FACULTAD 1

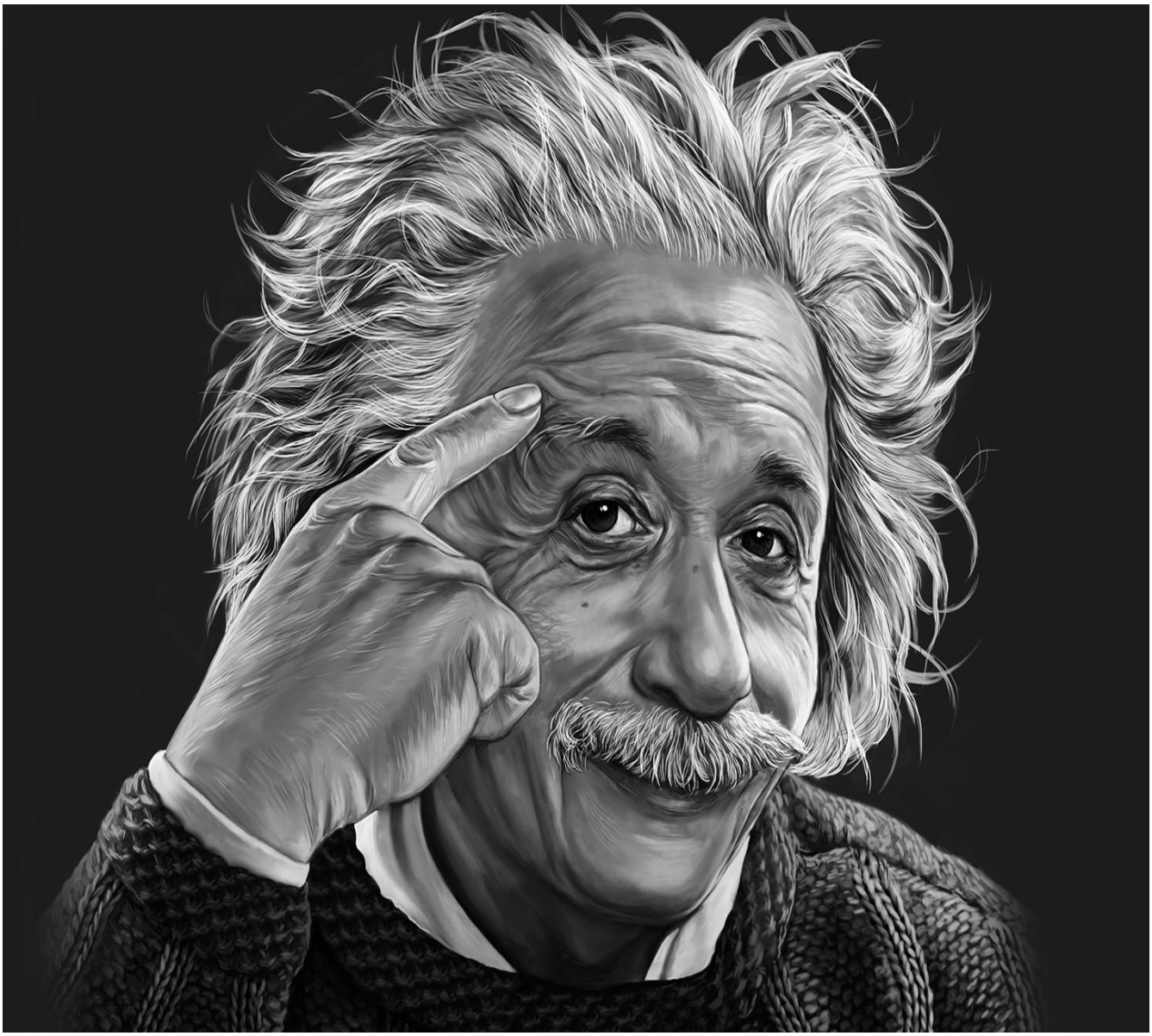
Aplicación móvil del sitio oficial de la Presidencia y Gobierno de Cuba.

Trabajo de diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Autor: Enma Nélida Piquero Moskaquiel
Tutor(es): MSc. Niurvis Legrá Pérez
MSc. Omara Aldama López
Esp. Ulises Araujo Cohen

La Habana, julio de 2021

"Año 63 de la Revolución"



“Siento una enorme gratitud por los que me dijeron “NO”. Gracias a ellos, lo hice yo mismo.”

Albert Einstein

DECLARACIÓN DE AUTORÍA

El autor del trabajo de diploma con título ***"Aplicación móvil del sitio oficial de la Presidencia y Gobierno de Cuba"***, concede a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la investigación, con carácter exclusivo. De forma similar se declara como único autor de su contenido. Para que así conste firma la presente a los <día> días del mes de <mes> del año 2021.

Enma Nélida Piquero Moskaquiel

_____
Firma del Autor

MSc. Niurvis Legrá Pérez

_____
Firma del Tutor

MSc. Omara Aldama López

_____
Firma del Tutor

Esp. Ulises Araujo Cohen

_____
Firma del Tutor

DATOS DE CONTACTO

Niurvis Legrá Pérez: Graduada de Lic. en Ciencias de la Computación en 2002. Posee categoría docente de Profesor Auxiliar y es Máster en Ingeniería de Software e Inteligencia Artificial. Ha impartido asignaturas de las disciplinas Práctica Profesional, Ingeniería de Software y Gestión Organizacional. Actualmente se desempeña como Decana de la Facultad 1. Correo electrónico: niurvis@uci.cu

Omara Aldama López: Graduada de Licenciada en Cibernética Matemática en la Universidad de La Habana en el año 1982. MSc en Ciencias de la Computación (2004). Directora de Infocomunicaciones, Palacio de la Revolución. Correo electrónico: omara@palacio.cu

Ulises Araujo Cohen: Ingeniero en Ciencias Informáticas, graduado en la UCI en el año 2012. Se desempeña como Especialista A en la Dirección de Infocomunicaciones de la UPSIPR desde el 2014, específicamente en el área de Software. Participa en el desarrollo de aplicaciones propias. Correo electrónico: uaraujo@palacio.cu

AGRADECIMIENTOS

A todos los presentes, por dedicarme su tiempo.

A todos los profesores que tuvieron el gusto de conocerme, que me enseñaron tanto de la profesión como de la vida, impulsándome siempre a seguir adelante. Gracias por sus lecciones, por soportarme, orientarme y su enorme paciencia hacia mí.

Gracias a Jorge, enciclopedia ambulante, por ayudarme en todo momento y escuchar mis disímiles problemas existenciales sobre la tesis, por su apoyo y amistad, que me permitieron aprender más.

A todo el grupo de trabajadores de palacio, que estuvieron ahí prestándome sus cuentas, dándome ánimo y, sobre todo, molestando.

A mis excelentes tutores mis respetos, porque a pesar de ser personas sumamente ocupadas, sacaron espacio para mí, gracias.

Mi grupo querido, años a su lado, peleando, comiendo, en las fiestas, todos ustedes han propiciado un enorme cambio, para bien, que han hecho la persona de que hoy soy.

A mi Hulk, por apoyarme en todo momento, entenderme, o intentarlo. Gracias por todo tu cariño, atenciones y las horas dedicadas a mi tesis, te amo.

Bori, no me olvido de ti, domador de leones, todo este tiempo me has demostrado lo buena persona que eres, te has preocupado mucho por cómo iba esa tesis, no tengo palabras para agradecerte, si no te lo había dicho antes, te has ganado mi cariño.

Mi querida UCI, gracias por todo lo que han aportado a mi formación, por fortalecer mis ideales, permitir mi desarrollo como futura profesional, pero más que nada por poner en mi camino a personas tan geniales, que espero perduren con el paso del tiempo.

DEDICATORIA

Dedico mi tesis, mi título, y todos los logros que me faltan por conseguir, a las dos mujeres más importantes de mi vida: mi madre y mi abuela que desde donde esté, nos está guiando y cuidando.

Pero más que nada, a mi madre.

Ella que ha estado en todos los momentos bellos y malos de mi vida.

Esa mujer que ha insistido por la necesidad de continuar mis estudios, que ha hecho lo imposible para conseguir este logro.

Amor de mi vida, mi insoportable, esta tesis, este logro es más tuyo que mío, sin ti no fuera posible y solo tú y yo sabemos los por qué.

Este es el primer granito de arena del castillo que pienso construirte.

RESUMEN

La Presidencia y Gobierno de Cuba son los principales pilares operativos del Estado, los cuales organizan y orientan a la sociedad. Para ello cuentan con un sitio web que facilita información sobre los aconteceres a nivel nacional e internacional, además de permitir la obtención de documentos regentes del país y conocer los contactos directos con las entidades gubernamentales que guían los procesos de la nación. En la presente investigación, se desarrolla una aplicación para dispositivos móviles con el objetivo de mejorar el acceso a los servicios y la información del sitio oficial de la Presidencia y Gobierno de Cuba. La aplicación se implementa para el sistema operativo móvil Android, este permite brindar y consumir los servicios REST (por sus siglas en inglés Representational State Transfer) del sitio web. El proceso de desarrollo es guiado por la metodología de desarrollo XP y como IDE de programación se utiliza Android Studio y como lenguaje de programación, Kotlin. Al concluir el desarrollo se somete la aplicación a pruebas para garantizar que brinde una solución idónea y segura, y a su vez, mantenga un funcionamiento estable.

PALABRAS CLAVES: aplicación móvil (apk), sistema operativo móvil Android, dispositivos móviles

ABSTRACT

The Presidency and Government of Cuba are the main operational pillars of the State, which organize and guide society. For this purpose, they have a website that provides information on national and international events, in addition to allowing the obtaining of documents governing the country and direct contacts with government entities that guide the processes of the nation. In this research, an application for mobile devices is developed with the objective of improving access to the services and information of the official site of the Presidency and Government of Cuba. The application is implemented for the Android mobile operating system, which allows providing and consuming the REST (Representational State Transfer) services of the website. The development process is guided by the XP development methodology and Android Studio is used as pro-gramming IDE and Kotlin as programming language. At the end of the development, the application is tested to ensure that it provides a suitable and secure solution, while maintaining stable operation.

KEYWORDS: *Android mobile operating system, mobile application (apk), mobile devices*

TABLA DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

OPINIÓN DEL(OS) TUTOR(ES)

<Contenido de la opinión de los tutores>

AVAL DEL CLIENTE

<Contenido del aval del cliente sobre la solución desarrollada>

INTRODUCCIÓN

Las tecnologías de la Información y las Comunicaciones (TIC) han cobrado un gran auge en lo que se ha denominado la era digital. La evolución del uso de las TIC en la administración pública implica revolucionar los sistemas de almacenamiento y empleo de información. Lo anterior sugiere según (Ávila Barrios. 2014)que puede fomentar la transparencia en la función pública y que se diversificarán los medios de comunicación y de contacto entre la ciudadanía y los diferentes niveles de servidores públicos.

A partir de estos cambios, la actividad de gobernar puede calificarse como intensiva en conocimiento e información, ya que la administración del sector público, la provisión de servicios de gobierno y la realización de políticas públicas se auxilian del manejo de información, bases de datos y sistemas para su eficiente desempeño.

En (Llodrà, B. 2009) se destaca que la adopción del uso de las TIC en la administración pública implica, entre otras, como mínimo: dotar de infraestructuras tecnológicas y de acceso a éstas, tanto a la administración como a la ciudadanía; organizar toda la información disponible; establecer canales de comunicación con formas de participación activa entre la administración y la ciudadanía.

En este sentido Cuba ha identificado las potencialidades que esta representa para el correcto avance de la sociedad en todas las esferas que esta abarca, ya que como refiere (Jover 2016). la tecnología más que como un resultado, único e inexorable, debe ser vista como un proceso social, una práctica, que integra factores psicológicos, sociales, económicos, políticos, culturales; siempre influido por valores e intereses.

Alineados en este camino, la Unidad Presupuestada de Servicios Internos (UPSI) del Palacio de la Revolución ha instado a la Dirección de Infocomunicaciones de la entidad, a crear proyectos que contribuyan a la soberanía del país, dando así, cumplimiento a la Política Integral para el perfeccionamiento de la informatización en Cuba (Política Integral, 2017), la cual se ha concebido como parte del Modelo Económico y Social del país.

Como parte de la estrategia de Informatización de la sociedad cubana, esta Dirección crea en el 2018 el portal web: ¨Presidencia y Gobierno de Cuba¨, cumpliendo de esta forma con la etapa de presencia de las instituciones de Gobierno en las redes. Posteriormente, con el objetivo de diversificar las vías de acceso a esta plataforma, fue desarrollada una aplicación para los dispositivos con sistema operativo Android. Sin embargo, la misma sólo muestra de manera íntegra la versión *responsive design* (diseño web adaptable) del sitio web y no realiza ninguna

gestión diferenciada de la información y las secciones, ni las jerarquiza como realizan habitualmente soluciones de este tipo. Este modelo de trabajo, supone un consumo de datos similar al realizado a través de navegadores web, pues no almacena los contenidos estáticos o secciones permanentes. Tampoco adapta el diseño original a la tendencia de este tipo de herramientas para que cumpla con las necesidades técnicas y comunicacionales propias de este tipo de aplicación.

A partir de la situación antes descrita, se puede definir el siguiente **problema de investigación:** ¿Cómo mejorar el acceso a los servicios y la información del sitio de la Presidencia y el Gobierno de Cuba?

Determinando como **objeto de estudio:** las aplicaciones para dispositivos móviles que hagan uso de las API (Application Programming Interface), estableciendo como **campo de acción:** aplicaciones gubernamentales para dispositivos móviles que utilicen API.

Para resolver el problema planteado, se propone como **objetivo general:** Desarrollar una aplicación para dispositivos móviles que permita mejorar el acceso a los servicios y la información del sitio oficial de la Presidencia y el Gobierno de Cuba.

A partir del objetivo general, se pueden definir los siguientes **objetivos específicos:**

- Investigar los referentes teóricos que permitan sustentar el desarrollo de la investigación.
- Seleccionar la metodología, tecnologías y las herramientas informáticas para la creación de la aplicación móvil que ayude al ciudadano el acceso a los servicios y la información del sitio de la Presidencia y el Gobierno de Cuba.
- Identificar los requisitos funcionales y no funcionales de la aplicación móvil que permita al ciudadano el acceso a los servicios y la información del sitio de la Presidencia y el Gobierno de Cuba.
- Diseñar una aplicación para dispositivos móviles que permita al ciudadano el acceso a los servicios y la información del sitio de la Presidencia y el Gobierno de Cuba.
- Implementar las funcionalidades de la aplicación móvil.
- Validar la aplicación móvil a partir de una secuencia de pruebas y estándares de softwares.
- Cargar la aplicación móvil terminada a la tienda cubana APKlis.

Se plantea como **idea a defender** que con el desarrollo de una aplicación para dispositivos móviles mejorará el acceso a los servicios y la información del sitio oficial de la Presidencia y el Gobierno de Cuba.

Con el desarrollo de la investigación, los posibles resultados están centrados en realizar un estudio bibliográfico sobre aplicaciones móviles con acceso a portales web para la información de Gobierno, así como desarrollar una API y una aplicación móvil del Sitio Oficial de la Presidencia y Gobierno de Cuba.

Los **métodos de investigación** utilizados para dar solución a la presente investigación son:
Teóricos:

- **Analítico-sintético:** En el análisis bibliográfico para extraer los elementos esenciales asociados a la problemática planteada y las tecnologías adecuadas para el desarrollo de aplicaciones móviles.
- **Histórico - Lógico:** Para el análisis en las diferentes fuentes bibliográficas sobre desarrollos y avances de las aplicaciones móviles gubernamentales que presentan una API en su confección.
- **Modelación:** En la representación mediante diagramas de las características, procesos y componentes de la aplicación propuesta, así como la relación existente entre ellos.
- **Sistémico:** Para concebir la unión racional de conceptos fundamentales como API, *responsive design* y aplicación móvil, inicialmente elementos dispersos, en una nueva totalidad, así como conocer los rasgos fundamentales de la aplicación móvil.

Empíricos:

- **Observación:** Para obtener el conocimiento y la información para el correcto análisis y diseño de la aplicación móvil.
- **Entrevista:** En encuentros con el cliente para conocer la información necesaria acerca de la problemática, necesidad del desarrollo de la propuesta de solución y definir sus funcionalidades.

La presente investigación se ha estructurado en tres capítulos, conformados de la siguiente forma:

Capítulo 1: Fundamentación teórica: En este capítulo se exponen los conceptos asociados a la solución de la problemática y varias de las soluciones existentes a nivel mundial sobre aplicaciones móviles gubernamentales que utilizan API. Además, se analizan las herramientas, lenguajes y tecnologías a tener en cuenta para el proceso de desarrollo de soluciones basadas en aplicaciones móviles.

Capítulo 2: Análisis y diseño de la propuesta de solución: En este capítulo se documenta todo el proceso de elaboración de la aplicación de manera detallada y se desglosan las características funcionales que debe cumplir el sistema de acuerdo a la metodología de desarrollo escogida. Se describen los patrones utilizados en la solución, así como las características propias del sistema a desarrollar en cuanto a medios y herramientas utilizados.

Capítulo 3. Implementación y pruebas de validación: En este capítulo se detalla la distribución física de los distintos componentes lógicos desarrollados, a través del modelo de despliegue y la organización del sistema mediante el modelo de componentes. Se explican los estilos de programación y estándares de codificación empleados y por último se describen las pruebas de software que demuestran que la aplicación funciona correctamente.

CAPÍTULO 1: FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS SOBRE EL OBJETO DE ESTUDIO

El presente capítulo tiene como objetivo abordar los diferentes elementos que brindan la base teórica y conceptual para el desarrollo de la aplicación móvil propuesta. Se analizan un grupo de herramientas y tecnologías que, desde el punto de vista práctico, pueden ser empleadas para lograr el propósito de la investigación. Se realiza, además, un análisis sobre los aspectos fundamentales del sistema operativo Android. De este modo, se podrá realizar una correcta interpretación de la situación problemática y del problema a resolver.

1.1 Conceptualización de la investigación

Con el objetivo de lograr una mayor comprensión del alcance de la investigación y esclarecer su objeto de estudio, a continuación, se relacionan los principales conceptos que ayudan a entender su desarrollo.

1.1.1 Dispositivos móviles

Tomando la definición de (Niño 2011; Garrido, Schlesinger y Hoganson 2011) un dispositivo móvil es un aparato de pequeño tamaño, con algunas capacidades de procesamiento, con conexión permanente o intermitente a una red, con memoria limitada, que ha sido diseñado específicamente para una función, pero que puede llevar a cabo otras funciones más generales. De acuerdo con esta definición existen multitud de dispositivos móviles, desde los reproductores de audio portátiles hasta los navegadores GPS, pasando por los teléfonos móviles, los PDAs o los Tablet PCs. Por lo general, las características esenciales de estos dispositivos son:

1. Son aparatos pequeños: la mayoría se pueden transportar en el bolsillo del propietario o en un pequeño bolso.
2. Tienen capacidad de procesamiento.
3. Tienen conexión permanente o intermitente a una red.
4. Tienen memoria (*RAM*, tarjetas *MicroSD*, *flash*, etc.).
5. Normalmente, se asocian al uso individual de una persona, tanto en posesión como en operación, la cual generalmente puede adaptarlos a su gusto.
6. Tienen una alta capacidad de interacción mediante la pantalla o el teclado.

1.1.2 Aplicaciones móviles

Una aplicación móvil también llamada app móvil o apk, es un tipo de software diseñado para ejecutarse en teléfonos inteligentes, tabletas y otros dispositivos móviles. Permite a los usuarios efectuar tareas de cualquier tipo: profesional, educativa, informativa, de ocio y/o servicios. (Luz 2018), califica una aplicación móvil como un programa adaptado a las características y especificaciones de los dispositivos móviles y que permite cubrir prácticamente cualquier necesidad de forma ubicua desde su descarga online.

Entre sus principales características se encuentran (L 2011; Gomis 2014):

- Adaptación a varios sistemas operativos.
- Diseño agradable y capacidades de interacción
- Analítica: Habilita la información necesaria para implementar estrategias de mejoramiento de la aplicación y para darle a entender a los usuarios que son importantes y que sus comentarios son escuchados.
- Actualizaciones: Permiten mejorar los errores, aquellas funcionalidades que quizá no están funcionando como deberían o que han recibido muchas critica parte de los usuarios.

Dentro del desarrollo de las aplicaciones móviles, existen diferentes clasificaciones. A continuación, la descripción de los tres tipos según (Guadalupe, 2015):

Aplicaciones nativas

Las aplicaciones nativas son aquellas desarrolladas bajo un lenguaje y entorno de desarrollo especifico, lo cual permite, que su funcionamiento sea muy fluido y estable para el sistema operativo que fue creada. La opción principal para desarrollar en Android es Java y Kotlin.

Aplicaciones web

Usadas para brindar accesibilidad a la información desde cualquier dispositivo, sin importar el sistema operativo, ya que solo se necesita contar con un navegador para acceder a esta. Son aquellas desarrolladas usando lenguajes para el desarrollo web como son: HTML, CSS y JavaScript y un marco de trabajo para el desarrollo de aplicaciones web, como por ejemplo Jquery mobile, Sencha, Kendo UI, entre otros.

Aplicaciones híbridas

Este tipo de aplicaciones, combinan aspectos de las aplicaciones nativas y de las aplicaciones web según más convenga. La facilidad que brinda este tipo de desarrollo es que no hay un entorno específico para utilizar en su desarrollo y la mayoría de las herramientas son de uso gratuito, también pudiendo integrarlo con las herramientas de aplicaciones nativas.

A la hora de desarrollar aplicaciones móviles existen muchas opciones a las cuales recurrir, dependiendo del tipo de información que se desea brindar y la forma en que se vaya a realizar. El uso de los recursos de los dispositivos móviles y su sistema operativo son datos fundamentales a tener cuenta. En esta investigación se ha decidido implementar una aplicación móvil de tipo nativa, a solicitud de las especificaciones de la UPSI, pues es un requisito indispensable la independencia tecnológica para conseguir una mayor autonomía, necesaria en las condiciones actuales en que se encuentra Cuba.

1.1.3 Sistemas operativos móviles

Un sistema operativo (SO) para móviles, es un conjunto de programas y órdenes capaces de ejecutarse sobre el hardware del dispositivo para satisfacer las necesidades del usuario. En los teléfonos móviles, estas suelen ser de conexión a la red, por lo que el SO móvil está muy orientado a la conectividad inalámbrica. Además, el SO es la parte del dispositivo que permite interactuar y darle órdenes al teléfono. Su principal función es administrar tareas y recursos del dispositivo, coordinar las actividades del software en base al hardware y organizar los archivos en directorios. Es, por tanto, un intermediario entre usuario y hardware.

En los últimos años, los dispositivos móviles han tenido un gran auge, esto conlleva a una amplia repercusión en casi todos los aspectos de la vida: relaciones personales y profesionales, ocio y consumo, y como no, las actividades económicas y empresariales. Los dispositivos en sí, y las tecnologías que los dotan de funcionalidades de todo tipo, evolucionan a vertiginosa velocidad, y generan en la sociedad nuevas tendencias y modelos de comportamiento que cada vez se hacen más importantes para todo tipo de empresas, pues pueden lograr o perder grandes oportunidades de negocio si pueden o no adaptarse a ellos. Dentro de los SO más utilizados en dispositivos móviles se encuentran Android, iOS y Windows Phone. A continuación, se muestra en la gráfica siguiente un estudio de la cuota de mercado del SO móvil en Cuba:

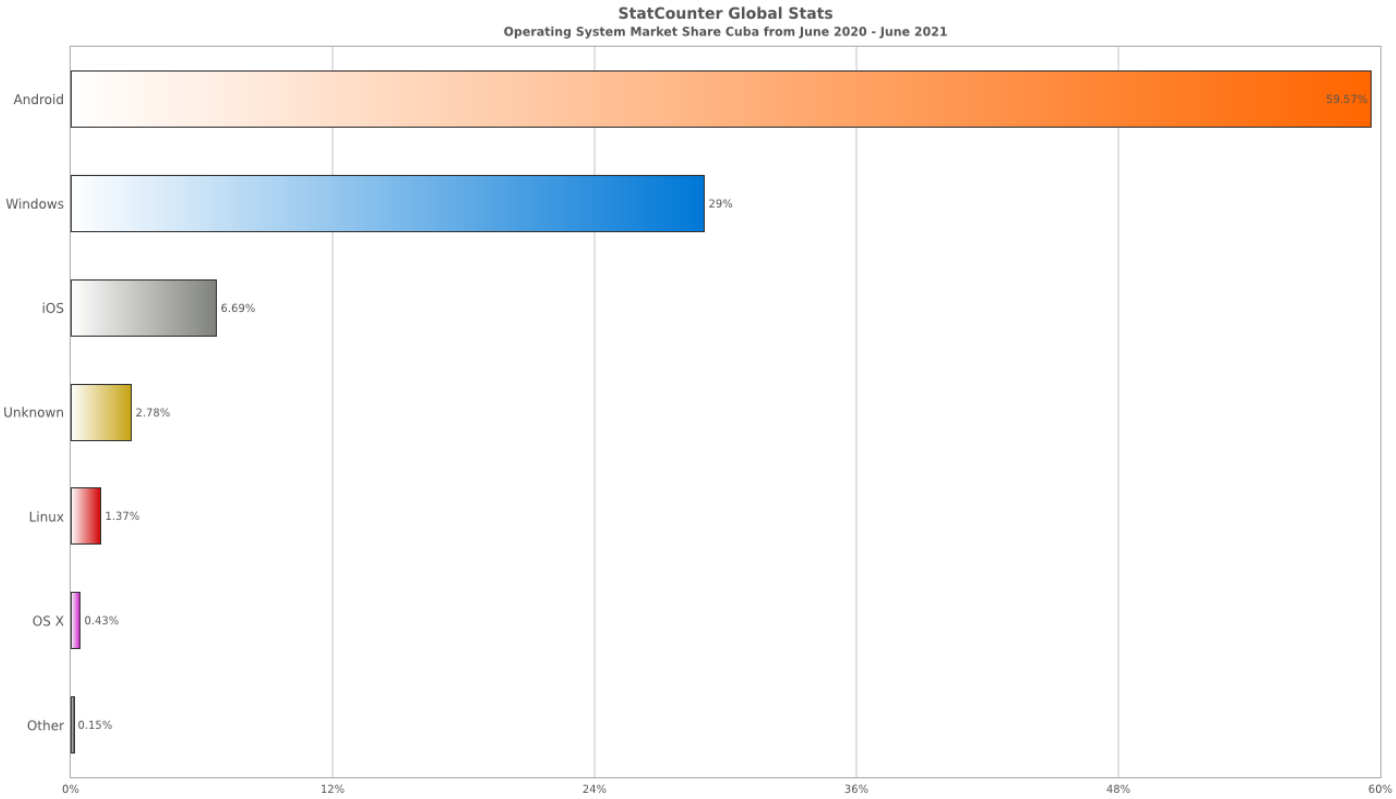


Ilustración 1: *Cuota de mercado del sistema operativo en Cuba («Mobile & Tablet Operating System Market Share Cuba»).*

Sistema Operativo iOS

iOS (anteriormente denominado iPhone OS) es un sistema operativo móvil de Apple. Originalmente desarrollado para el iPhone, siendo después usado en dispositivos como el iPod Touch, iPad y el Apple TV. La interfaz de usuario de iOS está basada en el concepto de manipulación directa, usando gestos multitáctiles. Los elementos de control contienen deslizadores, interruptores y botones. La respuesta a las órdenes del usuario son inmediatas y provee una interfaz fluida. La interacción con el sistema operativo incluye gestos como deslices, toques, pellizcos, los cuales tienen definiciones diferentes dependiendo del contexto de la interfaz.

Sistema operativo Windows Phone

Windows Phone es un sistema operativo móvil desarrollado por Microsoft, como sucesor de la plataforma Windows Mobile. A diferencia de su predecesor, está enfocado en el mercado de consumo generalista en lugar del mercado empresarial. Este sistema está diseñado para ser estéticamente similar a las versiones de escritorio de Windows, brindando la posibilidad de utilizar las herramientas de Office Mobile, Outlook Mobile e Internet Explorer. Como principal desventaja de su uso está la poca disponibilidad de aplicaciones.

Sistema operativo Android

En (Projects, 2021) se describe a Android como un sistema operativo y un *software stack* o una pila de software creados para una serie de dispositivos con diferentes factores de forma, como teléfonos, tabletas, wearables, televisores, automóviles y dispositivos conectados. Los objetivos principales de Android son crear una plataforma abierta disponible para que las operadoras, los fabricantes de equipos originales y los desarrolladores hagan realidad sus ideas y ofrezcan un producto de éxito en el mundo real que mejore la experiencia móvil de los usuarios.

Android bajo la definición de Google se considera un *"software stack"* o una pila de software, ya que está conformada por:

- El sistema operativo, donde todas las funciones se desarrollan.
- El middleware que permite la conexión entre redes
- Las aplicaciones que constituyen todos los programas que el teléfono puede ejecutar.

Para la realización de la propuesta de solución, se ha decidido utilizar el sistema operativo Android. Esta decisión responde a la política de utilización de software libre en la que se encuentra enmarcada Cuba. Es un sistema de código abierto, robusto y con miles de desarrolladores y aplicaciones en el mundo. Android domina la cuota de mercado de los smartphones con un 86.8% según el International Data Corporation en 2018. Samsung su contribuyente número uno, permanece en el lugar más alto en ventas. Todo lo anteriormente mencionado coloca a Android como la mejor opción para el desarrollo de la propuesta de solución, por lo que a continuación, se profundiza en este SO.

Android se torna realmente atractivo por diversas características al decir de (Polanco y Taibo 2011), entre ellas se encuentran:

1) Plataforma totalmente libre basado en Linux, esto permite desarrollar aplicaciones y/o modificar las ya existentes con lenguaje de Java.

2) Es multitasking permitiendo mantener distintas aplicaciones corriendo al mismo tiempo.

3) Compatible con una gran variedad de hardware en el mercado (tablets y dispositivos celulares de marcas como: Motorola, Samsung, ZTE, Huawei, Ericsson por nombrar algunas), permitiendo al usuario elegir el dispositivo que mejor se ajusta a sus necesidades.

4) Posee un portal llamado Android Market donde se tiene acceso a muchas aplicaciones que pueden ser utilizadas.

5) Permite realizar actualizaciones del sistema operativo en línea siempre y cuando el dispositivo soporte los requerimientos del mismo.

6) Puede operar soluciones tecnológicas referentes al uso de redes sociales, mensajería instantánea, correo electrónico, modificación y lectura de procesadores de palabras, hojas de cálculo, presentaciones, lectura de formatos pdf, entre otros.

7) Se puede conseguir mucha información a través de documentos web o libros.

8) Como característica importante: cuenta con el gran apoyo y la capacidad tecnológica proporcionada por su principal socio: Google.

Arquitectura de Android

Android es una pila de software de código abierto basado en Linux creada para una variedad amplia de dispositivos y factores de forma. En las siguientes líneas se dará una visión global por capas de cuál es la arquitectura empleada en Android. Cada una de estas capas utiliza servicios ofrecidos por las anteriores, y ofrece a su vez los suyos propios a las capas de niveles superiores, tal como muestra la siguiente:

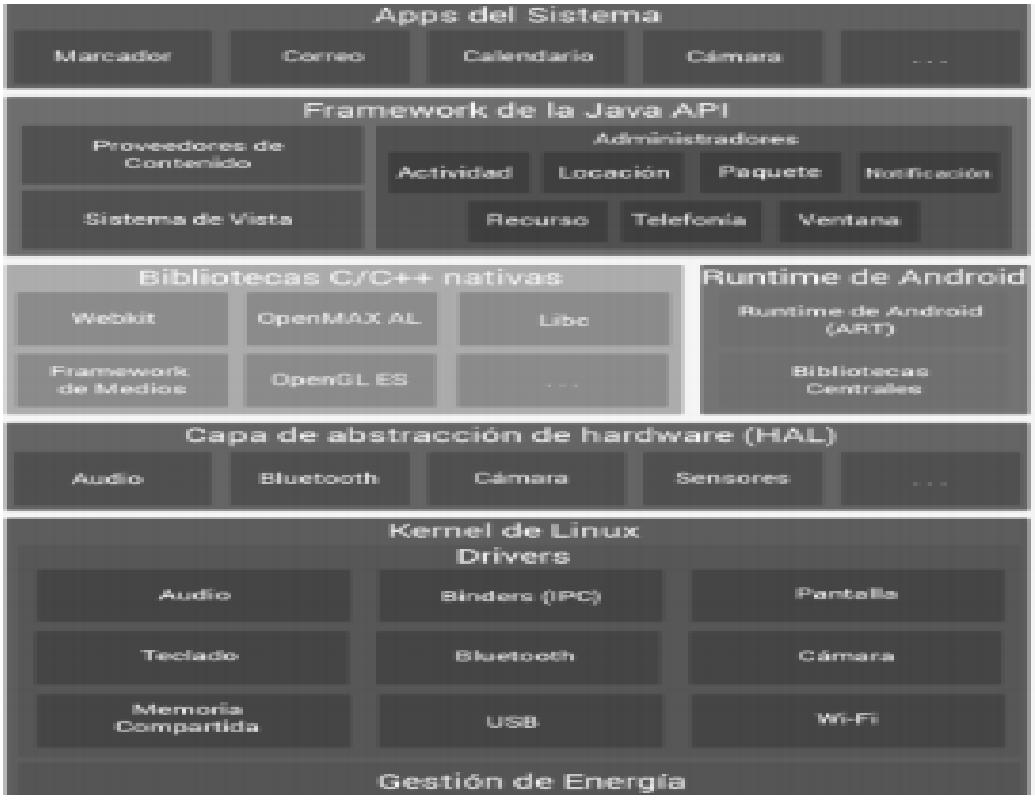


Ilustración 2: *Arquitectura del sistema operativo Android («Máster en Desarrollo de Aplicaciones Android - Arquitectura de Android»2020).*

Kernel de Linux: El núcleo de Android está formado por el sistema operativo Linux versión 2.6. Esta capa proporciona servicios como la seguridad, el manejo de la memoria, el multiproceso, la pila de protocolos y el soporte de drivers para dispositivos. Esta capa del modelo actúa como capa de abstracción entre el hardware y el resto de la pila. Por lo tanto, es la única que es dependiente del hardware.

Capa de abstracción de hardware: La **capa de abstracción de hardware** (en inglés, Hardware Abstraction Layer o HAL) es un elemento del sistema operativo que funciona como una interfaz entre el software y el hardware del sistema, proveyendo una plataforma de hardware consistente sobre la cual corren las aplicaciones. La HAL consiste en varios módulos de biblioteca y cada uno de estos implementa una interfaz para un tipo específico de componente de hardware, como el módulo de la cámara o de Bluetooth. Cuando el framework de una API realiza una llamada para acceder al hardware del dispositivo, el sistema Android carga el módulo de biblioteca para el componente de hardware en cuestión.

Runtime de Android: Para los dispositivos con Android 5.0 o versiones posteriores, cada app ejecuta sus propios procesos con sus propias instancias del tiempo de ejecución de Android (ART). El ART está escrito para ejecutar varias máquinas virtuales en dispositivos de memoria baja ejecutando archivos DEX, un formato de código de bytes diseñado especialmente para Android y optimizado para ocupar un espacio de memoria mínimo. Crea cadenas de herramientas, como Jack, y compila fuentes de Java en código de bytes DEX que se pueden ejecutar en la plataforma Android.

Antes de Android 5.0, Dalvik era el tiempo de ejecución del sistema operativo. Si tu app se ejecuta bien en el ART, también debe funcionar en Dalvik, pero es posible que no suceda lo contrario. En Android también se incluye un conjunto de bibliotecas de tiempo de ejecución centrales que proporcionan la mayor parte de la funcionalidad del lenguaje de programación Java; se incluyen algunas funciones del lenguaje Java en su versión 8.0, que el framework de la API de Java usa.

Bibliotecas C/C++ nativas: Muchos componentes y servicios centrales del sistema Android, como el ART y la HAL, se basan en código nativo que requiere bibliotecas nativas escritas en C y C++. La plataforma Android proporciona la API del framework de Java para exponer la funcionalidad de algunas de estas bibliotecas nativas a las aplicaciones (apps). Si se desarrolla una app que requiere C o C++, puedes usar el Native Development Kit (NDK) de Android para acceder a algunas de estas bibliotecas de plataformas nativas directamente desde tu código nativo.

Framework de la API de Java: Todo el conjunto de funciones del SO Android está disponible mediante API escritas en el lenguaje Java. Estas API son los cimientos que se necesitan para crear apps de Android simplificando la reutilización de componentes del sistema y servicios centrales y modulares, como los siguientes:

- Un sistema de vista enriquecido y extensible que puedes usar para compilar la interfaz de usuario de una app; se incluyen listas, cuadrículas, cuadros de texto, botones e incluso un navegador web integrable.

- Un administrador de recursos que te brinda acceso a recursos sin código, como strings localizadas, gráficos y archivos de diseño.

- Un administrador de notificaciones que permite que todas las apps muestren alertas personalizadas en la barra de estado.

- Un administrador de actividad que administra el ciclo de vida de las apps y proporciona una pila de retroceso de navegación común.

- Proveedores de contenido que permiten que las apps accedan a datos desde otras apps, como la app de contactos, o compartan sus propios datos. Los desarrolladores tienen acceso total a las mismas APIs del framework que usan las apps del sistema Android.

Aplicaciones del sistema: En Android se incluye un conjunto de apps centrales para correo electrónico, mensajería SMS, calendarios, navegación en Internet y contactos, entre otros elementos. Las apps incluidas en la plataforma no tienen un estado especial entre las apps que el usuario elije instalar; por ello, una app externa se puede convertir en el navegador web, el sistema de mensajería SMS o, incluso, el teclado predeterminado del usuario (existen algunas excepciones, como la app Configuración del sistema). Las apps del sistema funcionan como apps para los usuarios y brindan capacidades claves a las cuales los desarrolladores pueden acceder desde sus propias apps. Por ejemplo, si en tu app se intenta entregar un mensaje SMS, no es necesario que compiles esa funcionalidad tú mismo; como alternativa, puedes invocar la app de SMS que ya está instalada para entregar un mensaje al receptor que especifiques (Android Developers, 2018)..

1.1.4 Interfaces de Programación de Aplicaciones

Las Interfaces de Programación de Aplicaciones (API, por sus siglas en inglés) son un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones.

Las API permiten que sus productos y servicios se comuniquen con otros, sin necesidad de saber cómo están implementados. Esto simplifica el desarrollo de las aplicaciones y permite ahorrar tiempo y dinero. Las API le otorgan flexibilidad; simplifican el diseño, la administración y el uso de las aplicaciones, y proporcionan oportunidades de innovación, lo cual es ideal al momento de diseñar herramientas y productos nuevos (o de gestionar los actuales).

En resumen, las API le permiten habilitar el acceso a sus recursos y, al mismo tiempo, mantener la seguridad y el control. Cómo habilitar el acceso y a quiénes depende de usted. Seguridad de las API. Para conectarse a las API y crear aplicaciones que utilicen los datos o las funciones que estas ofrecen, se puede utilizar una plataforma de integración distribuida que conecte todos los elementos, incluidos los sistemas heredados y el Internet de las Cosas (IoT). (Redhat, 2021) describe tres enfoques respecto a las políticas de las versiones de las API:

Tabla 1: *Enfoques respecto a las políticas de las versiones de las API.*

Fuente: *(Redhat, 2021).*

Privado	**De partners**	**Público**
Las API solo se pueden usar internamente, así que las empresas tienen un mayor control sobre ellas. Esto permite a las empresas, un mayor control sobre sus API.	Las API se comparten con partners empresariales específicos, lo cual puede ofrecer flujos de ingresos adicionales, sin comprometer la calidad. Esto puede proporcionar flujos de ingreso adicionales, sin comprometer la calidad.	Todos tienen acceso a las API, así que otras empresas pueden desarrollar API que interactúen con las de usted y así convertirse en una fuente de innovaciones. Esto permite que terceros desarrollen aplicaciones que interactúan con su API, y puede ser un recurso para innovar.

Servicios REST

El término REST (Representational State Transfer) se originó en el año 2000, descrito en la tesis de Roy Fielding, padre de la especificación HTTP. Un servicio REST no es una arquitectura software, sino

un conjunto de restricciones con las que podemos crear un estilo de arquitectura de software, la cual podremos usar para crear aplicaciones web respetando HTTP. Según Fielding las restricciones que definen a un sistema RESTful serían:

1) **Cliente-servidor:** cada petición HTTP contiene toda la información necesaria para ejecutarla, lo que permite que ni cliente ni el servidor necesiten recordar ningún estado previo para satisfacerla. Aunque esto es así, algunas aplicaciones HTTP incorporan memoria caché. Se configura lo que se conoce como protocolo cliente-caché-servidor sin estado: existe la posibilidad de definir algunas respuestas a peticiones HTTP concretas como cacheables, con el objetivo de que el cliente pueda ejecutar en un futuro la misma respuesta para peticiones idénticas. De todas formas, que exista la posibilidad no significa que sea lo más recomendable.
2) **Interfaz uniforme:** para la transferencia de datos en un sistema REST, este aplica acciones concretas [POST (crear), GET (leer y consultar), PUT (editar) y DELETE (eliminar)] sobre los recursos, siempre y cuando estén identificados con una Identificador Uniforme de Recursos (URI, por sus siglas en inglés). Esto facilita la existencia de una interfaz uniforme que sistematiza el proceso con la información.
3) **Sistema de capas:** arquitectura jerárquica entre los componentes. Cada una de estas capas lleva a cabo una funcionalidad dentro del sistema REST. Esto ayuda a mejorar la escalabilidad, el rendimiento y la seguridad.

Las API son RESTful siempre que cumplan con las 6 limitaciones principales de un sistema RESTful:

- Arquitectura cliente-servidor: la arquitectura REST está compuesta por clientes, servidores y recursos, y administra las solicitudes con HTTP.
- Sin estado: el contenido de los clientes no se almacena en el servidor entre las solicitudes, sino que la información sobre el estado de la sesión se queda en el cliente. En su lugar, la información sobre el estado de la sesión está en posesión del cliente.
- Capacidad de caché: el almacenamiento en caché puede eliminar la necesidad de algunas interacciones cliente-servidor.
- Sistema en capas: las interacciones cliente-servidor pueden estar mediadas por capas adicionales, que pueden ofrecer otras funciones, como el equilibrio de carga, los cachés compartidos o la seguridad. Estas capas pueden ofrecer funcionalidades adicionales, como equilibrio de carga, cachés compartidos o seguridad.
- Código de demanda (opcional): los servidores pueden extender las funciones de un cliente transfiriendo código ejecutable.

- Interfaz uniforme: esta limitación es fundamental para el diseño de las API de RESTful e incluye 4 aspectos:
    - Identificación de recursos en las solicitudes: los recursos se identifican en las solicitudes y se separan de las representaciones que se devuelven al cliente.
    - Administración de recursos mediante representaciones: los clientes reciben archivos que representan los recursos. Estas representaciones deben tener la información suficiente como para poder ser modificadas o eliminadas.
    - Mensajes autodescriptivos: cada mensaje que se devuelve al cliente contiene la información suficiente para describir cómo debe procesar la información.
    - Hipermedios es el motor del estado de la aplicación: después de acceder a un recurso, el cliente REST debe ser capaz de descubrir mediante hipervínculos todas las otras acciones que están disponibles actualmente.

Estas limitaciones pueden parecer demasiadas, pero son mucho más sencillas que un protocolo definido previamente. Por eso, las API de RESTful son cada vez más frecuentes que las de SOAP. En los últimos años, la especificación de OpenAPI se ha convertido en un estándar común para definir las API de REST. OpenAPI establece una forma independiente del lenguaje para que los desarrolladores diseñen interfaces API de REST, que permite a los usuarios entenderlas con el mínimo esfuerzo.

En la actualidad la mayoría de las empresas utilizan API REST para crear servicios. Esto se debe a que es un estándar lógico y eficiente para la creación de servicios web. Estos sistemas están caracterizados por el uso de hipermedios (término que en el ámbito de las páginas web define el conjunto de procedimientos para crear contenidos que contengan texto, imagen, video, audio y otros métodos de información) para permitir al usuario navegar por los distintos recursos de una API REST a través de enlaces HTML. Para el desarrollo de la propuesta de solución se utilizará un API REST.

1.2 Análisis de aplicaciones móviles para sitios gubernamentales

El uso de las TIC por parte del sector público, con el objetivo de mejorar la información y la prestación de servicios, alentar la participación de los ciudadanos en el proceso de toma de decisiones y hacer que el gobierno sea más responsable, transparente y eficaz, es cada vez más notable (Local e-Governance, 2021). Esto se evidencia en la creación de aplicaciones móviles a nivel nacional e internacional, las cuales cumplen con características que permiten afirmar que han colaborado con la

existencia de un gobierno electrónico. A continuación, se exponen algunas de las aplicaciones que se analizaron con el propósito de identificar funcionalidades y características que pudieran incluir en la propuesta solución.

1.2.1 Aplicaciones a nivel internacional

Komgida gobierno Vasco

Esta aplicación móvil pone a disposición de los ciudadanos los datos públicos del gobierno del País Vasco. Con esta herramienta digital, los usuarios pueden acceder a la información de los diferentes organismos públicos de la región, como las diputaciones forales, juntas generales, ayuntamientos y hasta el propio Parlamento vasco, junto con los datos de las sociedades y entidades públicas que pertenecen al gobierno vasco. Además de la información institucional, también está disponible la información relativa a los representantes públicos y de sus medios de contacto directo, para que los ciudadanos que lo deseen puedan comunicarse directamente con los dirigentes vascos de los diferentes niveles de gobierno.

Partidos Públicos

Transparentar la actividad política chilena resulta más sencilla con esta *aplicación móvil*, considerada una plataforma “inédita”. Lo más destacado de sus funciones es que permite visualizar y comparar fácilmente la información más relevante de los partidos políticos como la publicación de sus finanzas, datos sobre afiliados y la presencia nacional, entre otros temas importantes. Además, permite hacer búsquedas con filtros y llevar un registro histórico de los contenidos que publican los partidos en la plataforma para hacer un seguimiento de su evolución. Esta aplicación es un impulso para el cumplimiento de la nueva Ley de Partidos Políticos de Chile y de los estándares internacionales de apertura de datos.

Presidencia RD

Esta es la aplicación oficial de la Presidencia de la República Dominicana. Esta permite acceder a las últimas noticias publicadas por la Dirección General de Comunicación (DICOM). La aplicación permite el acceso a informaciones generales, una galería completa de fotos, videos y otros documentos relacionados con las actividades propias del gobierno dominicano y que están disponibles en los canales

de comunicación digital del DICOM. Lo más innovador de esta aplicación es la posibilidad de recibir una notificación y ver en vivo las transmisiones que hace la Presidencia de la República.

1.2.2 Aplicaciones a nivel nacional

Segurmovil

Una forma de viabilizar el acceso a información - y las condiciones de cada seguro - ya se encuentra disponible en esta aplicación desarrollada para Android, de la Empresa de Seguros Nacionales (ESEN). Esta iniciativa posibilitará a los usuarios acceder a los modelos para autogestionar la concertación, modificación y reclamación de los diversos seguros con la compañía, y visibilizar de inmediato el estado de todas las pólizas contratadas, con datos relevantes como el valor asegurado o la prima a pagar. SegurMovil constituye una vía práctica y efectiva para obtener datos relevantes y herramientas que permitan a los clientes un mejor trabajo con los servicios que ofrece la ESEN.

Participación Popular

Facilita la denuncia de situaciones y problemas que, identificados por la población, es un vínculo directo a las instancias de gobierno. *"*El objetivo es que los ciudadanos puedan denunciar cualquier incidencia en diferentes sectores, problemas con salideros, alteración de precios, transporte, alcantarillado, alumbrado público. Pueden agregar también un campo a través de la opción otros*"* («Disponible en Apklis la aplicación Participación Popular que facilitará denuncias ciudadanas»).

Cada gobierno tiene sus propias intenciones e informaciones, aunque existen varias aplicaciones a nivel internacional y nacional, las aplicaciones mencionadas con anterioridad muestran contenidos específicos de los gobiernos correspondientes y no responden a la misión que tiene prevista el sitio de la presidencia. Una solución para el sitio de la presidencia debe integrar contenido de varias entidades gubernamentales e información actual de Cuba y el mundo. Es por ello que se decide realizar una aplicación propia. En la Tabla 9, se analizan estos sistemas:

Tabla 2: *Análisis de los sistemas homólogos, a partir de un grupo de características.*

Sistema	Software libre	Aplicación web	Dependencia de internet
Komgida gobierno Vasco	-	-	-
Servel	-	**X**	**X**

Presidencia RD	-	**X**	**X**
Participación Popular	**X**	-	**X**
SegurMovil	**X**	-	**X**

1.3 Tecnologías para el desarrollo del sistema

Teniendo en cuenta que el sistema a desarrollar es una aplicación que pertenecerá al conjunto de soluciones de la UPSI esta debe ser desarrollada empleando las tecnologías y herramientas de la Dirección de Infocomunicaciones, A continuación, se presentan las diferentes herramientas utilizadas durante el desarrollo de la aplicación móvil, así como las características que estas presentan.

1.3.1 Metodología de Desarrollo de Software

Una metodología es una colección de procedimientos, técnicas, herramientas y documentos auxiliares que ayudan a los desarrolladores de software en sus esfuerzos por implementar nuevos sistemas de información (Peñalvo, Holgado y Ingelmo, 2020). La selección y uso de la metodología adecuada es uno de los puntos vitales para el éxito de un proyecto. Esta dicta qué hacer y cómo hacerlo, es el cimiento que sostendrá todo el proceso de desarrollo posterior, por lo que resulta crítico tener buen juicio al seleccionar la que mejor se adapte a las características del proyecto en cuestión.

Las metodologías de desarrollo de software suelen clasificarse según sus características fundamentales y el enfoque utilizado. Hasta el momento han sido identificados dos grandes grupos: metodologías tradicionales o prescriptivas y metodologías ágiles o ligeras.

Metodología tradicional: Impone una disciplina de trabajo sobre el proceso de desarrollo del software, con el objetivo de conseguir un software más eficiente y de mayor calidad. Se centra fundamentalmente en el control del proceso, a través de una rigurosa definición de roles, actividades, artefactos, herramientas y notaciones para el modelado y documentación detallada (Fuentes 2015a).Otra de las características a destacar dentro de este enfoque son los altos costos que se producen al implementar un cambio y al no ofrecer una buena solución para proyectos donde el entorno es volátil (Toro y Rendón 2016).

Entre las principales metodologías tradicionales tenemos RUP y MSF. Estas centran su atención en llevar una documentación exhaustiva de todo el proyecto y en cumplir con un plan de proyecto, definido todo esto, en la fase inicial del desarrollo del proyecto.

Metodología Ágil: Es aquella que posibilita adaptar la forma de trabajo a las condiciones que presenta el proyecto, para conseguir flexibilidad e inmediatez en la respuesta para amoldar el proyecto y su desarrollo a las circunstancias específicas del entorno (Gabriel 2016).Dentro de las metodologías ágiles más utilizadas se encuentran AUP (Agil Unified Process), Scrum y XP (Extreme Programming).

Tabla 3: *Comparación entre metodología ágil y tradicional*

Fuente: *(Gabriel 2015).*

Metodologías ágiles	***Metodologías tradicionales***
Basadas en heurísticas provenientes de prácticas de producción de código.	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo.
Especialmente preparados para cambios durante el proyecto.	Cierta resistencia a los cambios.
Impuestas internamente (por el equipo)	Impuestas externamente.
Proceso menos controlado, con pocos principios.	Proceso mucho más controlado, con numerosas políticas/normas.
No existe contrato tradicional o al menos es bastante flexible.	Existe un contrato prefijado.
El cliente es parte del equipo de desarrollo.	El cliente interactúa con el equipo de desarrollo mediante reuniones.
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio.	Grupos grandes y posiblemente distribuidos.
Pocos artefactos.	Más artefactos.
Pocos roles.	Más roles.
Menos énfasis en la arquitectura del software.	La arquitectura del software es esencial y se expresa mediante modelos.
Poca documentación.	Documentación exhaustiva.
Muchos ciclos de entrega.	Pocos ciclos de entrega.

Como se muestra en la Tabla 2, se puede apreciar que las metodologías ágiles, son más baratas en tiempo y recursos, obteniendo los mismos o mejores resultados ante las metodologías tradicionales.

Metodología móvil

Dentro del mundo de las metodologías diseñadas específicamente para aplicaciones móviles se destacan *Mobile-D* una de las pocas metodologías con certificación CMMI que fue desarrollada por científicos del Instituto de Investigación Finlandés (VTT) y Proceso de desarrollo móvil en espiral desarrollado por Vahid Rahimian y Raman Ramsin. A continuación, se refieren algunas de las propiedades que presentan estas metodologías que son aplicables al desarrollo móvil:

Mobile – D

Es un modelo propuesto por Pekka Abrahamsson y su equipo del VTT (Valtion Teknillinen Tutkimuskeskus), en inglés Technical Research Centre of Finland) en Filandia. El ciclo del proyecto se divide en cinco fases: exploración, inicialización, producción, estabilización y prueba del sistema, como se muestra en la Ilustración 3. En general, todas las fases con la excepción de la primera fase exploratoria contienen tres días de desarrollo distintos: planificación, trabajo y liberación.

HMD

Otra metodología muy importante en la actualidad es HMD (*Hybrid Methodology Design* o Diseño de Metodología Híbrida), esta es una combinación del desarrollo adaptativo de software (*Adaptive Software Development*, ASD) y el diseño de nuevos productos (*New Product Development*, NPD). Parte de un ciclo de vida tradicional (análisis, diseño, implementación, pruebas y transición), incluyendo al final una fase de comercialización. En la primera iteración se divide la fase de análisis con la intención de mitigar riesgos de desarrollo; de la misma forma, el diseño también se segmenta para introducir algo de diseño basado en arquitectura.

Proceso de Desarrollo móvil en espiral

Es la más reciente propuesta de metodologías diseñadas específicamente para aplicaciones móviles y que aún se encuentra en etapa experimental. Esta propuesta metodológica utiliza el modelo de desarrollo en espiral como base e incorpora procesos de evaluación de la usabilidad. Prioriza la participación del usuario en todos los procesos del ciclo de vida de diseño, con el fin de garantizar un diseño centrado en el usuario, aun cuando se trata de un modelo de proceso orientado a proyectos grandes y costosos, ya que está destinado a ser un modelo de reducción de riesgos *(Fuentes 2015a; Gabriel 2015)*.

El proceso permite a los desarrolladores de aplicaciones móviles, detallar los criterios de usabilidad de la aplicación, el primer paso es identificar a los usuarios, las tareas y los contextos en los que se utilizará la aplicación móvil. El siguiente paso es dar prioridad a los atributos de usabilidad, identificar qué atributos son los más importantes para la aplicación, y para cada uno definir un conjunto de métricas para verificar el grado en que se cumplen en la aplicación final. El modelo en espiral tiene varias iteraciones, cada una producirá prototipos que ayudarán la evaluación del usuario *(Fuentes 2015a; Gabriel 2015)*.

XP (Extreme Programming)

Esta metodología ágil, es diseñada para entornos dinámicos en los cuales se presentan equipos de desarrollo pequeños (hasta 10 programadores), orientada fuertemente hacia la codificación y programación de esta forma disminuye el tiempo de desarrollo de los proyectos. Hace énfasis en la comunicación informal y se necesita muy poca documentación en el desarrollo del sistema. La planificación de la misma es por entregas y posee las siguientes características (M*(Metodología XP, 2015)*.

Para la presente investigación se asumirá una metodología ágil con el fin de aprovechar las facilidades que brinda en cuanto a la documentación y el entorno cambiante que tiene el proyecto. A raíz de que el proyecto se lleva a cabo por un solo desarrollador es más importante centrarse en la correcta obtención del sistema, sin importar cuán documentado esté, además, es seleccionada por el tiempo de desarrollo del sistema y la estrecha relación cliente – desarrollador, para lograr un proyecto flexible a los cambios que ocurran durante el proceso de desarrollo de software. Es por ello que, para la implementación de la aplicación móvil del Sitio de la Presidencia y Gobierno de Cuba, se decidió seleccionar como metodología a utilizar XP.

1.3.2 Herramienta de modelado

Visual Paradigm constituye una herramienta CASE profesional, que utiliza UML como lenguaje de modelado y que soporta el ciclo de vida completo del desarrollo de software, además de tener la ventaja de ser multiplataforma (Fowler 2018).

Características:

- Soporta todos los diagramas de entidad-relación.
- Genera Java, C#, C++, PHP y lenguaje de definición de datos (DDL) para todas las bases de datos populares.

- Permite diseñar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. El mismo proporciona abundantes tutoriales del Lenguaje de Modelado, demostraciones interactivas de UML y proyectos UML.
- Se integra con herramientas Java, como son: Eclipse/IBM, NetBeansIDE, entre otras («Ideal Modeling & Diagramming Tool for Agile Team Collaboration» 2017).

1.3.2 Lenguaje de programación

Para el desarrollo de aplicaciones Android, se pueden emplear diversos lenguajes de programación entre ellos Java, C, C++, Visual Basic, .NET y Kotlin. Para la implementación de la propuesta de solución se ha decidido utilizar el lenguaje Kotlin debido a las grandes potencialidades que posee.

Kotlin es un lenguaje de programación desarrollado por Jetbrains. Se ejecuta principalmente en la máquina virtual de Java (JVM), pero también brinda la capacidad de ejecutar JavaScript, lo que permite su uso para navegadores y desarrollo de lado del servidor. Es perfectamente adaptable para el desarrollo de aplicaciones de Android, ya que brinda todas las ventajas de un lenguaje moderno a la plataforma de Android sin introducir nuevas restricciones:

- **Compatibilidad:** Totalmente compatible con JDK 6, lo que garantiza que las aplicaciones de Kotlin puedan ejecutarse en dispositivos Android antiguos sin problemas. Las herramientas de Kotlin son totalmente compatibles con Android Studio y son compatibles con el sistema de compilación de Android.
- **Rendimiento:** una aplicación Kotlin se ejecuta tan rápido como una Java equivalente, gracias a una estructura de código de byte muy similar. Con el soporte de Kotlin para las funciones en línea, el código que utiliza lambdas a menudo se ejecuta incluso más rápido que el mismo código escrito en Java.
- **Interoperabilidad:** Es 100% interoperable con Java, lo que permite utilizar todas las bibliotecas de Android existentes en una aplicación Kotlin. Esto incluye el procesamiento de anotaciones, por lo que el enlace de datos y Dagger también funcionan.
- **Perfil:** Posee una biblioteca de tiempo de ejecución muy compacta, que se puede reducir aún más mediante el uso de ProGuard.
- **Tiempo de compilación:** Kotlin admite una compilación incremental eficiente, por lo que, si bien hay una sobrecarga adicional para las compilaciones limpias, las compilaciones incrementales suelen ser tan rápidas o más rápidas que con Java.

- **Curva de aprendizaje:** para un desarrollador de Java, comenzar con este lenguaje es muy fácil. El convertidor automatizado de Java a Kotlin incluido en el complemento Kotlin ayuda con los primeros pasos.

1.3.4 Entorno de desarrollo integrado

Un entorno de desarrollo integrado o entorno de desarrollo interactivo, en inglés Integrated Development Environment (IDE), es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software. Es un sistema de software para el diseño de aplicaciones que combina herramientas del desarrollador comunes en una sola interfaz gráfica de usuario (GUI). Generalmente, un IDE cuenta con las siguientes características:

1) *Editor de código fuente:* editor de texto que ayuda a escribir el código de software con funciones como el resaltado de la sintaxis con indicaciones visuales, el relleno automático específico del lenguaje y la comprobación de errores a medida que se escribe el código.

2) *Automatización de compilaciones locales:* herramientas que automatizan tareas sencillas e iterativas como parte de la creación de una compilación local del software para su uso por parte del desarrollador, como la compilación del código fuente de la computadora en un código binario, el empaquetado del código binario y la ejecución de pruebas automatizadas.

3) *Depurador:* programa que sirve para probar otros programas y mostrar la ubicación de un error en el código original de forma gráfica.

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android y se basa en IntelliJ IDEA. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece más funciones que aumentan la productividad durante la compilación de apps para Android, como las siguientes:

1) Un sistema de compilación basado en Gradle flexible.

2) Un emulador rápido con varias funciones.

3) Un entorno unificado en el que puedes realizar desarrollos para todos los dispositivos Android.

4) Instant Run para aplicar cambios mientras tu app se ejecuta sin la necesidad de compilar un nuevo APK.

5) Integración de plantillas de código y GitHub para ayudarte a compilar funciones comunes de las apps e importar ejemplos de código. • Gran cantidad de herramientas y frameworks de prueba.

6) Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versión, etc.

7) Compatibilidad con C++ y NDK.

8) Soporte incorporado para Google Cloud Platform, lo que facilita la integración de Google Cloud Messaging y App Engine («Android Developers» 2018).

Se ha decidido utilizar el IDE Android Studio ya que es el entorno oficial para desarrollar aplicaciones Android.

1.3.5 Sistema Gestor de Base de Datos

SQLite es una biblioteca en proceso que implementa un motor de base de datos SQL transaccional independiente, sin servidor y de configuración cero. El código para SQLite es de dominio público y, por lo tanto, es gratuito para cualquier uso, comercial o privado. SQLite es la base de datos más implementada del mundo, incluyendo varios proyectos de alto perfil. SQLite es un motor de base de datos SQL incorporado. A diferencia de la mayoría de las otras bases de datos SQL, SQLite no tiene un proceso de servidor por separado. SQLite lee y escribe directamente en archivos de disco ordinarios. Una base de datos SQL completa con múltiples tablas, índices, disparadores y vistas, está contenida en un solo archivo de disco. El formato de archivo de la base de datos es multiplataforma: puede copiar libremente una base de datos entre sistemas de 32 bits y de 64 bits o entre arquitecturas big-endian y little-endian. Estas características hacen que SQLite sea una opción popular como formato de archivo de aplicación (About SQLite, 2018).

Conclusiones del capítulo

En este capítulo se han abordado los elementos teóricos que dan sustento a la propuesta de solución del problema planteado, en tal sentido se puede arribar a las siguientes conclusiones:

- La definición de los principales conceptos asociados al dominio de la presente investigación y las relaciones entre estos, permite alcanzar una mayor comprensión de la propuesta de solución.
- El análisis de aplicaciones móviles gubernamentales a nivel nacional e internacional, permite identificar las tendencias en cuanto al desarrollo de estas aplicaciones, para lograr así, una mayor aceptación por parte de los usuarios.
- El estudio de la documentación sobre la metodología de desarrollo, así como las herramientas, tecnologías y lenguajes de programación, permite especificar el ambiente de desarrollo más adecuado para dar solución al problema planteado.

CAPÍTULO 2: PLANIFICACIÓN Y DISEÑO DEL SISTEMA

En el presente capítulo se realiza una descripción de la solución propuesta, a partir de su planificación y diseño. Se describen los procedimientos llevados a cabo para el desarrollo de la aplicación, haciendo uso de la metodología de desarrollo de software seleccionada. Se recopilan los requisitos funcionales y no funcionales que el sistema debe cumplir y se define el comportamiento del mismo, así como las restricciones del diseño, concebidas para la construcción de la aplicación.

2.1 Descripción del negocio

Para un mayor entendimiento del problema se procede a esclarecer las terminologías asociadas al negocio, definiendo los términos más importantes que serán de utilidad para la posterior confección de la aplicación.

2.1.1 Sitios presidenciales

Los sitios presidenciales según Gil-García, 2012, son una puerta de acceso integrada al sitio de Internet del gobierno estatal, y provee tanto a entidades externas como a personal de gobierno con un punto único de acceso en línea a recursos e información del estado. Este tipo de sitios, posiciona a la entidad frente a la ciudadanía, facilita a los usuarios el acceso a información relevante de otras entidades (a través de hipervínculos) y promueven la participación ciudadana en procesos democráticos como se refiere en (Giulianelli et al. 2013). Entre las características propias de este tipo de sitio web, se encuentran:

- Iconografía amigable (Facilita el acceso a la información).
- Lenguaje claro y preciso.
- Formatos estandarizados (Uniformidad de contenidos).
- Información actualizada.
- Resumen informativo con gráficos estadísticos (Permite una mejor comprensión).

2.1.2 Gobierno electrónico

Local e-Governance | United Nations Educational, Scientific and Cultural Organization, 2005 define el e-governance, electronic governance o gobierno electrónico, como el empleo, por parte del sector público, de tecnologías de la información y de la comunicación con el objetivo de facilitar la difusión de la información, ofrecer determinados servicios, alentar al ciudadano a que participe en el proceso de toma de decisiones y hacer el gobierno más responsable, transparente y eficaz. La siguiente clasificación, basada en la especificación de la UNESCO (Organización de las Naciones Unidas para la Educación,

la Ciencia y la Cultura) al decir de Giulianelli, 2013, muestra los conceptos básicos y paralelamente campos de aplicación de la gobernabilidad electrónica:

1. **Administración electrónica (e-administración):** Se refiere a la mejora de los procesos gubernamentales internos, la gestión de los funcionarios del sector público y los procesos de ejecución e información.
2. **Servicios electrónicos (e-servicios):** Se refiere a la mejora en el acto de proveer información y ofrecer acceso a servicios públicos a los ciudadanos. También están incluidas las informaciones sobre eventos, espectáculos, transporte público, bolsa de trabajo, políticas de empleo, licitaciones, mapas, etc. Como ejemplos de servicios interactivos se pueden mencionar: solicitudes de documentos públicos, solicitudes de documentos legales y certificados, expedición de permisos y licencias, otorgamiento de turnos, pagos on line de impuestos, tasas y servicios.
3. **Democracia electrónica (e-democracia):** Implica una mayor y más activa participación ciudadana en el proceso de toma de decisiones gracias a las TIC. Como ejemplos se pueden mencionar: encuestas, foros, chat, blogs, paneles, referéndums, listas de correo, boletín por mail, contacto directo con autoridades, responsables de áreas y representantes, opiniones y sugerencias de los ciudadanos, libro de quejas, preguntas frecuentes y sus respuestas.

2.1.3 Informatización de la sociedad

La informatización de la sociedad está definida en (Política Integral, 2017) como el proceso de utilización ordenada y masiva de las Tecnologías de la Información y las Comunicaciones en la vida cotidiana, para satisfacer las necesidades de todas las esferas de la sociedad, en su esfuerzo por lograr cada vez más eficacia y eficiencia en todos los procesos y por consiguiente mayor generación de riqueza y aumento en la calidad de vida de los ciudadanos .

Cuba ha identificado desde muy temprano la conveniencia y necesidad de dominar e introducir en la práctica social las TIC, y lograr así, una cultura digital como una de las características imprescindibles del hombre nuevo, lo que facilitaría a la sociedad cubana acercarse más hacia el objetivo de un desarrollo sostenible.

Nuestro país no queda exento de este ambicioso reto, para lograr este objetivo ha declarado su propia política para la informatización de la sociedad:

- Que el desarrollo de este sector se convierta en un arma para la defensa de la Revolución.
- Garantizar la ciberseguridad frente a las amenazas, los riesgos y ataques de todo tipo.
- Asegurar la sostenibilidad y soberanía tecnológica.

- Potenciar el acceso de los ciudadanos al empleo de las nuevas Tecnologías de la Informática.
- Preservar el desarrollo del capital humano asociado a la actividad.
- Desarrollar y modernizar coherentemente todas las esferas de la sociedad, en apoyo a las prioridades del país en correspondencia con el ritmo de desarrollo de la economía.
- Integrar la investigación, el desarrollo y la innovación con la producción y comercialización de productos y servicios.

2.1.4 Acceso a la información

El acceso a la información comprende diferentes temáticas que se refieren a la disponibilidad, la accesibilidad y la asequibilidad de la información. Estas, están representadas por el multilingüismo, los metadatos, la interoperabilidad, los programas informáticos de fuente abierta, el contenido libre, las licencias Creative Commons, así como las necesidades especiales de los discapacitados. El novedoso ambiente tecnológico y económico origina preocupaciones sobre la degradación del acceso a determinadas informaciones y conocimientos que antes se encontraban libremente. Al mismo tiempo existe Internet, que ofrece la oportunidad de compartir información, promover la diversidad de idiomas y rescatar idiomas que se estaban perdiendo. A partir del año 2003 la UNESCO ha trazado una serie de políticas para respaldar los esfuerzos mundiales a favor de los programas gratuitos y de código fuente abierto. También buscando la utilización de las TIC para lograr un acceso equitativo a la información sobre todo para las personas con discapacidad (UNESCO, 2017).

2.2 Planificación

La Planificación es la primera fase del ciclo de vida de la metodología XP. Los usuarios plantean a grandes rasgos las funcionalidades y requerimientos que desean obtener de la aplicación y el equipo de trabajo se familiariza con las tecnologías y herramientas seleccionadas para el desarrollo. Además, se realiza un análisis de los conceptos fundamentales asociados al proceso como se plantea en (Fuentes, 2015).

2.2.1 Propuesta de solución

Utilizando la información recopilada en el capítulo preliminar, se plantea el desarrollo de una apk con la intención de facilitar el acceso de la población a la información del sitio oficial de la Presidencia y Gobierno de Cuba. La misma contará con varias secciones en la que se presenta de forma organizada los contenidos. Algunos de estos contenidos los usuarios podrán descargarlos y/o acceder a enlaces de páginas de interés presentes en las secciones. Dispondrá de un buscador, el cual, permitirá buscar por letras, palabras o frases, obteniendo los resultados de forma inmediata. Tendrá un filtro donde se

pueden filtrar las noticias en dependencia de la categoría y fecha de esta. La apk además contará con un directorio de autores de las noticias publicadas donde puede verse los enlaces a sus redes sociales. Además, tendrá un directorio de autores del sitio que permitirá ver las bibliografías. Por último, tendrá la sección de contactos, donde estará la información para dirigirse a Atención a la Ciudadanía para presentar quejas.

2.3 Diseño

La metodología de desarrollo XP plantea prácticas especializadas que accionan directamente en la realización del diseño para lograr un sistema robusto y reutilizable. En esta fase se presentan los requisitos funcionales y no funcionales de la aplicación. Se trata en todo momento de conservar su simplicidad, o sea, crear un diseño evolutivo que vaya mejorando y que permita hacer entregas pequeñas y frecuentes de valor para el cliente, basado principalmente en el desarrollo de las tarjetas CRC tal y como se plantea en Kendall, 2005.

2.3.1 Requisitos funcionales

Los requerimientos funcionales como especifica Ramos, 2017, son aquellos directamente relacionados con las funciones o las reacciones que el sistema debe proporcionar. Son restricciones impuestas al funcionamiento del sistema, tales como las limitaciones de tiempo, presupuesto, proceso de desarrollo, la política de la organización, entre otras.

Después del encuentro con el cliente, se obtuvo un total de once (11) requisitos funcionales, a los cuales se les asignó una prioridad teniendo en cuenta la importancia fijada por el cliente a partir de sus necesidades. Los mismos se especifican en la siguiente tabla:

Tabla 4: *Requisitos funcionales de la apk.*

Fuente: Elaboración propia.

Requisito	Prioridad
RF1: Cargar imagen.	**MEDIA**
RF2: **Almacenar imágenes obtenidas.**	**MEDIA**
RF3: Mostrar contenido.	**Alta**
RF4: **Mostrar enlaces.**	**MEDIA**
RF5: Realizar búsqueda simple.	**Baja**
RF6: Realizar búsqueda avanzada.	**Baja**
RF7: Descargar materiales	**Alta**
RF8: **Mostrar contactos.**	**Alta**
RF9: Mostrar descripción del contenido.	**Baja**

RF10: **Mostrar autor(es) de las noticias.**	**Baja**
RF11: Mostrar contenido agrupado por etiqueta.	**Alta**

2.3.2 Requisitos no funcionales

Un requisito no funcional o atributo de calidad, en la ingeniería de sistemas y la ingeniería de software, especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, ya que estos corresponden a los requisitos funcionales. Por tanto, Young, 2001 se refiere a todos los requisitos que no describen información a guardar, ni funciones a realizar, sino características de funcionamiento. Por esto, suelen denominarse atributos de calidad de un sistema. Estos atributos se refieren a las características que hacen al sistema estable, usable, rápido, confiable y escalable.

Distribuidos en especificaciones de usabilidad, confiabilidad, portabilidad, eficiencia, y otros, se obtuvo un total de quince (15) requisitos no funcionales (RnF), los cuales se relacionan a continuación:

Requisitos de implementación

- *RnF 1:* La aplicación requiere un sistema operativo Android con versión 4.0 o superior.
- *RnF 2:* La aplicación requiere la publicación de los servicios REST para actualizar las publicaciones.
- *RnF 3:* La aplicación requiere que el protocolo de comunicación que se utilice sea HTTPS.
- *RnF 4:* La aplicación se implementará utilizando Android Studio como Entorno de Desarrollo Integrado (IDE), sistema de gestión de bases de datos: SQL y lenguaje de programación: Kotlin.

Requisitos de hardware (las propiedades mínimas de la PC deben ser)

- *RnF 5:* Microprocesador: Intel Core i3 o superior.
- *RnF 6:* Memoria RAM: 4 GB DDR4.
- *RnF 7:* Tarjeta de Red: Fast-Ethernet 100 MB/s.

Requisitos de interfaz

- *RnF 8:* El diseño visual de la aplicación deberá ser adaptable o adaptativo, de manera que se ajusten a los dispositivos que se estén utilizando para visualizarlos. Se debe tener en cuenta, el reducido tamaño de las pantallas de los dispositivos móviles y su variedad (generalmente entre 4 y 10 pulgadas de diagonal).
- *RnF 9:* La tipografía debe ser uniforme, de un tamaño adecuado.
- *RnF 10:* Debe existir un menú principal que comprenda todas las opciones disponibles al usuario y este debe estar accesible desde todas las vistas.

Confiabilidad

- *RnF 11:* El sistema debe ser tolerante a fallos, y mostrar sólo la información necesaria para orientar al usuario.

Portabilidad

- *RnF 12:* La propuesta de desarrollo debe ser capaz de ejecutarse en los dispositivos móviles, así como adaptar su interfaz a cualquier dispositivo, ya sea una tablet o teléfono celular.

Eficiencia

- *RnF 13:* La apk debe permitir que los usuarios interactúen con él de manera simultánea.
- *RnF 14:* El tiempo de demora de una petición al servidor debe ser menor de cinco (5) segundos aproximadamente.

Seguridad

- *RnF 15:* Ante los errores que puedan ocasionarse en la apk, no se deben mostrar detalles de información que puedan comprometer su seguridad e integridad.

2.3.3 Historias de usuarios

Las historias de usuario son un instrumento para el levantamiento de requerimientos para el desarrollo de un software, que ha emergido con la aparición de los nuevos marcos de trabajo de desarrollo ágil, como una de las diferentes técnicas que comprenden la programación extrema. Algunas características deseables de las historias de usuario según Hernández y Baquero, 2020 son:

- Que sean escritas por el usuario o por un analista de negocio que le represente.
- Frase corta que encaje en una tarjeta de 3 por 5 pulgadas.
- Debe describir el rol desempeñado por el usuario en el sistema, descrito de forma explícita.
- Debe describir el beneficio para el área de negocio que representa esta funcionalidad.

Para la presente investigación, se generaron un total de once (11) HU, a continuación, solo se muestran dos (2) de ellas, pertenecientes a los RF 7 y 8 respectivamente, especificados en el sub-epígrafe 2.3.1.

Tabla 5:*HU_7: Descargar materiales.*

Fuente: *Elaboración propia.*

Historia de usuario	
Número: **HU_7**	**Nombre:** Descargar materiales
Programador responsable: Enma N. Piquero Moskaquiel	Iteración asignada: 5
Prioridad: Alta	
Descripción: El usuario debe tener acceso a poder descargar las documentaciones presentes en la aplicación según su interés.	

Observaciones: Para realizar esta operación, el usuario debe estar online, es decir, conectado a una red wifi o por datos móviles. Además, las documentaciones deben estar en formato pdf.

Tabla 6:*HU_8 Mostrar contactos.*

Fuente: *Elaboración propia.*

Historia de usuario	
Número: HU_8	**Nombre:** Mostrar contactos
Programador responsable: Enma N. Piquero Moskaquiel	Iteración asignada: 1
Prioridad: Alta	
Descripción: En la sección ¨Contactos¨, se le muestra al usuario elementos de gran relevancia, tales como: • formulario de opiniones, en el cual se pueden incluir quejas, • dirección para emitir las quejas (Atención a la Población), • teléfonos para comunicar con el lugar y, • el correo electrónico de la presidencia.	
Observaciones: Para realizar alguna de estas operaciones, el usuario debe estar online, es decir, conectado a una red wifi o por datos móviles.	

2.3.4 Tarjetas CRC

Las tarjetas CRC (Clase-Responsabilidad-Colaboración) son una herramienta de *brainstorming* o tormenta de ideas - es una técnica de grupo para generar ideas originales en un ambiente relajado - usada como metodología para el diseño de software orientado a objetos, creada por Kent Beck y Ward Cunningham.

La técnica consiste en dibujar una tarjeta por cada clase u objeto, y dividirla en tres zonas:

1. En la parte superior, el nombre de la clase.
2. Debajo, en la parte izquierda, las responsabilidades de dicha clase. Son sus objetivos, a alto nivel.

3. A la derecha de las responsabilidades, los colaboradores, que son otras clases que ayudan a conseguir cumplir a esta con sus responsabilidades.

Características:

- Son un puente de comunicación entre diferentes participantes.
- Principales desventajas: lentitud y roces.
- Se recomienda un grupo de trabajo con representantes de las distintas partes.
- Tamaño recomendable de cinco a seis personas: variedad de estilos y no demasiadas divagaciones.
- Recomendación de equipo: 1 o 2 usuarios, 2 analistas, 1 diseñador y 1 moderador.
- Permite ver las clases como algo más que repositorio de datos, sino conocer el comportamiento de cada una en un alto nivel.
- La lluvia de ideas es una buena práctica para sugerir cómo rellenar las tarjetas CRC:

    1) Todas las ideas son buenas, no censura.
    2) Pensar rápido, la meditación después.
    3) Cada miembro debe tener un turno, sin presiones.
    4) Aligerar la situación, pausas para los roces.

Tabla 7: Tarjeta CRC.

Fuente: Elaboración propia.

Clase: Index	
Responsabilidades	**Colaboradores**
Se encarga de verificar las condiciones para que la imagen sea almacenada.	ModelController
Se encarga de todo el proceso de entrenamiento del modelo a utilizar.	DeepLearningController
Se encarga de validar si la imagen ha sido almacenada correctamente.	ModelRepository
Se encarga de definir los atributos necesarios para el almacenamiento de la imagen.	EntityModel

2..3.5 Arquitectura del software

El patrón modelo–vista–modelo de vista (en inglés, model–view–viewmodel, abreviado MVVM), es un patrón de arquitectura de software concebido por John Gossman en el año 2005. Este patrón junto a otros más conocidos como MVC o MVP tiene por objetivo simplificar las tareas de desarrollo y mantenimiento del software escrito con estos a través de la división de ocupaciones. A partir de lo planteado anteriormente, se refiere en (Kouraklis 2016) lo siguiente:

1) **Modelo:** Representa la capa de datos y/o la lógica de negocio, también denominado como el objeto del dominio. El modelo contiene la información, pero nunca las acciones o servicios que la manipulan. En ningún caso tiene dependencia alguna con la vista.
2) **View:** La vista es la parte encargada de la parte visual de la apk, no teniéndose que ocupar en ningún momento en el manejo de datos. En MVVM la vista tiene un rol activo, esto significa que en algún momento la vista recibirá u operará algún evento y, tendrá que comunicarse con el modelo, para poder cumplir el requerimiento.
3) **Modelo de vista:** Se encarga de ser la capa intermedia entre el modelo y la vista, procesando todas las peticiones que tenga la vista hacia el modelo, además de tener que ocuparse de manejar las reglas del negocio, la comunicación con aplicaciones externas o consumir datos desde alguna fuente.

A continuación, la Ilustración 3 muestra el funcionamiento de la arquitectura MVVM:

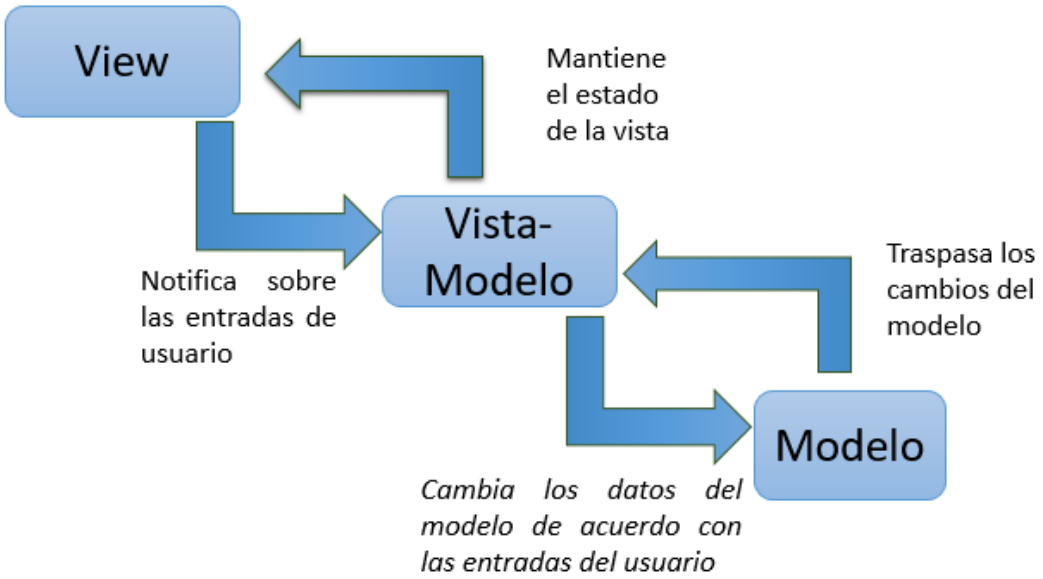


Ilustración 3: *Funcionamiento de la arquitectura MVVM (Kouraklis 2016).*

2.3.6 Modelo de datos

Un modelo de datos es un lenguaje orientado a hablar de una base de datos. Típicamente un modelo de datos permite describir como se explica en (Prieto 2012):

- Las estructuras de datos de la base: El tipo de los datos que hay en la base y la forma en que se relacionan.
- Las restricciones de integridad: Un conjunto de condiciones que deben cumplir los datos para reflejar la realidad deseada.
- Operaciones de manipulación de los datos: típicamente, operaciones de agregado, borrado, modificación y recuperación de los datos de la base.

Otro enfoque es pensar que un modelo de datos permite describir los elementos de la realidad que intervienen en un problema dado y la forma en que se relacionan esos elementos entre sí. No hay que perder de vista que una base de datos siempre está orientada a resolver un problema determinado, por lo que los dos enfoques propuestos son necesarios en cualquier desarrollo de software.

Por cuestiones de seguridad, el modelo de datos de la presente solución, no se puede mostrar ya que es clasificado.

2.3.7 Modelo de despliegue

Un modelo de despliegue como se detalla (Sarmiento 2013), consiste en una representación estructural de la arquitectura del sistema desde el punto de vista de la distribución de los artefactos del software en los destinos de despliegue; definiendo a los artefactos como representaciones de elementos concretos en el mundo físico que son el resultado de un proceso de desarrollo. A continuación, los usos que tiene este modelo:

- **Sistemas empotrados:** Un sistema empotrado es una colección de hardware con una gran cantidad de software que interactúa con el mundo físico.
- **Sistemas cliente-servidor:** Los sistemas Cliente-Servidor son un extremo del espectro de los sistemas distribuidos y requieren tomar decisiones sobre la conectividad de red de los clientes a los servidores y sobre la distribución física de los componentes software del sistema a través de nodos.

- **Sistemas completamente distribuidos:** En el otro extremo se encuentra aquellos sistemas que son ampliamente o totalmente distribuidos y que normalmente incluyen varios niveles de servidores.

A continuación, se muestra el diagrama de despliegue propuesto:

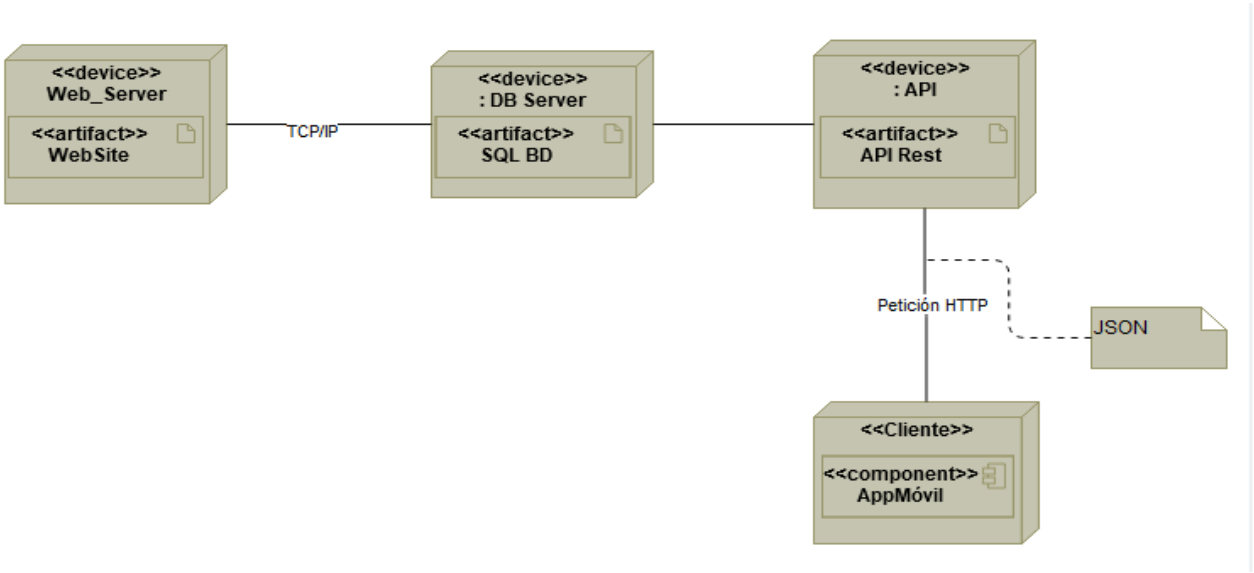


Ilustración 4: *Diagrama de despliegue (Elaboración propia).*

Los elementos que componen este diagrama de despliegue se describen de manera siguiente:

- **Nodos:** Elementos de procesamiento con al menos un procesador, memoria, y otros dispositivos.
- **Conectores**: Expresan el tipo de conector o protocolo utilizado entre el resto de los elementos del modelo.
- **Web_Server:** Este servidor contiene como artefacto al sitio web del cual se tomará la información para la apk, realizando la comunicación por el protocolo TCP/IP.
- **DB Server:** Servidor que almacena la base de datos que contiene la información del sistema, y se comunica con el servidor de aplicaciones de dicho sistema.
- **API:** En este nodo se guardan todas las direcciones de los recursos a manejar, mediante los servicios REST.
- **Cliente:** en este se instala la aplicación desde la cual se consumirán y se brindarán los servicios REST a partir de los recursos que posee la API REST.

Conclusiones del capítulo

Después de realizado el análisis y diseño de la propuesta de solución y haber generado los diferentes artefactos que aplican para la metodología XP, se puede concluir lo siguiente:

1. El estilo arquitectónico de la apk, evidencia que la solución propuesta cuenta con un alto grado de resistencia ante posibles modificaciones.
2. La generación de todos los artefactos requeridos por el modelo de desarrollo, documentan la solución propuesta, lo cual facilita su posterior mantenimiento (actualización o adición de funcionalidades).
3. Se abordaron los principales conceptos del dominio del problema lo que permitie un mejor entendimiento de la apk del sitio oficial de Gobierno y Presidencia de Cuba.
4. Con el análisis de las características del sistema se definen los requisitos funcionales y no funcionales que permiten desarrollar las distintas funcionalidades que debe presentar el sistema para solucionar las necesidades del cliente.

CAPÍTULO 3: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

Asegurar que la apk para el sitio oficial de la Presidencia y Gobierno de Cuba satisface los requerimientos del usuario y cumple con las especificaciones iniciales a través de un conjunto de procedimientos, actividades, técnicas y herramientas es el objetivo fundamental del capítulo. Así mismo, lo es aplicar técnicas específicas: las pruebas y las revisiones. Estas, permiten detectar y corregir el máximo número de errores en el sistema, antes de su entrega al cliente y, validar la investigación propuesta. Se abordan además en el capítulo, los estándares de codificación, los cuales permiten prácticas de programación seguras, confiables, comprobables y mantenibles.

3.1 Estándares de codificación

Algunos lo llaman coding by convention (codificación por convención) o simplemente Coding Standards (estándares de codificación). En (Desarrolladores de Android, 2021) lo definen como un paradigma de la programación que busca reducir el número de decisiones que el desarrollador tiene que tomar al momento de escribir el código. Todo lo anterior viene marcado por el mantenimiento del código, facilidad con que el sistema de software puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores, o mejorar el rendimiento.

3.1.1 Archivos

El nombre de los archivos fuentes escritos con Kotlin, deben escribir en formato CamelCase (y más específicamente PascalCase, que significa que la primera letra de cada una de las palabras se escriben en mayúscula) con la extensión kt. Además, todos los archivos fuentes deben estar codificados como UTF-8.

Si el archivo contiene una sola clase, su nombre debe ser el mismo que el nombre de la clase. Si un archivo contiene varias clases, o solo declaraciones de nivel superior, el nombre del archivo debe describir su contenido. Algunos ejemplos:

```
// MyClass.kt
class MyClass { }

// Bar.kt
class Bar { }
fun Runnable.toBar(): Bar = // …

// Map.kt
fun <T, O> Set<T>.map(func: (T) -> O)
fun <T, O> List<T>.map(func: (T) -> O
```

Ilustración 5: *Ilustración 6: Ejemplo de cómo nombrar archivos en kotlin.*

*Fuente: (Desarrolladores de Android, 2021)**.***

3.1.2 Estructura

Un archivo .kt comprende lo siguiente, en este orden y con una línea en blanco separando cada una de estas secciones:

Derechos de autor o encabezado de licencia (opcional)

1. Anotaciones a nivel de archivo
2. Declaración del paquete
3. Declaraciones de importación
4. Declaraciones de nivel superior

```
/*
 * Copyright 2017 Google, Inc.
 *
 * ...
 */
```

Ilustración 6: *Ejemplo de estructura en kotlin.*

*Fuente: (Desarrolladores de Android, 2021)**.***

3.1.3 Bloques con llaves

En los bloques when y if no se requieren llaves cuando no se bifurcan con else if/else y cuando ocupan una sola línea. De lo contrario, se requieren llaves para cualquiera de las sentencias. Las llaves siguen el siguiente estilo:

1. No hay salto de línea antes de la llave de apertura.
2. Salto de línea después de la llave de apertura, incluso en los bloques vacíos.
3. Salto de línea antes de la llave de cierre.
4. Salto de línea después de la llave de cierre solo si esa llave termina una declaración o termina el cuerpo de una función, un constructor o una clase con nombre.

La Ilustración 8 muestra ejemplo claro de este estándar de codificación:

```
return Runnable {
    while (condition()) {
        foo()
    }
}

return object : MyClass() {
    override fun foo() {
        if (condition()) {
            try {
                something()
            } catch (e: ProblemExcept
                recover()
            }
        } else if (otherCondition())
            somethingElse()
        } else {
            lastThing()
        }
    }
}
```

Ilustración 7: *Ejemplo del estándar de codificación de bloques con llaves en kotlin.*

Fuente: (Desarrolladores de Android, 2021).

Una excepción de esto es la clase *enum*, que cuando no tiene funciones ni constantes, opcionalmente puede presentarse en una sola línea:

```
enum class Respuesta { SI, NO, QUIZA
```

Ilustración 8: *Excepción del estándar de codificación de bloques con llaves en kotlin.*

Fuente: (Desarrolladores de Android, 2021).

Por su parte, un condicional *if/else* puede omitir llaves solo si la expresión completa se ajusta a una línea:

```
val value = if (string.isEmpty()) 0 e
```

Ilustración 9: *Ejemplo de omisión de llaves en expresiones en lenguaje kotlin.*

Fuente: (Desarrolladores de Android, 2021).

3.1.4 Identación y líneas

Con cada nuevo bloque la sangría aumenta en cuatro espacios, y cuando el bloque finaliza la sangría vuelve al nivel previo. Esto aplica tanto al código como a los comentarios. El código tiene un límite de ancho de 100 caracteres. Cuando una línea excede este límite, debe ajustarse teniendo en cuenta que la directiva principal para el ajuste de líneas, recomienda romper la línea en un nivel sintáctico superior.

Hay que tener en cuenta que el objetivo principal del ajuste de línea es tener un código claro, no necesariamente un código que se ajuste al menor número de líneas.

- Otras directrices respecto al ajuste de líneas:
- Cuando se rompe una línea en un operador sin asignación, la ruptura aparece antes del símbolo. Esto también se aplica al punto (.) y a los dos puntos (::) que hacen referencia a un miembro.
- Cuando se rompe una línea en un operador de asignación, la ruptura viene después del símbolo.
- Un nombre de método o constructor siempre permanece unida a la palabra que la precede.
- Cuando los parámetros de una función no caben en una sola línea, cada uno debe aparecer en su propia línea con una única sangría. En estos casos, el paréntesis de cierre y el tipo de retorno se colocan en su propia línea sin sangría (Ilustración 11).
- Cuando una función contiene una sola expresión, se puede representar como función de una expresión en una línea (Ilustración 12).
- La única vez que una función de expresión debe ajustarse a varias líneas es cuando abre un bloque (Ilustración 13).

```
fun <T> Iterable<T>.joinToString(
    separator: CharSequence = ", ",
    prefix: CharSequence = "",
    postfix: CharSequence = ""
): String {
    // …
}
```

Ilustración 10: *Ejemplo de codificación cuando los parámetros de una función no caben en una sola línea.*

*Fuente: (Desarrolladores de Android, 2021)**.***

```
override fun toString(): String {
    return "Hey"
}

override fun toString(): String = "Hey
```

Ilustración 11: *Ejemplo de representación de una función con una sola expresión.*

*Fuente: (Desarrolladores de Android, 2021)**.***

```
fun main() = runBlocking {
  // …
}
```

Ilustración 12: *Ejemplo de representación de una función ajustada a varias líneas.*

*Fuente: (Desarrolladores de Android, 2021)**.***

3.1.5 Espacios en blanco

Se debe deja una línea en blanco (nunca se recomienda más de una) entre miembros consecutivos de una clase (propiedades, constructores, funciones, clases anidadas, entre otras) con dos excepciones: entre dos propiedades consecutivas (espacio opcional para crear agrupaciones lógicas de propiedades asociadas) y entre las constantes de una clase *enum*.

También se debe dejar una línea en blanco entre declaraciones según sea necesario para organizar el código en subsecciones lógicas (y opcionalmente antes de la primera declaración en una función, antes del primer miembro de una clase o después del último miembro de una clase).

A las clases *enum* se les aplica estas reglas como a cualquier clase, teniendo en cuenta que cuando las constantes se colocan en líneas separadas no se requiere una línea en blanco entre ellas, excepto en el caso de que definan un cuerpo o bloque (Ilustración 14).

En cuanto a los espacios en blanco horizontales, deben aparecer separando cualquier palabra reservada *if, for,* o *catch* del paréntesis abierto que le sigue en esa línea (Ilustración 15).

```
enum class Answer {
    YES,
    NO,

    MAYBE {
        override fun toString() = """
    }
}
```

Ilustración 13: *Ejemplo de representación de la clase enum.*

*Fuente: (Desarrolladores de Android, 2021)**.***

```
// MAL
for(i in 0..1) {
}

// BIEN
for (i in 0..1) {
}
```

Ilustración 14: *Ejemplo del uso de los espacios en blancos horizontales.*

*Fuente: (Desarrolladores de Android, 2021)**.***

Al igual que en las otras guías de estilo de programación, los temas incluidos no solo abarcan problemas estéticos de formato, sino también otros tipos de convenciones o estándares de codificación. Sin embargo, esta investigación se enfoca, principalmente, en las reglas estrictas que se siguen a nivel

universal y se evitan consejos que no se puedan aplicar claramente (ya sea mediante personas o herramientas).

3.2 Pruebas de software

Según Pressman, las pruebas de software (en inglés software testing) son de vital importancia para comprobar que la aplicación desarrollada no presente problemas de ejecución y se encuentre libre de errores, ya que durante el proceso de desarrollo de software es frecuente que los desarrolladores, al programar las diferentes funcionalidades, obvien algunas posibilidades que pueden variar el resultado durante la ejecución del código; por tanto, estas pruebas constituyen la vía a través de la cual se asegura que el producto desarrollado está listo para ser entregado a los usuarios finales. A continuación, se realizan las pruebas recomendadas por la arquitectura utilizada y explicada anteriormente (XP).

3.2.1 Pruebas unitarias

Una prueba unitaria tiene como objetivo la lógica de un determinado método en una determinada clase en Android, Kotlin o Java. La verificación del código es para probar si el método puede funcionar normalmente. Android Studio, está diseñado para simplificar la ejecución de pruebas. Especifica las operaciones a llevar a cabo durante el proceso de pruebas y se encarga a su vez de definirlas y planificarlas.

En Android Studio al crear un proyecto nuevo ya viene implementada la configuración por defecto para realizar pruebas unitarias, para esto es necesario realizar los siguientes pasos:

1. En el archivo build.gradle dentro del directorio app:

1.1 Agregar las dependencias necesarias para ejecutar las pruebas utilizando el marco de desarrollo Junit:

```
testImplementation 'junit:junit:4.12'
androidTestImplementation 'com.android.support:support-annotations:28.0.0'
androidTestImplementation 'com.android.support.test:runner:1.0.2'
```

1.2 Definir el identificador de las pruebas a ejecutar:

testApplicationId **"com.simplemobiletools.presidenciagob.pro.test"**

1.3 Especificar la clase encargada de ejecutar las pruebas:

testInstrumentationRunner "androidx.test.runner.testInstrumentationRunner"

1.4 Establecer los directorios donde se almacenarán los reportes y los resultados:

testOptions {

reportDir = "$project.buildDir/**results/report"**

```
resultsDir = "$project.buildDir/results"
}
```

2. En app dentro del directorio java debe existir un paquete con el mismo nombre que se especificó en el paso 1.4 (si no se encuentra es necesario crearlo) crear las clases para realizar las pruebas, el nombre de estas siempre debe contener Test al final.
3. Para programar las pruebas solo es necesario que los métodos a testear contengan la anotación @Test y que comiencen con el nombre test:

```
    class NoticiasTest {
  @Test
  fun testGetSharedPreferences() {
      assertNull(null)
  }
}
```

4. Para ejecutar y compilar las pruebas solo es necesario seleccionar entre las configuraciones de ejecución la del test como se muestra en la Ilustración 16 y ejecutar la prueba presionando Run ( ).

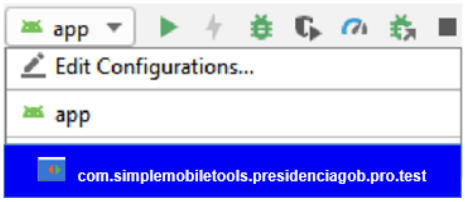


Ilustración 15: *Configuración a seleccionar para ejecutar Pruebas.*

Aplicando los casos de pruebas generados previamente e identificando los posibles fallos producidos al comparar los resultados esperados con los obtenidos. Es necesario ejecutar un total de 11 casos de pruebas para cada uno de los requisitos funcionales definidos en la aplicación propuesta. Ejemplo, el test NoticiaTest.kt, como se muestra en la Ilustración 17. Aplicando los casos de pruebas generados e identificando los posibles fallos producidos al comparar los resultados esperados con los obtenidos, es necesarios ejecutar un total de 11 casos de pruebas para cada uno de los requisitos funcionales. Los resultados.

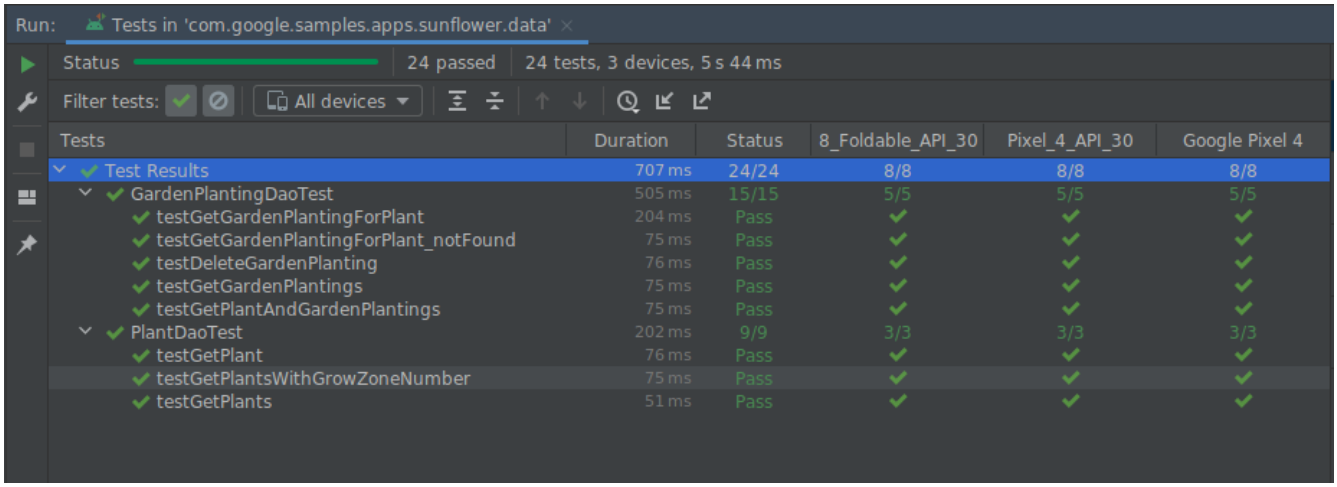


Ilustración 16: *Resultado de la prueba unitaria.*

3.2.2 Pruebas de aceptación

Las pruebas de aceptación (User Acceptance Testing, UAT) pertenecen a las últimas etapas previas a la liberación en firme de versiones nuevas a fin de determinar si cumplen con las necesidades y/o requerimientos de las empresas y sus usuarios. Al finalizar las pruebas automatizadas, que garantizan los requisitos tecnológicos del diseño inicial, se pasa a las pruebas manuales. Dichas pruebas manuales primero son hechas por usuarios internos en entornos intermedios: ambientes típicos de producción donde se verifica que se desempeña el modo necesitado.

Hay tres estrategias comunes para implementar una prueba de aceptación:

1. *Aceptación formal:* Proceso altamente gestionado y, a menudo, una ampliación de las pruebas del sistema.
2. *Aceptación informal o prueba de versión alfa:* No se definen rigurosamente procedimientos para realizar las pruebas. No hay casos de prueba particulares que seguir.
3. *Prueba de versión beta:* Es la menos controlada de las tres estrategias de prueba de aceptación. La cantidad de detalles y datos y el enfoque adoptado dependen totalmente del verificador individual.

En el anexo 10 se encuentra la carta de aceptación de Servicios Internos del Palacio de la Revolución, firmado por la Jefa de Departamento de Desarrollo y Mantenimiento de Software.

3.2.3 Pruebas de usabilidad

Este tipo de prueba, es realizada por parte del usuario como técnica para evaluar el p0roducto mediante pruebas con los mismos. Las pruebas de usabilidad consisten en seleccionar a un grupo de usuarios de una apk y solicitarles que lleve a cabo las tareas para las cuales fue diseñado. En este caso, las pruebas son realizadas de acuerdo a la confección de una lista de especificaciones de usabilidad (lista de chequeo).

Como se observa en la Tabla 8, de los 144 parámetros de la lista de chequeo, solo proceden 83. En una primera iteración se evaluaron como correctos 66, identificando 17 no conformidades, para un 80% de usabilidad. En una segunda iteración se solucionaron 15 de las no conformidades, obteniendo un sistema 97,6% usable. En la tercera iteración fueron resueltas las no conformidades restantes, corroborando en una última iteración que el sistema.

Tabla 8: *Valores de la lista de chequeo.*

Categoría de los indicadores	Indicadores	Proceden	Correctos	Incorrectos
Visibilidad del sistema	17	14	13	1
Libertad y control por parte del usuario	29	14	11	3
Consistencia y estándares	18	8	8	0
Estética y diseño minimalista	8	5	4	1
Prevención de errores	11	7	1	6
Ayuda a los usuarios a reconocer, diagnosticar y recuperarse de los errores	11	3	1	2
Ayuda y documentación	6	4	3	1
Flexibilidad y eficiencia	6	4	3	1
Lenguaje común entre sistema y usuario	11	6	6	0
Total	**144**	**83**	**66**	**17**

3.3 Interfaces de la propuesta de solución

Una vez desarrollado el sitio oficial para la Presidencia y Gobierno de Cuba, es posible visualizar las pantallas principales del mismo, donde se observa: el ícono de la apk, las noticias como actividad fundamental, así como el contacto para dirigirse a Atención al Ciudadano, y la red social twitter, que permite compartir las noticias en el perfil del usuario.

Ilustración 17: *Ícono de la aplicación de la Presidencia y Gobierno de Cuba.*

Ilustración 18: Listado de noticias de la apk Presidencia y Gobierno de Cuba.

Ilustración 19: *Menú que incluye datos de contacto y red social twitter.*

Cuánto se puede hacer desde el presente para garantizar el futuro; desde los pequeños espacios para impulsar al país; desde las empresas para potenciar y respaldar el desarrollo de los municipios donde se encuentran enclavadas. Fueron esas algunas de las principales interrogantes que marcaron el rumbo de los debates que se sucedieron en la tarde de este sábado, durante la Asamblea de Balance partidista del capitalino municipio de Boyeros, que estuvo presidida por el Primer Secretario del Comité Central del Partido Comunista y Presidente de la República, Miguel Díaz-Canel Bermúdez. Tomando como punto de partida el minucioso informe presentado por la Primera Secretaria del Comité Municipal del Partido, Susel Lameré García, las intervenciones evaluaron de manera crítica y objetiva el trabajo desarrollado por la Organización partidista. Es

Ilustración 20: *Muestra de las noticias al tocar "Leer Más".*

CONCLUSIONES DEL CAPÍTULO

Luego de la implementación y validación de la aplicación propuesta se arriba a las siguientes conclusiones:

- Al aplicar los estándares de codificación se facilita la comprensión y legibilidad del código fuente, permitiendo desarrollar la apk, en un menor tiempo y garantizando la facilidad y calidad de un futuro mantenimiento.
- El proceso de validación de la propuesta de solución, a través de la estrategia de pruebas especificada, arroja como resultado que la apk implementada, responde a los requerimientos definidos por el cliente.

CONCLUSIONES FINALES

Con la investigación propuesta se logró dar cumplimiento a los objetivos específicos planteados inicialmente, lo cual permite arribar a las siguientes conclusiones:

- Con el análisis del estudio realizado de los referentes teóricos y de las diferentes herramientas y tendencias relacionadas con la apk se comprueba que existen una serie de apk que se utilizan para la gestión de información, seleccionándose así las funcionalidades necesarias para la propuesta solución de acuerdo a las necesidades requeridas por el cliente.
- La selección de las herramientas, lenguajes y tecnología utilizados en la apk, permitió representar las etapas que se debía ejecutar para la implementación de la misma.
- Se alcanza el empleo de buenas prácticas en el desarrollo de la apk con la definición de la arquitectura, los estándares de codificación y los patrones de diseño.
- Con las pruebas realizadas a la apk Presidencia y Gobierno de Cuba, se garantiza la corrección de las no conformidades detectadas, demostrándose que el sistema constituye una solución funcional.

RECOMENDACIONES

Para el desarrollo de futuras investigaciones relacionadas con la presente, se propone:

- Se recomienda realizar actualizaciones a la apk según los nuevos estándares y diseños a nivel mundial.
- Hacer un seguimiento continuo a la apk con el fin de ir incorporando nuevas necesidades que permitan cubrir los nuevos requerimientos que puedan ir surgiendo.
- Publicar las nuevas actualizaciones de la apk en el distribuidor de aplicaciones cubano Apklis.

REFERENCIAS BIBLIOGRÁFICAS

About SQLite. [en línea], [sin fecha]. [Consulta: 12 julio 2021]. Disponible en: https://www.sqlite.org/about.html.

Android Developers. *Android Developers* [en línea], [sin fecha]. [Consulta: 12 julio 2021]. Disponible en: https://developer.android.com/.

ÁVILA BARRIOS., D., 2014. *El uso de las TICs en el entorno de la nueva gestión pública mexicana*. S.l.: s.n.

CALERO, C., MUÑOZ, C.C. y VELTHUIS, M.G.P., 2010. *Calidad del producto y proceso software*. S.l.: Editorial Ra-Ma. ISBN 978-84-7897-961-5.

Disponible en Apklis la aplicación Participación Popular que facilitará denuncias ciudadanas. *Granma.cu* [en línea], [sin fecha]. [Consulta: 9 julio 2021]. Disponible en: http://www.granma.cu/cuba/2019-09-15/disponible-en-apklis-la-aplicacion-participacion-popular-que-facilitara-denuncias-ciudadanas-15-09-2019-09-09-26.

FOWLER, M., 2018. *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. S.l.: Addison-Wesley Professional. ISBN 978-0-13-486512-6.

FUENTES, J.R.L., 2015a. *Desarrollo de Software Ágil: Extremme Programming y Scrum. 2ª Edición*. S.l.: IT Campus Academy. ISBN 978-1-5196-2014-9.

FUENTES, J.R.L., 2015b. *Desarrollo de Software Ágil: Extremme Programming y Scrum. 2ª Edición*. S.l.: IT Campus Academy. ISBN 978-1-5196-2014-9.

GABRIEL, E., 2015. Metodologías de desarrollo de software. , pp. 117.

GABRIEL, E., 2016. Metodologías de desarrollo de software. , pp. 117.

GARRIDO, J.M., SCHLESINGER, R. y HOGANSON, K., 2011. *Principles of Modern Operating Systems*. S.l.: Jones & Bartlett Publishers. ISBN 978-1-4496-2635-8.

GIL-GARCIA, J.R., 2012. *Enacting Electronic Government Success: An Integrative Study of Government-wide Websites, Organizational Capabilities, and Institutions*. S.l.: Springer Science & Business Media. ISBN 978-1-4614-2015-6.

GIULIANELLI, D.A., RODRÍGUEZ, R.A., VERA, P.M., MARKO, I.B., TRIGUEROS, A. y LARROSA, M.I., [sin fecha]. Análisis de Gobernabilidad Electrónica: Relevamiento de e-Servicios, e-Democracia, e-Transparencia y Comunicación en Sitios Web Municipales. , pp. 12.

GIULIANELLI, D.A., RODRÍGUEZ, R.A., VERA, P.M., TRIGUEROS, A. y MARKO, I., 2013. Los sitios web gubernamentales como facilitadores del gobierno electrónico. En: journalAbbreviation: Análisis de la evolución de los sitios web municipales del Conurbano Bonaerense, *XV Workshop de Investigadores en Ciencias de la Computación* [en línea]. S.l.: s.n., [Consulta: 8 julio 2021]. Disponible en: http://sedici.unlp.edu.ar/handle/10915/27209.

GOMIS, J.M.F., 2014. *Mobile-Learning: Estrategias para el uso de aplicaciones, smartphones y tablets en educación*. S.l.: Ana López Canosa. ISBN 978-84-617-1612-8.

GUADALUPE, G.M.M., 2015. USOS Y TIPOS DE APLICACIONES MÓVILES. , pp. 24.

HERNÁNDEZ BEJARANO, M. y BAQUERO REY, L.E., 2020. *Ciclo de vida de desarrollo ágil de software seguro*. S.l.: Editorial Los Libertadores. ISBN 978-958-54-7844-2.

Ideal Modeling & Diagramming Tool for Agile Team Collaboration. [en línea], [sin fecha]. [Consulta: 13 julio 2021]. Disponible en: https://www.visual-paradigm.com/.

JOVER, J.N., 2016. La ciencia y la tecnología como procesos sociales. Lo que la educación científica no debería olvidar, pp. 23.

KENDALL, K.E. y KENDALL, J.E., 2005. *Análisis y diseño de sistemas*. S.l.: Pearson Educación. ISBN 978-970-26-0577-5.

KOURAKLIS, J., 2016. *MVVM in Delphi: Architecting and Building Model View ViewModel Applications*. S.l.: Apress. ISBN 978-1-4842-2213-3.

L, C., Barbara, 2011. *Mobile Technology Consumption: Opportunities and Challenges: Opportunities and Challenges*. S.l.: IGI Global. ISBN 978-1-61350-151-1.

LLODRÀ, B., 2009. Adoptando el uso de las TIC en la Administración Pública, Infogedas, núm. 5. [en línea], [Consulta: 16 septiembre 2021]. Disponible en: : http://www.infogedas.es/sectorpublico/index.php?id=108.

Local e-Governance | United Nations Educational, Scientific and Cultural Organization. [en línea], 2005. [Consulta: 8 julio 2021]. Disponible en: http://www.unesco.org/new/en/natural-sciences/priority-areas/sids/sids-conferences/mauritius-conference-2005/themes/information-knowledge/transport-communication/local-e-governance/.

LUZ, C.G.M., 2018. *EDUCACIÓN Y TECNOLOGÍA: ESTRATEGIAS DIDÁCTICAS PARA LA INTEGRACIÓN DE LAS TIC*. S.l.: Editorial UNED. ISBN 978-84-362-7328-1.

Máster en Desarrollo de Aplicaciones Android - Arquitectura de Android. [en línea], [sin fecha]. [Consulta: 12 julio 2021]. Disponible en: http://www.androidcurso.com/index.php/recursos/31-unidad-1-vision-general-y-entorno-de-desarrollo/99-arquitectura-de-android.

Metodología Ágil Programación Extrema XP. , 2015. pp. 146.

MINISTERIO DE ECONOMÍA Y PLANIFICACIÓN (MEP), 2017. *Plan Nacional de Desarrollo Económico y Social hasta el 2030. Gaceta Oficial de la República de Cuba. 2017.* [en línea]. 4 julio 2017. S.l.: s.n. 45. Disponible en: http://www.gacetaoficial.gob.cu.

Mobile & Tablet Operating System Market Share Cuba. *StatCounter Global Stats* [en línea], [sin fecha]. [Consulta: 9 julio 2021]. Disponible en: https://gs.statcounter.com/os-market-share/mobile-tablet/cuba/.

NIÑO, J., 2011. *Introducción a los sistemas operativos (Sistemas operativos monopuesto)*. S.l.: Editex. ISBN 978-84-9003-048-6.

PEÑALVO, F.J.G., HOLGADO, A.G. y INGELMO, A.V., 2020. INGENIERÍA DE SOFTWARE I. , pp. 26.

POLANCO, K.M. y TAIBO, J.L.B., 2011. "Android" el sistema operativo de Google para dispositivos móviles. *Negotium: revista de ciencias gerenciales* [en línea], vol. 7, no. 19, pp. 79-96. [Consulta: 12 julio 2021]. ISSN 1856-1810. Disponible en: https://dialnet.unirioja.es/servlet/articulo?codigo=7165367.

Política Integral para el perfeccionamiento de la informatización de la sociedad en Cuba. [en línea], 2017. julio 2017. S.l.: s.n. [Consulta: 28 junio 2021]. Disponible en: https://www.mincom.gob.cu/.

PRIETO, I., EGUSQUIZA, A., DELGADO, F.J. y MARTÍNEZ, R., 2012. CityGML como modelo de datos para la representación, intercambio y visualización de información sobre el patrimonio arquitectónico. *Virtual Archaeology Review* [en línea], vol. 3, no. 5, pp. 48. [Consulta: 20 octubre 2021]. ISSN 1989-9947. DOI 10.4995/var.2012.4504. Disponible en: http://polipapers.upv.es/index.php/var/article/view/4504.

Projects. *opensource.google* [en línea], [sin fecha]. [Consulta: 12 julio 2021]. Disponible en: https://opensource.google/projects/.

RAMOS, D., NORIEGA, R., LAÍNEZ, J.R. y DURANGO, A., 2017. *Curso de Ingeniería de Software: 2ª Edición*. S.l.: IT Campus Academy. ISBN 978-1-5441-3253-2.

REDHAT, 2021. ¿Qué es una API? [en línea]. [Consulta: 24 septiembre 2021]. Disponible en: https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces.

SARMIENTO, F., Johana, 2013. Visión General de los Diagramas de Despliegue. *UML* [en línea]. [Consulta: 20 octubre 2021]. Disponible en: http://umldiagramadespliegue.blogspot.com/.

TORO, J.A.Q. y RENDÓN, L.E.O., 2016. *Metodología de desarrollo orientado a arquitectura de software*. S.l.: Universidad Tecnológica de Pereira. Facultad de Ingenierías Eléctrica, Electrónica, Física y Ciencias de la Computación. Ingeniería de Sistemas y Computación.

UNESCO, 2017. Comunicación e Información. Organización de las Naciones Unidas para la Educación la Ciencia y la Cultura. *UNESCO* [en línea]. [Consulta: 13 julio 2021]. Disponible en: https://en.unesco.org/.

YOUNG, R.R., 2001. *Effective requirements practices* [en línea]. S.l.: Boston, Ma. : Addison-Wesley. [Consulta: 18 octubre 2021]. ISBN 978-0-201-70912-4. Disponible en: http://archive.org/details/unset0000unse_g5k2.

ANEXOS

Anexo 1: HU _ Cargar imagen

Historia de usuario	
Número: HU_1	**Nombre: Cargar imagen.**
Programador responsable: Enma N. Piquero Moskaquiel	Iteración asignada:
Prioridad: Media	
Descripción: Cuando el usuario accede a una de las secciones disponibles en la apk, se muestran imágenes referenciales, por lo que se hace necesario poder cargar dichos elementos.	
Observaciones: La imagen debe estar en el formato JPG, PNG.	

Anexo 2: HU _ Almacenar imágenes obtenidas.

Historia de usuario	
Número: **HU_2**	**Nombre:** Almacenar imágenes obtenidas.
Programador responsable: Enma N. Piquero Moskaquiel	Iteración asignada:3
Prioridad: Media	
Descripción: Luego de encontrarse el usuario offline, las imágenes cargadas deben quedar almacenadas en la aplicación.	
Observaciones: La imagen debe estar en el formato JPG, PNG.	

Anexo 3: HU _ Mostrar contenido

Historia de usuario	
Número: **HU_3**	**Nombre: Mostrar contenido.**
Programador responsable: Enma N. Piquero Moskaquiel	Iteración asignada: 1
Prioridad: Alta	
Descripción: La aplicación debe ser capaz de mostrar el contenido, una vez que el usuario se conecte a la red wifi o datos móviles.	
Observaciones: El formato de letras debe tener uniformidad, para ser legible frente al usuario.	

Anexo 4: HU _ Mostrar enlaces

Historia de usuario	
Número: **HU_4**	**Nombre:** Mostrar enlaces.
Programador responsable: Enma N. Piquero Moskaquiel	Iteración asignada: 3
Prioridad: MEDIA	
Descripción: La aplicación muestra enlaces a: • Perfil de redes sociales como Twitter del usuario. • Perfiles en Twitter de los dirigentes del país. • Enlaces a páginas de interés. • Enlaces a páginas web de la prensa del país (los periódicos Granma y Juventud Rebelde, entre otros)	
Observaciones: Para acceder a estos enlaces, el usuario debe estar online, es decir, conectado a una red wifi o por datos móviles.	

Anexo 5: HU_ Realizar búsqueda simple.

Historia de usuario	
Número: **HU_5**	**Nombre:** Realizar búsqueda simple.
Programador responsable: Enma N. Piquero Moskaquiel	Iteración asignada: **4**
Prioridad: Baja	
Descripción: Al entrar en la apk, el usuario debe poder buscar entre las noticias, poniendo datos característicos de la (s) noticias que desea encontrar.	
Observaciones: Para realizar esta operación, el usuario debe estar online, es decir, conectado a una red wifi o por datos móviles.	

Anexo 6: HU_ Realizar búsqueda-

Historia de usuario	
Número: **HU_6**	**Nombre:** Realizar búsqueda avanzada.
Programador responsable: Enma N. Piquero Moskaquiel	Iteración asignada: 4
Prioridad: Baja	
Descripción: La apk debe permitir realizar una búsqueda entre las categorías de las noticias: nacionales, programas priorizados, visitas gubernamentales, internacionales, balances de trabajo, actividad oficial, artículos y títulos honoríficos. Además, permite escoger el año en que desea el usuario buscar una noticia, así como el mes (estas noticias deben ser escogidas entre los últimos cuatro años).	
Observaciones: Para realizar esta operación, el usuario debe estar online, es decir, conectado a una red wifi o por datos móviles.	

Anexo 7: HU_ Mostar descripción del contenido.

Historia de usuario	
Número: HU_9	**Nombre:** Mostrar descripción del contenido.
Programador responsable: Enma N. Piquero Moskaquiel	Iteración asignada: 2
Prioridad: Baja	
Descripción: El contenido presenta breves descripciones, para que el usuario decida si le interesa la noticia o no. De interesarle continuar leyendo la información, debe dar clic, en ¨LEER MÁS¨.	
Observaciones: Para realizar esta operación, el usuario debe estar online, es decir, conectado a una red wifi o por datos móviles.	

Anexo 8: HU_ Mostrar autor(es) de las noticias.

Historia de usuario	
Número: **HU_10**	**Nombre:** Mostrar autor(es) de las noticias.
Programador responsable: Enma N. Piquero Moskaquiel	Iteración asignada: 5
Prioridad: Baja	
Descripción: Cada noticia debe tener el/los autores, con los enlaces a sus perfiles de Twitter.	
Observaciones: Para acceder a los perfiles de los autores de las noticias, el usuario debe estar online, es decir, conectado a una red wifi o por datos móviles.	

Anexo 9: HU_ Mostrar contenido agrupado por etiqueta.

Historia de usuario	
Número: HU_11	**Nombre:** Mostrar contenido agrupado por etiqueta.
Programador responsable: Enma N. Piquero Moskaquiel	Iteración asignada:1
Prioridad: Alta	
Descripción: Permite la agrupación de contenido según su tipo.	
Observaciones:	

Anexo 10: Carta de aceptación de la entidad.

La Habana, 16 de noviembre de 2021
"Año 63 de la Revolución"

Por medio de la presente Unidad Presupuestada Servicios Internos Palacio de la Revolución), nos permitimos notificar la **ACEPTACION** del proyecto Sitio Oficial de la Presidencia y Gobierno de Cuba a llevarse a cabo por Enma Nélida Piquero Moskaquiel.

La culminación de este proyecto se llevó a cabo bajo las condiciones y características estipuladas en la convocatoria.

Saludos cordiales.

María de los Angeles Figueroa Valdés
Jefe del Departamento de Desarrollo y Mtto de Software
Dirección de Infocomunicaciones
Unidad Presupuestada Servicios Internos Palacio de la Revolución

Teléfono: 58592403
Correo: marangel@palacio.cu