Universidad de las Ciencias Informáticas

Facultad 3

Título: Sistema de encuestas para Android

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

Autores: Eiléen Caridad Leyva Moré

Jorge David Torres Valiente

Tutores: MSc. Julio C. Díaz Vera

Ing. Guillermo Manuel Negrín Ortiz

Declaración de Autoría

Declaramos ser autores del presente trabajo de diploma y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Autorizamos a dicho centro para que haga el uso que estime pertinente con este trabajo.

Para que así conste firmamos el presente a los ____ días del mes de _____ del año_____.

Eiléen Caridad Leyva Moré	Jorge David Torres Valiente
_____	_____
Firma del autor	Firma del autor
MSc. Julio Cesar Díaz Vera	Ing. Guillermo Manuel Negrín Ortiz
_____	_____
Firma del Tutor	Firma del Tutor

Datos de Contacto

Autores:

Eiléen Caridad Leyva Moré

Jorge David Torres Valiente

Tutores:

MSc. Julio C. Díaz Vera

Correo electrónico: jcdiaz@uci.cu

Ing. Guillermo Manuel Negrín Ortiz

Correo electrónico: gmnegrin@uci.cu

Dedicatoria

Cuando era niña mi abuela me contó que había amores que a pesar de la distancia durarían para siempre y en mi inocencia de niña no lo entendía. Y es ahora que puedo comprenderlo, es por esto que quiero dedicarle mi tesis a esas dos estrellitas que iluminan mi camino día a día, mis abuelos, a los cuales perdí hace muy poco tiempo y los extraño muchísimo.

Y a mi mamita, mi inspiración.

Eiléen

A mi mamá, a mi hermana, y a Lisbet por su apoyo incondicional.

Jorge David

Agradecimientos

A toda mi familia, en especial a mis tíos Nelson, Yoyo y Marlen, por siempre estar al pendiente de todo. Y a mis primos, Pabli, Nelsy, Rodo, Erne. Los quiero mucho.

A mi hermano Jorgito, por quererme tanto y por confiar y contar conmigo para todo, gracias por demostrarme que siempre te tendré cuando te necesite.

A mi hermano Nelsito, por demostrarme que debajo de esa armadura de acero hay un gran hombre, un osito de peluche y el mejor de los hermanos. Gracias por existir. Te amo

A Jose, por ser una de las personas más especiales en mi vida, por demostrarme tu cariño y apoyo incondicional.

A Yoel, por apoyarme en todas mis locuras, por quererme como si fuera tu hija, por cuidarme y ser el padre que no tuve, y en especial por hacer feliz a mi mami.

A mis tutores y en especial a Guille, por guiarnos durante todo este tiempo.

A mi oponente, por apoyarnos en la tesis y por ser un ejemplo para nosotros como profe y como hombre.

A todos los profes que contribuyeron a que hoy llegara hasta aquí.

A Alex, por soportarme durante los últimos 8 años de tu vida, por quererme y ayudarme en todo. Te quiero.

A mis Super Friends (Darlyn, Dayana, Ayala, Javi, Lis, Roniel, mi gordo) por compartir conmigo desde primer año y por demostrarme que soy la persona más paciente del mundo. Espero que, aunque estemos lejos sigamos siendo una familia.

A mis hermanitos postizos, Leanne y Dannis, por ser mis mejores amigos desde la niñez y por estar siempre cuando los necesito. Los quiero.

A todos mis amigos de acá de la universidad, en especial a mi puty, Frank, Ojeita, Carlos, Jorge Luis, Arian, Alfredo, Suny, Yendrie, Raykof, Alexis. A mis amigas Dany y Bia.

A Day, por ser tan especial conmigo, y por demostrarme que tengo una buena amiga.

A mi compañero de tesis, por siempre estar incondicionalmente para mí, por compartir conmigo esta difícil tarea de sacrificio y esfuerzo, por soportarme en los momentos de estrés y por ser mi amigo desde primer año.
Y por último y más importante, a mi mami, por ser mi hermana, mi madre, mi mejor amiga y mi padre. Por ser siempre incondicional, me ha hecho reír, me ha secado las lágrimas, me ha visto

triunfar, me ha regañado, pero siempre a mi lado. Cuando las demás puertas cierran, las de ellas siempre están abiertas, siempre ahí para mí. Es una bendición que me ha dado Dios, una verdadera amiga, ojalá y estuvieras a mi lado para siempre.

Eiléen

A mi mamá, por ser mi guía. Gracias por educarme, cuidarme y protegerme en cada momento y sobre todo por ser mi gran ejemplo.

A mi hermana, gracias por apoyarme.

A Lisbet, gracias por acompañarme durante todos estos años, por ser mi novia, mi amiga, mi hermana; por aguantar mi mal humor, gracias por todo. Te amo.

A mi compañera de tesis, por estar siempre al pendiente y compartir juntos cada momento, para poder cumplir con nuestro objetivo, por ser mi amiga desde primer año, y también por aguantarme por tantos años.

A mis tutores por la confianza, siendo guía y fuerza durante todo el proceso.

A mi oponente, gracias por estar al pendiente del proceso.

A todos muchas gracias.

Jorge David

Resumen

En la Universidad de la Ciencias Informáticas (UCI), las encuestas son realizadas con relativa frecuencia, no solo para conocer detalles del proceso docente-educativo y para determinar características del personal sino también para la evaluación de la calidad del proceso de desarrollo de software. Estas encuestas son aplicadas de forma manual, lo que trae consigo pérdida de tiempo y gasto de recursos humanos y materiales; o de forma digital, lo que requiere conexión para responderlas.

Android es uno de los sistemas operativos (SO) de mayor uso en el mundo, debido a que es un sistema orientado a dispositivos móviles, además de ser un sistema abierto, multitarea, que permite a los desarrolladores acceder a las funcionalidades principales del dispositivo mediante aplicaciones.

La presente investigación tiene como objetivo el desarrollo de una aplicación para Android que permita responder encuestas de forma digital en la UCI con o sin conexión. Para guiar el desarrollo de la aplicación se utilizó como metodología Programación Extrema (XP), Java como lenguajes de programación y Android Studio en su versión 2.3 como Entorno de Desarrollo Integrado (IDE).

Palabras clave: Android, encuestas, aplicación, conexión.

Índice general

Índice de figuras

Índice de tablas

Introducción

En esta era denominada “del conocimiento o de la información*”,* la recopilación de datos constituye el instrumento fundamental para la determinación de tendencias y la evaluación de la calidad de productos y servicios. Este proceso se auxilia de diversas técnicas, herramientas y métodos, entre las que se encuentran: la entrevista, la observación, el cuestionario y la encuesta. De los anteriormente mencionados la encuesta es la que goza de mayor prestigio por su dinamismo y utilidad para la toma de decisiones (Casorla, 2012). Las encuestas constituyen uno de los mecanismos más utilizados en la obtención de información de interés sobre una situación concreta de una población relativamente grande (Universidad de Champagnat , 2013).

Las instituciones académicas cada vez más se preocupan por el nivel de satisfacción de alumnos y profesores, estas han detectado que una de las características fundamentales para el éxito o fracaso de una persona, es su motivación. Para que un alumno o trabajador esté satisfecho y motivado, tienen que aparecer ciertos factores tales como una planificación docente adecuada, disponibilidad de recursos materiales y pedagógicos que ayuden a desarrollar y ampliar sus estudios. Para poder conocer su nivel de satisfacción, que condicionará en gran parte su éxito o fracaso en los estudios, es necesario realizar periódicamente una encuesta de satisfacción. También, para detectar posibles amenazas o fortalezas dentro del ámbito académico, es muy importante dejar a los alumnos evaluar al profesor y/o la asignatura impartida. Con una correcta encuesta sobre el profesor o sobre una asignatura en concreto, se pueden llegar a tomar decisiones que ayuden en la evolución académica de los alumnos. El que un profesor se encuentre a gusto con la asignatura impartida y con sus alumnos, se reflejará directamente en el éxito académico del centro (Zalazar, 2015).

El sector educacional cubano y en especial la enseñanza de pregrado y postgrado no escapan de este fenómeno, sino que para dar cumplimiento a su misión social se auxilia de este método para el control, seguimiento y evaluación del proceso docente-educativo en sus centros de enseñanza.

En la Universidad de las Ciencias Informáticas (UCI) las encuestas se utilizan para conocer detalles del proceso docente-educativo, para la evaluación de la calidad del proceso de desarrollo de software, así como para ayudar a determinar las características de los estudiantes o profesores. Este proceso al inicio de la UCI era realizado de forma manual, requiriendo la presencia de los encuestados en un local y un encuestador para la aplicación de la misma. El procesamiento de la información se realizaba utilizando hojas de cálculo de Microsoft Excel, trayendo consigo la pérdida de tiempo, el gasto de recursos humanos y materiales. Como solución a los problemas que traen

consigo responder encuestas de forma manual se decide utilizar sistemas informáticos para la gestión de encuestas de forma digital. *LimeSurvey*[1] es el sistema que actualmente se utiliza en la UCI para el control del proceso docente-educativo, este sistema a pesar de brindar una amplia gama de funcionalidades posee como limitante que requiere que los encuestados estén conectados a una red.

Para el contexto de la UCI, donde existe un número limitado de puntos de acceso wifi esto constituye una limitación a la hora de responder las encuestas. De igual modo se identifica el alto grado de penetración de los teléfonos inteligentes, principalmente los de sistema operativo (SO) Android, en la comunidad universitaria como una oportunidad para la realización de las encuestas, facilitando la tarea de coordinación de la aplicación de las mismas aumentando la accesibilidad.

Por tanto, surge la necesidad de dar solución a las situaciones antes expuestas y el **problema a resolver** consiste en ¿Cómo desarrollar una aplicación para el sistema operativo Android que posibilite responder encuestas desde un teléfono inteligente?

El **objeto de estudio** es el proceso de desarrollo de software.

De ello se deriva que el **campo de acción** que abarca este trabajo es el proceso de desarrollo de software para dispositivos móviles.

El **objetivo general** es desarrollar una aplicación para el sistema operativo Android que permita responder encuestas sin necesidad de estar conectado a una red informática.

De acuerdo con lo anterior los **objetivos específicos** son:

- Establecer los referentes teóricos que sustentan el objeto de estudio de la investigación.
- Realizar diseño e implementación de la aplicación para el sistema operativo Android.
- Establecer estrategia de pruebas para validar la aplicación Android.

Posible resultado:

Se espera obtener una aplicación para el sistema operativo Android que permita responder encuestas en la Universidad de las Ciencias Informáticas sin necesidad de estar conectado a una red, luego de haber sido descargadas de una aplicación Web.

Para el desarrollo de la investigación los **métodos teóricos** utilizados son:

- Análisis Histórico Lógico: Se utiliza para estudiar las formas de solución a problemas similares sobre la gestión de encuestas existentes en todo el mundo, esto permitirá constatar teóricamente cómo ha evolucionado este fenómeno.
- Analítico-Sintético: Se utilizó para el análisis de teorías y documentos existentes, se extrajeron los elementos más importantes de cada uno de los aspectos esenciales de las herramientas y la literatura seleccionada para el tema a estudiar.

[1] Aplicación de código abierto para la gestión de encuestas en línea.

Método empírico:

- Entrevista: Se utilizó en la investigación para precisar el problema a resolver, así como las necesidades existentes y los procesos que se llevan a cabo actualmente para la gestión de encuesta.

Estructura del documento:

El documento está estructurado en 3 capítulos, a continuación, se brinda una breve descripción de cada uno:

- Capítulo 1: Fundamentación teórica: En este capítulo se establecen todos los elementos teóricos de la investigación. Se realiza un estudio de los diferentes sistemas similares al que se desea implementar. Se define la metodología XP para el desarrollo del software.
- Capítulo 2: Análisis, diseño e implementación: A lo largo de este capítulo se presenta la solución propuesta con todos los aspectos definidos en la fundamentación teórica. Se generan los artefactos, asociados a la metodología XP, propuestos en cada una de sus fases, así como otros artefactos necesarios para un mayor entendimiento de la propuesta. Se describen los lenguajes y herramientas a utilizar en la elaboración de la solución.
- Capítulo 3: Validación de la solución: En este capítulo se realizan las pruebas necesarias para validar el producto obtenido. Se valida que el diseño realizado cumpla con la calidad requerida y que el sistema implementado satisface las necesidades de los clientes.

Capítulo 1: Fundamentación teórica

Introducción

En el presente capítulo de abordarán conceptos asociados al objeto de estudio, los cuales ayudan a un mejor entendimiento del mismo. Se describe la metodología que guiará el proceso de desarrollo del sistema informático que se propone como solución a la problemática existente.

1.1 Software

Según (Gomez, 2007) el software es un conjunto de programas, rutinas, datos o instrucciones que se encargan de ejecutar diferentes tareas en un dispositivo, ya sea éste una computadora, o un móvil. En español, software significa “parte blanda”, dado que es la parte intangible de cualquier dispositivo y es la que actúa como unión entre el usuario y el conocido como hardware.

En la clasificación de software destacan principalmente tres tipos, según (Gomez, 2007) ellos son:

- **Software de sistema:** Se encuentran dentro de esta clasificación no solo los sistemas operativos sino también se pueden encontrar los softwares que permiten la comunicación entre el hardware y el sistema operativo, los controladores de sistema para la ejecución de diversos sectores de la computadora e incluso programas que son capaces de administrar los recursos y de proporcionarle al usuario una interfaz para que pueda controlar la computadora de forma sencilla.
- **Software de Programación:** El software de programación es un conjunto de herramientas que permiten el desarrollo de aplicaciones de software, está dirigido a los programadores o desarrolladores de software, los cuales utilizan estas herramientas para crear, depurar y mantener sistemas.
- **Software de Aplicación:** Este es que se utiliza día a día, cada uno de los programas, aplicaciones o utilidades que se utilizan dentro de computadora o del dispositivo móvil, entran dentro de esta clasificación, es el resultado de la programación de software, enfocado hacia alguno de los sistemas operativos, es el software diseñado para el usuario final.

1.2 Aplicación móvil

Dentro del software de aplicación se encuentran las aplicaciones móviles, término proveniente del inglés App, que es una contracción de *application* con un tono informal, es un software que puede instalarse en un dispositivo móvil con la finalidad de extender sus funcionalidades, al igual que ocurre con la instalación de programas en computadoras de escritorio o portátiles (González, 2010)**.** Las diversas utilidades de las aplicaciones móviles, ya sea para comunicarse, jugar, comprar, leer, o tomar fotografías han pasado a formar parte de los hábitos de consumo de los usuarios de teléfonos inteligentes y tabletas.

De acuerdo con el tipo del contenido que se ofrece a los usuarios las apps se pueden agrupar en (González, 2010):

- **Aplicaciones de entretenimiento:** En esta categoría, se pueden encontrar las aplicaciones de juegos. Son aquellas que brindan diversión al usuario. Una de las características principales son sus gráficos, animaciones y efectos de sonido, lo cual permite a las personas tener mayor atención en el desarrollo del juego.
- **Aplicaciones sociales:** Las aplicaciones de este tipo surgieron para la interrelación de personas y construcción de redes, estas incrementan la comunicación de usuarios de diferente tipo, distancia y preferencias. Un caso de éxito en esta categoría es Facebook, con casi mil millones de usuarios se convirtió en una aplicación social que sirve de puente de sociabilización para muchas personas en el planeta.
- **Aplicaciones educativas e informativas:** La variedad del uso de las aplicaciones móviles también abarca la categoría de la educación e información. Estas aplicaciones son usadas como transmisoras de conocimientos y noticias, con lo cual se privilegia el acceso al contenido y estructura de la información presentada.
- **Aplicaciones utilitarias y de productividad:** Son basadas en la eficiencia ya que proporcionan herramientas para solucionar problemas o simplificar actividades, están centradas en la ejecución de tareas, cortas y rápidas.

Dentro de este último grupo se encuentran las aplicaciones utilizadas para la recopilación de datos, como es el caso de la aplicación que se propone como resultado de la investigación.

1.3 Dispositivo Móvil

Las aplicaciones móviles son instaladas en aparatos de pequeño tamaño, con algunas capacidades de procesamiento, con conexión permanente o intermitente a una red, con memoria limitada, que ha sido diseñado específicamente para una función, pero puede llevar a cabo otras funciones más generales, los cuales pueden definirse como dispositivos móviles. Una característica importante es el concepto de movilidad, los dispositivos móviles son pequeños para poder portarse y ser fácilmente empleados durante su transporte (González, 2010). Las ventajas de estos dispositivos son la comodidad del dispositivo en cuanto al tamaño, peso, dispersión, entre otros, las cuales les permiten a los usuarios realizar un uso del dispositivo sin mayor dificultad, el cual se convierte en una herramienta de trabajo y entretenimiento que provee características especiales a los usuarios. Les permite a los usuarios realizar tareas específicas con mayor comodidad, efectividad y eficiencia. Además, la energía que usan es almacenada en baterías, las cuales tienen un tiempo de uso sin estar conectado a la corriente (Wordpress, 2014).

Los dispositivos móviles más comunes son los reproductores de audio portátiles, los navegadores GPS (Sistema de Posicionamiento Global), los PDA (Asistente Personal Digital), los teléfonos móviles y las tabletas. Se prestará especial atención a los teléfonos inteligentes y las tabletas por ser los tipos de dispositivos más utilizados y conocidos en la actualidad, los que ofrecen mayor variedad de aplicaciones multimedia y los que más posibilidades de evolución presentan en este sentido (Valero, y otros, 2012).

Según (González, 2010) un ***smartphone* o teléfono inteligente** es un dispositivo electrónico que funciona como un teléfono móvil con características similares a las de un ordenador personal. Permite hacer llamadas y enviar mensajes de texto como un móvil convencional pero además incluye características cercanas a las de un ordenador personal. Una característica importante de los teléfonos inteligentes es que permiten la instalación de programas para incrementar el procesamiento de datos y la conectividad. Entre las características de los smartphones están las pantallas táctiles, la conectividad a internet y el acceso al correo electrónico. Otras aplicaciones que suelen estar presentes son las cámaras integradas, la administración de contactos, el software multimedia para reproducción de música y visualización de fotos y video-clips y algunos programas de navegación, así como, ocasionalmente, la habilidad de leer documentos de negocios en variedad de formatos como PDF y Microsoft Office.

Una **tableta** es una computadora (ordenador) portátil más grande que un *smartphone*. Se caracteriza por contar con pantalla táctil: esto quiere decir que para utilizar la tableta no se necesita *mouse* (ratón) ni teclado. Existen diversos formatos de tableta, las más populares miden entre 8 y 12 pulgadas y disponen de teclado virtual, aunque es posible añadirle un teclado físico a través de una conexión USB o Bluetooth. Al igual que los teléfonos inteligentes es posible instalarle otras aplicaciones (González, 2010).

1.3.1 Sistema Operativo para dispositivos móviles

Todos los dispositivos móviles deben contar con un sistema operativo que los provea de características similares a las de un computador. Un sistema operativo para dispositivos móviles es considerado el programa principal y este es capaz de administrar todos sus recursos para ser utilizados de manera eficiente (Shankland, 2008). Existen diferentes sistemas operativos para dispositivos móviles, entre los que se encuentran: Symbian, IOS, Blackberry OS, Windows Phone, Android, Firefox OS, Ubuntu Touch y Tizen (Shankland, 2008). El estudio se centrará en Android por las ventajas que posee y la gran cantidad de usuarios que lo utilizan.

Según el servicio de estadísticas (NetMarketShare, 2017) la cuota de mercado de sistemas operativos móviles a principio del año 2017 es el siguiente:

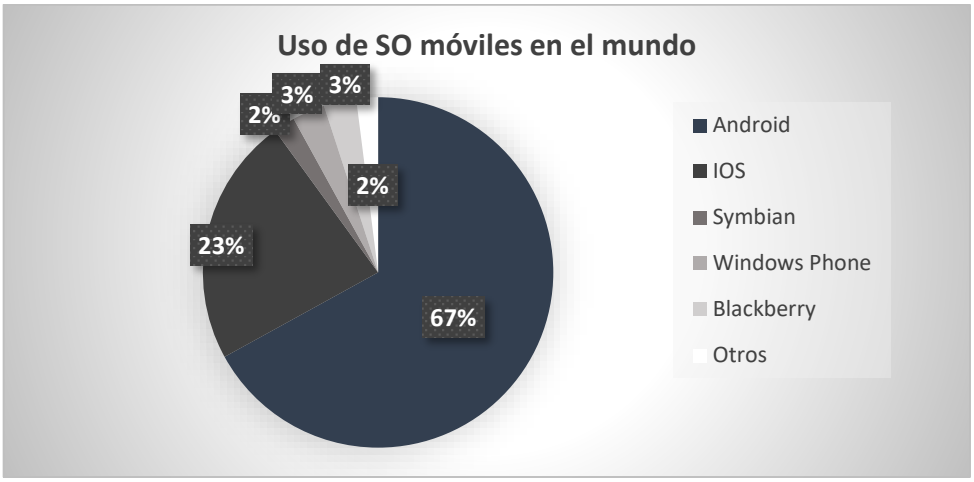

Ilustración 1 Uso de los SO móviles en el mundo a principio de 2017

Evidenciándose que Android es el sistema Android es el sistema operativo número uno en cuanto a popularidad, con una cuota de mercado cercana al 70%. El sistema operativo de Google se caracteriza por ser un sistema operativo basado en Linux y orientado a dispositivos móviles, como teléfonos inteligentes y tabletas. Android, a diferencia de otros sistemas operativos como IOS o Windows Phone, se desarrolla de forma abierta y se puede acceder al código fuente. Permite ser instalado en varios dispositivos de diferentes fabricantes. El mismo, es multitarea, por lo que permite ejecutar múltiples aplicaciones simultáneamente (NetMarketShare, 2017).

1.3.1.1 Arquitectura Android

Según (Vílchez, 2009) los principales componentes de la arquitectura de Android son:

Aplicaciones: Las aplicaciones base incluirán un cliente de email, programa de SMS, calendario, navegador y otros. Todas las aplicaciones escritas en Java.

Marco de trabajo: Los desarrolladores tienen acceso completo a las APIs[2] (Application Programming Interface) del marco de trabajo usado por las aplicaciones base. La arquitectura está diseñada para simplificar la reutilización de componentes.

Biblioteca: Android incluye un conjunto de bibliotecas desarrolladas en C y C++ algunas de las cuales no son desarrolladas por Google o por el proyecto Android, sino que son existentes en el mundo de *Open Source*.

Entorno de ejecución: Cada aplicación Android ejecuta su propio proceso, con su propia instancia de la máquina virtual Dalvik que ejecuta archivos en el formato Ejecutables Dalvik (Dalvik Executable .dex), optimizado para memoria mínima.

[2] Conjunto de funciones y procedimientos que cumplen una o muchas funciones con el fin de ser utilizadas por otro software.

Núcleo - Linux: Android depende en la última de sus versiones, del núcleo 3.0.31 de Linux para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red, y modelo de drivers. El núcleo también actúa como una capa de abstracción entre el hardware y el resto de la pila.

En la siguiente figura se muestra la arquitectura de Android descrita anteriormente:

Ilustración 2 Arquitectura de Android

1.3.1.2 Ventajas de Android

Este sistema operativo tiene numerosas ventajas entre las que se encuentran (Wordpress, 2012):

- Puede instalarse prácticamente en todo tipo de dispositivos, sean móviles, portátiles e incluso microondas.
- Puede adaptarse a la perfección a todo tipo de necesidades.
- Es libre con licencia Apache y código abierto para que un desarrollador no solo pueda modificar su código sino también mejorarlo.
- Garantiza que, en caso de haber un error, sea detectado y reparado con mayor presteza sin depender de nadie para pedir autorización a su cambio.
- El sistema Android es capaz de hacer funcionar a la vez varias aplicaciones y además se encarga de gestionarlas, dejarlas en modo suspensión si no se utilizan e incluso cerrarlas si llevan un periodo determinado de inactividad.

1.5 Requisito de alto nivel

Un requisito de alto nivel está asociado a un objetivo, meta o propósito que un sistema debe satisfacer. Los requisitos de alto nivel son un elemento fundamental para la confección de un proyecto, son este tipo de requisitos los que se utilizan para establecer el primer entendimiento entre la persona que desea el sistema y el equipo encargado de su desarrollo. Y es finalmente asociados

a estos requisitos que se evalúa si el sistema cumplimentó el objetivo para el que fue desarrollado (Rupp., 2015).

Por lo tanto, el requisito de alto nivel de la presente investigación es “Responder encuestas desde un dispositivo móvil con sistema operativo Android”. Por lo que se hace necesario tener claro el significado de encuesta, asumiendo a intereses de la investigación se toma como definición:

Encuesta: procedimiento utilizado para obtener información mediante preguntas dirigidas a una muestra de individuos representativa de la población o universo, de forma que de las conclusiones que se obtengan puedan generalizarse al conjunto de la población siguiendo los principios básicos de la inferencia estadística, ya que la encuesta se basa en el método inductivo, es decir, a partir de un número suficiente de datos se pueden obtener conclusiones a nivel general que en un futuro inmediato sirven para solucionar problemas existentes en el medio donde se realice la encuesta (Avilez, 2005).

1.6 Desarrollo de software

El desarrollo de Software es la realización sistemática de las actividades de planeación, diseño, codificación, pruebas, lanzamiento de productos de software nuevos cumpliendo con los requisitos especificados y con las normativas de seguridad de información (Guerrero, y otros, 2011).

El proceso de desarrollo de software no es único. No existe un proceso de software universal que sea efectivo para todos los contextos de proyectos de desarrollo. Debido a esta diversidad, es difícil automatizar todo un proceso de desarrollo de software. A pesar de la variedad de propuestas de proceso de software, existe un conjunto de actividades fundamentales que se encuentran presentes en todos ellos (Sommerville, 2002):

- **Especificación de software:** Se debe definir la funcionalidad y restricciones operacionales que debe cumplir el software.
- **Diseño e Implementación:** Se diseña y construye el software de acuerdo a la especificación.
- **Validación:** El software debe validarse, para asegurar que cumpla con lo que quiere el cliente.
- **Evolución:** El software debe evolucionar, para adaptarse a las necesidades del cliente.

Una perspectiva utilizada para determinar los elementos del proceso de desarrollo de software según (Letelier, 2003) es establecer las relaciones entre elementos que permitan responder “Quién” debe hacer “Qué”, “Cuándo” y “Cómo” debe hacerlo. En la figura que se muestra a continuación se muestran estos elementos y sus relaciones:

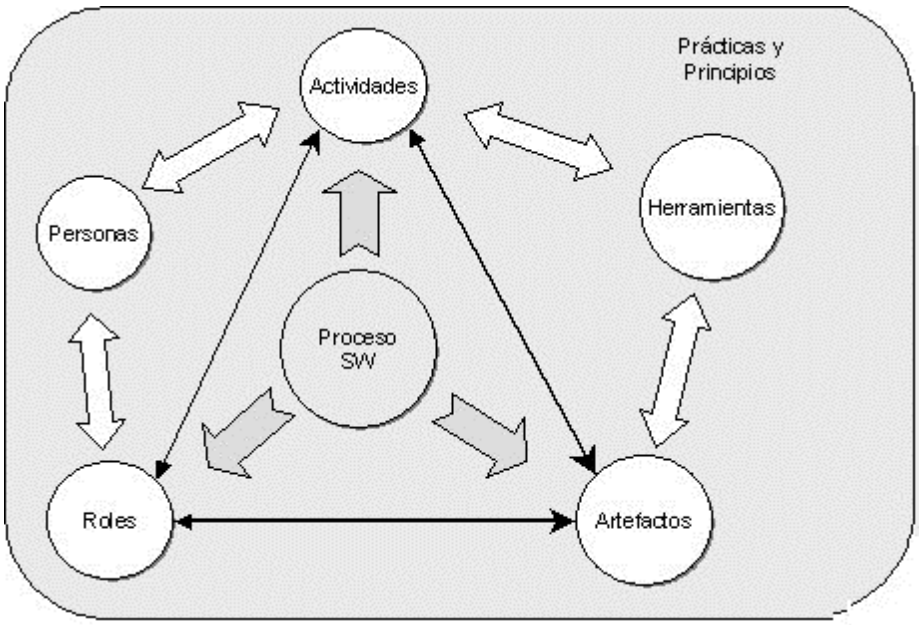

Ilustración 3 Relación entre elementos del proceso del software. Fuente: (Letelier, 2003)

En la figura se muestran los elementos de un proceso de desarrollo de software y sus relaciones. Así las interrogantes se responden de la siguiente forma (Letelier, 2003):

- **Quién:** Las Personas participantes en el proyecto de desarrollo desempeñando uno o más Roles específicos.
- **Qué:** Un Artefacto es producido por un Rol en una de sus Actividades. Las Herramientas apoyan la elaboración de Artefactos.
- **Cómo y Cuándo:** Las Actividades son una serie de pasos que lleva a cabo un Rol durante el proceso de desarrollo. El avance del proyecto está controlado mediante hitos que establecen un determinado estado de terminación de ciertos Artefactos.

La composición y sincronía de las actividades está basada en un conjunto de Principios y Prácticas. Las Prácticas y Principios enfatizan ciertas actividades y/o la forma como deben realizarse.

1.6.1 Desarrollo de software para dispositivos móviles

El desarrollo de aplicaciones móviles es el conjunto de procesos y procedimientos involucrados en la escritura de software para pequeños dispositivos inalámbricos de cómputo, como teléfonos inteligentes o tabletas (Rouse, 2005). Según la compañía de análisis de datos de aplicaciones móviles *App Annie*, el desarrollo de aplicaciones móviles seguirá en ascenso en todo el mundo.
Algunas tendencias relevantes sobre el desarrollo de aplicaciones móviles según *App Annie*:

- Las descargas de aplicaciones móviles en las tiendas iOS y Google Play llegaron a 26.000 millones en todo el mundo. Este número solo incluye las nuevas descargas y no las reinstaladas o las actualizaciones de las aplicaciones en las tiendas.
- Los juegos son las aplicaciones que generan más ingresos para los desarrolladores.

- El tiempo dedicado a las aplicaciones móviles también crece. En el caso de los teléfonos con el sistema operativo Android creció un 40%, acercándose a los 325.000 millones de horas en el tercer trimestre de 2017.
- El pago sin efectivo se está convirtiendo rápidamente en un nuevo estándar para muchos usuarios. En 2018 otra de las tendencias de los comercios será la aceptación de sistemas de pago por Apple Pay o Android Pay en cuanto a opciones predeterminadas y no sólo en negocios online.
- El crecimiento en términos de descargas y uso, demuestra que las aplicaciones se están convirtiendo cada vez más en el centro de la vida de las personas. Este valor se está traduciendo en un aumento de los ingresos para la industria.

El aumento en el uso de los dispositivos móviles por parte de los usuarios a nivel mundial se plantea como uno de los grandes motivos por los cuales el mercado de desarrollo de las aplicaciones móviles se ha vuelto más interesante.

1.7 Servicios Web

El *World Wide Web Consortium* (W3C) define un servicio web como un sistema de software designado para dar soporte a la interacción de máquina a máquina interoperativa a través de una red. Un servicio web realiza una tarea específica o un conjunto de tareas, y se describe mediante una descripción de servicio en una notación XML estándar llamada WSDL (*Web Services Description Language*). La descripción de servicio proporciona todos los detalles necesarios para interactuar con el servicio, incluidos los formatos de mensaje (que detallan las operaciones), los protocolos de transporte y la ubicación. Las aplicaciones basadas en servicios web son implementaciones en todas las tecnologías, con acoplamientos flexibles y orientados a componentes. Los servicios web se pueden utilizar individualmente o junto con otros servicios web, para llevar a cabo una agregación completa o una transacción empresarial (IBM, 2014). Es una tecnología que usa un grupo de protocolos y estándares que utilizan para intercambiar datos entre aplicaciones. Diferentes aplicaciones de software implementadas en distintos lenguajes de programación, ejecutadas sobre cualquier plataforma pueden utilizar los servicios web con el objetivo de intercambiar datos en redes de ordenadores como Internet. Los servicios web presentan características entre las que se encuentran que permiten ser consumidos desde cualquier tipo de aplicación sin tener en cuenta el lenguaje, facilitan el acceso a su contenido y funcionamiento, estos están orientados a la web y puede ser mezclados para proveer servicios integrados (Aperador, 2012).

Servidor Web

Un servidor Web es un programa que utiliza el protocolo de transferencia de hipertexto, HTTP (Hypertext Transfer Protocol), para servir los archivos que forman páginas Web a los usuarios, en respuesta a sus solicitudes, que son reenviados por los clientes HTTP de sus computadoras. Las computadoras y los dispositivos dedicados también pueden denominarse servidores Web (IBM, 2014).

1.8 Análisis de aplicaciones para la recopilación de datos existentes

Las aplicaciones para hacer encuestas llegaron a revolucionar el mundo, esta clase de herramientas han permitido recolectar información y datos en donde sea sin importar la conectividad, ubicación o características del lugar de estudio. Además, las aplicaciones para encuestas permiten recolectar datos que las encuestas tradicionales realizadas en papel no permitían como fotografías, audio, ubicación y hasta video (Questionpro, 2018). Las Encuestas *offline* permiten recolectar datos utilizando dispositivos móviles, gracias a estos los encuestadores pueden obtener información en diversos formatos como video, fotografía y audio (Moreapp, 2018).

Para crear encuestas *offline* existen una serie de plataformas que ofrecen este servicio:

QuestionPro

Se caracteriza por su plataforma estable, robusta y llena de funciones útiles para crear un estudio de mercados. Con el cuestionario listo solo hace falta descargar la aplicación offline de QuestionPro, vincular la cuenta y listo, es posible comenzar a recolectar datos. Todas las encuestas creadas en esta plataforma tienen un aspecto sencillo y agradable en dispositivos móviles sin necesidad de una configuración especial. Está disponible para dispositivos Android, iPhones, iPads y tabletas sin necesidad de una configuración especial (Questionpro, 2018).

Quicktapsurvey

Es una aplicación de encuestas que trabaja almacenando la información de manera local en el dispositivo. Los usuarios recolectan la información sin la necesidad de utilizar internet. Posteriormente, una vez que se finaliza el proceso de recolección, se sincroniza la información a través de internet con la plataforma para que se pueda visualizar y descargar la información generada. Esta aplicación no requiere de conexión a internet para responder las encuestas, permite trabajar en zonas remotas donde no hay conexión y es fácil de usar. Diseñada para recolectar datos utilizando iPads, iPhones y dispositivos Android (Quicktapsurvey, 2018).

MoreApp

Es una aplicación que proporciona una herramienta para deshacerse de la documentación, ahorrar tiempo y reducir los costes y el uso y desperdicio de papel. Con esta aplicación se puede construir fácilmente cualquier tipo de cuestionario digital. Esta permite realizar muchas funciones gracias a

sus widgets: permite firmar digitalmente; añadir hora, fecha y localización GPS y enviar un email con la encuesta en PDF; entre otros. Está disponible para Android e iOS (Moreapp, 2018).

Estas aplicaciones a pesar de sus amplias funcionalidades presentan como principal limitante que son de software privativo. Esto va en contra de las políticas del estado cubano y de la universidad, que promueven la utilización de software libre, por lo que no serán utilizadas. Se tomarán en cuenta algunas de sus características para el desarrollo de la aplicación.

1.9 Metodología de desarrollo

En el desarrollo de software y con la necesidad de que los proyectos lleguen al éxito y obtener un producto de gran valor, se hace necesario el empleo de una metodología. La metodología seleccionada debe ser aquella que mejor se ajuste a las características del equipo de desarrollo y las exigencias de los usuarios finales. Estas metodologías se clasifican en dos grandes grupos: metodologías pesadas o tradicionales y metodologías ágiles o ligeras:

Las metodologías tradicionales están enfocadas en el control de proceso donde se establecen rigurosamente las actividades que se realizarán y los artefactos que se obtendrán. Estas metodologías han demostrado ser efectivas y necesarias en proyectos de larga duración, pero han presentado problemas en proyectos donde el entorno es muy cambiante y en el cual se exige reducir el tiempo de desarrollo, sin que afecte la calidad del producto (Figueroa, y otros, 2007).

Las metodologías ágiles están enfocadas en el factor humano donde se establece la colaboración con el cliente y el desarrollo incremental del software con iteraciones muy cortas. Se caracterizan por ser flexibles al cambio y por poder reducir el tiempo de desarrollo del software demostrando así que es más importante contar con un software funcional que una documentación excesiva. Estas metodologías han demostrado tener éxito en proyectos pequeños (Figueroa, y otros, 2007).

Para poder llevar a cabo el desarrollo de la propuesta de solución en un corto período de tiempo, donde el cliente esté en constante participación y colaboración y poder realizar cambios en los requisitos si fuese necesario, se opta por una metodología ágil de desarrollo de software en lugar de una metodología tradicional. Las metodologías agiles más utilizadas son XP, SCRUM y AUP, teniendo en cuenta la respuesta a los cambios, el trabajo con el cliente, los planes de entrega de las iteraciones, el trabajo en equipo y el enfoque.

El Proceso Unificado Ágil (AUP) es una versión simplificada del Proceso Unificado Racional (RUP). Describe de manera fácil la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos válidos para RUP. AUP se enfoca principalmente en la gestión de riesgos, haciendo la propuesta de que aquellos elementos con alto riesgo tengan prioridad en el proceso de desarrollo y sean tratados en etapas tempranas del mismo. El proceso establece un

modelo más simple que el de RUP porque integra en una sola las disciplinas de modelado del negocio, requisitos y análisis y diseño, en el caso del resto de las disciplinas coinciden con las restantes de RUP (Flores, 2006).

Las principales características de la metodología AUP según (Flores, 2006) son:

- Abarca siete flujos de trabajos, cuatro de ingeniería y tres de apoyo: Modelado, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyectos y Ambiente.
- El modelado agrupa los tres primeros flujos de RUP (Modelado del negocio, Requerimientos y Análisis y Diseño).
- Dispone de cuatro fases igual que RUP: Creación, Elaboración, Construcción y Transición.

La metodología **Scrum** es un modelo de referencia que define un conjunto de prácticas y roles que pueden tomarse como punto de partida para el proceso de desarrollo de software. Esta metodología está indicada para proyectos con un alto grado de cambios en los requisitos. Su principal característica es que el desarrollo de software se realiza mediante iteraciones denominadas sprint, donde cada sprint representa un incremento del producto y las reuniones diarias que se llevan a cabo a lo largo del proyecto para la coordinación e integración de las actividades a desarrollar (Schwaber, 2010).

Las principales características de la metodología Scrum según (Schwaber, 2010) son las siguientes:

- Permite la creación de equipos auto organizados impulsando la localización de todos los miembros del equipo, y la comunicación verbal entre todos los miembros y disciplinas involucrados en el proyecto.
- Posibilita que los clientes puedan cambiar de idea sobre lo que quieren y necesitan durante el desarrollo del proyecto.
- Maximiza la capacidad del equipo de realizar las entregas rápidamente y de respuesta a los requisitos emergentes.

Extreme Programming (XP) es una metodología ligera de desarrollo de software que se basa en la simplicidad y la comunicación. Esta metodología se caracteriza por centrarse en fortalecer las relaciones interpersonales para el éxito del desarrollo de software, promueve el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen ambiente de trabajo. XP se basa en la interacción continua entre el cliente y el equipo de desarrollo caracterizándose por mantener una conversación fluida, además de poseer soluciones implementadas simples y la capacidad de afrontar los cambios. Los requisitos y el diseño se realizan de forma simple, a través

de las Historias de Usuario, Tareas de Ingeniería y las Tarjetas Clases-Responsabilidad-Colaborador (Letelier, y otros, 2003).

A continuación, se evidencian las **principales prácticas de XP:**

- **El juego de la planificación:** Es un espacio frecuente de comunicación entre el cliente y los programadores. El equipo técnico realiza una estimación del esfuerzo requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas y de cada iteración. Esta práctica se puede ilustrar como un juego, donde existen dos tipos de jugadores: Cliente y Programador. El cliente establece la prioridad de cada historia de usuario, de acuerdo con el valor que aporta para el negocio. Los programadores estiman el esfuerzo asociado a cada historia de usuario. Se ordenan las historias de usuario según prioridad y esfuerzo, y se define el contenido de la entrega y/o iteración, apostando por enfrentar lo de más valor y riesgo cuanto antes. Este juego se realiza durante la planificación de la entrega, en la planificación de cada iteración y cuando sea necesario reconducir el proyecto (Salo, y otros, 2002).
- **Entregas pequeñas:** La idea es producir rápidamente versiones del sistema que sean operativas, aunque obviamente no cuenten con toda la funcionalidad pretendida para el sistema, pero sí que constituyan un resultado de valor para el negocio. Una entrega no debería tardar más 3 meses.
- **Diseño simple:** Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto. La complejidad innecesaria y el código extra debe ser removido inmediatamente. (Salo, y otros, 2002) dice que en cualquier momento el diseño adecuado para el software es aquel que: supera con éxito todas las pruebas, no tiene lógica duplicada, refleja claramente la intención de implementación de los programadores y tiene el menor número posible de clases y métodos.
- **Pruebas:** Se basa en las pruebas realizadas a los principales procesos con el objetivo de detectar futuros errores.
- **Programación en parejas:** Toda la producción de código debe realizarse con trabajo en parejas de programadores. Según (Cockbun, 2000) en un estudio realizado para identificar los costos y beneficios de la programación en parejas, las principales ventajas de introducir este estilo de programación son: muchos errores son detectados conforme son introducidos en el código (inspecciones de código continuas), por consiguiente la tasa de errores del producto final es más baja, los diseños son mejores y el tamaño del código menor (continua discusión de ideas de los programadores), los problemas de programación se resuelven más rápido, se posibilita la transferencia de conocimientos de programación entre los miembros

del equipo, varias personas entienden las diferentes partes sistema, los programadores conversan mejorando así el flujo de información y la dinámica del equipo, y finalmente, los programadores disfrutan más su trabajo. Dichos beneficios se consiguen después de varios meses de practicar la programación en parejas.

- **Propiedad colectiva del código:** Cualquier programador puede cambiar cualquier parte del código en cualquier momento. Esta práctica motiva a todos a contribuir con nuevas ideas en todos los segmentos del sistema, evitando a la vez que algún programador sea imprescindible para realizar cambios en alguna porción de código.
- **Integración continua:** Cada pieza de código es integrada en el sistema una vez que esté lista. Así, el sistema puede llegar a ser integrado y construido varias veces en un mismo día. Todas las pruebas son ejecutadas y tienen que ser aprobadas para que el nuevo código sea incorporado definitivamente. La integración continua a menudo reduce la fragmentación de los esfuerzos de los desarrolladores por falta de comunicación sobre lo que puede ser reutilizado o compartido. (Fowler, 2001.) afirma que el desarrollo de un proceso disciplinado y automatizado es esencial para un proyecto controlado, el equipo de desarrollo está más preparado para modificar el código cuando sea necesario, debido a la confianza en la identificación y corrección de los errores de integración.

La mayoría de las prácticas propuestas por XP no son novedosas, sino que en alguna forma ya habían sido propuestas en ingeniería del software e incluso demostrado su valor. El mérito de XP es integrarlas de una forma efectiva y complementarlas con otras ideas desde la perspectiva del negocio, los valores humanos y el trabajo en equipo (Salo, y otros, 2002).

A continuación, se detallan las **principales ventajas** de la metodología XP según (Figueroa, y otros, 2007) que se hacen vigentes en este proyecto:

- Planificación más transparente para los clientes, conocen las fechas de entrega de las funcionalidades.
- Permite definir en cada iteración cuáles son los objetivos de la siguiente.
- La presión está a lo largo de todo el proyecto y no en una entrega final.

Fundamentación de la metodología a utilizar

Luego del estudio de las metodologías existentes se decide utilizar la metodología XP (programación extrema o extreme programming). XP está especialmente indicada para proyectos de pequeños equipos de trabajo, donde la programación es por parejas, lo que cumple con las características del equipo de desarrollo. Se orienta a una entrega rápida de resultados y una alta flexibilidad. Ayuda a que trabajen todos juntos, en la misma dirección, con un objetivo claro, permitiendo además seguir de forma clara el avance de las tareas a realizar. Se generan los

artefactos mínimos para la comunicación con el cliente. Esta consta de 4 fases: planificación, diseño, desarrollo y pruebas (Universidad Politécnica de Madrid, 2016).

Conclusiones

Al finalizar el presente capítulo se arribó a las siguientes conclusiones parciales:

- El análisis de los elementos teóricos relacionados con el desarrollo de aplicaciones para dispositivos móviles posibilitó una mejor comprensión de los conceptos asociados.
- El análisis del aumento constante de utilización del sistema operativo Android en los dispositivos móviles y la cuota en el mercado que ocupa el mismo, conllevaron a emplear dicha plataforma.
- La metodología de software que guía el proceso de desarrollo es XP, dada la interacción con el cliente, el poco tiempo de duración y que el equipo de desarrollo es pequeño.

Capítulo 2: Análisis, diseño e implementación

Introducción

En el presente capítulo se explica cuál es el funcionamiento del sistema, se identifican las necesidades de los usuarios. Se presenta además la propuesta del sistema y se especifican los requerimientos funcionales y no funcionales. Se presentan los artefactos generados en cada una de las fases de la metodología XP, y otros artefactos que ayudarán en el entendimiento de la propuesta.

2.1 Propuesta del sistema

Con ánimo de aprovechar la gran utilización de los dispositivos móviles, en especial los teléfonos inteligentes y las tabletas, se propone el desarrollo de una aplicación para dispositivos móviles que utilicen el sistema operativo Android para responder encuestas en la UCI. El funcionamiento de la propuesta se describe a continuación:

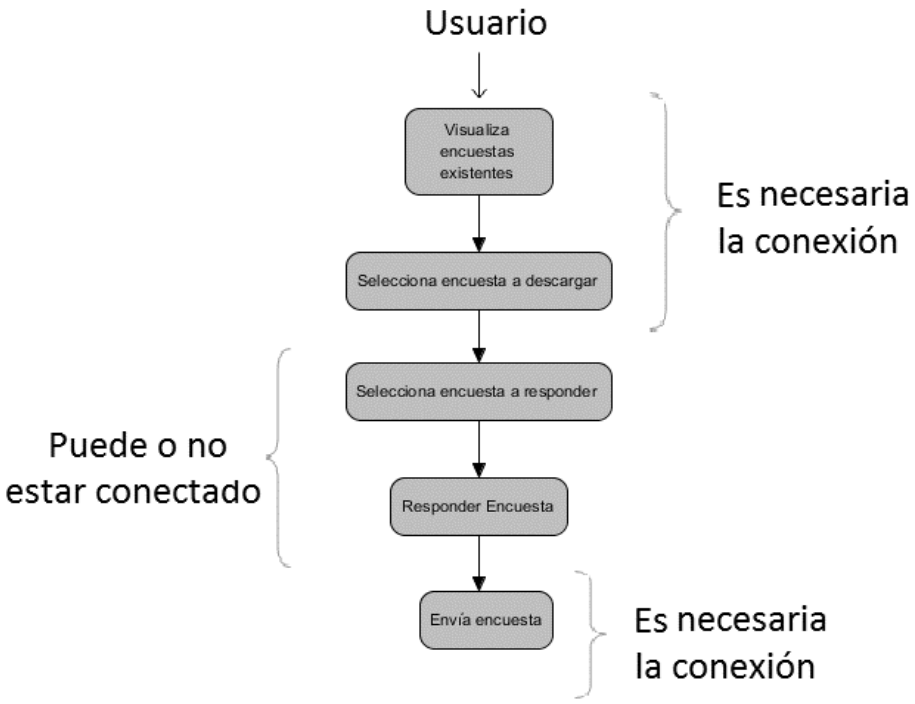

Ilustración 4 Funcionamiento de la Aplicación

A continuación, se describen cada uno de estas tareas:

Visualizar encuestas existentes: Los encuestados pueden acceder al listado de encuestas que están activas en el sistema.

Selecciona encuesta a descargar: Se debe seleccionar la encuesta que se desea descargar hacia el dispositivo.

Selecciona encuesta a responder: Cuando se desee responder la encuesta se debe seleccionar la encuesta y abrirla, este proceso se puede realizar con o sin conexión.

Responder encuesta: Se procede a responder la encuesta, este proceso se puede realizar con o sin conexión.

Enviar encuesta: Una vez que se tenga la encuesta respondida se debe enviar al sistema, para esto se hace necesaria la conexión.

El usuario poseerá la aplicación Android en el dispositivo, que fue previamente descargada. Para responder las encuestas tendrá que conectarse a un punto de acceso wifi, entrar al sitio de gestión de encuestas, seleccionar la encuesta a responder y descargarla, permitiéndole hacer uso de esta de la forma que guste el encuestado, sin la necesidad de estar conectado a la red. Para enviar la encuesta deberá conectarse nuevamente a la wifi.

2.2 Fase de planificación

Para el desarrollo se dividirá el trabajo en las fases propuestas por la metodología XP. En la fase de planificación se efectúa un diálogo entre las partes involucradas en el proyecto, incluyendo al cliente y a los programadores. Se comienza recopilando las historias de usuario, las cuales sustituyen a los casos de uso y se evalúa el tiempo de desarrollo de cada una por parte de los programadores.

2.2.1 Especificación de los requisitos del software

Como parte del análisis previo al diseño de la solución se emplearon las siguientes técnicas de obtención o captura de requisitos:

Según (Escalona, y otros, 2002) la **entrevista** es una técnica muy aceptada dentro de la ingeniería de requisitos. Permiten tomar conocimiento del problema y comprender los objetivos de la solución buscada, para tener una amplia visión de las necesidades del usuario. Se realizaron entrevistas con el cliente para obtener información acerca de las necesidades y perspectivas de la solución.

La **tormenta de ideas** una técnica de reuniones en grupo cuyo objetivo es que los participantes muestren sus ideas de forma libre. Consiste en la acumulación de ideas y/o información sin ser evaluadas (Escalona, y otros, 2002). En varias reuniones se desarrollaron tormentas de ideas con la participación del equipo de desarrollo y el cliente. Como resultado se logró generar opiniones sobre los requisitos a satisfacer por el software.

Los requerimientos del sistema constituyen las condiciones indispensables que debe poseer cualquier sistema para su correcto funcionamiento (Departamento de Ciencias de la Computacion e I.A, 2014).

2.2.1.1 Requisitos Funcionales

Expresan la naturaleza del funcionamiento del sistema (cómo interacciona el sistema con su entorno y cuáles van a ser su estado y funcionamiento) (Departamento de Ciencias de la Computacion e I.A, 2014). Luego de aplicadas las técnicas de entrevista y tormenta de ideas se identificaron un total de 7 requisitos funcionales (RF).

RF1: Mostrar encuestas disponibles

RF2: Descargar encuesta

RF3: Mostrar encuestas descargadas

RF4: Seleccionar encuesta a responder

RF5: Abrir encuesta

RF6: Responder encuesta

RF7: Enviar encuesta

RF8: Eliminar encuesta descargada

RF9: Guardar respuesta

2.2.1.2 Requisitos No Funcionales

Los requerimientos no funcionales son características que describen alguna forma o restricción para la realización de algún requerimiento (funcionalidad) o conjunto de ellas e inclusive todos los requerimientos. Se consideran los atributos del sistema, propiedades que debe tener el producto. Restricciones sobre el espacio de posibles soluciones (Departamento de Ciencias de la Computacion e I.A, 2014). A continuación, se listan los requisitos no funcionales de la aplicación Android a desarrollar.

1. **Usabilidad**

RNF1: El sistema debe tener un diseño sencillo, permitiendo la utilización del sistema sin mucho entrenamiento.

2. **Soporte**

RNF2: Garantía de instalación y prueba del sistema.

3. **Rendimiento**

RNF3: El tiempo de respuesta de cada funcionalidad debe ser menor o igual a 5 segundos.

4. Requerimientos de software

RNF4: Para la utilización del sistema se requiere el SO Android en su versión 4.0 o superior.

5. Requerimientos de hardware

RNF5: Capacidad disponible de 6 Mb en la memoria del dispositivo para la instalación y ejecución de la aplicación.

RNF6: El dispositivo debe disponer de conexión wifi y al menos 10Mb de espacio libre de memoria.

RNF7: La base de datos debe estar guardada en la raíz del teléfono.

2.2.2 Historias de usuario

Las historias de usuario son la técnica utilizada en XP para especificar los requisitos del software. Las mismas son escritas por los clientes como las tareas que el sistema debe hacer y su construcción depende principalmente de la habilidad que tenga el cliente para definirlas. Son utilizadas como el único documento de requisitos que se genera en XP.

A continuación, se muestran las historias de usuario Abrir encuesta, Enviar encuesta y Descargar encuesta, que son HU de alta criticidad, el resto de las historias se puede observar en el Anexo 1.

Historia de Usuario	
Número: HU-5	**Nombre historia de usuario**: Abrir encuesta
Usuario: encuestado	**Iteración Asignada:** 3
Prioridad del negocio: Alta	**Puntos estimados:** 2 días
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: Permite al usuario abrir la encuesta para poder responderla desde su dispositivo móvil. **Observaciones:** • La encuesta se puede abrir tanto de manera online como offline.	

Tabla 1 HU Abrir encuesta

Historia de Usuario	
Número: HU-7	**Nombre historia de usuario**: Enviar encuesta
Usuario: encuestado	**Iteración Asignada:** 3
Prioridad del negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semana
Descripción: Permite al usuario enviar la encuesta respondida a la web para que sea procesada. **Observaciones:** • Se debe estar conectado a la red para poder enviar la encuesta.	

Tabla 2 HU Enviar encuesta

Historia de Usuario	
Número: HU-2	**Nombre historia de usuario**: Descargar encuesta
Usuario: encuestado	**Iteración Asignada:** 3
Prioridad del negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semana

Descripción: Permite al usuario descargar la encuesta a responder hacia su dispositivo móvil. **Observaciones:** • Se debe estar conectado a la wifi para poder descargar la encuesta.

Tabla 3 HU Descargar encuesta

2.2.3 Plan de iteraciones

Las historias de usuarios seleccionadas para cada plan de entrega son desarrolladas y probadas en un ciclo de iteración de acuerdo al orden preestablecido, se estima el tiempo de duración de cada una de ellas. Cada historia de usuario se traduce en tareas específicas de programación (Joskowicz, 2012).

Iteración	HU	Prioridad	Duración de la HU	Duración total
1	Mostrar encuestas disponibles	Alta	4 días	17 días
	Descargar encuesta	Alta	7 días	
	Mostrar encuestas descargadas	Media	4 días	
	Seleccionar encuesta a responder	Alta	2 días	
2	Abrir encuesta	Alta	2 días	14 días
	Responder encuesta	Alta	7 días	
	Enviar encuesta	Alta	5 días	
3	Guardar respuesta	Alta	3 días	6 días
	Eliminar encuesta	Baja	3 días	

Tabla 4 Plan de iteraciones

En la tabla anterior se mostró el plan de iteraciones a realizar. Con el mismo se puede arrojar que el sistema será realizado en tres iteraciones, con 17, 14 y 6 días cada una, para un total de 37 días.

2.3 Fase de diseño

En la fase de diseño se confeccionan las tarjetas Clase Responsabilidad Colaborador (CRC) para crear diseños de clases orientados a responsabilidades. Se selecciona la arquitectura adecuada para el desarrollo de la herramienta. Además, se generan los artefactos modelo de dominio, modelo de datos y los estándares de codificación, los cuales no están comprendidos entre los artefactos a generar por la metodología, pero dada su marcada importancia para el entendimiento del sistema se ha decidido incluir su diseño.

2.3.1 Modelo de dominio

El modelo de dominio o modelo conceptual, es una representación de las clases conceptuales del mundo real, no de componentes de software. Representa en forma de diagrama de clases donde figuran los principales conceptos que se manejan en el dominio del sistema en desarrollo y roles del sistema en cuestión. El modelo desarrollado no consiste en un conjunto de diagramas que describen objetos de software con responsabilidades, o clases de software, sino que se puede considerar como un diccionario visual de las abstracciones relevantes del dominio (UCI, 2012).

Para una mejor comprensión del modelo de dominio mostrado se realiza a continuación una explicación de las clases que lo conforman:

Usuario: cualquier persona que estudie o trabaje en la Universidad, sin importar su categoría o lugar de residencia.

Encuesta: serie de preguntas que se le hace a un grupo de personas para detectar la opinión pública sobre un asunto determinado.

Pregunta: enunciado interrogativo que se emite con la intención de obtener alguna información.

Opción: posible elección.

Respuesta: serie de palabras escritas con que se responde.

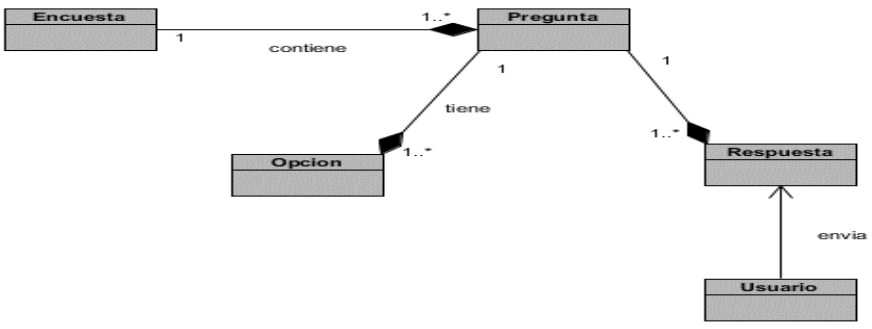

Ilustración 5 Modelo de dominio

2.3.2 Tarjetas clase-responsabilidad-colaborador (CRC)

Las tarjetas CRC se dividen en tres secciones que contienen la información del nombre de la clase, sus responsabilidades y sus colaboradores. Tienen la finalidad de obtener un diseño simple y fácil de comprender por parte de los programadores. A continuación, se muestra una tarjeta CRC perteneciente a la clase controladora ListadoEncuestaPresentador, las demás tarjetas pueden encontrarse en el Anexo 2:

Tarjeta CRC	
Clase: ListadoEncuestaPresentador	
Responsabilidades: ObtenerEncuestaBaseDatos MostrarEncuesta	**Colaboraciones:** Encuesta ConstructorEncuesta IListadoEncuestaView

Tabla 5 Tarjeta CRC ListadoEncuestaPresentador

2.3.3 Modelo de datos

Un modelo de datos es un conjunto de conceptos, reglas y convenciones que permite describir la representación lógica y física de los datos persistentes de la base de datos. Básicamente consiste en una descripción de algo conocido como contenedor de datos (donde se guarda la información), así como los métodos a almacenar y recuperar información de dichos contenedores.

Consiste en:

- Objetos (entidades que existen y que se manipulan).
- Atributos (características básicas de estos objetos).
- Relaciones (forma en que se enlazan los distintos objetos entre sí).

A continuación, se muestra el modelo de datos resultante de la base de datos:

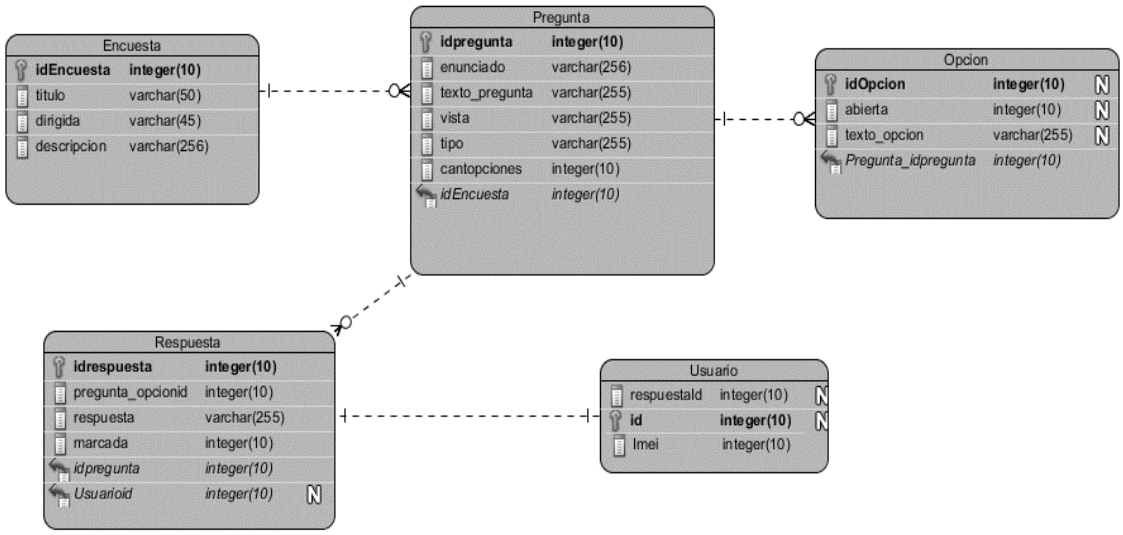

Ilustración 6 Diagrama de base de datos

2.3.4 Modelo conceptual

Para la realización del sistema es necesario tener en cuenta los tipos de preguntas que existen y la forma en la que se visualizarán las mismas en la aplicación Android. A continuación, se muestra un modelo conceptual en el que se evidencian los conceptos asociados a las encuestas, los tipos de preguntas, así como deben visualizarse en la aplicación Android dichos tipos:

- **Preguntas cerradas:** son las que requiere del encuestado simplemente selecciona la respuesta a partir de una o varias alternativas que el encuestador le presenta.
- **Preguntas abiertas:** son las que requiere que el encuestado elabore sus propias respuestas a las preguntas formuladas en la encuesta.
- **Preguntas mixtas:** estas preguntas son una combinación de las dos anteriores.

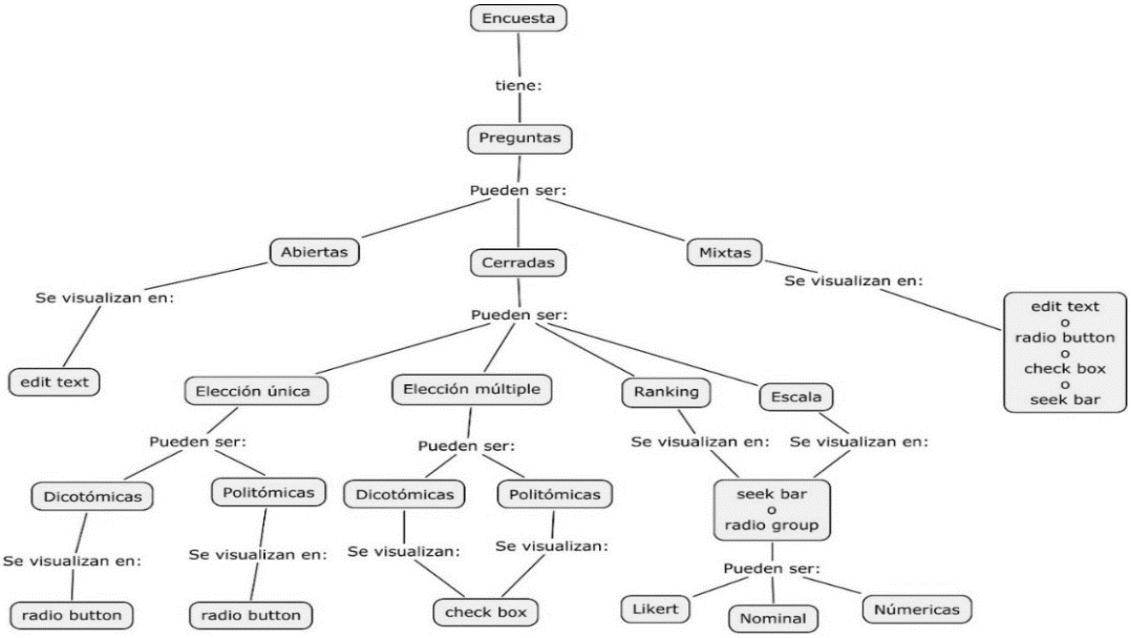

Ilustración 7 Modelo conceptual relacionado con las encuestas

2.3.5 Patrones de diseño

Según (Larman, 1999) un patrón es una descripción de un problema y su solución, que recibe un nombre y que puede emplearse en otros contextos. Muchos patrones ofrecen orientación sobre cómo asignar las responsabilidades a los objetos ante determinada categoría de problemas. Para el diseño y el desarrollo de la solución se hizo uso de un patrón arquitectónico y de un grupo de patrones de diseño.

Un patrón de diseño es una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias (Larman, 1999).

Patrones generales de software para asignación de responsabilidades (GRASP)

Los patrones GRASP (patrones generales de software para asignación de responsabilidades, siglas de General Responsibility Assignment Software Patterns) describen los principios fundamentales de la asignación de responsabilidades a objetos. A continuación se mencionan los patrones utilizados en el diseño de la solución.

- **Experto:** Asigna una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.

El uso de este patrón resulta de utilidad en las clases del modelo, las cuales contienen toda la información relacionada con los objetos persistentes que representan.

La figura muestra el uso de este patrón en la aplicación, perteneciente a la clase Pregunta:

```
private String enunciado;

public String getEnunciado(){
 return enunciado;
}
public String setEnunciado(String cadena){
 this.enunciado= cadena;
}
```

Ilustración 8 Patrón experto

- **Creador:** Guía la asignación de responsabilidades relacionadas con la creación de objetos.

Este patrón se pone de manifiesto en las clases controladoras del sistema. Brinda soporte a un bajo acoplamiento y mejora la reutilización. La figura muestra el uso de este patrón en la clase EncuestaDBhelper.

```
public void insertarEncuesta(ContentValues contentValues){
        SQLiteDatabase db = this.getWritableDatabase();
        db.insert(ConstantesBD.NT_ENCUESTA, null, contentValues);
        db.close();
```

Ilustración 9 Patrón Creador

- **Bajo acoplamiento:** Medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo acoplamiento no depende de muchas otras.
- **Alta cohesión:** Medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme.
- **Controlador:** Objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define el método de su operación.

Este patrón se evidencia en las clases controladoras, responsables de implementar las funcionalidades pertenecientes a una interfaz determinada. La figura muestra el uso de este patrón en la clase controladora ListadoEncuestaPresentador.

```
import android.content.Context;
import java.util.ArrayList;
import Modelo.Encuesta;
import basededatos.ConstructorEncuesta;
import davidproduction.Encuestamania.IListadoEncuestaView;
    public class ListadoEncuestaPresentador implements IListadoEncuestaPresentador {
        private IListadoEncuestaView iListadoSerieView;
        private Context context;
        private ArrayList<Encuesta> Encuesta;
        private ConstructorEncuesta constructorEncuesta;
        public ListadoEncuestaPresentador(IListadoEncuestaView iListadoSerieView, Context context) {
            this.iListadoSerieView = iListadoSerieView;
            this.context = context;
            obtenerEncuestaBaseDatos();
        }
    @Override
        public void obtenerEncuestaBaseDatos() {
            constructorEncuesta = new ConstructorEncuesta(context);
            Encuesta = constructorEncuesta.obtenerDatos();
            mostrarEncuestaRV();
        }
        @Override
        public void mostrarEncuestaRV() {
            iListadoSerieView.inicializarAdaptadorRV(iListadoSerieView.crearAdaptador(Encuesta));
            iListadoSerieView.generarLinearLayoutVertical();
        }
    }
```

Ilustración 10 Patrón Controlador

Banda de los cuatro (GOF)

Los patrones de diseño GOF (siglas de Gang of Four) son 23 y están clasificados según su ámbito en objeto y de clase y según su propósito en creacionales, estructurales y de comportamiento, detallados a continuación:

- **Comportamiento:** Los patrones de comportamiento están relacionadas con los algoritmos y la asignación de responsabilidades entre los objetos. Son utilizados para organizar, manejar y combinar comportamientos.
- **Creacionales:** Los patrones creacionales abstraen el proceso de creación de instancias y ocultan los detalles de cómo los objetos son creados o inicializados.

Para el desarrollo de la solución se empleó el patrón **Decorador**, que pertenece al grupo de los patrones estructurales. El mismo en el sistema permite añadir dinámicamente nuevas responsabilidades a un objeto, proporcionando una alternativa flexible a la herencia para extender funcionalidad. Se aplica con la intención de proporcionar una forma flexible de introducir o eliminar funcionalidad de un componente sin modificar su apariencia externa o su función.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_alignParentBottom="true"
android:layout_centerHorizontal="true"
android:layout_centerInParent="true"
android:layout_centerVertical="true"
android:orientation="vertical"
tools:context="cu.uci.tesis.davideileen.sistemadeencuesta.PrincipalActivity">
<Button
android:id="@+id/btnEncuestasDisponibles"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Encuestas Disponibles"
android:onClick="VistaVerEncuesta"
tools:layout_constraintTop_creator="1"
tools:layout_constraintRight_creator="1"
app:layout_constraintRight_toRightOf="parent"
android:layout_marginTop="191dp"
tools:layout_constraintLeft_creator="1"
app:layout_constraintLeft_toLeftOf="parent"
```

Ilustración 11 Patrón decorador

El patrón creacional **Constructor virtual** (Builder) también fue utilizado, este es un patrón que abstrae el proceso de creación de un objeto complejo, centralizando dicho proceso en un único punto.

```
import android.database.sqlite.SQLiteDatabase;
import android.widget.EditText;
import java.util.ArrayList;
import Modelo.Encuesta;
import cu.uci.tesis.davideileen.sistemadeencuesta;
public class ConstructorEncuesta {
   private Context context;
    public ConstructorEncuesta(Context context) {
        this.context = context; }
    public ArrayList<Encuesta> obtenerDatos() {
        EncuestaDbHelper db = new EncuestaDbHelper(context);
    return db.obtenerTodosLasEncuestas();}
public void insertarEncuestas (String n, String t, String c){
EncuestaDbHelper db = new EncuestaDbHelper(context);
ContentValues contentValues = new ContentValues();
    contentValues.put(ConstantesBD._ID, id);
    contentValues.put(ConstantesBD.TITULO, t);
    contentValues.put(ConstantesBD.Dirigida, dir);
    contentValues.put(ConstantesBD.DESCRIPCION, des);
    db.insertarEncuesta(contentValues);
} public void ModificarEncuestas (int id, String n, String t, String c){
        EncuestaDbHelper db = new EncuestaDbHelper(context);
        ContentValues contentValues = new ContentValues();
        contentValues.put(ConstantesBD.NOMBRE, n);
        contentValues.put(ConstantesBD.TEMPORADA, t);
        contentValues.put(ConstantesBD.CAPITULO, c);
        db.ModificarEncuesta(id, contentValues);}
public void EliminarEncuesta(int id) {
        EncuestaDbHelper db  = new EncuestaDbHelper(context);
        db.Eliminar(id);
    }
```

Ilustración 12 Patrón Builder

Mediante el uso de estos patrones se garantizó la asignación de las responsabilidades correspondientes a cada una de las clases, y el menor número de dependencias y relaciones entre estas, lo que contribuye a un correcto diseño del sistema, y facilita su implementación.

2.3.6 Patrón arquitectónico

Los patrones arquitectónicos representan el nivel más alto dentro del sistema de patrones y expresan el esquema de la estructura fundamental de la organización para sistemas de software (Perez, 2007).

Para el desarrollo de la aplicación Android se selecciona **Modelo Vista Presentador (MVP)**, el cual es la arquitectura utilizada por el SO Android. Esta es una derivación del patrón arquitectónico Modelo Vista Controlador (MVC). Es un patrón de arquitectura cuya finalidad es la de separar la lógica de negocio de la Vista, lo que lo diferencia del MVC es que el Presentador se comunica con la Vista a través de una interfaz, de forma que no tiene conocimiento alguno de la existencia de esta, a diferencia del MVC, donde el Controlador si tiene conciencia de su representación. Otra diferencia es que en MVP el Modelo y la Vista no tienen correspondencia alguna, es el Presentador el que actúa entre ambos (Develapps, 2016).

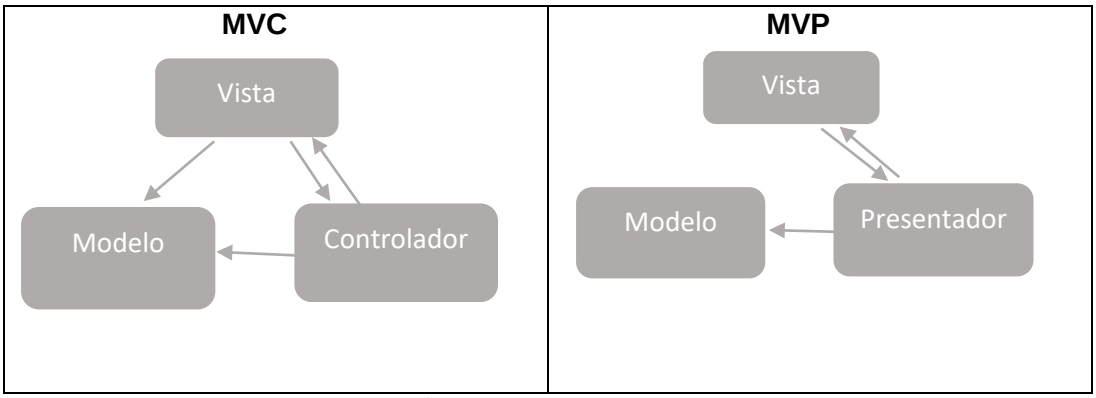

Ilustración 13 Diferencia entre MVC y MVP

El código queda dividido de la siguiente manera (Develapps, 2016):

- **Modelo:** Su responsabilidad es la de proporcionar, desde clases externas, los datos que necesita el Presentador, y dotarle de métodos para poder modificarlos.
- **Vista:** Suelen ser los Fragment o Activity correspondientes, quiénes implementan una interfaz con todos los métodos, los que permiten modificar y rellenar con datos los layouts, que se definen haciendo uso de los métodos correspondientes de los que dota Android.
- **Presentador:** Es una clase que se encarga de modificar la Vista, por lo que tiene una referencia a su interfaz desde donde lo hace. De esta forma, da igual la representacion de esa vista, el Presentador solo se encarga de decidir qué se muestra y no cómo se muesta.

2.3.7 Estándares de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Al comenzar un proyecto de software, se debe establecer un estándar de codificación para asegurarse de que todos los programadores del proyecto trabajen de forma coordinada. Para la realización del presente trabajo se establecen tres estándares de codificación:

Variable:

- Los nombres de las variables deben ser cortos y significativos.
- La elección de un nombre de variable debe ser mnemotécnica[3], esto es, diseñado para demostrar el propósito de su uso a cualquier observador.

Constantes:

- Los nombres de variables declaradas como constante deben ser todas en mayúsculas con palabras separadas por guion bajo ("_").

- Se debe seguir las mismas convenciones que se usan para variables con respecto a los prefijos para tipo de dato.

Funciones:

- Los nombres de los métodos deben empezar con una letra mayúscula y el resto de letras deben estar escritas en minúscula.
- Los nombres de los métodos deben ser verbos o palabras que identifiquen de manera general el objetivo del método.
- Los nombres de los métodos no pueden contener espacios ni caracteres especiales, sólo son permitidas las letras de la "a" a la "z" y los números del 0 al 9.
- Si el nombre de método requiere estar compuesto por más de una palabra, cada palabra adicional debe empezar con mayúscula.

[3] Una regla mnemotécnica es una oración corta y fácil de recordar que ayuda de manera artificiosa a relacionar palabras, con el objetivo de memorizar conceptos con más facilidad.

2.4 Fase de desarrollo

En esta fase de desarrollo o codificación los clientes y los desarrolladores del proyecto deben estar en comunicación para que los programadores puedan codificar todo lo necesario para el proyecto que se requiere. En esta fase está incluido todo lo referente a la codificación o programación del proyecto. Es importante en esta fase identificar y especificar las tareas de ingeniería a desarrollar por iteraciones, así como los lenguajes y herramientas necesarios para la realización del sistema.

2.4.1 Lenguajes, tecnologías y herramientas a utilizar

A fin de garantizar a lo largo de todo el proceso de desarrollo, que el sistema cumpla con los requerimientos que fueron planteados a desarrollar, se utilizaron un conjunto de tecnologías, lenguajes y herramientas, la cuales se describen a continuación.

2.4.1.1 Lenguajes de programación.

Un lenguaje de programación no es más que un lenguaje artificial que se puede usar para controlar el comportamiento de una máquina, principalmente una computadora. Estos se componen de un grupo de reglas sintácticas y semánticas que permiten enunciar instrucciones que luego serán interpretadas (Alegsa, 2009).

Java v1.8

Es el lenguaje de programación oficial y el soportado de forma nativa por Android, tiene gran compatibilidad con dicho SO por lo que se utilizará para el desarrollo del subsistema propuesto. Java es un lenguaje de programación con el que se puede realizar cualquier tipo de programa. Actualmente es un lenguaje muy extendido, por lo que cobra cada vez más importancia, tanto en el ámbito de Internet como en la informática en general. Está desarrollado por la compañía Sun Microsystems (Alvarez, 2001).

.

Lenguaje de modelado

UML v2.0

El Lenguaje Unificado de Modelado (**Unified Modeling Language**) es un lenguaje de modelado visual usado para especificar, visualizar, construir y documentar artefactos de un software. Capta la información sobre la estructura estática y el comportamiento dinámico de un sistema. Brinda apoyo a la mayoría de los procesos de desarrollo orientados a objetos (Rumbaugh, 2000).

2.4.1.2 Herramientas utilizadas.

Entorno de desarrollo integrado (IDE)

Para el desarrollo de aplicaciones para el SO Android el entorno más utilizado es **Android Studio**, el cual es un IDE[4] de desarrollo multiplataforma creado por Google e implementado en Java. Es

[4] Entorno de desarrollo integrado, programa informático compuesto por un conjunto de herramientas de programación (Alegsa, 2009).

una especie de escritorio de trabajo para un desarrollador. Este cuenta con algunas herramientas que facilitan el desarrollo de las aplicaciones, como por ejemplo el poder pre visualizar las aplicaciones en diferentes teléfonos inteligentes y tabletas electrónicas para saber cómo está quedando según como se está implementando y cómo se ve en los diferentes tipos de pantalla que existen (Android Studio, 2018). Este IDE se utilizará en su versión 3.0.1.

Para realizar aplicaciones sobre la plataforma Android se instala el plugin ADT y el SDK en el IDE de desarrollo. Un SDK de Android contiene las bibliotecas, el emulador, documentación, ejemplos de código, tutoriales y una versión de la plataforma (Larramendi, 2012).

Gestor de dase de datos

SQLite

Es una herramienta de software libre, que permite almacenar información en dispositivos de forma sencilla, eficaz, potente, rápida y en equipos con pocas capacidades de hardware, como puede ser una PDA o un teléfono celular. Agrega extensiones que facilitan su uso en cualquier ambiente de desarrollo. Esto permite que SQLite soporte desde las consultas más básicas hasta las más complejas del lenguaje SQL (Sqlite, 2014).

Herramienta de modelado

Visual Paradigm v 8.0

Es una herramienta profesional muy potente que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, generar código desde diagramas y generar documentación. Esta herramienta soporta UML (siglas de Unified Modeling Language), Notación para el Modelado de Procesos de Negocio (BPMN) y permite realizar ingeniería tanto directa como inversa (Pressman, 2002).

Herramienta de prototipado

Justinmind Prototyper v8.3

Es una herramienta para desarrolladores que facilita la tarea de prototipado de aplicaciones móviles, permitiendo crear bocetos interactivos sin necesidad de generar código (Applesfera, 2014). Permite confeccionar de forma rápida un diseño funcional, permite pruebas de concepto, acciones y eventos sobre los elementos de diseño colocados en las pantallas que podemos confeccionar con sólo arrastrar y soltar (Loogic, 2009).

Retrofit

Para la utilización de servicios se utilizó la biblioteca Retrofic, el cual es un cliente REST para Android y Java, desarrollada por Square, muy simple y fácil de aprender. Permite hacer peticiones GET, POST, PUT, PATCH, DELETE y HEAD; gestionar diferentes tipos de parámetros y parsear automáticamente la respuesta a un POJO (Plain Old Java Object) (Cadenas, 2016).

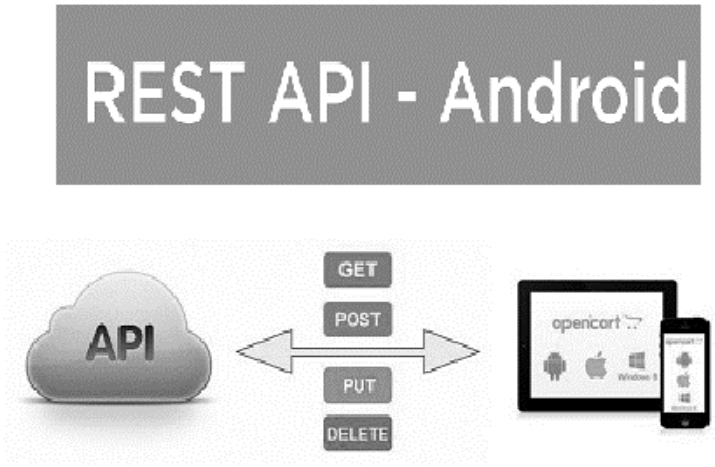

Ilustración 14 Cliente Rest para Android. Fuente (Cadenas, 2016)

2.4.2 Tareas de ingeniería

Las tareas de ingeniería son actividades que se derivan de las HU para simplificar su implementación. A continuación, se muestran las tareas a desarrollar para la implementación, por cada HU.

Iteración	Historia de usuario	Tareas de ingeniería
1	Mostrar encuestas disponibles	Implementar una funcionalidad que permita al usuario observar todas las encuestas disponibles.
	Descargar encuesta	Implementar una funcionalidad que permita al usuario descargar la encuesta hacia el dispositivo móvil.
	Mostrar encuestas descargadas	Implementar una funcionalidad que permita al usuario observar todas las encuestas que ya ha descargado.
	Seleccionar encuesta a responder	Implementar una funcionalidad que permita al usuario seleccionar la encuesta que desee responder

2	Abrir encuesta	Implementar una funcionalidad que permita al usuario abrir la encuesta en su dispositivo para después responderla.
	Responder encuesta	Implementar una funcionalidad que permita al usuario enviar la encuesta desde su dispositivo móvil hacia la web.
	Enviar encuesta	Implementar una funcionalidad que permita al usuario responder una pregunta.
3	Guardar respuesta	Implementar una funcionalidad que permita guardar temporalmente en la base de datos del dispositivo las respuestas que el usuario dio por cada respuesta.
	Eliminar encuesta descargada	Implementar una funcionalidad que permita al usuario eliminar la encuesta que ya no desee tener en su dispositivo.

Tabla 6 Tareas de Ingeniería

2.4.2.2 Tareas detalladas

A continuación, se muestran las tareas de ingeniería detalladas correspondientes a las historias de usuario Crear encuesta, Modificar encuesta y Descargar encuesta, el resto se pueden observar en el Anexo 3.

TAREA DE INGENIERÍA	
Número de la tarea:5	**HU:** Abrir encuesta
Nombre de la tarea: Implementar funcionalidad Abrir encuesta.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2 días
Fecha de inicio:21/4/18	**Fecha de fin:23/4/18**
Descripción: Implementar una funcionalidad que permita al usuario abrir la encuesta en su dispositivo para después responderla.	

Tabla 7 Tarea de ingeniería perteneciente a la HU Abrir encuesta

TAREA DE INGENIERÍA

Número de la tarea:7	**HU:** Enviar encuesta
Nombre de la tarea: Implementar funcionalidad Enviar encuesta.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 5 días
Fecha de inicio:2/5/18	**Fecha de fin:7/5/18**
Descripción: Implementar una funcionalidad que permita al usuario enviar la encuesta desde su dispositivo móvil hacia la web.	

Tabla 8 Tarea de ingeniería perteneciente a la HU Enviar encuesta

TAREA DE INGENIERÍA	
Número de la tarea:2	**HU:** Descargar encuesta
Nombre de la tarea: Implementar funcionalidad Descargar encuesta.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 7 días
Fecha de inicio:9/4/18	**Fecha de fin:13/4/18**
Descripción: Implementar una funcionalidad que permita al usuario descargar la encuesta hacia el dispositivo móvil.	

Tabla 9 Tarea de Ingeniería Descargar encuesta

2.4.3 Interfaz de usuario

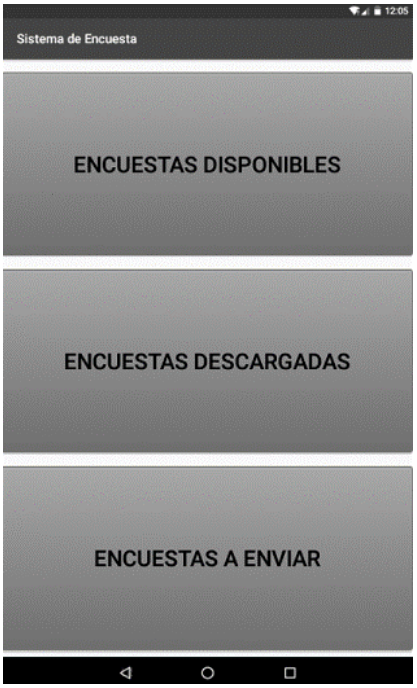

Ilustración 15 Interfaz: Vista principal

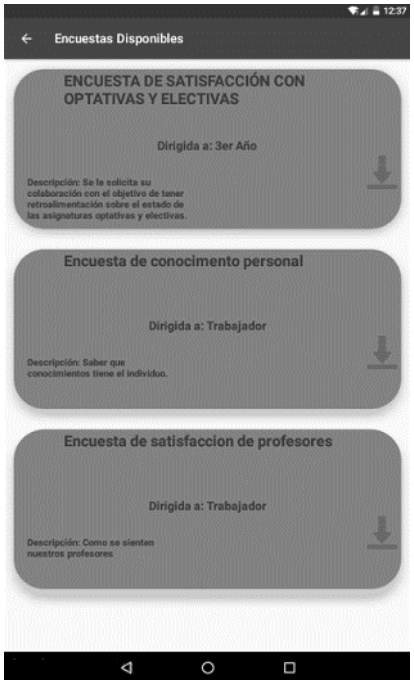

Ilustración 16 Prototipo de interfaz: Encuestas disponibles

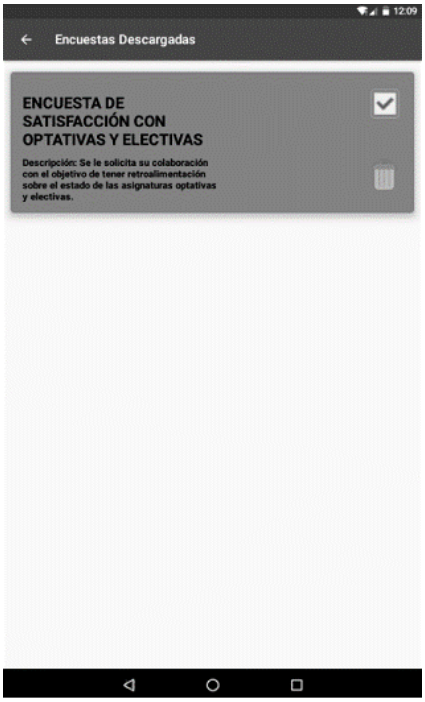

Ilustración 17 Prototipo de interfaz: Lista de encuestas a responder

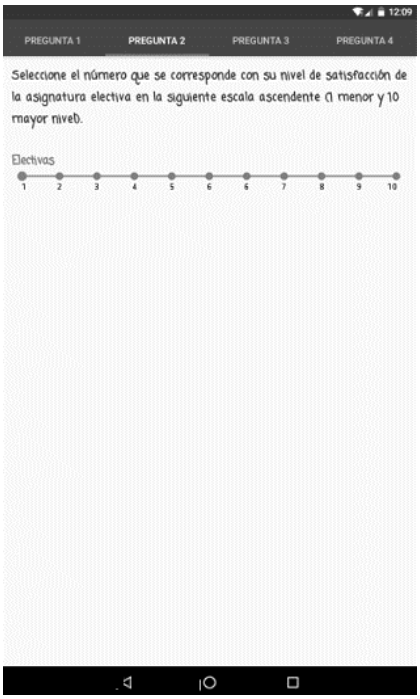

Ilustración 18 Interfaz: Encuesta a responder

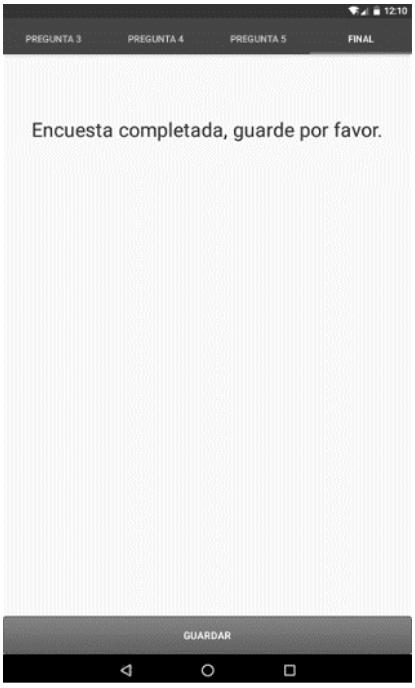

Ilustración 19 Interfaz Encuesta a guardar

Conclusiones

Al finalizar el presente capítulo se arribó a las siguientes conclusiones parciales:

- La confección de los artefactos historias de usuario, plan de iteraciones, tarjetas CRC, y el modelo de datos permitieron organizar el proceso de desarrollo del sistema.
- Se fundamentó la elección del patrón Modelo Vista Presentador para la arquitectura del sistema por ser el patrón arquitectónico propuesto para el sistema operativo Android.
- El uso de los estándares de codificación permitió proyectar calidad en el código generado.
- La confección de las tareas de ingeniería por iteración y de las tareas detalladas permitió simplificar la implementación de las historias de usuario.
- Realizada una revisión de los principales elementos teóricos para el desarrollo del trabajo, se decide desarrollar la aplicación utilizando el IDE Android Studio, con el uso del lenguaje de programación Java y como gestor de base de datos SQLite.

Capítulo 3: Validación de la solución

Introducción

En este capítulo se muestran los diseños de casos de prueba realizados para cada requisito funcional del sistema, permitiendo la validación de la propuesta de solución.

3.1 Fase de pruebas

El proceso de prueba se realiza de forma continua para asegurar durante todo el proceso de desarrollo, el éxito del producto. Esto permite detectar los errores en un plazo de tiempo corto.

Pruebas

Las pruebas presentan una interesante anomalía para el ingeniero del software. El ingeniero crea una serie de casos de prueba que intentan demoler el software construido. La prueba demuestra hasta qué punto las funciones del software parecen funcionar de acuerdo con las especificaciones y parecen alcanzarse los requisitos de rendimiento. Además, los datos que se van recogiendo a medida que se llevan a cabo las pruebas proporcionan una buena indicación de la fiabilidad del software y, de alguna manera, indican la calidad del software como un todo (Pressman, 2002).

3.1.1 Pruebas de Caja Blanca

Las pruebas de caja blanca (también conocidas como pruebas de caja de cristal o pruebas estructurales) se centran en los detalles procedimentales del software, por lo que su diseño está fuertemente ligado al código fuente. Habitualmente se aplican a las unidades de software. Su objetivo es comprobar los flujos de ejecución dentro de cada unidad, pero también pueden probar los flujos entre unidades durante la integración, e incluso entre subsistemas, durante las pruebas de sistema. La prueba de caja blanca es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los casos de prueba (Pressman, 2002).

La prueba del camino básico, es una prueba de “caja blanca” que consiste en verificar el código de nuestros sistemas de manera que comprobemos que todo funciona correctamente, es decir, se debe verificar que todas las instrucciones del programa se ejecutan por lo menos una vez (Mouse, 2017). Esta técnica permite al diseñador de casos de prueba obtener la complejidad lógica de un procedimiento o algoritmo y usar esta como guía para la definición de un conjunto básico de caminos de ejecución. Los casos de prueba obtenidos del conjunto básico garantizan que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa (Pressman, 2010).

La complejidad ciclomática es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa. Cuando se usa en el contexto del método de prueba del camino básico, el valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y proporciona un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecuta cada sentencia al menos una vez (Pressman, 2010). Se selecciona para la realización de las pruebas unitarias o pruebas de caja blanca el método o técnica prueba del camino básico, junto a la métrica complejidad ciclomática.

Los pasos para desarrollar la prueba del camino básico son (Mouse, 2017):

1.- Dibujar el grafo de flujo

2.- Calcular la complejidad ciclomática

3.- Determinar el conjunto básico de caminos independientes

A continuación, se describen cada uno de estos pasos.

Se selecciona como ejemplo el método deserializarEncuestaDeJson() de la clase EncuestaDeserializador.

```
private ArrayList<Encuesta> deserializarEncuestaDeJson(JsonArray encuestaResponseDatos){
    ArrayList<Encuesta> encuestas = new ArrayList<>();
    for (int i = 0; i < encuestaResponseDatos.size() ; i++) {
        JsonObject encuestaResponseDatoObject = encuestaResponseDatos.get(i).getAsJsonObject();
        int id = encuestaResponseDatoObject.get(ConstantesBD._ID).getAsInt();
        String titulo = encuestaResponseDatoObject.get(ConstantesBD.TITULO).getAsString();
        String dirigida = encuestaResponseDatoObject.get(ConstantesBD.DIRIGIDA).getAsString();
        String descripcion = encuestaResponseDatoObject.get(ConstantesBD.DESCRIPCION).getAsString();
        Encuesta encuestaActual = new Encuesta();
        encuestaActual.setIdencuesta(id);
        encuestaActual.setTitulo(titulo);
        encuestaActual.setDirigida(dirigida);
        encuestaActual.setDescripcion(descripcion);
        encuestas.add(encuestaActual);
    }
    return encuestas;
}
```

Ilustración 20 Método deserializarEncuestaDeJson()

Partiendo del fragmento de código anterior se obtiene el siguiente **grafo de flujo:**

.

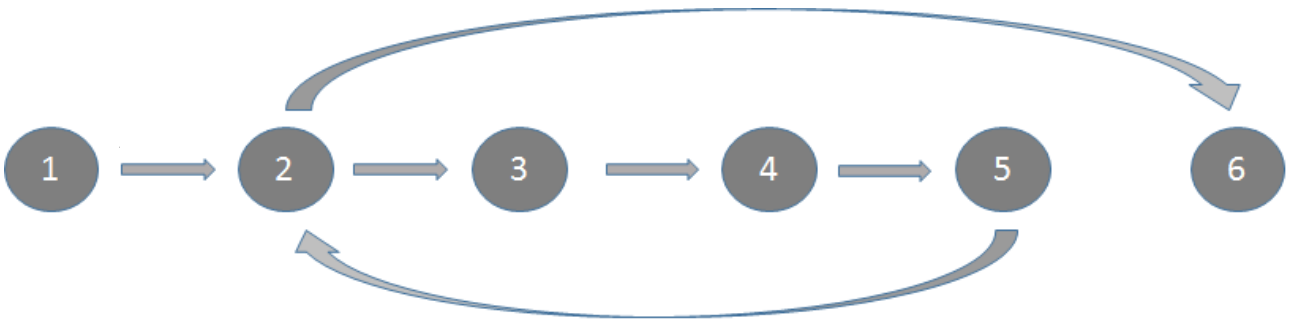

Ilustración 21 Representación del grafo de flujo de camino básico

Complejidad ciclomática:

Existen varias formas de calcular la complejidad ciclomática de un programa a partir de un grafo de flujo. A continuación, se evidencias dichas formas:

1. V (G) = A - N + 2 A: número de aristas
   N: el número de nodos
   V (G)=6 – 6 + 2= 2
2. V (G) = P + 1 P: número de nodos predicados
   V (G)= 1 + 1 = 2
3. V (G)= R R: número de regiones del grafo
   V (G)= 2

Luego de calculada la complejidad ciclomática mediante las tres fórmulas se evidencia que la complejidad es 2, lo cual indica que existen 2 caminos independientes, representando este valor el mínimo número de casos de pruebas.

Los caminos independientes resultantes son:

Camino 1: 1-2-3-4-5-2-6

Camino 2: 1-2-6

Se ejecutan los casos de prueba para cada camino independiente determinado en el grafo de flujo:

CASO DE PRUEBA CAMINO 1	
Condición de ejecución	Se recorre el arreglo hasta que llegue al final.
Entrada	Un arreglo de tipo json.

Resultados esperados	Se retorna un arreglo de encuestas dependiendo de la cantidad de objetos json que se pasen en la entrada.

Tabla 10 Caso de prueba camino 1

CASO DE PRUEBA CAMINO 2	
Condición de ejecución	Se recorre el arreglo hasta que llegue al final.
Entrada	Un arreglo de tipo json.
Resultados esperados	Se retorna un arreglo de encuestas vacío.

Tabla 11 Caso de prueba camino 2

3.1.2 Pruebas de Caja Negra

Las pruebas de caja negra, también denominada prueba de comportamiento, se centran en los requisitos funcionales del software. Estas permiten al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa (Pressman, 2002). Hacen referencia a pruebas que se llevan a cabo sobre la interfaz del software sin tener en cuenta el código de la aplicación. Los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta. Según (Pressman, 2010) estas pruebas tienen como propósito detectar:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

Descripción de los casos de prueba

Cada historia de usuario está asociada a una prueba funcional, las mismas se realizan en esta etapa del proyecto y en ellas se describen las posibles formas de utilización del software. Las pruebas funcionales no solo validan la transformación de una entrada en una salida, sino que validan una característica completa. En estos documentos de prueba se indican las posibles respuestas que tiene el software en la utilización de cada funcionalidad, así como los posibles mensajes de error, información o de aceptación que emite el software cuando se utiliza dicha funcionalidad (Larman, 1999). Para la realización de las pruebas se realizaron un total de 8 diseños de casos de pruebas (DCP).

Se entregaron los DCP y el software para ser evaluados por Guillermo Manuel Negrín Ortiz. A continuación, se muestra el comportamiento de las no conformidades (NC) según las iteraciones de pruebas. En la primera iteración se detectaron un total de 5 no conformidades, en la segunda iteración se resolvieron las anteriores, encontrándose 3 nuevas no conformidades, de las, quedando resultas todas en la tercera iteración.

Iteración	No Conformidad	Tipo
1	En el caso de prueba se especifica que al descargar la encuesta se muestra el mensaje “Encuesta descargada” y no se corresponde con la aplicación, que muestra el mensaje “Encuesta descargada con éxito”	Correspondencia
	El botón “Descargar” no cumple con la validación especificada en el caso de prueba.	Validación
	Cuando la encuesta se envía debería mostrar un mensaje que indique que la encuesta ha sido enviada con éxito.	Funcionalidad
	El botón “Ver encuestas” debería llamarse “Encuestas disponibles”	Recomendación
	Cambiar los colores de las interfaces por colores más agradables a la vista.	
2	Cuando hay problema con la conexión debería mostrar un mensaje indicándolo.	Recomendación
	Cuando se elimina una encuesta existente en el dispositivo debería mostrar el mensaje “Encuesta eliminada con éxito”	
	El botón de abrir la encuesta no muestra un icono q se corresponde con la acción.	Diseño
3	Fueron resueltas todas las NC	

Tabla 12 No conformidades detectadas

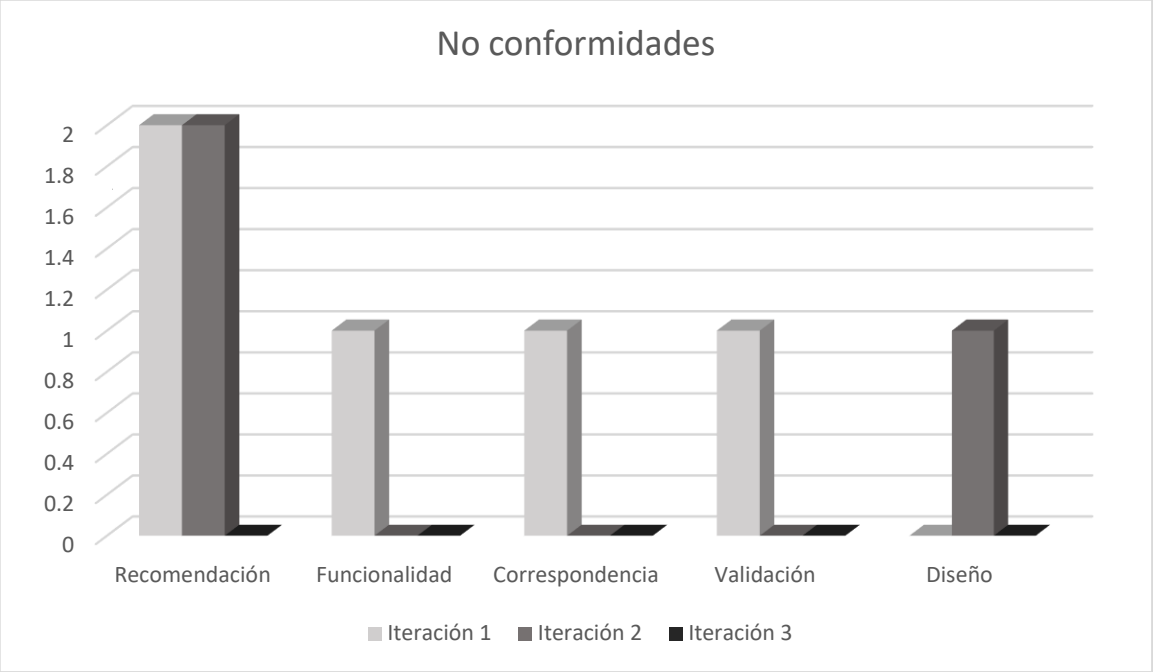

Ilustración 22 No conformidades

3.1.3 Métricas para diseño

Estas métricas cuantifican los atributos del diseño de manera que permiten evaluar su calidad. Dentro de las métricas para el diseño se encuentran las especializadas en diseño orientado a objetos, las cuales miden características de clases. Permiten averiguar cuan bien están definidas las clases y el sistema. Miden de forma cuantitativa la calidad de los atributos internos del producto como son la responsabilidad de las clases, la reutilización, la complejidad de implementación y mantenimiento (Pressman, 2010).

Se seleccionó la métrica orientada a clases Tamaño Operacional de Clase (TOC). Para evaluar las métricas son necesarios los valores de los umbrales para los parámetros de calidad. Algunos especialistas plantean umbrales basándose en el promedio de operaciones por clases obtenidos, estos valores fueron los aplicados en el diseño de la herramienta (Kidd, 1994).

Tamaño Operacional de Clase (TOC)

El tamaño operacional de una clase está dado por el número de métodos asignados a una clase. Para ello mide los siguientes atributos de calidad (Kidd, 1994):

- **Responsabilidad:** Responsabilidad asignada a una clase. Un aumento del TOC implica un aumento de esta responsabilidad, teniendo así demasiada responsabilidad, lo cual reduce la posibilidad de reutilización de la clase, lo que hace complicada la implementación y la prueba.

- **Complejidad de implementación:** Grado de dificultad que tiene implementar un diseño de clases determinado. Un aumento del TOC implica un aumento de la complejidad de implementación de una determinada clase.
- **Reutilización:** Grado de reutilización presente en una clase o estructura de clase. Un aumento del TOC implica una disminución del grado de reutilización de una determinada clase.

Luego de realizar la prueba, se obtuvieron los siguientes resultados:

- En la métrica responsabilidad los resultados revelaron un porciento de baja responsabilidad de 72%, de media responsabilidad de 12% y de alta responsabilidad de 16%.

Responsabilidad	Cantidad de clases	Promedio
Baja	23	44,23076923
Media	4	7,692307692
Alta	5	9,615384615

Tabla 13 Métrica Responsabilidad

A continuación, se muestra el gráfico de pastel:

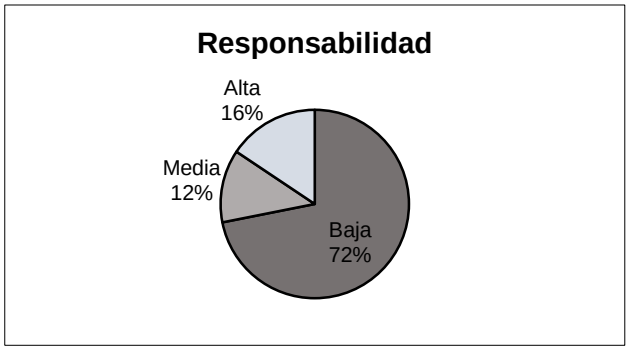

Ilustración 23 Métrica Responsabilidad

- En la métrica complejidad los resultados revelaron un porciento de baja complejidad de 75%, de media complejidad de 17% y de alta complejidad de 8%.

Complejidad	Cantidad de clases	Promedio
Baja	23	44,23076923
Media	4	7,692307692
Alta	5	9,615384615

Tabla 14 Métrica Complejidad

A continuación, se muestra el gráfico de pastel:

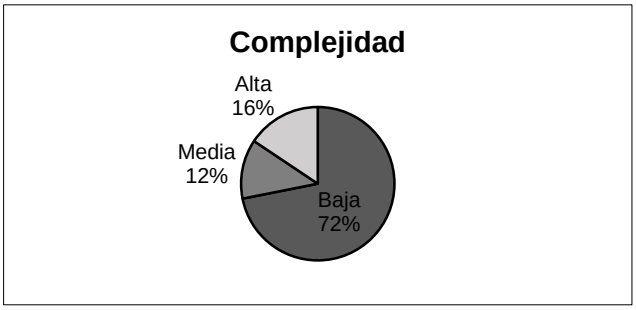

Ilustración 24 Métrica Complejidad

- En la métrica reutilización los resultados revelaron un porciento de baja reutilización de 75%, de media reutilización de 17% y de alta reutilización de 8%.

Reutilización	Cantidad de clases	Promedio
Alta	23	44,23076923
Media	4	7,692307692
Baja	5	9,615384615

Tabla 15 Métrica Reutilización

A continuación, se muestra el gráfico de pastel:

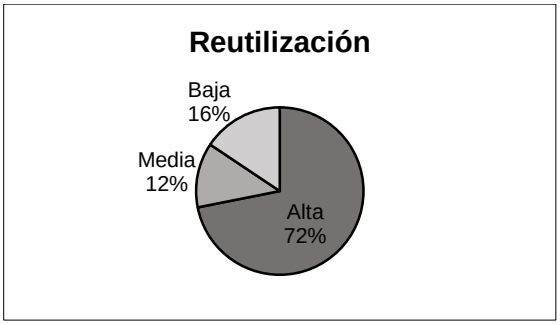

Ilustración 25 Métrica Reutilización

Luego de aplicadas las métricas se verificó que el diseño propuesto provee un nivel bajo de responsabilidad. Esto trae consigo un porciento bajo de complejidad de implementación, lo cual significa un menor grado de dificultad en el desarrollo de la herramienta. Con los resultados anteriores queda validado el diseño de la herramienta a desarrollar, atendiendo a los parámetros definidos.

3.1.4 Resultados de experimento con la aplicación

Con la finalidad de probar la aplicación desarrollada se le aplicó una encuesta a diez personas, de ellas uno era el cliente. Para ello se le pidió a los encuestados que instalaran la aplicación en sus dispositivos móviles. Fueron aplicadas un total de 6 preguntas, cada una con sus opciones. La prueba resultó satisfactoria, demostrando que la aplicación obtenida cumple con todos los requerimientos planteados en la fase de planeación. A continuación se muestran ejemplos de las intefaces de la aplicación al aplicar dicha encuesta:

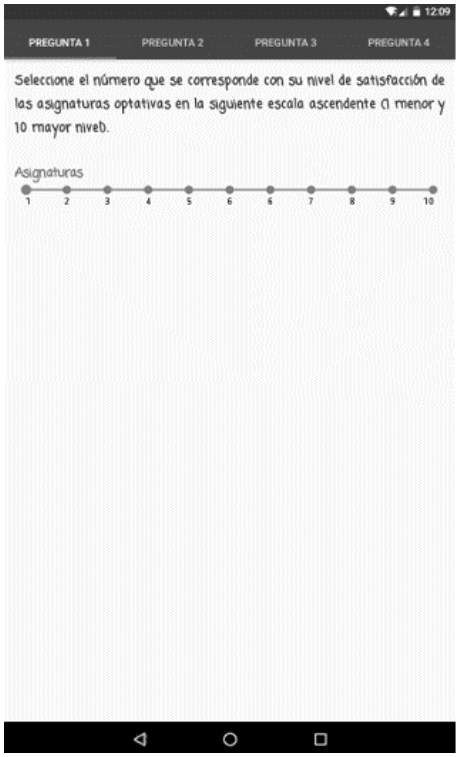

Ilustración 26 Pregunta 1

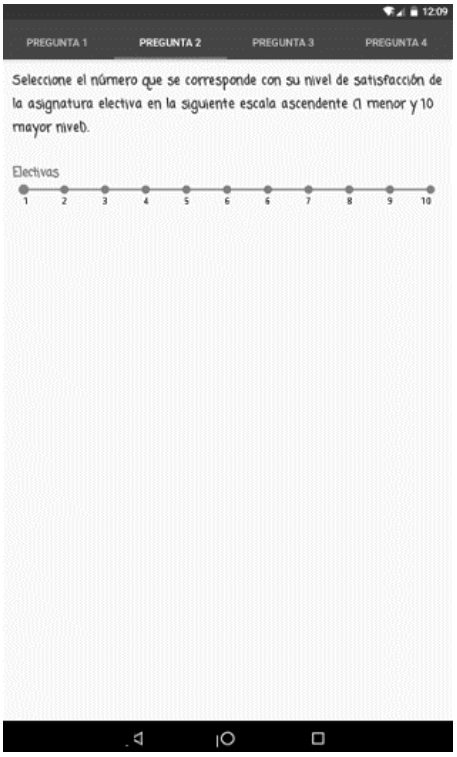

Ilustración 27 Pregunta 2

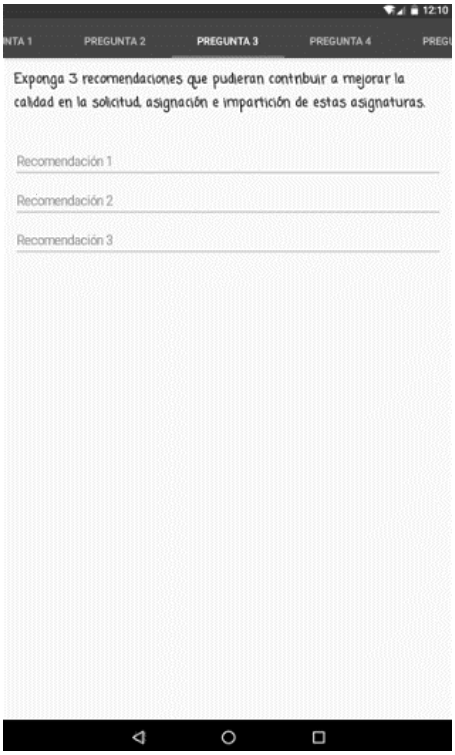

Ilustración 28 Pregunta 3

Ilustración 29 Pregunta 4

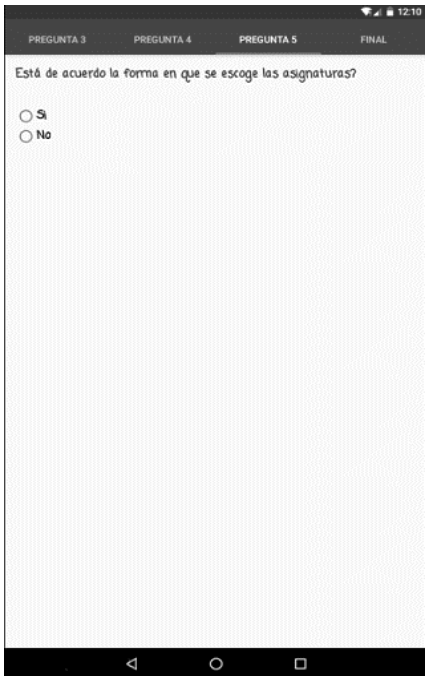

Ilustración 30 Pregunta 5

3.1.5 Aplicación Web

Para la realización de las pruebas se hizo necesario la creación de una aplicación web, que permitiera crear y descargar las encuestas creadas hacia los dispositivos.

Dicha Web de cumplir con los siguientes requisitos funcionales:

RF1: Crear encuesta

RF2: Eliminar encuesta

RF3: Autenticar usuario

RF4: Descargar apk

RF5: Crear pregunta

RF6: Eliminar pregunta

RF7: Crear opción

El funcionamiento de la Web se describe a continuación:

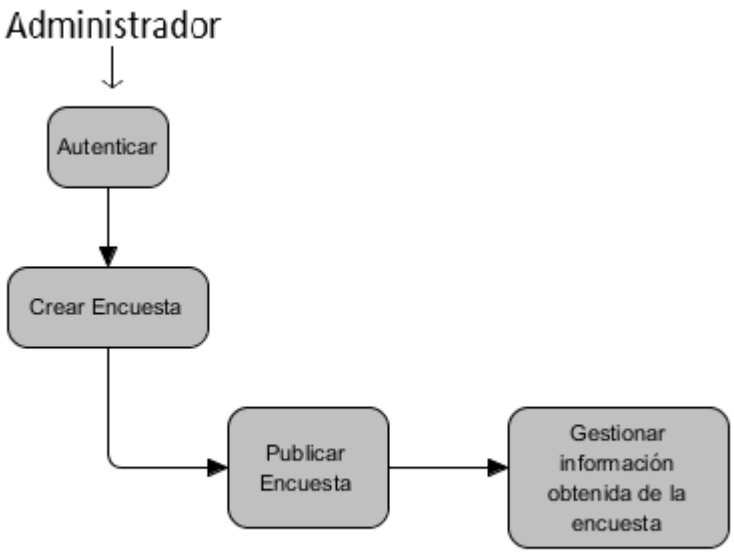

Ilustración 31 Funcionamiento de la Web

Autenticar: El administrador o persona encargada de crear las encuestas al entrar al sistema debe autenticarse.

Crear encuesta: Luego de autenticarse procederá a crear la encuesta a realizar.

Publicar encuesta: Una vez creada la encuesta debe guardarla para que los usuarios puedan responderla.

Gestionar información obtenida de la encuesta: Podrá gestionar todas las respuestas obtenidas.

Para la codificación de este sistema se hizo necesario la utilización de lenguajes y herramientas, los mismos se describen a continuación:

Lenguajes de programación

HTML 5

Es el lenguaje utilizado para la creación de páginas web, significa “HyperText Mark-Up Language” (Lenguaje para el Formato de Documentos de Hipertexto). Es un estándar reconocido en todo el mundo y cuyas normas define un organismo llamado World Wide Web Consortium, más conocido como W3C. Como se trata de un estándar reconocido, una misma página HTML se visualiza de forma similar en cualquier navegador de cualquier sistema operativo. El propio W3C define el lenguaje HTML como "un lenguaje reconocido universalmente y que permite publicar información de forma global" (Arjona, 2013).

CSS 3

CSS (hojas de estilo en cascada), viene del inglés Cascading Style Sheets, del que toma sus siglas es un lenguaje creado para definir la presentación de los documentos electrónicos definidos con HTML o XHTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para la creación de páginas web complejas. Se utiliza para definir el aspecto de todos los contenidos, es decir, el color, tamaño y tipo de letra de los párrafos de texto, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista (Pérez, 2013).

JavaScript

Es un lenguaje de programación que se utiliza principalmente para añadirle dinamismo a las páginas web. Una página web dinámica es aquella que incorpora efectos como aparición y desaparición de texto, animaciones, acciones que se activan al pulsar botones u otros elementos y ventanas con mensajes de aviso al usuario. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos, y se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios (Pérez, 2013).

PHP v7.1

PHP es el acrónimo de Hipertext Preprocesor. Es un lenguaje para programar scripts del lado del servidor, que se incrustan dentro del código HTML. Este lenguaje es gratuito y multiplataforma. PHP

se escribe dentro del código HTML, lo que lo hace realmente fácil de utilizar. Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado. La librería de funciones cubre desde cálculos matemáticos complejos hasta tratamiento de conexiones de red, por poner dos ejemplos (Alvarez, 2001).

Herramientas utilizadas

Visual Studio Code v1.23

Visual Studio Code es un editor de código, el cual soporta una cantidad considerable de lenguajes, ya sean propios de Microsoft como C#, F# y Visual Basic, o de otros como PHP, Phyton, Perl, SQL, shell scripting en Bash y Java, siendo este último el gran rival de .NET. También soporta Git y programación web con HTML, CSS y JavaScript, entre otros lenguajes (Muylinux, 2015).

Laravel v5.4

Laravel es uno de los frameworks de código abierto más fáciles de asimilar para PHP. Es simple, muy potente y tiene una interfaz elegante y divertida de usar. Fue creado en 2011 y tiene una gran influencia de frameworks. El objetivo de Laravel es el de ser un framework que permita el uso de una sintaxis refinada y expresiva para crear código de forma sencilla. Aprovecha todo lo bueno de otros frameworks y utiliza las características de las últimas versiones de PHP. La mayor parte de su estructura está formada por dependencias, especialmente de Symfony, lo que implica que el desarrollo de Laravel dependa también del desarrollo de sus dependencias (Arsys, 2018).

A continuación, se muestra las interfaces de dicha aplicación:

Interfaces la web:

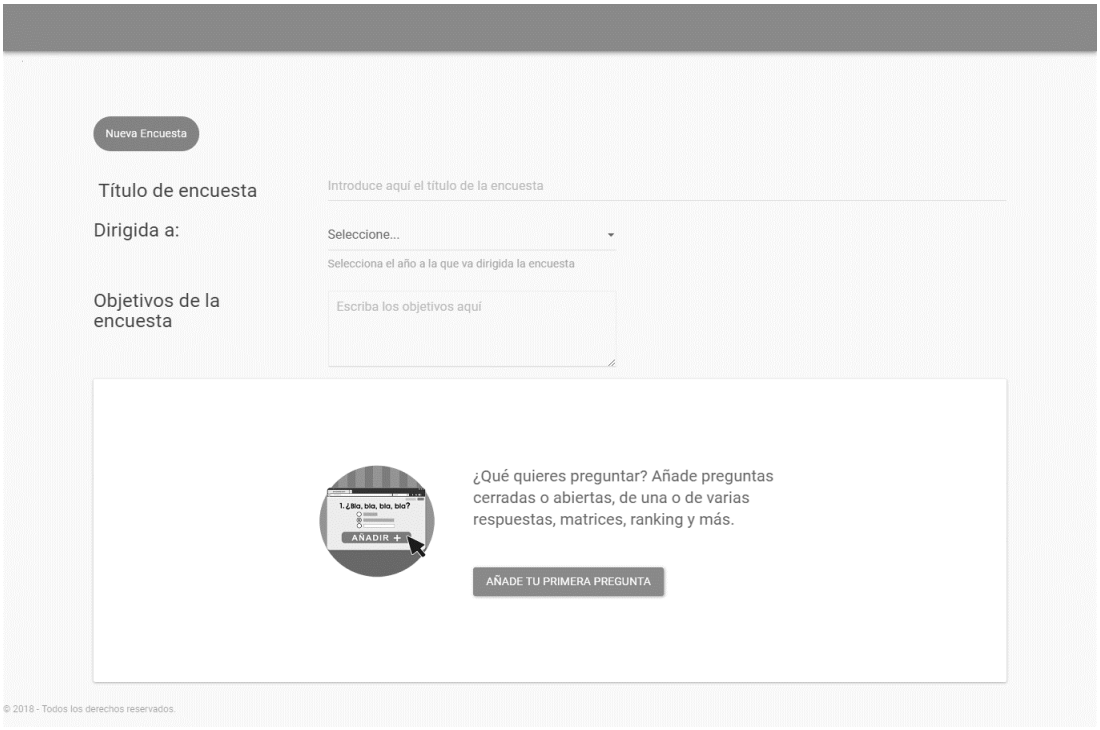

Ilustración 32 interfaz: Vista principal

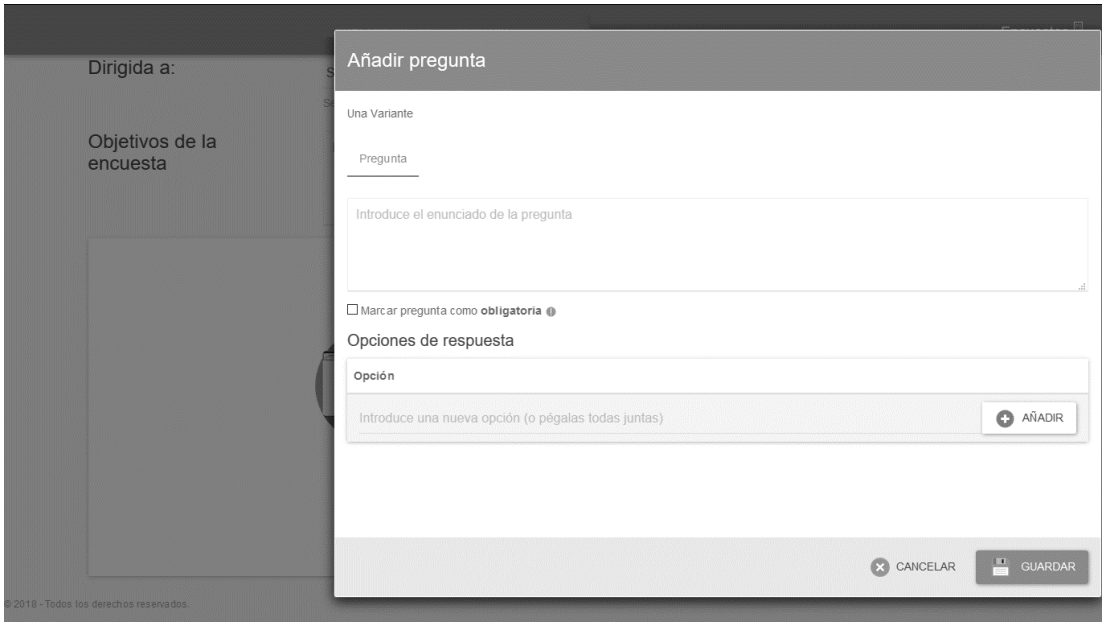

Ilustración 33 interfaz: Añadir pregunta

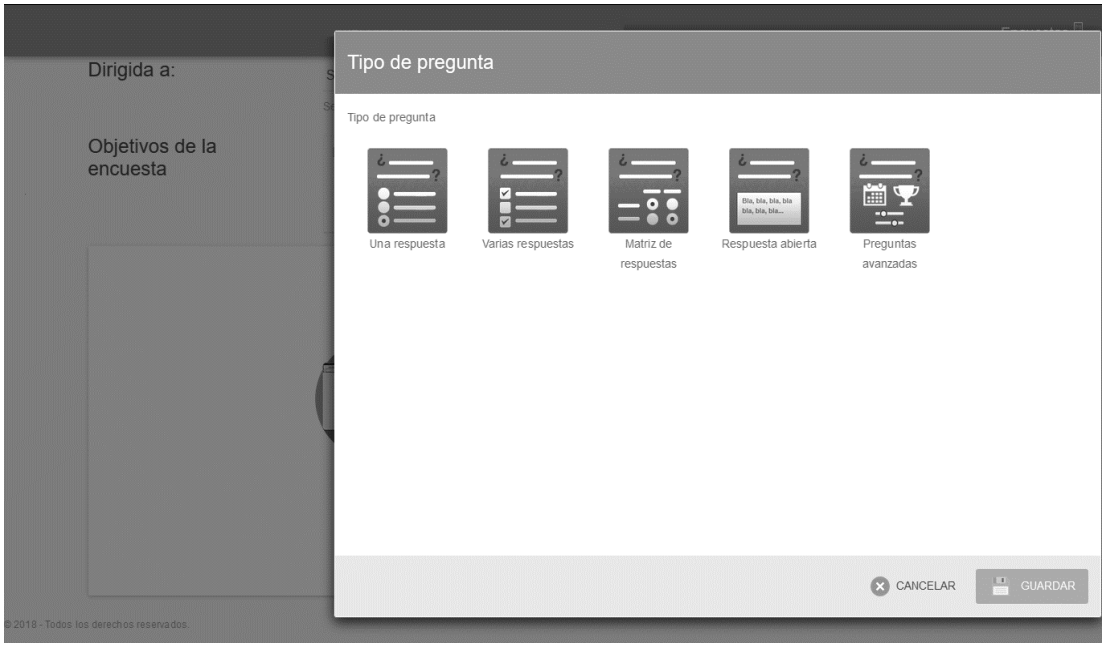

Ilustración 34 interfaz: Tipo de pregunta

Conclusiones

- La prueba de caja blanca realizada al sistema posibilitó la detección de errores, obteniéndose resultados satisfactorios.
- Las pruebas de caja negra, permitieron comprobar el correcto funcionamiento de la herramienta luego de realizadas varias iteraciones.

Conclusiones generales

Finalizada la presente investigación se puede concluir que se desarrollaron todas las tareas a fin de cumplir los objetivos propuestos, resaltando que:

- La elaboración del marco teórico referencial de la investigación permitió adquirir un mayor conocimiento sobre el objeto de estudio de la investigación.
- Durante el análisis y diseño de la solución se obtuvieron los artefactos propuestos por la metodología de desarrollo de software seleccionada, lo que permitió facilitar la implementación de las funcionalidades definidas.
- Las pruebas realizadas permitieron garantizar un correcto funcionamiento de la herramienta y cumplimiento de las necesidades y requisitos del cliente, avalado mediante el certificado de aceptación.
- La aplicación obtenida contribuye a la resolución de encuestas en la universidad con o sin conexión.

Referencias

E. d. codificación, «Estándares de codificación. [En línea] [Citado el: 21 de febrero de 2018.] https://docs.google.com/document/d/1rbxDFM0zsbFDNRZeM2FoXfRDbYSiSt6tCdbYPA0qdzs/edit?hl=en_US#bookmark=id.6e6a203b40fe.. .

2010. LimeService. [En línea] 20 de noviembre de 2010. http://www.limeservice.com/en/pricing.

Notadiario Bello. 2011. Ventajas y Desventajas de un Android. [En línea] 2011. http://www.notadiario.com/sci-tech/ventajas-ydesventajas-de-un-android/..

Alegsa, Leandro. 2009. ALEGSA.com.ar. . [En línea] 2009. http://www.alegsa.com.ar/Dic/lenguajes%20de%20programacion.php.

Alvarez, Miguel Angel. 2001. desarrolloweb.com. [En línea] 2001. http://www.desarrolloweb.com/articulos/497.php..

Android Studio. 2018. Android Studio. [En línea] 2018. https://developer.android.com/studio/intro/index.html?hl=es-419.

App Annie. 2018. App Annie. [En línea] 2018. https://www.appannie.com/en/.

Applesfera. 2014. Applesfera. [En línea] 2014. https://www.applesfera.com/aplicaciones-os-x-1/prototyper-una-herramienta-de-prototipado-de-aplicaciones.

Areatecnologia. 2016. areatecnologia. [En línea] 2016. http://www.areatecnologia.com/informatica/sistemas-operativos-moviles.html.

Arjona, Guillermo Prado. 2013. Características del lenguaje HTML. [En línea] 6 de diciembre de 2013. https://belenus.unirioja.es/~guprado/pagweb/carachtml.html.

Arsys. 2018. Arsys. [En línea] 2018. https://www.arsys.es/blog/programacion/que-es-laravel/.

Avieto México SA de CV. 2018. CompraSoft. [En línea] 2018. https://comprasoft.com/jetbrains/webstorm.

Avilez, José A. 2005. Recolección de datos. [En línea] agosto de 2005.

—. 2005. Recolección de datos. [En línea] agosto de 2005. http://www.monografias.com/trabajos12/recoldat/recoldat.shtml#quees.

Cadenas, Victor Garibay. 2016. devacademy. [En línea] 2016. https://stories.devacademy.la/mi-primer-app-con-retrofit-y-android-ac61a8954a2c.

—. 2016. devacademy. [En línea] 2016. https://stories.devacademy.la/mi-primer-app-con-retrofit-y-android-ac61a8954a2c.

Casorla, Yudith Cuba. 2012. Solución informática para la gestión de encuestas del Sistema. La Habana : s.n., 2012.

Cockbun, A., Williams, L. 2000. *"The Costs and Benefits of Pair Programming". Humans and Technology Technical Report.* 2000.

codificación, Estándares de. Estándares de codificación. [En línea] [Citado el: 21 de 02 de 2018.] https://docs.google.com/document/d/1rbxDFM0zsbFDNRZeM2FoXfRDbYSiSt6tCdbYPA0qdzs/edit?hl=en_US#bookmark=id.6e6a203b40fe..

Departamento de Ciencias de la Computacion e I.A. 2014. *Especificacion de requerimientos.* Granada : s.n., 2014.

Develapps. 2016. Develapps. [En línea] 2016. http://www.develapps.com/es/noticias/modelo-vista-presentador-mvp-en-android.

Eduteka. 2011. Tipos de pregunta. [En línea] 5 de diciembre de 2011.

Escalona, Maria Jose y Koch, Nora. 2002. *Ingeniería de Requisitos en Aplicaciones para la Web. Un estudio comparativo.* España : s.n., 2002.

Escobar, Yanvary. 2009. Monografías.com. [En línea] 2009. http://www.monografias.com/trabajos39/desarrollo-del-software/desarrollo-del-software.shtml.

Figueroa, Roberth, Solis, Camilo y Cabrera , Armando. 2007. Metodologías tradicionales y metodologías ágiles. 2007.

Firtman, Lic. Maximiliano. 2013. Video2Brain. Curso Básico Android. [En línea] 2013. http://www.video2brain.com..

Fiter, E. L. 2012. *Descripción, comparación y ejemplos de uso de las funciones de la toolbox de procesado digital de imágenes.* Madrid : s.n., 2012.

Flores, Lic. Ervin. 2006. *Metodología agiles Proceso Unificado Ágil (AUP). .* 2006.

Fowler, M. 2001.. "Continuous Integration". [En línea] 2001. www.martinfowler.com/articles/designDead.html.

Fowler, M., Beck, K., Brant, J. 1999. *"Refactoring: Improving the Design of Existing Code" .* s.l. : Addison-Wesley, 1999.

Fundación Universitaria de Las Palmas. 2013. *Cursos de Orientación Profesional: Creación de aplicaciones móviles en Android. .* 2013.

Fundéu BBVA. 2018. Fundación del Español Urgente. [En línea] 2018. [Citado el: 20 de marzo de 2018.] https://www.fundeu.es/recomendacion/aplicacion-alternativa-a-app/.

Gomez, Elena. 2007. Tiposde.eu. [En línea] 2007. https://tiposde.eu/tipos-de-software/.

González, Felipe Luis Martinez. 2010. Aplicaciones para dispositivos móviles. s.l., España : Universidad Politécnica de Valencia, 2010.

Guerrero, Gaby Lorena y Erazo, leydi Rocio. 2011. *Desarrollo de software.Definicion general del proceso.* UNIVERSIDAD DEL CAUCA. Popayan- Cauca : s.n., 2011.

IBM. 2018. IBM. [En línea] 2018. https://www.ibm.com/support/knowledgecenter/es/SSZLC2_8.0.0/com.ibm.commerce.developer.doc/concepts/csdmvcdespat.htm.

—. 2014. IBM. [En línea] 2014. https://www.ibm.com/support/knowledgecenter/es/SSMKHH_9.0.0/com.ibm.etools.mft.doc/ac55710_.html.

Joskowicz, Jose. 2012. *Reglas y prácticas en eXtreme Programing.* 2012.

Kidd, Mark Lorenz. 1994. *Object-Oriented Software Metrics. .* New Jersey : s.n., 1994.

Larman, C. 1999. *UML y patrones. Introducción al análisis y diseño orientado a objetos.* México : Prentice Hall Hispanoamericana, 1999.

Larman, Craig. 1999. *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* Mexico : Prentice Hall, 1999. págs. 189-209.

Larramendi, Claudia Beatriz. 2012. UCI. 2012.

Letelier, P. 2003. *Proyecto Docente e Investigador.* s.l. : DSIC, 2003.

Letelier, Patricio y Penades, Jose H. 2003. *Metodologías ágiles en el desarrollo de software: Extreme Programming (XP).* 2003.

LimeSurvey. 2011. LimeSurvey. [En línea] 5 de diciembre de 2011. http://www.limesurvey.org/.

Linux Fundation. 2018. Node. [En línea] 2018. https://nodejs.org/es/.

Loogic. 2009. Loogic. [En línea] 2009. https://loogic.com/justinmind-prototyper-25-prototipos-web-made-in-spain/.

Martin, R. 2002. "Continuos Care vs. Initial Design". [En línea] 2002. www.objectmentor.com/resources/articles/Continuous_Care.pdf .

McKenzie, Jamie. 2012. Definición ABC. Definición de Pregunta. [En línea] 2012. http://www.definicionabc.com/general/pregunta.php.

Moreapp. 2018. Moreapp. [En línea] 2018. https://www.moreapp.com/es/blog/encuestas-app-offline/.

Mouse, JC. 2017. JC Mouse. [En línea] 5 de octubre de 2017. http://www.jc-mouse.net/ingenieria-de-sistemas/caja-blanca-prueba-del-camino-basico.

Muylinux. 2015. Muylinux. [En línea] 2015. https://www.muylinux.com/2015/04/30/visual-studio-code-editor-codigo-microsoft-windows-os-x-gnu-linux/.

NetMarketShare. 2017. NetMarketShare. [En línea] 2017.

Oracle Corporation. 2018. [En línea] 2018. https://www.java.com/es/download/faq/whatis_java.xml.

—. 2012. The world's most popular open source database. [En línea] 15 de mayo de 2012. www.mysql.com.

Perez, Adriana Almeira y Sandra. 2007. *Arquitectura de Software: Estilos y Patrones.* Argentina : s.n., 2007.

Pérez, Javier Eguiluz. 2013. LibrosWeb. [En línea] 2013. http://www.librosweb.es/css.

—. 2013. LibrosWeb. [En línea] 2013. http://www.librosweb.es/javascript.

Pressman, Roger S. 2010. *Ingeniería del Software. Un Enfoque Práctico.* 2010.

Pressman, Roger S. 2002. *Ingeniería de Software, un enfoque práctico.* 5ta. s.l. : McGraw-Hill, 2002.

Questionpro. 2018. Questionpro. [En línea] 2018. https://www.questionpro.com/es/mobile/.

Quicktapsurvey. 2018. Quicktapsurvey. [En línea] 2018. https://www.quicktapsurvey.com/es/.

Rizo, Jorge Ernesto Castillo. 2014. *Curso UMI .* 2014.

Rouse, Margaret. 2005. *Desarrollo de aplicaciones móviles.* 2005.

Rumbaugh, James, Jacobson, Ivar y Booch, Grady. 2000. El Lenguaje Unificado de Modelado.Manual de referencia. Madrid : Pearson Educación, 2000.

Rupp., Klaus Pohl and Chris. 2015. *Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam-Foundation Level – IREB compliant.* 2015. volume 53.

Sacsa. 2011. Sacsa. [En línea] 2011. http://www.software-sacsa.com/id150.htm.

Salo, Ronkainen, Warsta, J. y Abrahamsson, J. 2002. *"Agile software development methods Review and analysis".* s.l. : VTT Publications, 2002.

Sanchez, Maria A Mendoza. 2004. *Metodologías del desarrollo de software.* 2004.

Schwaber, K. 2010. *Advanced Development Methods. SCRUM Development Process Retrieved. .* 2010.

Shankland, S. 2008. [En línea] 22 de Septiembre de 2008. .Retrieved from http://news.cnet.com/8301-17938_105-10047551-1.html.

Sommerville, I. 2002. *Ingeniería de Software.* s.l. : Pearson Educación, 2002.

Sqlite. 2014. sqlite. [En línea] 2014. http://sqlitedatabasegrupo.blogspot.com/p/caracteristicas-de-sqlite.html.

SurveyMonkey. 2012. SurveyMonkey. [En línea] 2012. http://es.surveymonkey.com/.

Taringa. 2011. Taringa. [En línea] 2011. https://www.taringa.net/posts/info/8589949/Sistemas-operativos-moviles-Comparacion.html.

The Number One HTTP Server On The Internet . [En línea] [Citado el: 2012 de mayo de 15.] http://httpd.apache.org/ .

UCI. 2012. Entorno Virtual de Aprendizaje. [En línea] 12 de mayo de 2012. http://eva.uci.cu/mod/resource/view.php?id=8500&subdir=/UML_y_Patrones.

Universidad de Champagnat . 2013. gestiopolis. [En línea] 2013. https://www.gestiopolis.com/encuesta-tipos-y-procedimiento-de-uso-en-investigacion-de-mercados/.

Universidad Politécnica de Madrid. 2016. *Metodología de desarrollo ágil para sistemas móviles.* Eapaña : s.n., 2016.

Valero, Carmen Cantillo, Roura Redondo, Margarita y Sánchez Palacín, Ana . 2012. *Tendencias actuales en el uso de dispositivos móviles.* España : UNE, 2012.

Vílchez, Ángel. 2009. *Qué es Android: Características y Aplicaciones.* 2009.

Web Services Description Language (WSDL) . [En línea] [Citado el: 2012 de mayo de 15.] http://www.w3.org/TR/wsdl .

Wordpress. 2012. wordpress. [En línea] 2012. [Citado el: 13 de noviembre de 2012.] https://scoello12.wordpress.com/ventajas-y-desventajas/.

—. 2014. wordpress. [En línea] 2014. https://aprendiendotecnologiaadmonb.wordpress.com/dispositivos-moviles-y-sus-sistemas-operativos/.

Zalazar, Diana Fernández. 2015. Telarañas de conocimiento. Educando en tiempos de internet. 2015.

Anexos

Anexo 1: Historias de Usuarios

Historias de Usuarios Aplicación Android

Historia de Usuario	
Número: HU-1	**Nombre**: Mostrar encuestas disponibles
Usuario: Encuestado	**Iteración Asignada:** 1
Prioridad del negocio: Alta	**Puntos estimados:** 4 días
Riesgo en desarrollo: Alto	**Puntos reales:** 4 días
Descripción: Permite al usuario ver las encuestas que se encuentran creadas en el sistema. **Observaciones:** • Es necesario tener conexión.	

Tabla 16 HU Mostrar encuestas disponibles

Historia de Usuario	
Número: HU-3	**Nombre**: Mostrar encuestas descargadas
Usuario: Encuestado	**Iteración Asignada:** 1
Prioridad del negocio: Media	**Puntos estimados:** 4 días
Riesgo en desarrollo: Medio	**Puntos reales:** 4 días

Descripción: Permite al usuario ver las encuestas que ha descargado hacia su dispositivo móvil. **Observaciones:** • Se puede realizar con o sin conexión.

Tabla 17 HU Mostrar encuestas descargadas

Historia de Usuario	
Número: HU-4	**Nombre**: Seleccionar encuesta a responder
Usuario: Encuestado	**Iteración Asignada:** 1
Prioridad del negocio: Alta	**Puntos estimados:** 2 días
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: Permite al usuario seleccionar la encuesta que desea responder. **Observaciones:** • Se puede realizar con o sin conexión.	

Tabla 18 HU Seleccionar encuesta a responder

Historia de Usuario	
Número: HU-6	**Nombre historia de usuario**: Responder encuesta
Usuario: Encuestado	**Iteración Asignada:** 2
Prioridad del negocio: Alta	**Puntos estimados:** 7 días
Riesgo en desarrollo: Alto	**Puntos reales:** 7 días

Descripción: Permite al usuario responder la encuesta. **Observaciones:** • La encuesta se puede responder tanto de manera online como offline.

Tabla 19 HU Responder encuesta

Historia de Usuario	
Número: HU-8	**Nombre**: Eliminar encuesta descargada
Usuario: Encuestado	**Iteración Asignada:** 3
Prioridad del negocio: Alta	**Puntos estimados:** 3 días
Riesgo en desarrollo: Alto	**Puntos reales:** 3 días
Descripción: Permite al usuario seleccionar la encuesta que desea eliminar de su dispositivo. **Observaciones:** • Se puede realizar con o sin conexión.	

Tabla 20 HU Eliminar encuesta descargada

Historia de Usuario	
Número: HU-9	**Nombre**: Guardar respuesta
Usuario: Encuestado	**Iteración Asignada:** 1
Prioridad del negocio: Baja	**Puntos estimados:** 3 días
Riesgo en desarrollo: Bajo	**Puntos reales:** 3 días

Descripción: Permite al usuario guardar las respuestas de la encuesta en su dispositivo. **Observaciones:** • Se puede realizar con o sin conexión.

Tabla 21 HU Guardar respuesta

Historias de Usuarios Aplicación Web

Historia de Usuario	
Número: HU-01	**Nombre historia de usuario**: Crear encuesta
Usuario: Persona encargada	**Iteración Asignada:** 3
Prioridad del negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semana
Descripción: Permite a la persona encargada de crear las encuestas observar la interfaz visual que presenta el mismo. **Observaciones:** • Para crear una encuesta esta entidad tiene tres campos, el título (no nulo), el objetivo y a quién va dirigida la encuesta.	

Tabla 22 HU Crear encuesta

Historia de Usuario	
Número: HU-04	**Nombre historia de usuario**: Eliminar encuesta
Usuario: Persona encargada	**Iteración Asignada:** 3
Prioridad del negocio: Baja	**Puntos estimados:** 2 días
Riesgo en desarrollo: Bajo	**Puntos reales:** 2 días

Descripción: Permite a la persona encargada eliminar la encuesta que ha creado. **Observaciones:** • Para eliminar solo se necesita el identificador (no nulo) de la encuesta.

Tabla 23 HU Eliminar encuesta

Historia de Usuario	
Número: HU-05	**Nombre historia de usuario**: Crear pregunta
Usuario: Persona encargada	**Iteración Asignada:** 3
Prioridad del negocio: Alta	**Puntos estimados:** 2 días
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: Permite a la persona encargada crear las preguntas que tendrán la encuesta a responder. **Observaciones:** • Para crear una pregunta se tiene que llenar el enunciado, la cantidad de opciones que tendrá, así como el texto de las opciones.	

Tabla 24 HU Crear pregunta

Historia de Usuario	
Número: HU-08	**Nombre historia de usuario**: Eliminar pregunta
Usuario: Persona encargada	**Iteración Asignada:** 3
Prioridad del negocio: Baja	**Puntos estimados:** 2 días
Riesgo en desarrollo: Bajo	**Puntos reales:** 2 días

Descripción: Permite a la persona encargada eliminar la pregunta que ha creado. **Observaciones:** • Para eliminar solo se necesita el identificador de la pregunta.

Tabla 25 HU Eliminar pregunta

Historia de Usuario	
Número: HU-10	**Nombre historia de usuario**: Crear opción
Usuario: Persona encargada	**Iteración Asignada:** 3
Prioridad del negocio: Alta	**Puntos estimados:** 2 días
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: Permite a la persona encargada crear las opciones que tendrán cada una de las preguntas de la encuesta a responder. **Observaciones:** • Para crear las opciones de debe seleccionar la cantidad de opciones que tendrá, así como el texto de las mismas.	

Tabla 26 HU Crear opción

Historia de Usuario	
Número: HU-13	**Nombre historia de usuario**: Eliminar opción
Usuario: Persona encargada	**Iteración Asignada:** 3
Prioridad del negocio: Baja	**Puntos estimados:** 2 días
Riesgo en desarrollo: Baja	**Puntos reales:** 2 días
Descripción: Permite a la persona encargada eliminar la opción que desee.	

Observaciones: • Para eliminar solo se necesita el identificador de la opción.

Tabla 27 HU Eliminar opción

Historia de Usuario	
Número: HU-13	**Nombre historia de usuario**: Descargar apk
Usuario: Persona encargada	**Iteración Asignada:** 3
Prioridad del negocio: Baja	**Puntos estimados:** 2 días
Riesgo en desarrollo: Baja	**Puntos reales:** 2 días
Descripción: Permite a los usuarios descargar la aplicación Android para poder instalarla en sus dispositivos. **Observaciones:** • Es necesario estar conectado a la red.	

Tabla 28 HU Descargar apk

Anexo 2: Tarjetas Clase-Responsabilidad-Colaborador

Tarjetas Clase-Responsabilidad-Colaborador Aplicación Android

Tarjeta CRC	
Clase: ConstructorEncuesta	
Responsabilidades: obtenerDatos obtenerEncuestasEnviar insertarEncuestas EliminarEncuesta verificarRespondida encuestaRespondida	**Colaboraciones:** ContentValues Context EditText EncuestaDBhelper Encuesta

Tabla 29 ConstructorEncuesta

Tarjeta CRC	
Clase: EncuestaDescargadasActivity	
Responsabilidades: onCreate generarLinearLayoutVertical crearAdaptador inicializarAdaptadorRV accionEliminar	**Colaboraciones:** Context CoordinatorLayout Snackbar AppCompatActivity LinearLayoutManager RecyclerView

Tabla 30 EncuestaDescargadasActivity

Tarjeta CRC	
Clase: EncuestaDescargadasAdapter	
Responsabilidades: onCreateViewHolder onBindViewHolder onClick getItemCount	**Colaboraciones:** Activity RecyclerView LayoutInflater View ViewGroup EncuestaDescargadaViewHolder EncuestaDescargadasActivity

Tabla 31 EncuestaDescargadasAdapter

Tarjeta CRC	
Clase: EncuestaEnviar	
Responsabilidades: onCreate generarLinearLayoutVertical crearAdaptador inicializarAdaptadorRV accionEliminar	**Colaboraciones:** IListadoEncuestaEnviarPresentador LinearLayoutManager EncuestaEnviarAdapter ConstructorEncuesta LinearLayoutManager RecyclerView

Tabla 32 EncuestaEnviar

Tarjetas Clase-Responsabilidad-Colaborador Aplicación Web

Tarjeta CRC	
Clase: EncuestaController	
Responsabilidades: Index create store destroy	**Colaboraciones:** Encuesta Pregunta Opcion EncuestaFormRequest

Tabla 33 Tarjeta CRC EncuestaController

Tarjeta CRC

Clase: PreguntaController	
Responsabilidades: index unaVarianteCrear variasVariantesCrear matriz1RespXfilaCrear respuestaAbierta1LineaCrear respuestaAbiertaVariasLineaCrear TextoExplicativoCrear RangoNumericoCrear create store	**Colaboraciones:** Encuesta Pregunta PreguntaFormRequest Opcion Request

Tabla 34 Tarjeta CRC PreguntaController

Tarjeta CRC	
Clase: UsuarioController	
Responsabilidades: Index create store destroy update show edit	**Colaboraciones:**

Tabla 35 Tarjeta CRC UsuarioController

Tarjeta CRC	
Clase: EncuestaFormRequest	
Responsabilidades: authorize rules	**Colaboraciones:**

Tabla 36 Tarjeta CRC EncuestaFormRequest

Tarjeta CRC	
Clase: PreguntaFormRequest	
Responsabilidades: authorize rules	**Colaboraciones:**

Tabla 37 Tarjeta CRC PreguntaFormRequest

Tarjeta CRC	
Clase: Encuesta	
Responsabilidades: pregunta	**Colaboraciones:**

Tabla 38 Tarjeta CRC Encuesta

Tarjeta CRC

Clase: Opción	
Responsabilidades: pregunta	**Colaboraciones:** Pregunta

Tabla 39 Tarjeta CRC Opción

Tarjeta CRC	
Clase: Pregunta	
Responsabilidades: encuesta opcion respuesta	**Colaboraciones:** Opcion Encuesta Respuesta

Tabla 40 Tarjeta CRC Pregunta

Tarjeta CRC	
Clase: Respuesta	
Responsabilidades: pregunta	**Colaboraciones:** Pregunta

Tabla 41 Tarjeta CRC Respuesta

Anexo 3: Tareas de Ingeniería

Tareas de Ingeniería Aplicación Android

TAREA DE INGENIERÍA	
Número de la tarea:1	**HU:** Mostrar encuestas disponibles
Nombre de la tarea: Implementar funcionalidad Mostrar encuestas disponibles.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 4 días
Fecha de inicio:2/4/18	**Fecha de fin: 5/4/18**
Descripción: Implementar una funcionalidad que permita al usuario observar todas las encuestas disponibles.	

Tabla 42 Tarea de ingeniería perteneciente a la HU Mostrar encuestas disponibles

TAREA DE INGENIERÍA	
Número de la tarea:3	**HU:** Mostrar encuestas descargadas
Nombre de la tarea: Implementar funcionalidad Mostrar encuestas descargadas.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 4 días
Fecha de inicio:13/4/18	**Fecha de fin:17/4/18**
Descripción: Implementar una funcionalidad que permita al usuario observar todas las encuestas que ya ha descargado.	

TAREA DE INGENIERÍA	
Número de la tarea:4	**HU:** Seleccionar encuesta a responder
Nombre de la tarea: Implementar funcionalidad Seleccionar encuesta a responder.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2 días
Fecha de inicio:18/4/18	**Fecha de fin:20/4/18**

Descripción: Implementar una funcionalidad que permita al usuario seleccionar la encuesta que desee responder.

TAREA DE INGENIERÍA	
Número de la tarea:6	**HU:** Responder encuesta
Nombre de la tarea: Implementar funcionalidad Responder pregunta.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 7 días
Fecha de inicio:24/4/18	**Fecha de fin:1/5/18**
Descripción: Implementar una funcionalidad que permita al usuario responder una pregunta.	

TAREA DE INGENIERÍA	
Número de la tarea:9	**HU:** Guardar respuesta
Nombre de la tarea: Implementar funcionalidad Guardar respuesta.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 3 días
Fecha de inicio:8/5/18	**Fecha de fin:11/5/18**
Descripción: Implementar una funcionalidad que permita guardar temporalmente en la base de datos del dispositivo las respuestas que el usuario dio por cada respuesta.	

Tareas de Ingeniería Aplicación Web

TAREA DE INGENIERÍA	
Número de la tarea:10	**HU:** Crear encuesta
Nombre de la tarea: Implementar funcionalidad Crear encuesta.	

Tipo de tarea: Desarrollo	**Puntos estimados:** 7 días
Fecha de inicio: 11/5/18	**Fecha de fin: 18/5/18**
Descripción: Implementar una funcionalidad que permita a la persona encargada crear una encuesta en el sistema.	

Tabla 43 Tarea de Ingeniería Crear encuesta

TAREA DE INGENIERÍA	
Número de la tarea:11	**HU:** Eliminar encuesta
Nombre de la tarea: Implementar funcionalidad Eliminar encuesta.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2 días
Fecha de inicio: 18/5/18	**Fecha de fin: 20/5/18**
Descripción: Implementar una funcionalidad que permita a la persona encargada eliminar una encuesta creada.	

Tabla 44 Tarea de ingeniería perteneciente a la HU Eliminar encuesta

TAREA DE INGENIERÍA	
Número de la tarea:12	**HU:** Crear pregunta
Nombre de la tarea: Implementar funcionalidad Crear pregunta.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2 días
Fecha de inicio: 20/5/18	**Fecha de fin: 22/5/18**
Descripción: Implementar una funcionalidad que permita a la persona encargada crear una pregunta.	

Tabla 45 Tarea de ingeniería perteneciente a la HU Crear pregunta

TAREA DE INGENIERÍA	
Número de la tarea:13	**HU:** Eliminar pregunta
Nombre de la tarea: Implementar funcionalidad Eliminar pregunta.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2 días
Fecha de inicio: 22/5/18	**Fecha de fin: 24/5/18**

Descripción: Implementar una funcionalidad que permita a la persona encargada eliminar una pregunta de la encuesta creada.

Tabla 46 Tarea de ingeniería perteneciente a la HU Eliminar pregunta

TAREA DE INGENIERÍA	
Número de la tarea:14	**HU:** Crear opción
Nombre de la tarea: Implementar funcionalidad Crear opción.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2 días
Fecha de inicio: 24/5/18	**Fecha de fin: 26/5/18**
Descripción: Implementar una funcionalidad que permita a la persona encargada crear las opciones que tendrá cada pregunta.	

Tabla 47 Tarea de ingeniería perteneciente a la HU Crear opción

TAREA DE INGENIERÍA	
Número de la tarea:15	**HU:** Eliminar opción
Nombre de la tarea: Implementar funcionalidad Eliminar opción.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2 días
Fecha de inicio: 26/5/18	**Fecha de fin: 28/5/18**
Descripción: Implementar una funcionalidad que permita a la persona encargada eliminar la opción que desee en la pregunta.	

Tabla 48 Tarea de ingeniería perteneciente a la HU Eliminar opción