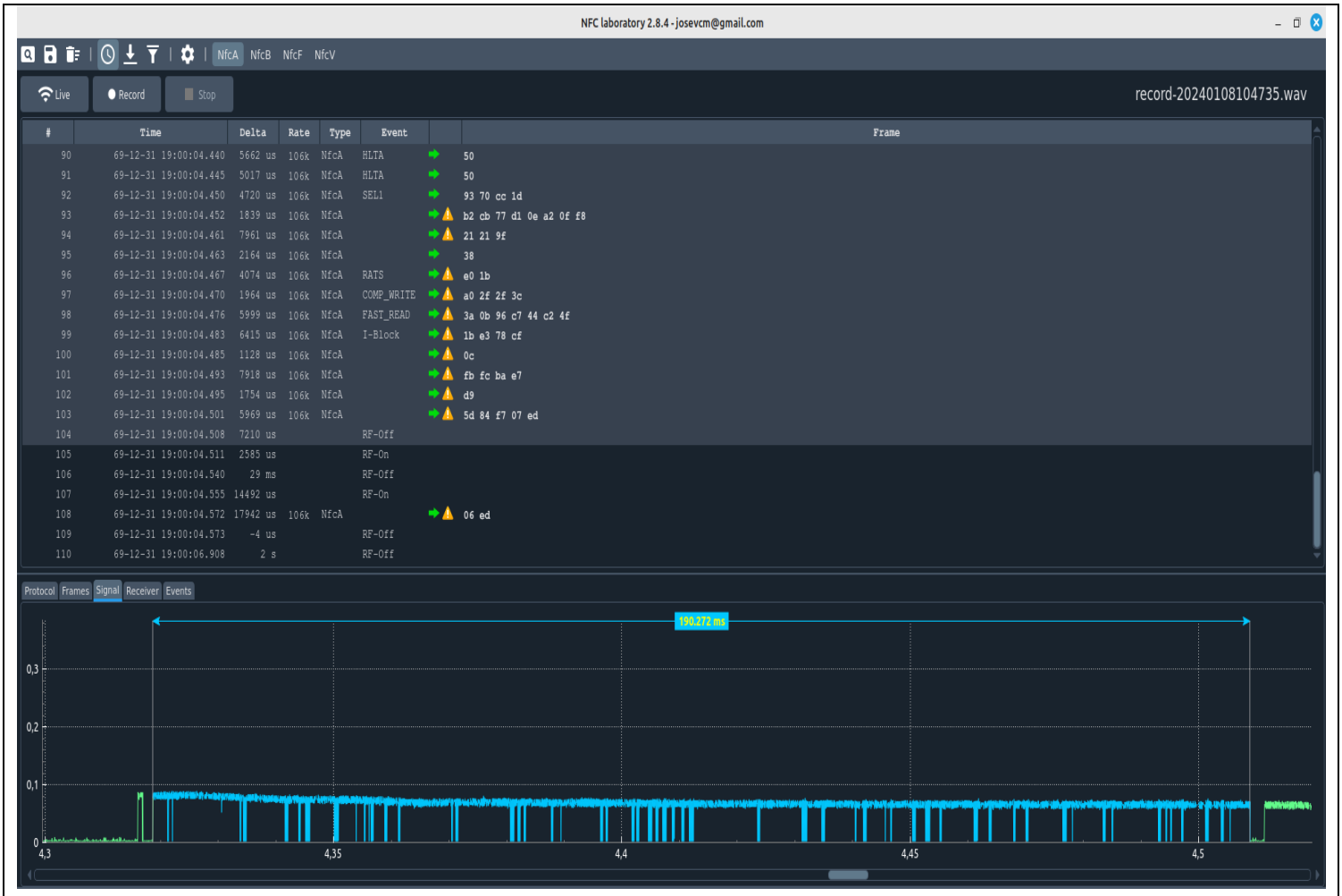
antena, parámetros de RF de SDR#. Estos factores convergen para definir la norma o factor de normalización de los datos obtenidos para este análisis de RF.

En resumen, el proceso de captura y análisis de señales con el dispositivo RTL-SDR Dongle y el software SDR# es un procedimiento necesario para el análisis RF en Telecomunicaciones. El uso del software Matlab, facilita la identificación y separación de la señal y el ruido desde sus muestras IQ.

- ✓ **Análisis y comprobación parámetros de capas superiores con Sniffer NFC**

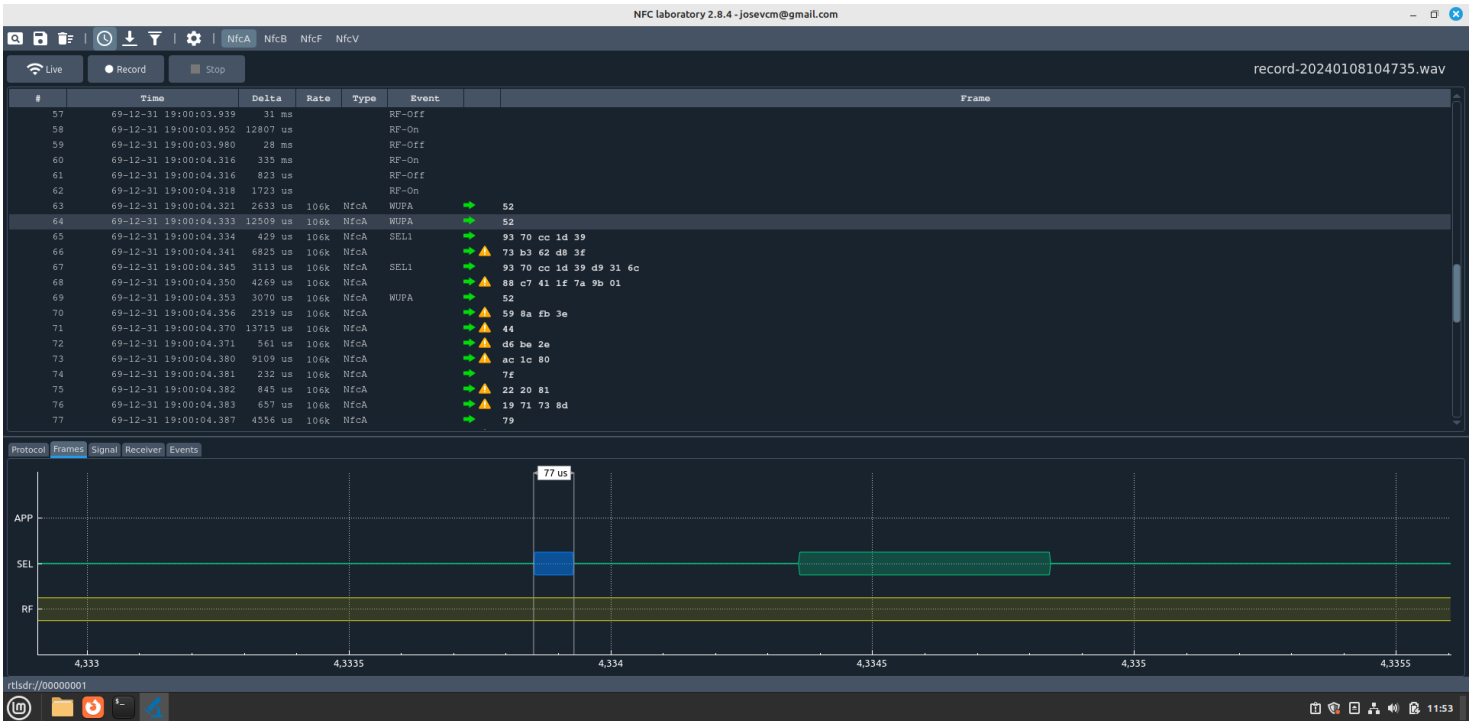
Para este análisis se realizan dos pruebas: identificar cada trama de acuerdo con los protocolos NDEF e ISO/IEC 14443-3 (APP-01) y verificar las tramas en capas superiores (APP-02).

ID	**Caso de Prueba**	**Descripción**	**Escenario**	**Objetivo**	**Resultado Esperado**	**Resultado Obtenido**	**Cumple/No Cumple**	**Referencia a la norma**
APP-01	Identificación de Tramas	Tramas de NFC	Con el uso del Sniffer NFC, identificar cada trama	Identificar las tramas de NFC	Tramas WUPA, SEL, RF, HLTA, RATS, etc.,	WUPA, SEL, RF, HLTA, RATS, COMP, FST_READ y BLOCK	CUMPLE	ISO/IEC 14443-3
RESULTADO								



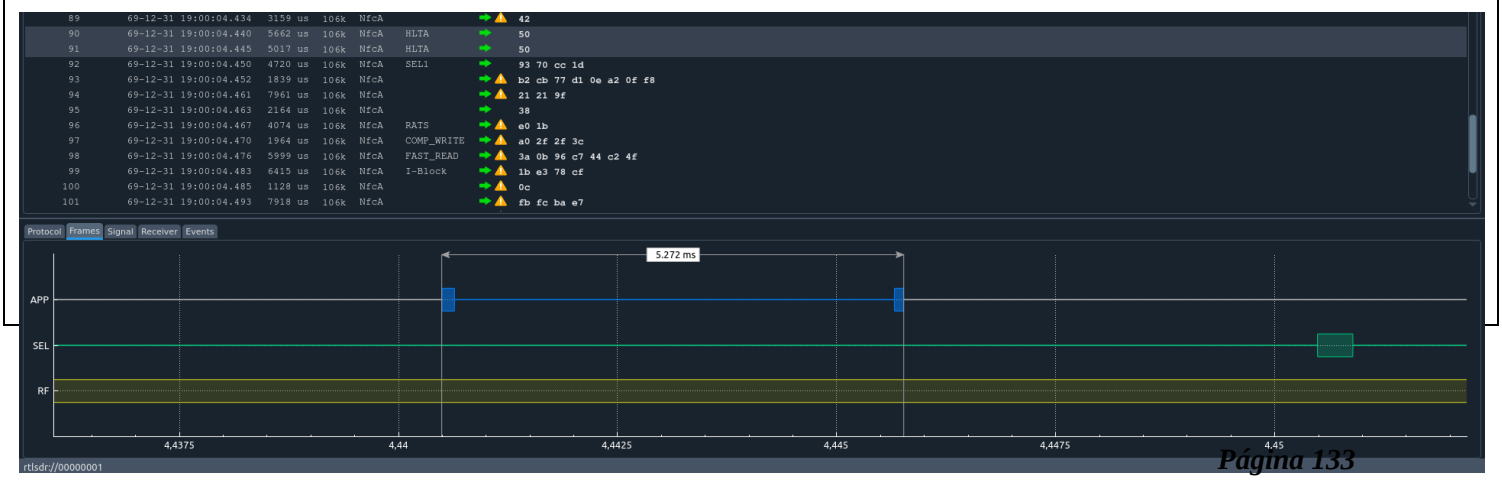
ID	Caso de Prueba	Descripción	Escenario	Objetivo	Resultado Esperado	Resultado Obtenido	Cumple/No Cumple	Referencia a la norma
APP-02	Verificación Tramas superiores	Tramas de NFC	Con el uso del Sniffer NFC, identificar cada trama	Identificar las tramas de NFC	Tramas WUPA, SEL, RF, HLTA, RATS, etc.,	WUPA, SEL, RF, HLTA, RATS, COMP, FST_READ y BLOCK	CUMPLE	ISO/IEC 14443-3
RESULTADO								

WUPA

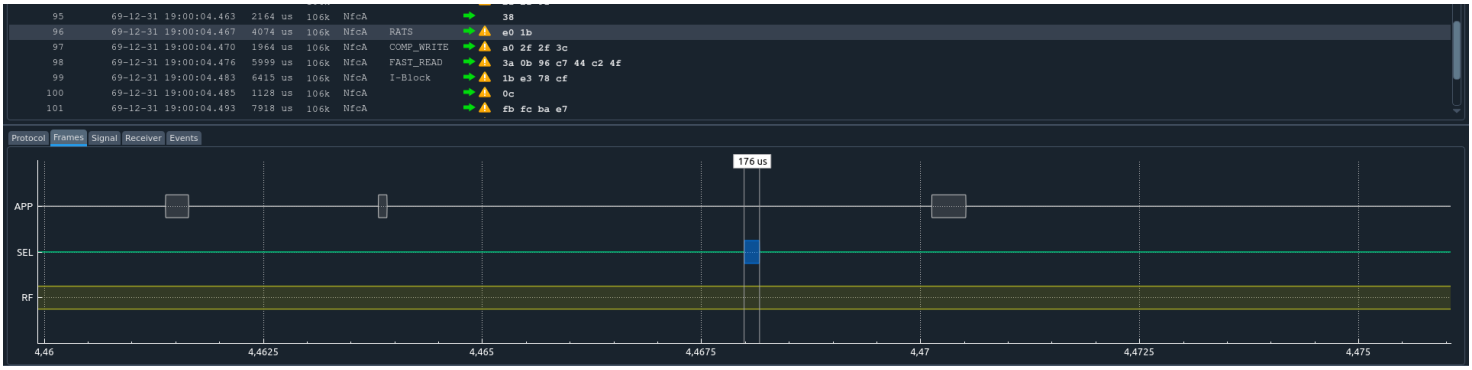


SEL

HLTA



RATS



COMP

FAST READ e I-BLOCK

- ✓ **Análisis y comprobación parámetros médicos**

En estos parámetros se analiza: frecuencia de operación permitida por OMS (MED-1), campo eléctrico teórico y campo magnético teórico permitido por la OSHA (MED-2 y MED-3) y potencia de la señal permitida por la FDA (MED-4).

ID	Caso de Prueba	Descripción	Objetivo	Resultado Esperado	Resultado Obtenido	Cumple/ No Cumple	Referencia a la norma
MED-1	Frecuencia	Frecuencia permitida por OMS	Verificar que la fc=13.56 MHz es no ionizante	Frecuencia no Ionizante	Frecuencia no Ionizante (50 Hz a 300 GHz)	CUMPLE	Electromagnetic fields (who.int)
MED-2	Campo Eléctrico	Campos Electromagnéticos permitidos por la OSHA	Verificar que el campo eléctrico de NFC comprendido entre 4 mVpeak/m – 20 mVpeak/m no es perjudicial para la salud (ISO/IEC 14443-2 8.2)	Campo Eléctrico sin efectos	Campo Eléctrico sin efectos (700kV/m2)	CUMPLE	https://obtienearchivo.bcn.cl/obtienearchivo?id=repositorio/10221/24093/2/Efectos%20sobre%20la%20salud%20humana%20de%20los%20campos%20el%C3%A9ctricos%20y%20magn%C3%A9ticos%20ELF.pdfo.int)
MED-3	Campo Mangmético	Campos Electromagnéticos permitidos por la OSHA	Verificar que el campo mágnetico de NFC comprendido entre 1,9uT - 9,4uT no es perjudicial para la salud (ISO/IEC 14443-2 6)	Campo Magnético sin efectos	Campo Magnético sin efectos (400 mT – 40 mT)	CUMPLE	https://obtienearchivo.bcn.cl/obtienearchivo?id=repositorio/10221/24093/2/Efectos%20sobre%20la%20salud%20humana%20de%20los%20campos%20el%C3%A9ctricos%20y%20magn%C3%A9ticos%20ELF.pdfo.int)

MED-3	Potencia	Potencia de radiación permitida	Verificar que la potencia de transmisión de NFC no sea perjudicial para la salud 2.12 mW/m^2 a 53.03 mW/m^2	Potencia sin daño a tejidos	Potencia sin daños a tejidos (<6.5 W/m^2)	CUMPLE	Power density (W/m^2) (who.int)

Es así como se comprueba los diferentes Testbed realizados y se deja el precedente abierto para que la Carrera de Medicina de la Universidad Técnica del Norte pueda utilizar la información para elaborar las normativas y códigos de ética para los implantes NFC subcutáneos.

En la Tabla 15, se detalla la matriz de pruebas del sistema en conjunto que se realizaron en los apartados anteriores, tanto de las pruebas del sistema para lectura y escritura de NFC con el aplicativo desarrollado, así como también los Testbed realizados a la tecnología NFC y su consecuencia en el ámbito médico.

Tabla 15: Matriz de Pruebas del Sistema

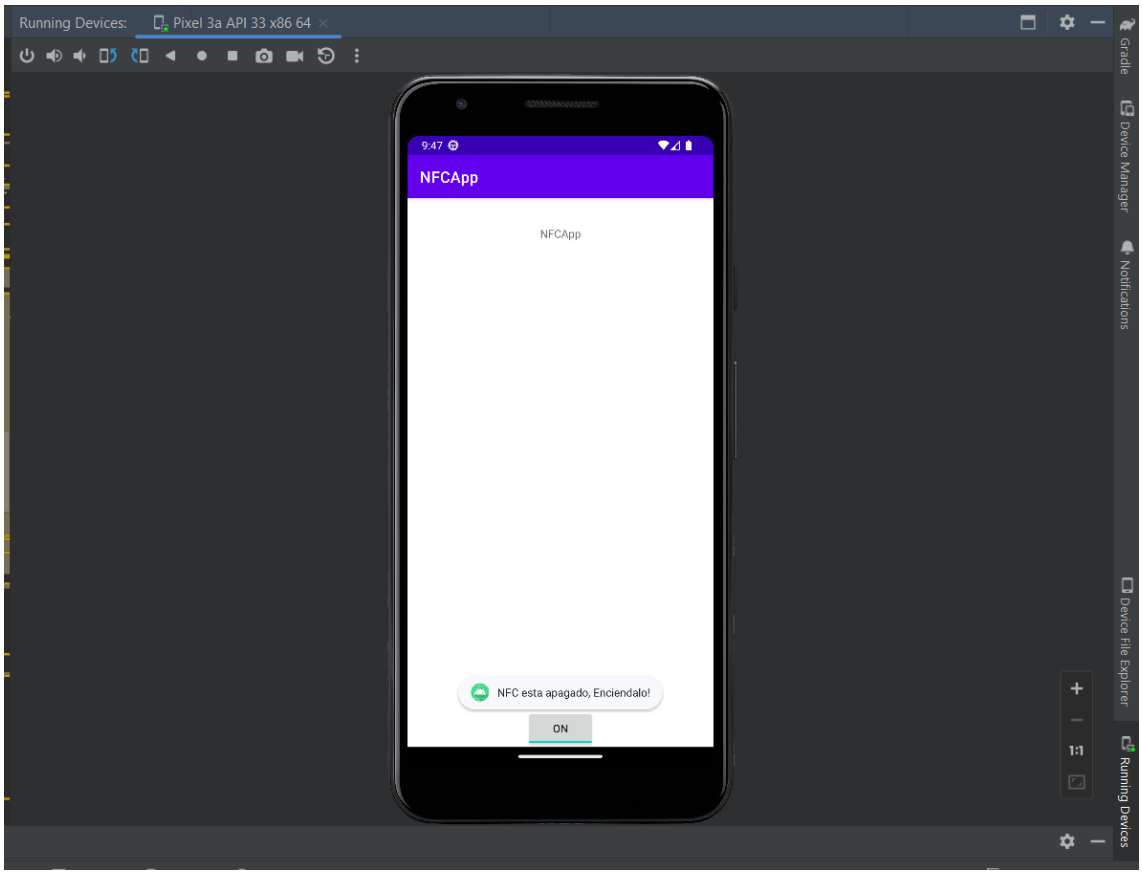
ID	Caso de prueba	Resultado Esperado	Cumple / No Cumple
01	Escritura de información en el chip NFC	La aplicación NFC Tools permite escribir toda la información del paciente en el chip NFC sin mostrar mensajes de Toast.	Cumple
02	Lectura de información desde el chip NFC	La aplicación desarrollada puede leer tanto la información almacenada en el chip NFC del paciente como la información del paciente en la base de datos, y mostrarla en la interfaz de usuario sin errores ni mensajes de Toast.	Cumple
03	Mostrar información del paciente en la aplicación	La aplicación muestra toda la información del paciente en la interfaz gráfica según lo programado, incluyendo su historial médico, datos de identificación, alergias y antecedentes médicos.	Cumple
04	Detección de etiquetas NFC	La aplicación es capaz de detectar cualquier etiqueta NFC y responder adecuadamente a su detección, sin errores ni fallos.	Cumple

05	Prueba de Rendimiento y Testbed	La aplicación cumple con los parámetros establecidos por NFC Forum, NDEF e ISO/IEC 18092 en términos, frecuencia, modulación, codificación, de velocidad de detección de etiquetas y procesamiento de datos, garantizando una experiencia de usuario receptiva incluso en situaciones de alta demanda.	Cumple

4.4.2. Fase de Emulación

Durante la etapa de emulación, se emplean emuladores de dispositivos Android con el propósito de recrear escenarios relacionados con la detección y lectura de etiquetas NFC en un entorno controlado. Este enfoque permite la evaluación precisa del comportamiento de la aplicación en situaciones simuladas, incluyendo la detección de etiquetas específicas y el procesamiento de datos NDEF. En el transcurso de esta fase, se llevan a cabo pruebas con diversas configuraciones de etiquetas NFC, lo que permite una evaluación integral de cómo la aplicación respondía ante diferentes tipos de datos y tecnologías NFC (ver Figura 27).

Figura 27: Emulación de la aplicación en Teléfono Google Pixel 3A.



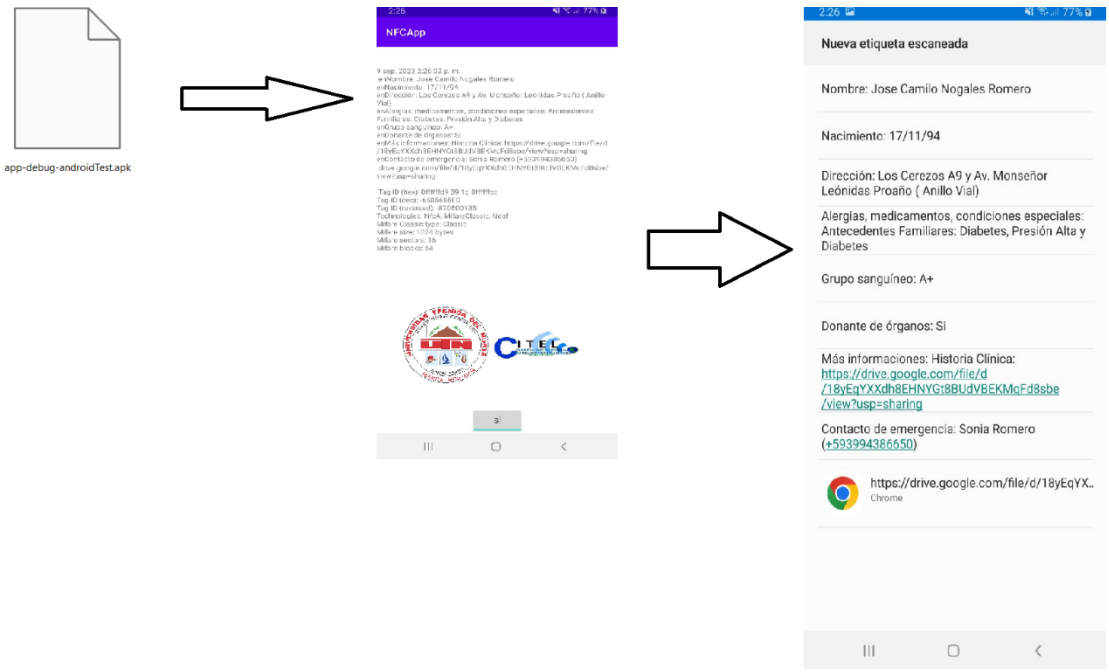
La etapa de emulación proporciona un entorno controlado que facilita el proceso de prueba y refinamiento de la funcionalidad NFC de la aplicación. Su relevancia radica en la capacidad de identificar y abordar posibles problemas de rendimiento, asegurando que la aplicación estuviera debidamente preparada para someterse a pruebas en dispositivos físicos.

4.4.3. Ejecución en Dispositivos Reales

Para poder llegar a esta etapa, una vez depurados todos los códigos implementados, se debe construir la aplicación en un formato apk, la cual se puede instalar como archivo en cualquier teléfono que cuente con el sistema operativo Android. Durante la etapa de ejecución en dispositivos reales, se llevan a cabo pruebas utilizando un dispositivo

concreto, en este caso, el Samsung A8+ 2018 con NFC y una versión de Android 9, tal y como se puede observar en la Figura 28. El propósito principal es validar el rendimiento y la funcionalidad de la aplicación en un entorno del mundo real, específicamente en este modelo de dispositivo. Las pruebas se centran en la detección y lectura de etiquetas NFC en dicho dispositivo, abarcando diversos tipos de etiquetas y tecnologías NFC específicas para este modelo. Además, se evalúa la capacidad de la aplicación para procesar datos NDEF y proporcionar retroalimentación en tiempo real a través de la interfaz de usuario.

Figura 28: Ejecución del Aplicativo en el Teléfono Samsung A8+



La ejecución en el dispositivo Samsung A8+ 2018 con Android 9 permite la verificación de la robustez y confiabilidad de la aplicación en situaciones cotidianas, garantizando una experiencia NFC fluida y sin problemas para los usuarios de este dispositivo en particular. También proporciona una retroalimentación que respaldó ajustes y mejoras precisas basadas en las pruebas realizadas en este entorno específico.

4.5.Etapa de Lanzamiento del Software

Durante la etapa de lanzamiento del software, se llevan a cabo diversas actividades para la preparación de la aplicación NFC antes de su implementación y uso por parte de los usuarios finales. Estas actividades comprenden la compilación definitiva de la aplicación, la confirmación de la completa funcionalidad y operatividad, así como la solución de posibles problemas de último momento.

4.5.1. Preparación del Ambiente de Despliegue

En lo que respecta a la preparación del entorno de despliegue, es esencial adecuar la aplicación para que sea compatible con una amplia gama de versiones de sistemas operativos, así como con diversas versiones de Kernel y posibles actualizaciones futuras. Durante esta fase, se llevan a cabo tareas como, la optimización de recursos, la gestión de permisos y la solución de problemas de compatibilidad con varias versiones de Android, utilizando el compilador de Android Studio para solucionar los problemas de compatibilidad que se puedan presentar en varios dispositivos, haciendo a la aplicación más universal.

4.5.2. Instalación y Configuración del Sistema NFC en Smartphone

Una vez que el aplicativo se compila y se resuelve todas las cuestiones de compatibilidad mediante la preparación del entorno de despliegue, se avanza hacia la etapa de escritura del chip implante utilizando la aplicación NFC Tools. Posteriormente, se procede con la instalación del aplicativo y se realiza la configuración necesaria para asegurar un despliegue rápido y automatizado.

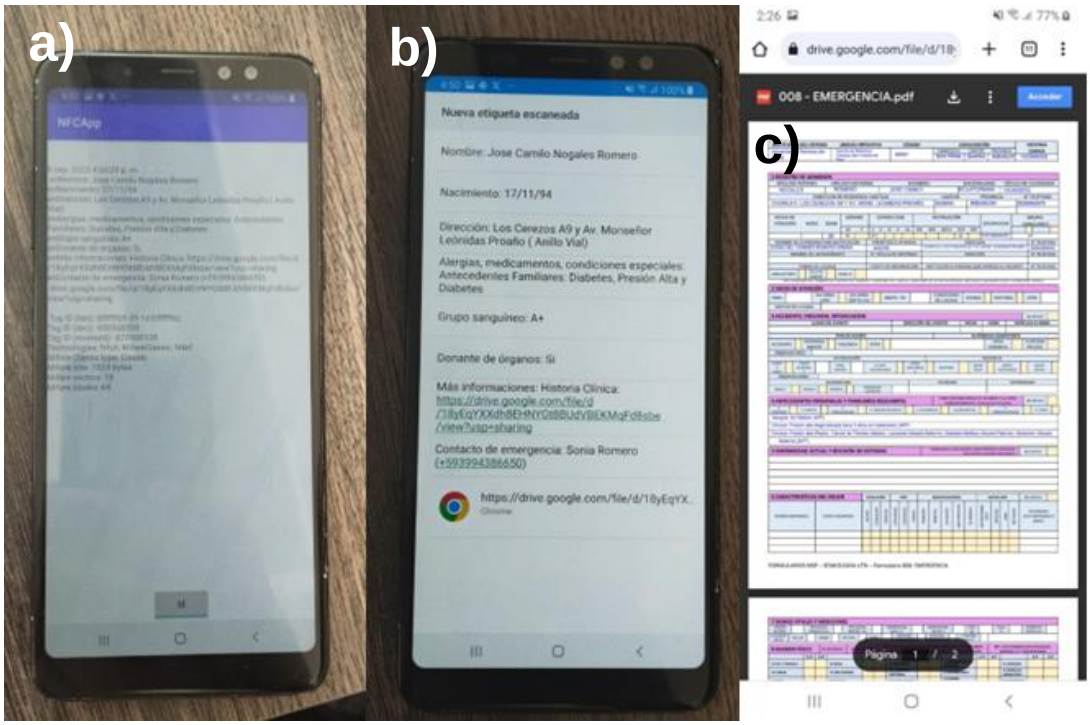
En este contexto, un smartphone se convierte en una herramienta fundamental para interactuar con etiquetas NFC, lo que requiere una configuración adecuada del hardware y del sistema operativo. Es así como en este apartado, se exploran los pasos y

consideraciones necesarios para habilitar y configurar la funcionalidad NFC en un smartphone, sentando las bases para una interacción efectiva con etiquetas NFC y aplicaciones relacionadas. Desde la habilitación del hardware NFC hasta la configuración de permisos y aplicaciones, se abordan las diversas facetas que garantizan que un smartphone esté listo para aprovechar al máximo la tecnología NFC. (ver Anexo 11)

4.5.3. Funcionamiento Final

La aplicación móvil desarrollada ofrece una experiencia de usuario centrada en la interacción con etiquetas NFC. Para comenzar, el usuario debe asegurarse de que su dispositivo Android tenga habilitada la funcionalidad NFC siguiendo una configuración inicial única.

Figura 29: Funcionamiento final del aplicativo en todas sus etapas. a) Inicio de la aplicación, b) Funcionamiento del aplicativo y c) Historia Clínica en PDF dentro del almacenamiento del chip.



Una vez configurado, el usuario inicia la aplicación en el smartphone, que proporciona una interfaz con un interruptor para activar o desactivar el modo NFC y una sección de visualización de mensajes y datos de etiquetas NFC como se observa en la Figura 29-a. Cuando el modo NFC está habilitado, el usuario puede acercar su dispositivo a una etiqueta NFC física para iniciar la lectura de datos. La aplicación detecta automáticamente la etiqueta y muestra la información relevante en su interfaz. Además, proporciona retroalimentación en tiempo real a través de mensajes Toast, informando al usuario sobre el estado de la operación NFC. (ver Figura 29-b), y finalmente en la Figura 29-c, se muestra la historia clínica almacenada dentro del chip con la información relevante del paciente.

Es así como, el aplicativo está listo para realizar pruebas en la Clínica de Simulación de la Universidad Técnica del Norte, en ambientes controlados y utilizando las diferentes etapas de triage de emergencia para evaluar los resultados en cada uno de los colores que la metodología Manchester lo propone.

CAPITULO V

PRUEBAS DE FUNCIONAMIENTO EN LA CLINICA DE SIMULACION UTN

En este apartado, se lleva a cabo la evaluación del aplicativo en el entorno controlado de la clínica de simulación de la UTN. El proceso se inicia con un análisis de las 6M's del aplicativo, lo que implica considerar aspectos como movilidad, momento, “yo”, multi-usuario, ganancias, máquinas o dispositivos, el entorno y la medición para garantizar su lanzamiento oficial.

Posteriormente, se diseñan los casos de prueba que abordarán cada uno de los colores que conforman el sistema de triaje Manchester, identificando qué tipos de emergencias se simularán en cada caso. Esto permite llevar a cabo pruebas comparativas entre el uso del aplicativo y el sistema de salud actual en términos de tiempo de respuesta y eficacia.

Los resultados obtenidos de estas pruebas se detallan, proporcionando datos cuantitativos que respalden las conclusiones finales de este proyecto de titulación, contribuyendo así a la toma de decisiones informadas sobre la viabilidad y utilidad del aplicativo en un entorno de atención médica real.

5.1. Análisis de las 6M’s

Como se analizó en el capítulo 3, las 6M’s representan una serie de elementos necesarios para ser considerados dentro de una aplicación para su entrega o en una tienda de aplicaciones o App Store (sea GooglePlay, AppStore, MIStore, etc.). Cada uno de estos atributos define la experiencia del usuario al usar el aplicativo y su

beneficio para el desarrollador en términos de ganancias económicas y alcance de reconocimiento para convertirse en un App Developer.

Para llevar a cabo el análisis, se procede a describir cada uno de los atributos presentes en la aplicación, los cuales forman parte de las 6M's (ver Tabla 16). A cada uno de estos atributos se le asigna una calificación en una escala de 1 a 5, donde el valor 1 indica la ausencia total del atributo en la aplicación, mientras que el valor 5 denota la presencia completa y destacada del atributo en el aplicativo. (Castro S., et al, 2016).

Tabla 16: Análisis de las 6M's para la aplicación desarrollada.

Atributo	Presencia en el Aplicativo	Calificación
Movimiento	El aplicativo es capaz de adaptarse y responder a los movimientos del usuario final.	5
Momento	La aplicación ofrece información relevante y oportuna en situaciones de emergencia médica.	5
Yo	Aunque no ofrece una experiencia altamente personalizada, se adapta a las preferencias de los médicos.	2
Multiusuario	Compatible con múltiples sistemas operativos Android, pero no con iOS.	4
Dinero	No se busca generar ingresos a través de la aplicación debido a su naturaleza de código abierto.	1
Máquinas	Permite la interacción con otros dispositivos o sistemas, como chips con sensores.	5

Consecuentemente, a continuación, se detalla cada atributo:

- **Movimiento (Calificación: 5):** La aplicación exhibe una excelente capacidad para adaptarse y responder a los movimientos del usuario final. Esto se evidencia en situaciones críticas de atención médica de emergencia, donde la aplicación permite una interacción fluida y receptiva. Los usuarios pueden acceder rápidamente a la información y realizar acciones necesarias, incluso en situaciones de alta presión, garantizando una experiencia de usuario óptima.
- **Momento (Calificación: 5):** La aplicación proporciona un valor relevante y oportuno en el momento adecuado, lo que es esencial en el contexto de emergencias médicas. Suministra información crucial para ayudar a los profesionales de la salud a tomar decisiones informadas y brindar atención de calidad a los pacientes.
- **Yo (Calificación: 2):** Aunque la aplicación no ofrece una experiencia altamente personalizada para cada usuario, se adapta a las preferencias y características individuales de los médicos. Esto permite cierto grado de personalización dentro de un contexto médico más amplio. Sin embargo, no proporciona una personalización completa para cada usuario, lo que resulta en una calificación moderada en este atributo.
- **Multiusuario (Calificación: 4):** El aplicativo muestra una buena compatibilidad con múltiples sistemas operativos Android, lo que lo hace accesible para un amplio grupo de usuarios en dispositivos Android. Sin embargo, no es compatible con iOS, lo que limita su alcance a los usuarios de dispositivos Apple. Aunque no es completamente multiplataforma, aún recibe una calificación sólida debido a su amplia compatibilidad en el ecosistema Android.

- **Dinero (Calificación: 1):** La generación de ingresos no es una prioridad ni un objetivo de este aplicativo. Su naturaleza de código abierto y la intención del desarrollador de hacerlo global y gratuito para mejorar los sistemas de atención médica de emergencia lo excluyen de generar ingresos directamente a través de la aplicación.
- **Máquinas (Calificación: 5):** El aplicativo permite la interacción con otros dispositivos o sistemas, como chips con sensores. Además, se reconoce que existen chips más avanzados que poseen sensores y comunicación IoMT (Internet of Medical Things), lo que amplía aún más las funcionalidades y la integración tecnológica de la aplicación. Esto le otorga la calificación más alta en este atributo, ya que demuestra una gran capacidad de interconexión con otras tecnologías médicas.

5.2. Diseño de Casos de Prueba dentro del Triaje de Emergencias

Los casos de prueba representan conjuntos de escenarios específicos y detallados diseñados para evaluar el rendimiento, la funcionalidad y la confiabilidad de un software o una aplicación en diversas situaciones. Estos escenarios se crean con el propósito de poner a prueba diversos aspectos del programa, abordando elementos como la interacción del usuario, la funcionalidad de las características, la respuesta ante situaciones excepcionales y la precisión de los resultados. Su función principal es asegurar que el software cumpla con los requisitos y expectativas establecidos, identificando cualquier defecto o anomalía que requiera corrección antes de su entrega final.

En este contexto, su función se centra en medir la eficacia y eficiencia de la aplicación desarrollada en situaciones de emergencia médica. Estos casos se diseñan específicamente para simular diversos escenarios médicos que involucran diferentes códigos de colores según el sistema de triaje Manchester (MSP, 2015), un estándar ampliamente utilizado para categorizar la gravedad de los pacientes en una unidad de emergencia. La aplicación se someterá a pruebas rigurosas para evaluar su capacidad de respuesta y utilidad en situaciones críticas de atención médica, detalladas en la Tabla 17.

Cada caso de prueba representará una situación de emergencia médica distinta, abarcando desde pacientes con afecciones menores hasta casos extremadamente graves. El objetivo principal es medir el tiempo transcurrido desde que el paciente llega a la unidad médica hasta que recibe atención y se le asigna un código de color de triaje. Esto proporcionará una medida precisa de la rapidez con la que la aplicación puede ayudar al personal médico a tomar decisiones críticas.

Estas pruebas se llevan a cabo en colaboración con internos rotativos de Medicina de la Universidad Técnica del Norte, quienes asumirán el rol de personal médico de emergencia. Esto garantizará que las pruebas se asemejen lo más posible a situaciones reales en un entorno clínico simulado. Los resultados de estas pruebas no solo serán esenciales para evaluar la aplicación, sino que también podrían tener un impacto significativo en la mejora de la atención médica de emergencia en el futuro. Esta etapa representa un paso importante en la validación y el perfeccionamiento de la aplicación desarrollada en un contexto médico realista y desafiante.

5.2.1. Simulación de Emergencias en cada Triaje y Contraste de Tiempos

Como se detalla en el apartado anterior, las pruebas, que se llevan a cabo tanto con el uso del chip implante como sin él, se centran en escenarios médicos que representan las diferentes categorías del triaje Manchester. El objetivo principal es evaluar la eficiencia y el tiempo de respuesta de la aplicación en situaciones de emergencia médica, proporcionando datos comparativos cruciales para determinar la utilidad y eficacia de la aplicación en entornos reales de atención médica. (ver Tabla 18)

En la Tabla 17, se detalla la metodología empleada para la ejecución de estas pruebas de simulación. Esto incluye la descripción de los escenarios médicos planteados para cada color del triaje Manchester, los procedimientos realizados tanto con el uso del chip implante como sin él.

Tabla 17: Casos de prueba para cada color del triaje Manchester

Color del Triaje Manchester	Prueba	Descripción	Detalle con el Chip Implante	Detalle sin el Chip Implante
Azul	Prueba 1	Malestar en la garganta (odinofagia).	Escaneo del chip para acceso a historial médico y alergias y condiciones especiales	Registro manual de antecedentes médicos, alergias y síntomas.
Verde	Prueba 2	Dolor de estómago, vómito y diarrea (gastroenteritis).	Escaneo del chip para acceso a historial médico y alergias y condiciones especiales	Registro manual de antecedentes médicos, alergias y síntomas.
Amarillo	Prueba 3	Heces con sangre, mareos y nauseas (Posible Sangrado Digestivo Bajo).	Escaneo del chip para acceso a historial médico y alergias y condiciones especiales	Registro manual de antecedentes médicos, alergias y síntomas.
Naranja	Prueba 4	Herida de gravedad debido a un corte	Escaneo del chip para acceso a historial	Registro manual de antecedentes

		(Laceración).	médico y alergias y condiciones especiales	médicos, alergias y síntomas.
Rojo	Prueba 5	Desmayo, paciente inconsciente, sangrado grave (Hemorragia masiva).	Escaneo del chip para acceso a historial médico y alergias y condiciones especiales	Registro manual de antecedentes médicos, alergias y síntomas.

Para desarrollar éstas pruebas se realizó un análisis previo entre los Internos Rotativos de Medicina (IRM) y médicos de la Carrera de Medicina de la UTN, siendo los escenarios más comunes y tratados en las emergencias médicas.

El procedimiento que se lleva a cabo en la simulación de cada escenario comienza con la llegada del paciente. La metodología actual establece que, una vez que el paciente arriba a la instalación médica se procede a su primera valoración utilizando el formulario MSP-008 "Emergencia". En este proceso, los datos del paciente, que incluyen antecedentes médicos, alergias, síntomas, posibles patologías, entre otros, se registran manualmente en el formulario. El principal propósito del formulario es mantener un registro sistemático de los datos recopilados durante la atención de los usuarios en el servicio de emergencia, como se muestra en el Anexo 12 donde se detalla cada una de las características que se deben llenar en cada campo del formulario.

La finalidad propuesta en este proyecto de titulación es omitir el llenado a mano de información en cada uno de los campos del formulario MSP-008 “Emergencia” y que dicha información se encuentre dentro del chip con los datos previos a la atención médica como son: Registro de Admisión, Inicio de Atención y Motivo, Enfermedades Actuales, Antecedentes Personales y Familiares (ver Anexo 13) y que dicha

información se presente en la interfaz gráfica del aplicativo para acelerar el tratamiento del paciente dentro de la unidad médica.

Una vez diseñados los casos de prueba para cada situación de triaje, se registran los tiempos en los escenarios propuestos. El cronometraje se inicia desde la llegada del paciente a la clínica de simulación hasta su atención y categorización según el nivel de emergencia.

La Tabla 18 presenta un desglose de los tiempos obtenidos en cada escenario de emergencia o urgencia. Asimismo, se incluye el porcentaje de reducción de tiempo logrado mediante el uso del sistema NFC en cada prueba realizada.

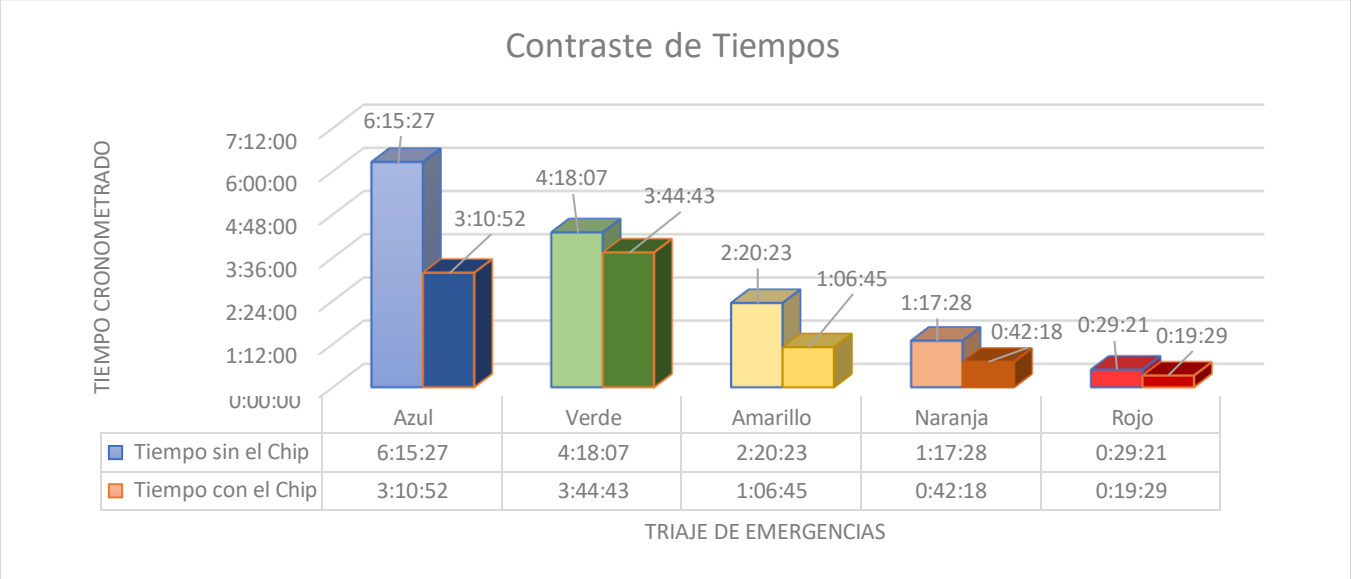
Tabla 18: Contraste de tiempos obtenidos dentro de los casos de prueba

Color de Triaje	Prueba	Tiempo sin el Chip	Tiempo con el Chip	Porcentaje de Reducción de Tiempo
Azul	Malestar en la garganta (odinofagia)	06:15.27	03:10.52	48.99%
Verde	Dolor de estómago, vómito y diarrea (gastroenteritis)	04:18.07	03:44.43	12.96%
Amarillo	Heces con sangre, mareos y nauseas (Posible Sangrado Digestivo Bajo)	02:20.23	01:06.45	52.25%
Naranja	Herida de gravedad debido a un corte (Laceración)	01:17.28	00:42.18	45.33%
Rojo	Desmayo, paciente inconsciente, sangrado grave (Hemorragia masiva)	00:29.21	00:19.29	33.99%

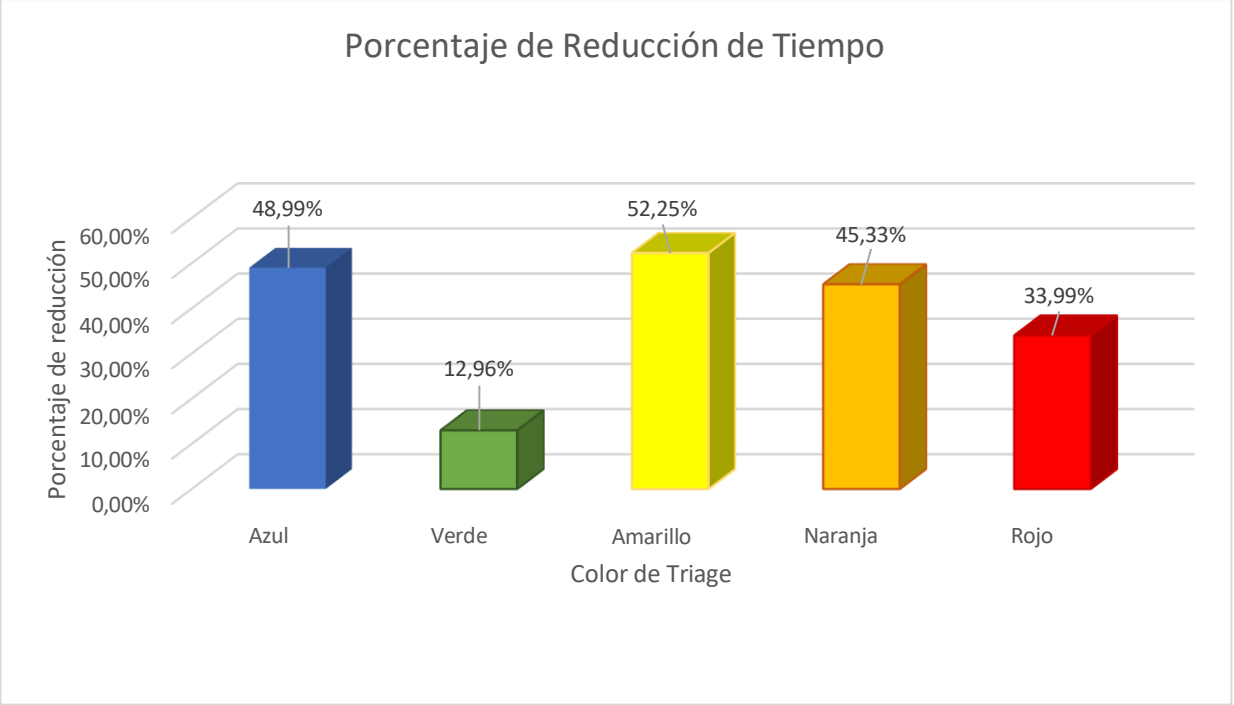
5.3. Análisis de los Resultados

Los resultados de los casos de pruebas en la clínica de simulación de la UTN revelaron hallazgos significativos en términos de la reducción del tiempo necesario para la atención médica de emergencia. La eficacia de la aplicación desarrollada, "NFC App", se hizo evidente al comparar los tiempos requeridos para la atención y categorización de pacientes con y sin el uso del chip implante NFC, como se observa en la Figura 28.

Figura 28: Diagrama de Barras del contraste de los tiempos en cada situación de Triaje



En la Figura 29, se observa los porcentajes de reducción de tiempo en cada nivel de triaje de emergencia según su color. Estos porcentajes reflejan la disminución del tiempo en cada uno de los colores mencionados anteriormente al utilizar el sistema NFC.



En el escenario de emergencia médica de categoría Azul, se observaron resultados notables. La utilización del chip implante y el aplicativo desarrollado condujo a una reducción del tiempo de aproximadamente el 48.99%. Este logro indica que la aplicación permitió que el proceso de atención se llevara a cabo de manera más rápida y efectiva en comparación con los métodos convencionales del sistema de triaje Manchester.

En el caso de emergencia de categoría Verde, donde los pacientes presentan síntomas menos graves relacionados con problemas gastrointestinales como vómitos y diarrea, la aplicación "NFC App" también demostró ser beneficiosa. Con el chip NFC implantado, se logró una reducción del tiempo del 12.96%. Aunque esta reducción es menor, sigue siendo significativa, ya que acelera el proceso de evaluación y atención de los pacientes.

El impacto positivo del uso de la aplicación "NFC App" se destacó aún más en el escenario de emergencia Amarillo. En este caso, se logró una impresionante reducción del tiempo del 52.25% mediante el chip implante. Esta cifra ilustra claramente cómo la aplicación puede agilizar la atención a pacientes con condiciones médicas que requieren intervención urgente.

En el caso de emergencia de categoría Naranja, que involucra heridas graves como laceraciones, el uso de la aplicación "NFC App" con el chip NFC implantado mostró una reducción del tiempo del 45.33%. Esta disminución de tiempo es interesante y resalta cómo la aplicación puede marcar la diferencia en situaciones médicas de urgencia. En tales casos, una atención médica rápida y precisa es fundamental para minimizar el riesgo de infecciones y complicaciones. El impacto del chip NFC se refleja claramente en la capacidad de acceder a la información del paciente de manera inmediata, lo que agiliza el proceso de atención y tratamiento.

El análisis de los resultados en el escenario de emergencia de categoría Roja pone de manifiesto una situación crítica en la que cada segundo es crucial para la supervivencia del paciente. En este contexto, el uso de la aplicación "NFC App" demostró una reducción notable en el tiempo necesario para la atención médica. Con el chip implante NFC, se logró una disminución del tiempo del 33.39%. Aunque esta cifra es ligeramente menor en comparación con otros colores de triaje, sigue siendo altamente significativa. En situaciones de emergencia de categoría roja, donde la vida del paciente está en riesgo inminente debido a hemorragias masivas u otras condiciones críticas, una reducción en el tiempo de atención puede marcar la diferencia entre la vida y la muerte.

El hecho de que la aplicación "NFC App" pueda acelerar el proceso de atención en este contexto es un indicio claro de su valioso potencial en entornos médicos de alta presión.

El promedio del porcentaje de reducción de tiempo en todas las pruebas fue del 38.704%. Estos resultados reflejan la eficacia y eficiencia de la aplicación "NFC App" en el contexto médico de emergencia. La capacidad de acceder rápidamente a la información vital del paciente, como alergias y antecedentes médicos, gracias al chip implante NFC, permitió a los profesionales médicos tomar decisiones más informadas y acelerar los procedimientos de atención. Esto, a su vez, puede tener un impacto significativo en la supervivencia y recuperación de los pacientes en situaciones críticas. En resumen, los datos de estas pruebas respaldan la importancia y el potencial de la aplicación "NFC App" como una herramienta importante para mejorar la atención médica de emergencia y salvar vidas (Anexo Video).

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

- Se estudió la literatura disponible acerca de las tecnologías NFC y su convergencia con la medicina para analizar los tiempos de demora en el proceso de triaje en instalaciones de salud pública. En base al contexto estudiado y a la discusión generada, las tecnologías NFC en conjunto con los chips implantes tienen el potencial de abrir nuevas oportunidades en la mejora de la atención médica de emergencia. La investigación en este ámbito es fundamental para comprender cómo la tecnología NFC, en particular el chip implante NFC X3, puede acelerar los procesos de triaje y, en última instancia, salvar vidas. La convergencia de tecnología y medicina representa un campo prometedor que podría revolucionar la atención médica de emergencia y mejorar significativamente la eficiencia y la precisión de los procedimientos de triaje. La literatura existente proporciona una base sólida, y futuras investigaciones en esta área pueden contribuir de manera significativa al avance de la atención médica en situaciones críticas.
- La necesidad de evitar la redundancia en los datos es esencial, ya que permite que la información se gestione de manera eficaz y precisa, lo que, a su vez, mejora la capacidad del sistema para brindar una atención médica de emergencia oportuna y precisa. Definir los parámetros y características del sistema de manera adecuada permitió una estructuración sólida y coherente de los procesos de envío y recolección de datos. Esto garantizó que la información esté disponible cuando se necesita, sin duplicados innecesarios, lo que puede ser

esencial en situaciones críticas de atención médica. Además, esta optimización en la gestión de datos puede resultar en una mayor eficiencia en el proceso de triaje y una atención médica más efectiva.

- La integración de metodologías en el desarrollo de este proyecto de titulación ha sido de gran utilidad, permitiendo abordar de manera más integral las necesidades de los stakeholders. Al trabajar en conjunto, estas metodologías han facilitado la convergencia y el cumplimiento de todas las características propuestas a lo largo del proyecto. No solo se ha logrado la programación del software, sino también se ha abordado el alcance del hardware, en este caso, los dispositivos móviles, y su compatibilidad con sistemas operativos diferentes. Cada una de las etapas descritas en esta unión de metodologías ha seguido un proceso característico que se refleja claramente en los resultados finales del proyecto. La realización de pruebas de escritorio en cada fase del proyecto se ha revelado como un elemento crucial para asegurar el funcionamiento del sistema de acuerdo con las especificaciones. Este enfoque temprano en la identificación de problemas y errores ha permitido un ahorro significativo de tiempo y recursos en comparación con la corrección de problemas en etapas más avanzadas del desarrollo.
- Durante las pruebas técnicas del sistema, especialmente en relación con la tecnología NFC, se verificó que la comunicación del chip implante cumple con los parámetros establecidos por la OMS para frecuencias no ionizantes, así como con los valores de potencia recomendados por la OSHA para prevenir posibles quemaduras internas en tejidos, y los campos magnéticos y eléctricos propuestos

por la FDA. Estos resultados son de importancia crítica para la formulación de un código de ética en la Carrera de Medicina de la UTN, específicamente en lo que respecta a la propuesta de utilización de chips implantados en seres humanos.

- La reducción promedio del 38.704% en los tiempos de respuesta, facilitada por la incorporación del chip NFC en el proceso de triaje de emergencias, constituye el núcleo esencial de este proyecto de titulación. Más allá de ser un simple indicador numérico, este resultado representa una mejora sustancial en la eficiencia y rapidez con que se enfrentan situaciones críticas de salud. Esta disminución de los tiempos no solo implica eficacia operativa, sino que tiene un impacto directo en la capacidad de preservar vidas. En contextos de emergencias médicas, donde cada segundo es crucial, este proyecto demuestra con evidencia cuantificable que la implementación del chip NFC en el triaje puede marcar la diferencia determinante entre la vida y la muerte.

Recomendaciones:

- Evaluar meticulosamente la escalabilidad del sistema, asegurando su viabilidad en diversas instituciones de salud con estructuras y procedimientos médicos heterogéneos.
- Adaptar el sistema para abordar las singularidades y necesidades específicas de cada entorno médico, garantizando su eficaz implementación.

- Explorar a fondo opciones para lograr una integración más efectiva con los sistemas existentes de registros médicos, priorizando la coherencia y fluidez en la gestión de datos clínicos.
- Implementar actualizaciones regulares basadas en la retroalimentación de los usuarios y las necesidades emergentes en el ámbito de la atención médica.
- Establecer un enfoque proactivo en la identificación y mitigación de posibles riesgos de seguridad, asegurando la confidencialidad y la integridad de la información.
- Evaluar activamente su integración en futuras versiones del sistema para mejorar la eficiencia y calidad de la atención médica de emergencia, anticipándose a las innovaciones tecnológicas relevantes.

REFERENCIAS BILBIOGRÁFICAS

Adam Marcus, G. D. D. L. N. V. R. F. A. K. L. S. (2009). *Using NFC-enabled mobile phones for public health in developing countries*. https://doi.org/10.1109/NFC.2009.25

Ahonen, T. T., & Barrett, J. (2002). *Services for UMTS : creating killer applications in 3G*. 372. https://www.wiley.com/en-ie/Services+for+UMTS%3A+Creating+Killer+Applications+in+3G-p-9780470014172

Anaya-Cantellán, A., & López-Martínez, I. (2014). La tecnología NFC en teléfonos celulares, sus retos y aplicaciones. In *Research in Computing Science* (Vol. 77).

Aoki, R., Miyata, A., Seko, S., Hashimoto, R., Ishida, T., Watanabe, M., & Ihara, M. (2016). An information display system with information scrapping user interface based on digital signage terminals and mobile devices for disaster situations. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9732*, 353–363. https://doi.org/10.1007/978-3-319-39516-6_34/FIGURES/9

Banafa, A. (2021). *Tecnología subcutánea: retos para los microchips implantables | OpenMind*. https://www.bbvaopenmind.com/tecnologia/innovacion/tecnologia-subcutanea-3-retos-microchips-implantables/

Campos, J. (2023). *GitHub - josevcm/nfc-laboratory: NFC signal and protocol analyzer using SDR receiver*. GitHub. https://github.com/josevcm/nfc-laboratory

Cao, Z., Chen, P., Ma, Z., Li, S., Gao, X., Wu, R. X., Pan, L., & Shi, Y. (2019). Near-Field Communication Sensors. *Sensors 2019, Vol. 19, Page 3947*, *19*(18), 3947. https://doi.org/10.3390/S19183947

Carlberg, M., Hedendahl, L., Ahonen, M., Koppel, T., & Hardell, L. (2016). Increasing incidence of thyroid cancer in the Nordic countries with main focus on Swedish data. *BMC Cancer*, *16*(1), 1–15. https://doi.org/10.1186/S12885-016-2429-4/FIGURES/11

Castro, S. A., Medina, B., & Camargo, L. L. (2016). Supervisión y Control Industrial a través de Teléfonos Inteligentes usando un computador de placa única Raspberry Pi. *Informacion Tecnologica*, *27*(2), 121–130. https://doi.org/10.4067/S0718-07642016000200015

Cruz Ángel, B. S. (2021). *Seguimiento y evaluación de personas en ambientes cerrados / abiertos*. https://repositorio.escuelaing.edu.co/handle/001/1520

ECMA. (2013). *Near Field Communication-Interface and Protocol (NFCIP-1) COPYRIGHT PROTECTED DOCUMENT*.

Gasca Mantilla, M. C., Camargo Ariza, L. L., & Medina Delgado, B. (2023). *Metodología para el desarrollo de aplicaciones móviles*.

Gómez, P., & Rivera, J. (2019). Un problema social: tiempos de espera en la consulta externa del Hospital Carlos Andrade Marín. *Estudios de La Gestión. Revista Internacional de Administración*, 121–146. https://doi.org/10.32719/25506641.2019.5.5

Hameed, S., Jamali, U. M., & Samad, A. (2015). Integrity protection of NDEF message with flexible and enhanced NFC signature records. *Proceedings - 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2015*, *1*, 368–375. https://doi.org/10.1109/TRUSTCOM.2015.396

Hernandez Goya, C., Rivero García, A., & Pino Caballero, G. (2014). *FastTriaje: un asistente para la clasificación de víctimas en situaciones de emergencia con autenticación robusta*.

Horikoshi, H., & Chujo, H. (2018). Prevention Method of Electromagnetic Interference by Implementing NFC Radio Active Signal for Touchpad. *2018 IEEE 7th Global Conference on Consumer Electronics, GCCE 2018*, 691–693. https://doi.org/10.1109/GCCE.2018.8574634

Kim, H. J., Hirayama, H., Kim, S., Han, K. J., Zhang, R., & Choi, J. W. (2017). Review of Near-Field Wireless Power and Communication for Biomedical Applications. *IEEE Access*, *5*, 21264–21285. https://doi.org/10.1109/ACCESS.2017.2757267

Kothari, N., Kannan, B., Glasgwow, E. D., & Dias, M. B. (2012). Robust Indoor Localization on a Commercial Smart Phone. *Procedia Computer Science*, *10*, 1114–1120. https://doi.org/10.1016/J.PROCS.2012.06.158

Li, Y. C. (Jack), Yen, J. C., Chiu, W. T., Jian, W. S., Syed-Abdul, S., & Hsu, M. H. (2015). Building a national electronic medical record exchange system -

experiences in Taiwan. *Computer Methods and Programs in Biomedicine*, *121*(1), 14–20. https://doi.org/10.1016/J.CMPB.2015.04.013

López-González, R., Sánchez-García, J., Fos-Guarinos, B., García-Castro, F., Alberich-Bayarri, Á., Soria-Olivas, E., Muñoz-Núñez, C., & Martí-Bonmatí, L. (2021). Automated Chest Radiographs Triage Reading by a Deep Learning Referee Network. *MedRxiv*, 2021.06.01.21257399. https://doi.org/10.1101/2021.06.01.21257399

Ministerio de Salud Pública. (2020). *Registro de Emergencias 2020*.

Ministerio de Salud Pública del Ecuador. (2015). *HOSPITAL GENERAL NAPOLEON DAVILA CORDOVA CODIGO HGNDC-GC-PEM MANUAL DE NORMAS Y PROCEDIMIENTOS*.

Schiffmann, A., Clauss, M., & Honigmann, P. (2020). Biohackers and Self-Made Problems: Infection of an Implanted RFID/NFC Chip: A Case Report. *JBJS Case Connector*, *10*(2), e0399. https://doi.org/10.2106/JBJS.CC.19.00399

Sevilla, V. A., & Abril, M. A. C. (2016). Implantación del chip subcutáneo: protección de datos, intimidad y libertad religiosa. *Bioderecho.Es*, *3*(3), 16 pág.-16 pág. https://revistas.um.es/bioderecho/article/view/241081

Shafeie, S., Chaudhry, B. M., & Mohamed, M. (2022). Modeling Subcutaneous Microchip Implant Acceptance in the General Population: A Cross-Sectional Survey about Concerns and Expectations. *Informatics*, *9*(1). https://doi.org/10.3390/informatics9010024

Singh, N. K. (2020). Near-field communication (NFC): An alternative to RFID in libraries. *Information Technology and Libraries*, *39*(2). https://doi.org/10.6017/ITAL.V39I2.11811

Strömmer, E., Kaartinen, J., Pärkkä, J., Ylisaukko-oja, A., & Korhonen, I. (2006). Application of near field communication for health monitoring in daily life. *Annual International Conference of the IEEE Engineering in Medicine and Biology - Proceedings*, 3246–3249. https://doi.org/10.1109/IEMBS.2006.260021

TIOBE. (2023). *TIOBE Index - TIOBE*. https://www.tiobe.com/tiobe-index/

Vikram, N., & Kashwan, K. R. (2016). Design of ISM band RFID reader antenna for IoT applications. *Proceedings of the 2016 IEEE International Conference on Wireless Communications, Signal Processing and Networking, WiSPNET 2016*, 1818–1821. https://doi.org/10.1109/WISPNET.2016.7566454

ANEXOS

ANEXO 1: NORMA ISO/IEC 18092 y el Chip Implante NFC X3

Característica	Norma ISO/IEC 18092:2023	Chip implante NFC X3
Frecuencia de operación	13,56 MHz	Opera a la misma frecuencia
Modos de comunicación	Define modos de comunicación activos y pasivos de NFCIP-1	Soporta estos modos de comunicación
Protocolo de transporte	Especifica un protocolo de transporte	Utiliza este protocolo para el intercambio de datos
Velocidades de bits	106, 212 y 424 kbit/s	Velocidad de Transmisión de 106 kbit/s

ANEXO 2

Discusión Sobre La Viabilidad De La Implementación Del Sistema En El Contexto De La Emergencia Médica.

La implementación del sistema de chip implante NFC en el contexto de la emergencia médica puede ser una valiosa herramienta para la atención rápida y eficiente de los pacientes en situaciones de emergencia. Los profesionales médicos tienen opiniones diversas sobre el uso de esta tecnología en la atención médica de emergencia.

Por un lado, algunos médicos están a favor de la implementación de esta tecnología, ya que permitiría la identificación rápida y precisa del paciente, así como la obtención de información valiosa sobre su historial médico y estado actual de salud. Además, la capacidad de almacenamiento de datos del chip implante NFC X3 permitiría la actualización continua de la información médica del paciente, lo que podría mejorar la atención médica a largo plazo.

Sin embargo, otros médicos están preocupados por la posible invasión de la privacidad y la seguridad del paciente en la implementación de esta tecnología. También se ha señalado que la lectura de los chips implantes NFC en situaciones de emergencia podría requerir la disponibilidad de dispositivos compatibles con NFC, lo que podría ser un obstáculo en algunos entornos de atención médica.

Además, se ha mencionado que la tecnología NFC no está exenta de problemas técnicos, como interferencias de radiofrecuencia, problemas de lectura y escritura, y falta de estandarización. Estos problemas podrían retrasar o interrumpir la atención médica de emergencia, lo que podría ser potencialmente peligroso para el paciente.

A pesar de estas preocupaciones, algunos médicos creen que la implementación del chip implante NFC X3 podría ser beneficioso en situaciones de emergencia médica, siempre y cuando se aborden adecuadamente los problemas de seguridad y privacidad del paciente, y se tomen medidas para garantizar la disponibilidad y confiabilidad de la tecnología.

Es por esa razón que para fundamentar de manera teórica este trabajo de titulación se desarrollaron diversas técnicas de recolección de información en el ámbito médico con profesionales del área para poder tener un contexto más ampliado acerca de cómo desarrollar el tema médico. Entrevistas, mesas redondas, reuniones investigativas, etc., fueron de mucha ayuda para poder validar la parte médica del proyecto con la parte técnica desarrollada en la elaboración de este proyecto.

En primera instancia se tuvo una entrevista con una doctora de la ciudad, quien tiene conocimiento y preparación nacional e internacional en el tratamiento de pacientes en emergencias y en el desarrollo investigativo de la medicina y de la tecnología mencionó: *"El utilizar a los seres humanos como conejillos de indias para probar ciertas tecnologías que no se tienen un conocimiento y un manejo amplio, puede llevar a formar pacientes insensibles quienes no tengan la manera de expresar a su tratante sus dolencias y solamente se necesite un análisis sistemático de su sentir"* (Parra I., 2022).

Por otra parte, en una mesa redonda realizada con un grupo de cinco docentes investigadores de la Carrera de Medicina de la Facultad de Ciencias de la Salud de la Universidad Técnica del Norte, (Anaya J., et al, 2023), Los expertos consultados coincidieron en que el chip implante NFC propuesto representa una alternativa

tecnológica viable para mejorar la rapidez en la obtención de información de los pacientes en situaciones de emergencia, lo cual permitiría una actuación más eficiente para salvaguardar la vida de los pacientes. Sin embargo, expresaron su preocupación acerca de la seguridad de la tecnología, dada la posibilidad de vulnerabilidades que puedan comprometer la confidencialidad y privacidad de la información de los pacientes. Por consiguiente, recomendaron que se realice una investigación exhaustiva en colaboración con el Consejo de Ética de la Carrera de Medicina de la UTN para garantizar que se cumplan los estándares éticos y legales en el uso de la tecnología NFC en el ámbito médico.

A este respecto, los expertos médicos enfatizaron la necesidad de garantizar la privacidad y la protección de los datos de los pacientes, lo que implica una implementación rigurosa de medidas de seguridad en la tecnología NFC. Por otro lado, destacaron que el uso de esta tecnología podría mejorar significativamente la eficiencia en el manejo de las emergencias médicas, especialmente en el contexto de los primeros auxilios y el triage. No obstante, señalaron que es importante considerar los límites de la tecnología NFC y reconocer que no puede reemplazar el juicio clínico y la evaluación médica profesional.

Asimismo, se discutió la necesidad de capacitar adecuadamente al personal médico y de emergencias en el uso de la tecnología NFC y en la interpretación de la información obtenida a través de ella. En este sentido, se subrayó la importancia de fomentar la colaboración y la comunicación efectiva entre los diferentes actores involucrados en el manejo de las emergencias médicas, incluyendo a los pacientes y sus familiares. Por otra parte, se destacó la importancia de la interoperabilidad y la compatibilidad de la

tecnología NFC con otros sistemas y dispositivos médicos existentes, a fin de garantizar una integración efectiva en los procesos de atención médica y de emergencias. También se mencionó la necesidad de llevar a cabo pruebas y validaciones rigurosas del chip implante NFC X3 y su software para asegurar su fiabilidad y seguridad en el uso clínico.

En definitiva, los expertos coincidieron en que la tecnología NFC y, en particular, el chip implante NFC X3 propuesto, presenta potencialidades interesantes para mejorar la eficiencia en la atención médica de emergencia. No obstante, destacaron la necesidad de realizar una investigación rigurosa y colaborativa que garantice la seguridad, privacidad y ética en su uso. Además, se enfatizó la importancia de la capacitación del personal médico y de la interoperabilidad con otros sistemas médicos existentes.

En una reunión científica con la Dra. Salome Gordillo, Coordinadora de la Carrera de Medicina de la FCCSS de la UTN, se llegó al acuerdo de que la investigación científica en el ámbito médico debía ser rigurosa y contar con la aprobación del Consejo de Ética correspondiente. En este sentido, se consideró que la propuesta tecnológica del chip implante NFC X3 debía ser evaluada no solo desde una perspectiva técnica, sino también bioética y médica.

Se acordó que la Carrera de Medicina llevaría a cabo el análisis médico y bioético de la propuesta tecnológica para asegurar la seguridad y viabilidad del uso del chip implante NFC X3 en el contexto de la emergencia médica. Se discutieron las posibles ventajas y desventajas de la implementación del sistema en este ámbito, incluyendo la agilidad en el acceso a la información del paciente y la rapidez en la toma de decisiones para salvar la vida de este.

Asimismo, se destacó la importancia de considerar las implicancias éticas y legales de la implementación del sistema, y se acordó que se debería seguir un proceso de investigación riguroso para evaluar su seguridad y eficacia en el contexto de la atención médica de emergencia. Se hizo hincapié en la necesidad de proteger la privacidad y confidencialidad de la información del paciente y garantizar que la tecnología no se utilice de manera indebida.

Más adelante, se procedió a llevar a cabo la implantación del chip NFC X3 en el sujeto de prueba bajo la supervisión de un cirujano principal del Hospital del IESS. Durante una entrevista con (Flores R., 2023), quien estuvo a cargo del procedimiento, se mencionó que la tecnología propuesta resulta viable en vista de la creciente aceptación de los "cyborgs" a nivel mundial. Estos individuos se caracterizan por haber sido implantados con dispositivos electrónicos para complementar el funcionamiento de sus organismos.

En base a las opiniones de expertos médicos, se concluye que la implementación del sistema chip implante NFC en el contexto de la emergencia médica es una alternativa viable y efectiva para la respuesta temprana y el acceso rápido a la información del paciente en la etapa previa al triage, lo que puede ser de gran ayuda en la toma de decisiones y el tratamiento oportuno del paciente.

Se puede afirmar que el uso de tecnología NFC en la medicina es una tendencia en crecimiento y que la implementación del chip implante NFC X3 representa una alternativa prometedora y viable para la gestión de emergencias médicas. No obstante, es necesario continuar con investigaciones y estudios adicionales para perfeccionar la tecnología y garantizar su eficacia y seguridad en todo momento.

ANEXO 3: User Guide Chip Implante NFC X3 (Becker S, 2021)

Our implants have different features and chipsets. Each of them is tailored to specific applications, so that size, connectivity, chipset and use form a symbiosis. Here you will find a small technical overview of your implant.

Model	Chipset	Manufacturer	Memory	Writable	Frequency
X1	EM4100 EM4200 EM4305	EM Microelectronic	8Bytes 8Bytes 512Byte	- - 512Byte	125kHz (LF)
X2	NTAG216	NXP®	888Bytes	556Byte	13,56MHz (HF)
X3 / Elite	M1 Classic Fudan S50	NXP® Fudan	1024Byte 4096Byte	716Byte	13,56MHz (HF)
X4	DESFire	NXP®	2048Byte 4096Byte 8192Byte	2048Byte	13,56MHz (HF)
X Range	iCode Sli	NXP®	106Byte	106Byte	13,56MHz (HF)

Compatibility

There are different approaches to detect or interpret compatibility with existing devices. Most modern reading antennas, which are coupled to relays to open doors, for example, support MIFARE Classic® and compatible chips such as the Fudan S50. Older locking systems may well use the 125Khz frequency (X1). If you try to read in the transponder card using a smartphone (NFC Tools app), the app will show you the chipset that is being used. However, if the smartphone does not respond to the transponder, it is possibly 125kHz technology.

Anexo 4. Codigo MainActivity.kt

```
//Nombre del aplicativo
package com.example.nfcapp

// Importa las librerias necesarias
import android.nfc.NfcAdapter
import android.nfc.Tag
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.util.Log
import android.widget.CompoundButton
import android.widget.Toast
import androidx.databinding.DataBindingUtil
import androidx.lifecycle.ViewModelProvider
import com.example.nfcapp.databinding.ActivityBinder
import com.google.android.material.snackbar.Snackbar
import kotlinx.coroutines.flow.collectLatest
import kotlinx.coroutines.launch

public class MainActivity : AppCompatActivity,
CompoundButton.OnCheckedChangeListener, NfcAdapter.ReaderCallback {

    // Define un objeto acompañante para almacenar el nombre de la
clase
    companion object {
        // Obtiene el nombre de la clase
        private val TAG = MainActivity::class.java.getSimpleName()
    }

    // Variables de instancia
    private var binder : ActivityBinder? = null
    private val viewModel : MainViewModel by lazy {
ViewModelProvider(this).get(MainViewModel::class.java)  }

    // Constructor vacío

    constructor() {

    }



    override fun onCreate(savedInstanceState : Bundle?) {
        // Empieza a crear la interfaz de usuario desde
activity_main.xml
        binder = DataBindingUtil.setContentView(this@MainActivity,
R.layout.activity_main)
        // Establece el ViewModel y el ciclo de vida
        binder?.setViewModel(viewModel)
        binder?.setLifecycleOwner(this@MainActivity)
        super.onCreate(savedInstanceState)
        // Configura el listener para el botón de alternancia

 binder?.toggleButton?.setOnCheckedChangeListener(this@MainActivity)
        // Inicia corutinas para realizar operaciones asíncronas
```

```
        Coroutines.main(this@MainActivity, { scope ->
            scope.launch( block = {
binder?.getViewModel()?.observeNFCStatus()?.collectLatest ( action = {
status -> Log.d(TAG, "observeNFCStatus $status")// Observa el estado
NFC del ViewModel y registra el estado en el Logcat
                if (status == NFCStatus.NoOperation)
NFCManager.disableReaderMode(this@MainActivity, this@MainActivity) //
Si el estado es NoOperation, deshabilita el modo de lector NFC
                else if (status == NFCStatus.Tap)
NFCManager.enableReaderMode(this@MainActivity, this@MainActivity,
this@MainActivity, viewModel.getNFCFlags(), viewModel.getExtras()) //
Si el estado es Tap, habilita el modo de lector NFC
            }) })
            scope.launch( block = {
binder?.getViewModel()?.observeToast()?.collectLatest ( action = {
message -> Log.d(TAG, "observeToast $message")// Registra el mensaje
en el Logcat
                Toast.makeText(this@MainActivity,            message,
Toast.LENGTH_LONG).show()// Observa el mensaje de Toast del ViewModel
            }) })
            scope.launch( block = {
binder?.getViewModel()?.observeTag()?.collectLatest ( action = { tag -
> Log.d(TAG, "observeTag $tag")// Observa los datos de la etiqueta del
ViewModel y registra los datos en el LogCat
                binder?.textViewExplanation?.setText(tag)// Muestra el
mensaje de Toast y Establece el texto del TextView con los datos de la
etiqueta
            }) })
        })
    }

    override fun onCheckedChanged(buttonView : CompoundButton?,
isChecked : Boolean) {
        if (buttonView == binder?.toggleButton)
        // Actualiza el estado NFC en el ViewModel
            viewModel.onCheckNFC(isChecked)
    }

    override fun onTagDiscovered(tag : Tag?) {
        // Lee los datos de la etiqueta y actualiza el ViewModel
        binder?.getViewModel()?.readTag(tag)
    }

    private fun launchMainFragment() {
        if
(getSupportFragmentManager().findFragmentByTag(MainFragment::class.jav
a.getSimpleName()) == null)
        // Agrega el fragmento Main al FrameLayout
            getSupportFragmentManager().beginTransaction()
                .add(R.id.frame_layout, MainFragment.newInstance(),
MainFragment::class.java.getSimpleName())

.addToBackStack(MainFragment::class.java.getSimpleName())
                .commit()
    }
}
```

Anexo 5: Código MainFragment.kt

```
// Nombre del Aplicativo
package com.example.nfcapp

// Importa las librerias necesarias
import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.CompoundButton
import android.widget.Toast
import androidx.databinding.DataBindingUtil
import androidx.fragment.app.Fragment
import androidx.lifecycle.ViewModelProvider
import com.example.nfcapp.databinding.FragmentBinder
import kotlinx.coroutines.flow.collectLatest
import kotlinx.coroutines.launch

// Declarar la clase MainFragment
class MainFragment : Fragment, CompoundButton.OnCheckedChangeListener
{

    companion object {
        private val TAG : String =
MainFragment::class.java.getSimpleName()

        // Esta función estática se utiliza para crear una nueva
instancia de MainFragment.
        public fun newInstance() : MainFragment = MainFragment()
    }

    private var binder : FragmentBinder? = null
    private val viewModel : MainViewModel by lazy {
ViewModelProvider(this).get(MainViewModel::class.java)  }

    // Constructor vacío
    constructor() {

    }

    override fun onCreateView(inflater : LayoutInflater, container :
ViewGroup?, savedInstanceState : Bundle?) : View? {
        // Elevar la vista del fragmento desde el archivo de diseño
AndoridManifest.xml
        binder = DataBindingUtil.inflate(inflater,
R.layout.fragment_main,container,false)
        // Establecer el ViewModel para la vista
        binder?.setViewModel(viewModel)
        // Establecer el ciclo de vida de esta vista
        binder?.setLifecycleOwner(this@MainFragment)
        // Devolver la vista elevada o la vista predeterminada si no
se puede acceder al chip
        return binder?.root ?: super.onCreateView(inflater, container,
savedInstanceState)
    }
```

```
    override fun onViewCreated(view : View, savedInstanceState :
Bundle?) {
        // Usar Coroutines para realizar operaciones asincrónicas
        Coroutines.main(this@MainFragment, { scope ->
            scope.launch( block = {
binder?.getViewModel()?.observeToast()?.collectLatest ( action = {
message ->
                Toast.makeText(requireContext(), message,
Toast.LENGTH_LONG).show()                 // Observar los mensajes de
Toast desde el ViewModel y mostrarlos en un Toast
            }) })
            scope.launch( block = {
binder?.getViewModel()?.observeTag()?.collectLatest ( action = { tag -
>
                binder?.textViewExplanation?.setText(tag)
// Observar los datos de la etiqueta desde el ViewModel y establecer
el texto en una vista de texto
            }) })
        })
        // Llamar al método original
        super.onViewCreated(view, savedInstanceState)
    }

    override fun onCheckedChanged(buttonView : CompoundButton?,
isChecked : Boolean) {
        if (buttonView == binder?.toggleButton)
        // Actualizar el estado del ViewModel cuando cambia el estado
del botón
        viewModel.onCheckNFC(isChecked)
    }
}
```

Anexo 6: Código MainViewModel.kt

```
// Nombre del Aplicativo (paquete)
package com.example.nfcapp

// Importa las librerias necesarias
import android.app.Application
import android.content.ContentValues
import android.nfc.NfcAdapter
import android.nfc.Tag
import android.nfc.NdefMessage
import android.nfc.NdefRecord
import android.nfc.tech.Ndef
import android.nfc.tech.MifareClassic
import android.nfc.tech.MifareUltralight
import android.os.Bundle
import android.util.Log
import androidx.lifecycle.AndroidViewModel
import kotlinx.coroutines.flow.*
import java.text.DateFormat
import java.text.SimpleDateFormat
import java.util.*
```

```
import kotlin.experimental.and

// Declarar la clase MainViewModel
public class MainViewModel : AndroidViewModel {
    //Declaracion de Variables
    companion object {
        private val TAG = MainViewModel::class.java.getSimpleName()
        private const val prefix = "android.nfc.tech."
    }

    private val liveNFC : MutableStateFlow<NFCStatus?>
    private val liveToast : MutableSharedFlow<String?>
    private val liveTag : MutableStateFlow<String?>

    constructor(application : Application) : super(application) {
Log.d(TAG, "constructor")
        // Inicialización del ViewModel
        liveNFC = MutableStateFlow(null)
        liveToast = MutableSharedFlow()
        liveTag = MutableStateFlow(null)
    }
    //Region para la generación y manejo de mensajes Toast
    private fun updateToast(message : String) {
Coroutines.io(this@MainViewModel, { //Acualizar Toast
        liveToast.emit(message) //Emitir Toast
    } ) }

    private suspend fun postToast(message : String) { Log.d(TAG,
"postToast(${message})") //Suspender Toast
        liveToast.emit(message) //Emitir Toast
    }
    // Métodos para observar los mensajes de Toast
    public fun observeToast() : SharedFlow<String?> {
        return liveToast.asSharedFlow()
    }
    //Fin de esta Region de Toast

    // Métodos para obtener las configuraciones NFC
    public fun getNFCFlags() : Int {
        //Obtencion de las banderas NFC
        return NfcAdapter.FLAG_READER_NFC_A or
        NfcAdapter.FLAG_READER_NFC_B or
        NfcAdapter.FLAG_READER_NFC_F or
        NfcAdapter.FLAG_READER_NFC_V or
        NfcAdapter.FLAG_READER_NFC_BARCODE //or
NfcAdapter.FLAG_READER_SKIP_NDEF_CHECK
    }

    public fun getExtras() : Bundle {
        // Obtiene las opciones extras NFC
        val options : Bundle = Bundle();
        options.putInt(NfcAdapter.EXTRA_READER_PRESENCE_CHECK_DELAY,
30000); //Funciones de lectura
        return options
    }
    //Metodos relacionados con el estado NFC
```

```
    public fun onCheckNFC(isChecked : Boolean) {
Coroutines.io(this@MainViewModel, { Log.d(TAG,
"onCheckNFC(${isChecked})")
        if (isChecked) {
            postNFCStatus(NFCStatus.Tap)
        } else {
            postNFCStatus(NFCStatus.NoOperation)
            postToast("NFC esta apagado, Enciendalo!")
        }
    } ) } //Cambia el estado de NFC y muestra los Mensajes

    public fun readTag(tag : Tag?) {
Coroutines.default(this@MainViewModel, { Log.d(TAG, "readTag(${tag}
${tag?.getTechList()})")
        postNFCStatus(NFCStatus.Process)
        val stringBuilder : StringBuilder = StringBuilder()
        val id : ByteArray? = tag?.getId()
        val ndef = Ndef.get(tag)
        if (ndef != null) {
            try {
                ndef.connect()

                // Lee el mensaje NDEF de la etiqueta
                val ndefMessage = ndef.ndefMessage
                val name = ndefMessage.records[0].payload
                val textData0 = String(name)

                val birth = ndefMessage.records[1].payload
                val textData1 = String(birth)

                val Loc = ndefMessage.records[2].payload
                val textData2 = String(Loc)

                val alergy = ndefMessage.records[3].payload
                val textData3 = String(alergy)

                val blood = ndefMessage.records[4].payload
                val textData4 = String(blood)

                val dona = ndefMessage.records[5].payload
                val textData5 = String(dona)

                val clinhist = ndefMessage.records[6].payload
                val textData6 = String(clinhist)

                val emcon = ndefMessage.records[7].payload
                val textData7 = String(emcon)

                val enlace = ndefMessage.records[8].payload
                val textData8 = String(enlace)

                // Muestra el contenido en el TextView o donde desees
                //liveTag.emit("${getDateTimeNow()} \n ${textData}")
                stringBuilder.append("${textData0}\n")
                stringBuilder.append("${textData1}\n")
                stringBuilder.append("${textData2}\n")
```

```
                stringBuilder.append("${textData3}\n")
                stringBuilder.append("${textData4}\n")
                stringBuilder.append("${textData5}\n")
                stringBuilder.append("${textData6}\n")
                stringBuilder.append("${textData7}\n")
                stringBuilder.append("${textData8}\n")


                ndef.close()
                postNFCStatus(NFCStatus.Read)
            } catch (e: Exception) {
                // Maneja cualquier excepción que pueda ocurrir al
leer la etiqueta NDEF
                Log.e(TAG, "Error al leer la etiqueta NDEF:
${e.message}")
                postToast("Error al leer la etiqueta NDEF:
${e.message}")
            }
        } else {
            // Si la etiqueta no es compatible con NDEF, muestra un
mensaje
            postToast("La etiqueta no es compatible con NDEF.")
        }

        stringBuilder.append("\n Tag ID (hex): ${getHex(id!!)} \n")
//Lectura de Datos
        stringBuilder.append("Tag ID (dec): ${getDec(id)} \n")
//Lectura de Datos
        stringBuilder.append("Tag ID (reversed): ${getReversed(id)}
\n") //Lectura de Datos
        stringBuilder.append("Technologies: ") //Lectura de Datos
        tag.getTechList().forEach { tech ->
            stringBuilder.append(tech.substring(prefix.length))
            stringBuilder.append(", ")
        }
        stringBuilder.delete(stringBuilder.length - 2,
stringBuilder.length)
        tag.getTechList().forEach { tech ->
            if (tech.equals(MifareClassic::class.java.getName())) {
                stringBuilder.append('\n')
                val mifareTag : MifareClassic = MifareClassic.get(tag)
                val type : String
                if (mifareTag.getType() == MifareClassic.TYPE_CLASSIC)
type = "Classic"
                else if (mifareTag.getType() ==
MifareClassic.TYPE_PLUS) type = "Plus"
                else if (mifareTag.getType() ==
MifareClassic.TYPE_PRO) type = "Pro"
                else type = "Unknown"
                stringBuilder.append("Mifare Classic type: $type \n")
                stringBuilder.append("Mifare size:
${mifareTag.getSize()} bytes \n")
                stringBuilder.append("Mifare sectors:
${mifareTag.getSectorCount()} \n")
                stringBuilder.append("Mifare blocks:
${mifareTag.getBlockCount()}")
```

```
            } //Lectura de Protocolo MIFARE del chip NFC x3
            if (tech.equals(MifareUltralight::class.java.getName())) {
                stringBuilder.append('\n');
                val mifareUlTag : MifareUltralight =
MifareUltralight.get(tag);
                val type : String
                if (mifareUlTag.getType() ==
MifareUltralight.TYPE_ULTRALIGHT) type = "Ultralight"
                else if (mifareUlTag.getType() ==
MifareUltralight.TYPE_ULTRALIGHT_C) type = "Ultralight C"
                else type = "Unkown"
                stringBuilder.append("Mifare Ultralight type: ");
                stringBuilder.append(type)
            }
        }
        Log.d(TAG, "Datum: $stringBuilder")
        Log.d(ContentValues.TAG, "dumpTagData Return \n
$stringBuilder")
        postNFCStatus(NFCStatus.Read)
        liveTag.emit("${getDateTimeNow()} \n ${stringBuilder}")
    } ) } //Lectura de la etiqueta NFC y sus datos


    public fun updateNFCStatus(status : NFCStatus) {
Coroutines.io(this@MainViewModel, {
        postNFCStatus(status)
    } ) }        // Actualiza el estado NFC y muestra mensajes si es
necesario


    private suspend fun postNFCStatus(status : NFCStatus) { Log.d(TAG,
"postNFCStatus(${status})")
        if (NFCManager.isSupportedAndEnabled(getApplication())) {
            liveNFC.emit(status)
        }
        else if (NFCManager.isNotEnabled(getApplication())) {
            liveNFC.emit(NFCStatus.NotEnabled)
            postToast("Active el NFC de su dispositivo!")
            liveTag.emit("Active el NFC de su dispositivo!")
        } else if (NFCManager.isNotSupported(getApplication())) {
            liveNFC.emit(NFCStatus.NotSupported)
            postToast("NFC No Soportado!")
            liveTag.emit("NFC No Soportado!")
        }
        if (NFCManager.isSupportedAndEnabled(getApplication()) &&
status == NFCStatus.Tap) {
            liveTag.emit("Proceda a la Lectura!")
        } else {
            liveTag.emit(null)
        }
    } //Mensajes de funciones de NFC

    public fun observeNFCStatus() : StateFlow<NFCStatus?> {
        return liveNFC.asStateFlow() //Devuelve el estado del NFC
    }
```

```
    //Metodos para obtener la información de las etiquetas
    private fun getDateTimeNow() : String { Log.d(TAG,
"getDateTimeNow()")
        // Obtiene la fecha y hora actual
        val TIME_FORMAT : DateFormat =
SimpleDateFormat.getDateTimeInstance()
        val now : Date = Date()
        Log.d(ContentValues.TAG,"getDateTimeNow()  Return
${TIME_FORMAT.format(now)}")
        return TIME_FORMAT.format(now)
    }

    private fun getHex(bytes : ByteArray) : String {
        // Convierte un arreglo de bytes a una representación
hexadecimal
        val sb = StringBuilder()
        for (i in bytes.indices.reversed()) {
            val b : Int = bytes[i].and(0xff.toByte()).toInt()
            if (b < 0x10) sb.append('0')
            sb.append(Integer.toHexString(b))
            if (i > 0)
                sb.append(" ")
        }
        return sb.toString()
    }

    private fun getDec(bytes : ByteArray) : Long {
        // Convierte un arreglo de bytes a un número decimal
        Log.d(TAG, "getDec()")
        var result : Long = 0
        var factor : Long = 1
        for (i in bytes.indices) {
            val value : Long = bytes[i].and(0xffL.toByte()).toLong()
            result += value * factor
            factor *= 256L
        }
        return result
    }

    private fun getReversed(bytes : ByteArray) : Long {
        // Convierte un arreglo de bytes invertido a un número decimal
        Log.d(TAG, "getReversed()")
        var result : Long = 0
        var factor : Long = 1
        for (i in bytes.indices.reversed()) {
            val value = bytes[i].and(0xffL.toByte()).toLong()
            result += value * factor
            factor *= 256L
        }
        return result
    }

    public fun observeTag() : StateFlow<String?> {
        return liveTag.asStateFlow() //Devuelve el estado de la
etiqueta
```

```
    }
}
```

Anexo 7: Código Corutines.kt

```
package com.example.nfcapp

import androidx.appcompat.app.AppCompatActivity
import androidx.fragment.app.DialogFragment
import androidx.fragment.app.Fragment
import androidx.lifecycle.*
import
com.google.android.material.bottomsheet.BottomSheetDialogFragment
import kotlinx.coroutines.CoroutineScope
import kotlinx.coroutines.Dispatchers
import kotlinx.coroutines.launch

public object Coroutines {
    //region de contextos de UI
    // Estas funciones se utilizan para ejecutar tareas en el hilo
principal de la interfaz de usuario (UI).
    // Ejecutar en el hilo principal, dentro de una Coroutine.

    fun main(work : suspend (() -> Unit)) =
        CoroutineScope(Dispatchers.Main.immediate).launch {
            work()
        }
    // Ejecutar en el hilo principal de una actividad
(AppCompatActivity).

    fun main(activity : AppCompatActivity, work : suspend ((scope :
CoroutineScope) -> Unit)) =
        activity.lifecycleScope.launch {

activity.getLifecycle().repeatOnLifecycle(Lifecycle.State.CREATED) {
                work(this)
            }
        }

    // Ejecutar en el hilo principal de un fragmento
(BottomSheetDialogFragment).

    fun main(fragment : BottomSheetDialogFragment, work : suspend
((scope : CoroutineScope) -> Unit)) =
        fragment.lifecycleScope.launch {

fragment.getLifecycle().repeatOnLifecycle(Lifecycle.State.STARTED) {
                work(this)
            }
        }
    // Ejecutar en el hilo principal de un fragmento (DialogFragment).
    fun main(fragment : DialogFragment, work : suspend ((scope :
CoroutineScope) -> Unit)) =
        fragment.lifecycleScope.launch {
```

```
fragment.getLifecycle().repeatOnLifecycle(Lifecycle.State.STARTED)  {
                work(this)
            }
        }
    // Ejecutar en el hilo principal de un fragmento (Fragment).
    fun main(fragment : Fragment, work : suspend ((scope :
CoroutineScope) -> Unit)) =
        fragment.lifecycleScope.launch {

fragment.getLifecycle().repeatOnLifecycle(Lifecycle.State.STARTED)  {
                work(this)
            }
        }
    //final de la region
    //region Operaciones de E/S (Input/Output)
    // Estas funciones se utilizan para ejecutar tareas de E/S en
hilos de fondo.

    // Ejecutar en un hilo de fondo, dentro de una Coroutine.
    fun io(work : suspend (() -> Unit)) =
        CoroutineScope(Dispatchers.IO).launch {
            work()
        }
    // Ejecutar en un hilo de fondo dentro del ámbito de un ViewModel.
    fun io(viewModel : ViewModel, work : suspend (() -> Unit)) {
        viewModel.viewModelScope.launch(Dispatchers.IO) {
            work()
        }
    }
    //fin de la region
    //region Tareas de cómputo intensivo en CPU
    // Estas funciones se utilizan para ejecutar tareas intensivas en
cómputo en hilos de fondo.

    // Ejecutar en un hilo de fondo con cómputo intensivo, dentro de
una Coroutine.
    fun default(work : suspend (() -> Unit)) =
        CoroutineScope(Dispatchers.Default).launch {
            work()
        }
    // Ejecutar en un hilo de fondo con cómputo intensivo dentro del
ámbito de un ViewModel.
    fun default(viewModel : ViewModel, work : suspend (() -> Unit)) =
        viewModel.viewModelScope.launch(Dispatchers.Default) {
            work()
        }
    //final de la region
    //region sin necesidad de correr bajo un contexto especifico
    // Esta función se utiliza cuando no es necesario ejecutar una
tarea en un contexto de hilo específico.

    // Ejecutar en cualquier contexto de hilo, dentro de una
Coroutine.
    fun unconfined(work : suspend (() -> Unit)) =
        CoroutineScope(Dispatchers.Unconfined).launch {
            work()
```

```
        }
    //final de la region
}
```

Anexo 8: Código NFC Manager

```
package com.example.nfcapp

import android.app.Activity
import android.content.Context
import android.nfc.NfcAdapter
import android.os.Build
import android.os.Bundle
import android.util.Log
import androidx.annotation.RequiresApi

public object NFCManager {
    // Etiqueta de registro para el seguimiento de registros
    private val TAG = NFCManager::class.java.getSimpleName()
    // Función para habilitar el "modo lector" NFC
    @RequiresApi(Build.VERSION_CODES.KITKAT)
    public fun enableReaderMode(context : Context, activity :
Activity, callback : NfcAdapter.ReaderCallback, flags : Int, extras :
Bundle) {
        try {

NfcAdapter.getDefaultAdapter(context).enableReaderMode(activity,
callback, flags, extras)
        } catch (ex : UnsupportedOperationException) {
Log.e(TAG,"UnsupportedOperationException ${ex.message}", ex)

        }
    }
    // Función para deshabilitar el "modo lector" NFC
    @RequiresApi(Build.VERSION_CODES.KITKAT)
    public fun disableReaderMode(context : Context, activity :
Activity) {
        try {

NfcAdapter.getDefaultAdapter(context).disableReaderMode(activity)
        } catch (ex : UnsupportedOperationException) {
Log.e(TAG,"UnsupportedOperationException ${ex.message}", ex)

        }
    }
    // Verificar si el dispositivo admite NFC
    public fun isSupported(context : Context) : Boolean {
        val nfcAdapter = NfcAdapter.getDefaultAdapter(context)
        return if (nfcAdapter == null) false
        else true
    }
    // Verificar si el dispositivo NO admite NFC
    public fun isNotSupported(context : Context) : Boolean {
        val nfcAdapter = NfcAdapter.getDefaultAdapter(context)
        return if (nfcAdapter == null) true
```

```
        else false
    }
    // Verificar si NFC está habilitado en el dispositivo
    public fun isEnabled(context : Context) : Boolean {
        val nfcAdapter = NfcAdapter.getDefaultAdapter(context)
        return if (nfcAdapter == null) false
        else nfcAdapter.isEnabled()
    }
    // Verificar si NFC NO está habilitado en el dispositivo
    public fun isNotEnabled(context : Context) : Boolean {
        val nfcAdapter = NfcAdapter.getDefaultAdapter(context)
        return if (nfcAdapter == null) true
        else nfcAdapter.isEnabled().not()
    }
    // Combinar verificaciones de compatibilidad y habilitación de NFC
    public fun isSupportedAndEnabled(context : Context) : Boolean {
        return isSupported(context) && isEnabled(context)
    }
}
```

Anexo 9: Código NFCStatus.kt

```
package com.example.nfcapp

enum class NFCStatus {
    NoOperation,
    Tap,
    Process,
    Confirmation,
    Read,
    Write,
    NotSupported,
    NotEnabled,
}
```

Anexo 10. Código AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission android:name="android.permission.NFC" />
    <uses-feature android:name="android.hardware.nfc"
android:required="true" />
    <!--
    <uses-permission android:name="android.permission.USB_PERMISSION"
/>
    -->
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
```

```
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.NFCApp"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
        </activity>

    </application>

</manifest>
```

ANEXO 11

Manual de Usuario Aplicativo NFC

MANUAL - APLICACIÓN MÓVIL

NFC APP

Creando ciencia...
Construyendo sueños

MANUAL - APLICACIÓN MÓVIL

NFC APP

CONTENIDO

TABLA DE FIGURAS

FIRMAS DE APROBACIÓN

Revisado por:

Ing. Mauricio Domínguez Ms.C.
Director

Elaborado por:

José Camilo Nogales Romero
Tesista

Revisado por:

Ing. Edgar Maya Ms.C.
Asesor

I. INTRODUCCIÓN

Manual Técnico del Sistema de Gestión de Información Médica basado en la tecnología NFC (Near Field Communication). Este documento tiene como objetivo proporcionar una guía completa para comprender, implementar y mantener el sistema diseñado para optimizar los procesos médicos, especialmente en el ámbito del triaje en instituciones de salud.

El sistema, desarrollado mediante la fusión de la metodología XP (eXtreme Programming) con principios de desarrollo de aplicaciones móviles, constituye una solución sólida para mejorar la eficiencia en la atención médica. Utilizando el chip implante NFC X3, posibilita un acceso rápido y seguro a la información médica esencial, agilizando los tiempos de atención y suministrando datos cruciales para la toma de decisiones médicas.

Este manual ha sido concebido como una herramienta completa y accesible para todos los participantes involucrados en la utilización y mantenimiento del sistema, desde los desarrolladores hasta los profesionales de la salud. A lo largo de estas páginas, se abordará la instalación del sistema, los requisitos técnicos, los casos de uso específicos y las mejores prácticas para garantizar un rendimiento óptimo.

II. ALCANCE

El presente proyecto tiene como alcance la agilización del sistema de gestión de información médica previo al triage de emergencias, utilizando el aplicativo desarrollado en el proyecto de titulación y el chip implante NFC X3. A lunes, 20 de noviembre de 2023, el sistema presenta una reducción de tiempo promedio del 30% en cada color del triage Manchester, probado en las Clínica de Simulación de la Universidad Técnica del Norte.

III. TEMAS A TRATAR

- ✓ Información de la Plataforma de Desarrollo y Versiones
- ✓ Configuración de Escritura del chip NFX X3 con el Aplicativo NFC Tools
- ✓ Instalación y Configuración del Aplicativo “NFC APP”
- ✓ Procedimiento de Uso del Aplicativo “NFC APP”

a) Información de la Plataforma de Desarrollo y Versiones

El aplicativo se lo desarrolló en el Software Android Studio Flamingo | 2022.2.1, en lenguaje Kotlin. Las configuraciones de build del aplicativo son:

Figura 1: Configuraciones de Build de SDK y de Android del aplicativo

Por esta razón se determina que el aplicativo es compatible para una versión mínima de Android KiTKat 4.4 con un nivel de API 19 y esta desarrollada para tener un rendimiento mejor en las versiones de Android 13 (Tiramisu), 14 (UpsideDownCake), 15 (Vanilla Ice Cream) y 16 (Vanilla Ice Cream – nivel de API 33).

b) Configuración de Escritura del chip NFX X3 con el Aplicativo NFC Tools

Para realizar el proceso de escritura en el chip, se utiliza el aplicativo NFC Tools, donde se van a registrar cada uno de los bloques de información necesarios dentro del chip, con los datos detallados del paciente que va a portar el chip dentro de su piel, para lo cual se sigue el procedimiento:

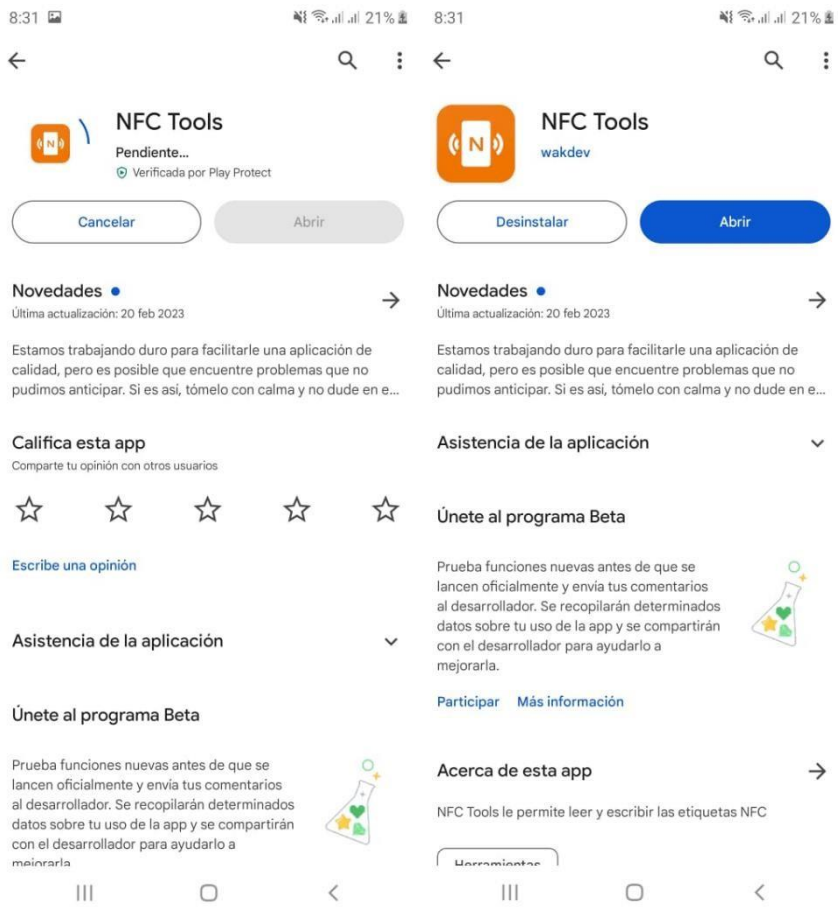


Figura 2: Descarga e instalación del aplicativo NFC Tools

2. Una vez instalado, se ejecuta la aplicación dando los permisos correspondientes para el acceso a NFC del dispositivo móvil

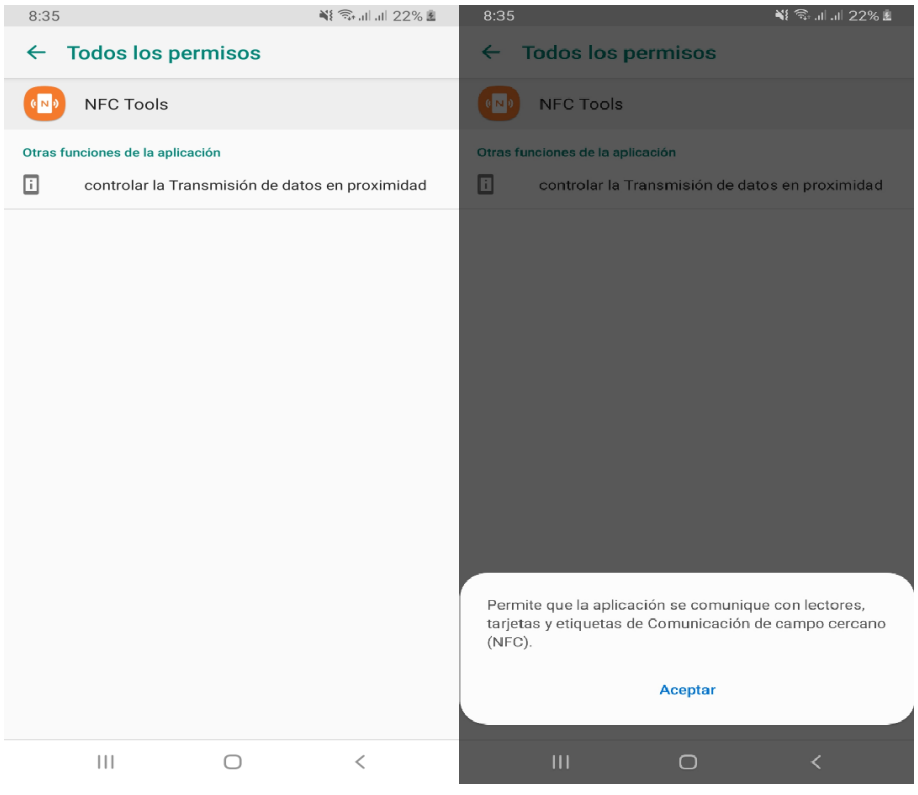


Figura 3: Verificación de permisos NFC de la aplicación NFC Tools

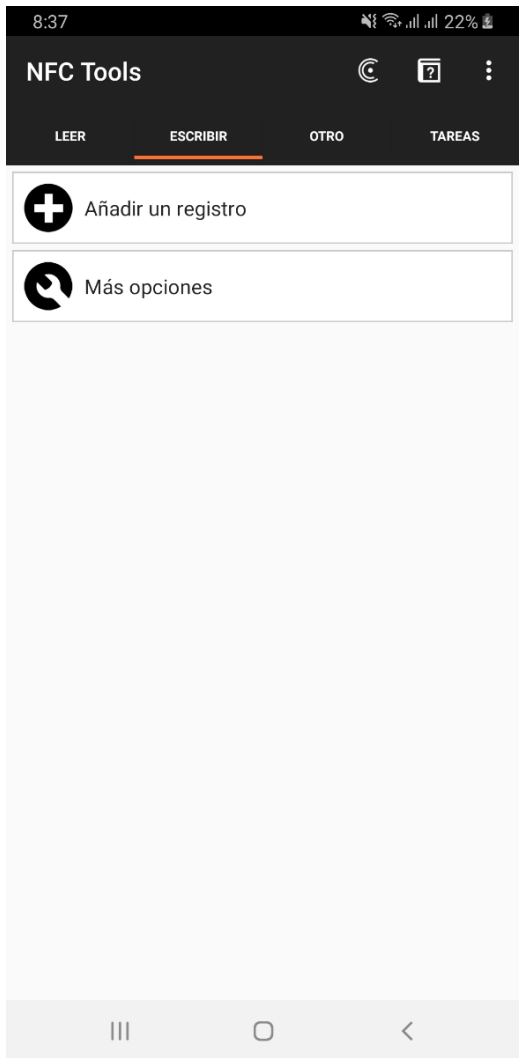


Figura 4: Pestaña para escritura NFC

4. Seleccionar la opción Datos Médicos de Emergencia.

Figura 5: Opción datos de Emergencia dentro de la opción añadir un Registro

5. Ingresar los datos y validar.

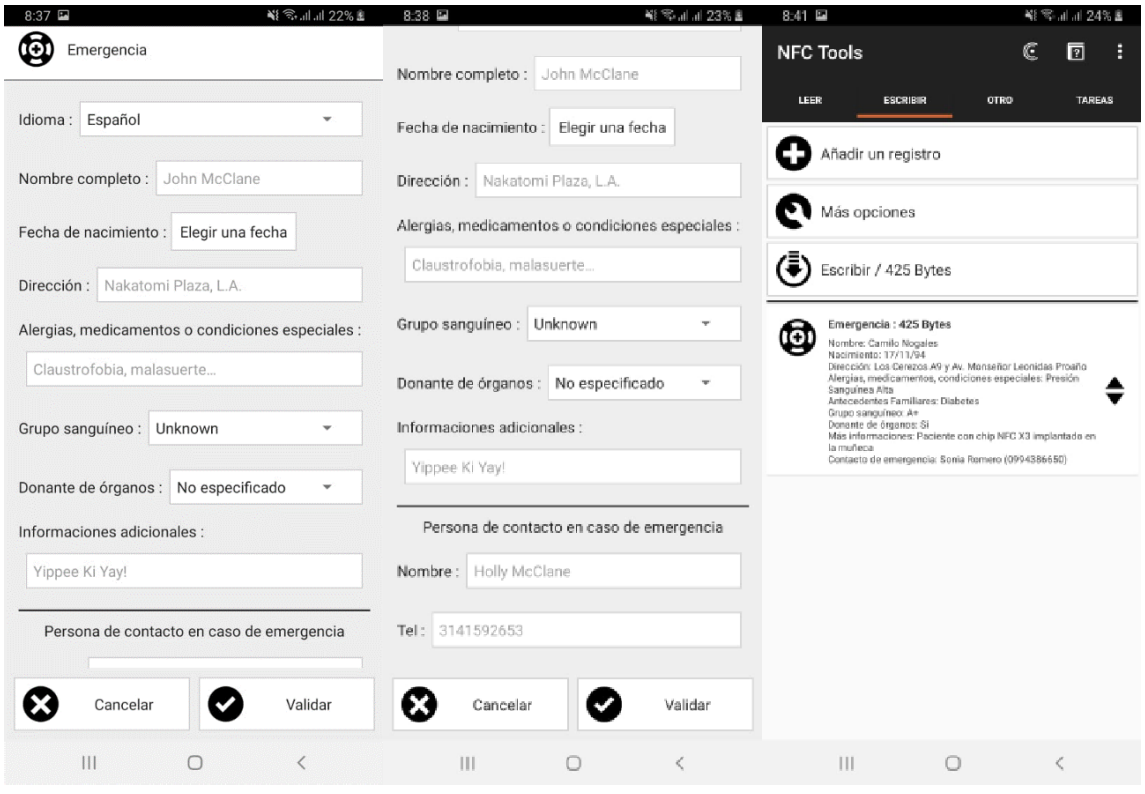


Figura 6: Ingreso de la información del paciente que será escrita en el chip implante

6. Proceder a la escritura acercando el dispositivo móvil hacia la antena NFC del chip implante.

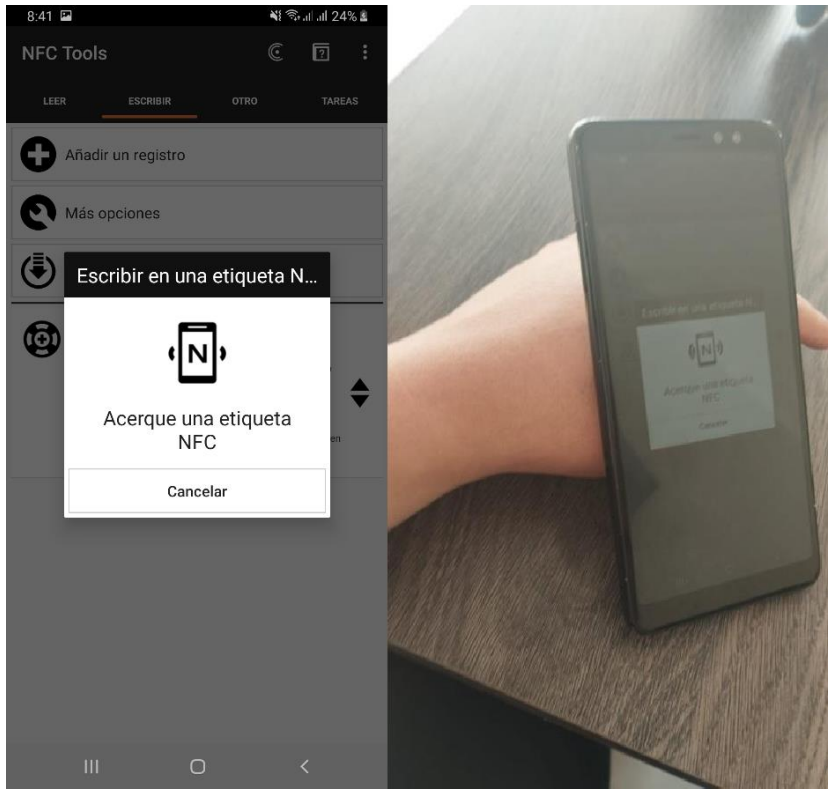


Figura 7: Proceso de escritura del chip implante

7. Verificar la escritura con el mensaje de validación.

Figura 8: Validación de Escritura del chip

8. Ingresar a la opción de escritura de datos url, donde se debe colocar el URL del archivo PDF de la historia clínica del paciente almacenada dentro del chip.

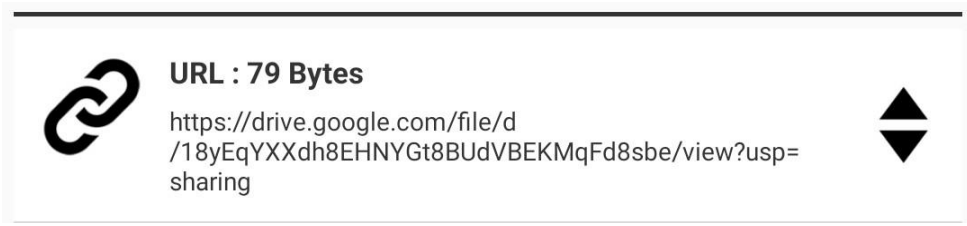


Figura 9: Escritura de la historia clínica dentro del chip como archivo PDF

9. Realizar la escritura nuevamente con el dispositivo hacia el chip.

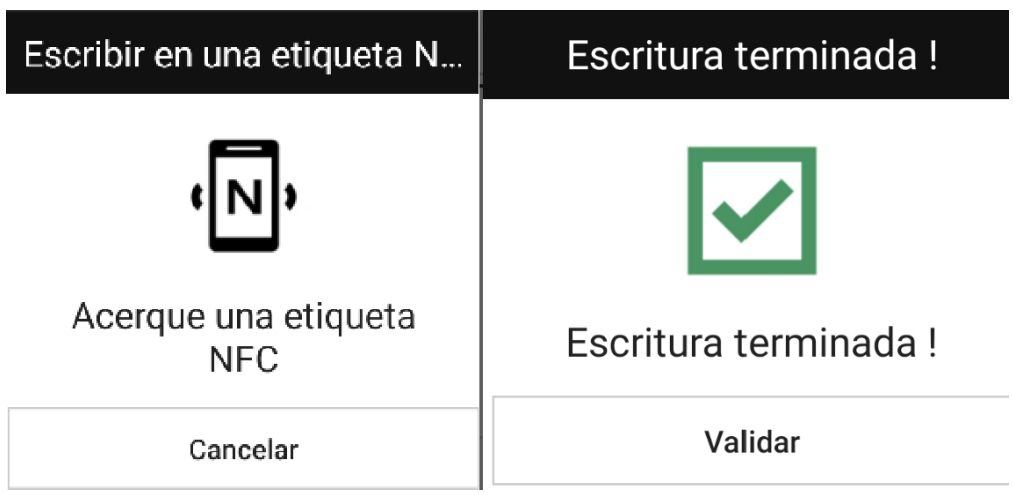


Figura 10: Escritura de la historia clínica dentro del chip

10. Verificar la escritura de la información.

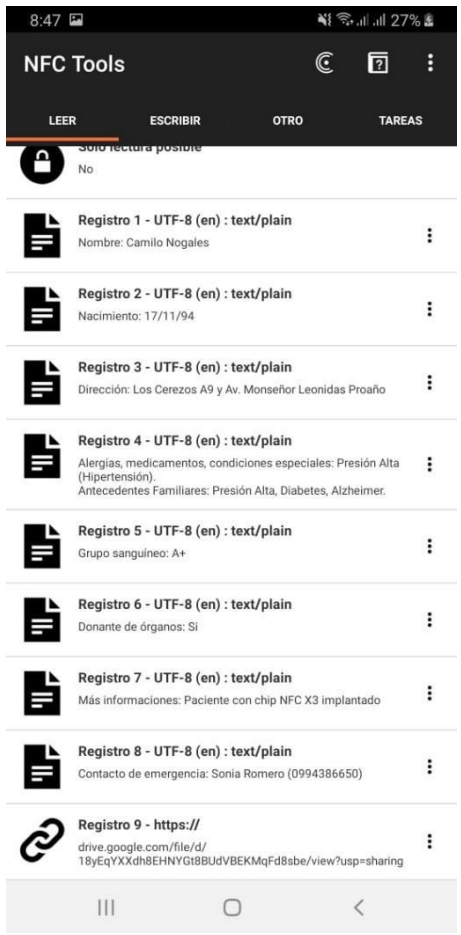


Figura 11: Verificación de escritura completa del chip

De esta manera, el chip implante tiene la información necesaria para proceder con la parte de lectura de este mediante el aplicativo desarrollado NFC APP.

NOTA: En caso de querer volver a ingresar la información, se debe seguir el mismo procedimiento a partir del apartado 2 ya que el chip permite varias escrituras o reescrituras de la información.

c) Instalación y Configuración del Aplicativo "NFC APP"

1. Acceder al enlace: NFC APP UTN para descargar la aplicación "NFC APP", desde un dispositivo móvil Android y descargar el aplicativo con formato .apk.

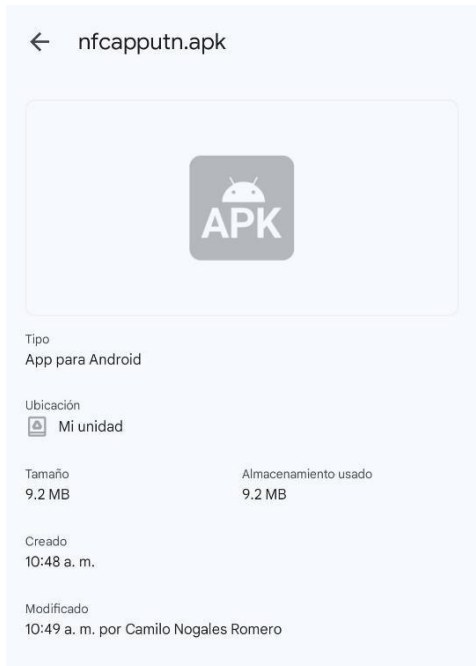


Figura 12: Aplicación almacenada en la nube

2. Una vez descargada la aplicación se procede a instalarla verificando que la opción de permisos para instalar aplicaciones desde orígenes desconocidos este activada en el teléfono móvil.

Figura 13: Verificación previa a la instalación del aplicativo de los permisos para instalar apps de fuentes desconocidas.

NOTA: Si la opción de permisos para instalar aplicaciones de orígenes desconocidos esta activada, seguir con el paso

3. Una vez se accede con clic en configuración, el dispositivo muestra la ventana de configuración donde se muestra que no existen permisos para instalar aplicaciones desde fuentes desconocidas. Se habilita la opción mostrada y Android nos muestra el mensaje de que tenemos claro los riesgos y amenazas que implica instalar aplicaciones de terceros.

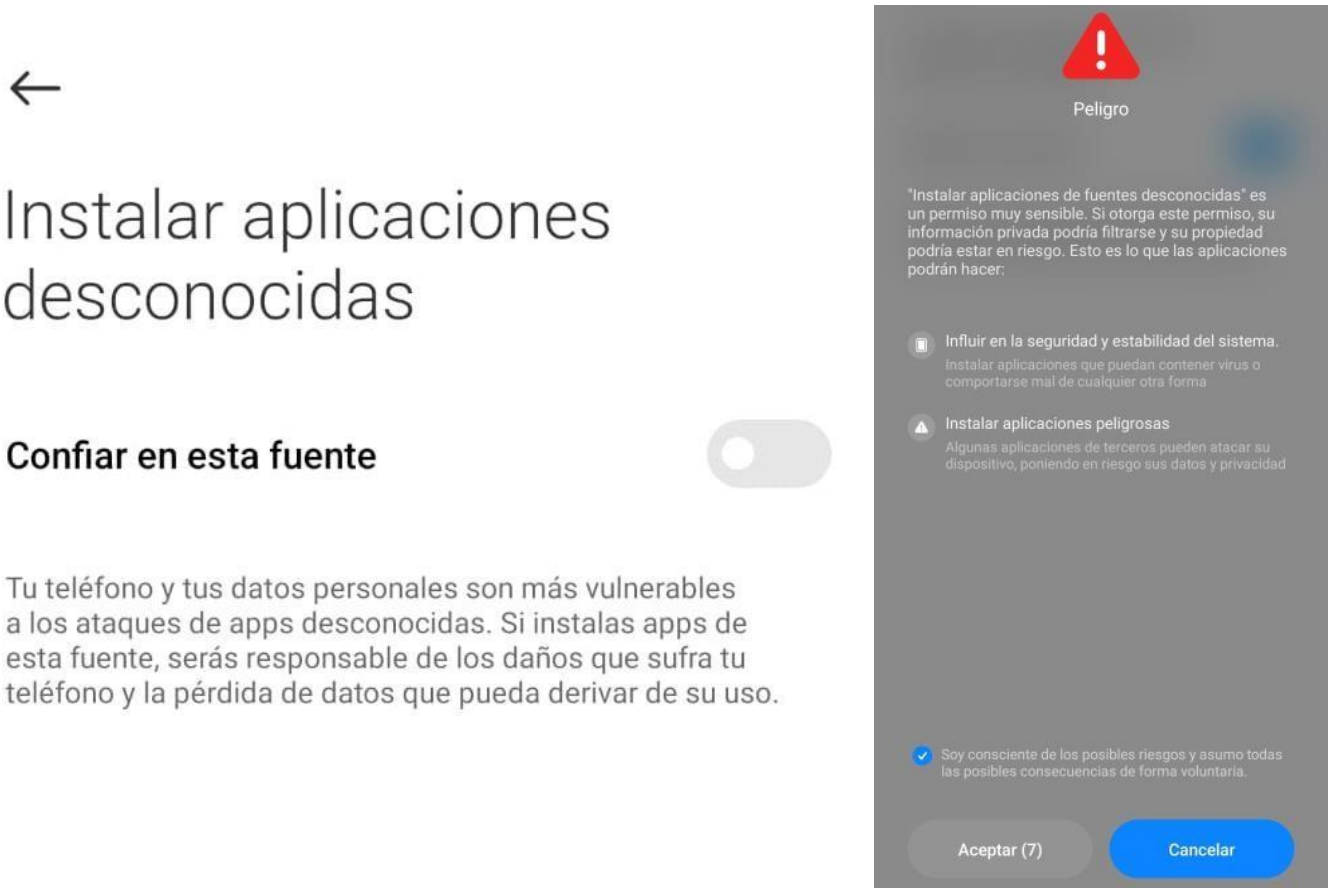


Figura 14: Consentimiento informado de permisos para instalar el aplicativo NFC APP por parte de Android

4. De esta manera se procede a la instalación del aplicativo una vez aceptamos los permisos, aceptando la instalación del aplicativo.

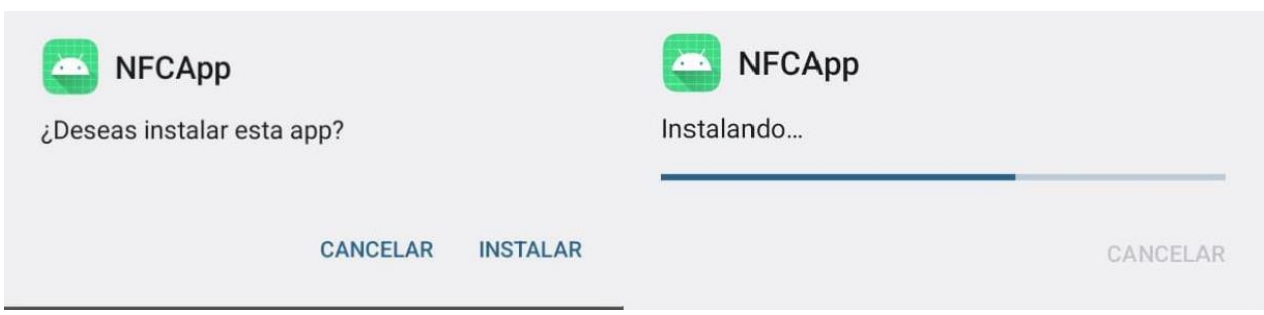


Figura 15: Instalación del aplicativo en el dispositivo Android.

5. Ya instalado, el sistema operativo Android realiza una prueba de seguridad donde se verifica que la aplicación no tiene ningún fin malicioso para su ejecución.

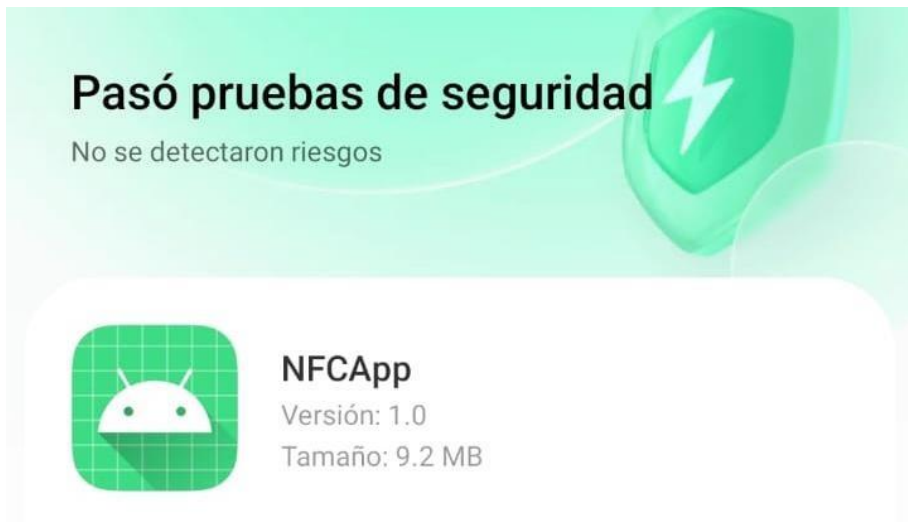


Figura 16: Pruebas de Seguridad del aplicativo.

6. Así tenemos la aplicación ya instalada y lista para su funcionamiento.

Figura 17: App lista para ser usada

d) Procedimiento de Uso del Aplicativo "NFC APP"

1. Una vez instalada la aplicación NFC APP, se ejecuta la misma y se activa el funcionamiento presionando el botón ON, donde un mensaje nos dirá que se proceda a la lectura

Figura 18: Inicio del Aplicativo

NOTA: En caso de no tener encendido el NFC de las configuraciones del dispositivo, la aplicación muestra un mensaje donde se explica al usuario que se enciende el NFC desde el aplicativo para su funcionamiento

2. Se acerca la etiqueta al lector y muestra los datos tantos del paciente como del chip implante en texto plano, lo que indica que la aplicación está lista para su funcionamiento sin necesidad de ejecutarla cada vez que un paciente llega a emergencias, esto considerando que los tiempos de triage son de vital importancia en las emergencias médicas.

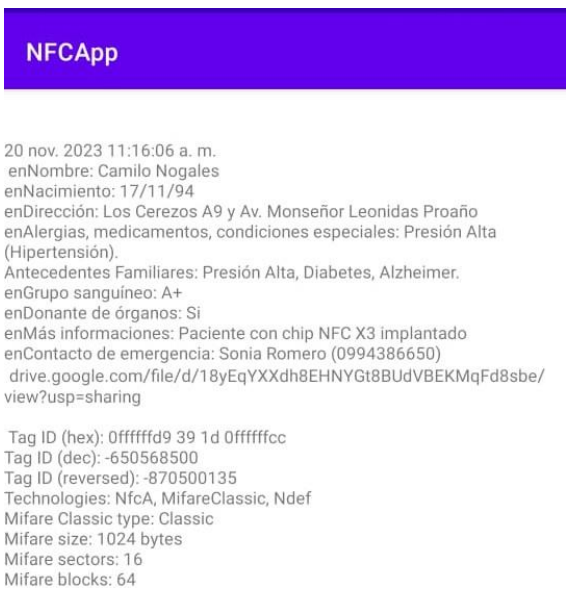


Figura 19: Ejecución de la aplicación

3. La aplicación en segundo plano se ejecuta automáticamente una vez que se detecte una etiqueta nueva, aquí se mostrará la información necesaria del paciente juntamente con su historia clínica de emergencia (Formulario MSP 008)

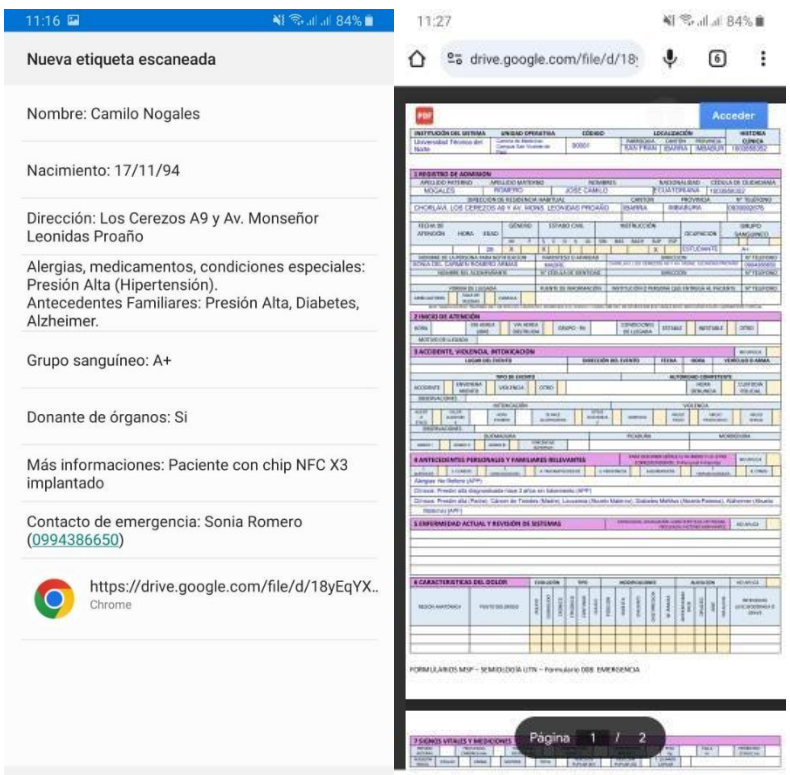


Figura 20: Funcionamiento completo del sistema NFC con la aplicación móvil NFC APP

NOTA: La aplicación no necesita los permisos para acceder al NFC del dispositivo, a diferencia de la aplicación NFC Tools, ya que, por la facilidad de instalación, el paquete de permisos concedidos se encuentra programado dentro del aplicativo.

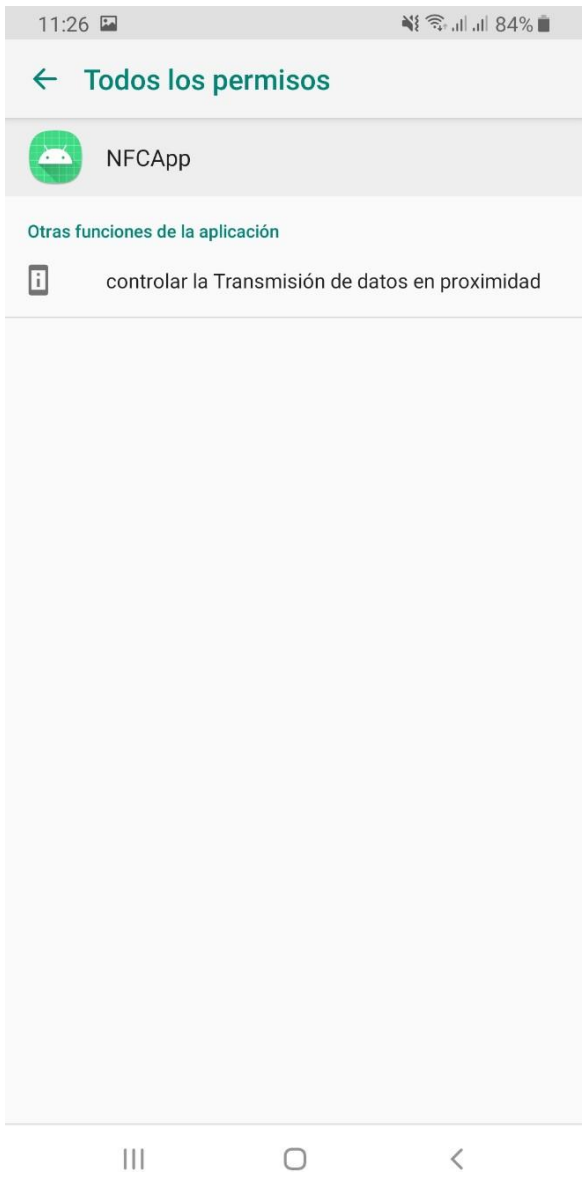


Figura 21: Permisos del Aplicativo

ANEXO 12

Manual Formulario Historia Clínica MSP-008

MINISTERIO DE SALUD PUBLICA DEL ECUADOR

MANUAL DE USO DE LOS FORMULARIOS BASICOS DE LA HISTORIA CLINICA UNICA

2009

Objetivo

Mantener disponible un registro sistemático de los datos recopilados durante la atención de a los usuarios en el servicio de emergencia

Nº	TÍTULOS	SUB TÍTULOS	INSTRUCCIONES DE LLENADO
	ANVERSO: EMERGENCIA (1)		
	INSTITUCIÓN Y UNIDAD OPERATIVA		**CÓDIGO Y LOCALIZACIÓN** **NÚMERO DE HISTORIA CLÍNICA**
1	REGISTRO DE ADMISIÓN		REGISTRAR LOS DATOS COMPLETOS DE IDENTIFICACIÓN Y DIRECCIÓN DEL USUARIO, FECHA Y LUGAR DE NACIMIENTO, PAÍS DE NACIONALIDAD Y GRUPO CULTURAL (SI AMERITA) REGISTRAR LA FECHA DE ADMISION Y LOS DATOS LABORALES REGISTRAR EL NOMBRE DE UN FAMILIAR O AMIGO PARA EL CASO DE QUE SEA NECESARIO LA PRESENCIA URGENTE, MARCAR "X" EN LA FORMA DE LLEGADA DOSCRIBIR LA FUENTE DE INFORMACIÓN, INSTITUCIÓN O PERSONA QUE ENTREGA AL PACIENTE Y NÚMERO TELEFÓNICO RESPECTIVO
2	INICIO DE ATENCIÓN Y MOTIVO		REGISTRAR LA HORA Y MARCAR "X" EN LA CAUSA QUE OCASIONA LA LLEGADA A EMERGENCIA MARCAR "X" SI SE HA NOTIFICADO LA EMERGENCIA A LA POLICIA Y SI EXISTE OTRO MOTIVO DE CONSULTA (ESPECIFICAR) REGISTRAR EL **GRUPO SANGUÍNEO** Y FACTOR Rh, SI ESTÁ DISPONIBLE
3	ENFERMEDAD ACTUAL Y REVISIÓN DE SISTEMAS		MARCAR "X" EN LAS OPCIOÑES SEÑALADAS Y REGISTRAR EL RESULTADO DEL INTERROGATORIO SOBRE CRONOLOGÍA, LOCALIZACIÓN, CARACTERÍSTICAS, INTENSIDAD, FRECUENCIA Y FACTORES AGRAVANTES DEL PROBLEMA
4	ACCIDENTE, VIOLENCIA, INTOXICACIÓN, ENVENENAMIENTO Y QUEMADURA		REGISTRAR LA FECHA,, EL LUGAR (HOGAR, VIA PÚBLICA,, FABRICA) Y DIRECCIÓN DONDE OCURRIÓ EL EVENTO MARCAR "X" SI EL PACIENTE ACUDE CON **CUSTODIA POLICIAL** MARCAR "X" SEGÚN EL TIPO DE EMERGENCIA Y ESCRIBIR LAS OBSERVACIONES MARCAR "X" SI EL PACIE TE TIENE ALIENTO ETÍLICO REGISTRAR EL VALOR ALCOCHECK, SI ESTÁ DISPONIBLE
5	ANTECEDENTES PERSONALES Y FAMILIARES		MARCAR "X" EN LOS ANTECEDENTES PERSONALES O FAMILIARES SEÑALADOS Y DESCRIBIR LOS DETALLES RESUMIDOS DEL ANTECEDENTE."
	REVERSO: EMERGENCIA (2)		

6	SIGNOS VITALES, MEDICIONES Y VALORES	**SIGNOS VITALES Y MEDICIONES**	REGISTRAR LOS DATOS RECOPILADOS DE PRESIÓN ARTERIAL, FRECUENCIA CARDIACA, FRECUENCIA RESPIRATORIA, TEMPERATURA, PESO, TALLA,
		VALORES	REGISTRAR LOS VALORES CALCULADOS DE LA ESCALA DE GLASGOW REGISTRAR LOS VALORES REACCIÓN PUPILAR DERECHA E IZQUIERDA, TIEMPO DE LLENADO CAPILAR Y SATURACIÓN DE OXÍGENO (SI SE DISPONE)
7	EXAMEN FÍSICO		MARCAR "SP" SI SE ENCUENTRA SIN PATOLOGIA, MARCAR "CP" SI SE ENCUENTRA CON PATOLOGIA REGISTRAR ABAJO EL NÚMERO Y LOS HALLAZGOS PATOLOGICOS
8	LOCALIZACION DE LESIONES		ESCRIBIR EN EL DIAGRAMA EL NÚMERO DE LA LESION CORRESPONDIENTE A LA REGION AFECTADA REGISTRAR INFORMACIÓN ADICIONAL SOBRE EL DIAGRAMA TOPOGRAFICO, SI ES NECESARIA UNA ACLARACIÓN
9	EMERGENCIA OBSTÉTRICA		REGISTAR LAS CARACTERÍSTICAS DE LA EMERGENCIA OBSTÉTRICA REALIZAR OBSERVACIONES ADICIONALES, SI AMERITA
10	SOLICITUD DE EXÁMENES		MARCAR "X" EN LAS CELDAS DE LOS EXÁMENES COMPLEMENTARIOS NECESARIOS ANOTAR UNA DESCRIPCION EN LA PARTE INFERIOR DEL BLOQUE.
11	DIAGNÓSTICOS DE INGRESO		REGISTRAR EL NOMBRE DEL **DIAGNÓSTICO DE INGRESO SI ES PRESUNTIVO O DEFINITIVO** ESCRIBIR LA CODIFICACIÓN DEL DIAGNÓSTICO SEGÚN LA CLASIFICACIÓN INTERNACIONAL DE ENFERMEDADES "CIE"
12	DIAGNÓSTICOS DE ALTA		REGISTRAR EL NOMBRE DEL **DIAGNÓSTICO DE INGRESO SI ES PRESUNTIVO O DEFINITIVO** ESCRIBIR LA CODIFICACIÓN DEL DIAGNÓSTICO SEGÚN LA CLASIFICACIÓN INTERNACIONAL DE ENFERMEDADES "CIE"
13	PLAN DE TRATAMIENTO	**INDICACIONES**	DESCRIBIR LAS INDICACIONES GENERALES PARA EL INICIO DEL TRATAMIENTO O LOS PROCEDIMIENTOS NECESARIOS
		MEDICAMENTOS	REGISTRAR EL NOMBRE GENÉRICO DEL PRINCIPIO ACTYIVO CON LA CONCENTRACIÓN Y PRESENTACIÓN RESPECIVA REGISTRAR LA POSOLOGÍA Y OTRAS INDICACIONES CORRESPONDIENTES.
14	ALTA		MARCAR "X" O SEGÚN LA OPCION CORRESPONDIENTE. REGISTRAR DATOS DEL DESTINO DEL PACIENTE Y SU CONDICIÓN AL SALIR, INDICANDO TAMBIÉN LA CAUSA DE SU ALTA O SALIDA (EJEMPLO: "TRATAMIENTO TERMINADO, ABANDONO VOLUNTARIO CON O SIN AUTORIZACIÓN MEDICA, ETC.)
FECHA	**HORA**	**NOMBRE DEL PROFESIONAL**	**CODIGO** **NUMERO DE HOJA**

INSTITUCIÓN DEL SISTEMA	UNIDAD OPERATIVA	COD. UO	COD. LOCALIZACIÓN			NUMERO DE HISTORIA CLINICA
			PARROQUIA	CANTÓN	PROVINCIA	

1 REGISTRO DE ADMISIÓN

Anverso

APELLIDO PATERNO	APELLIDO MATERNO	PRIMER NOMBRE	SEGUNDO NOMBRE	Nº CÉDULA DE CIUDADANÍA

DIRECCIÓN DE RESIDENCIA HABITUAL	Nº - MANZANA Y CA	PROVINCIA	ZONA (U/R)	Nº TELÉFONO

FECHA NACIMIENTO	LUGAR	EDAD EN AÑOS CUMPLIDOS	SEXO H	SEXO M	ESTADO CIVIL SOL	CAS	DIV	VIU	U-L	INSTRUCCIÓN ULTIMO AÑO APROBADO

FECHA DE ADMISIÓN	OCUPACIÓN	EMPRESA DONDE TRABAJA	TIPO DE SEGURO DE SALUD	REFERIDO DE:

EN CASO NECESARIO AVISAR A:	PARENTESCO - AFINIDAD	DIRECCIÓN	Nº TELÉFONO

FORMA DE LLEGADA						FUENTE DE INFORMACIÓN	INSTITUCIÓN O PERSONA QUE ENTREGA AL PACIENTE	Nº TELÉFONO
AMBULATORIO		AMBULANCIA		OTRO TRANSPORTE				

2 INICIO DE ATENCIÓN Y MOTIVO

HORA		TRAUMA		CAUSA CLÍNICA		CAUSA G. OBSTÉTRICA		CAUSA QUIRÚRGICA		GRUPO SANGUÍNEO Y FACTOR Rh	
NOTIFICACION A LA POLICIA		OTRO MOTIVO									

3 ENFERMEDAD ACTUAL Y REVISIÓN DE SISTEMAS

DESCRIBIR: CRONOLOGÍA - LOCALIZACIÓN - CARACTERÍSTICAS - INTENSIDAD - FRECUENCIA - FACTORES AGRAVANTES

VÍA AÉREA LIBRE		VÍA AÉREA OBSTRUIDA		CONDICIÓN ESTABLE		CONDICIÓN INESTABLE	

4 ACCIDENTE, VIOLENCIA, INTOXICACIÓN, ENVENENAMIENTO O QUEMADURA

FECHA Y HORA DEL EVENTO		LUGAR DEL EVENTO		DIRECCIÓN DEL EVENTO		CUSTODIA POLICIAL	

ACCIDENTE DE TRANSITO		CAÍDA		QUEMADURA		MORDEDURA		AHOGAMIENTO		CUERPO EXTRAÑO		APLASTAMIENTO		OTRO ACCIDENTE	
VIOLENCIA X ARMA DE FUEGO		VIOLENCIA X ARMA C. PUNZANTE		VIOLENCIA X RIÑA		VIOLENCIA FAMILIAR		ABUSO FÍSICO		ABUSO PSICOLÓGICO		ABUSO SEXUAL		OTRA VIOLENCIA	
INTOXICACIÓN ALCOHÓLICA		INTOXICACIÓN ALIMENTARIA		INTOXICACIÓN X DROGAS		INHALACIÓN DE GASES		OTRA INTOXICACIÓN		ENVENENAMIENTO		PICADURA		ANAFILAXIA	

OBSERVACIONES			
	ALIENTO ETÍLICO		VALOR ALCOCHECK

5 ANTECEDENTES PERSONALES Y FAMILIARES

DESCRIBIR ABAJO, REGISTRANDO EL NÚMERO RESPECTIVO

1. ALÉRGICO		2. CLÍNICO		3. GINECOLÓGICO		4. TRAUMATÓG.		5. QUIRÚRGICO		6. FARMACOLÓG.		7. OTRO ANTECEDENTE		8. NINGUN ANTECEDENTE	

6 SIGNOS VITALES, MEDICIONES Y VALORES

PRESIÓN ARTERIAL		F. CARDIACA min		F. RESPIRAT. min		TEMP. BUCAL °C		TEMP. AXILAR °C		PESO Kg		TALLA m	
GLASGOW	OCULAR (4)		VERBAL (5)		MOTORA (6)		TOTAL (15)		REACCIÓN PUPILA DER		REACCIÓN PUPILA IZQ		T. LLENADO CAPILAR		SATURA. OXIGENO	

7 EXAMEN FISICO Y DIAGNOSTICO

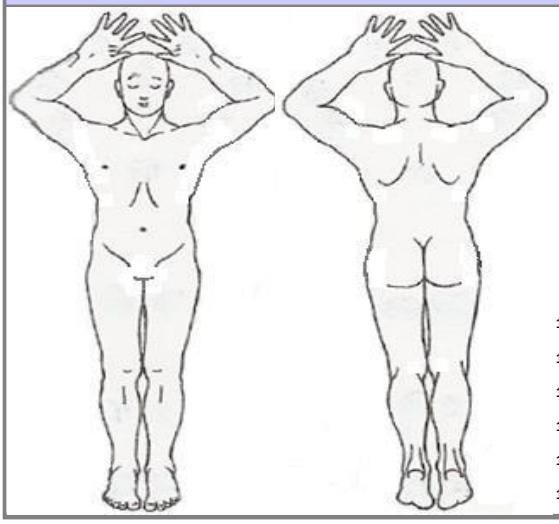
MARCAR "SP" SI SE ENCUENTRA SIN PATOLOGIA, MARCAR "**CP**" SI SE ENCUENTRA **C**ON **P**ATOLOGIA
REGISTRAR ABAJO EL NÚMERO Y LOS HALLAZGOS PATOLOGICOS

1. VIA AEREA OBSTRUIDA		3. CABEZA		3. CUELLO		4. TORAX		5. ABDOMEN		6. COLUMNA		7. PELVIS		8. EXTREMIDADE	

Reverso

8 LOCALIZACION DE LESIONES

ESCRIBIR EL NUMERO DE LA LESIÓN SOBRE LA REGIÓN CORRESPONDIENTE



1	HERIDA PENETRANTE	
2	HERIDA NO PENETRANTE	
3	FRACTURA EXPUESTA	
4	FRACTURA CERRADA	
5	CUERPO EXTRAÑO	
6	HEMORRAGIA	
7	MORDEDURA	
8	MUTILACION	
9	EXCORIACIÓN	
10	DEFORMIDAD	
11	HEMATOMA	
12	ERITEMA	
13	LUXACION / ESGUINCE	
14	QUEMADURA	
15	APLASTAMIENTO	

9 EMERGENCIA OBSTÉTRICA

GESTAS		PARTOS		ABORTOS		CESÁREAS	
FECHA ULTIMA MENSTRUACIÓN				SEMANAS GESTACIÓN		MOVIMIENTO FETAL	
FRECUENCIA C. FETAL		MEMBRANAS ROTAS		TIEMPO DE RUPTURA			
ALTURA UTERINA		PRESENTACIÓN					
DILATACIÓN		BORRAMIENTO		PLANO			
PELVIS ÚTIL		SANGRADO VAGINAL		CONTRACCIONES			

10 SOLICITUD DE EXÁMENES

REGISTRAR ABAJO COMENTARIOS Y RESULTADOS, ANOTANDO EL NUMERO

1. BIOMETRÍA		3. QUÍMICA SANGUÍNEA		5. GASOMETRÍA		7. ENDOSCOPIA		9. R-X ABDOMEN		11. TOMOGRAFÍA		13. ECOGRAFÍA PÉLVICA		15. INTERCONSULTA	
2. UROANALISIS		4. ELECTROLITOS		6. ELECTRO CARDIOGRAMA		8. R-X TÓRAX		10. R-X ÓSEA		12. RESONANCIA		14. ECOGRAFÍA ABDOMEN		16. OTROS	

11 DIAGNÓSTICO DE INGRESO

PRE= PRESUNTIVO
DEF= DEFINITIVO

		CIE	PRE	DEF
1				
2				
3				

12 DIAGNÓSTICO DE ALTA

PRE= PRESUNTIVO
DEF= DEFINITIVO

		CIE	PRE	DEF
1				
2				
3				

13 PLAN DE TRATAMIENTO

INDICACIONES		MEDICAMENTO PRINCIPIO ACTIVO, CONCENTRACIÓN Y PRESENTACIÓN	POSOLOGÍA
	1		
	2		
	3		
	4		

14 ALTA

DOMICILIO		CONSULTA EXTERNA		OBSERVACIÓN		INTERNACIÓN		REFERENCIA		EGRESA VIVO		EN CONDICIÓN ESTABLE		EN CONDICIÓN INESTABLE		DÍAS DE INCAPACIDAD	
SERVICIO DE REFERENCIA			ESTABLECIMIENTO							MUERTO EN EMERGENCIA		CAUSA PROBABLE					

CÓDIGO

FECHA		HORA		NOMBRE DEL PROFESIONAL			FIRMA		NUMERO DE HOJA	

INSTITUCIÓN DEL SISTEMA	UNIDAD OPERATIVA	CÓDIGO	LOCALIZACIÓN			HISTORIA CLÍNICA
			PARROQUIA	CANTÓN	PROVINCIA	
Universidad Técnica del Norte	Carrera de Medicina/ Campus San Vicente de Paúl	00001	SAN FRAN	IBARRA	IMBABUR	1003858352

1 REGISTRO DE ADMISION

APELLIDO PATERNO	APELLIDO MATERNO	NOMBRES	NACIONALIDAD	CÉDULA DE CIUDADANÍA
NOGALES	ROMERO	JOSE CAMILO	ECUATORIANA	1003858352

DIRECCION DE RESIDENCIA HABITUAL	CANTON	PROVINCIA	N° TELÉFONO
CHORLAVI, LOS CEREZOS A9 Y AV. MONS. LEONIDAS PROAÑO	IBARRA	IMBABURA	0939992678

FECHA DE ATENCIÓN	HORA	EDAD	GÉNERO		ESTADO CIVIL					INSTRUCCIÓN					OCUPACION	GRUPO SANGUINEO	
			M	F	S	C	D	V	UL	SIN	BAS	BACH	SUP	ESP			
		29	X		X								X		ESTUDIANTE	A+	

NOMBRE DE LA PERSONA PARA NOTIFICACION	PARENTESO O AFINIDAD	DIRECCION	N° TELEFONO
SONIA DEL CARMEN ROMERO ARMAS	MADRE	CHORLAVI, LOS CEREZOS A9 Y AV. MONS. LEONIDAS PROAÑO	0994386650
NOMBRE DEL ACOMPAÑANTE	N° CÉDULA DE IDENTIDAD	DIRECCIÓN	N° TELEFONO
FORMA DE LLEGADA	FUENTE DE INFORMACIÓN	INSTITUCIÓN O PERSONA QUE ENTREGA AL PACIENTE	N° TELEFONO
AMBULATORIO / SILLA DE RUEDAS / CAMILLA			

MAS= MASCULINO FEM= FEMENINO SOL= SOLTERO CAS= CASADO DIV= DIVORCIADO VIU= VIUDO UL= UNION LIBRE SIN= SIN INSTRUCCION BAS= BASICA BACH= BACHILLERATO SUPE= SUPERIOR ESPE= ESPECIAL

2 INICIO DE ATENCIÓN

HORA		VIA AEREA LIBRE		VIA AEREA OBSTRUIDA		GRUPO - Rh		CONDICIONES DE LLEGADA	ESTABLE		INESTABLE		OTRO	
MOTIVO DE LLEGADA														

3 ACCIDENTE, VIOLENCIA, INTOXICACION

NO APLICA

LUGAR DEL EVENTO	DIRECCIÓN DEL EVENTO	FECHA	HORA	VEHÍCULO O ARMA

TIPO DE EVENTO						AUTORIDAD COMPETENTE		
ACCIDENTE	ENVENENAMIENTO	VIOLENCIA	OTRO			HORA DENUNCIA		CUSTODIA POLICIAL
OBSERVACIONES								

INTOXICACIÓN								VIOLENCIA							
ALIENTO ETICO		VALOR ALCOCHECK		HORA EXAMEN		SE HACE ALCOHOLEMIA		OTRAS SUSTANCIAS		SOSPECHA		ABUSO FISICO		ABUSO PSICOLOGICO		ABUSO SEXUAL	
OBSERVACIONES															

QUEMADURA								PICADURA	MORDEDURA
GRADO I		GRADO II		GRADO III		PORCENTAJE SUPERFICIE			

4 ANTECEDENTES PERSONALES Y FAMILIARES RELEVANTES

PARA DESCRIBIR SEÑALE EL NUMERO Y LA LETRA CORRESPONDIENTE. P=Personal F=Familiar

NO APLICA

1. ALERGICOS		2. CLINICOS		3. GINECOLOGICOS		4. TRAUMATOLOGICOS		5. PEDIATRICOS		6.QUIRURGICOS		7. FARMACOLOGICOS		8. OTROS	

Alergias: No Refiere (APP)

Clínicos: Presión alta diagnosticada hace 3 años sin tratamiento (APP)

Clínicos: Presión alta (Padre), Cáncer de Tiroides (Madre), Leucemia (Abuelo Materno), Diabetes Mellitus (Abuela Paterna), Alzheimer (Abuela Materna) [APF]

5 ENFERMEDAD ACTUAL Y REVISIÓN DE SISTEMAS

CRONOLOGIA, LOCALIZACIÓN, CARACTERÍSTICAS, INTENSIDAD, FRECUENCIA, FACTORES AGRAVANTES.

NO APLICA

6 CARACTERISTICAS DEL DOLOR

NO APLICA

REGIÓN ANATÓMICA	PUNTO DOLOROSO	EVOLUCIÓN			TIPO			MODIFICACIONES					ALIVIA CON				INTENSIDAD LEVE MODERADA O GRAVE
		AGUDO	SUBAGUDO	CRONICO	EPISODICO	CONTINUO	COLICO	POSICION	INGESTA	ESFUERZO	DEFECACION	SE IRRADIA	ANTIESPASMODICO	OPIACEO	AINE	NO ALIVIA	

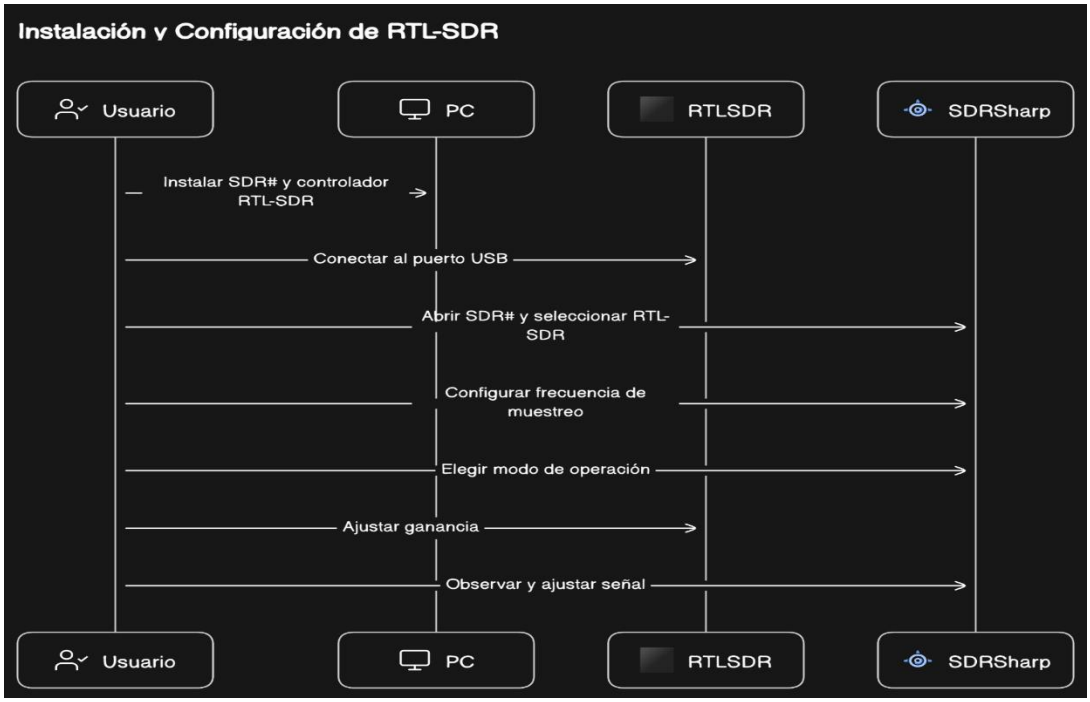
FORMULARIOS MSP – SEMIOLOGÍA UTN – Formulario 008: EMERGENCIA

Videos Procedimiento Práctico Chip Implante NFC

ANEXO PRUEBAS RADIOFRECUENCIA:

Instalación y Uso del AirSpy para NFC:

En la investigación, se explora la aplicación de tecnologías Near Field Communication (NFC) en conjunto con receptores de radio definidos por software (SDR) como SDR Sharp y RTL-SDR. Utilizando SDR Sharp con hardware AirSpy o RTL-SDR USB Dongle, se siguen pasos específicos para extraer señales NFC a 13.56 MHz. La configuración del receptor incluye la selección del dispositivo, ajuste de la frecuencia central y muestreo, elección del modo de operación, ajuste de ganancia y visualización de la señal en la interfaz gráfica. Posteriormente, se graban las señales IQ para su posterior procesamiento en el entorno de MATLAB. Este enfoque permite analizar y comprender las señales NFC, demostrando la versatilidad de los receptores SDR en la exploración y manipulación de comunicaciones NFC, abriendo posibilidades para aplicaciones de investigación y desarrollo.



Códigos de MatLab:

```
close all;
clear all;
clc;
```

Función de limpieza de ventanas, cierra todas las ventanas.
Función de limpieza de variables, eliminar todas las variables del workspace.
Función de limpieza de ventana de comandos.

```
[data,fs] = audioread('SDRSharp_20231211_163919Z_13560000Hz_IQ.wav');
t = 1:1:size(data(:,1)); % Vector tiempo en segundos a partir del
N=length(data); % tamaño del vector data
% Separar Canales IQ
q_sample = data(:,1) ;
i_sample = data(:,2) ;
IQ_samples = i_sample + 1i.*q_sample ; % señal I+jQ
pspectrum(IQ_samples,fs,'FrequencyLimits',[-13560000 13560000])
```

Función audioread para la lectura del archivo .wav generado por AirSpy

El vector data tiene un tamaño de 5106845x2, lo que significa que tiene 2 columnas, la primera columna pertenecen a los valores Q y la segunda a I.

Función pspectrum permite graficar el espectro de frecuencia. Los límites de frecuencia están dados por la fc de trabajo de NFC tanto en positivo como en negativo

```
%% Dibujar en el dominio del tiempo
figure;
%subplot(4,1,1)
plot(t,i_sample,Color='r')
title('Plot in the time domain data')
xlabel('Time(usec)')
ylabel('Amplitude')
legend('Time Domain Signal')


%% Densidad Espectral
fftsize=1024; % Tamaño de la Transformada de Fourier
nfft = floor(length(IQ_samples)/fftsize) ; % Coeficientes de la FFT
taxis = [0:fftsize/fs:(nfft-1)*fftsize/fs];
rf=13560000; % Frecuencia NFC
faxis = [rf-fs/2:fs/fftsize:rf-fs/2+(fftsize-1)*fs/fftsize];
```

Obtención de lo coeficientes de Fourier con la aproximación al inmediato inferior

Tamaño de los ejes de tiempos desde 0 hasta el número de muestras de la transformada de Fourier

Tamaño de los ejes de fq desde la fq de NFC – la Fs/2 hasta la fq de NFC mas el tamaño de fft.

```
[Pxx, F] = pspectrum(IQ_samples, fs, 'spectrogram'); % Grafica Espectrograma
figure; plot(faxis,10*log10(abs(Pxx)),'b')
xlabel('frequency(Hz)')
ylabel('PSD (W/HZ)')
%% IQ plot
figure
%subplot(4,1,4)
plot(real(IQ_samples(290000:293000)),"b");
hold on
plot(imag(IQ_samples(290000:293000)),"g");
legend("Inphase signal", "Quadrature signal");
title("IQ Data for 3000 points of acquired signal")
xlabel("Sample number");
ylabel("Voltage");
```

Códigos de MatLab para el cálculo de SNR teórico y práctico:

En el contexto del presente trabajo de titulación y el análisis en el ámbito de NFC, se desarrolla un código en MATLAB para llevar a cabo la evaluación de la Relación Señal a Ruido (SNR) de señales NFC. El código constituye una herramienta técnica para medir la calidad de la transmisión en sistemas de comunicación inalámbrica NFC. Para acceder al código y obtener una visión detallada de la metodología implementada, se invita a explorar el siguiente enlace de GitHub: https://github.com/CamiloNogalesR/calculossnrnfc/blob/main/CalculoSDR.m.

Este recurso ofrece una contribución significativa al entendimiento y optimización de la calidad de las señales NFC, facilitando la replicabilidad y revisión de los métodos utilizados.

Explicación de los códigos:

El código de MATLAB tiene como objetivo realizar el procesamiento y análisis de señales IQ provenientes de un archivo de audio en formato WAV. A continuación, se presenta una explicación detallada de las distintas secciones del código:

En primer lugar, se lleva a cabo la inicialización del entorno de trabajo mediante el cierre de gráficos previos, la eliminación de variables existentes y la limpieza de la ventana de comandos.

La segunda línea del código emplea la función audioread para leer un archivo de audio WAV. Las componentes en cuadratura (Q) e en fase (I) de la señal se extraen y se combinan para formar una señal compleja denominada IQ_samples.

Las señales IQ, también denominadas In-Phase (I) y Quadrature (Q), constituyen un componente esencial en el ámbito del procesamiento de señales y la modulación. Estas señales, representadas por dos componentes, I (en fase con la señal sinusoidal de

referencia) y Q (en cuadratura desplazada 90° en fase a la componente I), son fundamentales para la eficiente transmisión de información en sistemas de comunicación. En particular, se emplean comúnmente en técnicas de modulación, como la modulación QAM. La combinación de estas dos componentes forma una señal compleja, a menudo representada en forma rectangular como I+jQ, donde j es la unidad imaginaria. Esta representación en forma compleja permite describir completamente la señal, incluyendo su amplitud, frecuencia y fase.

La representación en el dominio del tiempo de la componente en fase (I) se visualiza en un gráfico para proporcionar una perspectiva temporal de la señal. Posteriormente, se calcula y muestra la densidad espectral de potencia de la señal IQ utilizando la función pspectrum.

El código también presenta un gráfico en el dominio del tiempo que representa las partes en fase (I) y en cuadratura (Q) de la señal IQ. Este gráfico ofrece una representación visual de la información contenida en la señal.

Flujograma del código: