UNIVERSIDAD NACIONAL AUTÓNOMA DE NICARAGUA
UNAN – LEÓN
DEPARTAMENTO DE COMPUTACIÓN
CARRERA INGENIERÍA EN SISTEMAS DE INFORMACIÓN
TESIS PREVIA A OBTENER EL TÍTULO EN INGENIERÍA EN SISTEMAS DE INFORMACIÓN

Tema: Desarrollo de una app móvil para dispositivos con Sistema Operativo Android, que servirá como diccionario digital español-mayangna; en el periodo de octubre del 2018 a febrero del 2019.

Autores: Br. Grace Joan Villanueva Mejía.
Br. Darwin Manuel Sánchez Corea.
Br. Maynor Teylor Frank.

Tutor: Ing. Miguel Ángel Bárcenas Lezama.

Resumen

Los **mayangna** son un pueblo indígena, también conocido como **sumu** o **sumo**, que habitan a lo largo de los ríos Waspuk o Huaspuc, Pispís y Bocay, en el nororiente de Nicaragua, así como de los ríos Coco y Patuca en la frontera con Honduras y mucho más al sur a lo largo del río Grande de Matagalpa. Los mayangna hablan el idioma sumo, que pertenece a la familia de las lenguas misumalpas.

Se distinguen los dialectos*tawahka, panamahka, Yuskuh y ulwa.*

Se organizan político – administrativamente en territorios y comunidades, con una instancia de coordinación denominada Gobierno de la Nación Sumu – Mayangna; los componen los siguientes 7 territorios demarcado y titulados: Mayangna Sauni As (Bonanza), Mayangna Sauni bu (Jinotega), Mayangna sauni bas (Siuna), Mayangna Sauni Arungka (Bonanza), Mayangna Sauni Tuahka (Rosita), Mayangna Sauni Awastingni (Bilwi),Mayangna Sauni Karawala (Bluefields) y 2 territorios sin titular (Mayangna Sauni Umra (Waspán), Mayangna Sauni Walakwas (Nueva Segovia)).

Sin embargo, el pueblo indígena Mayangna es muy poco conocido en el mundo e incluso en el mismo territorio nacional ya que muy poco se habla de ellos en las escuelas y las universidades por ejemplo cuáles son sus costumbres, que idioma hablan, como aprenderlo y donde poder encontrar información sobre este pueblo indígena.

Es por ello la necesidad de hacer una aplicación móvil que sirva como traductor y diccionario de español a mayangna y viceversa, para que el idioma llegue a conocerse de manera más sencilla, para ofrecerles a los turistas locales y extranjero una herramienta digital con la cual van a poder interactuar con los aborígenes de esa zona del país y conocer más sobre su cultura e idioma y así rescatar esta lengua indígena que es muy poco conocida dentro de nuestra cultura nicaragüense.

Dedicatoria

A mis padres Juan Rafael Villanueva Guevara, Norma A. Mejía de Villanueva, a mi esposo Adolfo H.Blanco Vargas, a mis Hermanos Axel Rafael Villanueva Mejía, Kharem Roxana Villanueva Mejía , Albert Alexander Villanueva Mejía, Alex Samuel Montoya, por su apoyo incondicional en esta etapa de vida y guiándome a hacer realidad mis sueños.

Grace Joan Villanueva Mejía.

A mis padres Noelia Corea Corea, Manuel de Jesús Sánchez Gómez, por su apoyo incondicional y estar allí siempre conmigo para poder cumplir mis metas y sueños en la vida.

Darwin Manuel Corea Sánchez.

Dedico este trabajo a mis padres Simón Abelino Teylor y Adela Frank y mis hermanos (as) por su incondicional apoyo moral para poder culminar mis estudios.

Maynor Teylor Frank

Agradecimientos

A Dios Primeramente por darnos la vida y bendecirnos a diario.

A mis padres y hermanos por brindarme su confianza y su apoyo, sin ustedes no hubiera podido culminar mi carrera.

A mi esposo, por apoyarme incondicionalmente y estar al pendiente de mí.

A mis amigos, por sus buenos deseos y brindarme su apoyo y confianza.

A mis maestros por brindarnos las herramientas más importantes de la vida.

A nuestro tutor Ing. Miguel Ángel Bárcenas Lezama por su atención y tiempo que nos brindó a lo largo de este proyecto.

Grace Joan Villanueva Mejía

A Dios, por darnos la vida y ser fuente de inspiración en el tiempo de nuestros estudios.

A mis padres por haber confiado en mí a lo largo de estos años.

A todos mis amigos y compañeros de universidad por sus buenos deseos y el apoyo brindado para salir adelante.

A mi prima Maryuri Guadalupe Obando Corea por su atención y su apoyo para prepararme para poder entrar a la universidad y cumplir mi meta de ser profesional.

Darwin Manuel Sánchez Corea.

Agradezco en primer lugar a Dios todo poderoso por haberme dado sabiduría e inteligencia para poder culminar mi estudio universitario y llegar a mi meta de ser profesional.

Agradezco profundamente a mis queridos padres y hermanos quienes me apoyaron psicológicamente y me instaron a seguir adelante con mis estudios.
A las autoridades de la Universidad Nacional Autónomo de Nicaragua (UNAN - León), por haberme dado la oportunidad de ingresar a tan prestigiosa universidad y así poder optar al título de Ingeniero en Sistemas de Información.
A todos los docentes que día a día me transmitieron sus conocimientos para así poder alcanzar un peldaño más en mi vida.
De manera especial al Ing. Miguel Bárcenas, por habernos guiado en la elaboración de esta investigación.

Maynor Teylor Frank.

Índice

Tabla de contenido

1- Introducción

Nicaragua es un país de naturaleza multiétnica; la constitución política de la República en sus artículos 5, 8 y 89; reconoce la existencia de los pueblos originarios y su derecho a mantener su identidad y cultura, tener sus propias formas de organización social, administrar sus asuntos locales y mantener las formas comunales de propiedad de sus tierras.

Uno de los pueblos originarios es la Etnia Mayangna, es una población minoritaria dentro de las etnias del Atlántico Norte y Sur. Su cultura ha sido determinada por los ríos, montañas y selvas, logrando desarrollar una vida sencilla de cazadores y recolectores; actualmente se dedican a la agricultura, la caza y la pesca.

Las Tecnologías de la Información y la Comunicación (TICs) están transformando nuestra vida, cambiando nuestras visiones del mundo y modificando los patrones de acceso al conocimiento y de interacción interpersonal. Progresivamente, se han ido incorporando en los diseños curriculares de todos los niveles de la enseñanza formal y no formal.

La tecnología tiene mucha importancia en los procesos de educación de edad temprana. Hoy en día todos los niños se adaptan muy fácilmente al uso de las nuevas tecnologías y es a los adultos a quienes les cuesta adaptarse.

Se ha visto en estudios de educación que los iconos, fotos y otros símbolos visuales son importantes, ya que facilitan el desarrollo del habla, la escritura y lectura. Muchos niños sienten que estos aparatos electrónicos les ayudan en su aprendizaje.

Sin embargo, siempre se deben respetar las áreas críticas en el círculo de aprendizaje temprano en los preescolares. La música, danza, artes visuales, actividades físicas, la exposición a la naturaleza y la construcción de habilidades sociales son áreas que no se deben comprometer.

El secreto de poder integrar la tecnología en aulas de preescolar es verla como cualquier otra herramienta o material para enseñar habilidades específicas y conceptos. El uso de la

tecnología en los colegios está supuesto para expandir, enriquecer, implementar, individualizar, diferenciar y extender el currículo.

Estudios demuestran que los niños muestran una diversidad en sus estilos de aprendizaje y que la manera óptima para que aprendan es más que solo de manera tradicional en donde una profesora les dirige de manera verbal la enseñanza. Todos los educadores deben ser más sensibles en cuanto a la enseñanza a estos diferentes estilos de aprendizaje, especialmente cuando hay una población de alumnos tan diversa a como sucede en nuestros países latinoamericanos.

La reflexión sobre la estructura y principios de funcionamiento de las tecnologías debe estar presente en la formación de docentes. Por ejemplo, como una materia o visión transversal de un área de materias dentro del plan de estudios de formación de docentes. En nuestra realidad, básicamente en Nicaragua, la formación acerca de la tecnología que reciben los docentes es escasa o nula. Por lo tanto, la visión que sustentan es meramente artefactual y, en muchos casos, es acompañada con una mirada tecno fóbica que nos aleja de la necesaria reflexión crítica que debe acompañarnos en este camino.

Debemos ser realistas y no creer que sea posible formar expertos en el uso de todas las tecnologías, sino profesionales críticos y responsables en esta área.

Adherimos a aquellas concepciones que enfocan a la tecnología educativa como una "forma de mirar y pensar la realidad".

El presente trabajo está dirigido a toda persona que quiera aprender el idioma Mayangna, Siendo esta una herramienta principal para su aprendizaje, cabe mencionar que dicha cultura es poco conocida en nuestro país, y ya q en las universidades, principalmente en la nuestra UNAN- león hay muchos estudiantes que hablan el idioma Mayangna se ha pensado en hacer esta aplicación para ser utilizada como herramienta de comunicación implementando un diccionario Español-Mayangna para facilitar la comunicación entre estos estudiante y entenderlos para aprender más de esta cultura que a lo largo que pasan los años se viene olvidando cada día.

Además, esta aplicación abrirá nuevas puertas en el sector turístico para ser utilizada como herramienta de comunicación entre los turistas locales y las personas que practican este lenguaje de comunicación y así dar a conocer la lengua, rescatar el idioma y desarrollar la economía de los pueblos Mayangna mostrando toda su belleza natural, cultura y costumbre que pose.

2- Planteamiento del problema

Este proyecto a realizar tiene como objetivo poder tener un mejor entendimiento del lenguaje mayangna, dirigido a todo público.

Unos de los problemas es que esta lengua no es habitual entre nosotros, no se conoce muy bien en el país, y es necesario que aprendamos un poco de este lenguaje, es por ello dicho proyecto, ya que le servirá de mucho a estudiantes de secundaria como a los universitarios que a diario nos conectamos con persona de diferentes zonas del país que poseen diferentes lenguajes.

Otra de las problemáticas vista en nuestro país es la falta de conocimiento sobre la existencia de este lenguaje, es por ello que como estudiantes queremos dejar este proyecto para ser aprovechado al máximo y que puedan obtener más beneficio en su estudio y enriquecer su lengua. Nuestro país es visitado a menudo por extranjeros y no existe una aplicación móvil donde puedan encontrar información sobre esta lengua y que mejor q realizar dicha aplicación en un sistema conocido por todo público y fácil manejo como son las aplicaciones móviles para dar a conocer este idioma.

La lengua mayangna ha sufrido el mismo destino que sus hablantes. La iglesia morava tomo la decisión de realización de la evangelización de los mayangna en miskito, como resultado muchos mayangna se han alfabetizado en miskito y en español, pero no pueden leer bien en su lengua materna.

3- Antecedentes

Un primer trabajo corresponde a Ronas Dolores Green Suazo dirigente legendario del territorio Tuahka de la fundación tuahka, que realiza dos ediciones del diccionario cuatrilingue Tuahka-Miskitu-español-Inglés, con palabras simples y compuestas, nombres de lugares y sus significados, términos y frases que habían desaparecido y fueron rescatados a través de una investigación minuciosa que se realizó en las comunidades tuahkas.

Cumpliendo con uno de los principales objetivos de la Fundación que es investigar, recopilar y divulgar las expresiones del idioma Tuahka o Mayangna, el cual se encuentra en peligro de extinción. Se pudo llevar a cabo ésta publicación con el valioso aporte de profesionales, educadores, técnicos y ancianos de las 14 comunidades Tuahkas.

Este trabajo se relaciona con esta investigación el cual es seguir el mismo objetivo de investigar, recopilar y divulgar las expresiones del idioma Mayangna.

Un dispositivo móvil es el más adecuado para este trabajo y es utilizado para tomas datos de encuestas en campo en el sector salud, educación, turismo, etc.

Dispositivos como teléfonos tienen el potencial de optimizar la recolección de los datos en campo, facilitando la labor de los Operadores/trabajadores/empleados. Pueden permitir además la conformación de bases de datos que posteriormente pueden ser asociadas a la información.

Por otra parte, los dispositivos móviles han tenido un desarrollo importante en los últimos dos o tres años. Hoy es frecuente que cuenten con capacidades de localización e interfaces amigables al usuario, como es el caso de Iphone, BlackBerry y los dispositivos con sistema operativo Android, a precios significativamente menores que en el pasado reciente, lo que los pone al alcance de un grupo mayor de la población. Paralelamente, el sistema operativo Android de Google ha forzado a muchos fabricantes a liberar herramientas de desarrollo y, en algunos casos, sistemas operativos –como es el caso de Symbian–, lo que facilita, no solamente el acceso a los dispositivos, sino el desarrollo de aplicaciones complejas sobre ellos.

4- Justificación

El dialecto mayangna es uno de los dialectos más antiguos en la historia de nuestro país, sin embargo, no es tan reconocido a nivel nacional solo es hablado en la costa caribe norte, costa caribe sur y parte de Jinotega. Las razones no existen diccionarios o libros del habla mayangna en las distintas librerías o bibliotecas del país para conocer más del dialecto.

Por ello la necesidad de hacer una aplicación móvil en Android que sirva como Diccionario y Traductor del dialecto Mayangna al español.

Esta App tiene como motivación consultar palabras de español a mayangna y viceversa; el usuario tendrá la facilidad de ingresar palabras y frases ya sea en español o en mayangna, así como informarse sobre la cultura, historia y lenguaje de los pueblos Mayangnas en nuestro país.

5- Objetivos

5.1- Objetivo general:

Desarrollo de una app para dispositivos con Sistema Operativo Android que servirá como diccionario digital Español - Mayangna, que servirá como diccionario y traductor para los estudiantes de la UNAN-LEÓN y personas en general para investigar, Recopilar y Promulgar la cultura indígena Mayangna, en periodo de Octubre del 2018 a febrero del 2019.

5.2- Objetivos específicos:

- Generar una base datos de frases y palabras debidamente enriquecida y documentada sobre el lenguaje Mayangna para el desarrollo del proyecto Android.
- Desarrollar la aplicación mediante la plataforma Android Studio versión 3.0
- Realizar actualizaciones de la base de datos mediante Servicios Web usando la librería Retrofit de Android.

Resultados esperados

El siguiente apartado se describe los resultados que se deben obtener una vez finalizado todas las etapas del proyecto.

- Diseñar las interfaces de la aplicación las cuales permitirán especificar cómo va a interactuar la aplicación con el usuario.
- Diseñar una aplicación en Android Studio con funciones públicas que permita comunicar la aplicación móvil con el servidor de base de datos y los servicios web.
- Realizada la codificación de la aplicación, esta se aplicará de acuerdo a las interfaces diseñadas, permitirá interactuar con el usuario para consultar los datos y con el servidor para validar y guardar estos datos.
- Realizar pruebas de la aplicación, se realizará un informe donde se especificarán los errores de código y diseño que los usuarios han encontrado. Así como recomendaciones o mejoras (Ver anexos).

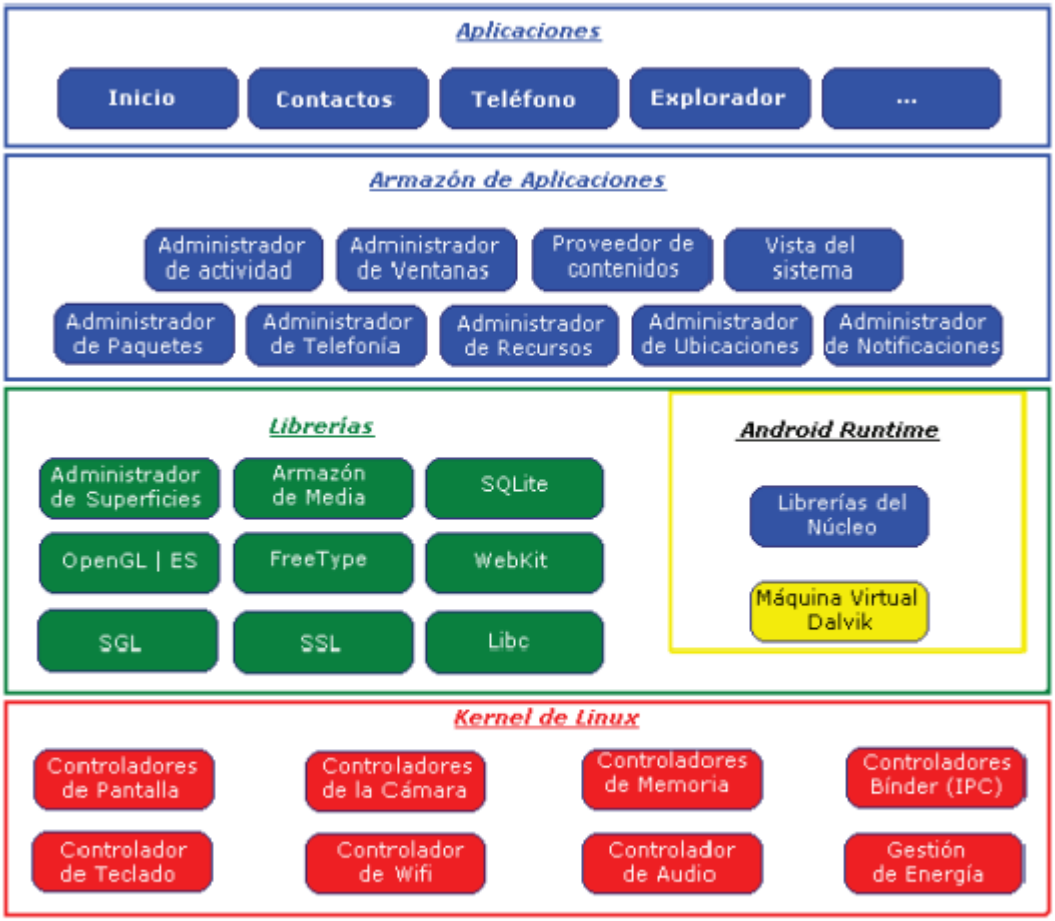
6. Marco Teórico

6.1- Android

Como hemos comentado, existen muchas plataformas para móviles (iPhone, Symbian, Windows Phone, BlackBerry, Palm, Java Mobile Edition, Linux Mobile (LiMo), etc.); sin embargo Android presenta una serie de características que lo hacen diferente. Es el primero que combina, en una misma solución, las siguientes cualidades:

- Plataforma realmente abierta: Es una plataforma de desarrollo libre basada en Linux y de código abierto. Una de sus grandes ventajas es que se puede usar y "customizar" el sistema sin pagar royalties.
- Portabilidad asegurada: Las aplicaciones finales son desarrolladas en Java, lo que nos asegura que podrán ser ejecutadas en una gran variedad de dispositivos, tanto presentes como futuros. Esto se consigue gracias al concepto de máquina virtual.
- Arquitectura basada en componentes inspirados en Internet. Por ejemplo, el diseño de la interfaz de usuario se hace en XML, lo que permite que una misma aplicación se ejecute en un móvil de pantalla reducida o en un netbook.
- Gran cantidad de servicios incorporados
- Gran cantidad de servicios incorporados: Por ejemplo, localización basada tanto en GPS como en redes, bases de datos con SQL, reconocimiento y síntesis de voz, navegador, multimedia, etc.
- Aceptable nivel de seguridad. Los programas se encuentran aislados unos de otros gracias al concepto de ejecución dentro de una caja que hereda de Linux. Además, cada aplicación dispone de una serie de permisos que limitan su rango de actuación (servicios de localización, acceso a Internet, etc.).
- Optimizado para baja potencia y poca memoria: Por ejemplo, Android utiliza la Máquina Virtual Dalvik. Se trata de una implementación de Google de la máquina virtual de Java optimizada para dispositivos móviles.
- Alta calidad de gráficos y sonido. Gráficos vectoriales suavizados, animaciones inspiradas en Flash, gráficos en 3 dimensiones basados en OpenGL. Incorpora los códecs estándar más comunes de audio y vídeo, incluyendo H.264 (AVC), MP3, AAC, etc.

6.1.1- Arquitectura Android



Ilustracion 1 Nucleo del sistema operativo Android

El núcleo Linux

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

Runtime de Android

Está basado en el concepto de máquina virtual utilizado en Java. Dado las limitaciones de los dispositivos donde ha de ejecutarse Android (poca memoria y procesador limitado) no fue posible utilizar una máquina virtual Java estándar. Google tomó la decisión de crear una nueva, la máquina virtual Dalvik, que respondiera mejor a estas limitaciones.
Algunas características de la máquina virtual Dalvik que facilitan esta optimización de recursos son: que ejecuta ficheros Dalvik ejecutables (.dex) (formato optimizado para ahorrar memoria). Además, está basada en registros. Cada aplicación corre en su propio proceso Linux con su propia instancia de la máquina virtual Dalvik. Delega al kernel de Linux algunas funciones como threading y el manejo de la memoria a bajo nivel.

También se incluye en el Runtine de Android el "core libraries" con la mayoría de las librerías disponibles en el lenguaje Java.

Librerías nativas

Incluye un conjunto de librerías en C/C++ usadas en varios componentes de Android. Están compiladas en el código nativo del procesador. Muchas de las librerías utilizan proyectos de código abierto. Algunas de estas librerías son:

- System C library: una derivación de la librería BSD de C estándar (libe), adaptada para dispositivos embebidos basados en Linux.
- Media Framework: librería basada en PacketVideo's OpenCORE; soporta codees de reproducción y grabación de multitud de formatos de audio, vídeo e imágenes MPEG4, H.264, MP3, AAC, AMR, JPG y PNG.
- Surface Manager: maneja el acceso al subsistema de representación gráfica en 2D y 3D.
- WebKit: soporta un moderno navegador web utilizado en el navegador Android y en la vista webview. Se trata de la misma librería que utiliza Google Chrome y Safari de Apple.
- SGL: motor de gráficos 2D.

- Librerías 3D: implementación basada en OpenGL ES 1.0 API. Las librerías utilizan el acelerador hardware 3D si está disponible, o el software altamente optimizado de proyección 3D.
- FreeType: fuentes en bitmap y renderizado vectorial. • SQLite potente y ligero motor de bases de datos relaciónales disponible para todas las aplicaciones.
- SSL: proporciona servicios de encriptación Secure Socket.

6.2- Componentes de una aplicación

Para diseñar una aplicación en Android, es necesario tener claros los elementos que la componen y la funcionalidad de cada uno de ellos. Ya hemos visto el ejemplo del "Hola Android", por lo que podemos intuir algunos de ellos. Uno de los aspectos más importantes a tener en cuenta es su funcionamiento.

Android trabaja en Linux, y cada aplicación utiliza un proceso propio. Se distinguen por el ID, un identificador para que solo ella tenga acceso a sus archivos. Los dispositivos tienen un único foco, la ejecución principal, que es la aplicación que está visible en la pantalla, pero puede tener varias aplicaciones en un segundo plano, cada una con su propia pila de tareas. La pila de tareas es la secuencia de ejecución de procesos en Android.

Se componen de actividades que se van apilando según son invocadas, y solo pueden terminarse cuando las tareas que tiene encima están terminadas, o cuando el sistema las destruye porque necesita memoria, por lo que tienen que estar preparadas para terminar en cualquier momento. El sistema siempre eliminará la actividad que lleve más tiempo parada. En caso de que el sistema necesitase mucha memoria, si la aplicación no está en el foco, puedeser eliminada por completo a excepción de su actividad principal.

Ilustración 2 Actividades en Android

Una de las características principales del diseño en Android es la reutilización de componentes entre las aplicaciones, es decir, dos aplicaciones diferentes pueden utilizar una misma componente, aunque esté en otra aplicación para así, evitar la repetición innecesaria de código, y la consiguiente ocupación de espacio. Los componentes son los elementos básicos con los que se construye el proyecto. Hay cuatro tipos, pero las aplicaciones se componen principalmente de actividades. Habrá tantas actividades como ventanas distintas tenga la aplicación. Sin embargo, por si solos, los componentes no pueden hacer funcionar una aplicación. Para ello están los intentos.
Todos ellos deben declararse en el AndroidManifest.xml (junto con otros elementos que se mostrarán después) con el mismo nombre que lleve la clase asociada. Por ejemplo, la clase MainActivity, será definida en el AndroidManifest con el mismo nombre.

Actividades

Una actividad (o Activity) es el componente principal encargado de mostrar al usuario la interfaz gráfica, es decir, una actividad sería el equivalente a una ventana, y es el medio de comunicación entre la aplicación y el usuario. Se define una actividad por cada interfaz del proyecto. Los elementos que se muestran en ella deben ser definidos en el fichero XML que llevan asociado (que se guarda en ./res/layout) para poder ser tratados en la clase NameActivity.class, que hereda de la clase Activity.

Dentro del fichero XML asociado a la actividad, se definen los elementos tales como ubicación de los elementos en la pantalla (layouts), botones, textos, checkbox, etc. Las actividades tienen un ciclo de vida, es decir, pasan por diferentes estados desde que se inician hasta que se destruyen. Sus 3 posibles estados son:

Activo: ocurre cuando la actividad está en ejecución, es decir, es la tarea principal.

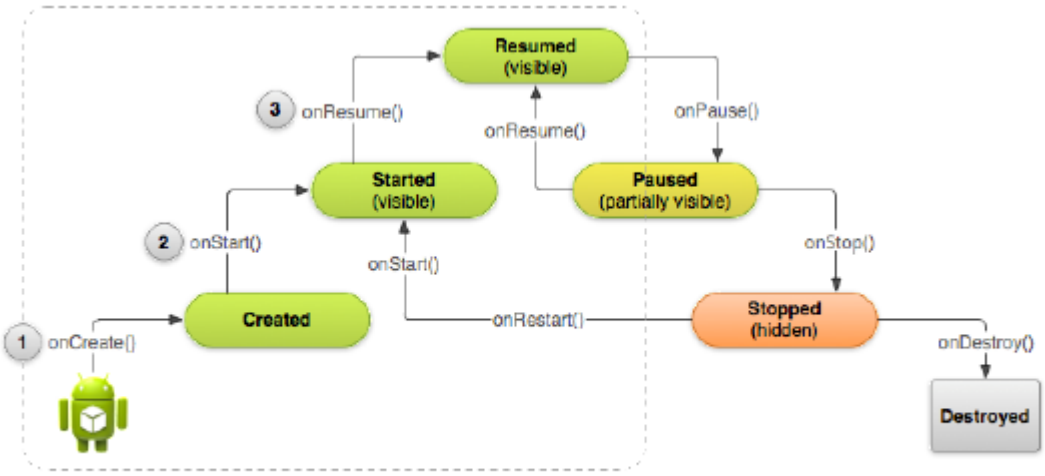
Pausado: la actividad se encuentra semi-suspendida, es decir, aún se está ejecutando y es visible, pero no es la tarea principal. Se debe guardar la información en este estado para prevenir una posible pérdida de datos en caso de que el sistema decida prescindir de ella para liberar memoria.

Parado: la actividad está detenida, no es visible al usuario y el sistema puede liberar memoria. En caso de necesitarla de nuevo, será reiniciada desde el principio.

Una vez definido el ciclo de vida, hay que tener en cuenta qué métodos son importantes en cada uno de ellos. Aquí están los métodos más importantes de una actividad:

- **OnCreate** (Bundle savedInstanceState): es el método que crea la actividad. Recibe un parámetro de tipo Bundle, que contiene el estado anterior de la actividad, para preservar la información que hubiera, en caso de que hubiera sido suspendida, aunque también puede iniciarse con un null si la información anterior no es necesaria o no existe.
- **OnRestart ():** reinicia una actividad tras haber sido parada (si continúa en la pila de tareas). Se inicia desde cero.
- **Onstart():** inmediatamente después de onCreate(Bundle savedInstanceState)
- **OnRestart ():** según corresponda. Muestra al usuario la actividad. Si ésta va a estar en un primer plano, el siguiente método debe ser onResume (). Si por el contrario se desarrolla por debajo, el método siguiente será onStop (). Es recomendable llamar al método onRestoreInstanceState() para asegurar la información
- **OnResume ():** establece el inicio de la interactividad entre el usuario y la aplicación. Solo se ejecuta cuando a actividad está en primer plano. Si necesita información previa, el método onRestoreInstanceState () aportará la situación en que estaba la actividad al llamar al onResume (). También puede guardar el estado con onSaveInstanceState ().
- **OnPause ():** se ejecuta cuando una actividad va a dejar de estar en primer plano, para dar paso a otra. Guarda la información, para poder restaurar cuando vuelva a estar activa en el método onSaveInstanceState (). Si la actividad vuelve a primer plano, el siguiente método será onResume (). En caso contrario, será onStop ().
- **OnStop ():** la actividad pasa a un segundo plano por un largo período. Como ya se ha dicho, el sistema puede liberar el espacio que ocupa, en caso de necesidad, o si la actividad lleva parada mucho tiempo.

- **OnDestroy ():** es el método final de la vida de una actividad. Se llama cuando ésta ya no es necesaria, o cuando se ha llamado al método finish ().



Ilustracion 3 Ciclo de vida de las actividades en un proyecto Android

Además de estos métodos, cabe destacar dos más, que son de vital importancia:

OnSavedInstanceState (): guarda el estado de una actividad. Es muy útil cuando se va a pausar una actividad para abrir otra

.

OnRestoreInstanceState (): restaura los datos guardados en onSavedInstanceState () al reiniciar una actividad. [3]

7- Android studio 3.0

Android Studio 3.0.0 es una versión importante que incluye una variedad de nuevas funciones y mejoras.

Complemento de Android para Gradle 3.0.0.

El nuevo complemento de Android para Gradle incluye una variedad de mejoras y nuevas características, pero mejora principalmente el rendimiento de compilación para proyectos que tienen una gran cantidad de módulos. Al usar el nuevo complemento con estos proyectos grandes, debe experimentar lo siguiente:

- Tiempos de configuración de construcción más rápidos debido a la nueva resolución de dependencia demorada.
- Resolución de dependencia con reconocimiento de variantes solo para los proyectos y variantes que está creando.
- Tiempos de construcción incrementales más rápidos al aplicar cambios simples al código o a los recursos.

Nota: Estas mejoras requieren cambios significativos que rompen algunos de los comportamientos, DSL y API del complemento. La actualización a la versión 3.0.0 puede requerir cambios en sus archivos de compilación y complementos de Gradle.

Esta versión también incluye lo siguiente:

- Soporte para Android 8.0.
- Soporte para construir archivos APK por separado basados en recursos de idiomas .
- Soporte para bibliotecas Java 8 y características de lenguaje Java 8 (sin el compilador Jack).
- Soporte para Android Test Support Library 1.0 (Android Test Utility y Android Test Orchestrator).
- Mejora de las velocidades de compilación ndk-build y cmake.
- Velocidad de sincronización de Gradle mejorada.

- AAPT2 ahora está habilitado por defecto.

7.2 - Soporte de Kotlin

Como se anunció en Google I / O 2017 , el lenguaje de programación Kotlin ahora es oficialmente compatible con Android. Por lo tanto, con este lanzamiento, Android Studio incluye el soporte de idioma de Kotlin para el desarrollo de Android.

7.3 - Android Profiler

El nuevo Android Profiler **reemplaza la herramienta Android Monitor** y proporciona un nuevo conjunto de herramientas para medir el uso de la CPU, la memoria y la red de su aplicación en tiempo real. Puede realizar el seguimiento de métodos basado en muestras para cronometrar la ejecución del código, capturar volcados de almacenamiento dinámico, ver asignaciones de memoria e inspeccionar los detalles de los archivos transmitidos por la red.

7.4 - Analizador de CPU

El Profiler de CPU le ayuda a analizar el uso de subprocesos de CPU de su aplicación activando un rastro de CPU instrumentado o de muestra. Luego, puede solucionar problemas de rendimiento de la CPU utilizando una variedad de vistas y filtros de datos.

7.5 - Memoria Profiler

Memory Profiler lo ayuda a identificar fugas de memoria y pérdida de memoria que pueden provocar tartamudeos, congelamientos e incluso bloqueos de la aplicación. Muestra un gráfico en tiempo real del uso de memoria de su aplicación, le permite capturar un volcado de pila, forzar colecciones de basura y rastrear asignaciones de memoria.

7.5.1- Network Profiler

Network Profiler le permite controlar la actividad de la red de su aplicación, inspeccionar la carga de cada una de sus solicitudes de red y volver a vincular el código que generó la solicitud de red.

Para obtener más información, consulte la guía Network Profiler .

7.6 - Perfil de APK y depuración

Android Studio ahora le permite **perfilar y depurar cualquier APK** sin tener que compilarlo desde un proyecto de Android Studio, siempre y cuando el APK esté diseñado para permitir la depuración y usted tenga acceso a los símbolos de depuración y los archivos fuente.

Para comenzar, haga clic en Perfil o depure APK desde la pantalla de bienvenida de Android Studio. O bien, si ya tiene un proyecto abierto, haga clic en Archivo> Perfil o depure APK desde la barra de menú. Esto muestra los archivos APK desempaquetados, pero no descompilar el código. Por lo tanto, para agregar correctamente los puntos de interrupción y ver los seguimientos de pila, debe adjuntar los archivos fuente de Java y los símbolos de depuración nativos.

7.7 - Soporte de aplicaciones instantáneas

El nuevo soporte para Android Instant Apps le permite crear aplicaciones instantáneas en su proyecto utilizando **dos nuevos tipos de módulos**: módulos de aplicación instantánea y módulos de funciones (estos requieren que instale el SDK de desarrollo de aplicaciones instantáneas).

7.8 - Módulos de cosas de Android

Nuevas plantillas de Cosas de Android en el Nuevo Proyecto y nuevos asistentes de Módulo para ayudarlo a comenzar a desarrollar dispositivos IOT con Android.

7.9 - Asistente de iconos adaptables

Image Asset Studio ahora es compatible con vectores arrastrables y le permite crear **iconos de iniciador adaptables para Android 8.0** al tiempo que crea iconos tradicionales (iconos "Legacy") para dispositivos más antiguos.

7.10 - Editor de diseño

El Editor de diseño se ha actualizado con una serie de mejoras, incluidas las siguientes:

- Nueva barra de herramientas de diseño e iconos.
- Diseño actualizado en el árbol de componentes.
- Mejoras en las inserciones de vistas de arrastrar y soltar.

- Nuevo panel de error debajo del editor, que muestra todos los problemas con sugerencias para corregir (si está disponible).
- Varias mejoras de UI para compilar ConstraintLayout, que incluyen las siguientes:
  - Nuevo apoyo para crear barreras .
  - Nuevo soporte para crear grupos: en la barra de herramientas, seleccione **Pautas> Agregar grupo** (requiere ConstraintLayout 1.1.0 beta 2 o superior)
  - Nueva interfaz de usuario para crear cadenas: seleccione varias vistas y luego haga clic con el botón derecho y seleccione **Cadena**.

7.11 - Inspector de diseño

El Inspector de diseño incluye mejoras para facilitar la depuración de problemas con los diseños de su aplicación, incluidas las propiedades de agrupación en categorías comunes y la nueva funcionalidad de búsqueda tanto en el **Árbol de visualización** como en los paneles de **Propiedades**.

8- Api Rest

Es una interfaz de programación de aplicaciones que se apoya en la arquitectura REST para el desarrollo de aplicaciones en red. Aprovechando el lenguaje HTML, permite que cualquier empresa cree aplicaciones web sin problemas, aunque siempre en base a las restricciones que supone.

- La transferencia de Estado Representacional (REST –Representational State Tranfer) fue ganando amplia adopción en toda la web como una alternativa más simple a SOAP y a los servicios web basados en el Lenguaje de Descripción de Servicios Web (Web Services Description Langiage – WSDL). Ya varios grandes proveedores de web 2.0 están migrando a esta tecnología, incluyendo a Yahoo!!!!!!, Google y Facebook, quienes marcaron como obsoletos a sus servicios SOAP y WSDL y pasaron a usar n modelo más fácil de usar, orientado a los recursos

- El lanzamiento del nuevo sistema REST como protocolo de intercambio y manipulación de datos en los servicios de internet cambió por completo el desarrollo de software a partir de 2000. Ya casi toda empresa o aplicación dispone de una API REST para creación de negocio.

- REST cambió por completo la ingeniería de software a partir del 2000. Este nuevo enfoque de desarrollo de proyectos y servicios web fue definido por **Roy Fielding**, el padre de la especificación HTTP y uno los referentes internacionales en todo lo relacionado con la Arquitectura de Redes, en su disertación 'Architectural Styles and the Design of Network-based Software Architectures'. En el campo de las APIs, REST (Representational State Transfer- Transferencia de Estado Representacional) es, a día de hoy, el alfa y omega del desarrollo de servicios de aplicaciones.

- En la actualidad no existe proyecto o aplicación que no disponga de una API REST para la creación de servicios profesionales a partir de ese software. Twitter, YouTube, los sistemas de identificación con Facebook… hay cientos de empresas que generan negocio gracias a REST y las APIs REST. Sin ellas, todo el crecimiento en horizontal sería prácticamente imposible. Esto es así porque REST es el estándar más lógico, eficiente y habitual en la creación de APIs para servicios de Internet.

- Buscando una definición sencilla, REST es cualquier interfaz entre sistemas que use HTTP para obtener datos o generar operaciones sobre esos datos en todos los formatos posibles, como XML y JSON. Es una alternativa en auge a otros protocolos estándar de intercambio de datos como SOAP (Simple Object Access Protocol), que disponen de una gran capacidad pero también mucha complejidad. A veces es preferible **una solución más sencilla de manipulación de datos como REST**.

4 principios de REST

- ❖ Utiliza los métodos de HTTP de manera explicita
- ❖ No mantiene estado
- ❖ Expone URls con forma de directorios
- ❖ Transfiere XML, JavaScript Object Natation (JSON) o ambos.

CARACTERÍSTICAS

- **Protocolo cliente/servidor sin estado**: cada petición HTTP contiene toda la información necesaria para ejecutarla, lo que permite que ni cliente ni servidor necesiten recordar ningún estado previo para satisfacerla. Aunque esto es así, algunas aplicaciones HTTP incorporan memoria caché. Se configura lo que se conoce como **protocolo cliente-caché-servidor sin estado**: existe la posibilidad de definir algunas respuestas a peticiones HTTP concretas como cacheables, con el objetivo de que el cliente pueda ejecutar en un futuro **la misma respuesta para peticiones idénticas**. De todas formas, que exista la posibilidad no significa que sea lo más recomendable.

- Las operaciones más importantes relacionadas con los datos en cualquier sistema REST y la especificación HTTP son cuatro: **POST** (crear), **GET** (leer y consultar), **PUT** (editar) y **DELETE** (eliminar).

- **Los objetos en REST siempre se manipulan a partir de la URI**. Es la URI y ningún otro elemento el identificador único de cada recurso de ese sistema REST. La URI nos facilita acceder a la información para su modificación o borrado, o, por ejemplo, para compartir su ubicación exacta con terceros.

- **Interfaz uniforme**: para la transferencia de datos en un sistema REST, este aplica acciones concretas (POST, GET, PUT y DELETE) sobre los recursos, siempre y cuando estén identificados con una URI. Esto facilita la existencia de una interfaz uniforme que sistematiza el proceso con la información.

- **Sistema de capas**: arquitectura jerárquica entre los componentes. Cada una de estas capas lleva a cabo una funcionalidad dentro del sistema REST.

- **Uso de hipermedios**: hipermedia es un término acuñado por **Ted Nelson** en 1965 y que es una extensión del concepto de hipertexto. Ese concepto llevado al desarrollo de páginas web es lo que permite que el usuario puede navegar por el conjunto de objetos a través de enlaces HTML. En el caso de una API REST, el concepto de hipermedia explica la capacidad de una interfaz de desarrollo de aplicaciones de proporcionar al cliente y al usuario los enlaces adecuados para ejecutar acciones concretas sobre los datos.

- Para cualquier API REST es obligatorio disponer del principio **HATEOAS** (Hypermedia As The Engine Of Application State - Hipermedia Como Motor del Estado de la Aplicación) para ser una verdadera API REST. Este principio es el que define que cada vez que se hace una petición al servidor y éste devuelve una respuesta, parte de la información que contendrá serán los hipervínculos de navegación asociada a otros recursos del cliente.

VENTAJAS QUE OFRECE REST PARA EL DESARROLLO

1. **Separación entre el cliente y el servidor**: el protocolo REST separa totalmente la interfaz de usuario del servidor y el almacenamiento de datos. Eso tiene algunas ventajas cuando se hacen desarrollos. Por ejemplo, mejora la portabilidad de la interfaz a otro tipo de plataformas, aumenta la escalabilidad de los proyectos y permite que los distintos componentes de los desarrollos se puedan evolucionar de forma independiente.

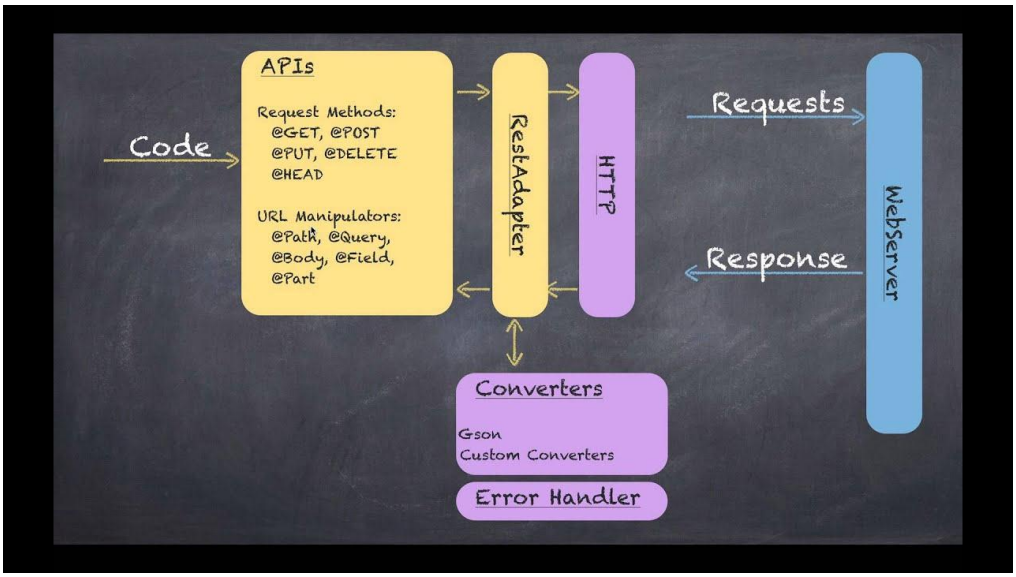
2. **Visibilidad, fiabilidad y escalabilidad**. La separación entre cliente y servidor tiene una ventaja evidente y es que cualquier equipo de desarrollo puede escalar el producto sin excesivos problemas. Se puede migrar a otros servidores o realizar todo tipo de cambios en la base de datos, siempre y cuando los datos de cada una de las peticiones se envíen de

forma correcta. Esta separación facilita tener en servidores distintos el *front* y el *back* y eso convierte a las aplicaciones en productos más flexibles a la hora de trabajar.

3. **La API REST siempre es independiente del tipo de plataformas o lenguajes**: la API REST siempre se adapta al tipo de sintaxis o plataformas con las que se estén trabajando, lo que ofrece una gran libertad a la hora de cambiar o probar nuevos entornos dentro del desarrollo. Con una API REST se pueden tener servidores PHP, Java, Python o Node.js. Lo único que es indispensable es que las respuestas a las peticiones se hagan siempre en el lenguaje de intercambio de información usado, normalmente XML o JSON.

9- Retrofit

Retrofit es un cliente REST para Android y Java, desarrollado por Square. Permite hacer peticiones GET, POST, PUT, PATCH, DELETE y HEAD y descargar datos en formato JSON o XML de manera sencilla. Una vez descargado se pasan en un POJO para poder tratar los datos en la aplicación.



Importar Retrofit

Para poder importar **Retrofit 2.0** en nuestro proyecto debemos añadir en el apartado dependencies del fichero *build.gradle* lo siguiente:

Compile 'com.squareup.retrofit2: retrofit: 2.0.2'

Sincronizamos Gradle y ¡listo!

En realidad no estaría todo listo ya que a diferencia de Retrofit 1.0 que venía con el conversor GSON por defecto, en la v2 debemos también importar el conversor que vamos a necesitar.

En la siguiente tabla vemos los diferentes tipos de conversor que existen:

CONVERSOR	Librería a importar
Gson	com.squareup.retrofit2:converter-gson:2.0.2
Jackson	com.squareup.retrofit2:converter-jackson:2.0.2
Moshi	com.squareup.retrofit2:converter-moshi:2.0.2
Protobuf	com.squareup.retrofit2:converter-protobuf:2.0.2
Wire	com.squareup.retrofit2:converter-wire:2.0.2
Simple XML	com.squareup.retrofit2:converter-simplexml:2.0.2

Primeros pasos

El primer paso es añadir la dependencia en nuestro proyecto via

MAVEN

```
<groupId>com.squareup.retrofit</groupId>

<ArtifactId>retrofit</artifactId>

<version>1.9.0</version>
```

GRADLE

```
Compile 'com.squareup.retrofit: retrofit: 1.9.0’
```

Código

A continuación necesitamos 3 clases para poder usar Retrofit

- POJO: Representación del Json o XML preveniente de la API REST
- Interface (Service): Definición de las diferentes llamadas a la API REST
- RestAdapter: Cliente que usaremos para realizar y configurar nuestras peticiones.
- Retrofit permite hacer las peticiones de manera síncrona o asíncrona, así como usando RxJava (para uso en Desktop).

Esta configuración se realiza mediante la respuesta del método, o añadiendo el parámetro Callback al final de la petición.

Síncrono

```
public interface GithubRepositoryService {

   @GET("/repos/{owner}/{name}")

   Repository repository(@Path("owner") String owner, @Path("name") String name);

}
```

Asíncrono

```
public interface GithubRepositoryService {

   @GET("/repos/{owner}/{name}")

   void repository(@Path("owner") String owner, @Path("name") String name,  Callback<R
epository> callback);

}
```

No se permite usar los dos tipos a la vez

Ejecución

Y a continuación creamos la instancia de RestAdapter y ejecutamos la petición.

```
public Repository getRepositorySync(String owner, String name) {

    RestAdapter restAdapter = new RestAdapter.Builder()

   .setEndpoint("https://api.github.com")

   .build();

   GithubRepositoryService service = restAdapter.create(GithubRepositoryService.class);

   return service.repository(owner, name);

}
```

Esté método solo nos provee del objeto Parseado, no podemos acceder a la respuesta HTTP en raw, o los headers.

```
 public void getRepositoryAsync(String owner, String name) {
```

```
RestAdapter restAdapter = new RestAdapter.Builder()
   .setEndpoint("https://api.github.com")
   .build();
   GithubRepositoryService service = restAdapter.create(GithubRepositoryService.class);
Callback callback = new Callback {
      @Override
      public void success(Repository repository, final Response response) { ... }


      @Override
      public void failure(final RetrofitError error) { ... }
   }
   service.repository(owner, name, callback);
}
```

El método succes se ejecutará si todo ha funcionado bien, y nos devuelve el Objeto parseado, y un Response, que contiene la respuesta HTTP en Raw, así como el código HTTP, los headers, etc…

Para hacer peticiones POST, PUT, PATCH y DELETE haremos exactamente lo mismo, crear la interface con sus anotaciones @POST, @DELETE… y llamar a RestAdapter.

10- Java

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 diez millones de usuarios reportados.

El lenguaje de programación Java fue originalmente desarrollado por James Gosling, de Sun Microsystems (constituida en 1982 y posteriormente adquirida el 27 de enero de 2010 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 compiladas a bytecode (clase Java), que puede ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente.

10.1-Orientado 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.

10.2 - Independencia de la plataforma

La segunda característica, la independencia de la plataforma, 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.

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 sólo puede ejecutarse en un tipo de arquitectura.

La portabilidad es técnicamente difícil de lograr, y el éxito de Java en ese campo ha sido dispar. Aunque es de hecho posible escribir programas para la plataforma Java que actúen de forma correcta en múltiples plataformas de distinta arquitectura, el gran número de estas con pequeños errores o inconsistencias llevan a que a veces se parodie el eslogan de Sun, "Write once, run anywhere" como "Write once, debug everywhere" (o "Escríbelo una vez, ejecútalo en cualquier parte" por "Escríbelo una vez, depúralo en todas partes").

El concepto de independencia de la plataforma de Java cuenta, sin embargo, con un gran éxito en las aplicaciones en el entorno del servidor, como los Servicios Web, los Servlets, los Java Beans, así como en sistemas empotrados basados en OSGi, usando entornos Java empotrados.

10.3 - El recolector de basura

En Java el problema de fugas de memoria se evita en gran medida gracias a la recolección de basura (o *automatic garbage collector*). El programador determina cuándo se crean los objetos, y el entorno, en tiempo de ejecución de Java (Java runtime), es el responsable de gestionar el ciclo de vida de los objetos. El programa, u otros objetos, pueden tener localizado un objeto mediante una referencia a éste. Cuando no quedan referencias a un objeto, el recolector de basura de Java borra el objeto, liberando así la memoria que ocupaba previniendo posibles fugas (ejemplo: un objeto creado y únicamente usado dentro de un método sólo tiene entidad dentro de éste; al salir del método el objeto es eliminado). Aun así, es posible que se produzcan fugas de memoria si el código almacena referencias a objetos que ya no son necesarios; es decir, pueden aún ocurrir, pero en un nivel conceptual superior. En definitiva, el recolector de basura de Java permite una fácil creación y eliminación de objetos y mayor seguridad.

10.4 - Aplicaciones autónomas

Este ejemplo necesita una pequeña explicación.

- Todo en Java está dentro de una clase, incluyendo programas autónomos.
- El código fuente se guarda en archivos con el mismo nombre que la clase que contienen y con extensión ".java". Una clase (`class`) declarada pública (`public`) debe seguir este convenio. Si tenemos una clase llamada `Hola`, su código fuente deberá guardarse en el fichero "Hola.java".
- El compilador genera un archivo de clase (con extensión ".class") por cada una de las clases definidas en el archivo fuente. Una clase anónima se trata como si su nombre fuera la concatenación del nombre de la clase que la encierra, el símbolo "$", y un número entero.
- Los programas que se ejecutan de forma independiente y autónoma, deben contener el método `"main()"`.
- La palabra reservada`"void"` indica que el método main no devuelve nada.

- El método main debe aceptar un array de objetos tipo String. Por acuerdo se referencia como"args", aunque puede emplearse cualquier otro identificador.

- La palabra reservada"static" indica que el método es un método de clase, asociado a la clase en vez de a una instancia de la misma. El método main debe ser estático o''de clase''.
- La palabra reservada **public** significa que un método puede ser llamado desde otras clases, o que la clase puede ser usada por clases fuera de la jerarquía de la propia clase. Otros tipos de acceso son"private" o"protected".
- La utilidad de impresión (en pantalla, por ejemplo) forma parte de la biblioteca estándar de Java: la clase '''System''' define un campo público estático llamado '''out'''. El objeto out es una instancia de '''PrintStream''', que ofrece el método '''println (String) ''' para volcar datos en la pantalla (la salida estándar).

- Las aplicaciones autónomas se ejecutan dando al entorno de ejecución de Java el nombre de la clase cuyo método main debe invocarse. Por ejemplo, una línea de comando (en Unix o Windows) de la forma java –cp. Hola ejecutará el programa del ejemplo (previamente compilado y generado "Hola.class"). El nombre de la clase cuyo método main se llama puede especificarse también en el fichero "MANIFEST" del archivo de empaquetamiento de Java (.jar).

•

10.6 - Applets

Las applet Java son programas incrustados en otras aplicaciones, normalmente una página web que se muestra en un navegador.

Actualmente HTML 5 ha eliminado el uso de la etiqueta <applet>. Pero todavía existe la forma de usarlo en HTML5

10.6 – Servlets

Los servlets son componentes de la parte del servidor de Java EE utilizadas para ampliar las capacidades del servidor. Encargados de generar respuestas a las peticiones recibidas de los clientes.

10.7 - Aplicaciones con interfaz

Swing es una biblioteca gráfica para Java. Incluye widgets para interfaz gráfica de usuario tales como cajas de texto, botones, listas desplegables y tablas.

10.8 - Entornos de funcionamiento

10.8.1 - En dispositivos móviles y sistemas embebidos

Desde la creación de la especificación J2ME (Java 2 Platform, Micro Edition), una versión del entorno de ejecución Java reducido y altamente optimizado, especialmente desarrollado para el mercado de dispositivos electrónicos de consumo, se ha producido toda una revolución en lo que a la extensión de Java se refiere.

Es posible encontrar microprocesadores diseñados para ejecutar bytecode Java y software Java para tarjetas inteligentes (JavaCard), teléfonos móviles, buscapersonas, set-top-boxes, sintonizadores de TV y otros pequeños electrodomésticos.

El modelo de desarrollo de estas aplicaciones es muy semejante a las *applets* de los navegadores, salvo que en este caso se denominan MIDlets.

10.8.2 - En el navegador web

Desde la primera versión de Java existe la posibilidad de desarrollar pequeñas aplicaciones (Applets) en Java que luego pueden ser incrustadas en una página HTML para que sean descargadas y ejecutadas por el navegador web. Estas mini aplicaciones se ejecutan en una JVM que el navegador tiene configurada como extensión (*plug-in*) en un contexto de seguridad restringido configurable para impedir la ejecución local de código potencialmente malicioso.

El éxito de este tipo de aplicaciones (la visión del equipo de Gosling) no fue realmente el esperado debido a diversos factores, siendo quizás el más importante la lentitud y el reducido ancho de banda de las comunicaciones en aquel entonces que limitaba el tamaño de las applets que se incrustaban en el navegador. La aparición posterior de otras alternativas (aplicaciones web dinámicas de servidor) dejó un reducido ámbito de uso para esta tecnología, quedando hoy relegada fundamentalmente a componentes específicos para la intermediación desde una aplicación web dinámica de servidor con dispositivos ubicados en la máquina cliente donde se ejecuta el navegador.

10.8.3 - En sistemas de servidor

En la parte del servidor, Java es más popular que nunca, desde la aparición de la especificación de Servlets y JSP (Java Server Pages).

Hasta entonces, las aplicaciones web dinámicas de servidor que existían se basaban fundamentalmente en componentes CGI y lenguajes interpretados. Ambos tenían diversos inconvenientes (fundamentalmente lentitud, elevada carga computacional o de memoria y propensión a errores por su interpretación dinámica).

Los servlets y las JSP supusieron un importante avance ya que:

- El API de programación es muy sencilla, flexible y extensible.
    - Los servlets no son procesos independientes (como los CGI) y por tanto se ejecutan dentro del mismo proceso que la JVM mejorando notablemente el rendimiento y reduciendo la carga computacional y de memoria requeridas.

- Las JSP son páginas que se compilan dinámicamente (o se pre compilan previamente a su distribución) de modo que el código que se consigue supone una ventaja en rendimiento substancial frente a muchos lenguajes interpretados.

  La especificación de Servlets y JSP define un API de programación y los requisitos para un contenedor (servidor) dentro del cual se puedan desplegar estos componentes para formar aplicaciones web dinámicas completas. Hoy día existen multitud de contenedores (libres y comerciales) compatibles con estas especificaciones.

  A partir de su expansión entre la comunidad de desarrolladores, estas tecnologías han dado paso a modelos de desarrollo mucho más elaborados con frameworks (pe Struts, Webwork) que se sobreponen sobre los servlets y las JSP para conseguir un entorno de trabajo mucho más poderoso y segmentado en el que la especialización de roles sea posible (desarrolladores, diseñadores gráficos,...) y se facilite la reutilización y robustez de código. A pesar de todo ello, las tecnologías que subyacen (Servlets y JSP) son substancialmente las mismas.

  Este modelo de trabajo se ha convertido en uno de los estándares *de facto* para el desarrollo de aplicaciones web dinámicas de servidor.

10.8.4 - En aplicaciones de escritorio

Hoy en día existen multitud de aplicaciones gráficas de usuario basadas en Java. El entorno de ejecución Java (JRE) se ha convertido en un componente habitual en los PC de usuario de los sistemas operativos más usados en el mundo. Además, muchas aplicaciones Java lo incluyen dentro del propio paquete de la aplicación de modo que se ejecuten en cualquier PC.

En las primeras versiones de la plataforma Java existían importantes limitaciones en las API de desarrollo gráfico (AWT). Desde la aparición de la biblioteca Swing la situación mejoró substancialmente y posteriormente con la aparición de bibliotecas como SWT hacen que el desarrollo de aplicaciones de escritorio complejas y con gran dinamismo, usabilidad, etc. sea relativamente sencillo.

10.8.5 - Plataformas soportadas

Una versión del entorno de ejecución Java JRE (Java Runtime Environment) está disponible en la mayoría de equipos de escritorio. Sin embargo, Microsoft no lo ha incluido por defecto en sus sistemas operativos. En el caso de Apple, éste incluye una versión propia del JRE en su sistema operativo, el Mac OS. También es un producto que por defecto aparece en la mayoría de las distribuciones de GNU/Linux. Debido a incompatibilidades entre distintas versiones del JRE, muchas aplicaciones prefieren instalar su propia copia del JRE antes que confiar su suerte a la aplicación instalada por defecto. Los desarrolladores de applets de Java o bien deben insistir a los usuarios en la actualización del JRE, o bien desarrollar bajo una versión antigua de Java y verificar el correcto funcionamiento en las versiones posteriores.

11- SQLite

Es una biblioteca escrita en leguaje C que implementa un Sistema de gestión de bases de datos transaccionales SQL auto-contenido, sin servidor y sin configuración. El código de SQLite es de dominio público y libre para cualquier uso, ya sea comercial o privado. Actualmente es utilizado en gran cantidad de aplicaciones incluyendo algunas desarrolladas como proyectos de alto nivel.

11.1 - Caracteristicas

SQLite es un sistema completo de bases de datos que soporta múltiples tablas, índices, triggers y vistas. No necesita un proceso separado funcionando como servidor ya que lee y escribe directamente sobre archivos que se encuentran en el disco duro. El formato de la base de datos es multiplataforma e indistintamente se puede utilizar el mismo archivo en sistemas de 32 y 64 bits.

La base de datos se almacena en un único fichero a diferencia de otros DBMS que hacen uso de varios archivos. SQLite emplea registros de tamaño variable de forma tal que se utiliza el espacio en disco que es realmente necesario en cada momento.

El código fuente está pensado para que sea entendido y accesible por programadores promedio. Todas las funciones y estructuras están bien documentadas.

Existe un programa independiente de nombre sqlite que puede ser utilizado para consultar y gestionar los ficheros de base de datos SQLite. También sirve como ejemplo para la escritura de aplicaciones utilizando la biblioteca SQLite.

11.2 - Implementación de SQL

La biblioteca implementa la mayor parte del estándar SQL-92, incluyendo transacciones de base de datos atómicas, consistencia de base de datos, aislamiento, y durabilidad (ACID), triggers y la mayor parte de las consultas complejas.

11.3 - Características omitidas de SQL

- Restricciones FOREIGN KEY,
- Soporte completo para triggers (disparadores)
- Soporte completo para ALTER TABLE, solamente implementa las instrucciones RENAME TABLE y ADD COLUMN.
- RIGHT y FULL OUTER JOIN, sólo está implementada la instrucción LEFT OUTER JOIN.
- Escribir en VIEWs, ya que las vistas en SQLite son de sólo lectura.
- GRANT y REVOKE, pues no tienen sentido en un sistema de bases de datos embebido.

11.4 - Tipos de datos

SQLite usa un sistema de tipos inusual. En lugar de asignar un tipo a una columna como en la mayor parte de los sistemas de bases de datos SQL, los tipos se asignan a los valores individuales.

Cada uno de los datos almacenados en una base de datos SQLite implemtenta alguno de los siguientes tipos:

- NULL, un valor nulo.
- INTEGER, un entero con signo que se almacena en 1, 2, 3, 4, 5, 6 o 8 bytes de acuerdo a la magnitud del valor.
- REAL, un número de coma flotante (real), almacenado en 8 bytes.
- TEXT, una cadena de texto almacenada con las codificaciones UTF-8, UTF-16BE o UTF-16-LE.
- BLOB, datos en formato binario, se almacenan exactamente como se introdujeron.

11.4.1 - Autoincremento

Una duda tradicional es el modo de implementar el autoincremento, principalmente para las llaves primarias. La forma más sencilla es declarar el tipo de dato como INTEGER PRIMARY KEY.

11.4.2 - Concurrencia

Varios procesos o hilos pueden acceder a la misma base de datos sin problemas. Varios accesos de lectura pueden ser servidos en paralelo. Un acceso de escritura sólo puede ser servido si no se está sirviendo ningún otro acceso concurrentemente.

11.5 - Ventajas

- Tamaño: SQLite tiene una pequeña memoria y una única biblioteca es necesaria para acceder a bases de datos, lo que lo hace ideal para aplicaciones de bases de datos incorporadas.
- Rendimiento de base de datos: SQLite realiza operaciones de manera eficiente y es más rápido que MySQL y PostgreSQL.
- Portabilidad: se ejecuta en muchas plataformas y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración.
- Estabilidad: SQLite es compatible con ACID, reunión de los cuatro criterios de Atomicidad, Consistencia, Aislamiento y Durabilidad.
- SQL: implementa un gran subconjunto de la ANSI – 92 SQL estándar, incluyendo sub-consultas, generación de usuarios, vistas y triggers.
- Interfaces: cuenta con diferentes interfaces del API, las cuales permiten trabajar con C++, PHP, Perl, Python, Ruby, Tcl, Groovy, Qt ofrece el plugin qsqlite, etc.
- Costo: SQLite es de dominio público, y por tanto, es libre de utilizar para cualquier propósito sin costo y se puede redistribuir libremente.

11.6 - Usos aconsejados de SQLite

- Formato de archivo de aplicaciones.
- Aplicaciones desktop.
- Bases de datos para dispositivos.
- Bases de datos de sitios web de pequeño y mediano tamaño.
- Enseñanza.

11.7 - Usos no aconsejados de SQLite

- Aplicaciones Cliente-servidor.
- Sitios web con gran cantidad de transacciones.
- Bases de datos muy grandes (SQLite soporta hasta 2 terabytes).
- Alta concurrencia.

11. 8 - Aplicaciones que utilizan SQLite

- Aplicaciones Cliente-servidor.
- Sitios web con gran cantidad de transacciones.
- Bases de datos muy grandes (SQLite soporta hasta 2 terabytes).
- Alta concurrencia.

11.8 – Ejemplos (ver anexos pag 85).

12- Diseño metodológico

Para realizar el presente trabajo primero se necesita conocer cómo funciona un traductor bilingüe.
Este traductor y Diccionario bilingüe se puede automatizar utilizando sistemas de información para el fácil acceso ya que los diccionarios en esta lengua son muy difíciles de obtener físicamente y hay poca información de este en internet.

Para estos casos conviene usar dispositivos portátiles (Teléfonos inteligentes, Tablet) ya que pueden estar sin conexión a energía por muchas horas, almacenar una gran cantidad de datos, además son más fáciles de manipular y trasladar.

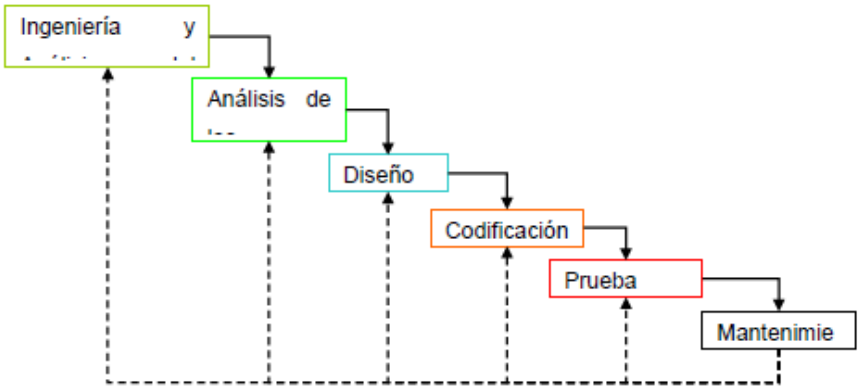
Se consultará libros y diccionarios propios de la lengua mayangna y apoyados por una persona nativa de la zona para documentar y extraer toda la información necesaria para realizar el proyecto Android. Toda esa información se almacenará inicialmente en la base de datos del dispositivo mediante inserciones durante la programación de la app.

Con esta información previamente insertada el usuario podrá consultar y agregar nueva información o actualizar la base de datos mediante el uso de servicios web que estarán disponible con solo estar conectado a internet apoyándonos de la librería Retrofit de Android.

De esta manera tendremos una idea clara de cómo automatizar estos procesos. Una vez que tengamos claro el funcionamiento del diccionario, se procederá a realizar un documento, que será una guía a la hora de desarrollar la aplicación.

Luego se procederá a diseñar las interfaces tomando como referencia la especificación de requisito.

A partir de las interfaces se iniciará con la codificación de aplicación. El desarrollo de la aplicación se realizará siguiendo el modelo en cascada, el paradigma del ciclo de vida abarca las siguientes actividades.



Ilustracion 4 Ciclo de vida en cascada

Debido a que el software es siempre parte de un sistema mayor el trabajo comienza estableciendo los requisitos de todos los elementos del sistema y luego asignando algún subconjunto de estos requisitos al software.

Análisis de los requisitos del software: el proceso de recopilación de los requisitos se centra e intensifica especialmente en el software. El ingeniero de software (Analistas) debe comprender el ámbito de la información del software, así como la función, el rendimiento y las interfaces requerida.

Diseño: el diseño del software se enfoca en cuatro atributos distintos del programa: la estructura de los datos, la arquitectura del software, el detalle procedimental y la caracterización de la interfaz. El proceso de diseño traduce los requisitos en una representación del software con la calidad requerida antes de que comience la codificación.

Codificación: el diseño debe traducirse en una forma legible para la máquina. El paso de codificación realiza esta tarea. Si el diseño se realiza de una manera detallada la codificación puede realizarse mecánicamente.

Prueba: una vez que se ha generado el código comienza la prueba del programa. La prueba se centra en la lógica interna del software, y en las funciones externas, realizando pruebas que aseguren que la entrada definida produce los resultados que realmente se requieren.

Mantenimiento: el software sufrirá cambios después de que se entrega al cliente. Los cambios ocurrirán debido a que hayan encontrado errores, a que el software deba adaptarse a cambios del entorno externo (sistema operativo o dispositivos periféricos), o debido a que el cliente requiera ampliaciones funcionales o del rendimiento.

Una vez desarrollada la aplicación se debe poner a prueba en un entorno real, el cual servirá para identificar algún requerimiento no especificado o error en diseño o código.

El modelo en cascada, a veces llamado ciclo de vida clásico, sugiere un enfoque sistemático y secuencial para el desarrollo del software, que comienza con la especificación de los requerimientos por parte del cliente y avanza a través de planeación, modelado, construcción y despliegue, para concluir con el apoyo del software terminado.

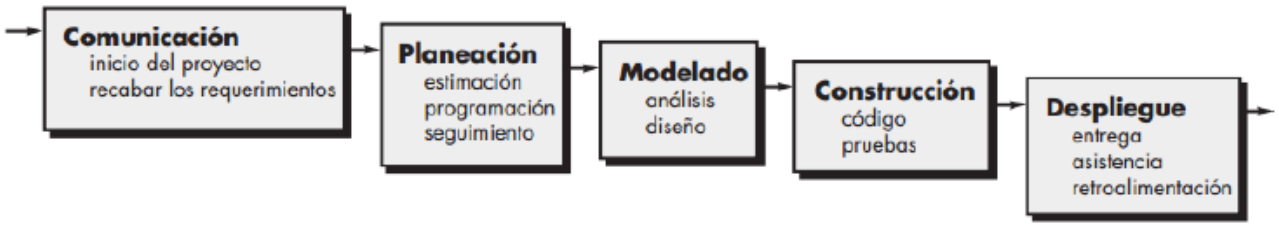


Figura: Modelo en cascada

Una variante de la representación del modelo en cascada se denomina modelo en V. donde se aprecia la relación entre las acciones para el aseguramiento de la calidad y aquellas asociadas con la comunicación, modelado y construcción temprana.

11- Especificación de Requisito Software (ERS)

11.1- Introducción

El siguiente apartado especifica las funciones que tendrá el sistema, así como el alcance del mismo.

11.1.1- Propósito

La especificación de requisito software tiene como propósito definir las especificaciones funcionales, no funcionales del sistema para una aplicación móvil que permitirá la gestión de la información del diccionario y traductor bilingüe Español-Mayangna, la cual será usada por los estudiantes y personas en general.

11.1.2- Alcance del sistema

Los estudiantes de la carrera de ingeniería han tenido la necesidad de crear una app para traducir palabras y frases del idioma mayangna al español y viceversa. Por lo tanto, el objetivo del presente trabajo es desarrollar una aplicación móvil que permita mostrar los datos de manera rápida y segura, a través de interfaces intuitivas. A la vez que permita a las personas aprender un poco más sobre la cultura Mayangna.

11.1.3- Definiciones, Acrónimos y Abreviaturas

En el siguiente apartado se especificará y definirá todos los términos usados en la especificación de requisito.

Definición.

Diccionario: Obra de consulta en que se recoge y se define o traduce, generalmente en orden alfabético, un conjunto de palabras de una o más lenguas o de una materia determinada.
Bilingüe: Que habla o emplea dos lenguas con igual perfección.
Mayangna: Los mayangna es un pueblo indígena, también conocido como sumu o sumo, que habitan a lo largo de los ríos Waspuk o Huaspuc, Pispís y Bocay, en el nororiente de Nicaragua.
Interfaces: formularios o ventanas que permiten a los usuarios interactuar con un dispositivo electrónico e ingresar datos a través de la solución diseñada.

11.1.4- Visión general del documento

Este documento consta de tres secciones. En la primera sección se realiza una introducción al mismo y se proporciona una visión general de la especificación de recursos del sistema.

En la segunda sección del documento se realiza una descripción general del sistema, con el fin de conocer las principales funciones que éste debe realizar, los datos asociados y los factores, restricciones, supuestos y dependencias que afectan al desarrollo, sin entrar en excesivos detalles.

Por último, la tercera sección del documento es aquella en la que se definen detalladamente los requerimientos que debe satisfacer el sistema.

11.2- Descripción General

Se describirán las funciones que tendrán el sistema, origen de dato de cada funcionalidad, el proceso a seguir y la salida que generara cada función.

11.2.1- Perspectiva del Producto

Este producto se utilizará para realizar traducciones del español a mayangna y viceversa, así como visualizar, agregar, eliminar y actualizar información mediante el uso de servicios web y que el usuario conozca sobre el origen de esta lengua nativa de la zona norte de Nicaragua.

11.2.2- Funciones del Producto

La aplicación móvil permitirá realizar lo siguiente:

- Traducir de español a mayangna.
- Traducir de mayangna a español.
- Agregar nuevas palabras.
- Agregar nuevas frases.
- Eliminar palabras.
- Eliminar frases.
- Actualizar frases y palabras usando servicios web.
- Mostrar información del lenguaje Mayangna (cultura, reseña histórica y lenguaje).
- Visualizar diccionario.
- Visualizar frases populares.

11.2.3- Características de los usuarios

El sistema estará orientado a usuarios que se están familiarizados a trabajar con dispositivos móviles con pantalla táctil. No se necesitará ningún conocimiento avanzado del dispositivo.

11.2.4- Restricciones

1. La aplicación sólo se podrá ejecutará bajo dispositivos con sistema operativo Android.

2. El dispositivo sólo puede trabajar una cierta cantidad de horas sin corriente eléctrica externa, luego tiene que estar conectado a una fuente de energía para trabajar correctamente.

3. Para mayor eficiencia de la aplicación el dispositivo debe de estar conectado a internet.

11.3- Requisitos Específicos

A continuación, se detallarán todas las funciones que tendrá la aplicación móvil.

RF1: Traducir de español a mayangna.

RF2: Traducir de mayangna a español.

RF3: Eliminar palabras.

RF4: Eliminar frases.

RF5: Agregar palabras.

RF6: Agregar frases.

RF7: Actualizar palabras.

RF8: Actualizar frases.

RF9: Mostrar información del lenguaje Mayangna.

RF10: Visualizar diccionario.

RF11: Visualizar frases populares.

11.3.1- Requisitos Comunes de las interfaces

A continuación, se especificarán los requisitos de usuario, hardware, software y comunicación con que debe contar la aplicación para su correcto uso.

11.3.1.1- Interfaces de usuario

Las interfaces de usuario están relacionadas con las pantallas, ventanas (formularios) que debe manipular el usuario para realizar una operación determinada. Dicha manipulación se realizará por medio de la pantalla táctil del dispositivo.

11.3.1.2- Interfaces de hardware

- Pantalla táctil del dispositivo: Todas las funcionalidades de la App se realizarán por medio de la pantalla táctil del dispositivo.

- Servidor de base de datos, necesario para que la aplicación funcione correctamente.

11.3.1.3- Interfaces de software

- Acceso a internet

11.3.1.4- Interfaces de comunicación

Esta aplicación móvil deberá guardar registros en una base de datos SQLite para consultar datos de la aplicación y hará consulta a los servicios web de para actualizar los registros guardados por los usuarios de la aplicación móvil.

11.3.2- Requisitos Funcionales

A continuación, se describirán a detalle las funciones que tendrá la aplicación. Cada una de estas funciones está representada con un único requisito donde se detalla toda la información necesaria sobre la función que realizará el sistema.

11.3.2.1- Requisito funcional 1

Número del requisito	RF1
Nombre del requisito	Traducir de español a mayangna
Tipo	Requisito
Fuente del requisito	Tabla

Introducción: El sistema permitirá la entrada de datos para visualizar la palabra en mayangna.

Entrada: Una cadena de caracteres que indicara la palabra en español.

Salida: La palabra en Mayangna.

11.3.2.2- Requisito funcional 2

Número del requisito	RF2
Nombre del requisito	Traducir de mayangna a español
Tipo	Requisito
Fuente del requisito	Tabla

Introducción: el sistema permitirá la entrada de datos para mostrar la palabra en español.

Entrada:

Una cadena de caracteres que indicara la palabra en Mayangna.

Salida:

La palabra en español.

11.3.2.3- Requisito funcional 3

Número del requisito	RF3
Nombre del requisito	Eliminar Palabras.
Tipo	Requisito
Fuente del requisito	

Introducción: El sistema contará con un cuadro de diálogo que se activará por medio de eventos, que le permitirá al usuario eliminar registro de la base de datos.

Entrada:

Una pulsación larga sobre la palabra que desea eliminar.

Salida:

Se activará un cuadro de dialogo que confirmara la eliminación de la palabra seleccionada y se enviara un mensaje que indicara que la palabra fue eliminada de la base de datos.

11.3.2.4- Requisito funcional 4

Número del requisito	RF4
Nombre del requisito	Eliminar Frases
Tipo	Requisito
Fuente del requisito	

Introducción: El sistema contará con un cuadro de diálogo que se activará por medio de eventos, que le permitirá al usuario eliminar registro de la base de datos.

Entrada:

Una pulsación larga sobre la palabra que desea eliminar.

Salida:

Se activará un cuadro de dialogo que confirmara la eliminación de la frase seleccionada y se enviara un mensaje que indicara que la frase fue eliminada de la base de datos.

11.3.2.5- Requisito funcional 5

Número del requisito	RF5
Nombre del requisito	Agregar palabras
Tipo	Requisito
Fuente del requisito	

Introducción: El sistema contará con un formulario en el cual el usuario puede ir agregando nuevas palabras a la base de datos local o servicio web.

Entrada:

Una cadena de caracteres que indicara la palabra en español.

Una cadena de caracteres que indicará la traducción de la palabra en español a Mayangna.

Salida:

Un mensaje que indica que el registro fue ingresado correctamente; si el dispositivo no está conectado a internet se enviara un mensaje que debe conectarse a internet para agregarla al servicio web sino se guardara en la base de dato local.

11.3.2.6 - Requisito funcional 6

Número del requisito	RF6
Nombre del requisito	Agregar Frases
Tipo	Requisito
Fuente del requisito	

Introducción: El sistema contará con un formulario en el cual el usuario puede ir agregando nuevas palabras a la base de datos local o servicio web.

Entrada:

Una cadena de caracteres que indicara la palabra en español.

Una cadena de caracteres que indicará la traducción de la palabra en español a Mayangna.

Salida:

Un mensaje que indica que el registro fue ingresado correctamente; si el dispositivo no está conectado a internet se enviara un mensaje que debe conectarse a internet para agregarla al servicio web sino se guardara en la base de dato local.

11.3.2.7 - Requisito funcional 7

Número del requisito	RF7
Nombre del requisito	Actualizar Palabras
Tipo	Requisito
Fuente del requisito	

Introducción: El sistema le permitirá al usuario agregar las palabras que se encuentra en el servicio web que han introducido los demás usuarios para actualizar la lista de palabras y agregarla en la base de datos local.

Entrada:

Ninguna.

Salida:

Se visualizará en la lista las nuevas palabras agregadas por el servicio web.

11.3.2.8 - Requisito funcional 8

Número del requisito	RF8
Nombre del requisito	Actualizar Frases
Tipo	Requisito
Fuente del requisito	

Introducción: El sistema le permitirá al usuario agregar las palabras que se encuentra en el servicio web que han introducido los demás usuarios para actualizar la lista de palabras y agregarla en la base de datos local.

Entrada:

Ninguna.

Salida:

Se visualizará en la lista las nuevas palabras agregadas por el servicio web.

11.3.2.9 - Requisito funcional 9

Número del requisito	RF9
Nombre del requisito	Mostrar información del lenguaje Mayangna
Tipo	Requisito
Fuente del requisito	

Introducción: El sistema contará con un menú que le permitirá al usuario visualizar la información del lenguaje Mayangna (cultura, reseña histórica y lenguaje.

Entrada:

Ninguna.

Salida:

Se visualizará la información de la cultura lenguaje y reseña histórica de la lengua Mayangna.

11.3.2.10 - Requisito funcional 10

Número del requisito	RF10
Nombre del requisito	Visualizar Diccionario.
Tipo	Requisito
Fuente del requisito	

Introducción: El sistema mostrará una lista de palabras en español con su respectiva traducción en mayangna, en la cual el usuario podrá visualizar.

Entrada:

Ninguna.

Salida:

Se visualizará una lista de palabras guardadas en la base datos.

11.3.2.11 - Requisito funcional 11

Número del requisito	RF11
Nombre del requisito	Visualizar Frases populares
Tipo	Requisito
Fuente del requisito	

Introducción: El sistema mostrará una lista de frases en español con su respectiva traducción en mayangna, en la cual el usuario podrá visualizar.

Entrada:

Ninguna.

Salida:

Se visualizará una lista de palabras guardadas en la base datos.

11.3.3- Requisitos de Rendimiento

- Se necesita una fuente de energía para que el dispositivo pueda funcionar durante muchas horas.
- Min SD Version API 21: Android 5.0 (Lollipop).
- Max Sdk Version API 26: Android 8.0 (Oreo).
- Accesso a Internet.

12. Diagramas de secuencias

12.1. Diagrama de secuencia 1: Traducir de Mayangna a Español.

La ilustración muestra la interacción de la aplicación con el servidor de base de datos. Al iniciar la aplicación el usuario observara una interfaz de bienvenida e instrucciones que quiera realizar (Escuchar palabra en español o traducir palabras de mayangna a español). Para realizar la traducción de Mayangna a Español la aplicación pedirá al usuario ingresar en la caja de texto la palabra mayangna que desea traducir al español, una vez insertado los datos el sistema ejecuta una tarea asíncrona para validar la palabra ingresada. Luego este con la información obtenida por la tarea asíncrona realiza una consulta a la base de datos para ver si existe esta palabra. Dependiendo de la información que el sistema gestor de base de datos retorne a la aplicación, enseguida la aplicación enviara a la tarea asíncrona un valor booleano (True, False). Con esta información obtenida la aplicación le dará al usuario la palabra traducida en Español.

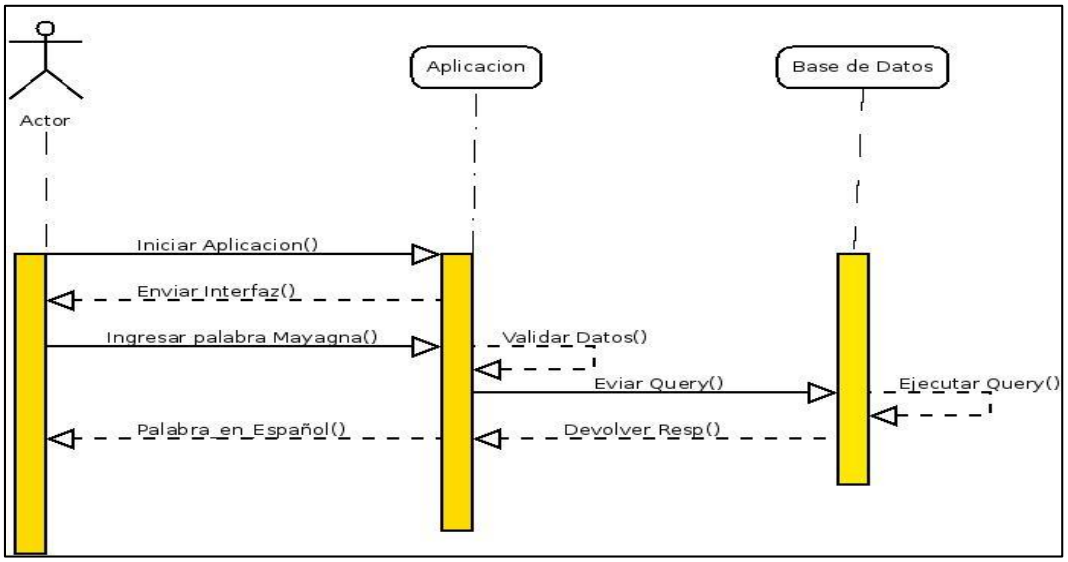
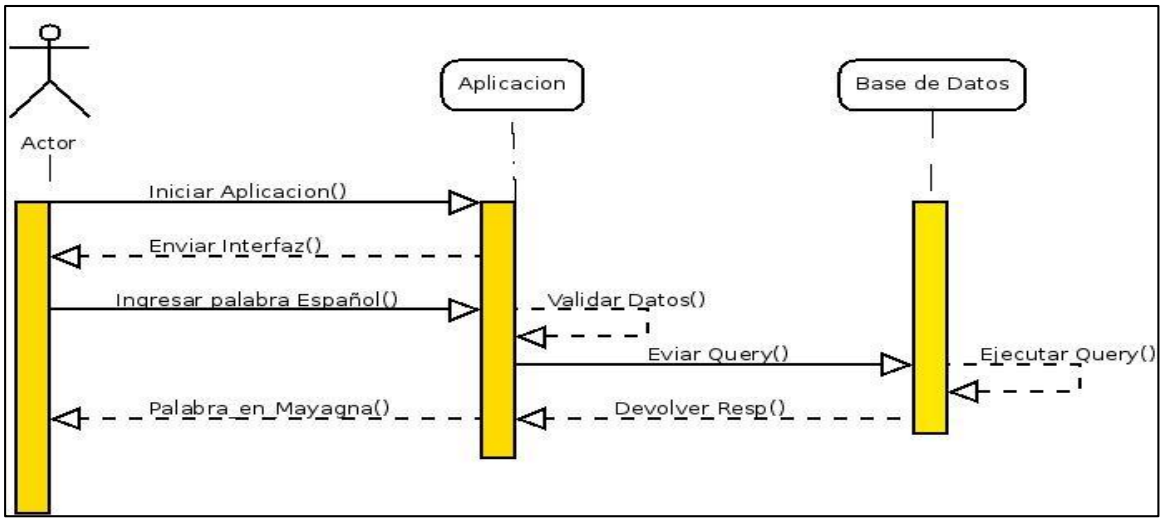


Ilustración 1 Diagrama de secuencia Traducir de Mayangna a Español.

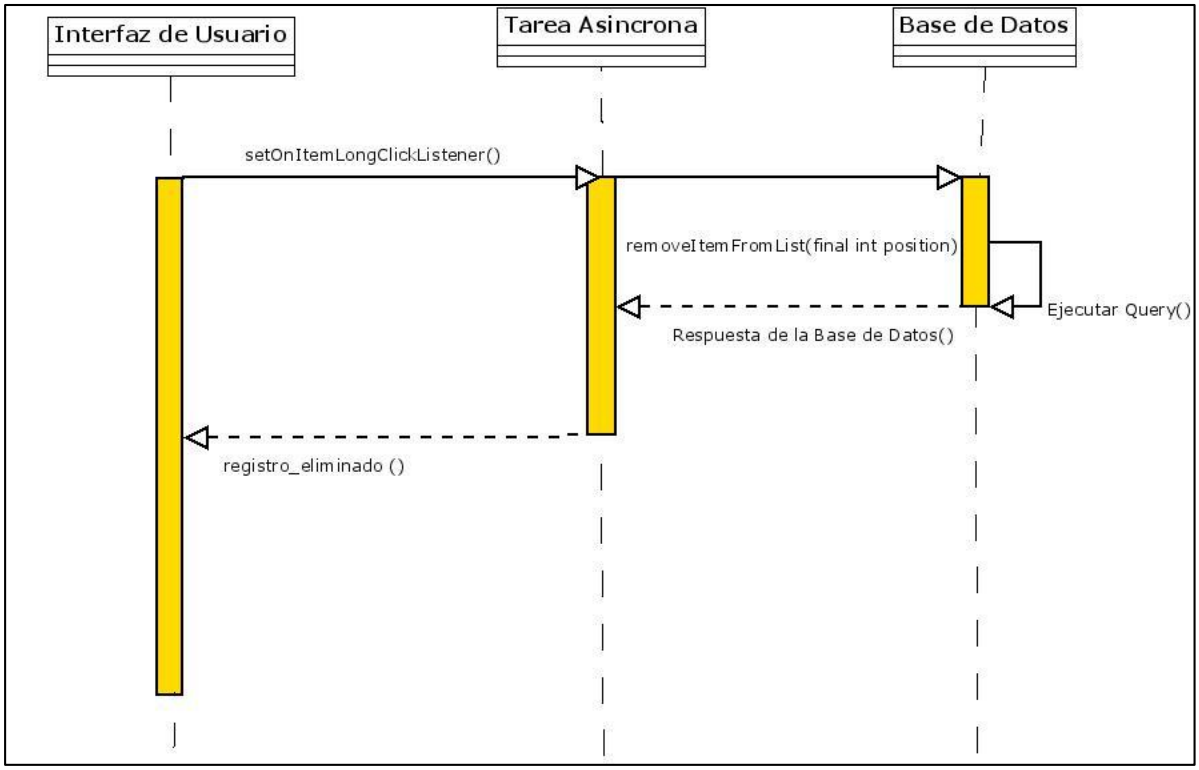
12.2. Diagrama de secuencia 2: Traducir de Español a Mayangna.

La ilustración muestra la interacción de la aplicación con el servidor de base de datos. Para realizar la traducción de Mayangna a Español la aplicación pedirá al usuario ingresar en la caja de texto la palabra en Español que desea traducir al Mayangna, una vez insertado los datos el sistema ejecuta una tarea asíncrona para validar la palabra ingresada. Luego este con la información obtenida por la tarea asíncrona realiza una consulta a la base de datos para ver si existe esta palabra. Dependiendo de la información que el sistema gestor de base de datos retorne a la aplicación, enseguida la aplicación enviara a la tarea asíncrona un valor booleano (True, False). Con esta información obtenida la aplicación le dará al usuario la palabra traducida en Mayangna.



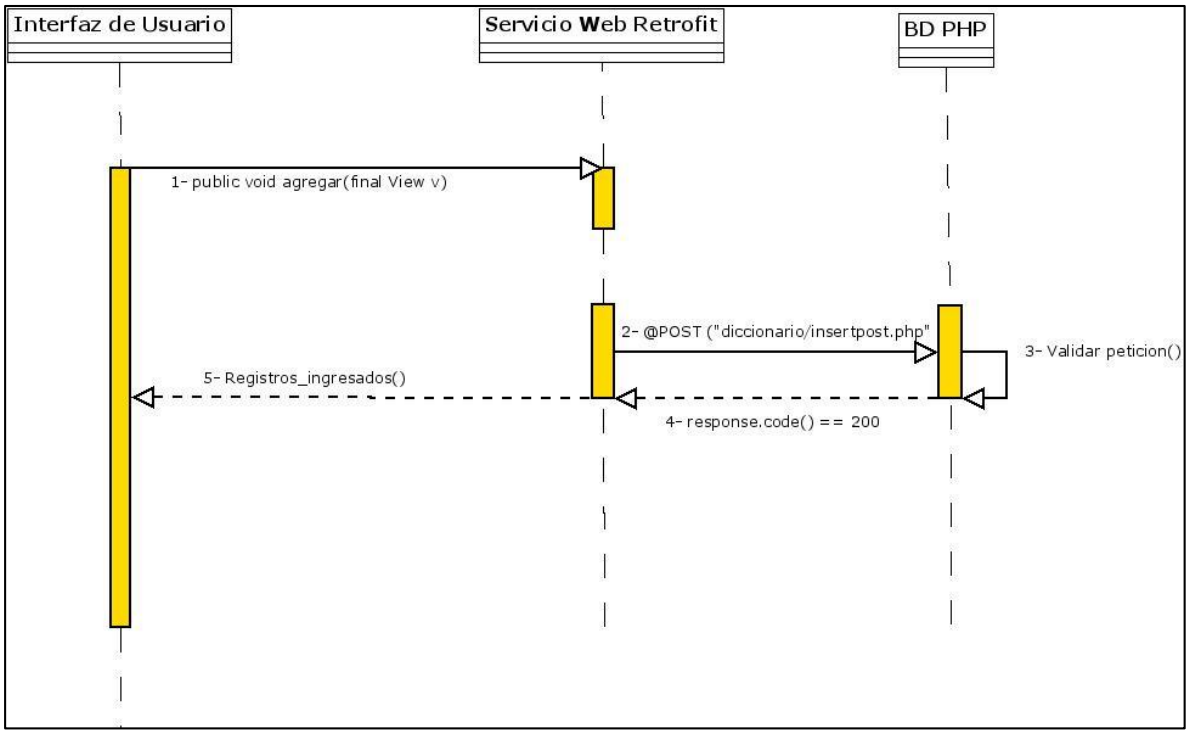
12.3. Diagrama de secuencia 3: Eliminar palabras.

La ilustración muestra la interacción de la aplicación con el usuario y el servidor de base de datos. Para realizar la eliminación de un registro de la base de datos, el usuariohabrá de activar el evento setOnitemLongClickListener de la aplicación, una vez ejecutado este evento el sistema ejecutara la tarea asíncrona para realizar la eliminación del registro seleccionado, entonces la aplicación móvil mostrara al usuario si desea confirmar la eliminación del registro de ser positivo la tarea asíncrona realizara una consulta a la base de datos para confirmar si existe este registro la aplicación enviara a la tarea asíncrona un valor booleano (True , False).Con esta información obtenida la aplicación le dará al usuario un mensaje que el registroha sido eliminado.

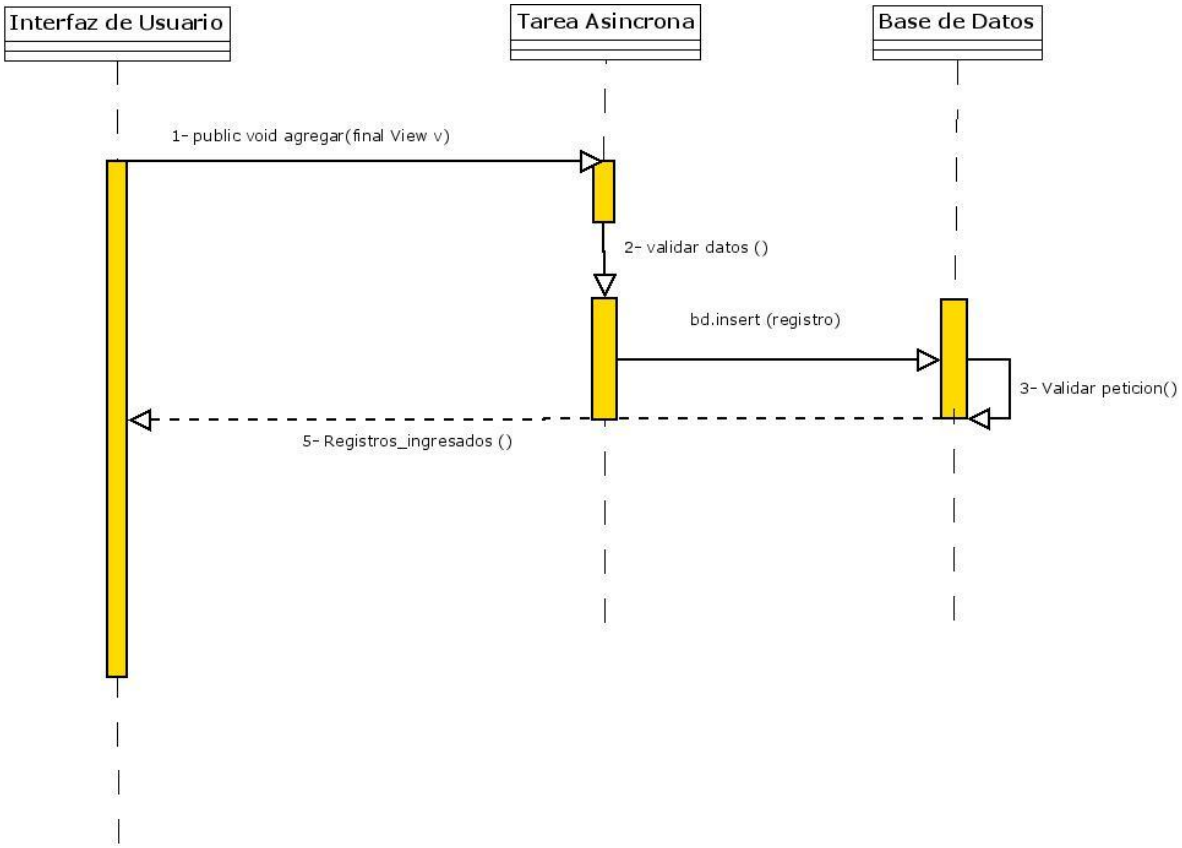


12.4. Diagrama de secuencia 4: Guardar Palabras (conexión a internet).

La ilustración muestra la interacción de la aplicación con el servicio Web; Para guardar las palabras nuevas que el usuario agregara para retroalimentarla base de datos en el servicio web que estarán disponibles para los demás usuarios al actualizar la lista de palabras. El usuario digitara la palabra en español y en mayangna, una vez insertado los datos el sistema ejecuta la tarea asíncrona (public void agregar (final View V)) que será enviada al servicio web retrofit; Luego este realizara la petición @POST para pasarle la URL del archivo PHP que realizara la inserción de las nuevas palabras en la base de datos que se encuentra en el servidor PHP esta validara la información recibida y retornara la respuesta asíncrona response.code()==200 al servicio web retrofit, con la información obtenida el servicio web enviara un mensaje a la aplicación que los datos han sido agregado satisfactoriamente.



12.4. Diagrama de secuencia 4: Guardar Palabras (sin conexión a internet).



14- Conclusiones.

Una vez analizada la información, diseñada y codificada la aplicación en el presente trabajo se ha llegado a las siguientes conclusiones:

- El diseño de la aplicación permite llevar a cabo de forma sencilla las consultas de traducción realizadas por los usuarios.
- La recopilación de información, para generar la base de datos con las frases y palabras se ha realizado exitosamente con el apoyo de personas nativas de la zona que domina el lenguaje Mayangna de la variación lingüística Panamahka.
- Android Studio nos ha concedido las herramientas necesarias para crear un diseño atractivo fácil de proyectar, así como la compilación rápida, para realizar la ejecución de la app en tiempo real gracias a su potente emulador, para hacer las respectivas pruebas de la app, gracias a sus beneficios le permite a la app integrase con las tecnologías usadas actualmente.
- Gracias a la integración de los servicios Web y Retrofit en nuestra app se pudo lograr el objetivo de permitirle a los usuarios de la app compartir con nuevas palabras que estos agreguen a la base de datos para seguir rescatando el lenguaje Mayangna en el país.

13. Recomendaciones.

Debido a que en la lengua Mayangna no existe una aplicación móvil que la publicitara como un ente educativo para aprender, rescatar y conocer un nuevo lenguaje, propio de los pueblos indígenas de Nicaragua, es recomendable que se realice una buena promoción para darla a conocer en las diferentes plataformas digitales e integrar nuevos idiomas como inglés, francés italiano etc; para ser traducido en el idioma mayangna.

Otro aspecto importante va dirigido a sus futuros desarrolladores en el caso de ampliar la aplicación móvil continuar con los estándares de diseño, ser ejecutable en IOS, implementar actualizaciones frecuentes de la base de datos de todos los usuarios, interactuar con los usuarios para capturar datos con la app y enviar mensajes push para cuidar a nuestro usuario e integrar las redes sociales para recomendar y difundir la nueva app para así alcanzar a más personas.

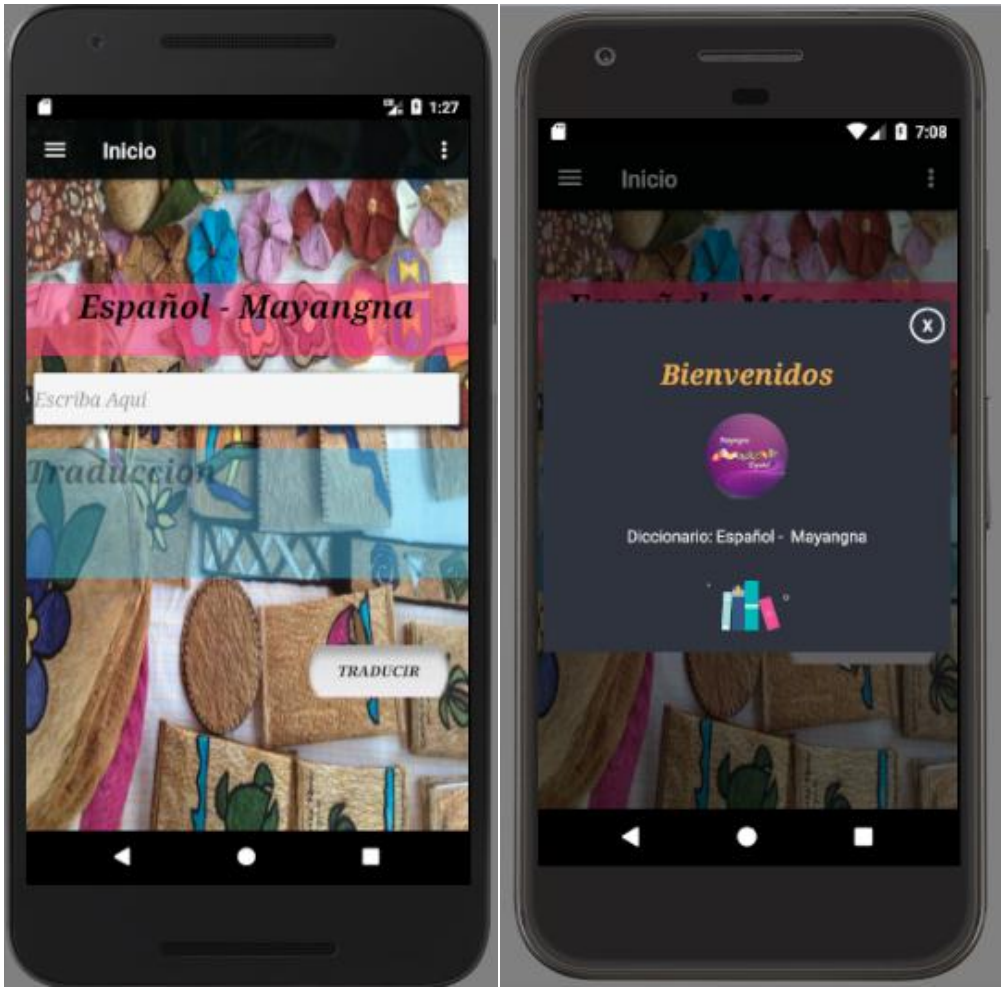
Otras de las recomendaciones para mejorar esta aplicación seria integrar clips de audios de las palabras y frases Mayangnas para que los turistas y la misma población nicaragüense conozca su pronunciación.

15. ANEXOS (Interfaces de la aplicación).

15.1 Actividad inicial bienvenida a la aplicación.

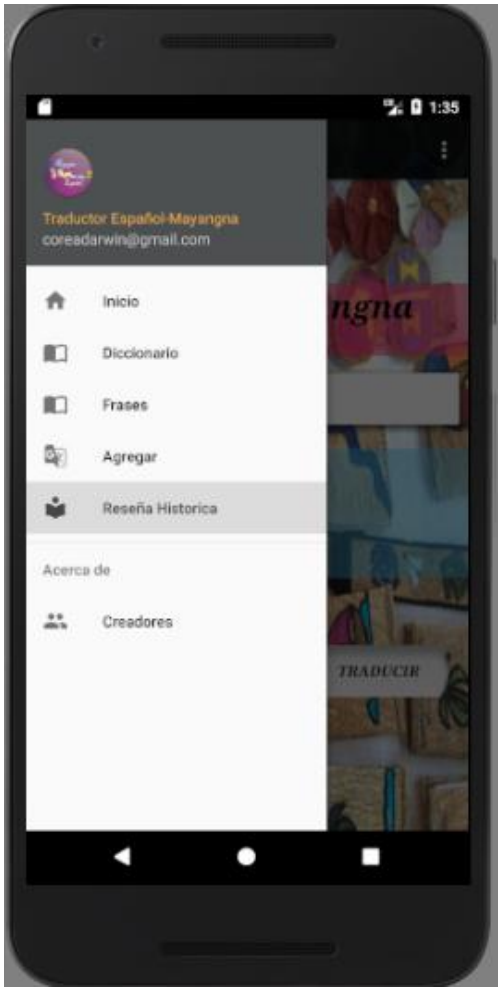
15.1 Actividad principal de la aplicación.

La siguiente actividad muestra el edit text principal para digitar las traducciones de las palabras y el botón TRADUCCIR para realizar esta petición.



15.3Menú.

La siguiente actividad muestra el menú principal,mostrando las diferentes funciones de la aplicación móvil como se puede observar en la siguiente imagen.

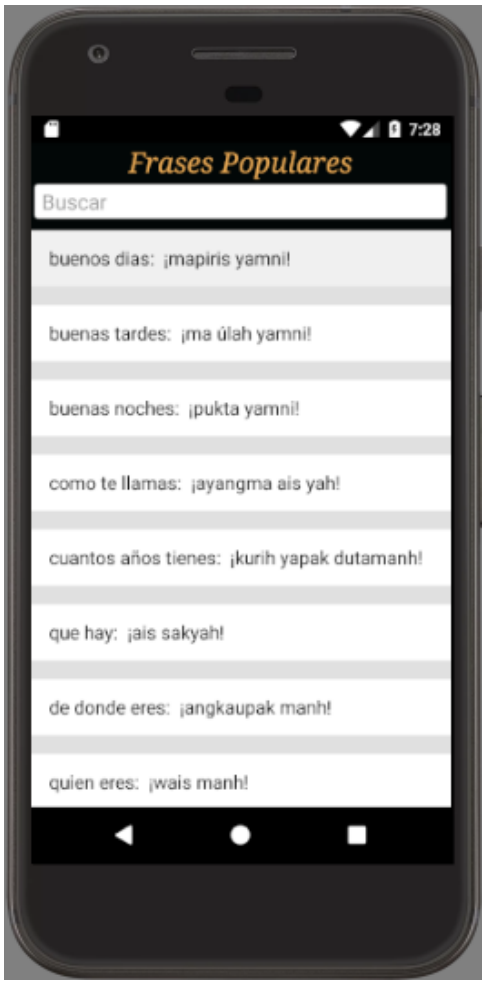


15.4 Diccionario.

La siguiente actividad muestra una de las funciones del menú principal (Diccionario) mostrando la lista de palabras guardadas en la base datos ordenada alfabéticamente, contiene una edit text para realizar búsqueda rápida en la lista una vez el usuario entre en esta actividad se actualizara la lista del diccionario automáticamente obteniendo las nuevas palabras que los demás usuarios han compartido y guardado en el servidor web.(El usuario tendrá que estar conectado a internet para obtener los nuevos datos.)

15.5 Frases.

La siguiente actividad muestra una de las funciones del menú principal (Frases) mostrando la lista de frases populares guardadas en la base datos ordenada alfabéticamente, contiene una edit text para realizar búsqueda rápida en la lista una vez el usuario entre en esta actividad se actualizara la lista de frases automáticamente obteniendo las nuevas palabras que los demás usuarios han compartido y guardado en el servidor web.(El usuario tendrá que estar conectado a internet para obtener los nuevos datos.)



15.6 Menú Agregar.

La siguiente actividad muestra el menú para agregar nuevos registros a la aplicación móvil para confirmar si el usuario quiere agregar palabras o frases.

15.7 Agregar palabras.

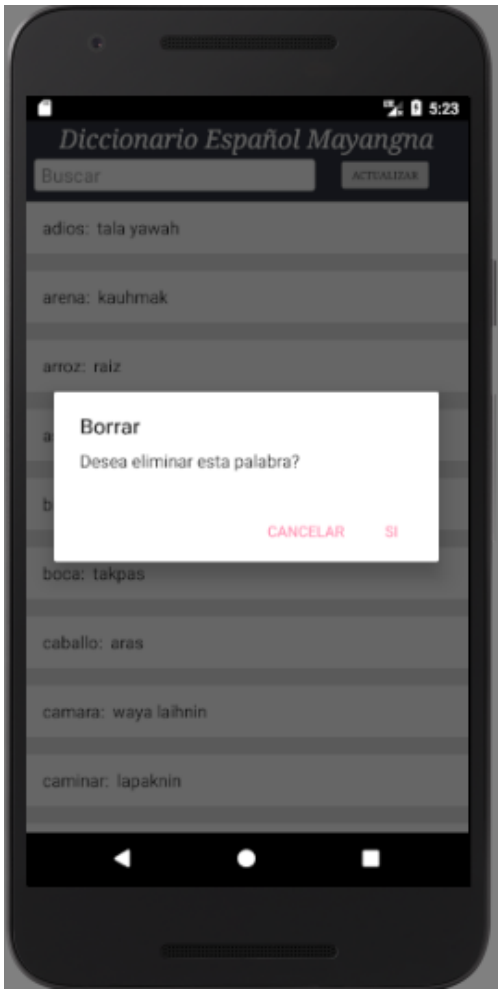
La siguiente actividad muestra dos edit text donde se digitarán las nuevas palabras y un botón para realizar la inserción esta acción hará una verificación si el usuario no está conectado a internet la tarea asíncrona mandara los nuevos datos a insertase en la base de datos local y si está conectado a internet la tarea asíncrona las guardará en el servidor web para que estén disponible para los demás usuarios.

15.8 Agregar Frases.

La siguiente actividad muestra dos edit text donde se digitarán las nuevas frases y un botón para realizar la inserción esta acción hará una verificación si el usuario no está conectado a internet la tarea asíncrona mandara los nuevos datos a insertase en la base de datos local y si está conectado a internet la tarea asíncrona las guardará en el servidor web para que estén disponible para los demás usuarios.

15.8 Eliminar Palabra o frase.

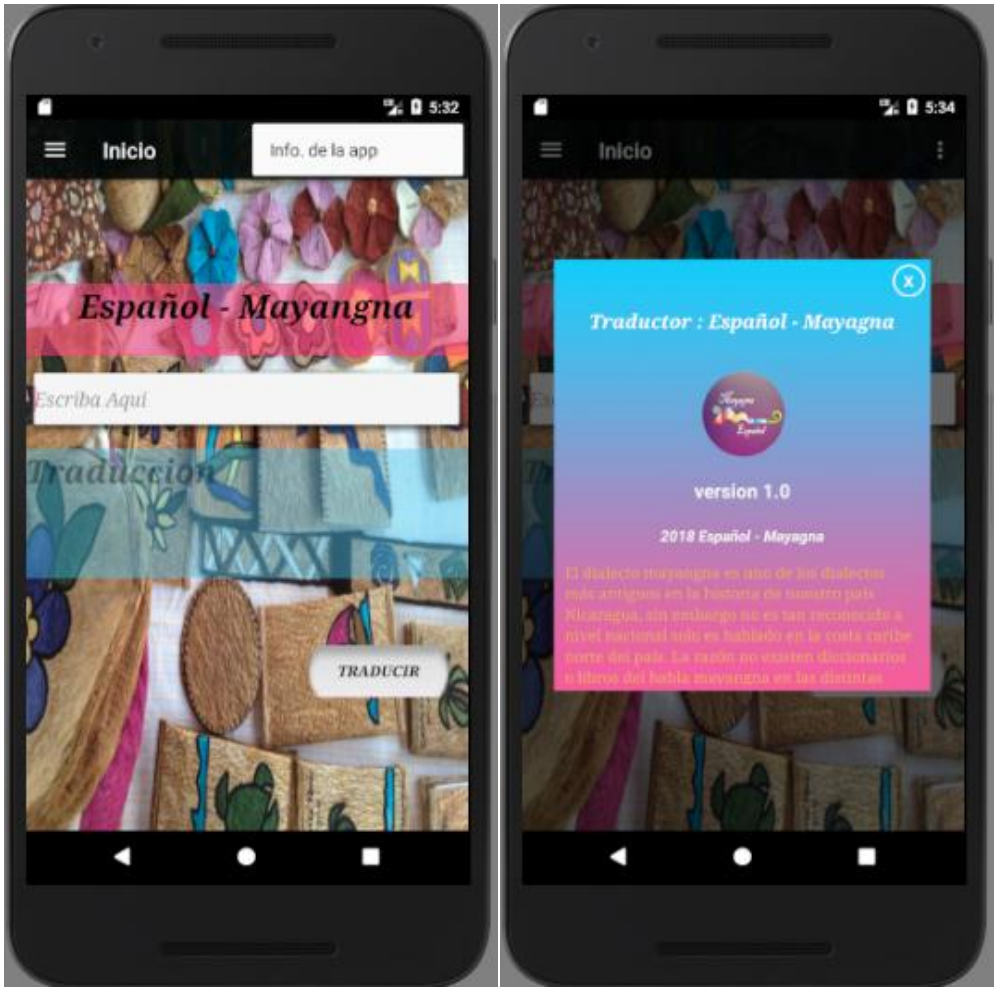
La siguiente actividad muestra la eliminación de registros de la base de dato, la que se realizara activando el evento setOnitemLongClickListener o pulsación larga mostrando una caga de texto con dos botones para confirmar la eliminación del registro.



15.8 Reseña Histórica.

La siguiente actividad muestra la información del dialecto mayangna dividiéndolo en tres tab de categoría (historia cultura y lenguaje) para que el usuario conozca más sobre este dialecto.

15.8 Información de la aplicación móvil.



15.9 Encuesta realizadas a turistas locales y extranjeros.

Encuesta a realizar por alumnos egresados de la carrera de ingeniería en sistemas para optar a su título universitario

Fecha_____

Seleccione al rango de edad al que pertenece

- 15 a18
- 18 a 20
- 20 a mas

¿Sabe quiénes son los Mayangnas?

- SI
- NO

¿Sabe que idioma hablan?

- SI
- NO

¿Cuenta con un teléfono celular inteligente o “Smartphone”?

- SI
- NO

¿Conoce de alguna aplicación que sirva para conocer la cultura e idioma de los Mayangnas?

- SI
- NO

¿Le gustaría que hubiera una?

SI

NO

En caso que su respuesta sea afirmativa en que plataforma para dispositivos móviles le gustaría que fuera

- ANDROID
- IOS
- WINDOW PHONE

TCL

```
#!/usr/bin/tclsh
if {$argc!=2} {
   puts stderr "Usage: %s DATABASE SQL-STATEMENT"
   exit 1
}
load /usr/lib/tclsqlite3.so Sqlite3
sqlite3 db [lindex $argv 0]
db eval [lindex $argv 1] x {
   foreach v $x(*) {
      puts "$v = $x($v)"
   }
   puts ""
}
db close
```

C

```
#include <stdio.h>
#include <sqlite3.h>

static int callback(void *NotUsed, int argc, char **argv, char **azColName){
        int i;
        for(i=0; i<argc; i++){
                printf("%s = %s\n", azColName[i], argv[i] ? argv[i] : "NULL");
        }
        printf("\n");
        return 0;
}
```

```
int main(int argc, char **argv){
        sqlite3 *db;
        char *zErrMsg = 0;
        int rc;

        if( argc!=3 ){
                fprintf(stderr, "Usage: %s DATABASE SQL-STATEMENT\n",
argv[0]);
                exit(1);
        }
        rc = sqlite3_open(argv[1], &db);
        if( rc ){
                fprintf(stderr, "Can't open database: %s\n", sqlite3_errmsg(db));
                sqlite3_close(db);
                exit(1);
        }
        rc = sqlite3_exec(db, argv[2], callback, 0, &zErrMsg);
        if( rc!=SQLITE_OK ){
                fprintf(stderr, "SQL error: %s\n", zErrMsg);
                sqlite3_free(zErrMsg);
        }
        sqlite3_close(db);
        return 0;
}
```

16-Bibliografía

21, T. (2015). Introducción a Retrofit. *WORDLINE TIEMPOS 21*. Obtenido de http://www.tempos21.com/web/blog/introduccion-a-retrofit/

Android, A. (s.f.). MultiAutoCompleteTextView With Example in Android Studio. *Abhi Android*.

BBVA. (2016). Retrofit una Librería para desarrollo de Android y Java. *BBVA*. Obtenido de https://www.beeva.com/beeva-view/desarrollo/retrofit-una-libreria-para-desarrollo-android-y-java/

BBVAopen4u. (2016). API REST: que es y cuáles son sus ventajas en el desarrollo de proyectos. *BBVA API_MARKET*. Obtenido de https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos

DEVELOPER, I. (2015). Servicios Web RESTful: Los Aspectos Básico. *IBM DEVELOPER*. Obtenido de https://www.ibm.com/developerworks/ssa/library/ws-restful/index.html

FIXTERGEEK. (s.f.). *Retrofit el mejor cliente REST para Android.* FIXTERGEEK.

INTECO. (2009). *INGENIERIA DE SOFTWARE: METODOLOGÍAS Y CICLOS DE VIDA.* Madrid, España: Laboratorio Nacional de Calidad de Software.

JAVA. (s.f.). Obtenido de https://www.java.com/es/download/faq/whatis_java.xml

Lenguaje de Programación JAVA. (s.f.). Obtenido de https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)

NICARAGUA, L. P. (23 de Noviembre de 2012). Importancia de la tecnologia en la Educación. *La Prensa Nicaragua, Seccion Tecnología*, pág. 94. Obtenido de http://www.laprensa.com.ni/2012/11/23/opinion/124869-importancia-de-la-tecnologia-en-la-educacion

Norwood, S. (Febrero, 1998). *Gramática de la Lengua Sumu - Mayangna.* Managua, Nicaragua: CIDCA - UCA.

Robins, T. (2012). *Cuaderno Culltural Sumu Mayangna 6.* Managua : CRAAN.

society, C. (1993). *Especificación de Requisitos según el Estándar de IEEE830.* C. Society.

Software, D. i. (2008). Introducción a los servicios web RESTful. *DOS IDEAS*. Obtenido de https://dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful

SQLite. (s.f.). Obtenido de https://www.ecured.cu/SQLite

SQLite. (s.f.). Obtenido de https://es.wikipedia.org/wiki/SQLite

Suazo, R. D. (2014). *Cuaderno Cuatrilingue Tuahka-Español-Miskiti-Ingles.* Managua : Bolonia Printing.