UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Propuesta de automatización de entrega de reporte semanal de rendimiento de cortadores, a caporales responsables de grupos, en frentes de corte de caña.

Trabajo de graduación presentado por Erik Geovanni Chávez Cuc
para optar al grado académico de Licenciado en Tecnología de Sistemas Informáticos.

Guatemala,

2022

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

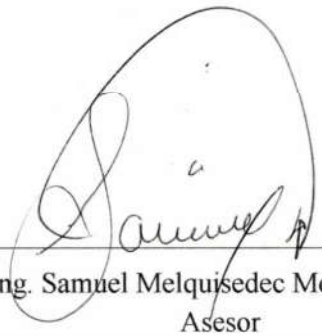
Propuesta de automatización de entrega de reporte semanal de rendimiento de cortadores, a caporales responsables de grupos, en frentes de corte de caña.

Trabajo de graduación presentado por Erik Geovanni Chávez Cuc
para optar al grado académico de Licenciado en Tecnología de Sistemas Informáticos.

Guatemala,

2022

Vo.Bo. :

(f) 

Ing. Samuel Melquisedec Molina Donis
Asesor

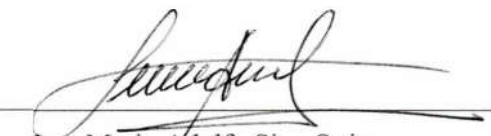
Tribunal Examinador:

(f) 

Ing. Samuel Melquisedec Molina Donis
Asesor

(f) 

Ing. David Humberto Mejicanos Arana
Examinador

(f) 

Ing. Mario Adolfo Sian Quisque
Director

Fecha de aprobación: Guatemala, 15 de diciembre de 2022

ÍNDICE

LISTA DE TABLAS

LISTA DE ILUSTRACIONES

RESUMEN

Tener una satisfacción laboral en el personal de corte es un tema importante en la agroindustria azucarera de Guatemala, y uno de los factores que intervienen en una satisfacción laboral alta, es el tema del salario y los cortadores de caña se ganan ese salario con base en su productividad y se conoce como THD que significa Toneladas Hombre Día. Si los cortadores no están satisfechos con su sueldo pueden detener toda la operación de un ingenio azucarero, es por eso que es sumamente importante que se les pueda mostrar de manera ágil, y eficaz sus rendimientos diarios, actualmente en la empresa azucarera donde se trabajó la presente propuesta, se hace por medio de reportes impresos que toman más de 18 horas en llegar desde la oficina que los imprime hasta las manos de los caporales líderes de los grupos de corte.

Para agilizar y automatizar este proceso, el presente proyecto explica el desarrollo de una aplicación móvil hecha con sistema operativo Android, la cual es de fácil manejo y acceso, tomando en cuenta las características del usuario final y que solo requiera actualizar los datos una vez por semana, almacenarlos en el mismo dispositivo para poder hacer las consultas necesarias sin temor a perder la conectividad al estar en campo agrícola en el sur del Guatemala.

La finalidad del proyecto es la de automatizar y reducir costos, al dejar de lado la necesidad de imprimir reportes semanales a más 30 personas.

I. INTRODUCCIÓN

Las aplicaciones móviles en la actualidad son muy utilizadas gracias a las facilidades de acceso a internet existentes, así como a los avances tecnológicos en *smartphones*, estos cuentan con sistemas operativos que facilitan desarrollar aplicaciones gratuitas que se pueden instalar en un dispositivo móvil sin mayores problemas, razón por la cual se optó por desarrollar una aplicación móvil que automatice la entrega de reportes en campo agrícola a los caporales o jefes de grupo de entre 40 a 60 cortadores de caña de azúcar, en la boca costa del pacífico de la república de Guatemala. Dichos cortadores pertenecen a una empresa dedicada a la producción de azúcar, empresa que desea mantener una satisfacción laboral alta y para ello es necesario que los cortadores puedan revisar su productividad de forma ágil y práctica.

El proceso actual es lento y rudimentario dependiendo directamente de la impresión de reportes y trasladándolos físicamente hasta donde se encuentren ejerciendo su labor de corte de caña, este proceso consume tiempo y recursos para su ejecución, viendo esta oportunidad de mejora se establecieron objetivos generales y específicos para desarrollar una aplicación móvil que agilice este proceso y su justificación.

Como siguiente paso se desarrolla el marco teórico que abarca la redacción de las bases teóricas que soportan el desarrollo de la aplicación móvil, luego su marco metodológico, donde se estipulan los requerimientos funcionales y de soporte, ilustrando por medio de diagramas sus funcionalidades, para finalizar con los resultados obtenidos y las conclusiones a las que se llegó al finalizar todo el proceso.

II. OBJETIVOS

A. Objetivo general

1. Propuesta de automatización de generación de reporte semanal de rendimiento de cortadores a caporales responsables de grupos en frentes de corte de caña, por medio de aplicación móvil basada en sistema Android.

B. Objetivos específicos

1. Reducir en al menos el 50% del tiempo actual del proceso de generación de reportes semanales, de rendimiento de cortadores de caña utilizando tecnología móvil con envío automático y consultas en tiempo real por medio de dispositivos móviles.

2. Reducir al 100% el uso de papel en el proceso de generación de reportes de rendimiento de cortadores en frentes de corte de caña.

3. Diseñar y crear un prototipo de una aplicación móvil para sistema operativo Android, usando el entorno de desarrollo Android Studio, con el lenguaje de programación java, para mostrar la productividad de cada cortador en la semana anterior.

4. Proveer a los caporales el 100% de la información procesada en oficinas sobre el rendimiento de los grupos de cortadores de cada frente de corte de caña por medio de una aplicación Android en un dispositivo móvil.

III. JUSTIFICACIÓN

La satisfacción del personal es un tema importante en las empresas hoy en día y no es más que la actitud general que tiene un individuo hacia su empleo, de tal manera se puede decir que, la satisfacción laboral se puede definir como el resultado conductual que presenta un trabajador hacia su empleo, por factores como el tipo de trabajo, el jefe o supervisor, compañeros, salario, ascenso, condiciones del sitio de trabajo, etc.

Tomando en cuenta la temática de proveer satisfacción al personal de corte de una empresa agroindustrial, sobre todo en el tema del salario por productividad, el presente trabajo responde a la necesidad de agilizar la entrega de información con respecto a dicha productividad de los cortadores de caña, a los caporales responsables de cada grupo de cada frente de corte de un ingenio azucarero.

Los cortadores antes de recibir su pago quieren saber cuántas toneladas de caña han cortado en las diferentes modalidades existentes, ya que sobre ese dato es calculado su salario, es por ello que es sumamente importante que los caporales de corte que así se llaman a los jefes o encargados de cierto número de cortadores, tengan un reporte de las toneladas cortadas por semana de su grupo de trabajo, tanto grupal como individual, esto con la finalidad de evitar molestias con los cortadores que repercutan en una baja producción en el corte de caña.

Actualmente, el proceso de entrega de reportes se hace de forma física con la impresión de los reportes de productividad, que actualmente ocupa un promedio de 3 hojas oficio por caporal por semana, cada frente de corte cuenta con un promedio de 8 grupos y son en total 4 frentes de corte, dando un total de 32 caporales a los que se les envían los reportes semanalmente durante 26 semanas, lo cual da un total aproximado de 2500 hojas tamaño oficio de reporte válido, no tomando en cuenta errores de impresión, ni reprocesos después de un cálculo inicial erróneo y vuelto a imprimir dichos reportes, esto solo en insumo de papel, no tomando en cuenta los insumos de la impresión en una máquina que

usa tóner como medio visual. Los reportes son impresos por el departamento de productividad y solo se hace después que se ha contabilizado la planilla inmediata próxima a pagar, que dentro de las políticas de la empresa debe hacerse todos los días miércoles antes de medio día, quedando así oficializado que el reporte se debe imprimir a más tardar el día jueves por la mañana.

En el anterior párrafo se plasmó la impresión del reporte a grandes rasgos, aquí se abordará la forma de entrega, tomando en cuenta que los frentes de corte están asignados en diferentes áreas llamadas zonas, las cuales están ubicadas en el municipio de Chiquimulilla departamento de Santa Rosa, los municipios de Masagua, La Gomera, Sipacate, Escuintla, Siquinalá, La Democracia, Santa Lucía Cotzumalguapa, La Nueva Concepción y Tiquisate todos en el departamento de Escuintla y en los municipios de Río Bravo y Patulul del departamento de Suchitepéquez.

Los frentes de corte viajan constantemente entre las diferentes áreas con base en una programación efectuada por el departamento de cosecha y las necesidades emergentes que se suscitan diariamente. Todos los frentes parten de un centro habitacional asignado desde el inicio de la temporada de corte de caña, viajan a la zona asignada para el corte y regresan al terminar el día de corte, en este centro habitacional pernoctan y reciben los alimentos de desayuno y cena, el almuerzo lo hacen en el área de corte asignado. Como quedó establecido, los reportes pueden imprimirse entre el miércoles por la tarde a jueves por la mañana, y la forma de que dichos reportes lleguen a los caporales es por medio del apoyo que se recibe de los camiones que llevan los alimentos a los diferentes frentes, es decir que si se logran imprimir a buena hora el día miércoles es posible que les llegue en el camión que lleva la cena al centro habitacional esa misma tarde, de lo contario tendrá que llevarlo hasta el día jueves por la mañana.

Los caporales revisan los reportes y toman notas de las inconsistencias que pueden ver en dicho reporte y esperan la llegada de los responsables de la oficina de productividad que se hacen los días jueves y viernes directamente en las áreas de corte, en esta visita los

caporales comunican las inconsistencias para que sean revisadas y corregidas previo al cierre definitivo de la planilla.

Lamentablemente, cuando la impresión se tarda, los mismos responsables de la oficina de productividad llevan los reportes directamente a los caporales en su visita al campo, reduciendo así el tiempo de revisión y poniendo en juego la satisfacción de los cortadores con respecto a su sueldo.

Esta propuesta responde a la necesidad de agilizar la entrega de información a los caporales, para que dispongan del suficiente tiempo para revisión y preparación de inconsistencias, evitando con ello el involucramiento de personal ajeno al proceso, para hacerlo directo entre la oficina de productividad y los responsables de grupo de corte de caña, asegurando de esta manera que la satisfacción del personal de corte no se ve afectada por concepto de sueldos.

IV. MARCO TEÓRICO

A. Cómo la tecnología ha impactado el desarrollo y productividad de los ingenios azucareros en Guatemala.

La historia del azúcar en Guatemala se remonta al siglo XVI, y el 17 de septiembre de 1957 se creó la Asociación de Azucareros de Guatemala -ASAZGUA- que es la organización encargada de regular la agroindustria azucarera, creando políticas gremiales y para una mejor coordinación en general ASAZGUA crea otras instituciones que le permiten proyectarse mejor con la sociedad, con los clientes extranjeros, invirtiendo conjuntamente en Investigación y Desarrollo, y para sobrellevar los retos del impacto al medio ambiente siendo estas:

- Fundazucar, creada en 1990 como un proyecto social de educación, salud y desarrollo de las comunidades.
- Cengicaña en 1992 con el objetivo de estudiar y desarrollar tecnología más eficiente para el cultivo y producción de caña de azúcar
- Expogranel en 1994 para ser más eficiente y competitivo en el tema de terminal de embarque de azúcar.
- Instituto Privado de Investigación sobre el Cambio Climático, para sobrellevar los desafíos y apoyar al medio ambiente.

Es precisamente la entidad Cengicaña la que ha desarrollado no solo investigación para mejorar las variedades de caña que se usan para la producción de azúcar, sino también ha desarrollado 15 aplicaciones para Agricultura de Precisión en caña de azúcar, se clasificaron en tres grupos: Capacitación, Información procesada y Aplicación tecnológica (web y/o Android). Entre las de "Capacitación" se encuentran:

- Planeamiento varietal (variedades).
- Uso de sistemas de información geográfica (Agricultura de Precisión).
- Uso de imágenes satelitales (Agricultura de Precisión).
- Índices de Vegetación (Agricultura de Precisión).

En el grupo de aplicaciones de "Información procesada" están:

- Boletín de plagas rata y barrenador (MIP).
- Mapas de propiedades químicas de suelos (Fertilización).
- Mapas de propiedades físicas de suelos (Riegos).
- Base de datos de productividad (Productividad).
- Imágenes de satélite Landsat 8 (Agricultura de Precisión).
- Imágenes de satélite Sentinel-2 (Agricultura de Precisión).

Entre las aplicaciones más elaboradas se encuentran las catalogadas como "Aplicación tecnológica", las cuales son las siguientes:

- Proyecto MAPA (Agricultura de Precisión, Aplicación web).
- CENGIRIEGOS (Riegos, Aplicación web);
- CENGIRIEGOS (Riegos, Aplicación Android),
- CENGIFERT (Fertilización, Aplicación web);
- Índices de Vegetación-CG (Agricultura de Precisión, Aplicación web)

Beneficios obtenidos del uso Software CENGIRIEGOS (Android), los principales beneficios a obtener con el uso de la herramienta son tres, el primero la optimización del uso del agua donde existe el ahorro de agua y el segundo la disminución del uso de diésel para operar el sistema, los dos están relacionados por la disminución de eventos de riegos y/o el tercero el aumento de la producción de toneladas de caña por hectárea, ya que se mantiene el óptimo de humedad que la planta requiere para una buena producción.

Al ver los beneficios a obtener con CENGIRIEGOS se puede apreciar claramente que la inversión en tecnología con manejo de datos obtenidos de diversas fuentes, se obtienen nuevas formas de hacer más eficientes los procesos que, a simple vista, no cambiaran mucho y que, sin embargo, impactan en la producción de azúcar, es por ello que en CENGICAÑA seguirán invirtiendo en desarrollo de tecnología que automatiza e impacta en el desarrollo del sector azucarero de Guatemala.

B. Tecnología móvil

En las últimas dos décadas se ha vivido una revolución en las comunicaciones, sobre todo las inalámbricas, que ha facilitado la movilidad de las personas al prescindir de un cable para comunicarse, también se ha visto una evolución de la tecnología móvil con prestaciones y características que llega a ser un sustituto de una computadora portátil o de escritorio. Sumado a lo anterior se ha visto una explosión de herramientas y lenguajes de programación para desarrollar aplicaciones sobre tales dispositivos, así como la manera de compartir y vender dichas aplicaciones en los mercados específicos para ellos, llamados tienda de aplicaciones o *app stores*.

Actualmente, más de un 75% de la población tiene un contrato de telefonía móvil. Esto quiere decir que la telefonía móvil de voz cuenta con más de 5.000 millones de usuarios en 212 países diferentes. Por otra parte, más del 60% de la población mundial se conecta a Internet con una conexión inalámbrica. (Josep Prieto Blázquez, 2011).

Todo lo anterior hace posible que muchas soluciones, como la del presente trabajo, puedan ser desarrolladas por medio de aplicaciones móviles de manera rápida, barata y accesible, a cada vez más personas e instituciones, la adaptabilidad, disponibilidad y accesibilidad de soluciones hacen que los desarrollos de software basado en tecnologías móviles crezcan y sean requeridas cada vez más por los usuarios que ven los beneficios de contar con soluciones de tecnologías de información de manera inmediata y en cualquier momento y en cualquier lugar, donde tenga servicio de comunicación inalámbrica.

En Guatemala existen 16.58 millones de habitantes (Banco Mundial y GSMA Intelligence, 2018), y existen dos empresas de telefonía móvil que dominan la escena, siendo ellas Claro que cuenta con 8,993,307 líneas móviles con una participación del 44.1 % del mercado y Tigo que tiene 11,397,364 líneas móviles que representa el 55.9 % (TeleSemana.com, 2021).

Como se puede apreciar se presume que hay más líneas de telefonía móvil que habitantes, esto da por hecho que las tecnologías móviles son el presente y futuro de desarrollo de soluciones tecnológicas.

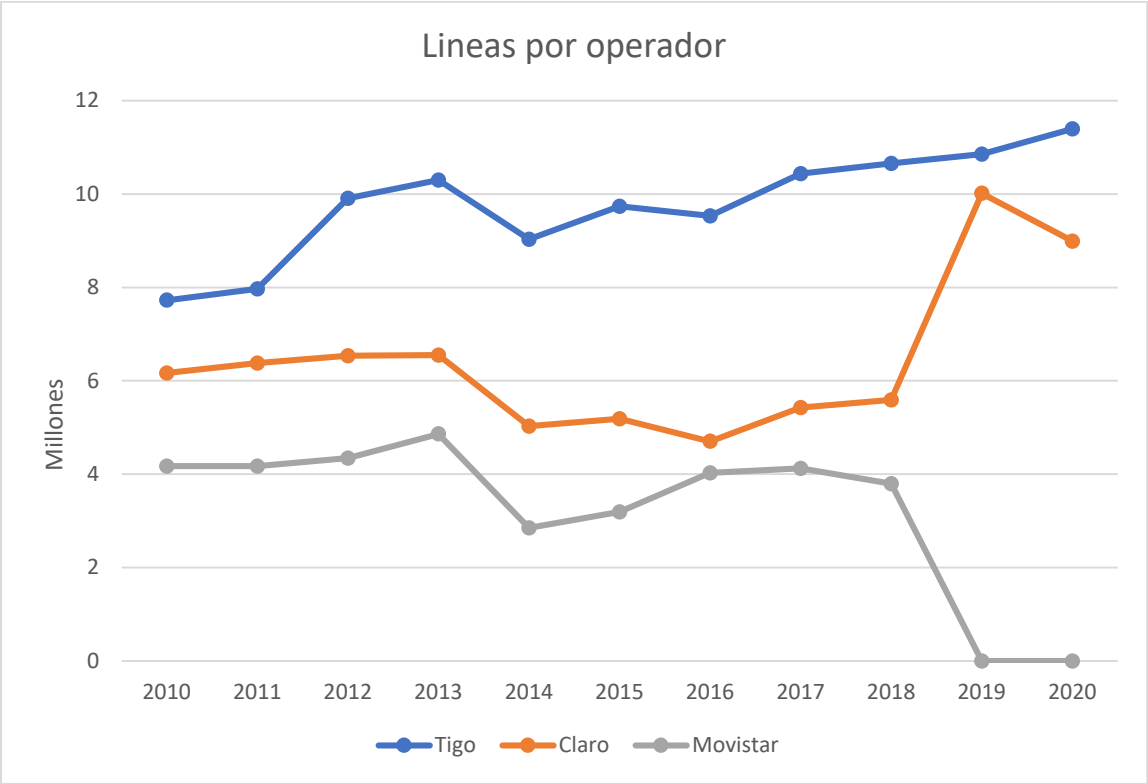


Ilustración 01 Líneas móviles en Guatemala

(TeleSemana.com, 2021)

En el tema de la cobertura, se puede ver claramente que la principal área de cobertura a nivel nacional, está focalizada en la región central, es decir la Capital del País y las Cabeceras Departamentales, así como en las carreteras principales, y mucho más inclinado a la región sur (costas del Océano Pacífico) que al resto de áreas rurales, esto es sumamente importante para el desarrollo de presente trabajo, debido a que la propuesta aquí planteada, se hace sobre una empresa Agroindustrial dedicada a la producción de azúcar, y su área de trabajo precisamente está en la región sur del país.

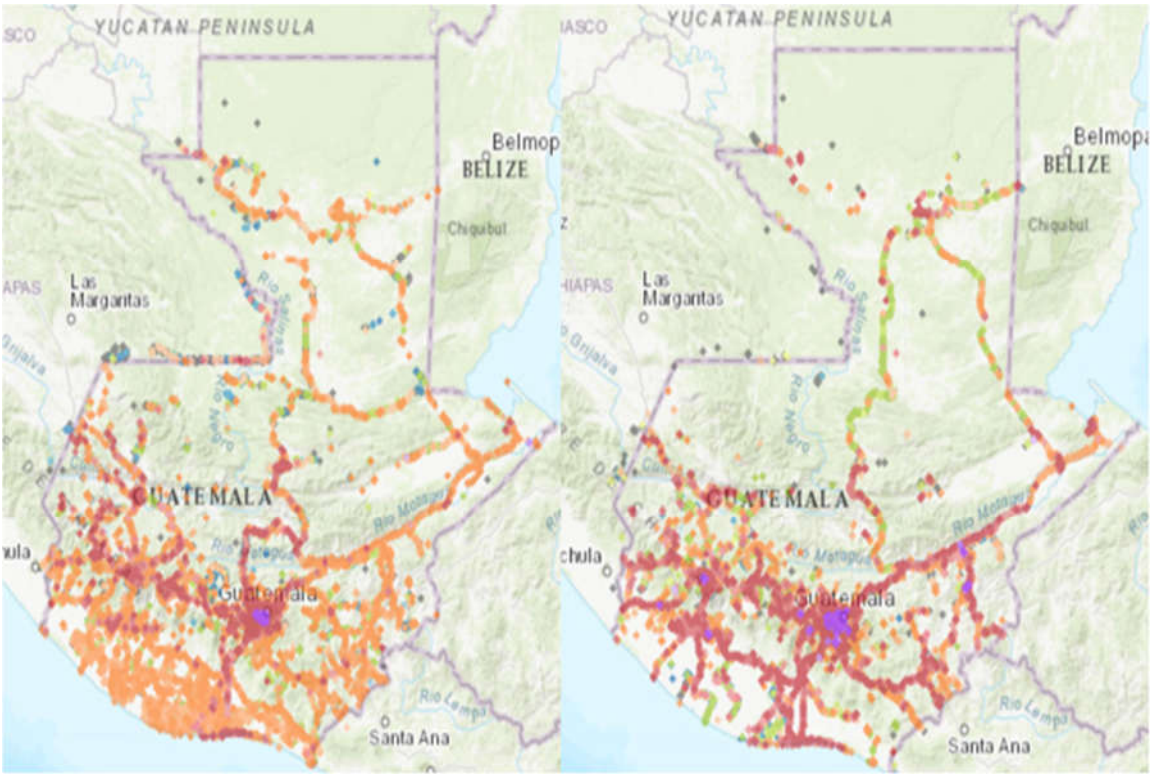


Ilustración 02 Mapa de cobertura Tigo y Claro

(npref.com, 2020)

C. Dispositivos móviles

Una gran cantidad de dispositivos electrónicos se clasifican actualmente como dispositivos móviles, desde teléfonos hasta tablet, pasando por dispositivos como impresoras térmicas portátiles, POS con lector de Chip y RFID. Con tanta tecnología clasificada como móvil, puede resultar complicado determinar cuáles son las características de los dispositivos móviles, para dar una mejor comprensión de lo que es un dispositivo móvil se puede verificar que cumpla con las siguientes características

Tamaño pequeño, se pueden transportar fácilmente por una sola persona, tienen capacidad de procesamiento por sí mismos, tienen capacidad de conexión a una red, tienen memoria, puede ser RAM, MicroSD, Flash, etc., tienen alta capacidad de interacción mediante pantalla, o teclado, al final, un dispositivo móvil puede definirse básicamente con cuatro características fundamentales, las cuales son:

- Movilidad
- Tamaño reducido
- Comunicación inalámbrica
- Interacción con las personas.

Se entiende por movilidad la cualidad de un dispositivo para ser transportado o movido con frecuencia y facilidad. Por tanto, el concepto de movilidad es una característica básica. Los dispositivos móviles son aquellos que son lo suficientemente pequeños como para ser transportados y utilizados durante su transporte. Por tamaño reducido se entiende a la cualidad de un dispositivo móvil de ser fácilmente usado con una o dos manos sin necesidad de ninguna ayuda o soporte externo, el tamaño reducido también permite transportar el dispositivo cómodamente por parte de una persona.

Otro concepto importante es el término inalámbrico, ya que por comunicación inalámbrica se entiende la capacidad que tiene un dispositivo de enviar o recibir datos sin la necesidad de un enlace cableado. Y se entiende por interacción el proceso de uso que establece un usuario con un dispositivo, entre otros factores, en el diseño de la interacción intervienen disciplinas como la usabilidad y la ergonomía. (Julián David Morillo Pozo, 2011)

D. Smartphones

Un *smartphone* es un teléfono móvil o celular que funciona con un sistema operativo móvil (OS), con el que los usuarios pueden conectarse a internet, instalar aplicaciones y llevar a cabo muchas de las actividades que podrían realizar en una computadora.

Los *smartphones* también funcionan como reproductores multimedia portátiles, cámaras digitales, videocámaras y dispositivos de navegación GPS. El sistema operativo equipa el dispositivo con capacidades informáticas avanzadas, ejecuta aplicaciones y permite que el dispositivo realice las siguientes funciones básicas:

- Acceder a las páginas Web y navegar por la Web utilizando redes de datos 4G y 3G y soporte wifi, junto con banda ancha móvil, comunicación de campo cercano y *Bluetooth*.
- Enviar correos electrónicos y sincronizar con múltiples cuentas de correo electrónico.
- Ver, editar y compartir documentos.
- Descargar archivos.
- Crear y reproducir listas de reproducción de música.
- Tomar fotos y grabar videos.
- Jugar juegos y ver películas.
- Comunicarse con amigos y familiares a través de llamadas telefónicas, mensajes de texto y videochats.

En un inicio existían los teléfonos celulares y como un dispositivo adicional los asistentes digitales personales, también conocidos como PDAs los teléfonos celulares se utilizaban sólo para realizar llamadas y las PDAs podían almacenar datos de contacto y listas de tareas pendientes además podían sincronizarse con la computadora, eventualmente las Palm obtuvieron conectividad inalámbrica y llegaron a enviar y recibir correos electrónicos.

Ilustración 03 Palm Pilot 5000

(Creative Commons BY-SA, 2005)

Los teléfonos celulares, por su parte, metieron mano en el campo de mensajería y posteriormente empezaron a agregar más características de los asistentes personales hasta convertirse en *Smartphones* que se conocen hoy en día.

Junto con las características básicas como la mensajería de texto, el correo electrónico y la navegación GPS, el *smartphone* es un dispositivo móvil que funciona como una computadora portátil debido a sus características y capacidades principales avanzadas. La primera característica distintiva del *smartphone* frente a otros teléfonos móviles es su sistema operativo móvil (OS), como Android de Google y iOS de Apple. Junto con el sistema operativo, un *smartphone* normalmente tiene acceso a internet de alta velocidad y un navegador integrado para mostrar páginas web. En un *smartphone*, la Web es accedida por redes de datos 4G o 3G, soporte wifi, banda ancha móvil, NFC o *Bluetooth*.

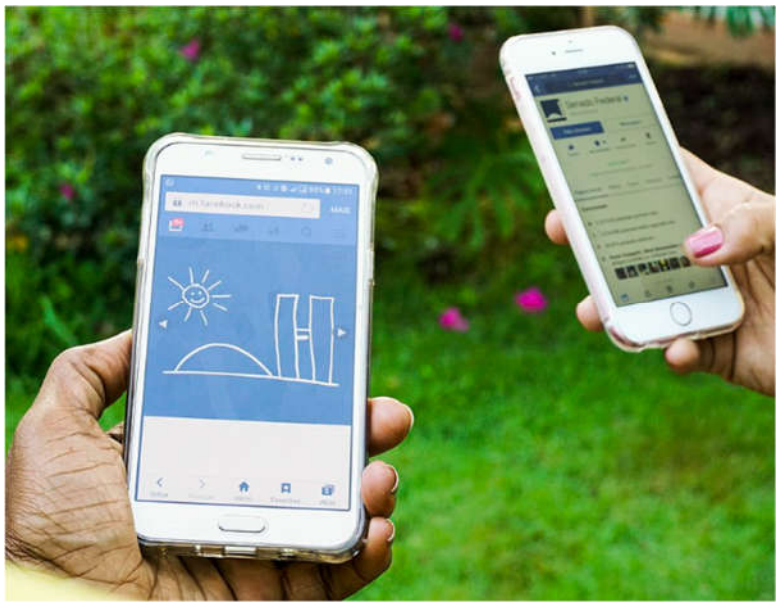


Ilustración 04 Dos Smartphones Samsung e iPhone

(Creative Commons BY-SA, 2016)

E. Redes inalámbricas

La definición de comunicaciones inalámbricas engloba desde una comunicación *Bluetooth* entre un teléfono móvil y un ordenador portátil hasta una comunicación de dos terminales de telefonía móvil GSM. Según el alcance, podemos establecer tres grandes grupos:

- Redes de área personal inalámbrica (WPAN: wireless personal area networks).
- Redes de área local inalámbrica (WLAN: wireless local area networks).
- Redes de área extendida inalámbrica (WWAN: wireless wide area networks). Podemos diferenciar dos tipos de WWAN, según quién controle su acceso:
    - Comunicación fija (FWWAN: fixed wireless wide area networks).
    - Comunicación móvil (MWWAN: mobile wireless wide area networks).

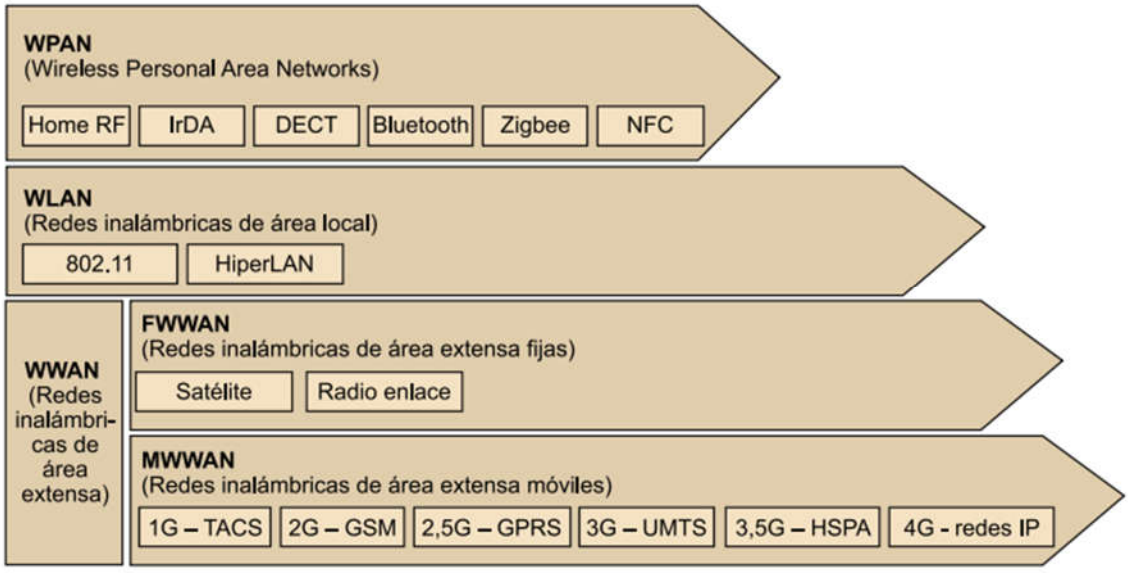


Ilustración 05 Clasificación de redes inalámbricas

(Josep Prieto Blázquez, 2011)

Una "generación" se refiere a un conjunto específico de estándares establecidos para las redes telefónicas. Con cada generación, la velocidad de esas redes aumenta.

En los años 80 apareció la 1G que comenzó a ganar popularidad en todo el mundo. Durante este tiempo, se vio la primera red celular comercial que utilizó señales analógicas.

Si bien fue innovador, hubo una serie de problemas con esta primera generación. Los teléfonos generalmente tenían poca duración de batería y mala calidad de voz, y la norma era que las llamadas fueran interrumpidas constantemente. Los teléfonos también eran mucho más grandes en este tiempo, lo que hizo que poseerlos y usarlos fuera bastante incómodo.

A principios de la década de 1990, el 2G comenzaba a surgir, esta segunda generación utilizó señales digitales en lugar de analógicas e incluyó una nueva herramienta digital para la transmisión inalámbrica llamada Global System for Mobile (o GSM), que se ha mejorado a lo largo de los años. El objetivo principal aquí era proporcionar una opción de comunicación más confiable y segura. Durante este tiempo, se introdujeron funciones como SMS, conferencias telefónicas, retención de llamadas, *roaming* interno y más, y con una mayor velocidad de datos, el estándar 2G podría usarse para enviar y recibir mensajes de texto y correos electrónicos.

En la década 2000 surgió la 3G, aquí es cuando los teléfonos móviles estuvieron disponibles para las masas, con este estándar tecnológico, se hicieron posibles tareas como descargar videos, realizar búsquedas en Internet, compartir fotos, realizar videollamadas, jugar juegos y participar en plataformas de redes sociales. El objetivo de 3G era mejorar los servicios de capacidad de datos y la transmisión de datos, manteniendo bajos costos y la capacidad de admitir una amplia gama de aplicaciones.

Durante la década de 2010, se introdujo 4G. Al realizar actualizaciones a la tecnología existente, LTE (o Long-Term Evolution) se convirtió en estándar durante esta generación. Esto ha permitido que los dispositivos tengan mayores tasas de datos y mayores capacidades para tareas multimedia. Las velocidades más rápidas son estándar para esta generación, junto con una mayor calidad, seguridad mejorada y costos más bajos.

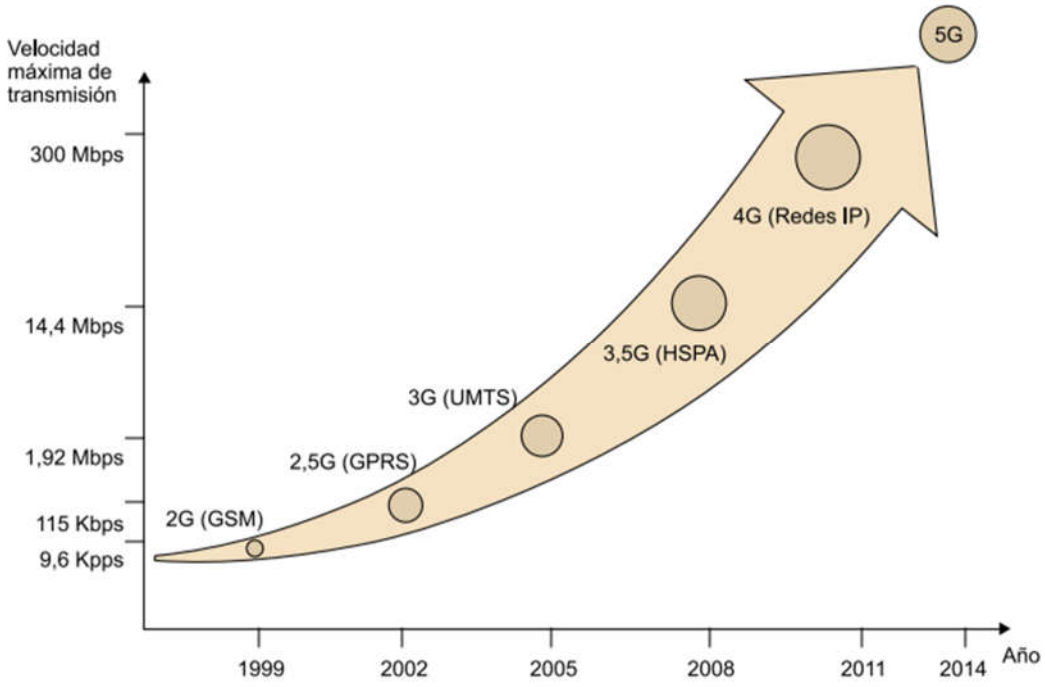


Ilustración 06 Evolución de las redes inalámbricas

(Josep Prieto Blázquez, 2011)

F. Sistemas operativos

El término sistema operativo define a todo el conjunto de programas capaces de administrar las funciones básicas de un ordenador, y la característica principal del sistema operativo es que permanece siempre cargado, es decir, desde que se enciende el dispositivo hasta que se apaga. Dicho de otra forma, el sistema operativo gestiona todo el tráfico de datos dentro del ordenador y entre la máquina y sus periféricos y actúa como intermediario entre el *hardware*, el *software* del sistema y los diversos programas para su correcto funcionamiento.

El sistema operativo crea un entorno seguro y confiable en el que el usuario puede ejecutar sus programas de manera conveniente y eficiente, algunas de las tareas de los sistemas operativos son:

- Iniciar el ordenador.

- Administrar los recursos del ordenador.
- Ejecutar y cerrar los programas de tu ordenador.
- Configurar nuevos programas informáticos.
- Transmitir información entre los componentes de tu ordenador y los programas o *software* del mismo.

G. Sistemas operativos móviles

Android es el sistema operativo para móviles más usado en todo el mundo., este SO es el responsable de motorizar miles de modelos de *smartphones*, se trata de un sistema operativo de código abierto y de distribución gratuita, lo que permite que muchos desarrolladores lo utilicen como sistema operativo de sus propios diseños. En este sentido, podemos encontrar infinidad de tablet y celulares inteligentes con Android como SO de reconocidos fabricantes como LG, Sony, Motorola, Huawei y otros, pero también dispositivos de origen y fabricantes desconocidos.

Un punto en donde Android destaca, y que posiblemente la aleje por mucho de los demás sistemas operativos móviles, es su tienda de aplicaciones, Google Play. En este sentido, Android cuenta con aproximadamente 3.000.000 de apps listas para ser descargadas y usadas, para lo cual lo único que necesitaremos es una cuenta con Google. Esta impresionante cantidad de apps se divide entre aplicaciones gratuitas y de pago.

Ilustración 07 Logotipo de SO Android

(Creative Commons BY-SA, 2016)

IOS está directamente asociado a un tipo específico de *hardware*, por lo que nos será prácticamente imposible obtener una copia del mismo para poder utilizarlo en nuestros proyectos. iOS es un sistema operativo para móviles de código cerrado y pago, desarrollado en 2007 por Apple originalmente con el propósito de ser usado en el iPhone, pero luego fue adaptado para otros dispositivos de la firma como el iPod y el iPad.

Es un sistema operativo para móviles muy seguro y con una alta resistencia a los ataques, debido fundamentalmente a sus raíces basadas en Unix y su sistema de permisos, lo que le otorgan una fortaleza que solo suele verse en los derivados de aquel sistema como Linux. En este sentido, una de las características de seguridad más conocidas de iOS es la activación por iCloud, procedimiento mediante el cual en caso de robo o pérdida permite bloquear o inutilizar el dispositivo si no tenemos las credenciales de acceso a dicho servicio. la estabilidad de este sistema operativo móvil, la que se debe fundamentalmente a haber sido desarrollado en torno a un *hardware* específico, aprovechando cada una de sus ventajas. Si a esto le sumamos un excelente diseño de programación, obtenemos un producto muy sólido y con una alta *performance*, algo que pocos sistemas pueden ostentar.

Ilustración 08 Logotipo de SO iOS

(Apple In., 2017)

Ilustración 09 Cuadro comparativo Android vs. iOS (Abamobile.com)

H. Aplicación móvil

Una aplicación móvil, o app (en inglés) es una aplicación informática diseñada para ser ejecutada en dispositivos móviles (*Smartphone*, tablet, etc.). Por lo general se encuentran disponibles a través de plataformas de distribución, operadas por las compañías propietarias de los sistemas operativos móviles como Google Play Store, de Google para Android, APP Store de Apple para iOS.

A diferencia de las aplicaciones diseñadas para computadoras de escritorio, las aplicaciones móviles se alejan de los sistemas de *software* integrados, en cambio, cada aplicación móvil proporciona una funcionalidad aislada y limitada, las aplicaciones se

instalan directamente sobre el sistema operativo del dispositivo haciendo que se encuentren allí de forma permanente y puedan ser usadas de forma continua, estas requerirán de la conexión a datos para su uso dependiendo de la naturaleza de las mismas, cada aplicación móvil proporciona una funcionalidad aislada y limitada que proporciona a los usuarios servicios y experiencias de calidad.

I. Arquitectura cliente – servidor

Cliente/servidor es una arquitectura de diseño de *software* en la que cada ordenador o proceso en la red es cliente o servidor. Normalmente, los servidores son ordenadores potentes dedicados a gestionar unidades de disco (servidor de ficheros), impresoras (servidor de impresoras), tráfico de red (servidor de red), datos (servidor de bases de datos) o incluso aplicaciones (servidor de aplicaciones), mientras que los clientes son máquinas menos potentes y usan los recursos que ofrecen los servidores.

Esta arquitectura implica la existencia de una relación entre procesos que solicitan servicios (clientes) y procesos que responden a estos servicios (servidores). Estos dos tipos de procesos pueden ejecutarse en el mismo procesador o en distintos.

La arquitectura cliente/servidor implica la realización de aplicaciones distribuidas. La principal ventaja de esta arquitectura es que permite separar las funciones según su servicio, permitiendo situar cada función en la plataforma más adecuada para su ejecución (Sergio Lujan Mora, 2001).

Un cliente es el que inicia un requerimiento de servicio, el requerimiento inicial puede convertirse en múltiples requerimientos a través de redes LAN o WAN, la ubicación de los datos o de las aplicaciones es totalmente transparente para el cliente (Robert Orfali, 1998).

Un servidor es cualquier recurso de cómputo dedicado a responder a los requerimientos del cliente, los servidores pueden estar conectados a los clientes a través de

redes LAN o WAN para proveer de múltiples servicios a los clientes, tales como impresión, acceso a bases de datos, fax, procesamiento de imágenes, etc. (Robert Orfali, 1998).

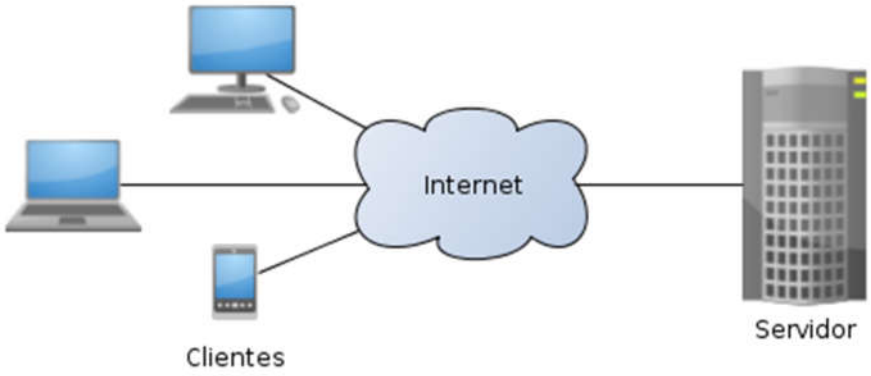


Ilustración 10 Arquitectura cliente - servidor
(Tiago de Jesús Neves, 2013)

J. Android

En los últimos años, los teléfonos móviles han experimentado una gran evolución, desde los primeros terminales, grandes y pesados, pensados solo para hablar por teléfono en cualquier parte, a los últimos modelos, con los que el término "medio de comunicación" se queda bastante pequeño.

Es así como nace Android, que es un sistema operativo y una plataforma software, basado en Linux para teléfonos móviles. Además, también usan este sistema operativo, tablet, *netbooks*, reproductores de música e incluso PC's. Android permite programar en un entorno de trabajo (*framework*) de Java, aplicaciones sobre una máquina virtual Dalvik (una variación de la máquina de Java con compilación en tiempo de ejecución). Además, lo que le diferencia de otros sistemas operativos, es que cualquier persona que sepa programar puede crear nuevas aplicaciones, *widgets*, o incluso, modificar el propio sistema operativo, dado que Android es de código libre, por lo que sabiendo programar en lenguaje Java, va a ser muy fácil comenzar a programar en esta plataforma.

Fue desarrollado por Android Inc., que posteriormente fue comprada por Google en 2005 para ser presentada dos años más tarde, en 2007, en el avance de los estándares abiertos en dispositivos móviles. El código fuente principal de Android es conocido comúnmente como Android Open Source Project (AOSP) y destaca por ser el sistema operativo móvil más usado en todo el mundo con una cuota de mercado de más del 90% en 2018. Los principales componentes del sistema operativo Android son los siguientes:

Núcleo Linux: El núcleo del sistema es Linux y actúa como una capa de abstracción entre el *hardware* del dispositivo y las aplicaciones instaladas. Además, el sistema operativo de Google depende de Linux para otros servicios básicos como la seguridad, gestión de memoria, gestión de procesos, pila de red o controladores.

Run time: El sistema operativo de Google para dispositivos móviles, incluye un conjunto de bibliotecas que proporcionan la mayor parte de las funciones disponibles, en las bibliotecas base del lenguaje de programación Java. Cada aplicación Android corre su propio proceso con su instancia a la máquina virtual Dalvik. Esta máquina ejecutaba hasta la versión 5.0 archivos en formato. dex, pero a partir de esa versión se utilizará el ART, que compila totalmente al momento de instalación de la aplicación.

Bibliotecas: El sistema operativo Android incluye un conjunto de bibliotecas de C o C++ que son utilizadas por varios componentes del sistema. Estas características se exponen a los desarrolladores a través del marco de las aplicaciones de Android. Entre estas bibliotecas, caben destacar a System C, bibliotecas de medios, de gráficos, 3 D o SQLite, entre otras.

Marco del trabajo de aplicaciones: El entorno de Google permite que los desarrolladores tengan acceso a las mismas API del entorno de trabajo utilizadas por las aplicaciones base. Y es que la arquitectura de Android está diseñada para simplificar la reutilización de componentes. Es decir, cualquier aplicación puede publicar sus capacidades y que otras aplicaciones puedan reutilizarlas dentro de unas reglas de seguridad.

Aplicaciones: Android cuenta con ciertas aplicaciones base que permiten el uso de las funciones básicas de un dispositivo como son, correo electrónico, mensajes de texto SMS, calendario, mapas, navegador, contactos y otros. Aplicaciones desarrolladas en lenguaje Java.

Desde su lanzamiento y hasta el día de hoy, Android ha recibido numerosas actualizaciones. Diferentes versiones del sistema que han ido arreglando fallos detectados, añadiendo nuevas funciones, soportes para nuevas tecnologías, etc. Unas versiones que curiosamente han sido desarrolladas bajo un nombre en código de un elemento relacionado con postres o dulces y que además ha ido respetando rigurosamente un orden alfabético, siendo sus versiones las siguientes:

Evolución del sistema operativo Android		
Nombre conocido	**Versión**	**Fecha de lanzamiento**
Apple pie	**1**	23 de septiembre de 2008
Banana bread	**1.1**	9 de febrero de 2009
Cupcake	**1.5**	25 de abril de 2009
Donut	**1.6**	15 de septiembre de 2009
Eclair	**2.0 - 2.1**	26 de octubre de 2009
Froyo	**2.2 - 2.3**	20 de mayo de 2010
Gingerbread	**2.3 - 2.7**	6 de diciembre de 2010
Honeycomb	**3.0 - 3.2.6**	22 de febrero de 2011
Ice cream sandwich	**4.0 - 4.0.5**	18 de octubre de 2011
Jelly bean	**4.1 - 4.3.1**	9 de julio de 2012
KitKat	**4.4 - 4.4.4**	31 de octubre de 2012
Lollipop	**5.0 - 5.1.1**	12 de noviembre de 2014
Marshmallow	**6.0 - 6.0.1**	5 de octubre de 2015
Nougat	**7.0 - 7.1.2**	15 de junio de 2016
Oreo	**8.0 - 8.1**	21 de agosto de 2017
Pie	**9**	6 de agosto de 2018
10	**10**	3 de septiembre de 2019
11	**11**	8 de septiembre de 2020
12	**12**	4 de octubre de 2021

Tabla 01 Evolución del sistema operativo Android

K. Entorno de ejecución

Un entorno de ejecución, o *run time environment*, es esencial para el funcionamiento de una aplicación de *software*, el término describe las condiciones bajo las cuales los programas de computadora se ejecutan en un sistema en tiempo de ejecución.

Inmediatamente después de iniciar un programa, entra en tiempo de ejecución, el programa envía comandos al procesador y tiene acceso a la memoria principal, sin embargo, al escribir programas, estos se encuentran inicialmente en un entorno de ejecución restringido. Esto facilita a los programadores el seguimiento de los programas y sus actividades durante el tiempo de ejecución, catalogarlos y corregir cualquier error.

En este caso, incluso una falla no detiene la grabación, ya que el entorno de ejecución asignado, RTE para abreviar, continúa ejecutándose y puede recopilar información sobre el proceso y los errores.

El entorno de ejecución respectivo le permite abrir los formatos de archivo respectivos sin un programa especial en la aplicación que está utilizando actualmente, esto significa que el entorno de ejecución abre un ecosistema digital, por así decirlo, y le ofrece más compatibilidad y funcionalidad sin tener que cambiar constantemente de aplicación. Incluso si los requisitos para los entornos de ejecución se vuelven bastante complejos en su uso. Los conceptos básicos de los RTE incluyen escribir y leer archivos, enviarlos a través de redes, administración de datos, funciones de clasificación y búsqueda y control de dispositivos de entrada y salida.

L. *Framework* de desarrollo

Es un esquema o marco de trabajo que ofrece una estructura base para elaborar un proyecto con objetivos específicos, una especie de plantilla que sirve como punto de partida para la organización y desarrollo de *software*, utilizar *frameworks* puede simplificar una tarea o proceso, de ahí que se trate de una de las herramientas habituales que manejan los programadores, porque les ayuda a ser más ágiles y productivos

Un *framework* sirve para crear un proyecto en menos tiempo, con un código más limpio y consistente, de manera rápida y eficaz. El *framework* ofrece una estructura base que los programadores pueden complementar o modificar según sus objetivos, el uso de *frameworks* permite, principalmente, agilizar procesos de desarrollo porque podemos reutilizar herramientas o módulos: ya que se tiene el 'esqueleto' o estructura sobre el que trabajar. El hecho de escribir código o desarrollar una aplicación más fácilmente sirve para tener una mejor organización y control de todo el código elaborado, pudiendo usarlo nuevamente en el futuro.

Se puede reutilizar código tantas veces como sea necesario, así mismo, se puede optimizar, con todas las ventajas que ello conlleva, también se puede afrontar tareas propias de programación de forma automatizada, lo que aumentará la velocidad a la hora de programar.

Reducir tiempos implica una mayor productividad, del mismo modo que reutilizar recursos conlleva a minimizar riesgos. Por ello, usar uno o varios *frameworks* supone una gran ayuda para programadores y desarrolladores, ya que facilita sus tareas de forma considerable, adicionalmente se obtienen las siguientes ventajas:

Favorecen el trabajo colaborativo: contar con esa estructura base, con unos estándares de programación, permite que distintos miembros de un mismo equipo trabajen de manera coordinada. Además, favorece que se comparta código y se reduzca la curva de aprendizaje de otros miembros del equipo o propia misma.

Minimiza la posibilidad de riesgos: usar *frameworks* hace más fácil encontrar errores, pero, sobre todo, evitarlos. Garantiza, por lo tanto, mayor seguridad y, además, es habitual que exista una comunidad de desarrolladores detrás del mismo a los que hacerles llegar cualquier duda relativa al uso del *framework*.

Fácil acceso a recursos e información útil: existen infinidad de *frameworks* y, cuando estos están muy extendidos, resulta muy fácil encontrar módulos, herramientas o información para usarlos. Además, pueden permitir utilizar programación avanzada a la que, de otra manera, sería mucho más difícil llegar.

M. Entorno de desarrollo integrado

Los desarrolladores utilizan diversas herramientas y recursos de *software* durante todo el ciclo de vida del desarrollo, que a menudo incluyen bibliotecas de códigos, editores de texto, compiladores y plataformas de prueba. Sin embargo, con cada herramienta adicional, el trabajo del desarrollador se complica un poco más. Seleccionar, aprender, implementar, configurar e integrar cada una de estas herramientas por separado exige tiempo y atención.

Un entorno de desarrollo integrado (IDE) reúne muchas de estas herramientas y recursos comunes para desarrolladores, lo que les permite acceder a ellos a través de una única interfaz gráfica de usuario (GUI). Idealmente, el usuario debería ser capaz de realizar la mayoría de las tareas de desarrollo de un proyecto determinado directamente desde dentro del IDE. Al emplear la visualización de datos y proporcionar una interfaz única y centralizada, los IDE hacen posible que los desarrolladores agilicen las tareas esenciales para una entrega más rápida de *software* y aplicaciones con un control más detallado.

Los IDE más eficaces son aquellos que proporcionan al desarrollador básicamente todo lo que necesita para crear y ejecutar aplicaciones. Sin embargo, no todos los IDE tienen los mismos componentes. Las herramientas más comunes incluidas en un paquete de *software* IDE son las siguientes:

Editores de texto probablemente la principal función de un IDE es el editor de texto. Básicamente, cada entorno de desarrollo integrado incluye un editor de texto donde los usuarios pueden escribir y revisar el código fuente. Por lo general, el editor de texto emplea una interfaz simple mediante el resaltado de sintaxis específica del lenguaje, pero

algunos IDE ofrecen opciones de control más visuales, incluidos los componentes de arrastrar y soltar.

Compiladores, los compiladores toman el código fuente de alto nivel creado en el editor de texto y lo traducen en un conjunto de instrucciones en el lenguaje de máquina que puede comprender la unidad de procesamiento central (CPU) de un ordenador digital.

Depuradores una vez que el código está escrito y compilado, se debe validar. Los depuradores están diseñados para ayudar a localizar errores en el código fuente, además de para probar el rendimiento y la funcionalidad de la aplicación. La depuración generalmente se produce a nivel de código-segmento, donde los desarrolladores pueden identificar y solucionar problemas antes de completar la aplicación final.

Finalización de código, Las opciones de finalización de código facilitan aún más las tareas de programación mediante la identificación y adición automática de componentes de código estándar. Los IDE con finalización del código ayudan a acelerar los ciclos de entrega y, al mismo tiempo, reducen la probabilidad de errores de codificación.

Compatibilidad con lenguajes de programación, aunque la mayoría de los IDE están diseñados para funcionar usando solo un lenguaje de programación específico (como Python, C++ o Ruby), algunos admiten varios de ellos.

Integraciones/complementos, un entorno de desarrollo integrado, reúne *software* esencial y herramientas de desarrollo de aplicaciones en una sola ubicación. Aun así, también debería poder funcionar como parte del ecosistema general de TI de una organización. Los IDE que permiten a los usuarios integrar otras herramientas relevantes tienden a crear un flujo de trabajo de desarrollo más optimizado que aquellos que carecen de capacidades de integración.

N. Entorno de desarrollo integrado Android Studio

Android Studio es actualmente el entorno de desarrollo de aplicaciones de Android más utilizado. Se basa en tres herramientas de desarrollo: JDK, SDK y NDK. Siendo ellas:

JDK: herramientas necesarias para el entorno de desarrollo de Java. En la actualidad, el lenguaje de programación principal utilizado para desarrollar aplicaciones de Android es el lenguaje Java, por lo que JDK es indispensable, pero hoy, cuando el lenguaje pro-hijo de Google, kotlin, está ocupando rápidamente el mercado, Java parece haberse enfrentado a una gran crisis.

SDK: *Software Development Kit*, un compilador de aplicaciones de Android, proporciona una colección de herramientas de uso común para el desarrollo de aplicaciones de Android. Se puede descargar directamente en Android Studio.

NDK: *Native Development Kit*, un compilador para código C / C ++. Proporciona principalmente la interfaz JNI, primero compile el código C / C ++ en la biblioteca so y luego llame a la biblioteca mediante el código Java a través de la interfaz JNI.

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

- Un sistema de compilación flexible basado en Gradle.
- Un emulador rápido y cargado de funciones.
- Un entorno unificado donde se puede desarrollar para todos los dispositivos Android.
- Aplicación de cambios para insertar cambios de código y recursos a la app en ejecución sin reiniciarla.
- Integración con GitHub y plantillas de código para ayudar a compilar. funciones de apps comunes y también importar código de muestra.
- Variedad de marcos de trabajo y herramientas de prueba.

- Herramientas de Lint para identificar problemas de rendimiento, usabilidad y compatibilidad de versiones, entre otros.
- Compatibilidad con C++ y NDK.
- Compatibilidad integrada con Google Cloud Platform, que facilita la integración con Google Cloud Messaging y App Engine.

La estructura de un proyecto con Android Studio incluye uno o más módulos con archivos de código fuente y archivos de recursos. Entre los tipos de módulos se incluyen los siguientes:

- Módulos de apps para Android.
- Módulos de biblioteca.
- Módulos de Google App Engine.

De manera predeterminada, Android Studio muestra los archivos del proyecto en la vista de proyecto de Android, como se ve en la Ilustración 11. Esta vista está organizada en módulos para que puedas acceder rápidamente a los archivos fuente clave de tu proyecto, cada módulo de app contiene las siguientes carpetas:

- **manifests**: contiene el archivo AndroidManifest.xml.
- **java**: contiene los archivos de código fuente Java, incluido el código de prueba de JUnit.
- **res**: contiene todos los recursos sin código, como diseños XML, strings de IU e imágenes de mapa de bits.

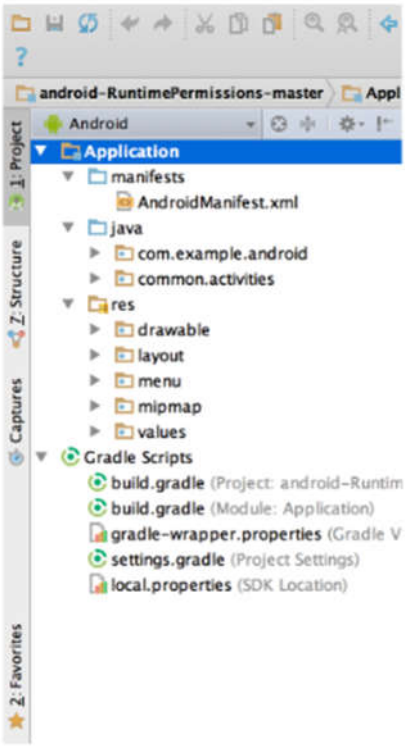


Ilustración 11 Vista de árbol de proyecto Android Studio (Developer.Android.com 2022)

O. Lenguaje de programación Java

Es un lenguaje de programación de propósito general, concurrente, orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (conocido en inglés como WORA, o "write once, run anywhere"), lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra. Java es, a partir de 2012, uno de los lenguajes de programación más populares en uso, particularmente para aplicaciones de cliente-servidor de web, con unos 10 millones de usuarios reportados.

El lenguaje de programación Java fue originalmente desarrollado por James Gosling de Sun Microsystems (la cual fue adquirida por la compañía Oracle) y publicado en 1995 como un componente fundamental de la plataforma Java de Sun Microsystems.

Su sintaxis deriva en gran medida de C y C++, pero tiene menos utilidades de bajo nivel que cualquiera de ellos. Las aplicaciones de Java son generalmente compiladas a bytecode (clase Java) que puede ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente.

La máquina virtual Java actúa como una capa de abstracción adicional entre la plataforma Java y el *hardware* de la máquina subyacente. El código fuente de Java solo puede ejecutarse en aquellas máquinas que tienen JVM instalada.

La compañía Sun desarrolló la implementación de referencia original para los compiladores de Java, máquinas virtuales, y librerías de clases en 1991 y las publicó por primera vez en 1995. A partir de mayo de 2007, en cumplimiento con las especificaciones del Proceso de la Comunidad Java, Sun volvió a licenciar la mayoría de sus tecnologías de Java bajo la Licencia Pública General de GNU.

El lenguaje Java se creó con cinco objetivos principales:

Debería usar el paradigma de la programación **orientada a objetos.** La primera característica, orientado a objetos ("OO"), se refiere a un método de programación y al diseño del lenguaje. Aunque hay muchas interpretaciones para OO, una primera idea es diseñar el software de forma que los distintos tipos de datos que usen estén unidos a sus operaciones. Así, los datos y el código (funciones o métodos) se combinan en entidades llamadas objetos. Un objeto puede verse como un paquete que contiene el "comportamiento" (el código) y el "estado" (datos). El principio es separar aquello que cambia de las cosas que permanecen inalterables. Frecuentemente, cambiar una estructura de datos implica un cambio en el código que opera sobre los mismos, o viceversa. Esta separación en objetos coherentes e independientes ofrece una base más estable para el diseño de un sistema *software.*

Debería permitir la ejecución de un mismo programa en **múltiples sistemas operativos**, significa que programas escritos en el lenguaje Java pueden ejecutarse

igualmente en cualquier tipo de *hardware*. Este es el significado de ser capaz de escribir un programa una vez y que pueda ejecutarse en cualquier dispositivo, tal como reza el axioma de Java, "write once, run anywhere". Para ello, se compila el código fuente escrito en lenguaje Java, para generar un código conocido como "bytecode" (específicamente Java bytecode)—instrucciones máquina simplificadas específicas de la plataforma Java. Esta pieza está "a medio camino" entre el código fuente y el código máquina que entiende el dispositivo destino. El bytecode es ejecutado entonces en la máquina virtual (JVM), un programa escrito en código nativo de la plataforma destino (que es el que entiende su *hardware*), que interpreta y ejecuta el código. Además, se suministran bibliotecas adicionales para acceder a las características de cada dispositivo (como los gráficos, ejecución mediante hebras o *threads*, la interfaz de red) de forma unificada. Se debe tener presente que, aunque hay una etapa explícita de compilación, el bytecode generado es interpretado o convertido a instrucciones máquina del código nativo por el compilador JIT (Just In Time). Hay implementaciones del compilador de Java que convierten el código fuente directamente en código objeto nativo, como GCJ. Esto elimina la etapa intermedia donde se genera el bytecode, pero la salida de este tipo de compiladores solo puede ejecutarse en un tipo de arquitectura. La licencia sobre Java de Sun insiste que todas las implementaciones sean "compatibles".

Debería incluir por defecto soporte para trabajo en red.

Debería diseñarse para ejecutar código en sistemas remotos de forma segura.

Debería ser fácil de usar y tomar lo mejor de otros lenguajes orientados a objetos, como C++.

Para conseguir la ejecución de código remoto y el soporte de red, los programadores de Java a veces recurren a extensiones como CORBA (*Common Object Request Broker Architecture*), *Internet Communications Engine* u OSGi respectivamente.

P. Modelo, vista, controlador

En el patrón de arquitectura MVC (modelo – vista – controlador), basado en el principio de separación de intereses, quizás el patrón de diseño Modelo/Vista/Controlador sea uno de los más utilizados en el desarrollo de proyectos informáticos, este patrón de diseño define tres actores con las siguientes responsabilidades:

El modelo contiene la información con la que el sistema trabaja, proporcionándosela a la vista para que pueda mostrarla y permitiendo realizar cambios en ella desde el controlador.

El controlador responde a acciones del usuario, modificando el modelo cuando sea necesario. Además, se comunica con la vista para que se actualice con los últimos cambios del modelo.

La vista presenta al usuario la información del modelo.

En la Ilustración 12 se muestra la dinámica de este patrón de diseño, que se detalla en los siguientes pasos:

- El usuario interacciona sobre la Vista.
- La Vista informa al Controlador de lo ocurrido.
- El Controlador decide que datos necesita de la Vista para llevar a cabo la tarea como respuesta a la interacción del usuario.
- El Controlador actualiza el Modelo.
- El Modelo informa a la Vista de que se ha actualizado.
- La Vista pide los datos de su interés para visualizarlos.

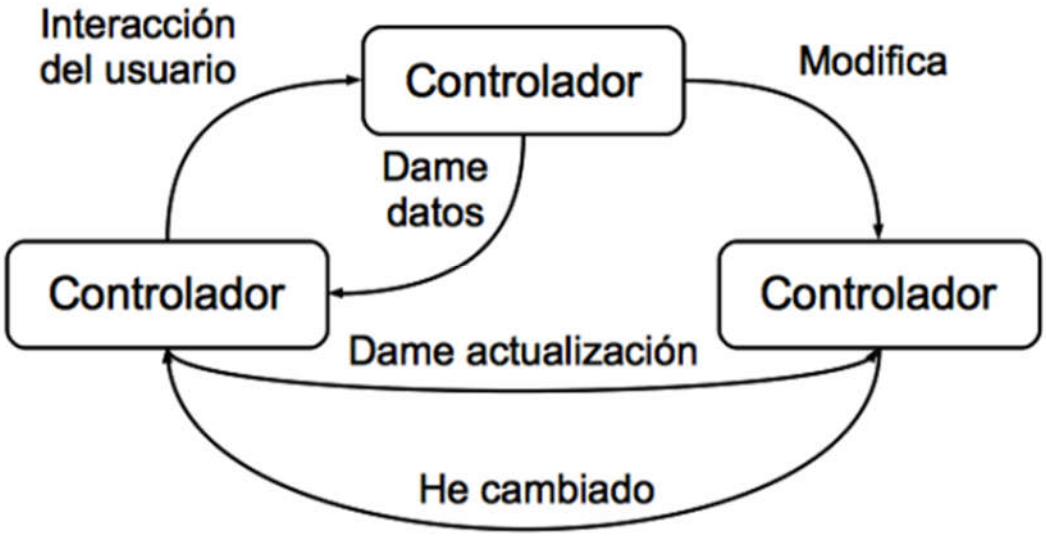


Ilustración 12 Dinámica del modelo MVC

(Oscar Belmonte Fernández, 2012)

Aunque en un primer momento, este patrón puede resultar muy engorroso, o parecernos que hay muchas idas y venidas entre el código de los actores, es todo lo contrario, gracias a esta división de responsabilidades, los posibles cambios en la implementación de uno de los actores son completamente transparente al resto, para poder implementar este patrón de diseño, la Vista debe conocer tanto al Controlador como al Modelo, y por su parte el Controlador debe conocer tanto a la Vista como al Modelo. Por su lado, el único actor que necesita conocer el Modelo es a la Vista.

Q. Bases de datos

Una base de datos es una recopilación organizada de información o datos estructurados, que normalmente se almacena de forma electrónica en un sistema informático. Normalmente, una base de datos está controlada por un Sistema de Gestión de Bases de Datos (DBMS). En conjunto, los datos y el DBMS, junto con las aplicaciones asociadas a ellos, reciben el nombre de sistema de bases de datos, abreviado normalmente a simplemente base de datos.

Los datos de los tipos más comunes de bases de datos en funcionamiento actualmente se suelen utilizar como estructuras de filas y columnas en una serie de tablas para aumentar la eficacia del procesamiento y la consulta de datos. Así, se puede acceder, gestionar, modificar, actualizar, controlar y organizar fácilmente los datos. La mayoría de las bases de datos utilizan un lenguaje de consulta estructurada (SQL) para escribir y consultar datos.

El SQL es un lenguaje de programación que utilizan casi todas las bases de datos relacionales para consultar, manipular y definir los datos, además de para proporcionar control de acceso. El SQL se desarrolló por primera vez en IBM en la década de 1970 con Oracle como uno de los principales contribuyentes, lo que dio lugar a la implementación del estándar ANSI SQL. El SQL ha propiciado muchas ampliaciones de empresas como IBM, Oracle y Microsoft. Aunque el SQL se sigue utilizando mucho hoy en día, están empezando a aparecer nuevos lenguajes de programación.

Las bases de datos han evolucionado drásticamente desde su inicio a principios de la década de 1960. Las bases de datos de navegación, como la base de datos jerárquica (que se basaba en un modelo de árbol y permitía una relación de uno a muchos) y la base de datos de red (un modelo más flexible que permitía relaciones múltiples), eran los sistemas originales que se utilizaban para almacenar y manipular datos. Aunque eran sencillos, estos primeros sistemas eran inflexibles. En la década de 1980, se hicieron populares las bases de datos relacionales, seguidas de las bases de datos orientadas a objetos en la década de 1990. Más recientemente, las bases de datos NoSQL surgieron como respuesta al crecimiento de Internet y la necesidad de acelerar la velocidad y el procesamiento de los datos no estructurados. Hoy en día, las bases de datos en la nube y las bases de datos de autogestión están abriendo nuevos horizontes en lo que respecta a la forma en la que se recopilan, se almacenan, se gestionan y se utilizan los datos.

R. Base de datos relacional

Una base de datos relacional es un tipo de base de datos que almacena y proporciona acceso a puntos de datos relacionados entre sí. Las bases de datos relacionales se basan en

el modelo relacional, una forma intuitiva y directa de representar datos en tablas. En una base de datos relacional, cada fila en una tabla es un registro con una ID única, llamada clave. Las columnas de la tabla contienen los atributos de los datos y cada registro suele tener un valor para cada atributo, lo que simplifica la creación de relaciones entre los puntos de datos.

El modelo relacional significa que las estructuras de datos lógicas (las tablas de datos, las vistas y los índices) están separadas de las estructuras de almacenamiento físicas. Gracias a esta separación, los administradores de bases de datos pueden gestionar el almacenamiento físico de datos sin que eso influya en el acceso a esos datos como estructura lógica. Por ejemplo, si se cambia el nombre del archivo de una base de datos, eso no significa que vayan a cambiar también los nombres de sus tablas.

La distinción entre lógico y físico se aplica también a las operaciones de base de datos, que son acciones claramente definidas que permiten a las aplicaciones manipular los datos y las estructuras de la base de datos. Con las operaciones lógicas, las aplicaciones pueden especificar el contenido que necesitan, mientras que las operaciones físicas determinan cómo se debe acceder a esos datos y llevan a cabo la tarea.

Para garantizar la precisión y accesibilidad continua de los datos, las bases de datos relacionales siguen ciertas reglas de integridad. Por ejemplo, una regla de integridad podría especificar que no se permite duplicar filas en una tabla, a fin de evitar que se introduzca información errónea en la base de datos.

El modelo de datos relacional proporcionó una forma estándar de representar y consultar datos que podría utilizar cualquier aplicación. Desde el principio, los desarrolladores se dieron cuenta de que la virtud principal del modelo de base de datos relacional era el uso de tablas, ya que era una forma intuitiva, eficiente y flexible de almacenar y acceder a información estructurada.

Con el tiempo, los desarrolladores comenzaron a usar el lenguaje de consulta estructurado (SQL) para escribir y hacer consultas en una base de datos, esto sería otra de las grandes virtudes de este modelo. Durante muchos años, el SQL se ha utilizado como el lenguaje para realizar consultas en bases de datos. Se basa en el álgebra relacional y proporciona un lenguaje matemático de uniformidad interna que facilita la mejora del rendimiento de todas las consultas en bases de datos. Otros métodos empleados necesitan definir consultas individuales.

El modelo relacional es sencillo pero muy potente, y lo utilizan organizaciones de todos los tipos y tamaños para una gran variedad de aplicaciones con datos. Las bases de datos relacionales se usan para rastrear inventarios, procesar transacciones de comercio electrónico, administrar cantidades enormes y esenciales de información de clientes y mucho más. Las bases de datos relacionales se pueden emplear para cualquier aplicación de datos en la que los puntos de datos se relacionen entre sí y deban gestionarse de forma segura, conforme a normas y de un modo uniforme.

Las bases de datos relacionales han existido desde la década de los setenta. En la actualidad, el modelo relacional sigue siendo el más aceptado para las bases de datos, gracias a todas sus virtudes.

Son cuatro las propiedades cruciales que definen las transacciones de las bases de datos relacionales: Atomicidad, Uniformidad, Aislamiento y Durabilidad. Estas suelen conocerse como ACID, su acrónimo en inglés.

La atomicidad: define todos los elementos que conforman una transacción completa de base de datos.

La uniformidad: define las reglas para mantener los puntos de datos en un estado correcto después de una transacción.

El aislamiento: impide que el efecto de una transacción sea visible a otros hasta que se establezca el compromiso, a fin de evitar confusiones.

La durabilidad: garantiza que los cambios en los datos se vuelvan permanentes cuando la transacción se haya fijado y hayamos llegado a un compromiso.

S. Lenguaje Unificado de Modelado (UML)

Las ramas más antiguas de la ingeniería han encontrado útil desde hace mucho tiempo representar los diseños mediante dibujos. Desde los inicios del *software*, los programadores han encapsulado sus conceptos en diversos tipos de dibujos o, más ampliamente, de modelos. La comunidad del *software* precisa de una forma de comunicar sus modelos, no solo entre los miembros de un proyecto, sino a todas las personas involucradas en él, y, con el paso del tiempo, a los desarrolladores de futuras generaciones. Necesita un lenguaje no solo para comunicarse con otros, sino para proporcionar un marco en el que desarrolladores individuales puedan pensar y analizar. Además, estos no pueden retener todo esto en sus cabezas durante meses o años. Tienen que registrarlo sobre papel o electrónicamente. El Lenguaje Unificado de Modelado (UML) es un lenguaje estándar de modelado para *software,* un lenguaje para la visualización, especificación, construcción y documentación de los artefactos de sistemas en los que el *software* juega un papel importante. Básicamente, UML permite a los desarrolladores visualizar los resultados de su trabajo en esquemas o diagramas estandarizados

El Lenguaje Unificado de Modelado (Unified Modeling Language, UML) fue creado para forjar un lenguaje de modelado visual común y semántica y sintácticamente rico para la arquitectura, el diseño y la implementación de sistemas de *software* complejos, tanto en estructura como en comportamiento. UML tiene aplicaciones más allá del desarrollo de *software.*

Es comparable a los planos usados en otros campos y consiste en diferentes tipos de diagramas. En general, los diagramas UML describen los límites, la estructura y el comportamiento del sistema y los objetos que contiene.

UML no es un lenguaje de programación, pero existen herramientas que se pueden usar para generar código en diversos lenguajes usando los diagramas UML. UML guarda una relación directa con el análisis y el diseño orientados a objetos.

Hay muchos paradigmas o modelos para la resolución de problemas en la informática, que es el estudio de algoritmos y datos. Hay cuatro categorías de modelos para la resolución de problemas: lenguajes imperativos, funcionales, declarativos y orientados a objetos (OOP). En los lenguajes orientados a objetos, los algoritmos se expresan definiendo 'objetos' y haciendo que los objetos interactúen entre sí. Esos objetos son cosas que deben ser manipuladas y existen en el mundo real. Pueden ser edificios, artefactos sobre un escritorio o seres humanos.

Los lenguajes orientados a objetos dominan el mundo de la programación porque modelan los objetos del mundo real. UML es una combinación de varias notaciones orientadas a objetos: diseño orientado a objetos, técnica de modelado de objetos e ingeniería de *software* orientada a objetos.

UML usa las fortalezas de estos tres enfoques para presentar una metodología más uniforme que sea más sencilla de usar. UML representa buenas prácticas para la construcción y documentación de diferentes aspectos del modelado de sistemas de *software* y de negocios.

El Object Management Group® (OMG®) es un consorcio internacional sin fines de lucro y de membresía abierta para estándares tecnológicos, fundado en 1989. Los estándares de OMG son promovidos por proveedores, usuarios finales, instituciones académicas y agencias gubernamentales. Los grupos de trabajo de OMG desarrollan estándares de integración empresarial para una amplia gama de tecnologías y una gama incluso más amplia de industrias. Los estándares de modelado de OMG, incluidos UML y Model Driven Architecture® (MDA®), permiten un eficaz diseño visual, ejecución y mantenimiento de *software* y otros procesos.

OMG supervisa la definición y el mantenimiento de las especificaciones de UML. Esta supervisión ofrece a los ingenieros y programadores la capacidad de usar un lenguaje para muchos propósitos durante todas las etapas del ciclo de vida del *software* en sistemas de cualquier tamaño, el OMG define los propósitos de UML de la siguiente manera:

Brindar a arquitectos de sistemas, ingenieros y desarrolladores de *software* las herramientas para el análisis, el diseño y la implementación de sistemas basados en *software*, así como para el modelado de procesos de negocios y similares.

Hacer progresar el estado de la industria permitiendo la interoperabilidad de herramientas de modelado visual de objetos. No obstante, para habilitar un intercambio significativo de información de modelos entre herramientas, se requiere de un acuerdo con respecto a la semántica y notación.

Además, OMG estipula que UML cumple con los siguientes requerimientos:

Establecer una definición formal de un metamodelo común basado en el estándar MOF (Meta-Object Facility) que especifique la sintaxis abstracta del UML. La sintaxis abstracta define el conjunto de conceptos de modelado UML, sus atributos y sus relaciones, así como las reglas de combinación de estos conceptos para construir modelos UML parciales o completos.

Brindar una explicación detallada de la semántica de cada concepto de modelado UML. La semántica define, de manera independiente a la tecnología, cómo los conceptos UML se habrán de desarrollar por las computadoras.

Especificar los elementos de notación de lectura humana para representar los conceptos individuales de modelado UML, así como las reglas para combinarlos en una variedad de diferentes tipos de diagramas que corresponden a diferentes aspectos de los sistemas modelados.

Definir formas que permitan hacer que las herramientas UML cumplan con esta especificación. Esto se apoya (en una especificación independiente) con una especificación basada en XML de formatos de intercambio de modelos correspondientes (XMI) que deben ser concretados por herramientas compatibles.

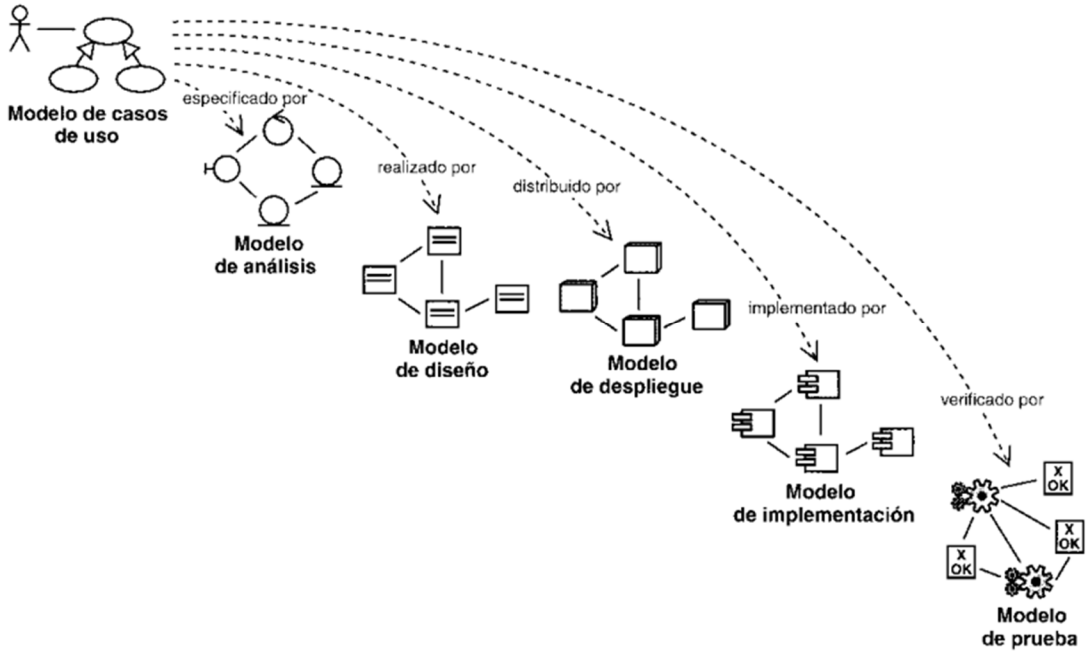


Ilustración 13 Modelo de proceso unificado

(Ivar Jacobson 2000)

V. METODOLOGÍA

A. Alcance.

El alcance de la propuesta es únicamente de permitir al usuario registrado tener acceso al reporte de productividad de cortadores, por semana, por frente, por grupo, por cortador, durante el periodo vigente de zafra (cosecha de caña), por medio de una aplicación móvil basada en sistema Android.

B. Funciones de la aplicación.

- Acceso de usuario registrado por medio de un *password*.

  Como existe un departamento de TI en la empresa se solicitó que la aplicación use las credenciales de los usuarios registrados. Por consiguiente, proporcionaron un *End Point* para poder sincronizarlos con la base de datos instalada en el dispositivo.

- Consulta de reportes

  Los reportes a los que se tiene acceso son únicamente del rendimiento de productividad de los cortadores de caña, y son únicamente del periodo actual de zafra (cosecha de caña) por lo cual no son de carácter histórico.

- Actualizar base de datos

  Este proceso se refiere a la descarga de datos al dispositivo móvil durante este pueda estar conectado a internet, esto se hace debido a que la mayor parte del día el usuario final está en campo agrícola y es donde más tiempo tienen para revisar dichos reportes y existe la posibilidad que no tengan cobertura en donde se encuentren, esto hace que la aplicación móvil no dependa mucho de su conectividad para hacer las consultas.

Los datos se obtienen por medio de un *End Point* que proporciono el departamento de TI. Dichas tablas son actualizadas por ellos cada semana después de que las planillas se contabilizan, proceso que ocurre entre martes después de medio día y miércoles antes de medio día.

C. Restricciones de desarrollo.

- Uso tablet Samsung Galaxy Tab SM-T560 versión Android 4.4.4 del año 2015.
- El modelo arquitectónico de desarrollo es modelo-vista-controlador.
- La aplicación tiene un diseño sencillo con colores institucionales proporcionados por el departamento de TI.
- El desarrollo se realiza en una máquina proporcionada por el Departamento de TI.
- Entre los aspectos de implementación, se usó:
    - El desarrollo es de tipo cliente-servidor
    - Reutilización a nivel de abstracción, objeto y componente.

D. Requisitos para el funcionamiento de la aplicación.

Para el funcionamiento de la aplicación es necesario la utilización de *Smartphone* o tablet con las siguientes características:

- Requerimiento de hardware
    - Memoria RAM mínimo 1 GB
    - Pantalla táctil con resolución mínima de 1440 x 720 píxeles
    - Almacenamiento interno mínimo 8 GB.

- Interfaces de *software*
    - Sistema operativo Android desde la versión 4.0 en adelante
    - La aplicación debe ser instalada directamente en el dispositivo del

usuario final, por medio del departamento de TI.

- Interfaces de comunicación
    - La aplicación solo necesita conectividad a internet únicamente para actualizar la base de datos, hecho que ocurre solo una vez a la semana.
    - El servidor y la aplicación se comunicarán entre sí, mediante un *web service* desarrollado por TI utilizando protocolos estándares en internet.

E. Suposiciones y dependencias.

- Se asume que los requisitos y requerimientos aquí descritos son estables.
- Los equipos en los que se instaló e instalara el prototipo deben cumplir los requisitos antes indicados para garantizar una ejecución correcta del mismo.

F. Características de los usuarios finales.

El usuario final es el caporal de un grupo de corte, es una persona cuya escolaridad mínima de estudios es de nivel básico, tiene experiencia en el proceso de corte de caña, y controla un promedio entre 40 y 60 cortadores.

Tipo de usuario	Usuario registrado
Rol:	Caporal de corte
Actividades en la aplicación:	- Ingreso a la aplicación - Actualizar base de datos - Consultar

Tabla 02 Características del usuario final

G. Requerimientos específicos.

1. Requerimientos funcionales.

Los requerimientos funcionales son aquellos requerimientos que dan sentido a la aplicación, ya que sin ellos no hay razón de ser de la misma.

Identificación del requerimiento:	RF01
Nombre del requerimiento:	Base de datos para el manejo local de la información
Características:	Utilizar una base de datos SQLite para el manejo estructurado de datos, de manera local en el dispositivo
Descripción del requerimiento:	Base de Datos SQLite para guardar de forma estructurada los datos que componen el rendimiento de los cortadores. Manteniendo la integridad de los mismos.
Prioridad del requerimiento:	Alta

Tabla 03 Requerimiento funcional base de datos

Identificación del requerimiento:	RF02
Nombre del requerimiento:	Autentificación de usuario.
Características:	Los usuarios deberán identificarse para acceder a la aplicación.
Descripción del requerimiento:	Todos los usuarios deben ingresar por medio una contraseña, segura
Prioridad del requerimiento:	Alta

Tabla 04 Requerimiento funcional autenticación de usuario

Identificación del requerimiento:	RF03
Nombre del requerimiento:	Actualizar base de datos
Características:	Debe actualizar la base de datos incluida en la aplicación una vez por semana
Descripción del requerimiento:	La aplicación debe conectarse con el servidor por lo menos una vez por semana para actualizar su base de datos, esto con el fin de actualizar los datos de la semana recién cerrada y no depender de conectividad directa al servidor por cada consulta, tomando en cuenta que la jornada laboral, del usuario final se lleva a cabo en campo agrícola donde hay carencia de cobertura de red y por consiguiente inexistente conexión a internet.
Prioridad del requerimiento:	Alta

Tabla 05 Requerimiento funcional actualizar base de datos

Identificación del requerimiento:	RF04
Nombre del requerimiento:	Consulta de datos
Características:	Función principal de la aplicación, dar los resultados de las consultas realizadas.
Descripción del requerimiento:	Mostrar los resultados de las consultas solicitadas, por rango de fecha, cortador o grupo
Prioridad del requerimiento:	Alta

Tabla 06 Requerimiento funcional consulta de datos

2. Requerimientos no funcionales.

Los requerimientos no funcionales son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a propiedades emergentes tales como:

Identificación del requerimiento:	RNF01
Nombre del requerimiento:	Interfaz del sistema.
Características:	La aplicación presentará una interfaz de usuario sencilla para que sea de fácil manejo a los usuarios de la misma.
Descripción del requerimiento:	El sistema debe tener una interfaz de uso intuitiva y sencilla.
Prioridad del requerimiento:	Alta

Tabla 07 Requerimiento no funcional interfaz del sistema

Identificación del requerimiento:	RNF02
Nombre del requerimiento:	Desempeño
Características:	La aplicación debe garantizar a los usuarios un desempeño adecuado en cuanto a la utilización, ofreciéndole una confiabilidad para uso
Descripción del requerimiento:	Garantizar el desempeño de la aplicación que no se salga a media función. Garantizar que el diseño de las consultas u otro proceso no afecte el desempeño de la base de datos, ni considerablemente el tráfico de la red.
Prioridad del requerimiento:	Alta

Tabla 08 Requerimiento no funcional desempeño

Identificación del requerimiento:	RNF03
Nombre del requerimiento:	Rendimiento
Características:	La aplicación debe mostrar los resultados de las consultas en un tiempo prudente
Descripción del requerimiento:	El tiempo de respuesta promedio de la aplicación no debe superar los 3 segundos.
Prioridad del requerimiento:	Alta

Tabla 09 Requerimiento no funcional rendimiento

Identificación del requerimiento:	RNF04
Nombre del requerimiento:	Seguridad.
Características:	La aplicación garantizará a los usuarios la seguridad de sus datos, y que solo será visible dentro del ámbito de la aplicación.
Descripción del requerimiento:	Verificar que será un usuario válido para la aplicación por lo que consulta directamente a la base de datos
Prioridad del requerimiento:	Alta

Tabla 10 Requerimiento no funcional seguridad

Identificación del requerimiento:	RNF05
Nombre del requerimiento:	Disponibilidad
Características:	El usuario puede hacer consultas a cualquier hora del día.
Descripción del requerimiento:	La aplicación debe estar disponible 100%, y al tratarse de una aplicación nativa se instalará directamente en el dispositivo móvil.
Prioridad del requerimiento:	Alta

Tabla 11 Requerimiento no funcional disponibilidad

H. Diagramas.

1. Caso de uso.

- Caso de uso general

Ilustración 14 Diagrama de caso de uso general

Caso de uso general aplicación reporte de productividad cortadores	
Responsable:	Usuario final
Descripción:	El usuario accede a la aplicación por medio de un *password*
Actividades:	* Actualizar base de datos. * Consulta rendimiento por empleado, grupo asignado, por rango de fecha.
Visualizaciones:	Verá la información solicitada agrupada conforme lo solicitó.

Tabla 12 Especificaciones diagrama caso de uso general

- Caso de uso actualización de base de datos

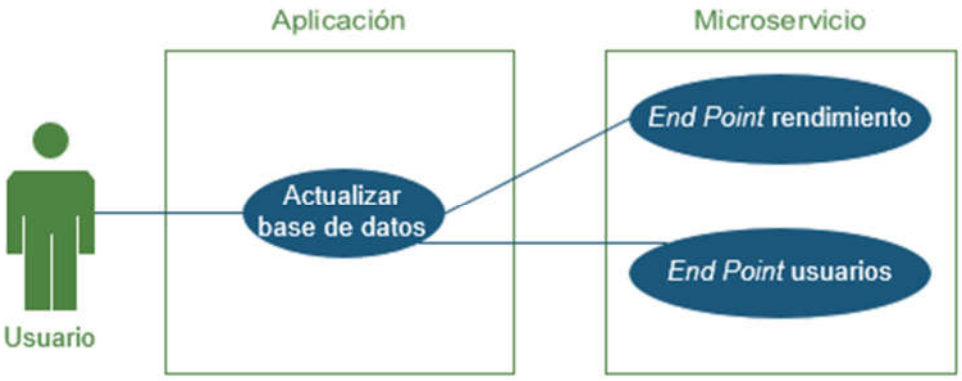


Ilustración 15 Diagrama de caso de uso actualización de base de datos

Caso de uso actualización de base de datos, aplicación reporte de productividad cortadores	
Responsable:	Usuario final
Descripción:	El usuario por medio de un botón actualiza el contenido de la base de datos del dispositivo con las tablas del servidor.
Actividades:	* Actualizar base de datos.
Visualizaciones:	Mensaje de finalización de tarea.

Tabla 13 Especificaciones diagrama caso de uso actualización base de datos

- Caso de uso consultas

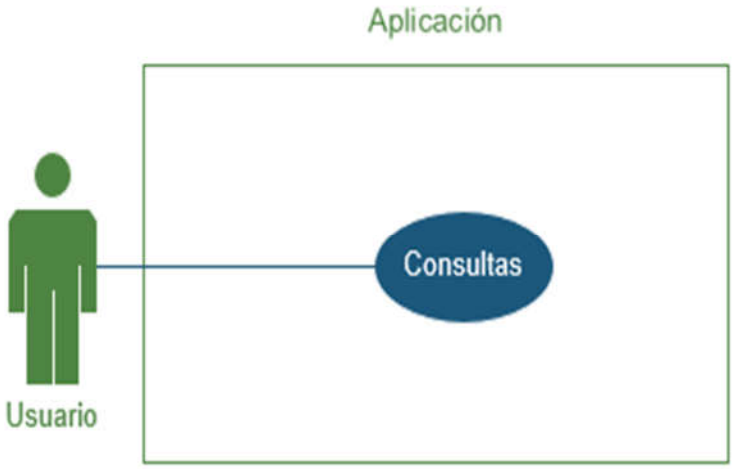


Ilustración 16 Diagrama de caso de uso consulta de datos

Caso de uso consulta de datos, aplicación reporte de productividad cortadores	
Responsable:	Usuario final
Descripción:	Consulta de datos por contador o grupo, por rango de fecha
Actividades:	* Ingresar datos para consulta
Visualizaciones:	* Pantalla de resultados a consulta realizada

Tabla 14 Especificaciones diagrama caso de uso consulta de datos

2. Entidad relación.

Es de recordar que la presente aplicación es únicamente de consulta de datos en tablas, por lo cual no procesa ningún tipo de dato por lo que la base de datos que tiene, siendo relacional no es necesario que las tablas que la componen tengan relación entre sí. Y es alimentada por medio de un *End Point* proporcionado por el departamento de TI. De esta manera solo se procede a mostrar las tablas que la componen, basadas en los datos que se obtienen de micro servicio.

Ilustración 17 Diagrama de entidad relación

3. De flujo por funciones.

- Inicio de sesión.

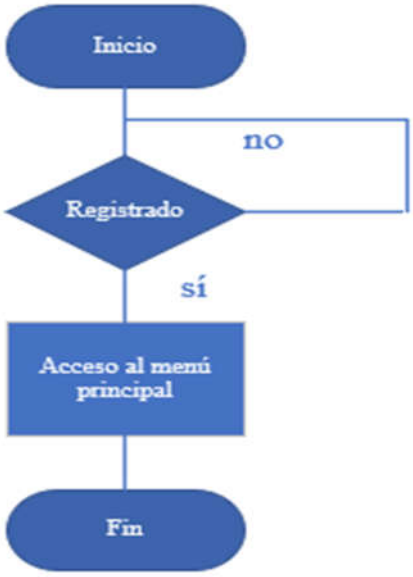


Ilustración 18 Diagrama de flujo inicio de sesión

- Actualizar base de datos.

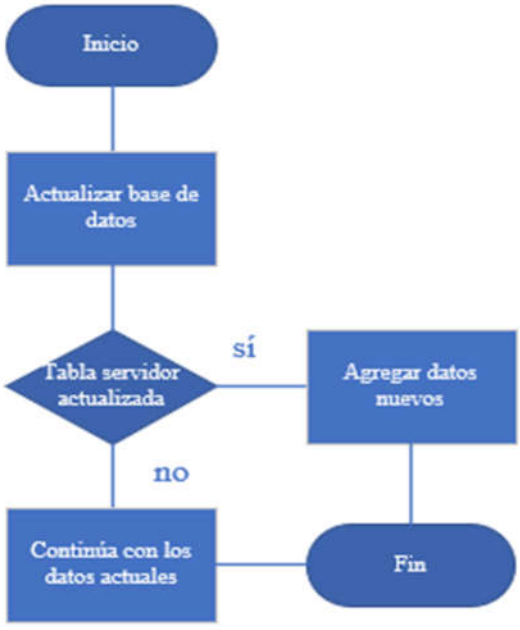


Ilustración 19 Diagrama de flujo actualizar base de datos

- Consultas.

Ilustración 20 Diagrama de flujo consultas

I. Diseño de interface.

1. Inicio de sesión.

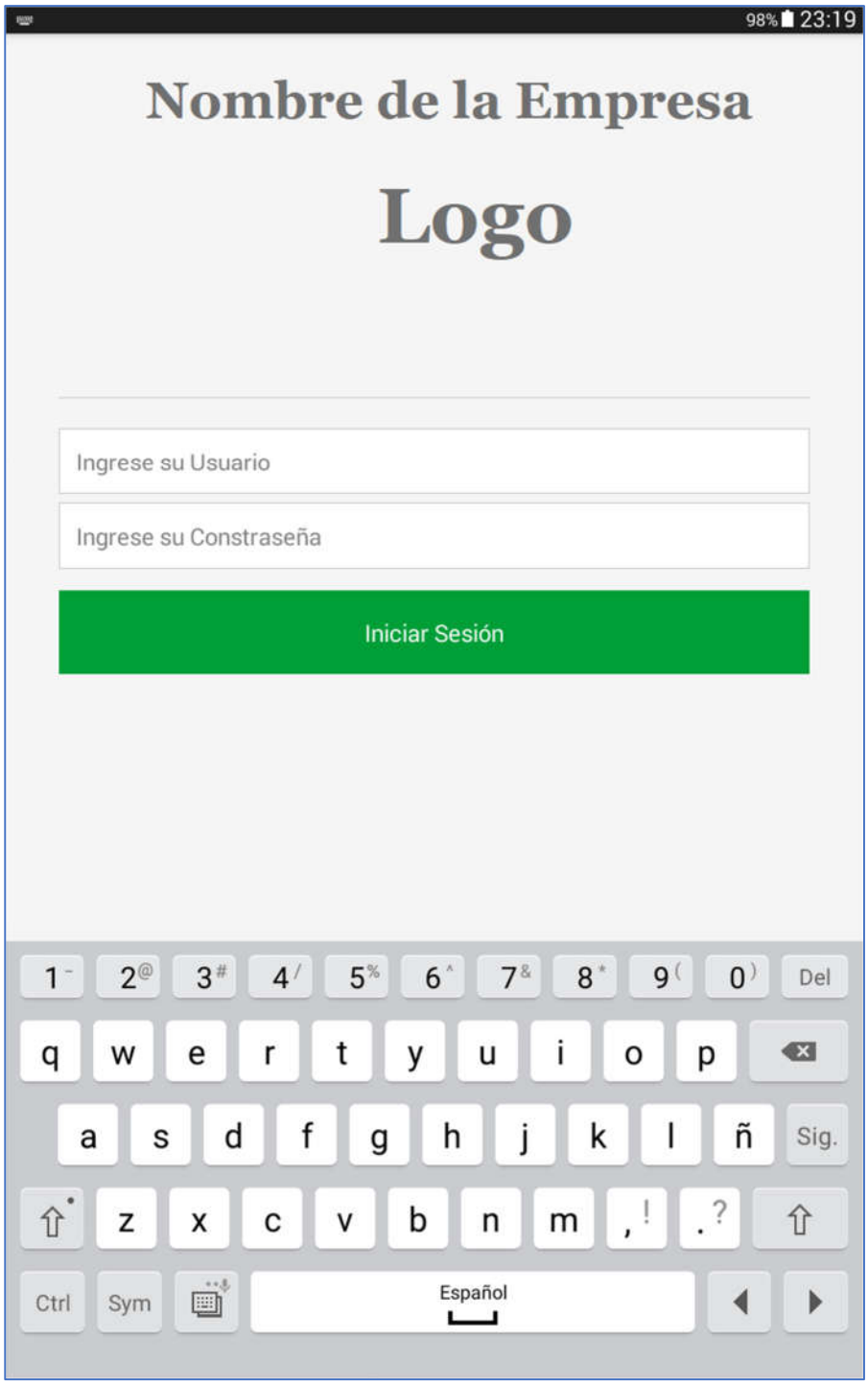


Ilustración 21 Diseño de interface inicio de sesión

2. Menú principal.

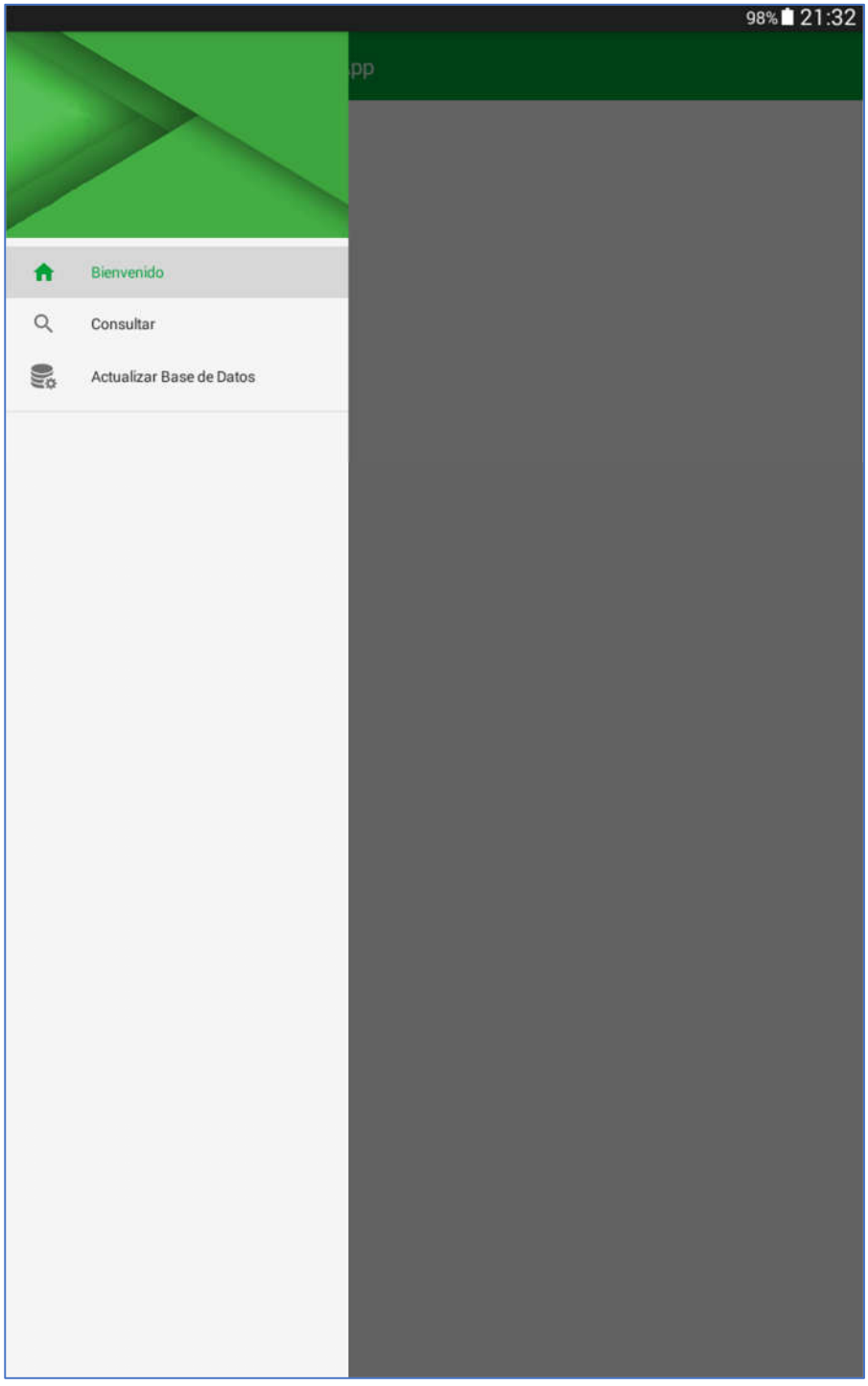


Ilustración 22 Diseño de interface menú principal

3. Actualizar base de datos

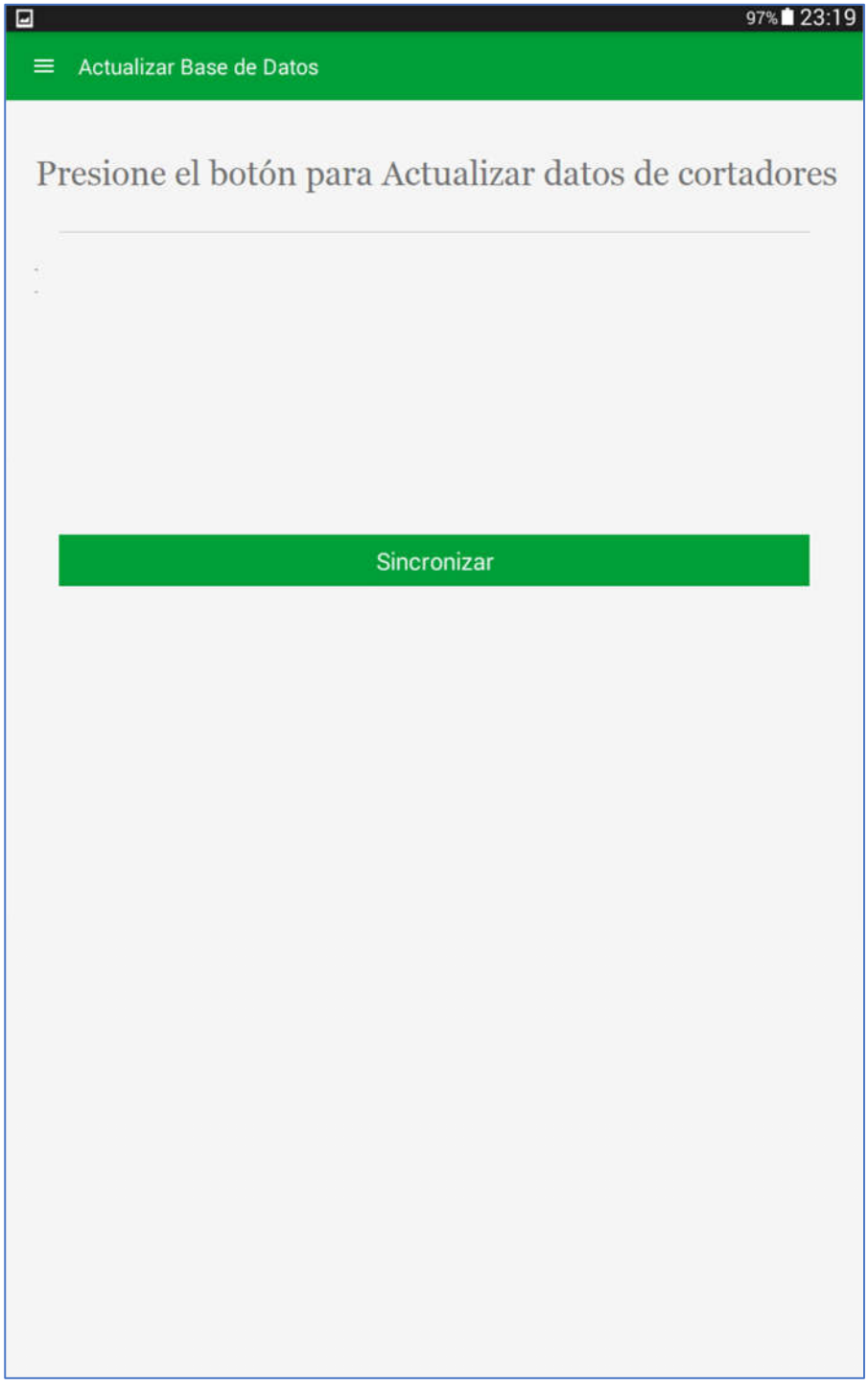


Ilustración 23 Diseño de interface actualizar base de datos

4. Consulta datos

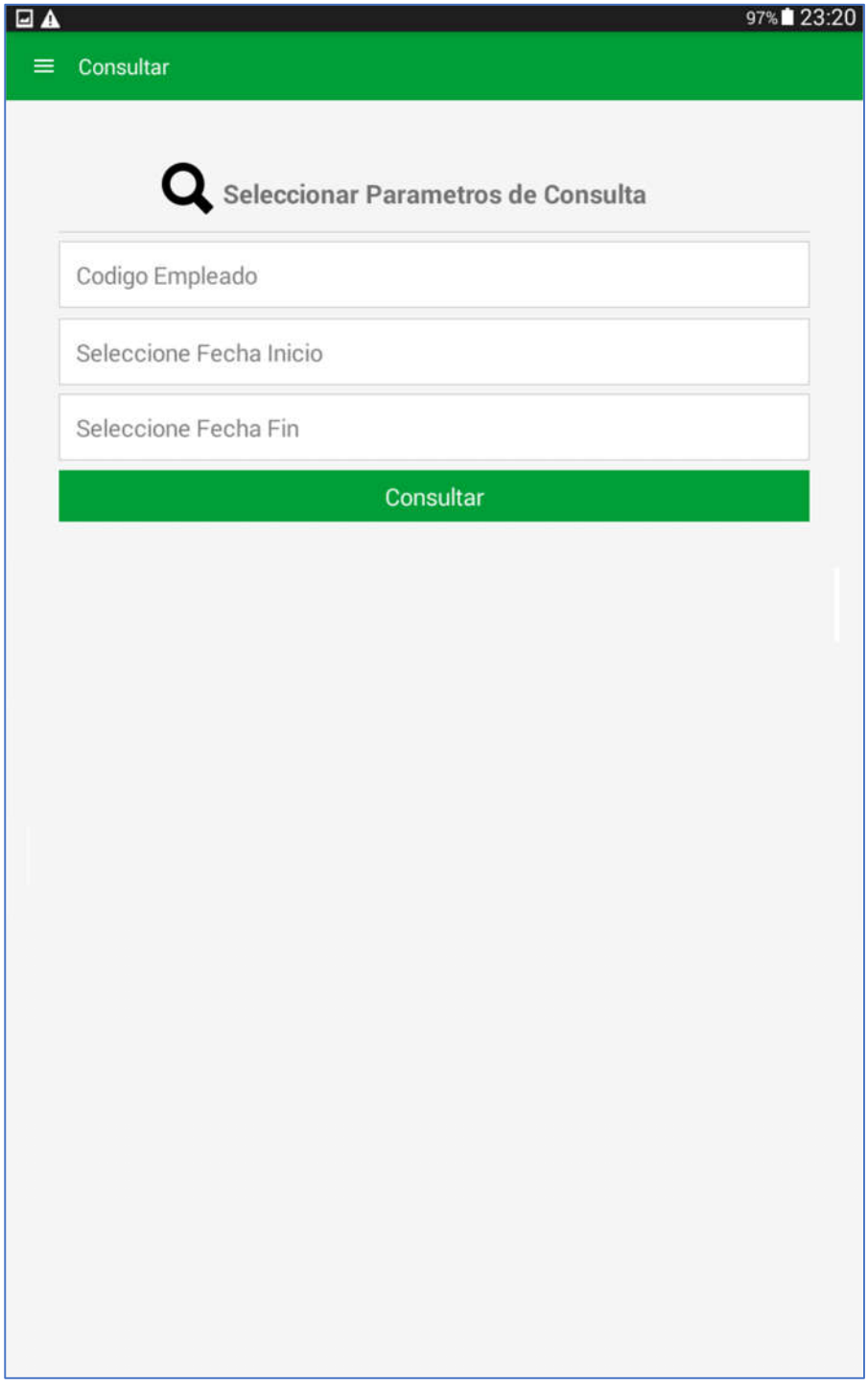


Ilustración 24 Diseño de interface consulta datos

5. Resultado de consulta

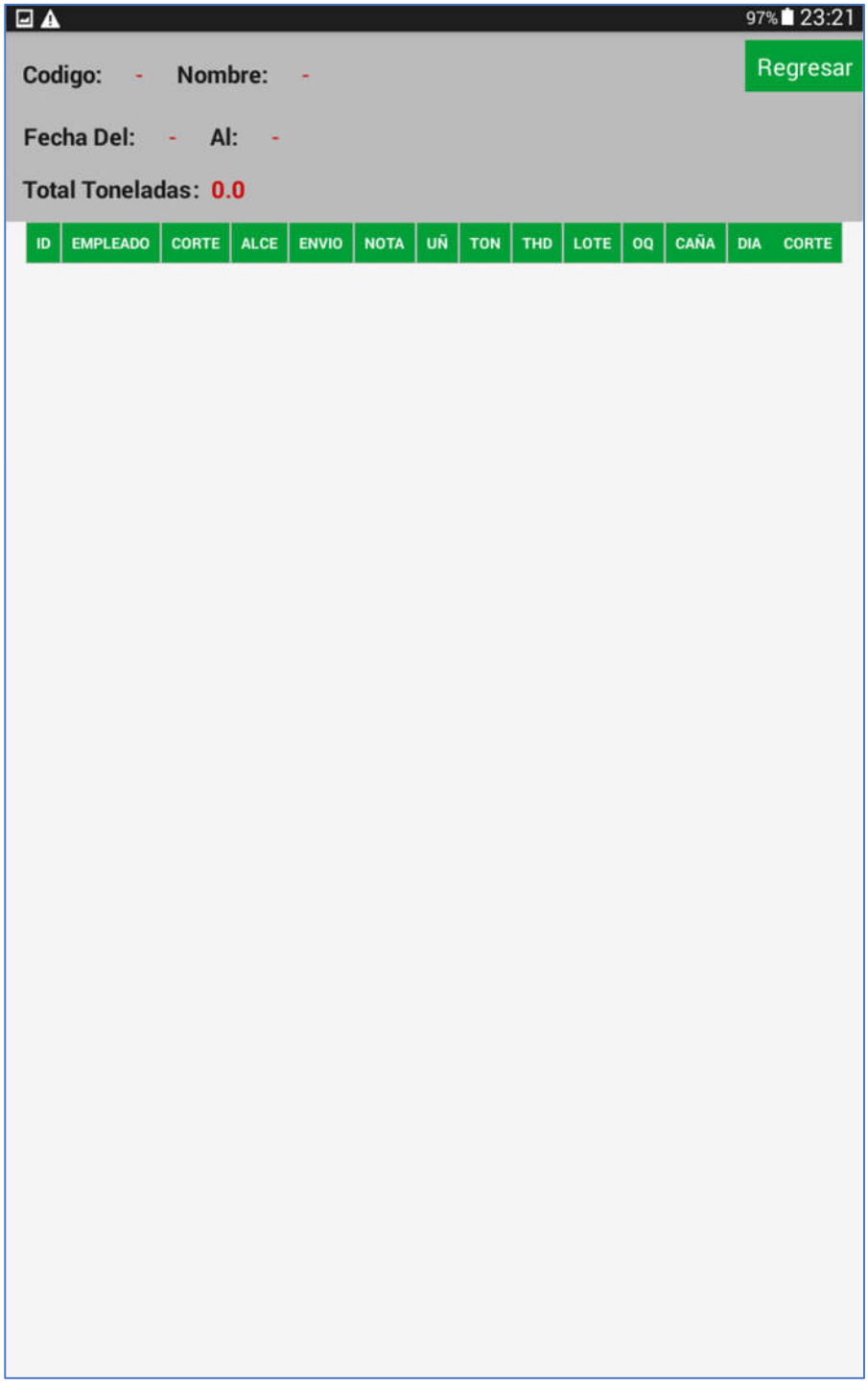


Ilustración 25 Diseño de interface resultado consulta

VI. RESULTADOS

1. Debido al perfil del usuario final se optó por una aplicación móvil basada en el sistema operativo Android, de manera sencilla, con colores institucionales, del tipo minimalista para una fácil adaptabilidad, lectura y uso.

2. Se verificó la cobertura de redes de telecomunicación en los centros habitacionales, y en los 3 centros la cobertura es buena se tiene buena señal tanto de telefonía como de internet, por lo cual el proceso de actualización de base de datos en la aplicación puede hacerse a más tardar en el horario de tarde-noche del día miércoles, cuando los grupos de corte regresan a descansar y cenar a sus respectivos centros habitacionales, proveyendo desde ese momento la información de rendimiento de cortadores, a los caporales reduciendo significativamente el tiempo de espera con respecto al proceso actual que depende de usar la ayuda de otro departamento (camiones de cocina) que llevan la alimentación a campo hasta el día jueves a medio día.

3. Al tener la información en una tabla de la base de datos instalada en el dispositivo móvil, la aplicación provee a los caporales el 100% de la información de rendimiento de los cortadores (de la zafra vigente), con disponibilidad inmediata en cualquier momento y lugar, sin preocuparse de tener conectividad en donde se encuentren, ya que solo es necesario sincronizarla una vez por semana.

4. El prototipo cumple con la finalidad de reducir el uso de papel en su totalidad, ya que se puede hacer la consulta directamente en el dispositivo móvil, pudiendo utilizarla las veces que considere necesario, con este beneficio se reduce el uso de papel y se optimiza el tiempo de los responsables de imprimirlos, clasificarlos y de llevarlos al departamento que colabora para hacerlos llegar hasta su destino.

5. Por medio de la aplicación móvil se logró la independencia de un tercer departamento que colaboraba (sin ser su responsabilidad directa) en la entrega de reportes y de esta

manera se mantenga en línea directa la comunicación entre los caporales y los responsables de la oficina de promotores de rendimiento.

VII. ANÁLISIS DE RESULTADOS

1. El objetivo principal que se deseaba cumplir con esta aplicación móvil fue la de proporcionar una herramienta tecnológica que permita entregar de forma eficiente y confiable los reportes de rendimiento de cortadores para que sean revisados por los caporales de corte en cualquier momento y lugar, para lograr dicho objetivo se implementaron las funcionalidades de base de datos, sincronización de la misma y consulta de reportes, todo ellos en una interfaz de usuario minimalista y de fácil lectura que proporciona los datos solicitados con base en un código de empleado y un rango de fechas.

2. Se logró que la información esté disponible desde el mismo día miércoles para su verificación por parte de los caporales, que lograron conectarse al microservicio de actualización por medio de la red de telecomunicaciones que cubre los centros habitacionales.

3. El microservicio mostró todos los atributos que la tabla en el dispositivo debía contener, y que son el resultado de la combinación de diferentes fuentes de datos como lo son orden de quema, envío de alce, tiquete de báscula, esta información es procesada en sus respectivos sistemas (ajenos a este trabajo) y que en conjunto proporcionan las Toneladas Hombre Día (THD = rendimiento de cortador). Todos los datos están disponibles hasta que se contabilice la planilla de cortadores, y se proceda al cierre semanal, es entonces que se consume el micro servicio y se actualiza la tabla de rendimientos en el sistema, método que demostró su eficacia durante las pruebas que se realizaron al hacer consulta de reportes desconectando de cualquier tipo de comunicación el dispositivo de pruebas que fue proporcionado.

4. El prototipo demostró que es factible reducir el 100% el consumo de papel para la revisión de rendimientos de cortadores, así como la entrega de los datos en aproximadamente un promedio de 10 horas antes que el tiempo que lleva actualmente.

5. Al lograr la independencia de ayuda de un departamento ajeno y sin responsabilidad directa en el proceso, se logra una eficiente gestión de la información, evitando con ello que personas ajenas al proceso tengan acceso a la información relevante al sueldo de los cortadores.

VIII. CONCLUSIONES

1. Se logró la creación de un prototipo funcional en sistema operativo Android que cumple con los objetivos planteados, ya que muestra a los caporales el 100% de la información del rendimiento de cortadores, lo cual coadyuva a mantener la satisfacción laboral de los mismos.

2. Los procesos de automatización pueden ser por medio de pequeñas aplicaciones de software, sin mucha inversión y con mayor impacto en la satisfacción de los clientes internos.

3. La penetración de dispositivos móviles en la vida cotidiana hace más fácil la adaptabilidad de procesos de índole laboral, haciendo más rápida la curva de aprendizaje en cuanto a la utilización de aplicaciones móviles.

4. Se alcanzó la finalidad de reducir el uso de papel en su totalidad, ya que se pueden hacer consultas directamente en el dispositivo móvil, con este logro además del papel se reduce el uso de los consumibles que usa la impresora.

IX. RECOMENDACIONES

1. Al ser una aplicación dedicada a un proceso específico de una empresa específica, la instalación de la aplicación debe ser directamente realizada por el departamento de TI en el dispositivo de uso del caporal.

2. La base de datos debe ser actualizada únicamente una vez por semana, al terminar el proceso contable de las planillas, y para que los caporales lo hagan eficientemente, los responsables de la oficina central, que tienen comunicación constante con los usuarios finales, pueden avisar por medio de un mensaje grupal informando de la disponibilidad de los datos.

3. Como la aplicación solamente muestra los datos de la zafra vigente, es importante que al finalizar dicho periodo la base de datos debe quedar limpia para estar disponible para el próximo periodo.

4. Es importante que se sigan buscando oportunidades para desarrollar aplicaciones que permitan mejorar la eficiencia y productividad, en la industria azucarera y que permitan también colaborar con reducir el consumo de insumos como el papel.

X. BIBLIOGRAFÍA

- Adeva, Roberto. (2022). *Qué es Android: todo sobre el sistema operativo de Google* Obtenido de: https://www.adslzone.net/reportajes/software/que-es-android/

- AzucardeGuatemala.tech. *Cengicaña como centro de tecnología y desarrollo* Obtenido de: https://azucardeguatemala.tech/titulo-cengicana-como-centro-de-tecnologia-y-desarrollo/

- Belmonte Fernández,Óscar; Granell Canut, Carlos; y Edozain Navarro, María del Carmen (2012). *Desarrollo de Proyectos Informáticos con Tecnología Java (1. edición)*. España: Licencia Creative Commons Reconocimiento-NoComercial 3.0

- Blázquez, Josep Prieto; Ramirez Vique, Robert; Morillo Pozo, Julián David y Prieto, Marc Domingo. (2011). *Tecnología y desarrollo en dispositivos móviles (1. Edición)* Barcelona: Editorial Eureca Media, SL.

- Developer.Android.com (2022). *Introducción a Android Studio* Obtenido de: https://developer.android.com/studio/intro?hl=es-419

- Edix.com. (2021. *Framework* Obtenido de: https://www.edix.com/es/instituto/framework/

- Ictea.com. (2022). *Base de Conocimientos ¿Qué es el lenguaje de programación JAVA?* Obtenido de: https://www.ictea.com/cs/index.php?rp=/knowledgebase/8790/iQue-es-el-lenguaje-de-programacion-JAVA.html

- Jacobson, Ivar; Booch Grady y Rumbaugh, James (2000) *El Proceso Unificado de Desarrollo de Software* (1. Edición) España: Pearson Educación S.A.

- Lenovo.com. (2022). *¿Qué es un Smartphone?* Obtenido de: https://www.lenovo.com/gt/es/faqs/pc-vida-faqs/que-es-un-smartphone/?orgRef=https%253A%252F%252Fwww.google.com%252F

- Lucidchart.com. (2022) *Qué es el lenguaje unificado de modelado (UML)*Obtenido de: https://www.lucidchart.com/pages/es/que-es-el-lenguaje-unificado-de-modelado-uml

- Luján Mora, Sergio (2001). *Programación en Internet: Clientes WEB* ePUB v1.0 España: Editorial Club Universitario.

- Marker, Graciela. S*istemas operativos para móviles* Obtenido de: https://www.tecnologia-informatica.com/sistemas-operativos-moviles/

- Nperf.com. *Mapa de Cobertura 3G / 4G / 5G, Guatemala (Redes móviles – Guatemala)* Obtenido de: https://www.nperf.com/es/map/GT/-/198155.Claro-Mobile/signal/?ll=15.42191039994707&lg=-92.208251953125&zoom=7

- Oracle.com. (2022). *¿Qué es una base de datos relacional (sistema de gestión de bases de datos relacionales)?* Obtenido de: https://www.oracle.com/ar/database/what-is-a-relational-database/

- Oracle.com.mx. (2022). *¿Qué es una base de datos?* Obtenido de: https://www.oracle.com/mx/database/what-is-database/

- Orfali, R., Harkey, D., & Edwards, J. (2002). *Cliente/servidor y objetos: Guía de supervivencia* (3. edición.). México: Oxford University Press.

- S. Jesús. (2022). *Sistemas operativos: ¿Qué son y cuántos existen?* Obtenido de: https://economia3.com/sistemas-operativos-que-son/

- Saravia, Andrea. (2020). *La evolución de las telecomunicaciones inalámbricas: de 1G a 5G* Obtenido de: https://www.ufinet.com/es/la-evolucion-de-las-telecomunicaciones-inalambricas-de-1g-a-5g/

- ServiceNow.com. (2022). *¿Qué es un entorno de desarrollo integrado (IDE)?* Obtenido de: https://www.servicenow.com/es/now-platform/what-is-ide.html

- Servisoftcorp.com. (2010). *Definición y cómo funcionan las aplicaciones móviles* Obtenido de: https://servisoftcorp.com/definicion-y-como-funcionan-las-aplicaciones-moviles/

- Stuber, Charles. (2021). *¿Qué es un entorno de ejecución? – Programación*. Obtenido de: https://tecnologiandroid.com/que-es-un-entorno-de-ejecucion/

- Telesemana.com. *Panorama de Mercado – Guatemala.* Obtenido de: https://www.telesemana.com/panorama-de-mercado/guatemala/

- WebyEmpresas.com. (2022). *Satisfacción del Personal (definición, tipos y como medirla)*. Obtenido de: https://www.webyempresas.com/satisfaccion-del-personal/.

XI. ANEXO

A. Características equipo de prueba.

Para realizar las pruebas se contó una Tablet Samsung SM-T560 proporcionada por el departamento de TI.

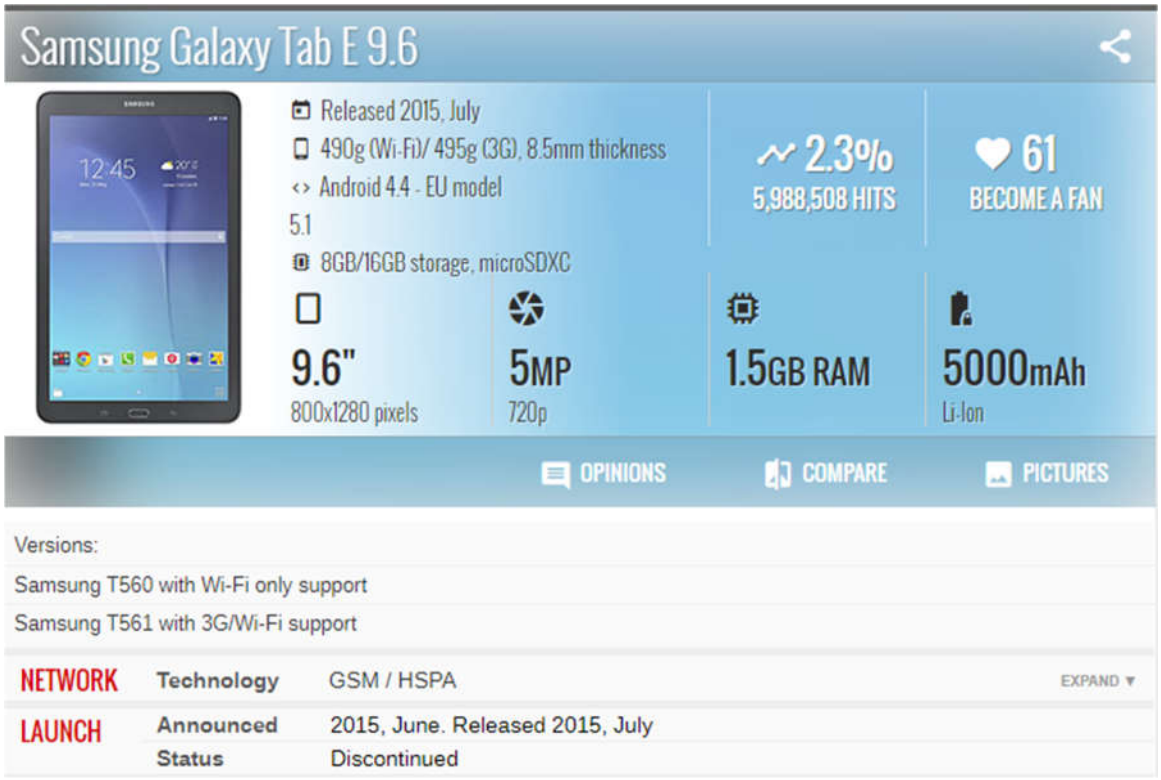


Ilustración 26 Equipo de prueba

XII. GLOSARIO

Android: Sistema operativo que usan algunos dispositivos electrónicos como teléfonos inteligentes y tablet.

Aplicación: Programa desarrollado en lenguaje de programación para ser utilizado en dispositivos electrónicos como computadoras, teléfonos inteligentes o tablet.

App: Aplicación de *software* que se instala en dispositivos electrónicos como teléfonos inteligentes y tablet.

End Point: Un lugar específico de una red donde un programa puede enviar y recibir datos.

Interface visual: Pantalla de interacción entre el usuario y la aplicación.

Microservicio: Son un enfoque para desarrollar una única aplicación como un conjunto de pequeños servicios, cada uno ejecutándose en su propio proceso y comunicándose con mecanismos ligeros, para ser utilizada por otro *software*.

***Password*:** Palabra clave que permite el acceso a la aplicación.

Usuario final: Persona que usará la aplicación móvil.