UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

La Universidad Católica de Loja

ÁREA TÉCNICA

TÍTULO DE INGENIERO EN INFORMÁTICA

Framework de seguridad en codificación de aplicaciones móviles híbridos y nativos.

TRABAJO DE TITULACIÓN

AUTORA: Ramos Vásquez, Consuelo del Pilar

DIRECTOR: Jaramillo Hurtado, Danilo Rubén, Msc.

CENTRO UNIVERSITARIO MADRID

2017

Septiembre, 2017

APROBACIÓN DEL DIRECTOR DEL TRABAJO DE TITULACIÓN

Máster.

Jaramillo Hurtado, Danilo Rubén.

DOCENTE DE LA TITULACIÓN

De mi consideración:

El presente trabajo de fin de titulación: “Framework de seguridad en codificación de aplicaciones móviles híbridos y nativos", realizado por Ramos Vásquez Consuelo del Pilar, ha sido orientado y revisado durante su ejecución, por cuanto se aprueba la presentación del mismo.

Loja, agosto de 2017

f)……

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

“Yo Ramos Vásquez Consuelo del Pilar declaro ser autora del presente trabajo de titulación: Framework de seguridad en codificación de aplicaciones móviles híbridos y nativos, de la Titulación de Ingeniería en Informática, siendo Jaramillo Hurtado, Danilo Rubén, M.S.c. director del presente trabajo; y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales. Además certifico que las ideas, conceptos, procedimientos y resultados vertidos en el presente trabajo investigativo, son de mi exclusiva responsabilidad.

Adicionalmente declaro conocer y aceptar la disposición del Art. 88 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: “Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado o trabajos de titulación que se realicen con el apoyo financiero, académico o institucional (operativo) de la Universidad”

Ramos Vásquez Consuelo del Pilar
C.I. 1803346236

DEDICATORIA

El presente trabajo de fin de grado está dedicado a Dios y a la Virgen de Lourdes, pues mi fe en ella ha sido fundamental para concluir mi formación.

A mi esposo, eterno optimista, que siempre ha creído en mí y quien está a mi lado, siendo un pilar fundamental en mi vida.

A mi padre, hermanas y cuñados, que con sus palabras de aliento hicieron que no desista.

A mis tías, amigos y compañeros, quienes me han apoyado cuando he necesitado.

Consuelo.

AGRADECIMIENTO

El primer agradecimiento quiero dárselo a Dios y a la Virgen de Lourdes por permitir que se cumplan mis metas.

También agradecer a la Universidad Técnica Particular de Loja y docentes, por facilitar la formación de los alumnos sin importar la situación geográfica en la que nos encontramos.

Un agradecimiento especial al Ingeniero Danilo Jaramillo, director de tesis, por su paciencia y orientación para concluir este trabajo.

Agradecer de manera especial a mi esposo por estar siempre presente, colaborando en todo momento, gracias por demostrarme tu amor en los momentos más difíciles que hemos compartido.

Agradecer a mi padre, hermanas, tías, que me han ayudado en todo lo posible, que han estado ahí siempre para brindarme todo su apoyo, palabras de aliento, y ánimo para concluir la carrera.

Hay dos personas a las que quiero agradecer de manera especial, porque sin la ayuda de ellos nunca hubiera sido posible avanzar y tener tiempo para salir adelante, los cuales estuvieron desde el inicio de esta etapa, a Miguel, mi cuñado, quien a pesar de la distancia y horarios me ha sabido orientar cuando más he necesitado y a Fernando, mi jefe, quien me ha apoyado de manera silenciosa, pero demostrándome que podía contar con él para todo cuanto necesitase.

Consuelo

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

La seguridad de las aplicaciones móviles en la actualidad es importante debido a que la mayoría de empresas han empezado a incluir como herramienta de trabajo los Smartphone, puesto que permiten realizar tareas en las cuales es necesario que los datos que almacenan o manipulan estén protegidos. El objetivo del presente trabajo es analizar las medidas de seguridad de los framework de desarrollo de aplicaciones móviles más relevantes, posteriormente agrupar las medidas de seguridad más utilizadas por las mismas para con ellas desarrollar un framework de seguridad siguiendo unos lineamientos. Una vez creado el framework de seguridad se implementara en la metodología Mobile-D para desarrollar una aplicación móvil que permita gestionar los pagos de facturas, consiguiendo de esta forma, una aplicación segura.

El resultado que se obtiene una vez concluido el trabajo es un framework con las medidas de seguridad que se puede implementar en cualquier metodología que se utilice para el desarrollo de una aplicación, consiguiendo obtener una aplicación confiable, que permita a las empresas gestionar y almacenar los datos de forma segura.

PALABRAS CLAVES: Android, Framework de seguridad, Mobile-D, Aplicación móvil.

ABSTRACT

The security of mobile applications today is important because most companies have begun to include SmartPhone as a tool, since they allow to perform tasks in which it is necessary that the data they store or manipulate are protected. The objective of the present work is to analyze the security measures of the most relevant mobile application development framework, later to group the security measures most used by them to develop a security framework following guidelines. Once the security framework is created, it will be implemented in the Mobile-D methodology to develop a mobile application that allows managing invoices payments, thus achieving a secure application.

The result obtained after the work is completed is a framework with security measures that can be implemented in any methodology that is used for the development of an application, obtaining a reliable application, which allows companies to manage and store the Data securely.

KEYWORDS: Android, Security Framework, Mobile-D, Mobile Application.

INTRODUCCIÓN

El presente trabajo de titulación trata sobre la creación de un Framework de Seguridad para el desarrollo de aplicaciones de dispositivos móviles de tipo híbrido, presentando de esta manera un marco para los desarrolladores, permitiendo crear un software que ofrezca seguridad al empresario, para que la información empresarial estratégica que se transmita a través de dispositivos móviles sea segura.

La característica principal del framework es presentar un modelo que permita desarrollar aplicaciones que protejan la información que almacena o maneja un móvil de posibles intromisiones de terceros que perjudiquen al usuario.

Para analizar esta problemática es necesario mencionar sus causas, una de ellas es el crecimiento de la tecnología móvil que permite estar conectado al internet en cualquier lugar sin las precauciones debidas para evitar posibles ataques a la información personal efectuada por la débil seguridad que proporcionan las aplicaciones sobre todo si se conecta a redes públicas de internet.

Este trabajo presenta como solución la creación de un framework de seguridad basado en un estudio, análisis y comparación de las normas de seguridad disponibles que permita desarrollar el framework mencionado.

El interés de este trabajo es la capacidad de diseñar un framework compatible con aplicaciones móviles híbridas y nativas de fácil acceso a los usuarios y que permita mejorar la seguridad móvil en conexiones de internet inseguras.

Bibliográficamente se consultó varios portales de programadores así como revistas especializadas y libros que permitieron el desarrollo de este trabajo, evidenciando su importancia y aplicación.

La metodología utilizada para el desarrollo del framework es el Estudio de Mapeo Sistemático (SMS) que facilitó el desarrollo ordenado del mismo.

El trabajo está dividido en seis capítulos que se explican a continuación:

- Capítulo I. Propuesta del Tema, en este capítulo se expone el tema de investigación con una introducción, descripción y justificación de la temática a tratar, así como también se define los objetivos del proyecto que se pretende alcanzar, la metodología utilizada para el desarrollo de la investigación y los resultados que se espera obtener una vez concluido el trabajo.
- Capítulo II. Marco Teórico, en este capítulo se fundamenta conceptualmente los temas tratados para la realización del trabajo de estudio utilizando fuentes bibliográficas de libros y estudios científicos realizados que sustenten el tema de investigación.
- Capítulo III. Estándares, normas y framework de seguridad disponibles en la actualidad para el desarrollo de aplicaciones móviles, en este capítulo se estudian varias alternativas de normas y framework para utilizarlas en el desarrollo de la investigación realizando un cuadro comparativo de las ventajas y desventajas que ofrece cada uno para poder escoger el que más convenga de manera técnica y fundamentada.
- Capítulo IV. Elaboración del Framework, este capítulo comprende una esquematización del framework tomando en cuenta los riesgos y las medidas que se va a tomar para evitar la inseguridad de la información.
- Capítulo V. Desarrollo del Prototipo, este capítulo comprende la elaboración de la aplicación, determinado la arquitectura de desarrollo y los pasos realizados para obtener el resultado final del mismo.

El proyecto de investigación pretende cumplir con los objetivos planteados y convertirse en una herramienta de libre acceso para proteger la seguridad de la información en redes inseguras.

El trabajo presento facilidades en su desarrollo por la amplia fundamentación teórica y acceso a herramientas para el desarrollo del framework. La metodología utilizada Systematic Mapping Study (SMS), esta metodología recopila y analiza estudios primarios, identificando que y cuantas evidencias hay disponibles sobre la seguridad de aplicaciones móviles, intentando identificar clústeres de evidencias y áreas donde fomentar la explicación.

CAPÍTULO I. PROPUESTA DEL TEMA

1.1. Título.

“FRAMEWORK DE SEGURIDAD EN CODIFICACIÓN DE APLICACIONES MÓVILES HÍBRIDOS Y NATIVOS"

1.2. Descripción y justificación.

1.2.1. Descripción del trabajo.

Este trabajo pretende desarrollar un marco de trabajo (framework), que permita la construcción de aplicaciones seguras para el sistema Android, enfocándonos en las que son desarrolladas para el ámbito de las pequeñas y medianas empresas (pymes). Posteriormente, desarrollar un prototipo que siga las normas del framework establecido, demostrando su correcto funcionamiento. Consecutivamente definir una métrica que permita valorar la seguridad obtenida al desarrollar una aplicación.

1.2.2. Descripción del problema.

A partir de la década de los noventa inicia un crecimiento exponencial del uso de telefonía móvil como lo indica (Cantillo, Roura, & Sánchez, 2012) en su informe para las Naciones Unidas. Cada vez es más accecible optar por un disposotivo móvil, llegando a convertirse en un artículo que no puede faltar en nuestro bolsillo y en algunos casos que tenemos siempre en nuestras manos. Más aún con la universalización del internet a través del cual los usuarios se conectan ágil y fácilmente a través de su dispositivo con el afán de estar siempre conectados a la información o redes sociales.

A pesar de la importancia de mantenerce comunicado en todo momento, los dispositivos móviles así como los de escritorio presentan la desventaja de ser vulnerables al robo de información o implantación de virus que deterioren el artefacto o filtren información, así pues la ONU (2016) advierte que 500 millones de dispositivos pueden ser atacados por piratas informáticos aprovechándose de las redes gratuitas y falta de seguridad de los dispositivos.

El INEC presenta en la Encuesta de Tecnologías de la Información y la Comunicación lo siguiente:

> “En el Ecuador en el año 2013 el 16,9% (1’261.944) de las personas de cinco años y más que tienen celular poseen un teléfono inteligente (Smartphone), lo que representa un crecimiento de 141% frente al 2011” (INEC, 2013)

Tomando en cuenta este crecimiento de la utilización de dispositivos inteligentes en nuestro país, varios municipios locales han desarrollado planes para la integración de zonas con internet de libre acceso en los principales centros turísticos que permiten el acceso a la comunicación. Para el 2013 en ciudades ecuatorianas como Riobamba, Cuenca, Guayaquil, Ibarra y Loja, los Municipios han aumentado este servicio en espacios públicos (EL COMERCIO, 2014)

El acceso a estas redes de libre acceso aumenta el riesgo de pérdida de la información de los dispositivos, teniendo en cuenta que no se fomenta una cultura de seguridad informática, así pues se han detectado ciertos casos importantes de robo de información de entidades privadas y públicas. Tomando en cuenta este riesgo es necesario que tanto los desarrolladores, como cada individuo tomen precauciones para no ser víctimas del robo de información y más aun presentándose vulnerables en las conexiones de internet libres.

En la actualidad, varios informes emitidos por diferentes compañías dedicadas a la protección de software, como: F-Secure, Kaspersky, RisklQ, etc. indican que un alto porcentaje de las amenazas desarrolladas están dirigidas para el sistema Android. En el boletín emitido por Kaspersky se indica que el 98.05% de los programas maliciosos para dispositivos, están destinados para android. (Chebyshev & Unuchek, 2014)

Otro problema adicional que se presenta con respecto a la seguridad, es que la mayoría de usuarios no utiliza ningún software de protección, por lo tanto la información del dispositivo puede estar expuesta a cualquier ataque.

Estos problemas, relacionados con las aplicaciones destinadas para las pequeñas y medianas empresas (pymes) pueden ocasionar el robo de información o pérdidas económicas a los empresarios, puesto que estos buscan software libre, sin conocer los riesgos a los que se exponen los datos manipulados por las aplicaciones.

Ante la popularidad que va teniendo el uso de los Smartphone en el ámbito empresarial, debido a que les permite movilizarse y realizar las operaciones, además de ofrecer nuevas oportunidades de negocio, que antes se realizaban expresamente en una oficina, empero, el incremento de ataques dirigidos a este sistema, como se indica en el blog del Instituto Nacional de Ciberseguridad de España, donde se recogen varios datos emitidos por empresas dedicadas a la seguridad, que manifiestan que los ataques a esta plataforma han aumentado considerablemente, por lo tanto, es de vital importancia plantearnos las siguientes interrogantes:

- ¿Las aplicaciones son seguras para los datos utilizados a través del dispositivo?
- ¿Qué tipo que aplicaciones son más vulnerables?
- ¿Utilizan los desarrolladores metodologías de seguridad? (Martínez A. , 2014)

1.2.3. **Justificación**

El presente tema de estudio es de gran importancia puesto que trata de un tema de actualidad y que se está desarrollando habitualmente pues el mundo se mueve a través de la comunicación, así pues tanto de manera personal como empresarial se pretende mantener la información de manera segura.

Una vez identificado algunos de los problemas relacionados con la seguridad de las aplicaciones empresariales, el propósito de este proyecto, es desarrollar un marco de normas (framework) para construir aplicaciones seguras, lo que permitirá salvaguardar la integridad de las aplicaciones y los datos que manipulen las mismas.

Existe información bibliográfica expuesta en internet, así como libros y blogs en los que se soporta esta investigación y al ser una carrera de desarrollo es factible su ejecución.

Finalmente este trabajo de investigación refleja el desarrollo educativo en el cual se plasma la teoría y la práctica creando conocimiento nuevo que contribuye al mejoramiento de los problemas actuales.

1.3. Objetivos

1.3.1. **Objetivo general**

Realizar el análisis de frameworks, metodologías o estándares de seguridad para crear un Framework de seguridad para desarrollar aplicaciones móviles nativas e híbridas.

1.3.2. **Objetivos específicos**

- Investigar estándares, normas y frameworks de seguridad, disponibles en la actualidad para el desarrollo de aplicaciones móviles.
- Elaborar el framework de seguridad que incluya recomendaciones a nivel de arquitectura, y codificación.
- Desarrollar el prototipo del aplicación que responda a los lineamientos planteados para su evaluación

- Presentar los resultados de las pruebas realizadas sobre el funcionamiento del framework y demostrar su efectividad.

1.4. Planteamiento de solución al problema.

El presente trabajo propone la creación de un FRAMEWORK de seguridad para dispositivos móviles el cual se detalla a continuación.

Para realizar el framework se tendrá en cuenta los siguientes factores utilizados para desarrollar aplicaciones:

a. Directrices de codificación segura.
b. Validación de datos procedentes de terceros.
c. Autorizaciones específicas que se utiliza para controlar la información proporcionada.
d. Gestión de errores que se utiliza.
e. Cifrado de datos utilizados en el dispositivo.
f. Protección de capa de transporte.
g. Gestión de sesiones.

Las métricas proporcionan datos en base a unos parámetros predeterminados, lo que permite demostrar que los planteamientos utilizados son eficaces. Por lo tanto utilizaremos métricas que analicen los siguientes aspectos:

- Métricas de procesos de seguridad.
  Estas métricas permitirán determinar si los procesos de seguridad utilizados cumplen con las políticas y normas de seguridad establecidos para desarrollar aplicaciones seguras.
- Métricas de riesgos de seguridad.
  Estas métricas analizaran las medidas preventivas y correctivas que se utilizan en la construcción de software para prevenir los riesgos a los que están expuestos los datos manipulados.

Una vez desarrollada la aplicación y definidas las métricas se procederá a analizar la aplicación, para verificar la eficacia del framework desarrollado.

1.5. Metodología para la solución del problema.

Para el desarrollo de este trabajo se utilizará la siguiente estrategia de desarrollo:
Fases y tareas de la estrategia de investigación.

Para realizar la investigación se utilizará la metodología Systematic Mapping Study (SMS), esta metodología recopila y analiza estudios primarios, permitiendo identificar qué y cuantas evidencias hay disponibles sobre la seguridad de aplicaciones móviles, seleccionando clústeres de evidencias y áreas donde fomentar la explicación.

Las fases y tareas de la estrategia de investigación utilizará la primera etapa de la metodología SMS, y son las siguientes:

1. Planificación de la revisión
    a. Identificar la necesidad de la revisión.
    b. Definir las preguntas de investigación
    c. Desarrollo de protocolo de revisión.
    d. Evaluar el protocolo de revisión.

2. Ejecución de la revisión, para analizar la información obtenida se continuara con la segunda y tercera etapa de la metodología SMS.
    a. Localizar literatura.
    b. Selección de estudios primarios.
    c. Análisis y síntesis de Datos.

Las aplicaciones para móviles tienen al igual que cualquier software informático vulnerabilidades que ponen en peligro la información manipulada, las aplicaciones nativas para móviles permiten obtener un mejor rendimiento del dispositivo y aprovechar de mejor forma las características del mismo, pero necesitan un desarrollo más complejo y por ende es más costoso, mientras que las aplicaciones híbridas se basan en lenguajes HTML5, CSS o JavaScript, ampliamente conocidos, lo cual permite un desarrollo rápido y sencillo de implementar, permitiendo ahorrar tiempo y dinero. El presente estudio presentara unos lineamientos de seguridad aplicables en aplicaciones híbridas y nativas , obteniendo una aplicación segura, que permita a los usuarios para proteger y gestionar eficientemente la información que es el fin último de la informática.

CAPÍTULO II. MARCO TEÓRICO

2.1. Normas

La palabra “norma” viene del latín norma; con ella se designa en primer término, aunque no exclusivamente, un mandato, una prescripción, una orden, aunque esto no supone que sea la única función de la norma, pues autorizar, permitir, derogar, también son funciones de las normas. El Diccionario de la Real Academia de la Lengua Española lo define como la “regla que se debe seguir o a que se deben ajustar las conductas, tareas, actividades, etc.”. Así pues, tenemos que las normas dirigen todas las acciones del hombre, y el sentido que toma esa dirección, dependerá en gran medida del tipo de norma a la cual se sujete el individuo, toda vez que existe una variedad de ellas, las cuales se aplican a diferentes aspectos de la persona. (UNIDEP, 2011)

Son reglas de conductas que nos imponen un determinado modo de obrar o de abstenernos. Las normas pueden ser establecidas desde el propio individuo que se las impone, y en este caso son llamadas normas autónomas, como sucede con las éticas o morales. Así, una persona ayuda a un necesitado porque se lo ordena su propia conciencia, y cuyo castigo también es personal, y está dado por el remordimiento. Una norma es una regla que debe ser respetada y que permite ajustar ciertas conductas o actividades. Las normas se enfocan más en los procesos por los que tienen que pasar los productos y los estándares especifican la calidad con la que debe contar los productos. (Karron, 2013)

2.1.1. Características de las normas

Las normas se caracterizan en razón del sujeto que las emite, así como de su exigencia, su cumplimiento y el ámbito de aplicación de la misma. Existen muchas semejanzas y puntos de contacto entre los tipos de normas, para establecer una diferenciación entre ellas nos valemos de sus características. En ese sentido tenemos las siguientes propiedades:

- Autonomía: en este supuesto el individuo actúa conforme a su libre albedrío, es decir, la conducta con la que obra el sujeto es de acuerdo con su voluntad.
- Heteronomía: consiste en que la norma es dictada por un sujeto distinto al que debe acatarla.
- Unilateralidad: se refiere a que frente al sujeto que está obligado al cumplimiento de la norma, no existe otro que le exija que acate a ésta.
- Bilateralidad: en este caso se imponen deberes y se conceden facultades por lo que existen dos o más partes.

- Interioridad: es la que regula la conducta interior de las personas conforme a la voluntad de ésta, es decir, la intención de la persona.
- Exterioridad: es la que corresponde a la conducta que manifiesta el sujeto de manera exterior.
- Incoercible: en ella no se aplica la fuerza para su cumplimiento.
- Coercibilidad: se caracteriza por tener la posibilidad de aplicar la fuerza para su cumplimiento. (UNIDEP, 2011)

2.1.2. Clasificación de normas

Los juicios que constituyen el mundo normativo rigen la conducta del ser humano en sociedad y se establecen de acuerdo al medio social al que se aplican, es decir, cada norma se encarga de regular los diferentes aspectos de la sociedad, teniendo de ésta manera 4 (cuatro) tipos de normas que cubren el semblante de la sociedad de la cual somos parte y son útiles para abordar los problemas prácticos de una manera eficaz, permitiéndonos saber las posibles opciones que tenemos en relación a la conducta de los demás y con respecto a la propia, introduciendo así el orden en la vida social.

- Normas Morales: son las que el ser humano realiza en forma consciente, libre y responsable con el propósito de hacer el bien, son propias del ser humano y su sanción, en caso de incumplimiento, hemos de responder a nosotros mismos y la sanción o castigo es el remordimiento de conciencia.
- Normas de Trato Social (Sociales): son reglas creadas por la sociedad y cuyo incumplimiento trae el rechazo por parte del grupo social. Estas responden también a la denominación de usos sociales, reglas de trato externo o la de los convencionalismos sociales. Estas reglas pueden presentarse en forma consuetudinaria, como mandatos de la colectividad, como comportamientos necesarios en algunos grupos. Son ciertas prácticas admitidas en la sociedad.
- Normas Religiosas: están integradas por el conjunto de normas manifestadas al hombre por Dios. Son preceptos obligatorios que regulan la conducta del hombre en relación con la divinidad, emanan directamente de Dios o de sus representantes en la Tierra, cuyo cumplimiento está impuesto por la fe. Las normas religiosas, por su naturaleza, participan en gran parte de los rasgos de las normas morales, ya que el contenido de ambas tiene como fin los aspectos interiores de los individuos.
- Normas Jurídicas: Son reglas de conducta de carácter obligatorio que han sido o creadas por un órgano reconocido por el Estado y cuyo incumplimiento trae como

> consecuencia la aplicación de la fuerza (coercitivamente). En esta clase de normas no importa la voluntad del sujeto a quien van dirigidas para su cumplimiento ya que es indiferente que esté de acuerdo o no en acatarlas, pues la característica esencial de las normas jurídicas es la obligatoriedad y la posibilidad que tiene la autoridad de hacerlas cumplir por medio de la fuerza. Tanto las normas morales como las normas jurídicas se encaminan a la creación del orden; pero es diferente el orden propio de la moral del orden característico de las normas jurídicas. Los mandatos contenidos en las normas morales tienen una finalidad ética, pues solo buscan la realización del bien, por lo que se dirigen a la conciencia de los individuos. (UNIDEP, 2011)

2.1.3. Normas para la aplicación de móviles

A la hora de desarrollar aplicaciones móviles debemos tener en cuenta que éstas deben cumplir determinados aspectos legales para poderlas lanzar al mercado. Además, tanto como desarrolladores, como siendo usuarios de la app, el diseño y poco espacio en la pantalla del dispositivo móvil nos hace descuidar la legalidad en apps móviles.

Muchos accedemos a la gratuidad de las apps a cambio a la privacidad y los desarrolladores aceptan carencias y vacíos en la legalidad en apps por lograr llegar a un público más amplio. Algunos puntos que debemos tener en cuenta para este trabajo son los siguientes:

Funcionalidades

Hay que tener muy en cuenta las cosas que podemos hacer y las que no desde la app. Siempre tenemos que utilizar medios lícitos, por lo que debemos tener claro que lo que no se pueda hacer offline o mediante campañas de marketing tradicional, no se podrá hacer desde nuestra app. Siempre tendremos que pedir permiso al usuario para que nos deje ejecutar determinadas funcionalidades, siempre dentro de la legalidad y marco legal.

Derechos propios y de terceros

Antes que nada, se debe tener las licencias de los recursos que se utilizarán, ya sean librerías de programación, bases de datos, elementos gráficos, etc. Del mismo modo que se leerán las condiciones para evitar problemas. Finalmente, es importante recordar que se deberá proteger el código fuente una vez terminado el desarrollo de la app, para evitar plagios, copias o imitaciones del trabajo realizado.

Licencia y condiciones de uso

La redacción de las licencias de uso y condiciones que el usuario deba aceptar para poder hacer uso de la app, deben ser adecuadas a la normativa y a la vez deben permitir que se pueda eximir el desarrollador de cuantas responsabilidades que se puedan dar por el mal uso que se haga de la app desarrollada. Hay que dedicar todo el tiempo que haga falta para el desarrollo de las mismas, pues esa será la mejor defensa que se disponga, en caso de mal uso por parte del usuario.

Información y permisos

Si la aplicación móvil va a necesitar acceder a los contactos de la agenda o a contenidos del móvil, ya sea por cuestión de pagos, cesión de datos o instalación de cookies o simplemente compartir contenidos, el usuario ha de ser informado y deben ser validadas por el mismo de forma sencilla y lo más clara posible antes de su instalación. Además se debería incluir una opción de configuración, donde se pueda denegar los permisos en caso de que el usuario cambie de opinión.

Política de Cookies

La aceptación de las cookies es importante tanto en páginas web como en dispositivos móviles a la hora de descargar aplicaciones móviles. Dependiendo del tamaño de la pantalla de los dispositivos móviles, debe ser visible un aviso informativos con la información básica sobre qué son las cookies, la finalidad de éstas, quien las instala y como rechazarlas.

Informar al usuario

Una gran parte de las aplicaciones móviles pueden ser consideradas como “servicios de la sociedad de la información”, aunque solamente sea por la publicidad que contienen. Por eso es que hay que cumplir con las obligaciones que la legislación implica para estos servicios. La principal obligación más fácil de cumplir es la de informar a los usuarios de los aspectos marcados por la ley, en secciones comúnmente denominadas “acerca de” o “quiénes somos”. Estos apartados proveen al usuario de información respecto a los creadores y quiénes hay detrás de las aplicaciones móviles. Incluye aspectos como el nombre y dominio de la empresa, los datos de inscripción del Registro Mercantil, NIF, la adhesión a códigos de conducta, etc. (Soto, 2014)

2.2. Estándares

Es un conjunto de reglas que deben cumplir los productos, procedimientos o investigaciones que afirmen ser compatibles con el mismo producto. Los estándares ofrecen muchos beneficios, reduciendo las diferencias entre los productos y generando un ambiente de estabilidad, madurez y calidad en beneficio de consumidores e inversores. Los esfuerzos que se están realizando y los ya realizados han perseguido distintos objetivos que van desde la definición de API(Interface de Programación de Aplicaciones), los formatos de los ficheros con la información de parámetros biométricos, la encriptación de la información biométrica, la interacción entre dispositivos biométricos diferentes, etc. (Karron, 2013)

Estándares en aplicaciones móviles

Desarrollo

Los estándares son ciertos lineamientos básicos que las aplicaciones deben tener con el objeto de soportar el desarrollo del software como son las resoluciones, la capacidad el tipo de código fuente, y ciertos aspectos adicionales que se debe tomar en cuenta para que la aplicación se desarrolle con normalidad. Existen varios estándares para el desarrollo de aplicaciones móviles, en este caso particular de investigación los estándares utilizados son:

- Las aplicaciones deben soportar las distintas resoluciones de los distintos tipos de dispositivos.
- El instalador de las aplicaciones no debe superar los 10 MB.
- No embeber imágenes y videos como contenido estático que hagan que los instaladores sean más pesados.
- El código fuente de las aplicaciones deberá ser simple, fácil de comprender, escalable, flexible y deberá ser acompañado de la documentación necesaria para poder asegurar su continuidad soportando un futuro cambio de proveedor.
- Generar y proveer los manuales de uso necesarios para el ciudadano que utiliza la aplicación móvil.
- Control de Versiones:
    - **versión v 0.0.1**: Resolución de bugs de versiones actuales.
    - **versión v.0.1.0**: Nueva funcionalidad dentro de la versión actual.
    - **versión v.1.0.0**: Representan un cambio sustancial respecto de la funcionalidad o de la estética actual del sitio.

- Las aplicaciones con sistemas de registración deben seguir los lineamientos definidos por la Coordinación de Identidad Digital y Tarjeta Inteligente de Servicios Digitales.
- La aplicación debe brindar la posibilidad de operar en forma total o parcial en modo offline (sin conexión). La sincronización y actualización de contenidos se llevarán a cabo una vez que el dispositivo se encuentre nuevamente en modo online.
- El desarrollo se realizará para que sea compatible con la última versión más estable del sistema operativo y contemplando las versiones previas que aún siguen siendo populares. (GitHub, Inc., 2017)

Diseño

El diseño constituye la estructura de la aplicación y en base a ésta se determina la manera como debe estar **Elaborado** el software teniendo en cuenta los permisos respectivos según el organismo de control pertinente para evitar copias sin autorización y posibles problemas legales. Los principales aspectos a tomar en cuenta para el diseño son:

- Los nombres de las aplicaciones deberán contar con la aprobación de la Dirección Nacional de Servicios Digitales.
- Los iconos lanzadores de las aplicaciones serán creados y provistos por la Coordinación de Diseño de Servicios Digitales. (GitHub, Inc., 2017)

Colores

Identifica la paleta de colores a utilizar esta es definida en el manual de identidad visual para web y aplicaciones móviles. Los usos son los siguientes:

- Primario:#0072BC Es el color primario que se utiliza dentro de la UI en elementos como links, botones, etc.
- Secundario: #00B9F1 Se utiliza para ciertos elementos del contenido que necesitan ser destacados, por ejemplo en iconos.
- Complementario: #FD4138 Se utiliza para elementos que necesiten un destaque diferencial y en ciertos elementos para dar calidez en páginas muy extensas que no contengan fotografías.
- Neutros:
- Texto: #111111
- Gris claro: #767676

- Bordes y detalles: #CCCCCC
- Fondo: #F5F5F5
- Blanco: #FFFFFF. (GitHub, Inc., 2017)

Tipografías

La familia tipográfica estándar es Roboto dada su alta legibilidad en medios digitales. Para textos largos se usa Droid Serif para cansar menos al lector, ya que por su forma las letras se identifican más fáciles. Ambas familias tipográficas están hechas por Google y su uso es libre y gratuito bajo licencia Apache 2.0. (GitHub, Inc., 2017)

Estética y elementos gráficos

Como estética se usa un sistema visual minimalista "flat" en donde predomina el diseño del contenido evitando el uso de ornamentos que distraigan. Los elementos se componen de figuras simples con colores plenos, evitando el uso de degradados. Se recomienda el uso de fotografías como parte del contenido, ya que hace más amena la lectura y ayuda a la comunicación. Ver los estándares de contenido para más información.

Usabilidad

La usabilidad comprende la manera cómo se va a desempeñar la aplicación, incluye la resolución de pantallas, el funcionamiento el estado de la aplicación así como la interactividad con el usuario y las etiquetas requeridas, a continuación se detalla los principales aspectos de este ítem:

1. Diseñar para todas las resoluciones de pantalla de computadoras, tabletas y celulares posibles.
2. Usar convenciones de estética y funcionamiento de cada plataforma o sistema operativo (Ej: Guía Google, Guía iOS, Guía BlackBerry, Guía Windows Phone. Para sitios web se deben respetar sólo los lineamientos de identidad visual).
3. Entender qué funcionalidades son las más usadas (sobre la base de la medición de analíticas y pruebas de usabilidad) para hacerlas más fácil de acceder en futuras versiones.
4. Mostrar al usuario el estado del sistema en todo momento. Dónde está (títulos y breadcrumbs), lo que está haciendo y de que se trata (títulos y descripciones), cuál es su progreso (indicadores de carga y barras de progreso), etc).
5. Brindar una respuesta inmediata a cada interacción (un cambio visual, un mensaje de carga, etc).

6. Los elementos interactivos deben tener un estilo para cuando está:
    a. En estado normal
    b. En foco
    c. Activo (cuando se está haciendo clic o tap)
    d. Visitado (para enlaces dentro de un párrafo).
7. Los botones o áreas interactivas deben ser grandes como para no necesitar precisión al hacer clic o tap. Esta necesidad está basada en la ley de Fitts. En aplicaciones móviles el mínimo de un área interactiva debe ser 44x44pts.
8. Los campos de formularios deben mostrar su etiqueta al estar completos. Ver ejemplos de usos correctos e incorrectos.

Seguridad

La seguridad es uno de los aspectos más importantes puesto que de nada sirve una aplicación bien estructurada si presenta vulnerabilidad para que terceros puedan acceder e intervenir en el software e intervenir la información por parte de terceros que manejen la informática. Aquí se estructura las barreras de entrada así como un monitoreo constante de quien puede ingresar y realizar cambios en la aplicación que filtre información del usuario. Los principales aspectos que se refiere a la seguridad son:

- La comunicación de web services debe ser encriptado usando un certificado SSL/TLS
- Todas las URL que lanza una aplicación deben ser https.
- Solo se deben pedir al usuario los permisos mínimos y estrictamente necesarios.
- Los datos de los usuarios usados en la registración deben ser guardados con seguridad.
- Limitar cantidad de intentos de logins.
- Se deben actualizar los framework de desarrollo para evitar vulnerabilidades de seguridad.

Métricas

Métrica

Las métricas son medidas efectuadas sobre cualquier aspecto del software, permitiendo cuantificar y determinar si el mismo cumple con los parámetros requeridos en cuanto a desarrollo, código, diseño y calidad. Las métricas se clasifican de forma general en métricas de producto, métricas de proceso y métricas de calidad. Las métricas de producto miden el

producto del software desde el desarrollo del mismo, hasta la instalación, valorando el diseño y tamaño del mismo. Las métricas de proceso valoran el tiempo de desarrollo, el esfuerzo empleado, la metodología utilizada y la experiencia de los programadores. Las métricas de calidad valoran las características intrínsecas del software desarrollado, como son la corrección, fiabilidad, integridad, mantenibilidad, portabilidad, etc. Las métricas para medir la integridad se encarga de evaluar la capacidad del software para resistir a los ataques accidentales o intencionados relacionados con la seguridad del mismo. (Soria & Cóndor, 2015)

La métrica constituye cualquier medida destinada a conocer el tamaño de un software con sus características individuales con lo que se realizar comparaciones y también planificaciones de proyectos de desarrollo. Así pues la métrica busca determinar la seguridad del framework que se genera en esta investigación a través de la integración de los lineamientos investigados en el metodología de desarrollo. Entre los aspectos más importantes tenemos:

- Análisis realizado por Google Analytics deben incorporar el ID y métricas provistos por la Coordinación de Análisis de Datos de Servicios Digitales.
- Registros de errores y fallas.
- Seguridad de acceso.
- Seguridad de los datos almacenados.

En el presente trabajo, dado que se está valorando la seguridad de una aplicación se utilizará métricas de integridad, se definirán dos atributos adicionales: amenaza y seguridad. Amenaza es la probabilidad de que un ataque de un tipo determinado ocurra en un tiempo determinado. La seguridad es la probabilidad de que se pueda contrarrestar un tipo de ataque determinado. La integridad del sistema se puede definir corno:

Integridad = [1- (amenaza x (1 - seguridad))]

Donde se suman la amenaza y la seguridad para cada tipo de ataque.

Aplicación nativa en Android

La aplicación nativa es aquella que se desarrolla específicamente para un sistema operativo determinado, es decir, es a la medida de cada uno de ellos, por lo que en el caso de aplicaciones nativas se debe crear una con cada lenguaje según el caso, así pues, en lo que se refiere a Android el lenguaje Java el más utilizado. Tomando en cuenta esto se expone dos casos:

Caso 1: Aplicaciones sin código de seguimiento pre-implementado.

Las aplicaciones sin código de seguimiento no recopilan datos de forma automática, por lo tanto, no puede realizar análisis estadísticos o comerciales de forma breve, actualmente es necesario contar con una herramienta de análisis para sacar mayor beneficio a las aplicaciones desarrolladas. Google, a través Google Analytics, ofrece este software de forma gratuita, por lo tanto se puede utilizar con cualquier aplicación previamente desarrollada, para implementar código de seguimiento en aplicaciones Android deberá consultar el documento disponible en la red: " Implementación de Google Tag Manager“ (Google, s.f.)**.**

Caso 2: Aplicaciones con código de seguimiento previamente implementado.

En caso de poseer código de seguimiento integrado se deberá brindar permisos de administración a una cuenta proporcionada únicamente para el fin de análisis de información. Asimismo, se deberá modificar el mismo para que se adapte a los estándares propuestos en el documento de "Guía de estandarización de métricas digitales". También se puede configurar los datos que se recolecten para que se emitan la información que se desea analizar,

Publicación

La publicación consiste en la exposición de las aplicaciones en el mercado informático respetando la legislación pertinente con respecto a la propiedad intelectual de las aplicaciones y no caer en el plagio de software. La publicación debe realizarse tomando en cuenta los siguientes aspectos:

- Las aplicaciones se subirán a las distintas tiendas de Servicios Digitales, excepto en caso de aplicaciones ya existentes.
- Las aplicaciones las firmará Servicios Digitales con la firma y certificados propietarios.
- Los ministerios deben entregar las cuentas de las distintas tiendas y/o repositorios donde estén alojadas las aplicaciones a Servicios Digitales y las firmas digitales (alias y contraseñas) y/o certificados para publicar aplicaciones.
- Los ministerios que dispongan de sus propios proveedores deberán trabajar en un repositorio provisto por Servicios Digitales.
- Los identificadores de cada aplicación deben ser únicos para cada dispositivo y tener el siguiente formato:

- Los datos y las capturas de pantallas de las aplicaciones a ser publicadas deben ser provistas por cada Ministerio/Secretaría, respetando las plantillas provistas por la Coordinación de Aplicaciones Móviles.
- Los datos de contactos/web/detalle de cada aplicación deben pertenecer a cada Ministerio/Secretaría y ser provistos para la carga en las tiendas. (GitHub, Inc., 2017)

2.3. Framework

(Gutiérrez, s.f.) expone sobre el tema lo siguiente:

En general, con el término framework, nos estamos refiriendo a una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un framework se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las últimas piezas para construir una aplicación concreta. Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. Un framework Web, por tanto, podemos definirlo como un conjunto de componentes (por ejemplo clases en java y descriptores y archivos de configuración en XML) que componen un diseño reutilizable que facilita y agiliza el desarrollo de sistemas Web.

Según esta definición se concluye que un Framework es una herramienta que permite la generación una aplicación personalizada, con el objetivo de acelerar procesos de programación usando la ya existente con el uso de patrones de programación constituyéndose en un software fácil de desarrollar y de bajo costo.

Existen varios tipos de framework web: orientados a la interfaz de usuario, como Java Server Faces, orientados a aplicaciones de publicación de documentos, así tenemos Coocon, orientados a la parte de control de eventos, Struts y algunos que incluyen varios elementos por ejemplo Tapestry. La mayoría de framework web se encargan de ofrecer una capa de controladores de acuerdo con el patrón MVC o con el modelo 2 de Servlets y JSP, ofreciendo mecanismos para facilitar la integración con otras herramientas para la implementación de las capas de negocio y presentación. (Gutiérrez, s.f.)

2.3.1. Framework aplicables al proyecto

Como se pudo ver en la definición de framework, este constituye un marco referencial de normas que permiten a los programadores crear aplicaciones seguras, además incluye recomendaciones sobre la arquitectura, codificación y métrica del software.

A continuación, en la tabla No. 1, se listan quince de los framework utilizados por su fácil manejo y costo.

Tabla 1. Top 15 framework más utilizados

Framework Web y móviles	**Descripción**
Angular.js	Un framework basado en JavaScript. de Código abierto mantenido por Google para el Desarrollo Web Front End es decir web de una sola páginaque permite crear aplicaciones SPA Single-Page Applications. Está dentro de la familia de frameworks como BackboneJS o EmberJS.
React	Liberado por Facebook, en JavaScript, permite desarrollar aplicaciones móviles para IOS y Android es una biblioteca Javascript de código abierto para crear interfaces de usuario. Se usa para aplicaciones que usan datos que cambian todo el tiempo. Su objetivo es ser sencillo, declarativo y fácil de combinar
Ionic	Para móviles, usando HTML, Js, Sass y Angular. Es una herramienta, gratuita y open source, para el desarrollo de aplicaciones híbridas basadas en HTML5, CSS y JS. Está construido con Sass y optimizado con AngularJS.
Meteor	Es un framework para aplicaciones web con JavaScript libre y de código abierto escrito usando Node.js. Meteor facilita un la creación rápida de prototipos y produce código multiplataforma (web, Android, iOS).
Ruby on Rails	Framework MVC basado en Ruby, orientado al desarrollo de aplicaciones web. Es un entorno de desarrollo web de código abierto que está optimizado para la satisfacción de los programadores y para la productividad sostenible. Permite escribir un buen código evitando que se repita y favoreciendo la convención antes que la configuración.
CodeIgniter	Poderoso framework PHP liviano y rápido. n framework para el desarrollo de aplicaciones en php que utiliza el MVC. Permite a los programadores Web mejorar la forma de trabajar y hacerlo a mayor velocidad. Al igual que cualquier framework está pensado para gente que tiene un dominio, al menos medio, del lenguaje de programación PHP.
Kohana	Un fork de CodeIgniter. Es un framework para aplicaciones web para PHP5 que implementa el patrón de Modelo Vista Controlador Jerárquico (HMVC). Sus principales objetivos se basan en ser seguro, ligero, y fácil de utilizar.

Framework Web y móviles	Descripción
Django	Framework Python que promueve el desarrollo rápido y el diseño limpio es un framework para aplicaciones web gratuito y open source que respeta el patrón de diseño conocido como Modelo–vista–controlador
CakePHP	Framework MVC para PHP de desarrollo rápido es un famework libre, de código abierto, para el desarrollo rápido de aplicaciones para PHP. Es una estructura fundamental para la ayudar a los programadores a crear aplicaciones web. Permitirte trabajar de forma estructurada y rápida y sin pérdida de flexibilidad.
Zend Framework	Framework para PHP 5, simple, claro y open-source para PHP desarrollado por **Zend**, empresa encargada de la mayor parte de las mejoras hechas a PHP, por lo que se podría decir que es el framework "oficial
Yii	Framework PHP de alto rendimiento basado en componentes. es un framework orientado a objetos, software libre, de alto rendimiento basado en componentes, PHP y framework para aplicaciones web.
Pylons	Es un framework web ligero y hace mucho énfasis en la flexibilidad y el rápido desarrollo. Es un proyecto de código abierto que se desarrolla un conjunto de marcos de aplicaciones web escritas en Python.
Catalyst	Framework para aplicaciones web MVC elegante. Es una estructura de código libre para aplicaciones web escrito en Perl. Soporta la arquitectura MVC, así como soporta algunos patrones web experimentales. Está altamente inspirado en Ruby on Rails, Maypole y Spring. Catalyst promueve la reutilización de los módulos de Perl que ya soportan bien los que requieren las páginas Web. La forma en que Catalyst soporta la arquitectura MVC
Node.js	Framework de código abierto basado en el lenguaje de programación ECMAScrip, creado por Ryan Dahl y apadrinada por la empresa Joyent. Este entorno permite construir programas de red escalables, utiliza el motor V8 JavaStript escrito por Google.
Symfony	Framework full-stack. Es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web.
TurboGears	Próxima generación construida sobre Pylons. Es un megaframework para desarrollo web de código abierto, escrito en Python. Fue creado en el año 2005 por Kevin Dangoor. Es un stack web completo, que abarca desde Pylons, SQLAlchemy, Genshi, Mako, Reponze y ToscaWidgets. Está

Framework Web y móviles	Descripción
	diseñado basado en la arquitectura Modelo–vista–controlador parecido a Stratus o Ruby on Rails, diseñado para generar rápidamente aplicaciones web en Python y que sean fáciles de mantener.

Fuente: (Alcalde, 2016)

Elaborado por: Ramos, Consuelo (2016)

Dentro de los framework de uso exclusivo para aplicaciones móviles tenemos los siguientes expresados en la tabla No. 2.

Tabla 2. Framework para aplicaciones móviles

Framework	Característica
React	SDK de HTML5 usando tecnologías web como HTML, CSS y Javascript.
Meteor	Esta escrito íntegramente en Javascript
Ionic	Usa patrones MVC y MVVM ofrecen gran versatilidad
App Celerator	Permite crear aplicaciones nativas para móviles basándonos en Javascript. Además, no sólo incluye esto, sino automatización de test y una gran comunidad detrás.
jQuery Mobile	En este caso hablamos de HTML5, el cual nos permite escribir una sola versión del código para hacerlo funcionar en diferentes plataformas. Es, sin lugar a dudas, uno de los más utilizados por parte de los desarrolladores
Corona SDK	Es una de las opciones más interesantes. Es muy fácil para adaptarse a él y sobre todo es utilizado en el ámbito de los juegos**.** Eso no quita que se utilice para cualquier tipo de aplicación. Además, tiene más de 500 APIs y soporte para interfaces nativas.
The App Builder	HTML5, el cual nos ofrece una interfaz sin código para desarrollar más rápido. Además, uno de sus aspectos más interesantes es que nos permite publicar directamente una aplicación en Google Play.
PhoneGap	Este framework, patrocinado por Adobe y Apache, nos permite crear aplicaciones multiplataforma que soporte HTML5, CSS y JavaScript, permitiendo incluso embeber este código en aplicaciones para acceder a determinado hardware del dispositivo.
Xamarin	Nos permitirá generar nuestra aplicación para iOS (.APP) y para Android (.APK), la cual ya sí correrá de forma nativa. Gracias a esto, surge una de las grandes ventajas de Xamarin: la reutilización de código. En cualquier aplicación multiplataforma que hayamos desarrollado, hay módulos iOS que hemos tenido que portar a Java, o módulos Android que hemos tenido que portar a Objective-C.Pero en este caso, al desarrollar todas las plataformas en la misma tecnología, no es necesario reescribir el código, al poder

	reutilizar módulos ya implementados. La cosa no queda ahí, pues al poder desarrollar aplicaciones nativas para Windows Phone, Windows 8 y web en C# y .NET, resulta que la reutilización del código la estamos exportando a aún más plataformas que iOS o Android. Según Xamarin, deberíamos poder reutilizar el 90% del código aproximadamente

Fuente: (Zamora, 2015)
Elaborado por: Ramos, Consuelo (2016)

OWASP

Iniciales en inglés de Open Web Application Security Project, en inglés 'Proyecto abierto de seguridad de aplicaciones web') es un proyecto de código abierto dedicado a determinar y combatir las causas que hacen que el software sea inseguro.

El proceso OWASP integral recomienda:

- Un sistema tendrá un número de funciones predefinidas y un conjunto de atacantes que puedan orientar razonablemente instancias del sistema en desarrollo. En conjunto, estos deben constituir el conjunto de actores que deben ser considerados en el mal uso de casos.
- Al igual que en los casos de uso tradicionales, establecer que los actores interactúan con un caso de uso y cómo lo hace mostrando su comunicación y asociación. También como se hace tradicionalmente, se puede dividir a los casos de uso y actores en paquetes sin llegar a ser demasiado difícil de manejar.
- Los casos de mal uso importantes deben estar representados visualmente, en el formato típico de casos de uso, con pasos de un mal uso (por ejemplo, un fondo sombreado). Este en particular se debe hacer cuando el mal uso es efectivamente una anotación de un legítimo caso de uso.
- Esos casos de mal uso que no se representan visualmente pero siguen siendo importantes para comunicar al usuario, debe ser documentado, al igual que cualquier problema no manejado por el modelo de casos de uso.

Aseguramiento de la Aplicación

Los siguientes principios permiten proveer el desarrollo de aplicaciones confiables, con el objetivo de obtener un producto de calidad la tabla No. 3 las detalla a continuación:

Tabla 3. Aseguramiento de la aplicación

Principio General	**Recomendaciones para el aseguramiento**
Autenticación	- Asegúrese que todas las conexiones internas y externas (usuario y entidad) pasan por una forma adecuada y apropiada de autenticación. - Asegúrese que todas las páginas cumplen con el requisito de autenticación. - Pasar las credenciales de autenticación o información sensible sólo a través del método HTTP "POST", no acepte método HTTP "GET". - Asegúrese de que las credenciales de autenticación no atraviesan "el cable" en forma de texto claro.
Autorización	- Asegurar la aplicación ha definido claramente los tipos de usuario y los derechos de dichos usuarios. - Conceda solamente aquellas autorizaciones necesarias para llevar a cabo una función determinada. - Asegúrese de que la autorización mecanismos funciona correctamente, fallará de forma segura, y no puede ser eludido. - No exponga las cuentas y operaciones privilegiadas externamente.
Gestión de Cookies	- Asegurar que las actividades no autorizados no puedan tener lugar a través de la manipulación de la cookies. - Cifrar toda las cookies si contiene datos sensibles. - Asegurar bandera seguro se establece para evitar la transmisión accidental sobre "el cable" en un momento de manera no segura. La bandera seguro dicta que la cookie sólo se debe enviar sobre los medios seguros, como Secure Sockets Layer (SSL). - No almacenar información privada sobre las cookies y si fuera necesario, sólo almacenar lo que es indispensable.
Datos / validación de entrada	- Todas las entradas externas deben ser examinadas y validadas.
Manejo De Errores / La fuga de información	- Asegúrese de que la solicitud es de una manera segura. - Asegurar los recursos se liberan si se produce un error. - No exponga los errores del sistema para el usuario.

Fuente: (Recio & Provencio, 2003)
Elaborado por: Ramos, Consuelo (2016)

En cuanto al aseguramiento de la aplicación en lo referente a móviles debemos considerar lo siguiente:

- Utilizar las directrices de codificación segura para proteger la aplicación contra los riesgos de seguridad en el código nativo, como desbordamientos de buffer y vulnerabilidades de cadena de formato.
- Validar los datos procedentes de los servicios web de terceros y otras fuentes no confiables antes de integrar en la aplicación.
- Uso de un permiso específico de la plataforma para proporcionar sólo los conjuntos mínimos de capacidades requeridas para que una aplicación pueda cumplir con su uso previsto. Esto se aplica al acceso de capacidades tales como la lectura de la ubicación GPS del dispositivo, tomar fotos o grabar audio.
- Evitar escribir la información de identificación personal (PII) y otros datos sensibles al dispositivo cuando sea posible. Si esto no puede deben evitarse:
  - Asegúrese de que todos los datos escritos en el dispositivo se cifran. Discos duros cifrados como, todos los datos deben ser cifrada para la mantenibilidad.
  - Preste especial atención a la gestión de claves de cifrado. Aplicaciones de almacenamiento de claves de cifrado en el dispositivo junto a los datos cifrados no proporcionan seguridad adecuada; atacantes probablemente será capaz de recuperar el datos cifrados, las claves utilizadas para cifrar los datos y las rutinas de aplicación explicando cómo los datos fue cifrada.
  - Nunca almacene información extremadamente sensible como contraseñas y la información de tarjeta de crédito en el dispositivo.
  - Guarde todos los datos de las plataformas fijas que están bajo su control administrativo.
- No permita que los archivos confidenciales sean en modo de lectura y escritura. Las protecciones de archivos de plataforma proporcionadas no pararán atacantes con acceso físico al dispositivo, pero puede ser eficaz en la protección contra ataques de otras aplicaciones maliciosas instaladas en el dispositivo.
- No instalar aplicaciones (OAuth por ejemplo) que asumen que la plataforma móvil es un entorno de confianza.

2.3.2. Arquitectura del Software

Es un nivel de diseño que hace foco en aspectos más allá de los algoritmos y estructuras de datos de la computación; el diseño y especificación de la estructura global del sistema es un nuevo tipo de problema.

En la presente investigación representa el diseño integral del framework que se desarrolla compatible para móviles nativos e híbridos poniendo énfasis en la seguridad del software

creando las ventajas y desventajas de los ya existentes y crear uno que en te caso permita el desarrollo empresarial.

Framework para la arquitectura del software presenta el esquema en la Figura No. 1

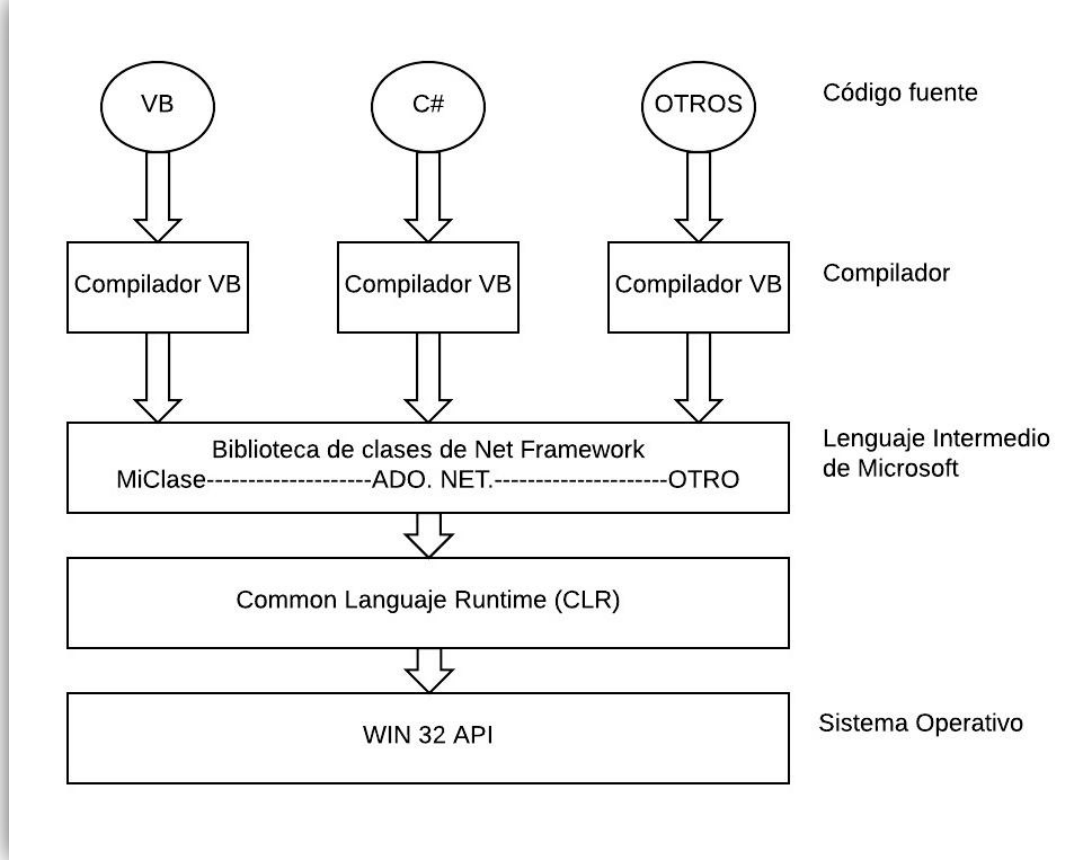


Figura 1. Arquitectura de framework.
Fuente: Adaptada de (Recio & Provencio, 2003)
Elaborado por: Ramos, Consuelo (2016)

2.3.3. Arquitectura del Common Langauje Runtime

Corresponde al lineamiento general que el framework mantiene para su desarrollo y que debe cumplirse para obtener un mejor resultado sobre todo en lo que respecta a seguridad.

La figura No. 2 muestra el esquema de framework.

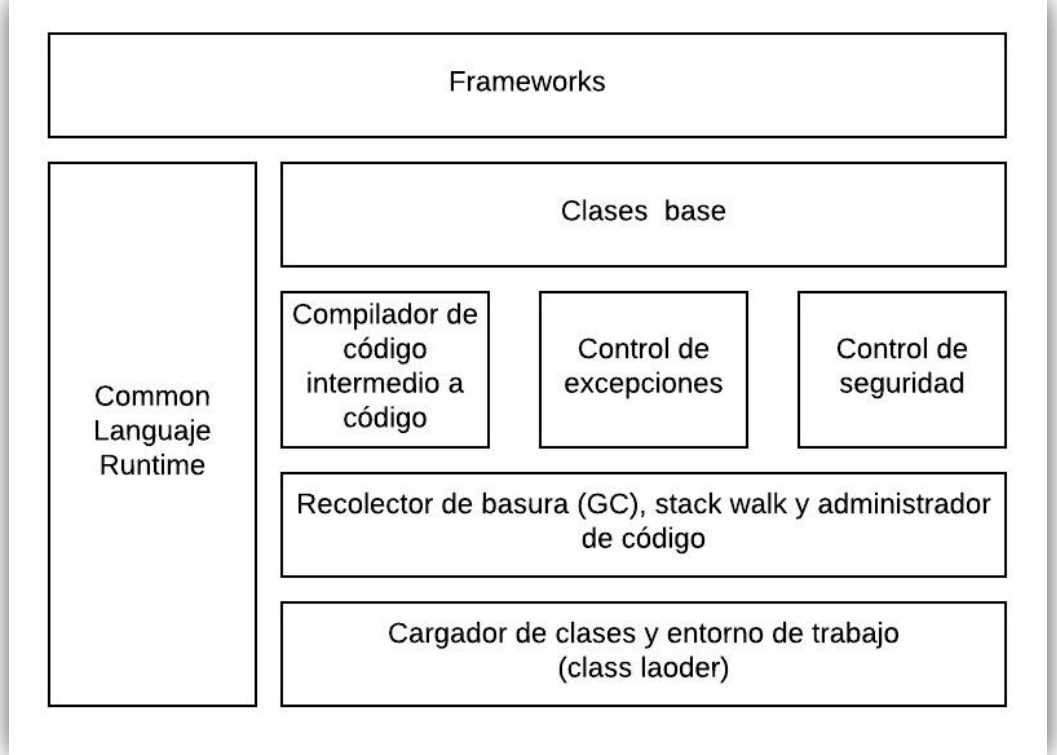


Figura 2. Arquitectura del framework common lenguaje rutine.
Fuente: (Recio & Provencio, 2003)
Elaborado por: Ramos, Consuelo (2016)

A continuación se detalla el esquema del framework expuesto:

Clases base

Definen el entorno de trabajo sobre el que se apoya el código

Compilador de código intermedio a código nativo

Para poder ejecutar el lenguaje intermedio de Microsoft (MSIL), primero debe compilarse a código nativo con Common Language Runtime para la arquitectura del equipo de destino. .NET Framework proporciona dos mecanismos para realizar esta conversión: Un compilador Just-In-Time (JIT) de .NET Framework, y la herramienta Generador de imágenes nativas (Ngen.exe) de .NET Framework.

Control de excepciones

Controlar errores cuando considere que sea razonable y limpiar los recursos después. Pueden generar excepciones Common Language Runtime (CLR), .NET Framework, las

bibliotecas de otros fabricantes o el código de aplicación. Las excepciones se crean mediante la palabra clave throw.

Control de seguridad

Desarrollar código enfocado a la seguridad es una tarea compleja y, por ello, frecuentemente se recurre a la adopción y uso de framework que se enfocan en atender distintas áreas de la seguridad como pueden ser el control de acceso, el cifrado y la validación de entradas entre otras.

Control de recursos recolector de basura

Se dedica a la recolección de basura o spam, control de la pila y como administrador de códigos.

Cargador de clases (Class Loader).

Llama al puntero de inicio del procedimiento, establece su entorno de memoria y mantiene la ejecución bajo control.

2.4. Computación móvil

(Montiel, Hernández, & López, 2012) manifiestan que:

La computación móvil como una disciplina emergente en la computación marca una tendencia futura hacia el “Teletrabajo” o “e-trabajo”, que es la actividad a distancia con el uso de dispositivos móviles, sistemas computacionales e Internet. Los usuarios de dispositivos móviles se incrementan anualmente, dando pie a que el servicio móvil evolucione rápidamente, y que requiera nuevas tecnologías tanto de hardware con bajos recursos y alta eficiencia como de software

El avance tecnológico ha creado un salto en lo que se refiere a la comunicación y desarrollo de la información especialmente en países desarrollados como Estados unidos y Japón donde cada día aparecen nuevos dispositivos móviles con avances que permiten no solamente la comunicación oral, sino el manejo de múltiples herramientas para transmitir la información en varios tipos y esquemas, debilitando por otra parte la comunicación estática como el computador de escritorio y los teléfonos fijos. En Latinoamérica estamos atravesando este cambio generacional gracias a la apertura de mercados generado por la globalización, en estos países se está iniciando el consumo masivo de dispositivos

inteligentes que permiten mejorar la comunicación y a transmisión de la información descuidando la seguridad de la transferencia de datos a un lado. Es por tal razón que se ha iniciado un estudio considerable de temas como el software, donde se conjugan aplicaciones, sistemas operativos, bases de datos móviles entre otras; hardware en lo referente a la optimización de sistemas electrónicos en costo y fidelidad para su uso, educación en lo que tiene que ver a modelos educativos, programas, metodología de aprendizaje entre otras; la interacción entre humano y computador para ver la adaptabilidad, sensores de dispositivos móviles y personalización de tecnologías según el usuario; y finalmente comunicación en lo referente a redes inalámbricas globales como GSM de tercera y cuarta generación. Todo esto se conjuga como tecnología de aplicación móvil y está en pleno desarrollo. (Montiel, Hernández, & López, 2012)

2.4.1. Las aplicaciones móviles

(Software Assurance, 2012) expone sobre las aplicaciones móviles lo siguiente:

Las aplicaciones móviles así como otras aplicaciones necesitan un diseño y arquitectura seguros para mitigar los riesgos de seguridad. Creando una arquitectura y diseño seguros se puede atacar a aplicaciones para móviles con complicados modelos de amenazas, porque un atacante puede acceder y robar información en algún momento, la exposición de un móvil es más alta que la de un computador de escritorio.

(Gasca, Camargo, & Medina, 2014) en su investigación concluye que:

Las aplicaciones móviles deben considerar las características del entorno de ejecución de la aplicación que dispone el teléfono, para garantizar el correcto funcionamiento de la misma. Las aplicaciones móviles pueden ayudar a solventar los problemas de tipo particular o general de la sociedad, debido a sus características de movilidad y ubicuidad. Las etapas de la metodología del framework se muestran en la Figura No. 3

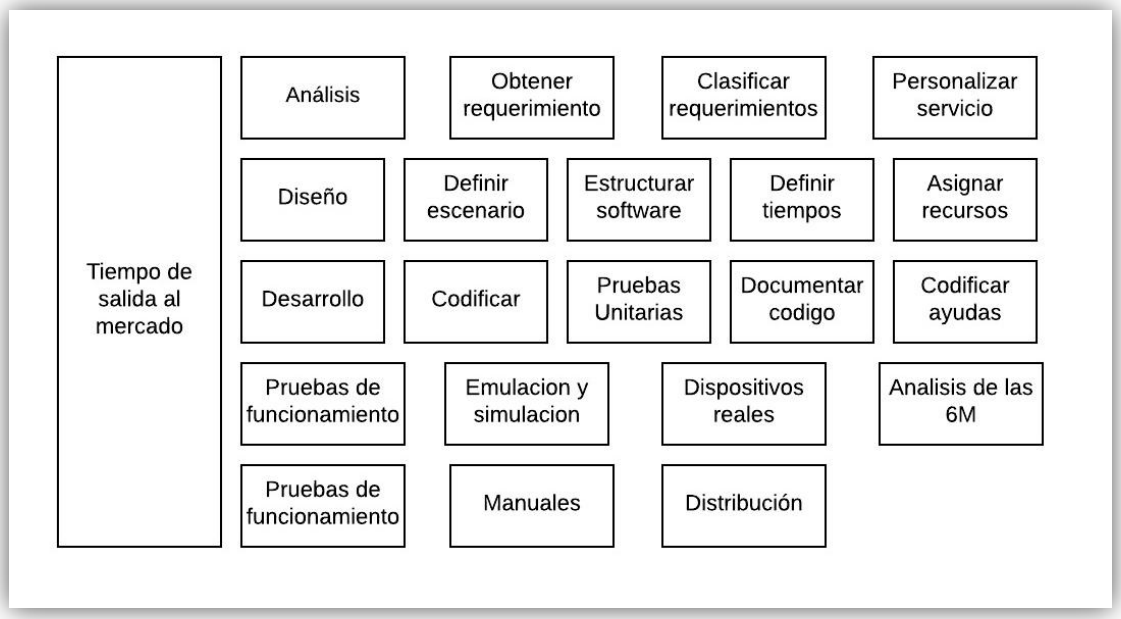


Figura 3. Etapas de la metodología para el desarrollo de aplicaciones móviles
Fuente: Adaptado de (Gasca, Camargo, & Medina, Metodología para el desarrollo de aplicaciones móviles, 2014)
Elaborado por: Ramos, Consuelo (2016)

Las etapas de la metodología para el desarrollo de aplicaciones móviles expuesta en la figura antecedente se detalla a continuación:

Análisis

El análisis se refiere al tratamiento de la aplicación con el cliente en la cual a través de entrevistas se define sus requerimientos para que el software se personalice y brinde los servicios requeridos a través de una constante retroalimentación.

Consta de tres tareas que son:

- Obtener requerimientos: se sugiere hacer una serie de entrevistas al cliente, para que manifieste los síntomas del problema o necesidades que se pretenden solucionar con las tecnologías móviles, o simplemente, para que señale las características que debe tener la aplicación.
- Clasificar los requerimientos: una vez identificados los requerimientos que debe tener el software, se procede a clasificarlos. Dichos requerimientos se pueden clasificar en entorno, mundo, funcionales y no funcionales.
- Personalizar el servicio: adicionalmente se deben analizar aspectos de la cotidianidad del cliente como preferencias, costumbres y particularidades del usuario, con el propósito de garantizar la aceptación del servicio. (Gasca,

Camargo, & Medina, Metodología para el desarrollo de aplicaciones móviles, 2014)

Diseño

El proceso de diseño tiene cuatro tareas que se explican a continuación:

1. Definir el escenario: las aplicaciones móviles se pueden diseñar para ejecutarse en diferentes escenarios, dependiendo del sistema de conexión y sincronización con el servidor o aplicación central; el proceso de sincronización se realiza para insertar, modificar o borrar información. Entre los diferentes escenarios se encuentran los siguientes:
    1 Desconectado: los procesos se realizan en el dispositivo móvil desconectado, después de terminar el proceso, si se requiere, puede conectarse con una aplicación central mediante el proceso de sincronización.
    2 Semiconectado: los procesos pueden ejecutarse en el dispositivo móvil desconectado, pero se requiere establecer conexión en algún momento para terminar el proceso, al sincronizar la información con el servidor o aplicación central. En los escenarios desconectado y semiconectado se recomienda utilizar los protocolos y tecnologías que se ajusten al servicio y capacidades tecnológicas del dispositivo. Algunos son: Media Transfer Protocol (MTP), Near Field Communication (NFC), SlowSync, FastSync, SyncML, entre otros.
    3 Conectado: el dispositivo debe estar siempre conectado con la aplicación central o servidor para su correcto funcionamiento, no se almacenan datos o archivos en el móvil, la sincronización se realiza mediante la validación de formularios, usualmente se utiliza el Protocolo de Transferencia de Hipertexto (Hypertext Transfer Protocol, HTTP).
2. Estructurar el software: se deben utilizar algunos diagramas de Modelado de Lenguaje Unificado, Unified Modeling Language (UML), según las necesidades del proyecto, modelando el sistema desde varias perspectivas Se sugiere traducir los requerimientos obtenidos de la etapa anterior en un diagrama que describa en forma objetiva el servicio por implementar. Además, definir un patrón de diseño para flexibilizar, modular y reutilizar lo desarrollado; la selección del patrón de diseño debe estar acorde con el escenario del servicio. Algunos patrones que se ajustan a los escenarios de las aplicaciones móviles son: modelo vista de controlador, diseño de capas, entre otros.

A continuación la Figura No. 4 muestra el posible diagrama para el desarrollo de aplicaciones móviles.

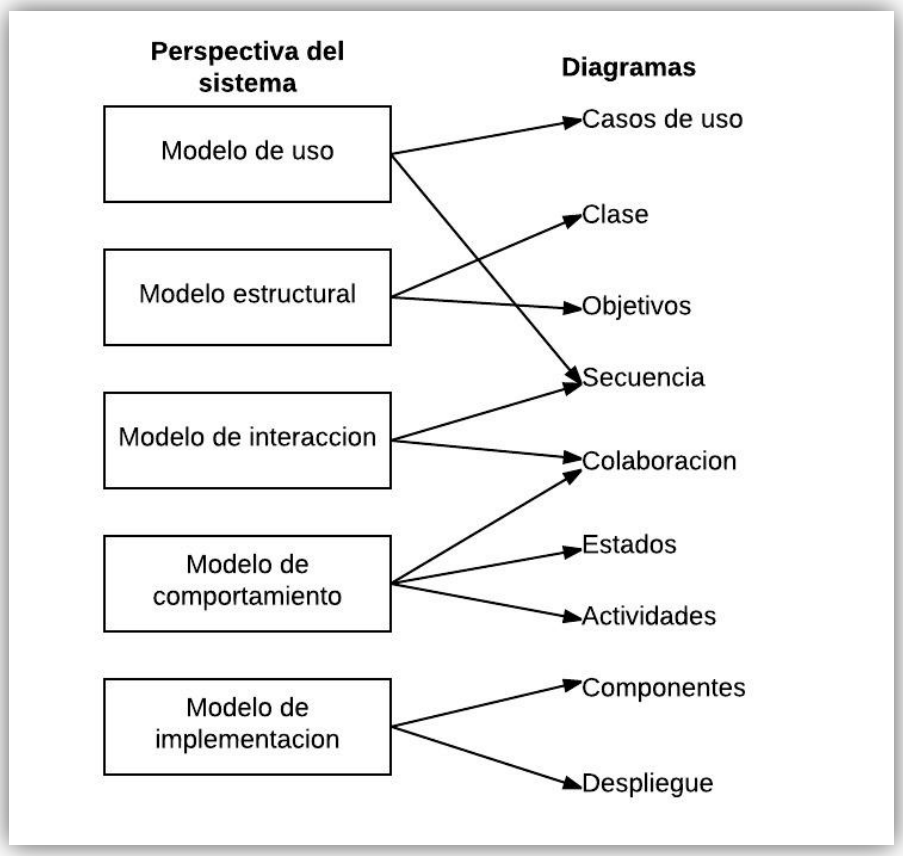


Figura 4. Posibles diagramas para el desarrollo de aplicaciones móviles
Fuente: Adaptado de (Gasca, Camargo, & Medina,2014)
Elaborado por: Ramos, Consuelo (2016)

3. Definir tiempos: se establecen los plazos para cada una de las actividades restantes, con el objetivo de terminar la aplicación a tiempo para su salida al mercado. Se debe tener en cuenta el diseño computacional del software realizado en la tarea anterior y, las características volátiles y dinámicas de los servicios móviles.
4. Asignar recursos: se asignan los recursos para realizar cada actividad y alcanzar los objetivos propuestos, se deben considerar recursos humanos, financieros y tecnológicos. Además, se deben seleccionar las herramientas para el desarrollo de la aplicación móvil. (Gasca, Camargo, & Medina, Metodología para el desarrollo de aplicaciones móviles, 2014)

Desarrollo

El desarrollo es la parte medular donde se programa los diseños aprobados por el cliente y se establece todo lo requerido con el mismo a través del desarrollo propio del software, enmarca las siguientes actividades:

- Codificar: se escribe en el lenguaje de programación seleccionado, cada una de las partes definidas en los diagramas realizados en la etapa de diseño.
- Pruebas unitarias: se verifica el funcionamiento de la aplicación. En primer lugar, se comprueba la correcta operación de cada elemento desarrollado objeto, clase, actividad, documento, entre otros en forma individual; posteriormente, se pone en funcionamiento el conjunto de elementos, comprobando la interrelación entre ellos. Se ejecuta y se observan los resultados obtenidos, para compararlos con los esperados.
- Documentar el código: a medida que se codifica y se prueba cada elemento, se redacta la pequeña documentación sobre lo desarrollado.
- Codificar ayudas: además del manual de instalación y de usuario, deben existir una serie de ayudas que informen de manera didáctica lo que puede hacer el usuario con la aplicación, estas ayudas deben ser codificadas en el mismo lenguaje de programación e integrada en la interfaz de aplicación para visualizarlas en el móvil. (Gasca, Camargo, & Medina, Metodología para el desarrollo de aplicaciones móviles, 2014)

Pruebas de funcionamiento

Las pruebas de funcionamiento se realizan a través de las actividades siguientes. (Gasca, Camargo, & Medina, Metodología para el desarrollo de aplicaciones móviles, 2014):

- Emulación y simulación: se realizan pruebas simulando el escenario y emulando el dispositivo móvil, explorando todas las utilidades y funciones de la aplicación, introduciendo diferentes datos, inclusive erróneos, para medir la funcionalidad y el nivel de robustez del software. Si se encuentran algunas fallas, se debe regresar a la etapa de codificación en la fase de desarrollo para solucionar los problemas, si las pruebas son satisfactorias se procede a la etapa de pruebas con dispositivos reales.
- Dispositivos reales: deben hacerse pruebas de campo en equipos reales para medir el desempeño y el rendimiento del aplicativo. Si se encuentran fallas en el tiempo de ejecución, si el software no cumple con los requerimientos

especificados, o si el cliente solicita un cambio de última hora, hay que regresar a la fase de diseño para reestructurar y solucionar el inconveniente presentado.

- Análisis de las 6 M's: para valorar el potencial de éxito del servicio, se sugiere buscar un grupo de expertos en el campo del desarrollo móvil para que utilicen el método de evaluación de las 6 M's, y califiquen la presencia de los seis atributos en la aplicación desarrollada.

Entrega

(Gasca, Camargo, & Medina, Metodología para el desarrollo de aplicaciones móviles, 2014) exponen las siguientes actividades para la entrega:

- Manuales: el objetivo es el entrenamiento; una aplicación móvil debe constar de un manual del sistema donde se indique el proceso de instalación, la atención a posibles fallas en el tiempo de ejecución y, las especificaciones técnicas mínimas de hardware y software que requiere el equipo, para el funcionamiento adecuado del aplicativo desarrollado.
- Distribución: se define el canal de comercialización de la aplicación, con el propósito de adecuar la aplicación al medio de distribución. A continuación se mencionan algunos de los canales de distribución existentes.

2.4.2. Riesgo de las aplicaciones móviles

Un Smartphone comprometido, automáticamente pone en riesgo: La clave de la/s red/es a la que el dispositivo se conecta; El usuario de la cuenta de correo utilizada; El usuario de la cuenta de mensajería instantánea utilizada; Accesos VPN que puedan haber configurados; Información de contactos personales; Información de acceso en distintas aplicaciones: Facebook, LinkedIn, etc. (Macia, Lanfranco, & Venosa, 2014)

Es muy alto el riego de que los datos puedan ser interceptados por terceros, pues no solamente podría haber atracos a la intimidad, sino también extorciones e incluso espionaje, no solamente los Smartphone están expuestos a esto, sino también las grandes centrales informáticas privadas y públicas que pueden ser vulnerables a ataques informáticos. Así pues la Tabla No. 4 muestra los riesgos y recomendaciones para las aplicaciones móviles.

Tabla 4. Riesgos y recomendaciones para aplicaciones móviles

Riesgo de aplicaciones.	**Detalle de riesgos de aplicaciones.**	**Recomendaciones para aplicaciones.**
1. Almacenamiento Inseguro de datos	Incluye: el almacenamiento de datos en el dispositivo sin codificar, almacenar datos en caché no destinados para el almacenamiento a largo plazo, los permisos débiles o globales, y la plataforma no aprovecha las mejores prácticas	Conservar sólo lo que es absolutamente necesario; Nunca utilizar las zonas de almacenamiento público (por ejemplo, tarjetas SD); apalancamiento de contenedores seguros y las API de cifrado de archivos de plataforma proporcionada; y no conceder permisos grabables archivos legibles mundo o mundo
2. Debilidad de los controles del lado del servidor	Esto se aplica a los servicios de retorno, que no puede confiar en el cliente. Los controles existentes pueden necesitar ser reevaluado (por ejemplo, de comunicaciones de banda).	Entender que los riesgos adicionales de las aplicaciones móviles pueden introducir en las arquitecturas existentes y utilizar la riqueza de conocimientos (por ejemplo OWASP Top 10 Web, Nube Top 10, Cheat Sheets, Guías de desarrollo).
3. Insuficiente protección de la capa de transporte	Incluye: falta total de encriptación para los datos transmitidos; débil cifrado de datos en tránsito; y se ignoraron fuertes encriptaciones de cifrado de seguridad.	Asegúrese de que todos los datos sensibles que salen del dispositivo se cifran, esto incluye datos a través de redes de operadores, Wi-Fi, etc.
4. La inyección del lado del cliente	Con el teléfono móvil en el lado del cliente de una aplicación web, todavía hay algunas caras familiares / ataques como XSS e inyección de SQL, pero además hay nuevos giros, como abusar del marcador de teléfono o SMS y abusando los pagos en la aplicación	Desinfectar o escapar de datos no confiables, antes de emitir o ejecutarlo; declaraciones preparadas utilizados para llamadas de base de datos (concatenación es una mala práctica); y reducir al mínimo las capacidades nativas sensibles ligados a la funcionalidad web híbrida
5. Pobre Autorización y Autenticación	Mala aplicación de la autorización y autenticación en un dispositivo móvil puede causar acceso no autorizado y la escalada de privilegios por los usuarios	La información contextual puede mejorar el proceso de autenticación, pero sólo como parte de una autenticación de varios factores; Nunca utilizar una ID o identificación de abonado como un único autenticador; y autenticar todas las llamadas a la API a los recursos pagados
6. Inadecuada Gestión de la	Sesiones móviles son generalmente más largas, ya	No utilice identificador de dispositivo como un identificador

sesión	que es conveniente y proporciona un mejor uso	de sesión. Hacen que los usuarios vuelvan a autenticar cada cierto tiempo y asegurarse de que las fichas pueden ser revocadas rápidamente en caso de un dispositivo robado / perdido
7. Las decisiones de seguridad de entrada no son de confianza	Las decisiones de seguridad inadecuadas a través de entradas no confiables pueden dar lugar a los atacantes sin pasar por los permisos y modelos de seguridad. Esto es similar pero diferente dependiendo de la plataforma, el IOS puede tener sus esquemas de URL abusados y Android puede tener sus intentos de abuso.	Compruebe los permisos de llamada en los límites de entrada; pedir al usuario autorización adicional antes de permitir la consumación de los recursos pagados; cuando no se pueden realizar comprobaciones de permisos, garantizar se requieren pasos adicionales para poner en marcha acciones sensibles.
8. Canal lateral fuga de datos	Mezcla de no deshabilitar características de la plataforma y defectos programáticos pueden dejar datos sensibles en lugares no deseados, tales como memorias caché web y capturas de pantalla	Entender lo que las bibliotecas de la 3 ª parte de la aplicación esté haciendo con los datos del usuario; Nunca ingrese las credenciales, PII, u otros datos sensibles a los registros del sistema; eliminar los datos sensibles antes de tomar capturas de pantalla; antes de lanzar aplicaciones, depurar a guardar los archivos creados, escritos, o modificados de ninguna manera; y probar que la aplicación a través de tantas versiones de la plataforma como sea posible
9. Criptografía Rota	Codificación, la ofuscación, y la serializarían no se considera cifrado	No guarde la llave con los datos cifrados; utilizar lo que su plataforma ya ofrece; y no desarrollan la criptografía de la casa
10. Sensible Divulgación de información	Las aplicaciones pueden ser revertidos por ingeniería con relativa facilidad, la ofuscación eleva el listón, pero no elimina el riesgo. La ingeniería inversa de la aplicación puede abrir claves de la API, contraseñas, y la lógica de negocio sensibles.	No almacenar las claves de la API privadas en el cliente; mantener la lógica de negocio de propiedad y sensibles en el servidor; y nunca codificar la contraseña.

Fuente: (Software Assurance, 2012)
Elaborado por: Ramos, Consuelo (2016)

2.4.3. Tipos de aplicaciones móviles

Las aplicaciones móviles se clasifican según su uso así pues se puede identificar el tipo de navegador, así como su origen con el objeto de abarcar a la mayoría y adaptarlos a los diversos sistemas operativos. Tomando en cuenta estos aspectos (Rodriguez, 2003) esquematiza los tipos de aplicaciones en tabla 5

Tabla 5. Tipos de aplicaciones móviles

Tipo	Descripción
Acceso de Navegador	Escritos en HTMLS Java Script y CSS3 Desarrollo rápido y barato, pero no tan potente como el nativo
App. Híbridas – Web	Código HTML5 Librerías de ejecución de Worldlight juntas para ser ejecutadas en una shell nativa
App. Nativas	Específica de plataforma. Requiere conocimiento experto, son costosas y tienen un ciclo largo de desarrollo. Pueden brindar una rica experiencia de usuario
App. Híbridas – Mezcla	El usuario añade al código web, código nativo para necesidades únicas y para maximizar la experiencia de usuario

Fuente: (Rodriguez, 2003)
Elaborado por: Ramos, Consuelo (2017)

En la tabla 6 se compara las aplicaciones móviles en base a varios indicadores:

Tabla 6. Comparación entre las aplicaciones móviles

	Web	Híbrida	Nativa
Costes de desarrollo	Razonable	Razonable	Caro
Tiempo de desarrollo	Corto	Corto	Largo
Portabilidad	Alto	Alto	Ninguna
Rendimiento	Rápido	Velocidad nativa si justifica	Muy rápido
Funcionalidad Nativa	No	Todas	Todas
Distribución de AppStores	No	Si	Si
Extensibilidad	No	Si	Si

Fuente: (Rodriguez, 2003)
Elaborado por: Ramos, Consuelo (2016)

Como se puede observar las aplicaciones híbridas son de mejor aplicación tanto por su costo razonable, como por su existencia en tiendas de aplicaciones y el rendimiento que incluso puede ser igual que el nativo que por su parte es costoso, de desarrollo largo y no tiene portabilidad.

2.5. Software seguro

Al desarrollar un software se presentan algunos problemas siendo uno de los principales los defectos en código pues es necesario tener entrenamiento continuo y experiencia para comprender de manera extensa su funcionamiento integral y poder resolverlos, debemos tomar en cuenta que inclusive los más simples y básicos errores en el código puede ser aprovechado por terceros y de esta manera ser atacados y explotados de varias maneras. Esto se debe que al ser desarrollado en una plataforma vulnerable y ser de creación exclusiva del programador, ésta puede ser atacada (Pimienta, Aguilar, Ramírez, & Gallegos, 2014)

2.5.1. Propiedades del Software Seguro

Castellaro, Romaniz, Ramos, & Pessolani en su estudio Hacia la Ingeniería de Software Seguro concluyen que se debe tomar en cuenta propiedades fundamentales y conducentes para el desarrollo de software. Las Propiedades Fundamentales son aquellas que se refieren principalmente a los atributos de seguridad del Software, es decir aquellas cuyo objetivo es proteger al software como tal en todos los aspectos de su desarrollo. Las propiedades fundamentales son:

- Confidencialidad: el software debe asegurar que cualquiera de sus características (incluidas sus relaciones con su ambiente de ejecución y sus usuarios), los activos que administra y/o su contenido son accesibles sólo para las entidades autorizadas e inaccesibles para el resto.
- Integridad: el software y los activos que administra son resistentes y flexibles a la subversión (modificaciones no autorizadas del código, los activos administrados, la configuración o el comportamiento del software por parte de entidades autorizadas). Esta propiedad se debe preservar durante el desarrollo del software y su ejecución.
- Disponibilidad: el software debe estar operativo y accesible a sus usuarios autorizados (humanos o procesos) siempre que se lo requiera; y desempeñarse con una performance adecuada para que los usuarios puedan realizar sus tareas

en forma correcta y dar cumplimiento a los objetivos de la organización que lo utiliza.

Para las entidades que actúan como usuarios (generalmente asociadas con los usuarios finales) se requieren dos propiedades adicionales:

- Responsabilidad: todas las acciones relevantes relacionadas con la seguridad de una entidad que actúa como usuario se deben registrar y trazar a fin de poder establecer responsabilidades; la trazabilidad debe ser posible tanto durante la ocurrencia de las acciones registradas como a posteriori.
- No Repudio: la habilidad de prevenir que una entidad que actúa como usuario desmienta o niegue la responsabilidad de acciones que han sido ejecutadas. Asegura que no se pueda subvertir o eludir la propiedad Accountability. En consecuencia, los efectos de vulnerar la seguridad del software se pueden describir en términos de los efectos sobre estas propiedades fundamentales. (Castellaro, Romaniz, Ramos, & Pessolani, s.f.)

2.5.2. Servicios de seguridad y criptografía

(Carrera, 2012) concluye que "Un servicio de seguridad es cualquier servicio de procesamiento o comunicación que es provisto por un sistema para dar una clase específica de protección a los recursos del sistema".

En la tabla 7 se expone los servicios de seguridad que provee X.800 y RFC 2828

Tabla 7. Servicios de seguridad de X.800 y RFC2828

SERVICIO	EXPLICACIÓN
Autenticación	Identificación de que una comunicación es auténtica
Confidencialidad	Protege la seguridad de datos transmitidos de ataques pasivos
Integridad	Protege modificación de mensaje así como el orden de los mismos
Innegabilidad	Previene que se niegue la existencia de mensajes transmitidos
Control de Acceso	Limita el control y acceso al sistema mediante enlaces de comunicación

Fuente: (Stallings, 2010)
Elaborado por: Ramos, Consuelo (2016)

Los diez principales riesgos de seguridad identificados por Owasp con sus respectivas recomendaciones como se muestra en la tabla 8.

Tabla 8. Riesgos y controles de seguridad según OWASP

Número	RIESGOS	CONTROLES DE SEGURIDAD
1	Debilidad de los controles sensibles	Identificar los controles sensibles

2	Almacenamiento inseguro de datos	Manejar de forma segura las credenciales del usuario
3	Protección insuficiente en la etapa de transporte	Asegurar que los datos son protegidos al transmitirlos
4	Fuga de datos no intencionada	Implementar de forma correcta la autorización, autenticación y manejo de sesiones
5	Pobre autorización y autenticación	Mantener la seguridad con el backend
6	Criptografía rota	Integración de datos segura con otros servicios y aplicaciones
7	Inyección del lado del cliente	Prestar atención específica recogida y utilización de información de usuarios
8	Decisiones de seguridad a través de entradas no confiables	Implementar controles para prevenir el acceso no autorizado a los recursos de pago
9	Manejo de sesión incorrecto	Asegurar la distribución segura y el suministro de las aplicaciones móviles
10	Falta de protección a nivel binario	Revisar cuidadosamente cualquier código de error de educación

Fuente: (Martínez A. , 2014)
Elaborado por: Ramos, Consuelo (2017)

2.6. Metodologías enfocadas al desarrollo de software seguro

Las metodologías para desarrollo de software seguro han ido evolucionando a través del tiempo desde la década de los sesenta y en la actualidad podeos determinar dos tipos: Las metodologías Tradicionales, las metodologías ágiles y las metodologías híbridas.

Metodologías ágiles para desarrollo de Software

Los métodos ágiles nacen a principios de la década de los 90 en contraposición a lo que representaban los métodos tradicionales. Esta explosión de metodologías llevó a que, en febrero del 2001, tras una reunión celebrada en Utah, USA, se acuñara formalmente el término “ágil” aplicado al desarrollo de software. En esta misma reunión participó un grupo de 17 expertos de la industria del software, incluyendo algunos de los creadores o impulsores de metodologías de software, con el objetivo de esbozar los valores y principios que deberían permitir a los equipos desarrollar software rápidamente y respondiendo a los cambios que pudieran surgir a lo largo del proyecto. (Leiva & Villalobos, 2015)

Estas metodologías tienen por objetivo y como su nombre lo indica agilitar el proceso de desarrollo de software ahorrando tiempo y dinero en su ejecución, así como otorgando factibilidad para que una gran número de máquinas y dispositivos puedan ejecutar las

actividades, sin embargo presentan un porcentaje más elevado de errores que pueden ser aprovechados por terceros para su intromisión.

Las principales metodologías se presentan en la tabla No. 9 a continuación:

Tabla 9. Métodos ágiles para desarrollo de software 1990-2000

Método	**Acrónimo**	**Autor (es)**
Adaptive software development	ASD	Highsmith 2000
Agile modeling	AM	Ambler 2002
Cristal methods	CM	Cockbum 1998
Agile RUP	dX	Booch, Martin, Newkirk 1998
Dynamic solutions delivery model	DSDM	Stapleton 1997
eXtreme Programming	XP	Beck 1999
Feature-driven development	FDD	Charette 2001, Mary y Tom Poppendieck
Rapid development	RAD	McConnell 1996
Microsoft solutions framework	MSF	Microsoft 1994
Scrum	Scrum	Sutherland 1994

Fuente: (Leiva & Villalobos, 2015)
Elaborado por: Ramos, Consuelo (2017)

Metodología híbrida

Los métodos híbridos constituyen una mezcla de prácticas y artefactos que no necesariamente provienen de una misma metodología, ni son una variación de una metodología ágil o tradicional. Los métodos híbridos basan su existencia en las debilidades de los métodos anteriormente nombrados, con la finalidad de crear un método robusto pero al mismo tiempo flexible, que combine las bondades de dos o más metodologías ágiles. Los métodos híbridos pretenden retomar las ventajas de los métodos mencionados anteriormente, de tal forma que son una combinación de las mejores prácticas de cada uno de ellos. La nueva tendencia en ingeniería de software es diseñar metodologías híbridas. Esta propuesta es atribuida a Ivar Jacobson, uno de los tres creadores de UML (Unified Modeling Language - Lenguaje Unificado de Modelado, Object Management group, 2011); creador de UP (Unified Process, Proceso Unificado), y ahora creador de EssUP (Essential Unified Process). EssUP es una metodología híbrida que combina RUP con Scrum. (Leiva & Villalobos, 2015)

Son metodologías que combina a las tradicionales y ágiles con el objetivo de crear un software bien definido conceptualmente como los tradicionales, pero con la flexibilidad de los ágiles, dotando al diseñador de una herramienta que oferte mayor seguridad y desempeño.

Entre las principales metodologías se expone las siguientes:

- Correctness by Construction (CbyC)
- Security Development Lifecycle (SDL)
- Security Requirements Engineering Process
- SQUARE
- CoSMo
- UMLsec

Corrección por la construcción (Correctness by Construction CbyC)

Es un método efectivo para desarrollar software que demanda un nivel de seguridad crítico y que además sea demostrable. La empresa Praxis ha utilizado CbyC desde el año 2001 y ha producido software industrial con taza de defectos por debajo de los 0.05 defectos por cada 1000 líneas de código, y con una productividad de 30 líneas de código por persona al día. Las metas principales de ésta metodología son obtener una taza de defectos al mínimo y un alta resilencia al cambio; los cuales se logran debido a dos principios fundamentales: que sea muy difícil introducir errores y asegurarse que los errores sean removidos tan pronto hayan sido inyectados. CbyC busca producir un producto que desde el inicio sea correcto, con requerimientos rigurosos de seguridad, con definición muy detallada del comportamiento del sistema y un diseño sólido y verificable (Croxford & Chapman, 2005).

Fases de la Metodología CbyC CbyC (ver Figura No.5) combina los métodos formales con el desarrollo ágil; utiliza notaciones precisas y un desarrollo incremental que permite mostrar avances para recibir retroalimentación y valoración del producto. (Leiva & Villalobos, 2015)

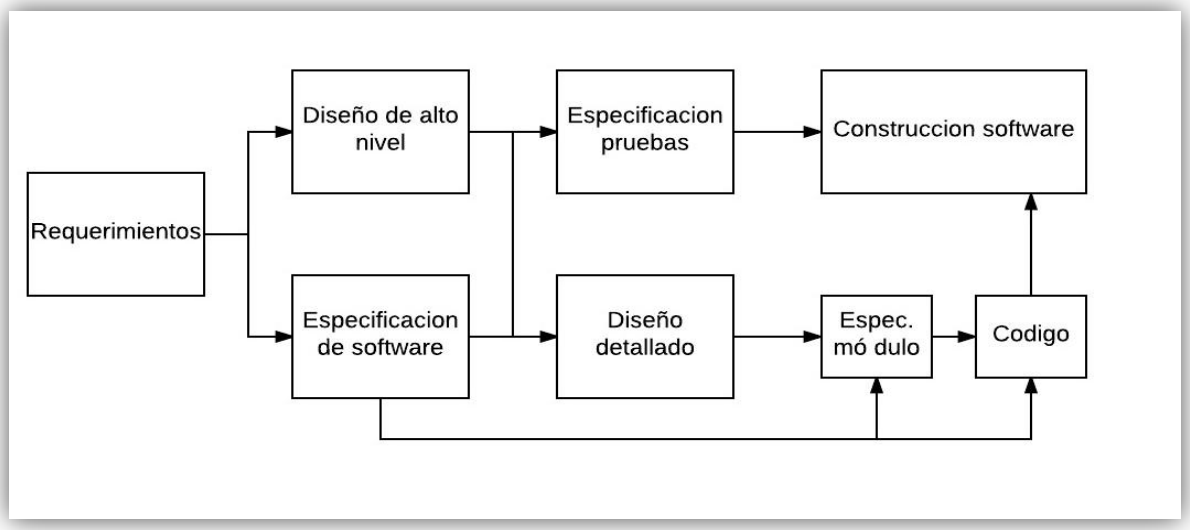


Figura 5. Fases de la metodología (CbyC)
Fuente: Adaptado de (Leiva & Villalobos, 2015)
Elaborado por: Ramos, Consuelo (2016)

Ciclo de vida del desarrollo de la seguridad (Security Development Lifecycle SDL)

Es un proceso para mejorar la seguridad de software propuesto por la compañía de Microsoft en el año 2004; con dieciséis actividades enfocadas a mejorar la seguridad del desarrollo de un producto de software (Peterson, 2011).

Las prácticas que propone SDL van desde una etapa de entrenamiento sobre temas de seguridad, pasando por análisis estático, análisis dinámico, fuzz testing del código hasta tener plan de respuesta a incidentes. Una de las características principales de SDL es el modelado de amenazas que sirve a los desarrolladores para encontrar partes del código, donde probablemente exista vulnerabilidades o sean objeto de ataques (Korkeala, 2011).

Existen dos versiones del SDL, la versión rígida y la orientada al desarrollo ágil. Las diferencias versan en que la segunda desarrolla el producto de manera incremental y en la frecuencia de la ejecución de las actividades para el aseguramiento de la seguridad. La versión rígida del SDL es más apropiada para equipos de desarrollo y proyectos más grandes y no sean susceptibles a cambios durante el proceso (ver Figura No. 6). SDL ágil es recomendable para desarrollos de aplicaciones web o basados en la web (Wood & Knox, 2012).

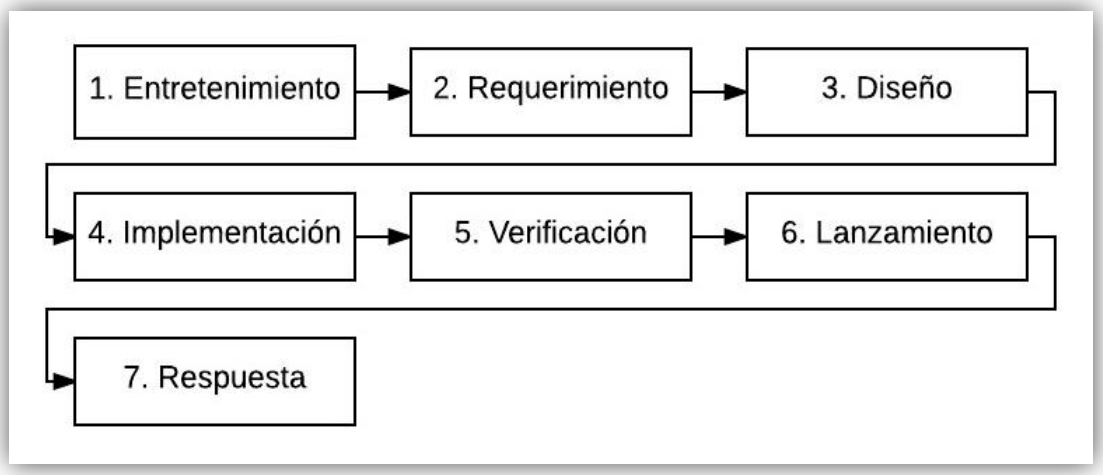


Figura 6. Fases de la metodología SDL
Fuente: (Leiva & Villalobos, 2015)
Elaborado por**:** Ramos, Consuelo (2016)

Proceso de ingeniería de requisitos de seguridad (**Security Requirements Engineering Process SREP)**

El SREP es un método basado en activos y orientado a riesgos, que permite el establecimiento de requisitos de seguridad durante el desarrollo de las aplicaciones.

Básicamente lo que define este método es la implementación de Common Criteria durante todas las fases del desarrollo de software CC es un estándar internacional ISO/IEC 15408 para seguridad informática, cuyo objetivo es definir requisitos seguros que les permitan a los desarrolladores especificar atributos de seguridad y evaluar que dichos productos si cumplan con su cometido. En este proceso también se apunta a integración con el Systems Security Engineering Capability Maturity Model (ISO/IEC 21827), que define un conjunto de características esenciales para el éxito de los procesos de ingeniería de seguridad de una organización. En contraste con su derivado, el CMM, SSE-CMM establece una plataforma para mejorar y medir el desempeño de una aplicación, en cuanto a principios de ingeniería de seguridad, en vez de centrarse en las 22 Process Areas.

Esta metodología trata cada fase del CVDS como a un mini proceso o iteración, dentro de las cuales se aplican las actividades SREP (ver Figura No. 7) que permiten identificar y mantener actualizados los requisitos de seguridad de la fase, permitiendo mitigar efectivamente los riesgos asociados a cada una. (Marulanda & Ceballos, 2012)

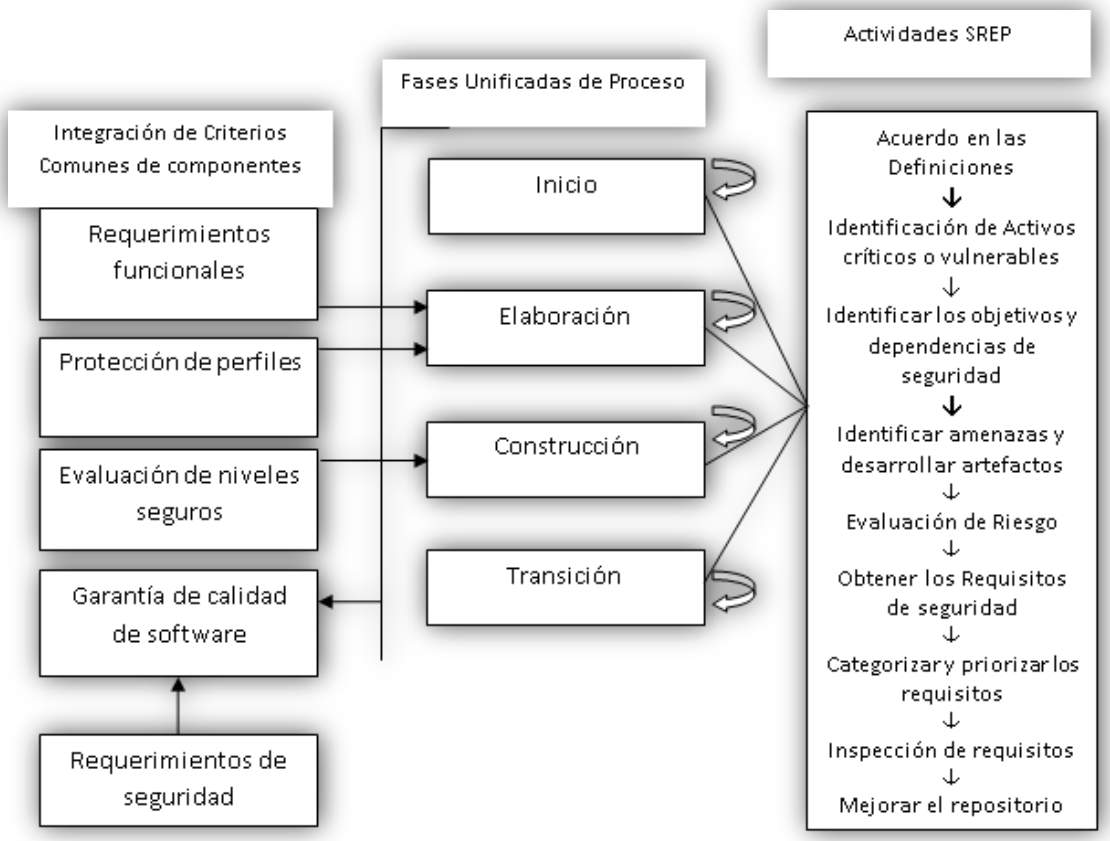


Figura 7. Proceso SREP
Fuente: Adaptado de (Marulanda & Ceballos, 2012)
Elaborado por: Ramos, Consuelo (2016)

Las nueve actividades definidas para cada iteración son las siguientes:

1. Acuerdo en las definiciones, donde se verifica que todos los participantes del software aprueben y estén de acuerdo en la definición de un conjunto de requisitos de seguridad que se adapte a las políticas de la compañía;
2. Identificación de activos críticos o vulnerables, que consiste en realizar una lista detallada de los activos de información más importantes para la organización, con base en el Security Resources Repository, que debe ser obtenido previamente;
3. Identificar los objetivos y dependencias de seguridad, donde se debe tener en cuenta las políticas de seguridad y los aspectos legales para definir correctamente hacia qué se apunta para definir el nivel necesario de seguridad requerido; (4) identificar amenazas y desarrollar artefactos, aquí es necesario identificar todas las amenazas que puedan afectar cada uno de los activos para desarrollar artefactos; (5) evaluación del riesgo, en esta etapa se seleccionan los riesgos a tratar y cuáles a mitigar o transferir, teniendo presente los objetivos definidos y los activos críticos de la aplicación; (6) mejorar los requisitos de seguridad, donde se documentan los requisitos

de seguridad que se deben cumplir, de acuerdo con las amenazas encontradas en el nivel necesario para la aplicación, también se definen las pruebas de seguridad que se aplicarán a cada requisito o conjunto de requisitos; (7) categorizar y priorizar los requisitos, porque una vez identificados se deben ordenar por importancia de acuerdo al impacto que tengan en el cumplimiento de los objetivos; (8) inspección de requisitos, para garantizar la calidad y la consistencia de los hallados deben ser validados por el equipo de trabajo, para que se acepten y refinen de acuerdo con las observaciones encontradas y (9) mejora el repositorio, aquí se deben recopilar todos los documentos, diagramas y modelos que se hayan encontrado durante el ciclo de desarrollo y deben quedar consignados en el SRR. (Marulanda & Ceballos, 2012)

Metodología para aplicaciones móviles

Cuadrado (Square)

El modelo SQUARE —Security Quality Requirements Engineering— propone varios pasos para construir modelos de seguridad desde las etapas tempranas del ciclo de vida del software. En el proceso del modelo se hace un análisis enfocado a la seguridad, los patrones de ataque, las amenazas y las vulnerabilidades y se desarrollan malos casos de uso/abuso. Los pasos son:

- Acuerdo en las definiciones. Se debe generar un acuerdo con el cliente en cuanto a las definiciones de seguridad, como en el control de acceso, la lista de control de acceso, el antivirus, entre otras.
- Identificar metas de seguridad. Que se trazan con base en las propiedades de seguridad definidas en el documento.
- Desarrollar artefactos. Diagramas de arquitectura, casos de mal uso, casos de uso de seguridad, identificar activos y servicios esenciales, árboles de ataques, patrones de ataque, requisitos de seguridad, mecanismos de seguridad.
- Evaluación de los riesgos. Se analizan las amenazas y vulnerabilidades y se definen estrategias de mitigación.
- Elicitación de los requisitos.
- Clasificar los requisitos. De acuerdo con el nivel o las metas de seguridad.
- Priorizar los requisitos. (1) Esenciales, si el producto no puede ser aceptado si él, (2) condicionales, si requerimiento incrementa la seguridad, pero el producto se acepta sin él y (3) opcionales, si es de baja prioridad frente a los esenciales y condicionales. (Marulanda & Ceballos, 2012)

CoSMo

Debido a la necesidad de integrar aspectos de seguridad en el proceso de modelado de software, el modelado conceptual debe abarcar los requisitos y los mecanismos de seguridad de alto nivel. Los autores trabajan en el desarrollo de un método de modelamiento conceptual de seguridad al que denominan CoSMo —Conceptual Security Modeling—. Antes de tener la visión de que los mecanismos de seguridad pueden hacer cumplir los requisitos, se elaboran cuestiones fundamentales de políticas de seguridad; las cuales consisten de un conjunto de leyes, normas y prácticas que regulan cómo una organización gestiona, protege y distribuye información sensible. Cada requisito de seguridad lo puede ejecutar uno o más mecanismos de seguridad, resultando en una matriz de requisitos y mecanismos. Ambos se definen genéricamente porque se usan para modelar la seguridad en el nivel conceptual. En primer lugar, los autores de CoSMo tratan de mostrar cómo integrar las consideraciones de seguridad en el marco de modelado de procesos conceptuales. Después, enumeran de forma sistemática los requisitos de seguridad frecuentes e indican claramente cuáles son los mecanismos para hacerlos cumplir.

En general, un requisito de seguridad exigido no es parte del diagrama de casos de uso, sino de la descripción del caso de uso. En CoSMo, es posible modelar este requisito en el nivel conceptual, incluso en un diagrama de casos de uso. (Marulanda & Ceballos, 2012)

UMLSec

Es una metodología de desarrollo basada en UML para especificar requisitos de seguridad relacionados con integridad y confidencialidad. Mediante mecanismos ligeros de extensión de UML es posible expresar los estereotipos, las etiquetas, las restricciones y el comportamiento de un subsistema en presencia de un ataque. El modelo define un conjunto de operaciones que puede efectuar un atacante a un estereotipo y trata de modelar la actuación del subsistema en su presencia. Esta metodología define varios estereotipos que identifican requisitos de seguridad, como *secrecy*, en el que los subsistemas deben cumplir con que ningún atacante pueda ver su contenido mientras este esté en ejecución y *secure link*, con el que se asegura el cumplimiento de los requisitos de seguridad en la comunicación a nivel físico; lo que pretende evaluar es que un atacante en una dependencia de un subsistema estereotipada, como *secrecy*, que tenga dos nodos *n*, *m* que se comunican a través de un enlace *l* nunca pueda leer el contenido del subsistema. (Marulanda & Ceballos, 2012)

Revisaremos algunos de los modelos, metodologías, estándares y certificaciones que se enfocan o se pueden utilizar para desarrollar aplicaciones móviles de forma segura:

OSSTMM

(Open Source Security Testing Methology Manual) es una metodología de pruebas de seguridad gratuita y abierta del instituto ISECOM. La premisa principal del manual de esta metodología es que “Los hechos no provienen de grandes saltos de descubrimiento, sino más bien de los pequeños pasos y cuidadosos de verificación”. El manual de esta metodología se encuentra en constante y continua revisión y mejoramiento, es revisado por pares de pruebas de seguridad. En general la OSSTMM trata la seguridad operacional para conocer y medir cual es el nivel del funcionamiento de la seguridad. (Ramírez, s.f.)

ISSAF

(Information System Security Assessment Framework) El Marco de Evaluación de Seguridad de Sistemas de Información es una metodología estructurada de análisis de seguridad en varios dominios y detalles específicos de test o pruebas para cada uno de estos. Su objetivo es proporcionar procedimientos muy detallados para el testing de sistemas de información que reflejan situaciones reales. ISSAF proponen cinco fases: I Planeación, II Evaluación, III Tratamiento, IV Acreditación y VI Mantenimiento. (Ramírez, s.f.)

CREST

(Consejo de Auditores de Seguridad Registrados Éticos) existe para servir a las necesidades de un mercado de la información de seguridad global que cada vez más requiere los servicios de una capacidad de pruebas de seguridad regulada y profesional. Proporciona certificaciones reconocidas a nivel mundial, para personas que prestan servicios de pruebas de penetración. (Ramírez, s.f.)

CHECK IT

Health Check es un esquema creado para garantizar que las redes sensibles de los gobiernos y la infraestructura crítica nacional fueran probadas y garantizadas para un alto nivel de seguridad alto y constante. La metodología tiene como objetivo identificar las vulnerabilidades de las tecnologías de la información y las redes que puedan comprometer la seguridad, confidencialidad, disponibilidad de la información contenida en los sistemas informáticos. (Ramírez, s.f.)

ISACA

Se fundó en 1967 y se ha convertido en una organización global para el gobierno de la información, control, seguridad y auditoria de los profesionales de la auditoría de sistemas. Sus normas de auditoría y de control de los sistemas de información son seguidas por los profesionales de todo el mundo y sus temas de investigación en el área. CISA Certified Information Systems Auditor es la certificación principal de ISACA. Desde 1978, el examen CISA ha medido la excelencia en el área de auditoría, control y seguridad, y se ha convertido en una certificación mundialmente reconocida y adoptada en todo el mundo como símbolo de logro. Actualmente han desarrollado estrategias y actividades en el área de la seguridad móvil para que sean implementados en el gobierno de tecnología de las organizaciones y algunos principios de seguridad que deben ser implementados. (Ramírez, s.f.)

En la tabla 10 se realiza un análisis comparativo sobre las metodologías para aplicaciones móviles a continuación:

Tabla 10. Análisis comparativo de metodologías para aplicaciones móviles

Metodología	**Iniciales**	**Referencia**	**Observación**
Open Source Security Testing Methology Manual	OSSTMM	Los hechos no provienen de grandes saltos de descubrimiento, sino más bien de los pequeños pasos y cuidadosos de verificación gratuita	Gratuita
Information System Security Assessment Framework	ISSAF	Seguridad en varios dominios y detalles específicos de test o pruebas	FASES: I Planeación, II Evaluación, III Tratamiento, IV Acreditación y VI Mantenimiento
Consejo de Auditores de Seguridad Registrados Éticos	CREST	Servicios de una capacidad de pruebas de seguridad regulada y profesional	Certificados a nivel mundial
Health Check	CHECK IT	Alto y constante nivel de seguridad de las redes sensibles de los gobiernos y la infraestructura crítica nacional	Identificar las vulnerabilidades de las tecnologías de la información

Sistemas de Información para Auditorías	ISACA	Implementados en el gobierno de tecnología de las organizaciones y algunos principios de seguridad	Certificado por Certified Information Systems Auditor

Fuente: Investigación
Elaborado por: Ramos, Consuelo (2016)

Una vez revisadas varias metodologías de software seguro se concluye que dentro de las tres principales la mejor opción son lo híbridos debido a que brindan una alta seguridad flexible a cualquier tipo de dispositivo dejando de lado la complicidad de los tradicionales y le propensión a errores de los ágiles.

Dentro de este contexto es importante considerar que los dispositivos móviles están en constante evolución y este tipo de metodología es el más adecuado pues siempre está al tanto de los cambios en el mercado tecnológico.

Tomando en cuenta esta referencia dentro de las metodologías de software seguro para móviles se considera el de mejor adaptabilidad y por su manejo simplificado utilizar para el proyecto el método open source security testing methology (OSSTMM) manual pues está disponible para el usuario y siempre está actualizándose con respecto a los avances tecnológico comunicacionales.

CAPÍTULO III. FRAMEWORK DE SEGURIDAD DISPONIBLES EN LA ACTUALIDAD PARA EL DESARROLLO DE APLICACIONES MÓVILES.

Una vez revisadas las diversas metodologías para desarrollar el framework de seguridad para crear una aplicación, enfatizamos la investigación en cuatro de ellas para determinar que lineamientos de seguridad utilizan para su desarrollo.

3.1. Django

Django es una plataforma desarrollada en Python, que es un lenguaje de programación sencillo, flexible y que permite unirse a infraestructuras más complejas por su adaptabilidad. Tiene una velocidad mejorada y un consumo óptimo de banda ancha. Además soporte de pantalla táctil lo que facilita crear contenidos en móviles. Finalmente su plataforma está en varios idiomas lo que mejora su desarrollo.

Este framework contempla el siguiente lineamiento:

Django brinda un modelo que permite estructurar y manipular los datos relacionados de la aplicación web desarrollada. Este modelo comprende las medidas de seguridad que se indican a continuación:

- Autentificar el acceso Verificar (autentificación) que un usuario es quien dice ser (Normalmente comprobando un nombre de usuario y una contraseña contra una tabla de una base de datos)
- Verificar que el usuario está autorizado (autorización) a realizar una operación determinada (normalmente comprobando una tabla de permisos)

Siguiendo estos requerimientos, el sistema aut/aut de Django consta de los siguientes componentes:

- Usuarios: Personas registradas en tu sitio web
- Permisos: Valores binarios (Si/No) que indican si un usuario puede o no realizar una tarea determinada.
- Grupos: Una forma genérica de aplicar etiquetas y permisos a más de un usuario.
- Mensajes: Un mecanismo sencillo que permite enviar y mostrar mensajes del sistema usando una cola.
- Perfiles: Un mecanismo que permite extender los objetos de tipo usuario con campos adicionales.
- Protección contra Cross-site request forgery (CSRF) (falsificación de peticiones inter-sitio)

- Evita que un sitio Web malicioso induzca a un usuario a cargar sin saberlo una URL desde un sitio al cual dicho usuario ya se ha autenticado, por lo tanto saca ventaja de su estado autenticado
- Asegurar que los datos almacenados en campos de formulario ocultos no han sido manipulados.
- Middleware de seguridad
- Generación de URL secretas de una sola vez para permitir el acceso temporal a un recurso protegido, por ejemplo un archivo descargable que un usuario ha pagado.
- Generar URL de "recuperar mi cuenta" para enviar a los usuarios que han perdido su contraseña.

3.2. Yii

Este es un framework de carácter genérico que puede ser adaptado a todo tipo de aplicaciones web es adecuado para el desarrollo de aplicaciones de gestión de portales, foros, sistemas de administración de contenidos (CMS), sistemas de comercio electrónico (e-commerce), etc. El requerimiento principal del fraework Yii es tener un Servidor Web con soporte PHP 5.1.0 o superior. Tiene un marco totalmente basado en Programación Orientada a Objetos. Como la mayoría de los frameworks PHP, Yii es un framework MVC (modelo-vista-controlador).

Yii es muy eficiente y es creado para el desarrollo de aplicaciones Web. Además no es derivado de otro, sino es de creación única según la experiencia y conocimientos del desarrollador basado a varios marcos y aplicaciones. Las medidas de seguridad utilizadas por el framework Yii se detallan a continuación:

- Gestionar la entrada en sesión de un usuario.
- En el método login() de este modelo se crea una instancia del componente
- UserIdentity (en protected/components), que es el que se encarga de validar el usuario y contraseña introducidos y devolver los datos del usuario o un código de error si no son correctos. y el usuario quedará registrado en la sesión.
- Prevención de secuencias de comandos entre sitios
- Comprobar la entrada de los usuarios antes de mostrarlos la codificación HTML con la entrada del usuario.
- Prevención de la falsificación de solicitudes entre sitios

- Almacenar un valor aleatorio en una cookie y comparar este valor con el valor enviado a través de la solicitud POST
- Prevención de ataques de cookies
- Expirar las sesiones apropiadamente, incluyendo todas las cookies y tokens de sesión, para reducir la posibilidad de ser atacado.
- Evite las secuencias de comandos entre sitios que provocan que el código arbitrario se ejecute en el navegador de un usuario y exponga sus cookies.
- Valide los datos de la cookie y detecte si están alterados.
- Yii está equipado con medidas de seguridad para ayudar a prevenir a las aplicaciones web hechas en yii de ataques como inyección SQL, cross-site scripting (XSS), cross-site request forgery (CSRF), y de manipulación de cookies

El framewor Yii se basa en tres lineamientos principales de seguridad que abarcan a los generales pero distribuidos en aspectos diferentes para abarcar un espectro más amplio de protección de datos

3.3. Meteor

Está construida sobre Node.js. utiliza JavaScript y es capaz de compartir código entre cliente y servidor.

Es una plataforma muy potente y muy sencilla de aprender. Meteor permite crear una aplicación web en tiempo real en cuestión de horas. Y si ya hemos hecho desarrollo web, estaremos familiarizados con JavaScript, y ni siquiera tendremos que aprender un nuevo lenguaje. Por otro lado, se asume cierta familiaridad con los conceptos y la sintaxis básica de JavaScript.

Las medidas de seguridad utilizadas por Framework Meteor se indica a continuación:

- Publicación automática e inseguridad
- Reglas de permitir o negar
- Publicaciones y suscripciones
- Verificación y comprobación de los argumentos de auditoría
- Adición de cuentas sin permiso
- Creación de métodos de sólo servidor
- Agregar validación entre clientes y formularios
- Setting.json
- Agregar esquemas a sus colecciones

- Uso de ganchos en el Router de hierro

Como se puede ver el Framework Meteor es uno de los más completos en cuanto a seguridad se refiere, cumpliendo los lineamientos básicos y provee herramientas adicionales de seguridad para la protección de datos.

3.4. JQueryMobile

Es un Framework javaScript para el desarrollo rápido y fácil de sitios webs optimizados para teléfonos móviles**.** Este framework acelera la velocidad y optimiza varias tareas utilizando el lenguaje JavaScript. Ahora con JQueryMobile, evitamos conocer la lógica específica de cada dispositivo y nos centramos en la programación para un solo fin, el navegador de un teléfono móvil.

Las principales características de JQueryMobile son:

- Themes personalizados: El framework permite el uso de themes ya creados y da la posibilidad de crear nuevos themes y trabajar con ellos.
- Tamaño reducido: Toda la librería comprimida pesa menos de 12K.
- Facilidad de uso: Destaca la facilidad para el desarrollo de interfaces de usuario de dispositivos móviles.
- Multiples plataformas: IOS, Android, Blackberry, Palm WebOS, Symbian, Windows Mobile, etc.
- Soporte HTML5: Como su nombre indica, soporta las nuevas etiquetas HTML5.

Las medidas de seguridad utilizadas por este marco se presenta a continuación:

- Cajas de Texto
- Deslizadores
- Switchers
- Elementos de tipo radio
- Elementos de tipo checkbox
- Elementos de tipo select
- Pone a disposición de los desarrolladores la posibilidad de que automáticamente se cree un botón para volver atrás en las páginas de nuestras aplicaciones.
- Proporciona soporte para múltiples páginas dentro de un único documento HTML.
- En caso de que en la petición Ajax se produzca un error, jQuery Mobile mostrara un pequeño mensaje de error que desaparecerá a los pocos segundos.

- Respuesta a ataques
- Listados anidados. jQuery Mobile se encargará de mostrar las listas de forma separada. Esto es, en primer lugar se mostrará la lista principal y al pinchar en cada uno de los elementos se mostrará la sublista incluida en ese elemento.
- A medida que vayamos aumentando el número de botones en la barra de navegación, jQuery Mobile se encargará automáticamente de posicionarlos correctamente en el navegador.
- Al hacer clic en cualquier enlace dentro de un cuadro de diálogo, jQuery Mobile se encargará automáticamente de cerrarlo y mostrar la transición correcta.

jQuery Mobile también pone a disposición de los desarrolladores la posibilidad de que automáticamente se cree un botón para volver atrás en las páginas de nuestras aplicaciones

Este framework presenta varios de los lineamiento generales que se presentan de manera más global.

3.5. Node.js

Este framework de código abierto está basada en JavaScript del lado del servidor y es asíncrono, se ejecuta en el motor V8 desarrollado por Google para su uso en el navegador Chrome, esta función permite crear aplicaciones concurrentes escalables, permitiendo que un solo equipo maneje miles de conexiones simultaneas. Entre algunos de los usos para los que se utiliza esta herramienta es para crear aplicaciones web, aplicaciones móviles, scripts para administrar sistemas.

Las principales características de este framework son:

- Programación orientada por eventos, esto permite que cada conexión dispare una ejecución de evento dentro del proceso, consiguiendo de esta forma que la aplicación creada soporte más usuarios.
- Node.js utiliza el motor V8 de Google, se ejecuta en el lado del servidor, proporcionando compilar y ejecutar de forma muy rápida.
- No es necesario aprender un nuevo lenguaje para utilizarlo.

Las medidas de seguridad más relevantes que utiliza este framework se listan a continuación:

- Implementar el límite de velocidad para evitar ataques.
- Utiliza el middleware csurf para proteger la solicitud entre sitios.
- Para proteger ataques de scripts filtra y sanee la entrada de usuarios.

- Permite utilizar la herramienta sqlmap para detectar vulnerabilidades de inyección de SQL en la aplicación.
- Dispone de Node Security Project, que proporciona herramientas e información para ayudar a mantener la seguridad de Node.

Análisis de metodologías.

Una vez que se ha investigado los framework que se utilizan para desarrollar aplicaciones entre los aspectos más relevantes de la investigación tenemos:

Django al igual que Yii cubren un espectro más alto de seguridades pues ambos ponen énfasis en los permisos de ingreso de usuarios a través de contraseñas que al momento de introducirlas determinan los permisos es decir qué puede o no realizar el usuario en su sesión según el nivel jerárquico que disponga, finalmente presenta una autenticación a través de la cual se confirma que el usuario con su clave sea el correcto así como la determinación de accesos. Por otra parte Meteor presenta un modelo de identificación simple sin mayor detalle en este aspecto y JQueryMobile en cambio muestra un modelo que se enfoca más en la creación de sitios web.

Las medidas de seguridad que utilizan Meteor, JQueryMobile, Django y Node.js son modelos muy explícitos que exponen métodos de protección más complejos pero eficientes, este último dispone de una lista de comprobación para verificar si la aplicación utiliza paquetes con vulnerabilidades conocidas, los otros disponen de medidas como el denegar acceso cuando se ha intentado ingresar la clave por más de tres ocasiones, tiempo máximo de uso para volver a solicitar clave, creación de copias de seguridad en caso de que se necesite realizar una auditoría informática entre otros.

Yii y Django presentan un modelo en el cual de manera práctica se puede realizar auditorías sobre el uso, cambios y visitantes que ha tenido la aplicación lo cual provee de una herramienta segura para la creación de la aplicación empresarial.

JQueryMobile y Yii permiten expirar la sesión dentro de un tiempo determinado, comprobar Cookies y verificar anidados así la aplicación tendrá un tiempo límite de interacción antes de cerrarse y solicitar reingreso de clave, verificar que los cookies utilizados realmente sirvan para mejorar una a aplicación y no contenga virus y finalmente verificar los datos anidados para evitar ataques posteriores, por otra parte Meteor y Django presentan un modelo en el que se crea un seguro contra ataques dentro de servidores por tal razón como es una aplicación para móviles se utilizará lo expuesto por JqueryMobile y Yii

Otras funcionalidades que Django, Metor y JQueryMobile presentan es un modelo que permite la generación de un URL para recuperar cuenta de usuario en caso de que se pierda la clave, así como la creación de un botón automático con acceso exclusivo para la recuperación de datos anteriores en caso de necesitar esta información, además existe la posibilidad de crear una validación entre clientes, todo esto permitiría que la aplicación que se desarrolla en la presente investigación brinde a la empresa de información pasada y presente sobre el manejo administrativo y validaciones entre tablas. Yii por su parte expone recuperación de información automática de aplicaciones adaptable a cualquier aplicación.

CAPÍTULO IV. ELABORACION DEL FRAMEWORK DE SEGURIDAD

4.1. Lineamientos de framework

Metodología para el desarrollo

Los métodos relacionados con el desarrollo de aplicaciones móviles deben incluir un marco de lineamientos a aplicar fase a fase durante todo el progreso de desarrollo. Si bien estas medidas no garantizan una seguridad en su totalidad, pues no son aplicables todas las normas de seguridad a las aplicaciones que se pretende crear, debido a que cada aplicación varía según las necesidades que pretende satisfacer, por lo tanto, lo que propone este framework es un marco de normas siguiendo los lineamientos indicados en la tabla nº 11 con el objetivo de reducir las vulnerabilidades del software.

Tabla 11. Lineamientos del framework propuesto y sus fases.

FRAMEWORK				
IDENTIFICACIÓN	**PROTECCIÓN**	**DETECCIÓN**	**RESPUESTA**	**RECUPERACIÓN**
1. Manejo de accesos 2. Ambiente de negocios 3. Administración 4. Valoración del riesgo 5. Estrategia de Gestión de Riesgos	1. Control de acceso 2. Seguridad de datos 3. Información de Protección, Procesos y Procedimientos 4. Mantenimiento 5. Protección tecnológica	1. Eventos y anomalías 2. Monitoreo Continuo de seguridad 3. Detección de procesos	1. Plan de respuesta 2. Comunicaciones 3. Análisis 4. Mitigación 5. Mejoras	1. Plan de recuperación 2. Mejoras 3. Comunicaciones

Fuente: (Esaú, 2016)
Elaborado por: Ramos, Consuelo (2017)

Una vez analizados los frameworks disponibles para el desarrollo de aplicaciones, se ha organizado las medidas de seguridad utilizadas por cada uno de los mismos según los lineamientos indicados anteriormente para proceder a modelar el framework de seguridad. A continuación se presenta la tabla nº 12 con las medidas de seguridad más utilizadas de cada framework analizada y entre paréntesis se muestra la característica relacionada con el lineamiento.

Tabla 12. Medidas de seguridad utilizadas por framework analizadas.

Lineamiento	**Medidas utilizadas por framework**	**DJANGO**	**JQUERY MOBILE**	**YII**	**METEOR**	**NODE.JS**
Identificación	Ingresar usuario y contraseña a una base de datos (1)	X		X		X
	Determinar permisos para tareas (1)	X		X		X
	Autentificar usuario (1)	X		X		X
Protección	Solicitar prueba de ingreso de un humano (1)	X	X		X	
	Determinar tiempo máximo de espera para ingresar clave (2)	X	X		X	X
	Restringir ingreso si se falla en la clave y usuario varias ocasiones seguidas (2)	X	X		X	X
	Administrar la información de acceso según usuario (3)	X	X		X	X
	Incluir un botón de regresar (3)	X	X		X	
	Crear copias de seguridad para en caso de intromisión volver a un punto anterior de modificaciones (4)	X	X		X	
	Incluir preguntas adicionales para usuario administrador (3)	X	X		X	
	Solicitar contraseña en caso de permanecer por más de 30 minutos dentro de la aplicación (5)	X	X		X	
	Incluir protección contra Cross-site (5)	X	X		X	X
Detección	Incluir auditoría para revisar cambios no autorizados (1)	X		X		
	Prevención de ataque de cokies (2)	X		X		X
	Determinar visitantes a la aplicación (2)	X		X		
	Ejecutar actualizaciones semanales de visitas (3)	X		X		
Respuesta	Expirar las sesiones apropiadamente (4)		X	X		

Lineamiento	Medidas utilizadas por framework	DJANGO	JQUERY MOBILE	YII	METEOR	NODE.JS
	Validar los datos de cookie y detectar alteraciones (3)		X	X		X
	Enlistar anidados (4)		X	X		
Recuperación	Generar URL para recuperar cuenta (1)	X	X			
	Crear botón automático para recuperar estados anteriores en las aplicaciones (2)	X	X			
	Agregar validación entre clientes y formularios (1)				X	X

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

4.2. Modelado de framework.

Fases de cada uno de los lineamientos del framework.

Identificación

En este lineamiento se realizara un análisis a los requisitos de la aplicación a desarrollar, es de suma importancia comprender bien los mismos, pues en ocasiones el usuario final prefiere tener una brecha de inseguridad, pero que el software satisfaga sus necesidades. Para identificar los riesgos a los que está expuesta con respecto al acceso de uso de la aplicación, este lineamiento estará compuesto por las siguientes fases indicadas en la figura 8.

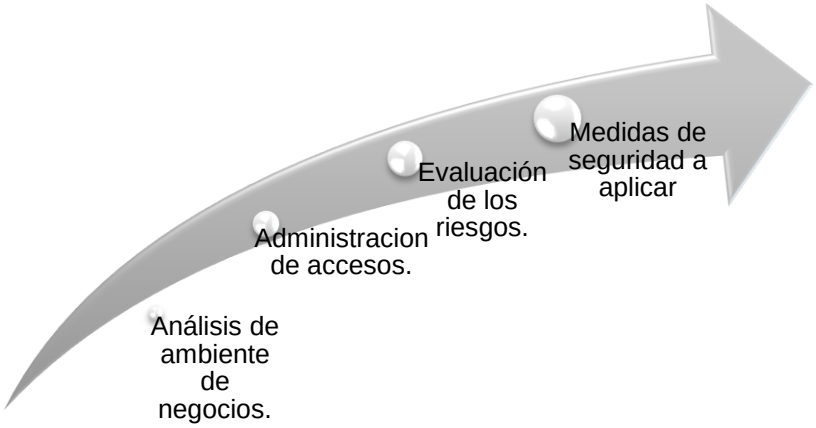


Figura 8. Fases del lineamiento de identificación.
Fuente: Investigación
Elaborado por: Ramos, Consuelo (2017)

- Análisis de ambiente de negocios.

  Realizar una observación del negocio para el cual se está desarrollando la aplicación para determinar cuál es el entorno principal de la organización que utilizara el software, cuales son los usuarios principales y cual es fin principal de la aplicación.
- Administración de accesos a la aplicación.

  En esta fase se procura determinar el número de usuarios que tendrá acceso a la aplicación para el uso de la misma.
- Evaluación de los riesgos.

  Efectuar un análisis para determinar los riesgos de seguridad derivados de la administración de accesos.
- Medidas de seguridad a aplicar.

  En este punto se debe seleccionar las normas de seguridad que se utilizaran para proteger a la aplicación de los riesgos encontrados. Entre las medidas de seguridad analizadas las que se deben tener en cuenta son:
  - Gestión de usuarios

    Utilizar algoritmos de encriptación para que los nombres de usuarios sean almacenados en la base de datos.
  - Autenticación

    Utilizar algún algoritmo de encriptación para encriptar la clave antes de ser almacenada en la base de datos.
  - Autorización

    Gestionar en la base de datos los permisos de los usuarios para acceder a los datos.

Protección

En esta fase se aplicara una análisis para implementar medidas que permitan proteger la información manipulada por la aplicación, pues una vez que los usuarios tengan acceso a la misma, la seguridad de los datos puede verse vulnerada, en la figura 9 se indica las fases de este lineamiento.

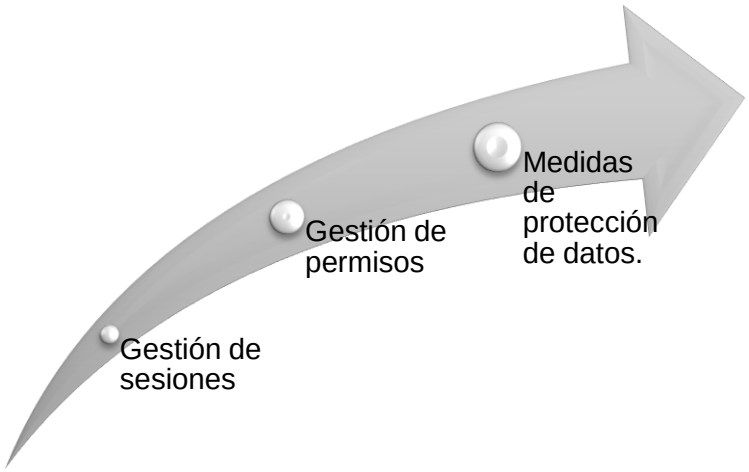


Figura 9. Fases del lineamiento de protección.
Fuente: Investigación
Elaborado por: Ramos, Consuelo (2017)

- Gestión de sesiones

  Una vez determinada la forma de acceso de los usuarios se debe gestionar la sesión que tendrá la aplicación, de forma que se puedan aplicar medidas de seguridad para evitar brechas en la seguridad de la aplicación.

- Gestión de permisos

  Es importante predeterminar los permisos que tienen los usuarios de la aplicación, de esta forma se puede controlar que es lo que realiza cada usuario.

- Medidas de protección de datos.

  En este punto se deben tomar en cuenta las siguientes medidas de seguridad, según requiera la aplicación:

  - Solicitar prueba de ingreso de un humano.

    Usar código de verificación para comprobar que la persona que accede al sistema es humano, de esta forma se evita que los bots realicen ataques.

  - Evitar almacenar contraseñas.

    Las aplicaciones no deben almacenar contraseñas en el lado cliente a través de cookies.

  - Bloqueo de acceso tras un determinado número de intentos de accesos fallidos.

    En el lado del servidor utilizar un contador de ingreso de contraseñas, para que una vez superado el numero preestablecido se bloquee el acceso. Una vez bloqueado solo el administrador puede proporcionar los permisos nuevamente.

  - Gestionar la caducidad de las sesiones.

Para evitar que las sesiones se queden abiertas se debe determinar tiempo de vida de las cookie.

- Administrar la información de acceso de usuario.

  Establecer en lo posible, el menor número posible de administradores generales de la aplicación para que puedan administrar los usuarios de la aplicación.

Detección

Este lineamiento se encarga de detectar los cambios o intrusiones no autorizadas que pueda sufrir la aplicación, para ello se completaran las siguientes fases indicadas en la figura 10.

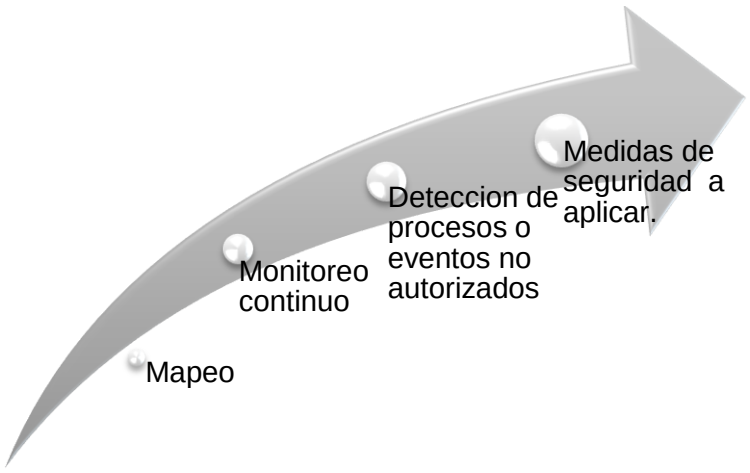


Figura 10. Fases del lineamiento de detección.
Fuente: Investigación
Elaborado por: Ramos, Consuelo (2017)

- Mapeo

  Esta fase implica entender cómo funciona, las relaciones y la estructura que tiene la aplicación, conocimiento que facilitara la detección de cualquier intrusión.

- Monitoreo continuo

  Este es un proceso que verificara el funcionamiento de la las actividades más vulnerables, para verificar que su ejecución no varíe de la forma estándar de funcionamiento.

- Detección de procesos o eventos no autorizados.

  Esta fase cuando detecta una vulnerabilidad debe recopilar toda la información disponible hasta este momento para determinar las medidas que se tomaran.

- Medidas de seguridad a aplicar en esta fase:

- Realizar una auditoría periódica para revisar los cambios no autorizados.
  Para efectuar auditorias se puede utilizar como referente OWASP, que está orientada al análisis de seguridad, los puntos a tomar en cuenta para la auditoria son: análisis funcional, análisis técnico, diseño, desarrollo y realización de pruebas de seguridad. También, en las aplicaciones que lo permitan se pueden utilizar herramientas automáticas como Nikto o W3AF para detectar vulnerabilidades.
- Prevención de ataque de cookies.
  Implementar la extensión HttpOnly al final de una cabecera de cookie, permitirá que la información de las mismas sean solo de acceso exclusivo del lado del servidor (Alarcon, 2009).
- Determinar visitantes a la aplicación.
  Notificar y solicitar permisos al usuario final para la utilización de cookies propias y de terceros.
- Ejecutar actualizaciones semanales de visitas.

Respuesta

Este lineamiento es importante porque deberá presentar soluciones a las intrusiones, ataques o cualquier anomalía detectada en la aplicación. La eficacia y la rapidez con la que se responda a un ataque permitirá salvaguardar la información del usuario. Para esta fase se realizara las fases indicadas en la figura 11.

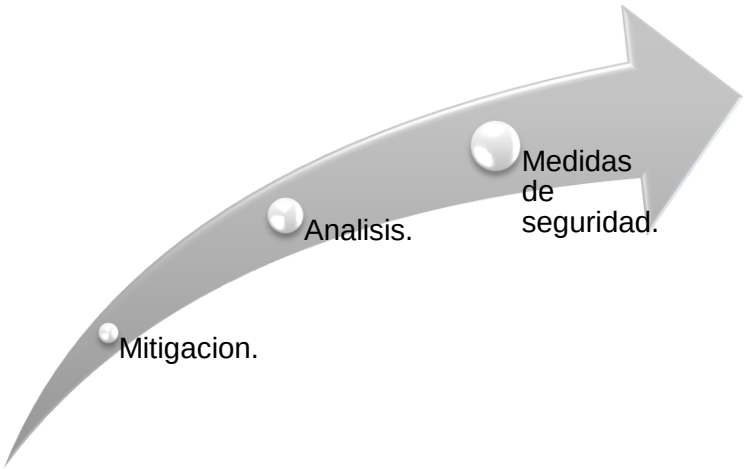


Figura 11. Fases del lineamiento de respuesta.
Fuente: Investigación
Elaborado por: Ramos, Consuelo (2017)

- Mitigación

  Esta fase se encarga de aplicar normas de seguridad para mitigar los riesgos inherentes a la aplicación.

- Análisis

  Si la aplicación ha recibido un ataque no previsto se deberá realizar un análisis exhaustivo del ataque a la aplicación para poder determinar que partes son las afectadas y tomar las medidas necesarias para que la aplicación vuelva a funcionar de forma correcta.

- Medidas de seguridad.

  Una aplicación debe prever los riesgos a los que es vulnerable y aplicar medidas de seguridad de respuesta para actuar en caso de que la aplicación lo requiera.

  - Expirar sesiones.

    Aplicar tiempos preestablecidos de cierre de la aplicación si esta esta inactiva.

  - Validar datos de cookies y detectar alteraciones.

    Verificar que los datos que se almacenen en las cookies sean los estrictamente necesarios para el correcto funcionamiento de la aplicación, de esta forma será más factible detectar cuando los datos almacenados están siendo manipulados.

  - Enlistar anidados.

    Las funciones anidadas contienen una función dentro de ellas, es decir tienen como elemento necesario otra función para poder realizar una operación. Estas deben estar en una lista, donde se explique la función de cada una de ellas para detectar vulnerabilidades y presentar una respuesta en caso de un fallo de seguridad.

Recuperación

Ser capaz de recuperarse de una brecha importante (o menor) de datos (u otro desastre de datos) de forma rápida y confiable es lo que dará confianza a los usuarios finales. Los planes de recuperación necesitan ser documentados y probados, luego revisados al menos anualmente. Esta fase consta de los puntos indicados en la figura 12.

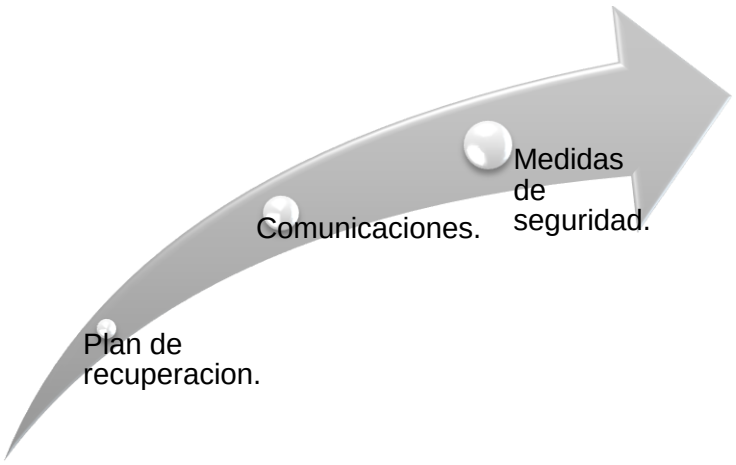


Figura 12. Fases del lineamiento de recuperación.
Fuente: Investigación
Elaborado por: Ramos, Consuelo (2017)

- Plan de recuperación

  El principal objetivo de esta fase es garantizar la capacidad de recuperación de la información y reducir las consecuencias una vez sufrido un ataque. Una vez preparado el plan se debe guardar en lugar seguro y accesible, en este plan se enumerará las funciones predefinidas que la aplicación pueda realizar para determinar el grado de vulnerabilidad de la aplicación, y las potenciales exposiciones del software para introducir controles que eviten la filtración de la información o manipulación de la misma, pues al tratarse de información contable puede acarrear problemas serios con el desarrollo empresarial y toma de decisiones.

- Comunicaciones.

  La estrategia de comunicación tiene por objetivo propiciar la participación de los involucrados para tomar las decisiones necesarias en caso de presentarse un desastre.

- Medidas de seguridad.

  En este punto se aplicaran las siguientes medidas de seguridad que utilizan las metodologías analizadas:

  - Generar URL para recuperar cuenta.

    Proporcionar un enlace para recuperar la cuenta de forma segura.

  - Proveer un botón automático para recuperar un estado anterior de la aplicación.

    En las aplicaciones que lo permitan se debe proporcionar medios para volver a un estado anterior, restaurando la última copia guardada.

  - Agregar validación entre clientes y formularios.

Para autenticar y validar es necesario establecer un protocolo de comunicación entre las partes de forma que la parte verificadora (normalmente) prueba validar que la parte que se identifica (normalmente un usuario) realmente es quien dice que es.

- Realizar copias de seguridad.
  Proporcionar medios para realizar copias de seguridad.

4.3. Framework de seguridad aplicaciones móviles en el ciclo de desarrollo.

El framework de seguridad para aplicaciones presentado anteriormente debe formar parte de cualquier metodología de desarrollo de software que se utilice para crear una aplicación, esto permitirá que la misma sea confiable. De una manera resumida se puede decir que la integración del framework de seguridad en el ciclo de desarrollo consiste en ir incluyendo una fase de seguridad sobre una fase del ciclo de desarrollo.

En la figura 13 se indica, a modo de ejemplo, el framework de seguridad incluida en el ciclo de desarrollo de aplicaciones móviles, Mobile-D.

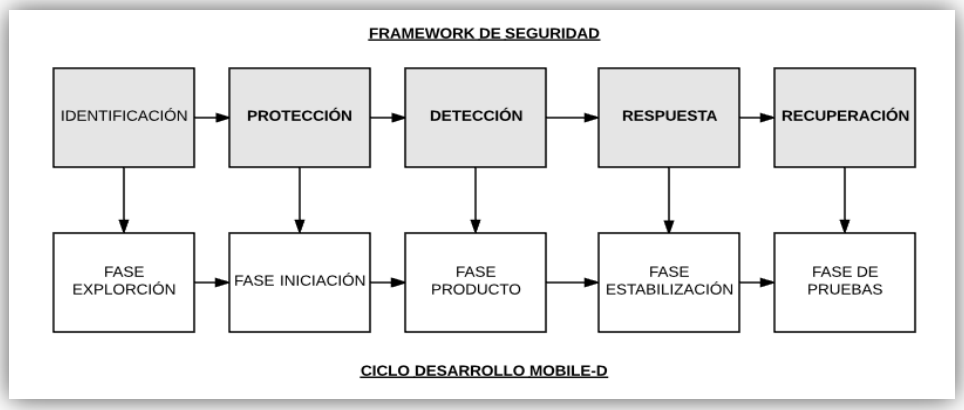


Figura 13. Incorporación del framework de seguridad en un ciclo de desarrollo
Fuente: Investigación
Elaborado por: Ramos, Consuelo (2017)

La fase de identificación integrada en la fase de inicio aplicara un análisis a los requisitos para poder utilizar las medidas de seguridad que ayuden a prevenir los problemas de autenticación. La fase de protección una vez incluida en la fase de elaboración ayudara a incluir medidas que protejan la información que manipula la aplicación. La fase de detección integrada en la fase de desarrollo permitirá que se detecte las partes más vulnerables de la aplicación y se apliquen normas de desarrollo para intentar mitigar los posibles efectos que pueda causar una falla de seguridad. La fase de respuesta y recuperación una vez incluidas

en la fase de transición permitirá que se apliquen normas de seguridad que tomen medidas para responder a un fallo, a recuperar la información o el estado de la aplicación.

Se asume que los ataques de seguridad son inevitables, conforme pasa el tiempo van apareciendo nuevas formas de vulnerar la seguridad, por lo que se debe incluir un framework de seguridad para poder contrarrestar los efectos que esto pueda ocasionar, incrementado de esta forma la calidad del producto final.

4.5. Medidas que se utilizaran en el desarrollo del prototipo.

Una vez seleccionado el prototipo en el que se aplicara el framework de seguridad, en el cual se han incluido las medidas de seguridad recogidas en la tabla 13, se realiza un análisis para determinar cuáles son compatibles con nuestro proyecto. Posteriormente se desarrolla la aplicación y se integran las medidas compatibles.

Tabla 13. Medidas de seguridad a utilizar en prototipo.

Lineamiento	Medidas de seguridad	Utilizada	Detalle de porque no se aplica en nuestro proyecto
Identificación	1. Ingresar usuario y contraseña a una base de datos	SI	
	2. Determinar permisos para tareas.	SI	
	3. Autentificar usuario	SI	
Protección	4. Solicitar prueba de ingreso de un humano.	SI	
	5. Determinar tiempo máximo de espera para ingresas clave	NO	En una app móvil no funciona el tiempo máximo de conexión, ya que esta entra en modo de hibernación, para continuar desde el punto que se quedó la última vez.
	6. Restringir ingreso si se falla en la clave y usuario varias ocasiones seguidas	SI	
	7. Administrar la información de acceso según usuario	SI	
	8. Incluir un botón de regresar	NO	Esta opción no se puede incluir en nuestro proyecto

Lineamiento	Medidas de seguridad	Utilizada	Detalle de porque no se aplica en nuestro proyecto
			debido a que los datos que se ingresan quedan grabados en la base de datos y solo se pueden eliminar pulsando la opción establecida para tal acción.
	9. Crear copias de seguridad para en caso de intromisión volver a un punto anterior de modificaciones	NO	Este rol pertenece al administrador de la base de datos y del servidor, pues como medida de seguridad, no se almacenan los datos en la aplicación.
	10. Incluir preguntas adicionales para usuario administrador	NO	En esta aplicación no tendría una función relevante, lo que haría es retrasar el tiempo de acceso a la información.
	11. Solicitar contraseña en caso de permanecer por más de 30 minutos dentro de la aplicación	NO	
	12. Incluir protección contra Cross-site	NO	El cross-site o llamada de dominio a dominio por ser una aplicación cliente - servidor, donde el cliente necesita llamar al dominio del servidor para poder ejecutar las tareas requeridas.
Detección	13. Incluir auditoría para revisar cambios no autorizados	NO	Esta es una actividad que se realiza una vez que la aplicación este siendo utilizada.
	14. Prevención de ataque de cookies	NO	En la aplicación desarrollada para evitar una ataque se almacenara la información estrictamente necesaria.
	15. Determinar visitantes a la aplicación	NO	No se aplica debido a que solo tienen acceso usuarios registrados, no acepta visitas.
	16. Ejecutar actualizaciones semanales de visitas	NO	No se aplica debido a que no admite visitas.
Respuesta	17. Expirar las sesiones	SI	

Lineamiento	Medidas de seguridad	Utilizada	Detalle de porque no se aplica en nuestro proyecto
	apropiadamente		
	18. Validar los datos de cookie y detectar alteraciones	NO	Al ser una aplicación móvil con usuarios registrados y para almacenar datos no se aplica.
	19. Enlistar anidados	SI	
Recuperación	20. Generar URL para recuperar cuenta	NO	No se aplica a nuestro prototipo puesto que las cuentas serán administradas por un usuario con permisos asignados.
	21. Crear botón automático para recuperar estados anteriores en las aplicaciones	NO	No se aplica porque la aplicación no guarda estados y cualquier inconveniente de la base de datos solo puede ser reparada por el administrador del servidor.
	22. Agregar validación entre clientes y formularios	SI	

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

CAPÍTULO V. IMPLEMENTACIÓN FRAMEWORK DE SEGURIDAD EN EL DESARROLLO DE APLICACIÓN

5.1. Desarrollo de aplicación.

Normas generales de la aplicación.

Funcionalidades

La aplicación desarrollada es de carácter empresarial, permite almacenar información relacionada con los pagos de facturas, para que el personal de administración pueda visualizar la fecha y forma de pago de las mismas, al instalar esta aplicación en un Smartphone e ingresar los datos la información estará disponible en todo momento.

Derechos propios y de terceros

Se debe proceder a registrar los derechos de autor de la aplicación para evitar plagios o reclamos posteriores así como revisar todos los elementos utilizados para evitar inconvenientes legales.

Licencia y condiciones de uso

La aplicación que se desarrolla es exclusivamente para la empresa y el gerente de la misma por tal su uso es estrictamente administrativo por lo que se deslinda al autor de la presente investigación de posibles malos usos por tanto el manual de licencias y usos de la aplicación se lo entregará al empresario.

Información y permisos

Para la aplicación el usuario mantendrá comunicación con el programa contable de la empresa para actualizar datos y mantener la información presente en su móvil para la toma de decisiones. La información solamente podrá ser requerida por el gerente de la empresa para su uso personal.

Informar al usuario

Se añadirá en la aplicación el icono “acerca de nosotros” donde se le informará a usuario todo cuanto se le pueda informar sobre los objetivos, manejo y recopilación de la información sí como el uso lícito que debe darse de la misma

Estándares

La aplicación se fundamentará en los estándares existentes que permitan determinase en los siguientes aspectos:

- ✓ **Desarrollo,** dentro del desarrollo de la aplicación se debe tomar en cuenta:
  - La aplicación debe ser compatible con la mayoría de dispositivos móviles con sistema Android.
  - Control de Versiones v.1.0.0. para mejorar la funcionalidad de la aplicación.
  - La aplicación permitirá en forma parcial cuando esté fuera de línea y no parará con el servicio.
- ✓ **Diseño,** se desarrolla una aplicación con una interfaz intuitiva, utilizando iconos claros, los mismos que se controlara para evitar problema de plagio.

5.2. Estrategia de desarrollo de aplicación.

Esquema de solución:

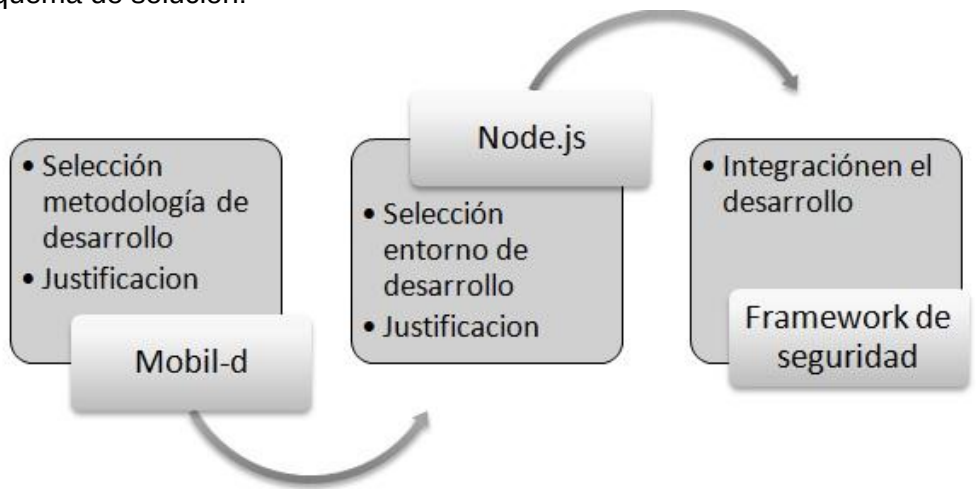


Figura 14. Esquema de desarrollo.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

En la figura 14 se detalla el esquema que se utilizara para desarrollar la aplicación, primero seleccionaremos la metodología de desarrollo, luego seleccionaremos el entorno de desarrollo y posteriormente integraremos el framework de seguridad en el desarrollo de la aplicación.

Metodología y justificación.

La metodología de desarrollo de aplicaciones móviles que se utiliza es Mobile-D, está basada en otras tecnologías como Rational Unified Process, Extreme Programming y Crystal Mehodologies. Mobile-D fue creada como parte del proyecto “ICARUS” en el 2004, su propósito es obtener pequeños ciclos de desarrollo de forma rápida, con pequeños grupos de personas.

La metodología Mobile-D se divide en cinco fases: exploración, iniciación, producto, estabilización y de pruebas. Las fases, a excepción de la primera están planificadas para tres días de desarrollo: planificación, trabajo y liberación. (Paco Blanco, 2009)

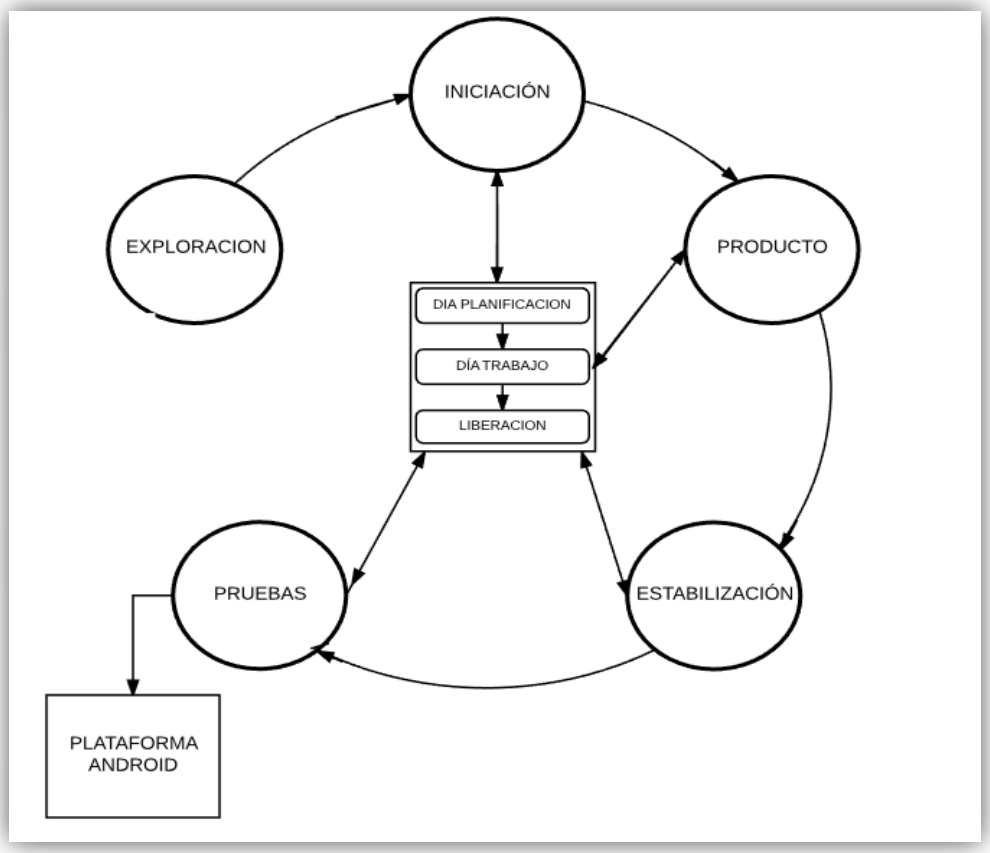


Figura 15. Diagrama Mobile-D
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

En la figura 15 se visualiza el ciclo de la metodología que se utiliza, el motivo por la cual se trabajara con esta metodología es porque está diseñada para pequeños grupos y la entrega de trabajos es de forma rápida, pues una vez acabado el ciclo se tendrá una aplicación ejecutable en el sistema Android.

Entorno de desarrollo.

El entorno que se utiliza es Node.Js, que es opensource, debido a que permite realizar diferentes configuraciones y dispone de gran cantidad de librerías, además este permite construir aplicaciones altamente escalables y dispone de módulos que permite facilitar el trabajo con MySQL y php.

Se utiliza XAMPP, que es un conjunto de aplicaciones software libre y por su facilidad para realizar configuraciones, de las cuales se utiliza apache como servidor y Mysql para administrar la base de datos.

La aplicación será desarrollada en lenguaje PHP, porque es un lenguaje de código abierto y es fácil de aprender para las personas con poca experiencia en el desarrollo.

5.3. Desarrollo de aplicación

5.3.1. Fase de exploración.

Esta fase permite la planificación y la definición de conceptos básicos del proyecto. Se indica el alcance del proyecto y el establecimiento de sus funcionalidades. Esta fase es importante, pues establece una base fundamental para controlar el desarrollo del software.
Los objetivos de esta fase son:

- Establecer los actores necesarios para el desarrollo de la aplicación.
- Recolección de requisitos para el desarrollo de la aplicación.
- Definición del alcance y funcionalidades.

Las entradas de la fase de exploración son:

- La propuesta de la aplicación.
- Las normas y restricciones a utilizar.
- Detalle de los requisitos iniciales.

Los entregables de esta fase son:

- Documento de plan de proyecto.
  En este documento se engloba de manera general la información relativa al proyecto con el fin de proporcionar un enfoque inicial para encaminar el desarrollo de la aplicación. En este documento se incluye el alcance que tendrá la aplicación, el equipo que se encargara de la llevarlo a cabo, los requerimientos iniciales y los

criterios para dar por concluida y aceptada la aplicación. El detalle del mismo lo puede encontrar en el ANEXO 2.

- Documento de especificación de requerimientos

  Este documento describe de forma completa cada uno de los requisitos necesarios que se necesitan para desarrollar la aplicación, el formato utilizado para el desarrollo del mismo es el definido por el estándar IEEE-830, este documento además proporciona datos concretos referentes al propósito, el público al que está destinada la aplicación, las perspectivas, funciones y características que tendrá la aplicación Facturas. Los requisitos que recoge este documento están clasificados en: funcionales, no funcionales, de interfaz de software, de interfaz de hardware, cada uno de los cuales describe las interacciones que realizara la aplicación. El detalle de este documento lo pueden encontrar en el ANEXO 3.

5.3.2. **Implementación de framework de seguridad en la fase de exploración.**

Una vez definida la fase de exploración y obtenidos los requisitos iniciales de la aplicación, como se indica en la figura 16, se integrara la primera fase del framework de seguridad.

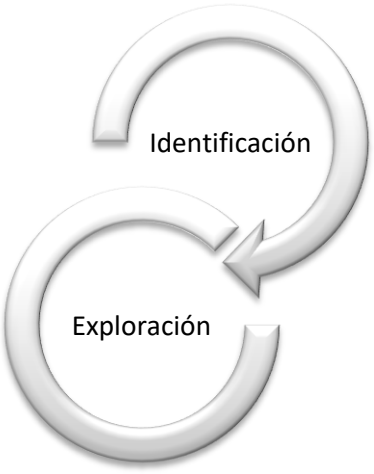


Figura 16. Integración de fases de identificación y exploración.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Una vez obtenidos los documentos de la fase de explotación, se procede a aplicar las fases del lineamiento de identificación:

Análisis de ambiente de negocios.

Esta aplicación en concreto está desarrollada para el uso de pequeñas y medianas empresas, prestando un servicio que ayude a controlar la gestión del pago de facturas. El entorno en el que utilizara la aplicación será la del área administrativa de una empresa y debido a los datos que se ingresen en la aplicación será necesario que solo el personal encargado de la contabilidad y gestión de pagos acceda a la aplicación.

Administración de accesos

Esta aplicación una vez instalada estima que no sobrepase de cinco usuarios principales y un administrador general y los cuales deben estar correctamente identificados.

Evaluación de riesgos

En la tabla 14 se indica los posibles riesgos que puedan afectar a la seguridad de la aplicación.

Tabla 14. Riesgos

Riesgo	Detalle
Acceso a la aplicación	Robo de datos de acceso a la aplicación.
Manipulación de datos.	No existe ningún tipo de restricción para la manipulación de datos existentes en aplicación,
Robo de datos.	Cualquier persona que pueda acceder a la aplicación podrá visualizar los datos y utilizarlos con fines distintos a los que se recolectaron.

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Medidas de seguridad que se aplicarán en el desarrollo de la aplicación.

- Para el acceso a la aplicación será necesario contar con usuario y contraseña (M1) [1]
- Se utilizara el algoritmo de encriptación MD5 para almacenar el usuario y contraseña. (M 3)[1]
- En la base de datos se configurara los permisos que se asignan a cada usuario. (M 2)[1]

[1] Estas medidas se presentan en la tabla 13.

5.3.3. **Fase de iniciación**

Esta fase está compuesta por tres días, uno para la planificación, uno para la configuración y el ultimo para la publicación de los resultados. El principal propósito es preparar y verificar las cuestiones fundamentales del desarrollo de la aplicación.
Los objetivos de esta fase son:

- Comprender los requisitos iniciales y de forma global todo el proyecto.
- Establecer los requisitos técnicos, físicos y humanos que se utilizara.

Las entradas de la fase de iniciación son:

- Documento de plan del proyecto
- Documento de requisitos iniciales

Los entregables de esta fase son:

- Documento con descripción del diseño
  El documento de descripción de diseño realiza una descripción detallada de la arquitectura de la aplicación facturas, en el cual se incluyen diagramas de: paquetes, despliegue, actividad, secuencia, los mismos que permiten representar de forma gráfica los componentes por la cual estará compuesta la aplicación, los procesos y comportamientos que realizara la misma. También se incluye el modelo de la base de datos que se utilizara para la aplicación. La descripción de todo este documento se encuentra en el ANEXO 5.
- Ilustración de requisitos mediante casos de uso.
- Funcionalidad implementada.

Ilustración de requisitos mediante casos de usos.

Los casos de uso ilustran los requerimientos del sistema al mostrar las respuestas que se dan a los eventos en el mismo, en la tabla 15 se indica el modelado de requerimientos, que es el paso previo para posteriormente representar los casos de uso, en donde se determinan los actores y los requisitos necesarios para cada una de las representaciones de los casos de uso de la aplicación facturas.

Tabla 15. Modelado de casos de uso.

Actores del Sistema

Nro.	Actor	Descripción
A1.	Administrador	Persona que asigna permisos
A2.	Usuario	Ingresa, modifica y borra los datos en el aplicación

Casos de uso

Nro.	Caso de Uso	Detalle	Req.(s)
CU1	Acceso a la aplicación	Ingreso y validación de usuario y contraseña.	REQ5
CU2	Registro de facturas	Ingreso datos de facturas.	REQ1/REQ3
CU3	Visualizar estado de pagos	Visualización en pantalla del estado de pagos.	REQ1/REQ3/REQ57REQ4/REQ5
CU4	Otorgar permisos de usuario.	El administrador asigna permisos a los usuarios según la necesidad.	REQ5/REQ6

Actor – Caso de Uso

Actor	Caso(s) de Uso	Observación
A2	CU2	Registro de datos de facturas.
A2	CU1	Dispone de un determinado número de intentos.
A1	CU4	Administrador es el encargado de asignar los permisos en la base de datos.
A2	CU3	Visualizara en pantalla los resultados, según los permisos que tenga asignados.

Fuente: Fundamentos de ingeniería de software.
Elaborado por: Ramos, Consuelo (2017)

Una vez realizado el modelado de los casos de uso, en la tabla 16 se ilustra el caso de uso de ingresos a la aplicación de usuarios, en donde se detalla de forma en que opera la aplicación al ingresar un usuario para operar dentro de la aplicación.

Tabla 16. Acceso a la aplicación.

Nombre:	Acceso a la aplicación	Requerimiento: REQ5
Descripción	Ingreso y validación de datos.	
Autor	Consuelo Ramos	
Fecha	19/05/17	
Actores	A1- Administrador. / A-Usuario	
Precondición:	Conexión al servidor Visualización de campos de ingreso	
Pos condición:	Visualizar la datos en pantalla.	
Flujo Normal		
Actor	**Sistema**	
1. Accede a la aplicación 3. Ingresa usuario 4. Ingresa contraseña. 7. Realiza alguna actividad 9. Finaliza y sale del sistema	2. Muestra los campos a rellenar 5. Valida datos ingresados 6. Sistema muestra el menú principal	
Flujo Alterno		
FA1. Sistema muestra mensaje "Datos erróneos", si los datos no son correctos.		
FA2.Sistema muestra mensaje "error al conectar al servidor" si no se dispone de conexión		
Escenarios		
1. Usuario: Aplicación		

Fuente: Fundamentos de ingeniería de software.
Elaborado por: Ramos, Consuelo (2017)

El caso de uso que se presenta en la tabla 17 representa el ingreso de datos de facturas, en donde se detalla el procedimiento que se realiza para de ingresar todos los datos de la factura que se requiere que este dentro de la aplicación.

Tabla 17. Caso de uso ingreso de datos de factura.

Nombre:	Ingreso datos de facturas.	Requerimiento: REQ1/REQ3
Descripción	Ingreso de datos de facturas	
Autor	Consuelo Ramos	
Fecha	19/05/17	
Actores	A-Usuario	
Precondición:	El usuario debe tener claves de acceso y permisos necesarios. Visualización de campos de ingreso	
Pos condición:	Conexión a la base de datos	
Flujo Normal		
Actor	**Sistema**	
1. Accede a la aplicación 3. Selecciona opción de menú 5. Ingresa datos 6. Selecciona opción guardar 9. Finaliza y sale del sistema	 2. Visualización de menú 4.Muestra los campos a rellenar 7. Almacena datos	
Flujo Alterno		
FA1. Sistema muestra mensaje "Datos erróneos", si los datos no son correctos. FA2.Sistema muestra mensaje "error al conectar al servidor" si no se dispone de conexión		
Escenarios		
1. Usuario: Aplicación		

Fuente: Fundamentos de ingeniería de software.
Elaborado por: Ramos, Consuelo (2017)

En la tabla 18 se ilustra el caso de uso de visualizar el estado de pagos de facturas, donde se detalla de forma en que opera la aplicación al momento de seleccionar uno de los criterios de búsqueda predefinidos en la aplicación.

Tabla 18. Visualizar estado de facturas.

Nombre:	Visualizar estados de facturas.	Requerimiento: REQ1/REQ3/REQ57REQ4/REQ5
Descripción	Seleccionar criterios de búsqueda	
Autor	Consuelo Ramos	
Fecha	19/05/17	
Actores	A1- Administrador. / A2-Usuario	
Precondición:	Conexión al servidor Visualización de parámetros Ser administrador de la aplicación.	
Pos condición:	Visualizar la datos en pantalla.	
Flujo Normal		
Actor	**Sistema**	
1. Accede a la aplicación		
	2. Muestra los criterios de búsqueda	
3. Selecciona los criterios		
4. Pulsa icono de búsqueda		
	5. Valida datos ingresados	
	6. Muestra los datos buscados	
7. Visualiza datos		
9. Finaliza y sale del sistema		
Flujo Alterno		
FA1. Sistema muestra mensaje "no existe datos", si no reúne los criterios		
FA2.Sistema muestra mensaje "error al conectar al servidor" si no se dispone de conexión		
Escenarios		
1. Usuario: Aplicación		

Fuente: Fundamentos de ingeniería de software.
Elaborado por: Ramos, Consuelo (2017)

En la tabla 19 se ilustra el caso de uso del proceso de asignación permisos de usuario, detallando como opera la aplicación cuando se desea asignar o modificar los permisos para determinados usuarios.

Tabla 19. Caso de uso otorgar permisos de usuario.

Nombre:	Otorgar de permisos del usuario.	Requerimiento: REQ5/REQ6
Descripción	Asignar permisos a los usuarios.	
Autor	Consuelo Ramos	
Fecha	19/05/17	
Actores	A1- Administrador. / A2-Usuario	
Precondición:	Conexión al servidor	
Pos condición:	Acceso a usuarios	
Flujo Normal		
Actor	**Sistema**	
1. Accede a la base de datos 3. Selecciona la opción de usuarios. 2.Asigna permisos preestablecidos a usuario determinado. 9. Finaliza y sale del sistema	2. Muestra menú 4. Almacena datos	
Flujo Alterno		
FA1.Sistema muestra mensaje "error al conectar al servidor" si no se dispone de conexión		
Escenarios		
1. Usuario: Aplicación		

Fuente: Fundamentos de ingeniería de software.
Elaborado por: Ramos, Consuelo (2017)

En la figura 17 se representa el diagrama de caso de uso de la interacción entre el usuario y la aplicación.

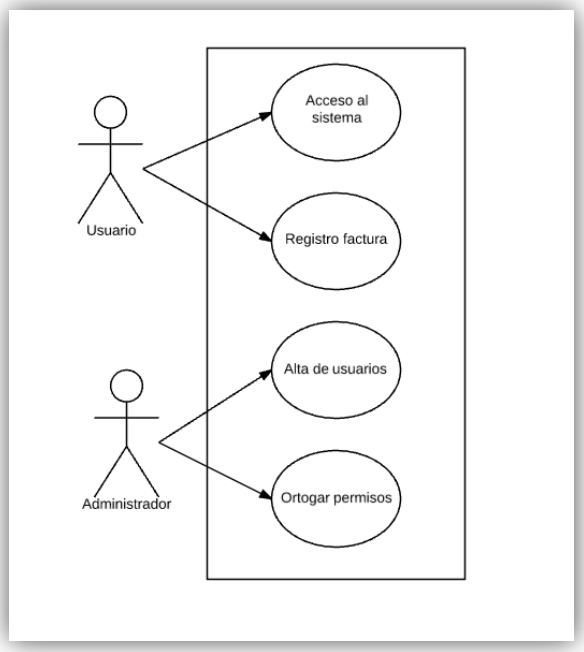


Figura 17. Diagrama caso de uso.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Funcionalidad implementada

Una de las primeras funciones que se van a realizar para empezar el desarrollo de la aplicación es la descarga de la aplicación XAMPP ,software que se utilizara como servidor y para el diseño y la configuración de la base de datos.

Se descarga y configura el servidor para posteriormente generar la base de datos, quedando esta función ya implementada. A continuación, en la imagen indicada en la figura 18, se presenta la captura de pantalla de la estructura de la base de datos.

Figura 18. Base de datos de aplicación facturas.
Fuente: Análisis de framework..
Elaborado por: Ramos, Consuelo (2017)

5.3.4. **Implementación de framework de seguridad en la fase de iniciación.**

Una vez realizada la fase de iniciación, donde se han realizado los documentos de la descripción del diseño, los diagramas de cados de uso, y también se ha creado la base de datos que se utilizara para la aplicación.

En la figura 19 se representa la integración de la segunda fase del framework de seguridad en la fase de iniciación.

Figura 19. Integración de fases de protección e iniciación
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

El lineamiento de protección consiste en realizar un del análisis de los requisitos indicados para la aplicación y determinar qué medidas de seguridad se emplearan para la gestión de sesiones y la gestión de permisos de usuarios. Para el desarrollo de la aplicación facturas se aplicaran las siguientes medidas de seguridad:

- Solicitar prueba de ingreso de un humano.
  A la hora de desarrollar la aplicación se tendrá en cuenta que se debe utilizar un código de verificación, para comprobar que la persona que accede al sistema es humano, con el objetivo de evitar que un robot realice un uso indebido de la aplicación. (M 4)[2]
- Evitar almacenar contraseñas.
  Las aplicación facturas no almacenara las contraseñas en el lado cliente a través de cookies.
- Bloqueo de acceso tras un determinado número de intentos de accesos fallidos.
  En el lado del servidor se utiliza un contador de ingreso de contraseñas, para que una vez superado el numero preestablecido se bloquee el acceso. Una vez bloqueado solo el administrador puede proporcionar los permisos nuevamente. (M 6)[2]
- Administrar la información de acceso de usuario.
  En esta aplicación existirá un solo administrador general de la aplicación, quien será el encargado de administrar los usuarios de la aplicación. (M 7)[2]

En la figura 20 que se presenta a continuación, se encuentra las capturas de pantallas de cada una de las tablas que compone la base de datos que se utiliza para la aplicación, en las cuales ya se han incluido los campos necesarios para el desarrollo de la aplicación.

[2] Estas medidas se presentan en la tabla 13

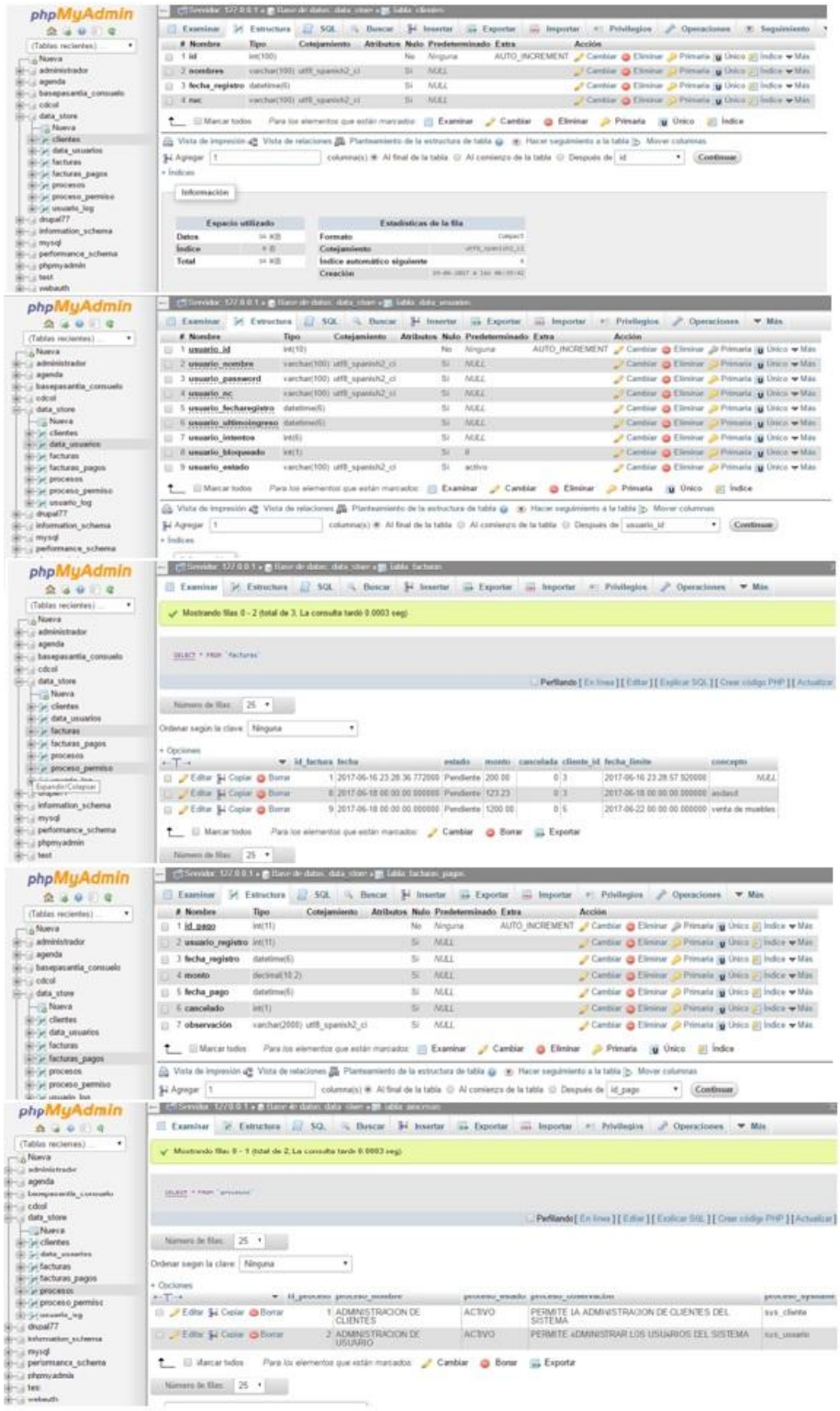


Figura 20. Tablas de base de datos de aplicación facturas.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

5.3.5. **Fase de producto.**

Antes de iniciar esta fase ya se ha instalado Node.js. y las librerías necesarias para generar el proyecto. El propósito de esta fase es implementar funcionalidades a la aplicación.
Los objetivos de esta fase son:

- Implementar funcionalidades indicadas en los requerimientos.
- Incluir las medidas de seguridad obtenidas de la implementación del framework de seguridad.
- Priorizar en la funcionalidad básica para permitir múltiples ciclos de mejora.

Las entradas de la fase de producto son:

- Funcionalidad implementada (creación de tablas de base de datos).
- Requisitos de seguridad obtenidos de la integración del framework de seguridad (identificación – exploración y protección – iniciación) que se aplicaran en el desarrollo de la aplicación.
- Síntesis de recursos gastados.
- Manuales, especificaciones de entorno de desarrollo.
- Material de apoyo.

Los entregables de esta fase son:

- Funcionalidad implementada.
- Medidas de seguridad implementadas.
- Documento de pruebas.
- Lista de problemas encontrados.

Función y medidas de seguridad implementadas

En esta fase se empieza el desarrollo de código de los procesos que se utilizaran para satisfacer los requisitos de la aplicación, además se incluye el código necesario para implementar las medidas de seguridad.

En la figura 21, se visualiza la carpeta con la estructura de las clases que se utiliza para crear la aplicación.

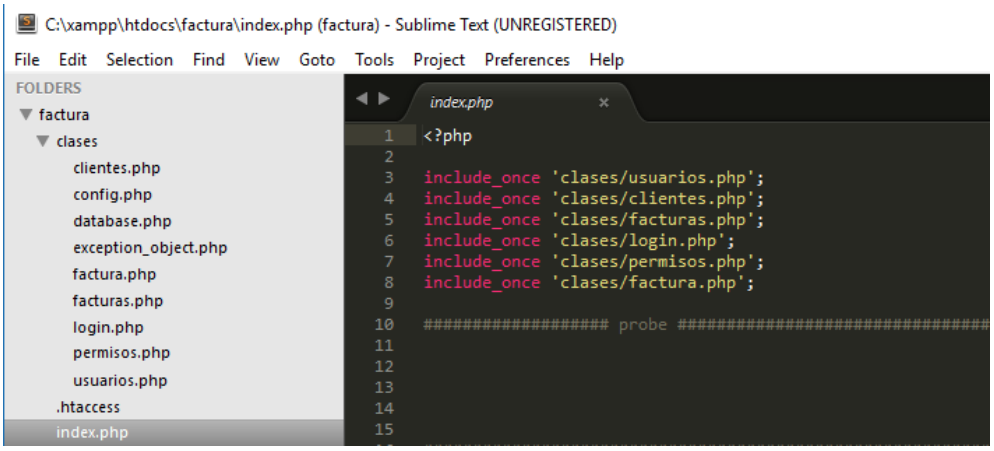


Figura 21. Carpeta con componentes de aplicación facturas.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Las medidas de seguridad resultantes de la integración del framework de seguridad que se implementan son:

Identificación – Exploración

- Para el acceso a la aplicación será necesario contar con usuario y contraseña, si el usuario o contraseña son muy cortos emitirá un mensaje informando al usuario. En la figura 22, que se presenta a continuación se visualiza el código empleado. (M 1)[3]

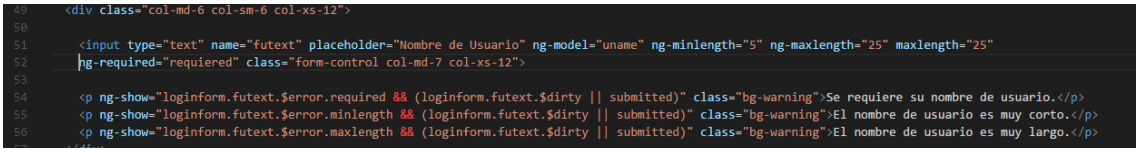


Figura 22. Fragmento de código de usuario y contraseña.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

- Se utilizara el algoritmo de encriptación MD5 para almacenar el usuario y contraseña. En la imagen 23 se indica el código empleado. (M 3)[3]

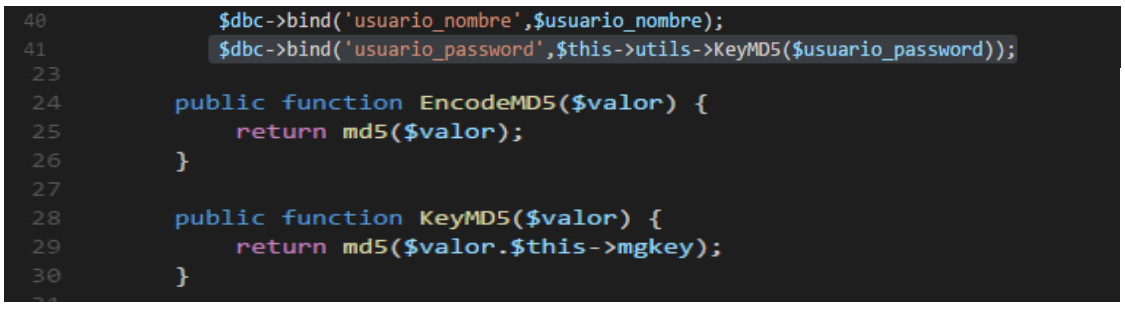


Figura 23. Fragmento de código MD5
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

[3] Código de implementación de medidas de seguridad de tabla 13.

- En la base de datos se configurara los permisos que se asignan a cada usuario.

Protección – Iniciación

- Se incluye un verificador de humanos para que se incluya al momento de ingresar a la aplicación. En la figura 24 se visualiza el código que permite realizar esta función(M4)[4].

```
$scope.genCode = function(){
try {
  var x = Math.floor((Math.random() * 8) + 1) + 49;
  var x1 = Math.floor((Math.random() * 23) + 1) + 64;
  var x2 = Math.floor((Math.random() * 23) + 1) + 64;
  var x3 = Math.floor((Math.random() * 23) + 1) + 64;
  var str = String.fromCharCode(x,x1,x2,x3);
  return str;
} catch (e) {
  return "3ADED"
} finally {

}
}
$scope.codigo = $scope.genCode();
```

Figura 24. Fragmento de código para verificación de humanos.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

- Se incluye como método de seguridad un bloqueo de acceso tras un determinado número de intentos de accesos fallidos. En la figura 25 se muestra el código empleado para tal función (M 6)[4]

```
#actualizando estado del usuario
if($lastid['contador']=='0'){
  $dbc = new Database();
  $dbc->query($this->updateintentos_usuario);
  $dbc->bind('nname',$usuario_nombre);
  $dbc->execute();


private $updateintentos_usuario = "update `data_usuarios` set `usuario_intentos` =  `usuario_intentos` + 1 ,`usuario_bloqueado` = case when
 (`usuario_intentos` + 1) >= 3 then 1 else 0 end where `usuario_nombre` = :nname;";
private $userestado = "select `usuario_intentos` as intentos,`usuario_bloqueado` as bloqueado from  data_usuarios where `usuario_nombre` = :nname;";
```

Figura 25. Fragmento de código controlar el acceso.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

[4] Código de implementación de medidas de seguridad de tabla 13.

Pruebas de aplicación.

El propósito de estas primeras pruebas es verificar si el desarrollo de la aplicación está siendo correcto, además de verificar el funcionamiento de cada uno de los procesos para continuar el desarrollo incremental. Para realizar las pruebas se tendrá en cuenta el nombre de la prueba, los criterios de aceptación y los resultados le las mismas.

En la tabla 20 se recoge las primeras pruebas realizadas.

Tabla 20. Pruebas de aplicación.

Prueba	Criterio	Resultado
Clase proveedores	Se visualiza los campos especificados.	Pendiente
	Permite añadir clientes	Correcto
	Se almacenan correctamente los datos.	Pendiente
Clase factura	Permite añadir facturas.	Correcto
	Muestra las opciones de guardar y cancelar	Correcto
	Se almacenan los datos de forma correcta	Correcto
	Se visualiza los proveedores almacenados.	Correcto
Clase facturas	Se visualiza el listado de las facturas almacenadas.	Correcto
	Dispone de criterios de búsqueda de facturas	Pendiente
	Presenta los resultados de las búsquedas realizadas.	Pendiente
	Permite editar el estado de las facturas.	Pendiente.
Clase usuarios	Permite añadir nuevos usuarios	Correcto
	Se visualiza correctamente los campos	Correcto
Clase permisos	Permite administrar permisos para los usuarios	Correcto
	Permite cambiar el estado de un usuario	Pendiente
Clase login	Solicita clave al acceder a la aplicación	Correcto
	Se encriptan las contraseñas	Correcto
	Se visualiza el campo de verificación de persona	Correcto
	Se reconocen los campos de usuario, clave y verificación.	Correcto

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Listado de errores

Los principales errores encontrados en el desarrollo de la aplicación y en el momento de realizar las pruebas se recogerán en la tabla 21. Los inconvenientes encontrados servirán para mejorar el desarrollo de la aplicación, pues permitirá obtener un funcionamiento óptimo.

Tabla 21. Listado de errores.

Nombre aplicación		Facturas	
Fase de pruebas		Fase producto	
Nº	**Área**	**Detalle**	**Solución**
1	Codificación	Los campos de formularios no se adaptan a la pantalla.	Se procede a distribuir mejor los campos.
2	Codificación	Los datos no están almacenando de forma correcta.	Se revisó el código para corregir y verificar el correcto funcionamiento.
3	Diseño y codificación	No se visualizan todos los criterios de búsqueda	Se revisa los requerimientos y se incluye los criterios de búsqueda necesarios.
4	Codificación	No se visualizan de forma correcta las búsquedas.	Se codifica nuevamente para presentar las búsquedas requeridas.
5	Codificación	No se puede cambiar el estado de pago de una factura.	Se modifica el código para solucionar ese problema.
6	Codificación	Edición de permisos de usuario.	Se corrige verificando el código para que aparezcan los campos necesarios para edición de usuarios.

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

5.3.6. **Implementación de framework de seguridad en la fase de producto.**

En la fase de producto se ha implementado funciones y se han incluido las normas de seguridad que se pueden aplicar en el desarrollo de esta aplicación, para consecutivamente realizar las pruebas de los errores existentes.

En la figura 26 se representa la integración de la tercera fase del framework de seguridad en la fase de producto.

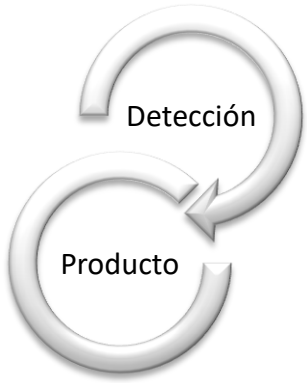


Figura 26. Integración de fases detección y producto.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

- Mapeo

  En esta fase se identifica las funciones principales de la aplicación, para poder detectar cualquier intromisión, en la figura 27 se visualiza el código con las principales funciones de la aplicación. [5]

```
.config(function($routeProvider,$locationProvider) {
  $locationProvider.hashPrefix('');
  $routeProvider.when('/', {templateUrl:'home.html',  controller: 'loginCtrl' , reloadOnSearch: false})
  .when('/config', {templateUrl:'configapp.html',  controller: 'configCtrl' , reloadOnSearch: false})
  .when('/facturas', {templateUrl:'facturas.html',  controller: 'facturasCtrl' , reloadOnSearch: false})
  .when('/factura/:id', {controller:'facturaCtrl',templateUrl:'factura.html', reloadOnSearch:false })
  .when('/cliente/:id', {controller:'clienteCtrl',templateUrl:'cliente.html', reloadOnSearch:false })
  .when('/clientes', {controller:'clientesCtrl',templateUrl:'clientes.html', reloadOnSearch:false })
  .when('/usuario/:id', {controller:'usuarioCtrl',templateUrl:'usuario.html', reloadOnSearch:false })
  .when('/usuarios', {controller:'usuariosCtrl',templateUrl:'usuarios.html', reloadOnSearch:false })
  .when('/usuariocambio/:id', {controller:'usuarioscambioCtrl',templateUrl:'usuariocontrasena.html', reloadOnSearch:false })
  .when('/usuariopermisos/:id', {controller:'usuariopermisosCtrl',templateUrl:'permisos.html', reloadOnSearch:false })
  .when('/pagos/:id', {controller:'pagosfactCtrl',templateUrl:'pagosfactura.html', reloadOnSearch:false })
  .when('/pago/:id', {controller:'pagofactCtrl',templateUrl:'pagofactura.html', reloadOnSearch:false });

});
```

Figura 27. Mapeo de funciones.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017

- Medidas de seguridad a aplicar en esta fase:

  Una vez concluido el desarrollo de la aplicación se realizara una auditoria de forma mensual para revisar si existen cambios y si estos están o no autorizados.

[5] Ejecución de las fases del lineamiento de detección.

5.3.7. **Fase de estabilización.**

El propósito de esta fase es integrar todos los módulos y comprobar el correcto funcionamiento de la aplicación.
Los objetivos de esta fase son:

- Finalización de la aplicación.
- Finalizar la documentación.
- Mejorar y garantizar la calidad de la aplicación

Las entradas de la fase de producto son:

- Funcionalidad implementada
- Artefactos de desarrollo relacionado.

Los entregables de esta fase son:

- Funcionalidad implementada de la aplicación.
- Documentación del producto finalizado.

Funcionalidad implementada de la aplicación.

Una vez desarrollados y probados de forma individual los procesos que componen la aplicación, se procede a integrar y probar en el navegador para verificar el funcionamiento.

A continuación se presenta las capturas de pantalla del funcionamiento tota de la aplicación.

1. Iniciar el servidor virtual para la aplicación.
   Iniciamos el símbolo de sistema (cmd), tecleamos la ubicación del proyecto y luego la palabra gulp. En la figura 28 se visualiza el proceso detallado anteriormente.

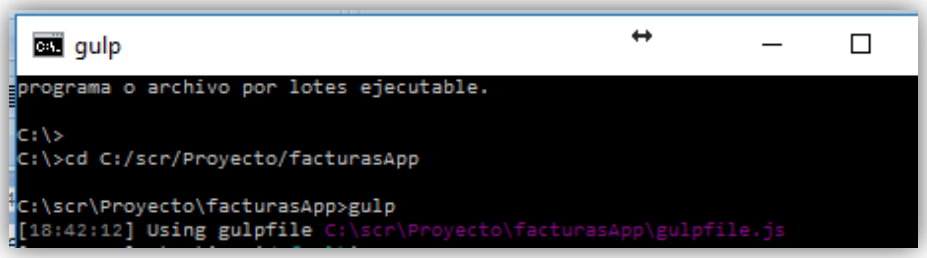


Figura 28. Iniciar servidor virtual.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

2. Configuración del servidor en la aplicación.

En el navegador se ingresa localhost:8000 y se visualizara la pantalla principal de la aplicación. En la figura 29 se visualiza la pantalla principal de la aplicación.

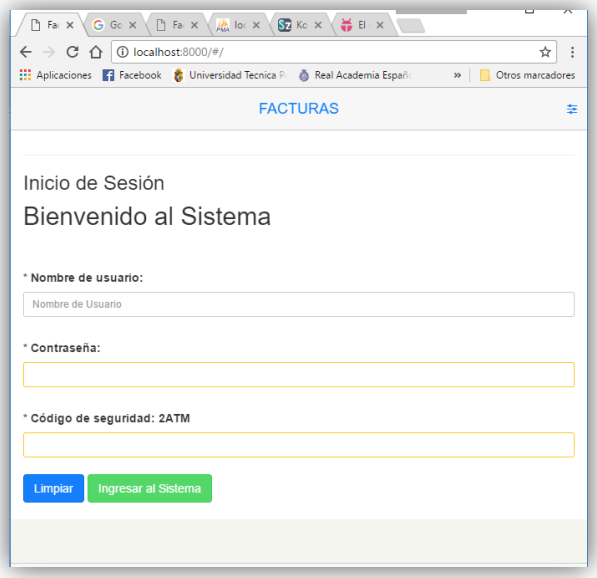


Figura 29. Pantalla inicial de aplicación facturas.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

3. En el lado superior derecho se encuentra el icono que permite la conexión de la aplicación con el servidor, en el campo servidor se ingresa la dirección del ordenador que tiene el servidor Mysql. En la figura 30 se visualiza la pantalla donde deberá ingresar la dirección del servidor.

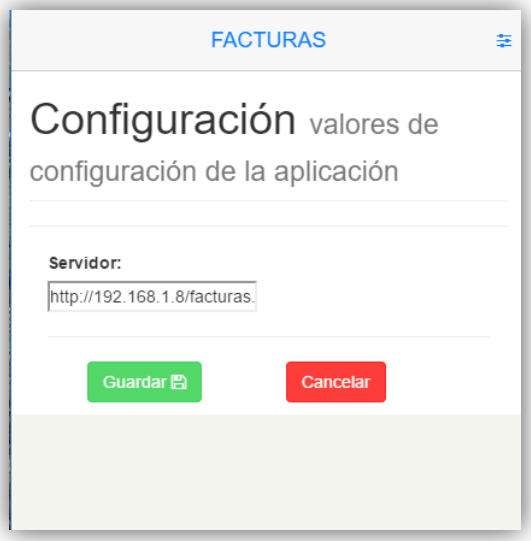


Figura 30. Pantalla de configuración de servidor.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

4. Una vez conectado al servidor se ingresa el usuario, contraseña y el código de verificación. En la figura 31 se visualiza la pantalla de la aplicación de inicio de sesión.

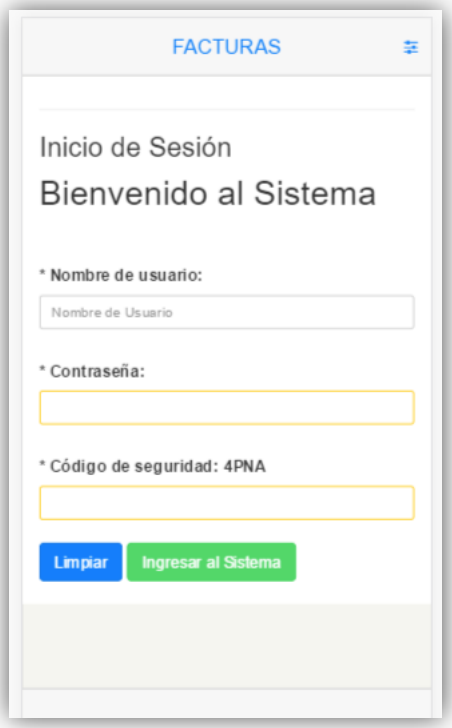


Figura 31. Pantalla de inicio de sesión.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

5. Una vez accedido al sistema se puede visualizar el menú y consecutivamente elegir la opción con la cual se vaya a operar. En la figura 32 se visualiza la pantalla de bienvenida y la pantalla que muestra el menú principal de la aplicación, en el menú principal se presentan tres opciones: facturas, proveedores y usuarios, además de la opción de cierre de sesión, permitiendo realizar el cierre de forma apropiada. (M17)[1]

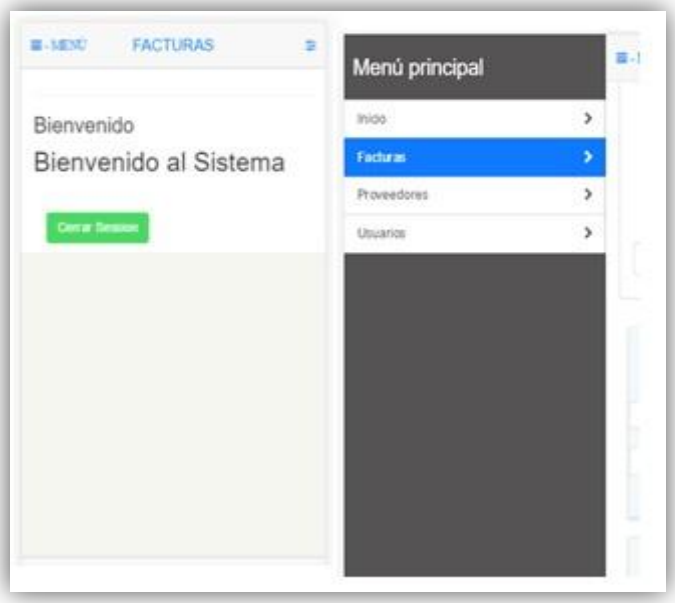


Figura 32. Pantalla de inicio de bienvenida y de menú principal.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

6. Una vez seleccionada la opción de facturas, se visualiza la pantalla con el nombre de ¨facturas¨, en el cual se puede visualizar el listado de las facturas ingresadas y seleccionar criterios de búsqueda, en la misma pantalla, al pulsar la opción añadir se visualiza otra pantalla con nombre de ¨factura¨, que permite ingresar los datos de las facturas que se almacenaran en la base de datos. En la figura 33 se muestra la captura de las dos pantallas antes descritas.

Figura 33. Pantalla de facturas.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

7. El menú de usuarios permite visualizar los usuarios incluidos, agregar usuarios y editar permisos de los mismos. En la figura 34 se visualiza la captura de pantalla de las opciones antes mencionadas.

Figura 34. Pantalla de usuarios.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

8. Cuando se selecciona la opción de proveedores, se visualiza el listado de los proveedores almacenados en la base de datos, además que permite añadir nuevos proveedores. En la figura 35 se visualiza la captura de pantalla del menú de proveedores.

Figura 35. Pantalla de proveedores.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Documentación de producto finalizado.

A continuación se presenta un mapa de la documentación del producto, donde se organiza en tres secciones, en la figura 36 se indica toda la información organizada.

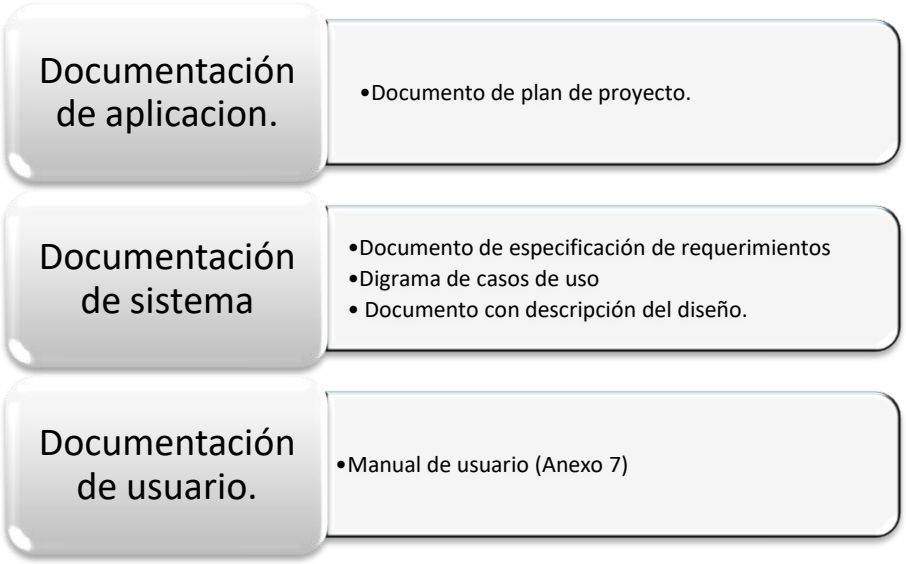


Figura 36. Mapa de documentación.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

5.3.8. Implementación de framework de seguridad en la fase de estabilización.

En la fase de estabilización se ha integrado todo el proyecto y se han comprobado que los módulos funcionen de forma correcta.

En la figura 37 se representa la integración de la cuarta fase del framework de seguridad en la fase de estabilización.

Figura 37. Integración de las fases de respuesta y estabilización.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

En la integración de esta fase se procura aplicar medidas que seguridad que ayuden a mitigar los riesgos inherentes de la aplicación. Como medida de seguridad a implementar en esta fase es enlistar las funciones anidadas, de forma que en caso de surgir alguna contingencia, la información esté disponible de forma rápida. para en la aplicación facturas solo existe la clase Índex que anida que a las otras existentes, en la figura 38 se visualiza lo antes expuesto (M19) [6]

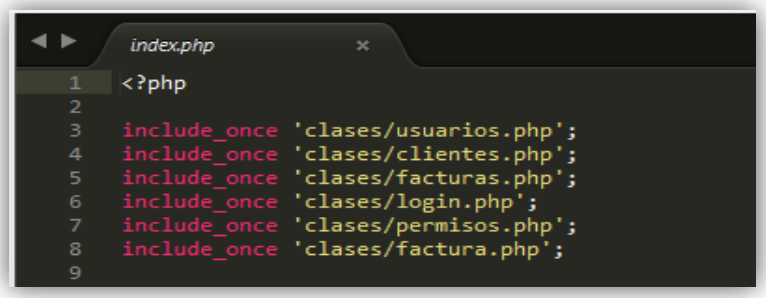


Figura 38. Clases anidadas.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

También como medida de seguridad se ha incluido la opción de cierre de sesión, de forma que permita finalizar la aplicación de forma correcta.

5.3.9. **Fase de pruebas.**

Una vez concluidas las fases de desarrollo anteriores y cumplidos los requisitos establecidos, en esta fase se realizara un testeo hasta la versión estable, si existen errores se reparan, pero no se desarrollara nada nuevo.

Los objetivos de esta fase son:

- Realizar pruebas de la aplicación basadas en las documentación producida en el proyecto.
- Proveer información de defectos encontrados.
- Obtener una aplicación que cumpla lo esperado y libre de errores.

Las entradas de la fase de producto son:

- Funcionalidad implementada
- Funcionalidades de usuarios definidas.
- Descripción de interfaz de usuario que se utilizara para realizar las pruebas.

[6] Implementación de medidas de seguridad de tabla 13

Los entregables de esta fase son:

- Documento de pruebas, el detalle de este documento se presenta en el ANEXO 4.
- Versión estable de aplicación.

Versión estable de aplicación.

Después de obtener la versión estable de la aplicación se procede a instalar en el dispositivo móvil. En la figura 39 se visualiza una captura de pantalla donde se encuentra el icono de la aplicación instalada con el nombre “Facturas”.

Figura 39. Instalación de aplicación en el smartphone.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

5.3.10. **Implementación de framework de seguridad en la fase de pruebas.**

En la fase de pruebas se ha obtenido la versión estable de la aplicación y se ha comprobado que los módulos funcionen de forma correcta.

En la figura 40 se representa la integración de la quinta fase del framework de seguridad en la fase de pruebas.

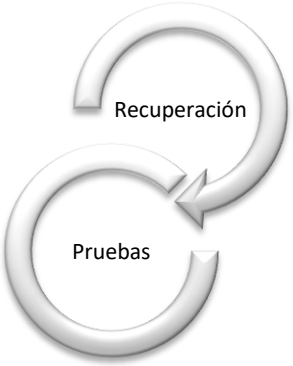


Figura 40. Integración de fase de recuperación y pruebas.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Esta fase propone medidas para que la información almacenada en la base de datos utilizada por la aplicación, en caso de sufrir una pérdida de datos debido a una falla de seguridad o mala manipulación de datos esta se pueda recuperar.
Entre las medidas de seguridad que se pueden aplicar en esta aplicación son:

- Plan de recuperación

  Este plan permitirá garantizar la capacidad de recuperación de los datos manipulados por la aplicación y permitirá que la empresa recobre su actividad laboral lo antes posible, el detalle de este documento se encuentra el ANEXO 6
- Estrategia de comunicaciones

  El plan de comunicaciones permite transmitir información en el momento en que se necesite realizar determinadas acciones. El plan de comunicación debe ser sugerido a la empresa que utilizara la aplicación y deberá contener los siguientes puntos: objetivo principal del plan, táctica de comunicación, plantilla que permita recoger y organizar el plan de acción. El detalle de este documento se encuentra en el ANEXO 7.
- Realizar copias de seguridad en una unidad que permita salvaguardar la información.

6. Métrica.

Métrica de integridad.

En el presente trabajo se utilizara la métrica de integridad para valorar la seguridad de la aplicación creada, puesto que la integridad es la característica que permite determinar la capacidad de la aplicación para proteger el acceso no autorizado a la misma y la protección para evitar que los datos de la misma pueden ser modificados sin autorización o robados. La integridad es muy importante puesto que un fallo de seguridad de la aplicación puede afectar económicamente a la empresa.

Para medir la integridad se tomaran en cuenta dos atributos:

- De amenaza, que son las probabilidades de que un ataque sucede.
- De seguridad, probabilidad de repeler una amenaza.

La fórmula utilizara el porcentaje estimado para cada atributo y empleara la fórmula establecida para valorar la habilidad de la aplicación para resistir ataques.

Para considerar el rango de la integridad de la aplicación se tendrá en cuenta las escalas de aceptación presentados en la tabla 22.

Tabla 22. Métrica de Integridad.

Escala de aceptación.	
Inaceptablemente bajo	0,1 - 0,20
Bajo	0,21 - 0,40
Medio	0,41 - 0,50
Elevado	0,51 - 0,75
Muy elevado	0,76 - 1

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

En la tabla 23 se analiza los atributos referentes a la integridad de la aplicación factura y se aplica la fórmula para determinar integridad de la misma.

Tabla 23. Aplicación de métrica de integridad.

Amenaza	Probabilidad de ataque	Seguridad	Posibilidad de resistir	Integridad =1-(amenazas(1-seguridad))
Acceso a aplicación	0,25	Acceso con usuario y clave	0,80	0,95
Manipulación de datos por robots	0,3	Verificación de acceso de persona humana	0,85	0,96
Actividades no autorizadas en la base de datos.	0,35	Determinar privilegios para manipulación de base de datos.	0,80	0,93
Divulgación de datos y acceso no autorizado	0,4	Determinar privilegios para manipulación de base de datos.	0,70	0,88
Robo de contraseñas	0,2	Encriptar usuario y clave	0,80	0,96
Ataque de fuerza bruta	0,35	Control de bloqueo activado, tras varios intentos de acceso con contraseña incorrecta.	0,80	0,93
Inyección SQL	0,4	Utilización de librería PDO de php para realizar la conexión y enviar los parámetros	0,75	0,90
Consultas de SQL	0,4	Consultas predefinidas.	0,85	0,94

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Una vez analizados y valorados los cada uno de los atributos para determinar la integridad de la aplicación se obtiene un promedio de 0,93, por lo tanto se puede decir que la aplicación desarrollada, según la escala de aceptación indicada en la tabla 31, la integridad de la aplicación es muy elevada.

7. Resumen de integración de fases y detalle de medidas de seguridad aplicadas.

En la tabla 24 se detalla las medidas de seguridad utilizadas en la integración de las fases de la metodología Mobile-D y el framework de seguridad, además se especifica las fases de los lineamientos de seguridad utilizados para la creación del prototipo de Facturas.

Tabla 24. Integración de fases, medidas de seguridad y fases de lineamiento aplicadas.

Integración de fases	Medidas de seguridad	Detalle de aplicación de medidas y fases de lineamientos.
Implementación de fases de identificación en fase de exploración.	1. Ingresar usuario y contraseña a una base de datos 2. Determinar permisos para tareas. 3. Autentificar usuario	En esta implementación se analiza los requisitos de la aplicación para considerar los riesgos existentes y determinar qué medidas se utilizaran y como se aplicarán.
Implementación de fases de protección en fase de iniciación	4. Solicitar prueba de ingreso de un humano. 8. Restringir ingreso si se falla en la clave y usuario varias ocasiones seguidas 9. Administrar la información de acceso según usuario	En esta implementación se aplica las medidas que servirán para proteger la aplicación. La primera funcionalidad de la aplicación es la creación de base de datos, en el cual se administra el acceso a la información almacenada en la misma. El resto de medidas se aplica posteriormente.
Implementación de fases de detección en fase de producto		Las fases del lineamiento presentan el mapa de las principales funciones de la aplicación y recomienda a la empresa beneficiaria a realizar una auditoría, pues debe realizarse cuando ya tenga almacenados los datos.
Implementación de fases de respuesta en fase de estabilización.	17. Expirar las sesiones apropiadamente 19. Enlistar anidados	Esta integración comprueba que el código desarrollado incluya medidas que ayuden a mitigar los riesgos inherentes a la aplicación. Para expirar las sesiones de forma adecuada se incluye la opción que permita cerrar la sesión de forma apropiada y la documentación de la aplicación especifica cuáles son las funciones anidadas.
Implementación de fases de recuperación en fase de pruebas.	23. Agregar validación entre clientes y formularios.	En esta implementación se desarrolla las fases de lineamiento de recuperación, lo que permitirá una vez terminada la aplicación obtener medios para restablecer el funcionamiento de la aplicación y se comprueba que la validación de usuarios funcione correctamente.

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

CONCLUSIONES

- Las medidas de seguridad que utilizan los framework analizados no se presentan de forma explícita, lo que dificulta determinar cuales ofrecen mayor seguridad a las aplicaciones para dispositivos móviles desarrolladas.
- El uso de metodologías especificas para el proceso de desarrollo de aplicaciones móviles, como Mobile-D, facilita la inclusión de elementos del framework de seguridad minimizando los riesgos asociados.
- Utilizar medidas de seguridad en la creación de la aplicaciones, permite en mayor medida atenuar los riesgos de seguridad informática, pues una vez que el usuario final empieza a utilizar la aplicación no se puede contrastar el impacto que tendrán los ataques. En este trabajo las medidas de seguridad utilizadas para autentificar usuarios para acceder a la aplicación y la utilización de consultas predefinidas y asignación de permisos en la base de datos, nos permitió ofrecer una aplicación segura, con lo cual el usuario final puede utilizarla con un alto porcentaje de confianza.
- El framework desarrollado reúne y organiza según los lineamientos las medidas de seguridad obtenidas del análisis realizado, proporcionado al desarrollador una utilidad esencial para establecer los medios que permitan evitar ataques o que la aplicación sea mínimamente vulnerable ante problemas de seguridad.
- La utilización de la métrica de integridad, para analizar la seguridad de la aplicación ha permitido valorar los atributos y las medidas de seguridad empleadas en el desarrollo de la aplicación, facilitando valorar los defectos y mejorar los mismos, para que la aplicación creada cumpla con los objetivos esperados.
- La aplicación desarrollada, una vez analizada con la métrica de integridad, ha obtenido un promedio alto de integridad, de igual forma, todas las pruebas realizadas de las funcionalidades que tiene la aplicación se cumplieron satisfactoriamente, por lo cual puede ser utilizada por cualquier empresa.
- En general, todo el trabajo realizado ha permitido afianzar los conocimientos obtenidos a lo largo de la carrera, pues se ha realizado investigaciones, utilizado una metodología, desarrollado planificación de proyecto, desarrollado de documentos de requisitos y de diseño, codificación de aplicación, lo cual permite demostrar que se está en capacidad de realizar investigaciones de un tema concreto y da seguridad para enfrentar un trabajo profesional.

RECOMENDACIONES.

- Realizar una investigar a fondo de los conceptos que se tratan en cada apartado de los trabajos a realizar, leer y suscribirse a foros o blogs de cada uno de los temas, así mismo se debe considerar las recomendaciones que se manifiestan en dichas comunidades, pues nos ayudará en momentos determinados del desarrollo de una aplicación.
- En el desarrollo de la aplicaciones se debe utilizar lenguajes donde curva de aprendizaje sea baja y además posea un gran número de comunidades lo cual facilita encontrar documentación en internet, como es el caso de PHP que fue utilizado en el presente trabajo.
- Es recomendable utilizar una metodología para crear cualquier aplicación de software, debido a que al desarrollar la documentación de diseño y arquitectura de la aplicación se va ilustrando los pasos a seguir, facilitando el desarrollo incremental de la aplicación, además de conseguir obtener la documentación de todo el proyecto de forma organizada, lo que permitirá realizar revisiones futuras o en caso de surgir contingencias, igualmente servirá para transmitir los conocimientos en caso de cambio de programador.
- La utilización de un framework de seguridad desde el inicio del desarrollo de un proyecto permitirá definir la estructura y considerar la seguridad desde el punto cero, pues al obtener los requisitos de la aplicación, se pueden considerar los riesgos de seguridad inherentes a determinados requisitos de la futura aplicación y determinar que posibles medidas de seguridad se pueden aplicar en el desarrollo.
- Los framework poseen un núcleo estructurado y ordenado, lo que permite al desarrollador seguir un línea de desarrollo, colocando todo de forma ordenada, lo que permite que la creación de software sea rápida y segura, pues cuando la aplicación crezca se sabrá donde está situado cada uno de los componentes.
- Otro de los puntos por la cual es recomendable utilizar un framework de seguridad, es la mantenibilidad del código, pues en caso de sufrir una vulnerabilidad de seguridad o realizar un cambio significativo se podrá encontrar cada módulo y componente de forma clara y ordenada, lo cual facilitara realizar las mejoras necesarias para que la aplicación funcione de forma óptima.

BIBLIOGRAFÍA

[1]. (s.f.).

[2]. (Diciembre de 2013). *ReCIBE, 2*(3), 1-17.

[3]. Alarcon, J. M. (05 de 17 de 2009). *Seguridad Web: Evitar ataques XSS con Cookies accesibles solo desde el servidor.* Recuperado el 2017 de 06 de 18, de www.jasoft.org: http://www.jasoft.org/Blog/post/PermaLinkaspxguid=dcc42726-707a-45e4-a2bc-52b535a.aspx

[4]. Alcalde, A. (14 de Agosto de 2016). *El baúl del programador*. Recuperado el 2016, de Los 15 Mejores Frameworks gratuitos para Aplicaciones Web/Móvil [Actualizado]: https://elbauldelprogramador.com/los-10-mejores-frameworks-gratis-de-aplicaciones-web/

[5]. Aquilino, J., Izquiero, R., Cueva, J., López, B., & Joynes, L. (s.f.). Sistema de Métricas para tiempo Real en Aplicaciones Java.

[6]. Bolaños, S., González, R., Medina, V., & Barón, J. (Junio de 2014). Conceptual framework language – CFL –. *Dyna, 81* (185), 124-131.

[7]. Bolaños, S., Medina, V., & Aguilar, J. (2009). Principios para la Fromalización de la Ingeniería de Software. *Ingeniería*, 31-37.

[8]. Booch, G. R. (2005). *The Unified Modeling Language User Guide.* Addison-Wesley.

[9]. Cantillo, C., Roura, M., & Sánchez, A. (Junio de 2012). Tendencias actuales en el uso de dispositivos móviles en educación. *La Educación*(147), 1-21.

[10]. Carrera, E. (2012). El Costo de la Seguridad en Disposotivos Móviles. *EIDOS*, 1-7.

[11]. Castellaro, M., Romaniz, S., Ramos, J., & Pessolani, P. (s.f.). Hacia la Ingeniería de Software Seguro. 1-10.

[12]. Chebyshev, V., & Unuchek, R. (24 de Febrero de 2014). *Amenazas para dispositivos móviles en 2013*. Obtenido de Secure list: https://securelist.lat/analysis/boletin-de-seguridad-de-kaspersky/72491/amenazas-para-dispositivos-mviles-en-2013/

[13]. Ekuni, R., Vaz, L., & Amodeo Bueno, O. F. (2011). Levels of processing: the evolution of a framework. *Psychology & Neuroscience, 4*(3), 333-339.

[14]. EL COMERCIO. (6 de Noviembre de 2014). Más zonas Wi-Fi gratuitas en las ciudades del Ecuador. *EL COMERCIO*.

[15]. Esaú, A. (14 de Junio de 2016). *OpenWebinars*. Obtenido de Ionic Framework, ventajas y desventajas: https://openwebinars.net/blog/ionic-framework-ventajas-desventajas/

[16]. Gamma, E., Helm, R., Johnson, R., & Vlisides, J. (1995). *Design Patterns.* Addison-Wesley.

[17]. Gasca, M., Camargo, L., & Medina, B. (Abril-Junio de 2014). Metodología para el desarrollo de aplicaciones móviles. *Tecnura, 18*(40), 20-35.

[18]. Gasca, M., Camargo, L., & Medina, B. (Abril - Junio, de 2014). Metodología para el desarrollo de aplicaciones móviles. *Tecnura, 18*(40), 20 - 35 .

[19]. GitHub, Inc. (2017). Obtenido de https://github.com/argob/estandares/blob/master/estandares-apps.md

[20]. Google. (s.f.). *Google Analytics Solutions*. Recuperado el 18 de 06 de 2017, de Tag Manager: https://support.google.com/tagmanager/answer/6103696?hl=es

[21]. Gutiérrez, J. (s.f.). *¿Qué es un framework web?* Recuperado el 12 de septiembre de 2016, de http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf

[22]. Hatton, R. (2005). *Swt: A Developer's Notebook.* O'Reilly & Associates.

[23]. Heller, E. (2007). *Psicología del Color.* Gustavo Gili.

[24]. INEC. (2013). *Encuesta de Tecnologías de la Información y la Comunicación.*

[25]. Jarillo, J., & Martínez, J. (1991). The international expansion of Spanish firms: towards an integrative framework for international strategy. *Corporate and industry strategies for Europe*, 282-302.

[26]. Karron. (14 de Abril de 2013). *WordPress*. Obtenido de https://karron10.wordpress.com/2013/04/14/normas-y-estandares-en-proyectos-de-ti-2/

[27]. Leiva, I., & Villalobos, M. (2015). Método ágil híbrido para desarrollar software en dispositivos móviles. *Ingeniare. Revista Chilena de Ingeniería, 23*(3), 473-488.

[28]. Macia, N., Lanfranco, E., & Venosa, P. (2014). Seguridad en dispositivos móviles: un enfoque práctico. *Workshop de Investigadores en Ciencias de la Computación*, 837-841.

[29]. Maeda, J. (2005). *Las Leyes de la Simplicidad.* Gedisa S.A.

[30]. Martínez, A. (16 de Septiembre de 2014). *Desarrollo seguro de aplicaciones para dispositivos móviles*. Obtenido de Certesi: https://www.certsi.es/blog/desarrollo-seguro-de-aplicaciones-para-dispositivos-moviles

[31]. Martínez, A. (2014). *INCIBE*. Obtenido de Volver Desarrollo seguro de aplicaciones para dispositivos móviles.: https://www.incibe.es/blogs/post/Seguridad/BlogSeguridad/Articulo_y_comentarios/desarroll

[32]. Marulanda, C., & Ceballos, J. (2012). UNA REVISIÓN DE METODOLOGÍAS SEGURAS EN CADA FASE DEL CICLO DE VIDA DEL DESARROLLO DE SOFTWARE . *Ing. USBMed*, 1-8.

[33]. Meyer, B. (1999). *Construcción de Software Orientado a Objetos.* Prentice Hall.

[34]. Montiel, J., Hernández, E., & López, J. (Diciembre de 2012). Computación Móvil. *Ingeniare. Revista Chilena de Ingeniería, 20*(3), 282-283.

[35]. Morin, E. (2005). *Introducción al Pensamiento Complejo.* Gedisa S.A.

[36]. ONU. (22 de Junio de 2016). La ONU advierte de fallos de seguridad en teléfonos móviles. *ABC*.

[37]. Paco Blanco, J. C. (2009). Metodología de desarrollo ágil para sistemas móviles Introducción al desarrollo con Android y el iPhone. 1-30. (D. e. 2009, Ed.)

[38]. Pimienta, R., Aguilar, G., Ramírez, M., & Gallegos, G. (noviembre de 2014). Métodos de programación segura en Java para aplicaciones móviles en Android. *Ciencia Ergo Sum, 21*(3), 243-248.

[39]. Ramírez, G. (s.f.). La seguridad en aplicaciones móviles: estrategias en el mundo actual. *Escuela de Ciencias Básicas Tecnología e Ingeniería*, 1-17.

[40]. Recio, F., & Provencio, D. (10 de Diciembre de 2003). *desarrollo net.com*. Obtenido de Net Framework: http://www.desarrolloweb.com/articulos/1328.php

[41]. Rezende, S. (Abril - Junio de 2003). Internationalisation Processes: an Analytical Framework. *RAC - Revista de Administração Contemporânea, 7*(2), 137-156.

[42]. Rodriguez, M. (2003). Definición de una arquitectura para teléfonos móviles. *IBM*.

[43]. Rosen, K. (2004). *Matemática Discreta.* Mc Graw Hill.

[44]. Software Assurance. (18 de Mayo de 2012). Software Assurance Pocket Guide Series. *Development, Volume V – Version 2.0, May 18, 2012*, 1-37.

[45]. Soria, I., & Cóndor, E. (2015). *Compendio de Ingenieria de software.*

[46]. Soto, M. (2 de Octubre de 2014). *Tecnobitt*. Obtenido de Legalidad en appsmóviles : prácticas y aspectos legales a tener en cuenta: http://tecnobitt.com/legalidad-en-apps-moviles/

[47]. Stallings, W. (2010). *Cryptography and Network Security: Principles and Prac tice.* New York: Prentice Hall.

[48]. Teufel, B. S. (1995). *Compiladores conceptos fundamentales.* Addison: Wesley.

[49]. Tucker, A., & Noonan, R. (2002). *Programming Languages Principles and Paradigms.* McGraw Hill.

[50]. UNIDEP. (10 de 2011). *Derecho*. Obtenido de https://derecho1.files.wordpress.com/2011/10/la-norma-concepto-caracterc3adsticas-y-clasificacic3b3n.pdf

[51]. Zamora, J. (11 de Octubre de 2015). *El androide libre*. Obtenido de Los mejores framewirk oara desarrolladores Android: http://www.elandroidelibre.com/2015/10/los-mejores-frameworks-para-desarrolladores-android.html

ANEXOS

Anexo 1

1. Actividades del SDL para la seguridad.

Actividades del SDL para la Seguridad	
1. Entrenamiento	**5. Verificación**
Entrenamiento de seguridad básica	Análisis dinámico
2. Requerimientos	Fuzz Testing
Establecer requerimientos de seguridad	Revisión de la superficie de ataques
Crear umbrales de calidad y límites de errores	**6. Lanzamiento**
Evaluación de los riesgos de seguridad y privacidad	Plan de respuesta a incidentes
3. Diseño	Revisión de seguridad final
Establecer requerimientos de diseño	Aprobar y archivar lanzamiento
Análisis de la superficie de ataques	**7. Respuesta**
Modelado de amenazas	Ejecutar el plan de respuesta a incidentes
4. Implementación	
Utilizar herramientas aprobadas	
Prohibir funciones no seguras	
Análisis estático	

Figura 41. Actividades SDL para la seguridad
Fuente: (Leiva & Villalobos, 2015)
Elaborado por: Ramos, Consuelo (2016)

2. Conceptual framework lenguage

Para (Bolaños, González, Medina, & Barón, 2014) en su estudio exponen que el Framework "es un modelo de lenguaje enfocado en la abstracción y contextualización de conocimiento. Cuando se está comunicando una idea, este lenguaje se puede usar de dos maneras, oral y escrito"

Contextualización

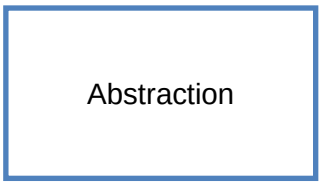


Figura 42. Lenguaje conceptual framework
Fuente: (Bolaños, González, Medina, & Barón, 2014)
Elaborado por: Ramos. Consuelo (2016)

El Lenguaje Conceptual framework es un lenguaje capaz de capturar en una imagen una rigurosa expresión representada en una estructura sintáctica. Este lenguaje tiene el poder de convertir un lenguaje formal con esquematización simbólica en dos objetivos principales; la abstracción y la contextualización. La abstracción es un mecanismo que filtra las características esenciales utilizando modelos cognitivos como: paradigmas, valoraciones, principios y comportamientos. Utiliza la introspección como mecanismo de tratar al

conocimiento de manera individual. La contextualización usa la observación para responder a un fenómeno externo, usa la inmersión como mecanismo para explicar al fenómeno a través de estructuras externas.

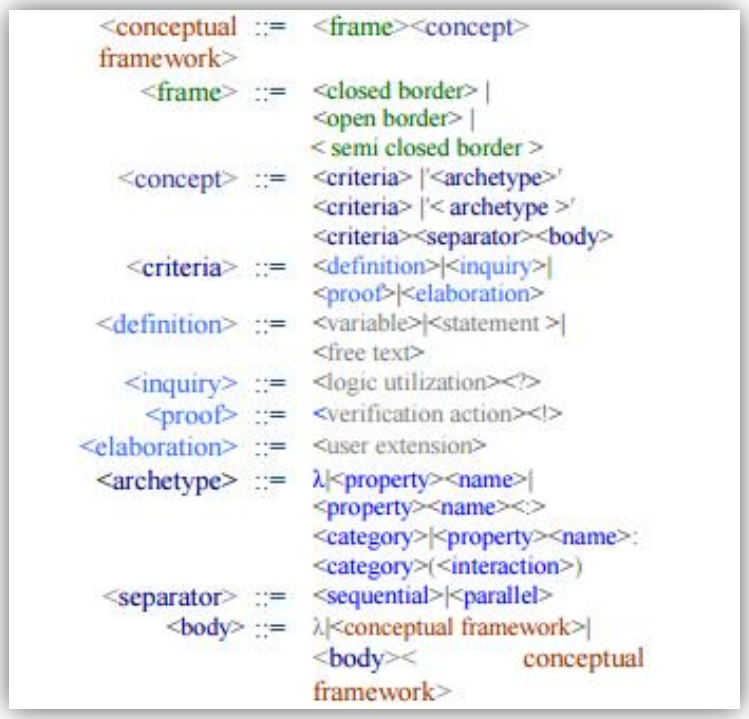
Por tal razón la abstracción y la contextualización son la base del Lenguaje Conceptual framework tanto de manera individual como colectiva. Algunos paradigmas como la estructuración, la orientación a objetos incluyen modelos declarativos basados en la abstracción, dejando de lado el potencial de la inmersión útil en la solución de modelos. (Tucker & Noonan, 2002)

2.1. Escritura en Lenguaje Conceptual framework

Para la formalización del Lenguaje Conceptual framework es primordial el desarrollo de gramática, derivación o producción y lenguaje.
La función está dada por:
G = (V, T, S, P)
Consiste de un vocabulario **V,** un subconjunto **T** de **V,** y el símbolo **S** de **V –T** y **P** de producción. **V-T** se denota por **N.** Los elementos de **N** son llamados elementos no terminales. **V** representa el alfabeto que es un conjunto finito y una palabra en **V** es un subconjunto de **V** elementos. Finalmente el lenguaje generado por **G** se denota como **L(G)** y el conjunto de elementos terminales se remplaza con **S** obteniendo la ecuación:
$(\boldsymbol{G}) = \{\boldsymbol{w} \in \boldsymbol{T}* \mid \boldsymbol{s}* \Rightarrow \boldsymbol{w}\}$ (Rosen, 2004)

2.2. Producción de Lenguaje Conceptual framework en la forma backus nur (BNF)

La BNF fue inicialmente creada para definir la estructura sintáctica de programación de lenguaje. (Teufel, 1995). La CFL sigue la siguiente estructura sintáctica:



El CFL como un lenguaje está estructurado como sigue:

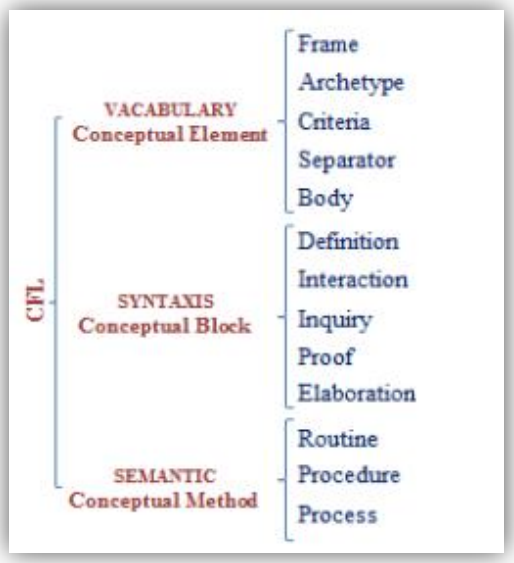


Figura 43. Estructura del CFL como lenguaje
Fuente: (Bolaños, González, Medina, & Barón, 2014)
Elaborado por: Ramos, Consuelo (2016)

2.3.Vocabulario (elementos conceptuales)

Está conformado por dos elementos principales que son: Frame (marco), que separa un específico concepto del universo y su contextualización, y Concepto que muestra el dominio

del conocimiento extraído del universo, lo componen el arquetipo, criterios, separadores y cuerpo.

A continuación se expone ilustraciones referentes al tema:

Figura 44. Frame (Marco)
Fuente: (Bolaños, González, Medina, & Barón, 2014)
Elaborado por: Ramos, Consuelo (2016)

El Marco muestra la relación del universo en su contexto y la relación con la frontera de integración de manera específica.

Figura 45. Concepto
Fuente: (Bolaños, González, Medina, & Barón, 2014)
Elaborado por: Ramos, Consuelo (2016)

El concepto como tal que extrae el conocimiento de la manera más concreta posible sobre el universo y define sus funciones y partes.

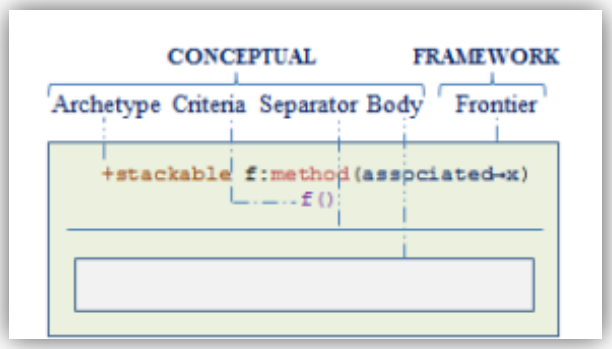


Figura 46. Concepto- Marco
Fuente: (Bolaños, González, Medina, & Barón, 2014)
Elaborado por: Ramos, Consuelo (2016)

El CFL se muestra en su relación de integración de las bases conceptuales con la frontera del FRAMEWORK a través de la asociación y métodos.

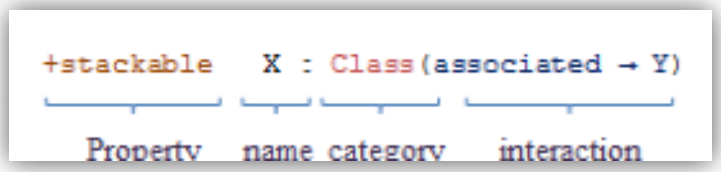


Figura 47. Arquetipo
Fuente: (Bolaños, González, Medina, & Barón, 2014)
Elaborado por: Ramos, Consuelo (2016)

El Arquetipo nos enseña la relación e interacción del Framework escrito en forma de función.

Sintaxis del Lenguaje Conceptual framework

La sintaxis del CFL se basa en un concepto de seguridad y está formado por definiciones, consultas, interacciones, investigaciones, pruebas y elaboraciones.
Las definiciones constituyen cualquier lenguaje de programación con un concepto de seguridad del CFL, el cual tiene tres tipos de definición: variables, declaraciones y anotaciones. Las variables se utilizan para formar contenidos de información; las declaraciones son usadas para proponer invocaciones, respuestas y frases generales; y finalmente las anotaciones son utilizadas para la documentación.
Las interacciones permiten administrar de manera interna y externa para comunicar las condiciones deseadas para la aplicación del programa.

Las investigaciones son presentadas como un modelo de formulación de interrogantes acerca de variable que pueden tener, esta investigación puede ser directo o recursivo. Es directa cuando se la puede manejar para tomar una dirección u otra, y es recursivo cuando se puede tomar diversas opciones varias veces.
Las pruebas permiten tener escenario s con diferentes resultados debido a variables que s e pueden presentar en un tiempo determinado, así pues se enmarca en lo referente a la experimentación, demostración, discusión entre otras.
La Elaboración permite desarrollar el lenguaje a través de premisas, operaciones determinadas y conclusiones obtenidas. (Bolaños, González, Medina, & Barón, 2014)

A continuación se esquematiza la Sintaxis del CFL:

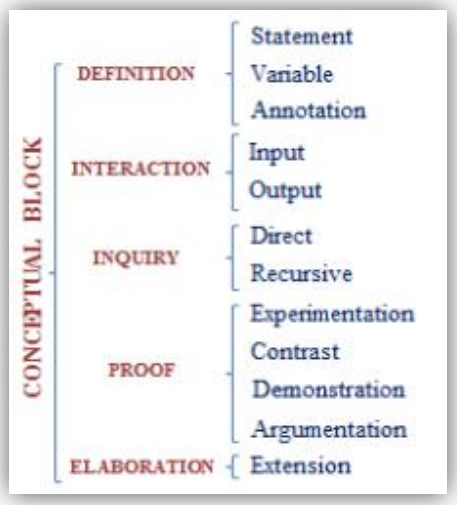


Figura 48. Sintaxis del CFL
Fuente: (Bolaños, González, Medina, & Barón, 2014)
Elaborado por: Ramos, Consuelo (2016)

Semántica de Lenguaje Conceptual framework

La semántica del CFL se refiere a la configuración del Lenguaje Conceptual, con esta semántica se representa la solución de un problema. El método conceptual forma un módulo que depende del intercambio de la información que pude llegar a ser un **proceso** y si recibe y produce información en el contexto; un **procedimiento** si se recibe y produce información para el contexto; y **rutina** si no recibe ni produce información para el contexto. (Meyer, 1999)

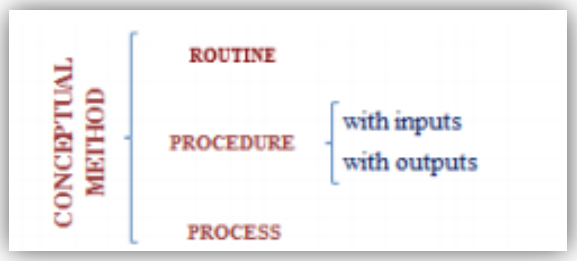


Figura 49. Método conceptual
Fuente: (Bolaños, González, Medina, & Barón, 2014)
Elaborado por: Ramos, Consuelo (2016)

Esquema frente al diagrama del Lenguaje Conceptual framework

El CFL produce esquemas, diferentes diagramas, esquemas de relaciones implícitas a través del orden y anidamiento de los marcos. Las composiciones relacionales están simuladas por una secuencia horizontal, las relaciones de herencia y realización están

dadas por una secuencia vertical entre los métodos conceptuales, si los métodos tienen alta categoría éstos se localizan por la parte superior, y si tiene una subclase es localizada abajo. (Gamma, Helm, Johnson, & Vlisides, 1995)

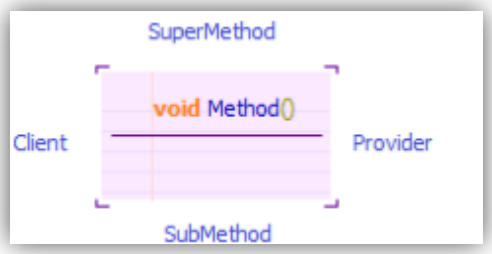


Figura 50. Esquema frente al diagrama
Fuente: (Gamma, Helm, Johnson, & Vlisides, 1995)
Elaborado por: Ramos, Consuelo (2016)

Códigos de colores en CFL

El CFL para sus esquemas usa un código de colores que sirve para realzar el significado. Los códigos de colores para el CFL son los siguientes:

- Definición usa el color verde que representa verdad, tranquilidad y desarrollo.
- Interacción usa el color anaranjado que representa sorpresa, transformación y socialización.
- Investigación utiliza el color azul que representa ciencia, idealismo y funcionalidad.
- Las Pruebas que usan el color amarillo que representa advertencia y creatividad.
- La Elaboración usa el color rojo que representa prohibición, peligro y dinamismo.

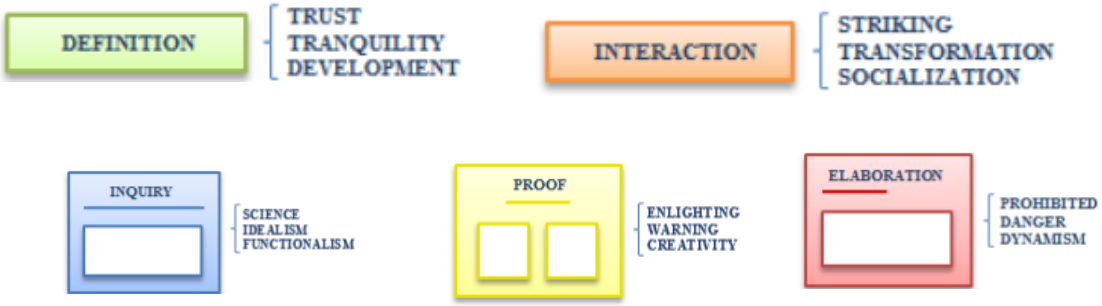


Figura 51. Código de colores del CFL
Fuente: (Heller, 2007)
Elaborado por: Ramos, Consuelo (2016)

Comparación del CFL con otros lenguajes

El CFL es un lenguaje muy peculiar dentro de la informática y permite la expresión de similares modelos de orientación a objetos al igual que los lenguajes UML y JAVA. (Booch, 2005)

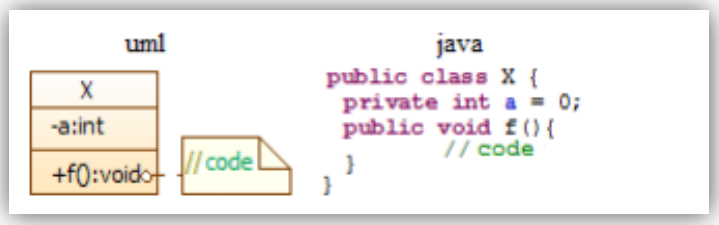


Figura 52. Clase en UML y JAVA
Fuente: (Heller, 2007)
Elaborado por: Ramos, Consuelo (2016)

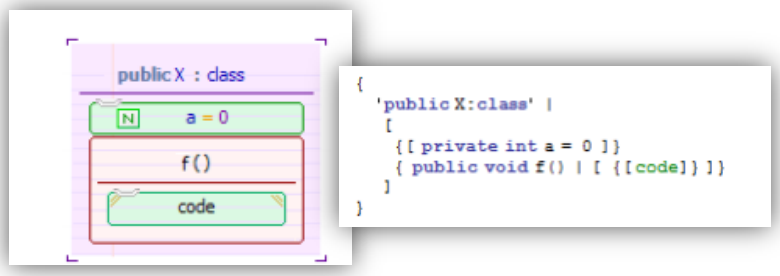


Figura 53. Categoría en CFL
Fuente: (Heller, 2007)
Elaborado por: Ramos, Consuelo (2016)

Principios del CFL

El CFL tiene tres principios fundamentales que son: simplicidad, robusto y completo. Simple puesto que reduce trabajo gracias a que abstrae los niveles más altos de programación y lo configura de acuerdo a las necesidades del modelo además es automático simplificando la línea de aprendizaje lo que marca diferencia con otros modelos de lenguaje de programación (Maeda, 2005). Robusto porque mantiene información que puede ser verificable y validable. Completo pues se base en documentación de la información y mensurable (Morin, 2005).

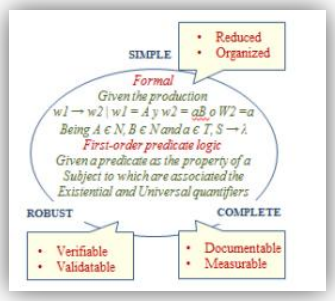


Figura 54. Principios de CFL
Fuente: (Maeda, 2005).
Elaborado por: Ramos, Consuelo (2016)

Métrica del CFL

La métrica en un software permite desarrollar el control del mismo, el CFL tiene dos métricas y son: balance de incertidumbre y densidad algorítmica.

Atributo/Metodología	CbyC	SDL
Utilización de métodos formales	X	-
Utiliza desarrollo iterativo	X	-
Entrenamiento en temas de seguridad	-	X
Establece requerimientos de seguridad	X	X
Estudia la trazabilidad de los requerimientos	X	-
Asistencia de personal especializado en la seguridad	-	X
Realiza modelado de amenazas	-	X
Análisis estático	X	X
Análisis dinámico	-	X
Fuzz testing	-	X
Revisión de código	X	X
Desarrolla un plan en caso de incidentes de seguridad	-	X

Figura 55. Comparación de CbyC y SDL
Fuente: (Leiva & Villalobos, 2015)
Elaborado por: Ramos, Consuelo (2016)

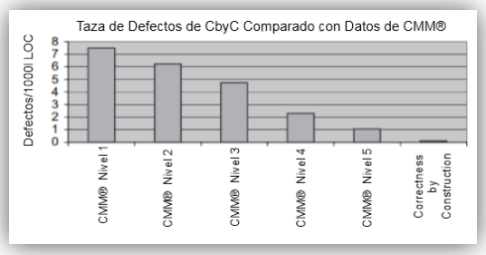


Figura 56. Taza de defectos de CbyC comparado con datos CMM
Fuente: Hall, 2007
Elaborado por: Ramos, Consuelo (2016)

Anexo 2

DOCUMENTO DE PLAN DE PROYECTO.

1. Nombre aplicación

Facturación

2. Introducción

La aplicación que se desarrollara está pensada para empresas y autónomos, para dar solución a los pagos de las facturas que se realizan de distintas formas. El proceso de pago es una actividad importante en la empresa, y debido a los errores cometidos a la hora de realizar el pago por el mal almacenamiento de los datos de las facturas, esta aplicación almacenara todos los datos y formas de pago de los proveedores.

3. Alcance del trabajo

Este proyecto realizara la captura, análisis, especificación y validación de requerimientos de software para el desarrollo de la aplicación, los aspectos que pretende solucionar son:

- Proveedor: el proveedor emite una factura por los servicios prestados.
- Área de administración: cuando el personal de administración recibe la factura procede a registrar en las cuentas contables.
- Forma de pago: una vez registrada la factura se debe tener en cuenta la fecha y forma de pago.

4. Equipo del proyecto

- Desarrollador de la aplicación
- Persona administrativo de una empresa.

5. Requerimientos para el proyecto

Captura de requerimientos:

- Definición del propósito del sistema a desarrollar.
- Identificamos las fuentes para obtener los requerimientos.
- Se realizarán entrevistas y reuniones con el personal de la empresa para identificar las actividades actuales y necesidades.
- Utilizaremos varios métodos y técnicas para capturar los requerimientos.
- Se documentaran los requerimientos

Análisis de requerimientos:

- Descomposición de los requisitos
- Agrupar y clasificar los requisitos.
- Aprobación de los usuarios la lista de requisitos.

- Negociación de prioridades.

Especificación de requerimientos:

- Elaborar los documentos de definición de sistema, documento de requisitos de sistema y documento de requisitos de software, para que puedan ser analizados y revisados.

Validación de requerimientos:

- Se revisara los documentos de los requisitos para asegurar que se define el software correctamente.

6. Cierre del proyecto:

Este proyecto finalizara con la entrega de los siguientes documentos:

- Documento de especificación de software.
- Arquitectura de la solución.
- Entrega de aplicación
- Recomendaciones.
- Informe final de trabajo y presentación del mismo.

7. Criterios de aceptación

El presente proyecto será aceptado cuando cumpla con los siguientes requisitos:

- Cumpla con los requisitos establecidos.
- Se tenga el consentimiento de las partes interesadas.

Anexo 3

DOCUMENTO DE ESPECIFICACIÓN DE REQUERIMIENTOS DE APLICACIÓN FACTURAS.

1. Introducción

1.1. Propósito

Este documento versión 1.0 recoge las necesidades empresariales que pretende satisfacer esta aplicación para prestar un servicio de almacenamiento de las fechas de pago de las facturas y de esta forma comprobar de una forma rápida y sencilla los vencimientos de las mismas. La especificación de estos requisitos permitirá crear un diseño que se ajuste expuesto.

1.2. Convenciones del documento

El presente documento está redactado en base a la plantilla del estándar IEEE-830.

1.3. Público objetivo y sugerencias de lectura

Este documento está dirigido a:

- Personal administrativo.
- Personas encargadas de desarrollar el sistema.

1.4. Alcance

La aplicación móvil , es un software de gestión que permitirá realizar lo siguiente:

- Ingresar nombre del proveedor.
- Ingresar número de identificación
- Ingresar la fecha de pago
- Ingresar el importe a pagar
- Seleccionar la forma de pago.
- Seleccionar el estado del pago.
- Almacenar estos datos en una base de datos.
- Emitir informes de los pagos.

1.5. Referencias

- Documento de visiona miento y problemática.

2. Descripción general

2.1. Perspectiva del producto

La aplicación “Facturas” funcionará una vez instala en un Smartphone con sistema android y almacenara todos los datos registrados en una base de datos creada especialmente para la aplicación.

2.2. Funciones del producto

La aplicación Facturas, de almacenamiento de datos de facturas, constará de las funcionalidades:

- Ingreso de datos.
  - Ingreso de nombre, apellido o nombre comercial de un proveedor.
  - Ingreso de numero de identidad.
- Seleccionar forma de pago.
  - Seleccionar forma de pago entre las opciones disponibles.
- Almacenar datos.
  - Almacenara los datos en una base de datos.

2.3. Clases y características de usuario

Los usuarios que intervienen son:

- Personal administrativo: personas que registran los datos y consultan la base de datos.
- Proveedores: empresas y personas que emiten una factura.
- Gerente de la empresa: encargado de utilizar y administrar la aplicación.

2.4. Ambiente de operación

La aplicación será ejecutable en Smartphone con sistema android.

2.5. Restricciones de diseño e implementación

- La aplicación deberá adaptarse a los requisitos establecidos.
- Utilizará una base de datos para almacenar los datos.
- El lenguaje utilizado para desarrollar la aplicación será PHP**.**

2.6. Documentación de usuario

Al ser una aplicación móvil, toda la capacitación y ayuda necesaria se realizara vía online.

2.7. Suposiciones y dependencias

Suposiciones.

- La aplicación debe ser ejecutable en el sistema android para que los clientes la puedan utilizar.
- Las personas que puedan acceder a manipular los datos almacenados contaran con los permisos necesarios.
- La aplicación apache como servidor y Mysql, para la base de datos, todos ellos incluidos en XAMPP.

3. Características de la aplicación para almacenar datos de facturas.

Aquí se describen los requisitos relativos al desarrollo de la aplicación. Todos los requisitos se identifican unívocamente mediante un código, siendo este el que será utilizado como referencia cada vez que sea necesario mencionarlo a lo largo del ciclo de vida del proyecto.

3.1. Ingreso de datos.

3.1.1. Descripción y prioridades

Cuando la factura de un proveedor es emitida y posteriormente receptada por una empresa, el personal administrativo ingresa a la aplicación para que realice el ingreso de los datos de la factura y seleccione la forma y fecha de pago. Este proceso tiene una prioridad alta, pues es en esta fase es donde se almacenan los datos más importantes de los clientes.

Los requisitos que se recogen van a cubrir las siguientes necesidades:

1. Adecuación: Capacidad de la aplicación para realizar las tareas y objetivos de usuario especificados.

2. Exactitud: Capacidad de la aplicación para proporcionar los resultados acordados, con el grado necesario de precisión.

3. Interoperabilidad: Capacidad de la aplicación para interactuar con el sistema seleccionado.

3.1.2. Requerimientos funcionales

- **REQ-FUN-01 Registro de datos de proveedores.**

  Entrada: El ingreso de datos consiste en proporcionar un formulario, en donde el usuario pueda cumplimentar los siguientes campos:

  - Nombre.
  - Número de identificación

- Numero de factura
- Concepto
- Fecha de factura
- Fecha de pago.
- Monto
- Estado
- Generar efectos.
- Selecciona el método de pago de las opciones presentadas.

Proceso:

- La aplicación comprueba si el número de identificación ya está registrado en la base de datos.
- Si no existe registro, procede a almacenar los datos ingresados en la base de datos.

Salida:

- Si el número de identificación se encuentra en la base de datos, se presentara un mensaje en el que se indicara que ya existe ese registro.
- Aplicación indicara un mensaje donde indique que el registro se realizó correctamente.

- **REQ-FUN-02. Modificar o eliminar estados de pago.**

Entrada:

El usuario introduce el número de identificación.

Proceso:

- La aplicación visualiza el registro correspondiente al número indicado.
- El usuario selecciona la opción de modificar o eliminar.

Salida:

- Si el número de identidad existe, mostrara un mensaje en pantalla indicando el error.
- Después de ejecutada la opción seleccionada, se visualizara un mensaje indicando la acción ejecutada.

- **REQ-FUN-03. Seleccionar método de pago.**

Entrada:

El usuario selecciona de un menú despegable las opciones de método de pago.

Proceso:

La aplicación tiene almacenadas las formas de pago disponibles, efectivo y crédito,

Salida:

Los datos seleccionados se almacenaran en la base de datos.

- **REQ-FUN-04. Visualizar informes**

Entrada:

El usuario selecciona uno de los listados que permite generar la aplicación.

Proceso:

La aplicación presentara en la pantalla los criterios para visualizar un listado.

Salida:

Se visualizara en la pantalla del dispositivo el listado de las facturas, según el criterio seleccionado.

- **REQ-FUN-05. Utilizar usuario y contraseña para acceder a la aplicación.**

Entrada:

El usuario debe tener usuario y contraseña para acceder a la aplicación.

Proceso:

- La aplicación presentara los campos para ingresar el usuario y contraseña
- Superado el número de intentos se bloqueara.

Salida:

- Si los datos son correctos se visualizara en la pantalla del dispositivo el menú.
- Si los datos no son correctos se indicara un mensaje de error.

- **REQ-FUN-06. Asignación de permisos para determinadas funciones de la aplicación**.

Entrada:

El administrador asignara en la base de datos los permisos a cada usuario.

Proceso:

- La aplicación presentara los campos para ingresar el usuario y contraseña
- El usuario selecciona la actividad

Salida:

- Si el usuario dispone de los permisos se accederá a la actividad seleccionada.
- Si no dispone de permisos se indicara un mensaje de error.

3.1.3. Requerimientos no funcionales.

REQ-NO-FUN-01 El tipo o clase de base de datos para la recolección de información es MySQL.

REQ-NO-FUN-02 La aplicación podrá albergar información de más de 5000 facturas, los cuales se almacenara en MySQL.

4. Requerimientos de interfaz externa

4.1. Interfaz de usuario

REQ-NO-FUN-03 La interfaz debe ser intuitiva y adaptable a la pantalla de cualquier dispositivo móvil.

REQ-NO-FUN-04 La aplicación estará diseñada de una forma muy estructurada para que permita al usuario operarla y ejecutarla con mayor velocidad,

4.2. Interfaz de hardware

REQ-NO-FUN-06 La aplicación se debe instalar en Smartphone.

4.3. Interfaz de software

REQ-FUN-04**.** La aplicación debe permitir su instalación en Smartphone con sistema android.

5. Requerimientos no funcionales

5.1. Requisitos de rendimiento

REQ-NO-FUN-05 La emisión de avisos y confirmación no debe superar los 3 segundos.

Anexo 4

Documento de pruebas

Los resultados de las pruebas realizadas en la aplicación se detallan en el presente documento. Se analiza cada uno de los módulos y se comprueba si se han cumplido los requerimientos recogidos al inicio de la aplicación.

Propósito

El propósito de las pruebas realizadas es verificar que los componentes de la aplicación se ejecuten perfectamente al desarrollar las funciones requeridas. Se identificaran los errores existentes y la aceptación del usuario final al momento de utilizar la aplicación.

Alcance

El enfoque principal de la aplicación será verificar la funcionalidad de la aplicación en los siguientes aspectos:

- Gestión de registro de facturas
- Gestión de usuarios
- Gestión de generar las búsquedas según los criterios establecidos

Estrategia utilizada para realizar pruebas

Para realizar las pruebas se tendrá en cuenta los requisitos y casos de uso que se especificaron en el documento de descripción de diseño de la aplicación. Las pruebas se realizan examinando cada una de las funcionalidades implementadas con el fin de comprobar si el resultado es el esperado y deseado.

Las pruebas serán realizadas por el desarrollador y por el usuario final una vez instalada en el móvil. Para realizar las pruebas se utilizara la plantilla indicada en la tabla 24, detallando el proceso de las comprobaciones realizadas.

Tabla 25. Plantilla de pruebas

<Nombre caso prueba>	**<Código del CP>**
Descripción: <Descripción del caso de prueba>	
Prerrequisitos **<Enumerar los prerrequisitos para la prueba>**	
Pasos: Pasos generales para la prueba, basados en los escenarios de los casos de uso, si existen.>	
Resultado esperado: **<Resultado esperado de la prueba>**	
Resultado obtenido: <Resultado obtenido de la ejecución del caso de prueba>	

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Detalle de pruebas

Tabla 26. Validación de datos de entrada.

Validación de datos	**001**
Descripción: Se accede a la aplicación y se visualiza la pantalla de acceso.	
Prerrequisitos Estar registrado en el sistema para poder acceder.	
Pasos: 1. Acceder a la aplicación. 2. Rellenar el campo de usuario 3. Rellenar campo de contraseña 4. Ingresar el código de verificación que se visualiza.	
Resultado esperado: Visualizar el mensaje de bienvenido al sistema y la visualización del menú principal.	
Resultado obtenido: Correcto.	

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Tabla 27. Ingreso datos de factura.

Ingreso de datos de factura	002
Descripción: Realizar el ingreso de los datos de las facturas en la aplicación para almacenar los datos de las mismas..	
Prerrequisitos - Estar registrado en el sistema para poder acceder. - Estar ingresados los datos del proveedor.	
Pasos: 1. Acceder a la aplicación. 2. Selección en el menú la opción de Facturas. 3. Selecciona el botón añadir factura. 4. Se visualiza la pantalla Factura. 5. Introducir fecha de factura 6. Ingresar el importe de la factura 7. En el campo proveedor se selecciona el nombre del proveedor almacenado previamente. 8. Introducir la fecha límite de pago 9. Se pulsa el botón guardar.	
Resultado esperado: Almacenamiento de los datos de la factura en la base de datos.	
Resultado obtenido: Correcto.	

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Tabla 28. Registro de usuarios.

Alta nuevo usuario	003
Descripción: Dar de alta a un nuevo usuario de la aplicación	
Prerrequisitos - Conexión al servidor - Ser administrador de la aplicación.	
Pasos: 1. Acceder a la aplicación en calidad de administrador. 2. Selección en el menú la opción de usuarios 3. Selecciona la opción de añadir 4. Rellenar los campos de nombre, usuario y contraseña. 5. Se pulsa el botón guardar.	
Resultado esperado: Almacenamiento y alta de nuevo usuario.	
Resultado obtenido: Correcto.	

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Tabla 29. Asignación de permisos a usuarios.

Asignación permisos a usuarios.	004
Descripción: Asignar permisos a los usuarios para que puedan operar dentro de la aplicación.	
Prerrequisitos - Conexión al servidor - Ser administrador de la aplicación.	
Pasos: 1. Acceder a la aplicación en calidad de administrador. 2. Selección en el menú la opción de usuarios 3. Selecciona la opción de permisos del usuario que deseamos asignar o modificar 4. Seleccionamos los permisos que se desean asignar	
Resultado esperado: Asignación de permisos a usuarios de sistema.	
Resultado obtenido: Correcto.	

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Tabla 30. Generar listado de facturas.

Generar listado	**005**
Descripción: Visualizar un listado de facturas según el criterio seleccionado.	
Prerrequisitos Usuario registrado en la aplicación.	
Pasos: 1. Acceder a la aplicación con usuario y contraseña. 2. Selección en el menú la opción de facturas. 3. Seleccionar el criterio de búsqueda que se necesita. 4. Seleccionar el icono de lupa, para generar los resultados.	
Resultado esperado: Visualización de listado de facturas, según el criterio seleccionado.	
Resultado obtenido: Correcto.	

Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Una vez concluidas las pruebas realizadas en la aplicación se obtiene la versión estable de la misma.

Anexo 5

DOCUMENTO DE DESCRIPCIÓN DE DISEÑO.

1. Introducción

Este documento describe la arquitectura de la aplicación Facturas. La aplicación estará destinada para el uso del personal administrativo de las pequeñas y medianas empresas, permitirá registrar y almacenar los datos referentes al pago de las factura.

En el detalle de arquitectura se describirá los componentes y el diagrama de componentes que se utilizara para desarrollar la aplicación.

2. Arquitectura

A continuación se describirá las diferentes vistas de arquitectura que se aplicaran para el desarrollo de la aplicación Facturas.

2.1. Vista lógica de la aplicación

Diagrama de paquetes

El diagrama que se presentara a continuación, en la figura 1, se indica los componentes principales de la aplicación utilizando el modelo arquitectónico (MVC modelo vista controlador).

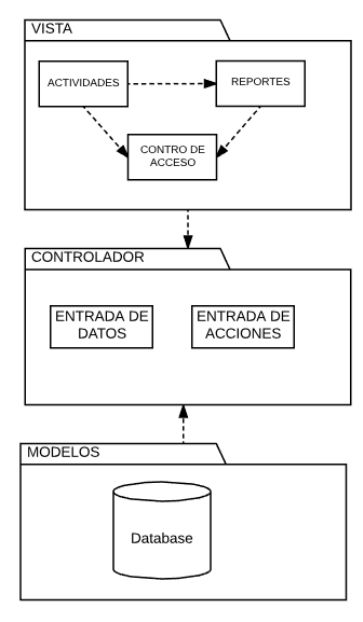


Figura 1. Diagrama de paquetes.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Siguiendo el patrón arquitectónico, el desarrollo se maneja en tres capas, la primera capa vista, indica la información de la interfaz de la aplicación con el usuario, la segunda capa controlador, recibe las entradas de las vistas como eventos y estos los traduce como solicitudes de servicio, y la tercera capa modelos, es la que se encarga de encapsular y/o almacenar datos.

En la figura 17, se muestra que la capa vista envía los datos a la capa controlador, que es la que manejara la lógica de las funciones y esta transmitirá los datos a la capa de modelos para que todos los registros ingresados se almacenen en la base de datos.

2.3. Vista física de la aplicación

Diagrama de despliegue

Este diagrama indica los dos componentes físicos de la aplicación, el un componente representa al cliente y el otro al servidor. El componente del cliente lo maneja el usuario desde el Smartphone, la aplicación instalada se conecta al servidor realizando una petición y el servidor maneja dicha petición a nivel controlador y responde a las solicitudes realizadas.

En la figura 2 se representa el diagrama de despliegue para visualizar los procesos y componentes.

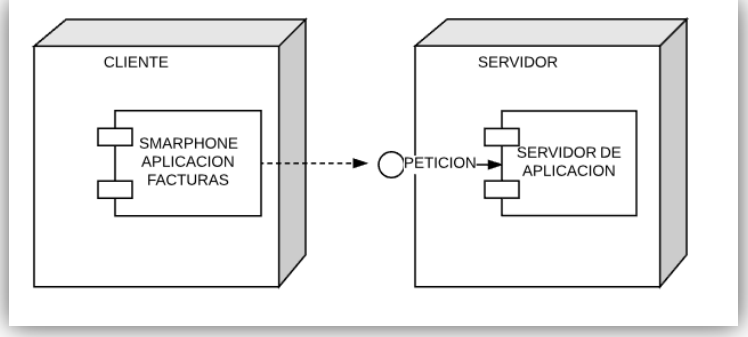


Figura 2. Diagrama de despliegue
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

2.4. Vista de procesos de la aplicación

Diagrama de actividad

Este diagrama permite detallar los procesos que tendrá la aplicación, uno de los procesos más importantes por temas de seguridad es el de acceso a al sistema usando un usuario y contraseña para poder realizar las operaciones disponibles, es muy importante que los usuarios al momento de acceder estén autenticados, si no disponen de autenticación no podrán acceder a la aplicación. Una vez accedido al sistema y verificado los permisos que dispone el usuario se podrán ingresar datos, visualizar pagos, eliminar o modificar estados de pagos.

A continuación, para clarificar el proceso descrito anteriormente, se visualiza el diagrama de actividad en la figura 3.

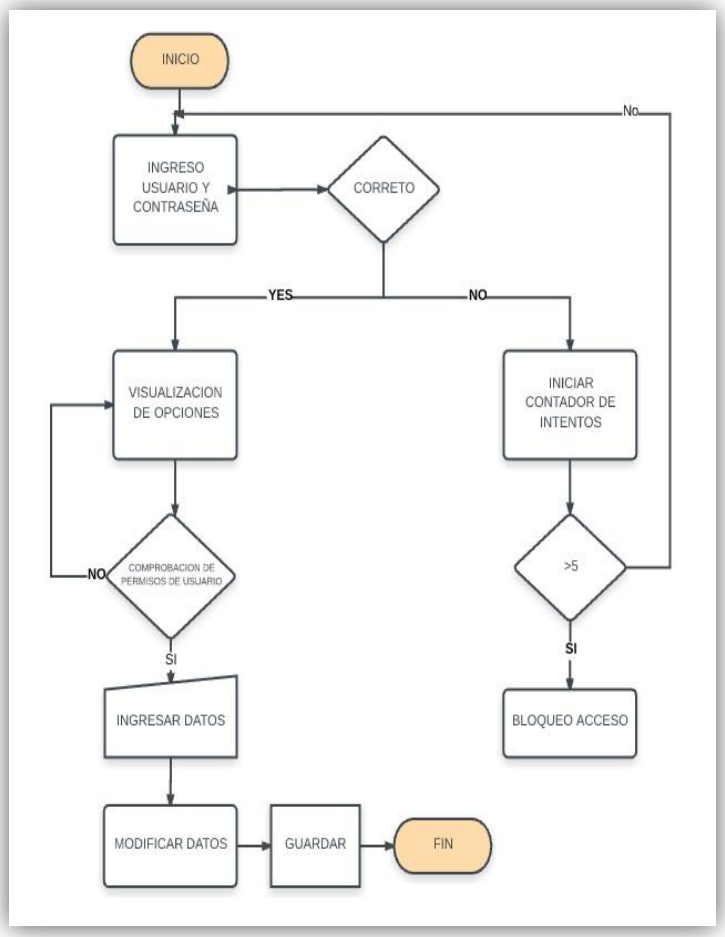


Figura 3. Diagrama de actividad
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

3. Diseño detallado

3.1. Comportamiento de la aplicación

Diagrama de secuencia.

En esta diagrama se describe el comportamiento de la aplicación. Se indica la interacción del usuario cuando realiza una actividad. Cuando el usuario ingresa al sistema, en la pantalla principal se identificara, posteriormente se visualizara el menú principal, en el cual se seleccionara una opción y el sistema validara los permisos para realizar una actividad. Una vez realizada la actividad se guardaran los datos en la base de datos y se retornara la confirmación de que la operación se ha realizado con éxito. Esta secuencia de pasos se visualiza en la figura 4, que se presenta a continuación.

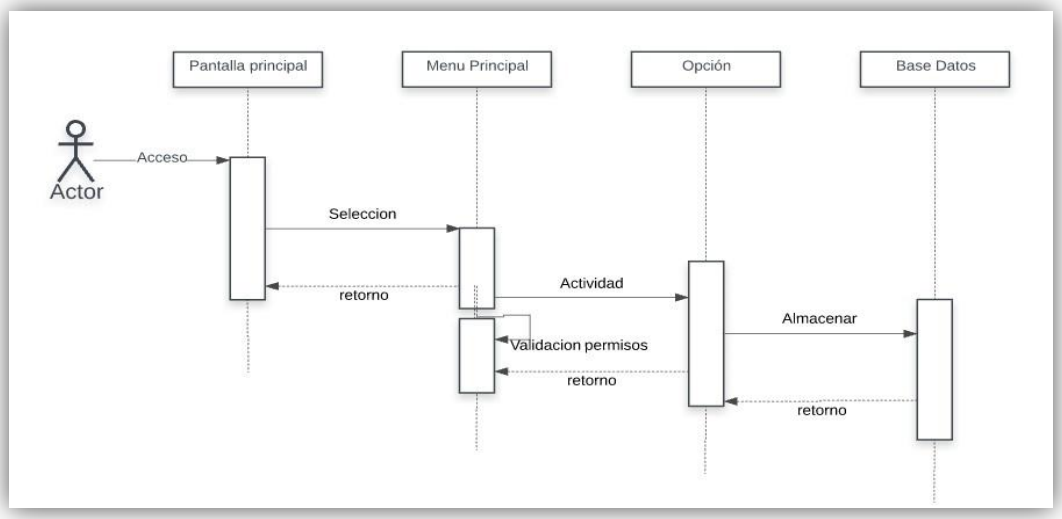


Figura 4. Diagrama de secuencia.
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

4. Persistencia

4.1. Modelo de tablas.

En la figura 5 se presenta un diagrama que detallara la estructura que tendrá la base de datos de la aplicación.

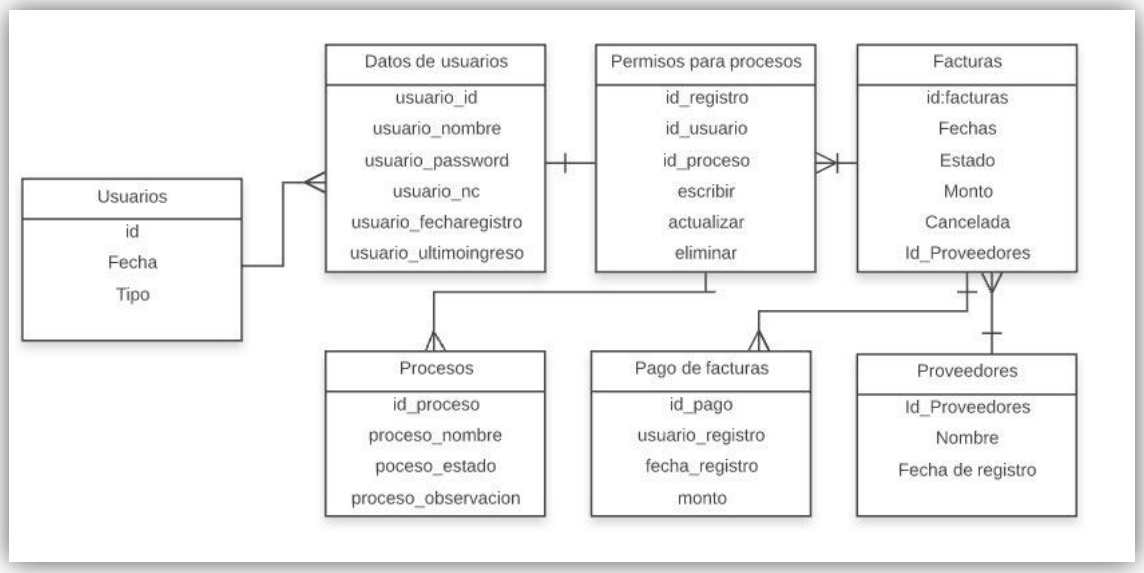


Figura 5. Diagrama de base de datos
Fuente: Análisis de framework.
Elaborado por: Ramos, Consuelo (2017)

Anexo 6

GUIA PLAN DE RECUPERACIÓN ANTE DESASTRES

1. INFORMACIÓN GENERAL

1.	**OBJETIVO**	Establecer las actividades, roles y responsabilidades que permitan mantener la continuidad de los datos almacenados en la aplicación, en caso de que ocurrencia un fallo de seguridad o cualquier otro evento de desastre.
2.	**RESPONSABLE**	Usuario principal.
3.	**ALCANCE**	Este plan se centrara en salvaguardar el uso y la informacion almacenada en la aplicación Facturas

Tipo de Componente	Descripción	Tiempo de Interrupción Tolerable (RTO)
Aplicacion	Facturas	24 horas ((1 día hábil)

2. CONDICIONES GENERALES

1. El plan de recuperacion de desastres (DRP) está enfocado a la protección de los datos almacenados en la base de datos que es utilizada por la aplicación facturas.
2. Supuestos: La efectividad en la ejecución de este documento guía, ante la ocurrencia de un evento de desastre, se fundamenta en los siguientes supuestos:
    - Se dispone de la infraestructura y recursos que soportan las estrategias de contingencia y recuperación.
    - El usuarios que ejecutaran esta guía, o su suplente, se encuentran disponibles.
    - Se han realizado una simulación de desastre al menos 1 vez al año, y han funcionado.
    - La realización de respaldos de las bases de datos e información se realiza de acuerdo a los procedimientos y frecuencias establecidas.

3. GUÍA DEL PLAN DE RECUPERACIÓN ANTE DESASTRES

1. Escenarios de desastre

Los escenarios de desastre, interrupción mayor o un evento contingente que contempla este documento guía son:

Aplicación móvil:

No disponibilidad de acceso a la aplicación:

- Rodo de móvil.
- Hackeo de móvil
- Hackeo base de datos.
- Caída o perdida de movil.

Infraestructura de Comunicaciones:

No disponibilidad de los servicios de comunicaciones por fallas en:

- Conexión de internet.
- Conexión
- Router core

Infraestructura de Servidores

No disponibilidad de la infraestructura por fallas en:

- Servidor Apache
- Servidor Mysql

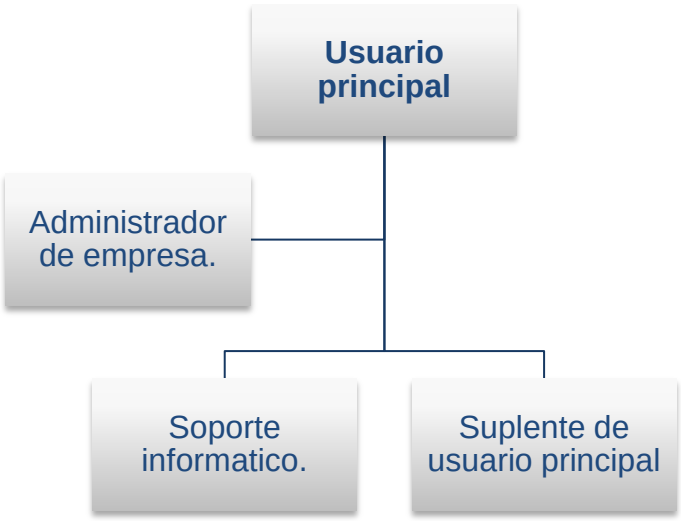
Infraestructura de Bases de datos, Almacenamiento y Respaldo

No disponibilidad de datos e información por:

- Corrupción de la base de datos
- Borrado o pérdida de datos
- Falla total o parcial del servidor de respaldo

NOTA: CUALQUIER ESCENARIO NO CONTEPLADO ANTERIORMENTE, NO HA SIDO CONSIDERADO EN EL PRESENTE PLAN.

4. ROLES Y RESPONSABILIDADES

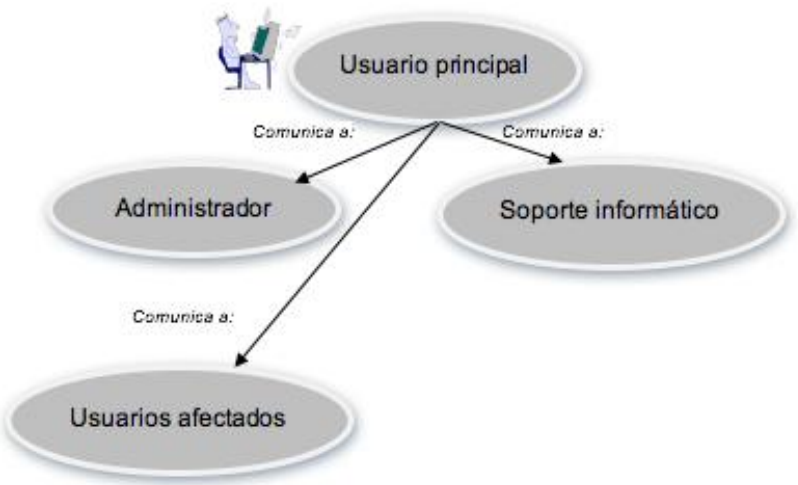


Las responsabilidades definidas para cada rol son:

ROL	ANTES DEL EVENTO D INTERRUPCIÓN	DURANTE EL EVENTO DE INTERRUPCIÓN	DESPÚES DEL EVENTO DE INTERRUPCIÓN
Usuario principal	- Estar pendiente de la actualización del DRP y recursos requeridos. - Realizar simulación una vez al año. - Gestionar la consecución de los recursos para el DRP. - Comunicar al suplente que corresponda sobre la situación de contingencia.	- Evaluar y activar el DRP y las estrategias de recuperación y contingencia. - Comunicar al administrador el estado de la operación de Contingencia. - Informar el momento en que opera en contingencia y que puede suceder con la prestación del Servicio - Liderar la operación bajo contingencia. - Comunicar al soporte informático.	- Velar por la actualización del DRP acorde con los inconvenientes y oportunidades de mejora visualizados durante el evento de interrupción. - Informar al administrador sobre el retorno a la normalidad y agradecer la comprensión y apoyo de todos en esta situación.
Usuario suplente	Estar pendiente de cuando el usuario principal esté ausente	- Ejecutar las actividades asignadas al usuario principal.	- Ejecutar las actividades asignadas al usuario principal.
Soporte informático	- Participar en la ejecución de las pruebas al DRP - Conocer la empresa y los recursos que dispone	- Apoyar a los involucrados en el DRP, en actividades administrativas y logísticas ante una contingencia, entre otras. - Logística de desplazamiento, si es requerido - Asesoramiento en adquisición de nuevos móviles. - Reestablecer la aplicación y las copias de seguridad.	- Reportar los inconvenientes y oportunidades de mejora del DRP
Administrado	- Estar al tanto y aprobar la persona responsable.	- Prestar apoyo a usuario principal - Prestar apoyo a informático.	- Cerciorarse de que el DRP se actualice. - Valorar la eficacia del personal para ejecutar el DRP - Valorar los servicios informáticos recibidos.

5. ÁRBOL DE LLAMADAS

Cuando se presente un desastre, interrupción o evento contingente, se debe seguir la siguiente cadena de llamadas:



Los datos de contacto se encuentran al final del dcumento.

6. ACTIVIDADES DE NOTIFICACIÓN, EVALUACIÓN Y ACTIVACIÓN DEL DRP

¿Quién reporta un incidente, interrupción?

a. Los usuarios deben reportar el incidente cuando:

- NO se pueden utilizar la aplicacion.
- NO hay red de comunicaciones.
- NO hay acceso a la base de datos
- CUALQUIER otro evento de tecnología que afecte la prestación del servicio

b. El personal administrativo debe reportar el incidente cuando:

- Se haya perdido un telefono movil.
- Haya incidentes que afecten al servidor
- Se haya sufrido un robo del terminal.
- CUALQUIER otro evento que afecte o pueda afectar al sistema de informacion.

c. El encargado debe atender el incidente de acuerdo a lo establecido en el presente documento.

- El incidente afecta la disponibilidad de la aplicación.
- El incidente afecta la disponibilidad de la red de comunicaciones a nivel general.
- Ningún usuario tiene acceso a la base de datos.

En cualquiera de los casos, debe escalarlo a los funcionarios responsables.

d. El soporte tecnico debe realizar un diagnóstico sobre el incidente presentado, teniendo en cuenta:

- Naturaleza e impacto del incidente.
- Estrategias definidas en el DRP aplicables u otras soluciones potenciales
- Tiempo estimado de solución del incidente.

e. El nformatico asignado, coordina la ejecución de las actividades para recuperar la aplicación y los datos de la misma teniendo en cuenta:

- Detención de la replicación de datos
- Verificación de la disponibilidad de copias de seguridad.
- Verificacion de conexión de redes.
- Estado de los dispositivos moviles.

f. En caso de que se necesite remplazar el hardware.

- Asesorar para la compra de equipos.
- Sugerencia de proveedores

g. El informatico comunica la solución del incidente a la entidad

h. La empresa encargada del soporte informatico, en conjunto con los profesionales especializados, definen la estrategia de retorno a la normalidad, , teniendo en cuenta:

- Fecha prevista de volver a realizar actividad normal.
- Indicaciones especiales a aplicar en el proceso de retorno.
- Consideraciones especiales con respectos a la recuperación de la información y mantener la integridad de los datos.

i. El usuario principal, en conjunto con el soporte tecnico en la atención del incidente, documentan el incidente e identifican oportunidades de mejora para fortalecer el DRP.

j. Se realiza el cierre del incidente, interrupción mayor o evento contingente, y se continúa con la ejecución del procedimiento de acciones preventivas y correctivas del SGSI.

7. ACTIVIDADES DE MANTENIMIENTO

Es responsabilidad del encargado del plan desarrollar las nuevas versiones al DRP, y la comunicación de las mismas a todos los funcionarios involucrados en el mismo.

La actualización y mantenimiento al DRP se debe realizar:

- Pasado un año desde la última actualización.
- Cuando han ocurrido cambios en la aplicación o servidores,objeto del alcance de esta guía.
- Si los resultados de las pruebas requieren actualización del DRP o sus procedimientos.
- Cuando hay cambios en el personal que operaría el DRP.
- Cuando los resultados de auditorías así lo indican.

Algunas actividades a realizar para mantener vigente el DRP, son:

No	Actividad	Responsable	Frecuencia
1.	Actualización de los procedimientos de recuperación y contingencia de la plataforma tecnológica	Usuario principal	Cuando hayan cambios que afecten a la aplicación.
2.	Realizar pruebas periódicas para verificar el correcto funcionamiento de los sistemas respaldados	Usuario princiapl	De forma trimestral.
3.	Ejecución del procedimiento de respaldo de daatos.	Encargados asignados.	Una vez al día.
4.	Obtener imagen del sistema de servidores y equipos de red.	Encargados asignados	Semestral.

8. DISTRIBUCIÓN DE LA GUIA: PLAN DE RECUPERACIÓN ANTE DESASTRES

Este este plan se entregara bajo las siguientes consideraciones:

- Se debe entregar una copia final COMPLETA del DRP a:
  - Empresa que brinda soporte tecnico.
  - Usuario principal encargado.
  - Administrador de empresa.
  - Existir una copia en el archivo de documentos de la empresa.

1. RECURSOS MÍNIMOS REQUERIDOS

La infraestructura necesaria para soportar los procesos misionales de la entidad que serán recuperados en una contingencia es:

Cant.	Equipo	Caracteristicas	Almacenamiento
1	Disco duro	Compatible sistema operativo de empresa	1 Terabyte
1	Movil de sustitucion	Sistema android.	1 g de RAM y 16 G internos.

Anexo 7

Plan de comunicación.

1. INTRODUCCIÓN

Objetivo

El principal objetivo del presente documento es establecer la forma que se transmitirá la información en caso de necesidad de soporte técnico o fallos de seguridad.

Alcance

El presente documento está destinado a las personas que hagan uso de la aplicación facturas.

2. OBJETIVOS DEL PLAN

- Identificar las personas receptoras del mensaje.
- Establecer los canales de comunicación
- Expectativas y beneficios esperado.

3. ESTRATEGIA DE COMUNICACIÓN

- Que información se transmitirá?
    - Toda la información que sea referente a la aplicación o datos manipulados por lo misma.
- Audiencia objetivo
    - Personas usuarias de aplicación
- Canales que se utilizara para transmitir información
    - Correo electrónico
    - Misiva

      Nota : Utilizar plantilla indicada en la tabla1.

4. PLAN DE ACCIONES DE COMUNICACIÓN

Acciones de comunicación

Esta plantilla se utilizara por cada motivo de comunicación identificando cada uno de con los siguientes campos:

- Código identificativo de la acción.
- Descripción de la acción de comunicación y objetivos principales perseguidos.
- Responsable de la realización de dicha acción.
- Audiencia objetivo: Perfil o colectivo al que va dirigida la comunicación.
- Dependencias/Condicionantes: Se indicarán las posibles dependencias con otras acciones dentro del plan, así como condicionantes necesarios para la realización de la misma.
- Recursos humanos y materiales necesarios para el desarrollo de la acción.
- Canales de comunicación: Identificar las herramientas, medios o información necesarios para llevar a cabo la comunicación de la acción en cuestión.
- Observaciones o comentarios que se consideren de interés.

<Motivo de comunicación>	
Código	
Descripción/Objetivos	
Responsable(s)	
Audiencia objetivo	
Dependencias/Condicionantes	
Recursos humanos y materiales	
Canales de comunicación	
Observaciones	

Anexo 8

Aplicación Facturas

Manual de Usuario

Versión: 001

Fecha: 30/06/2017

HOJA DE CONTROL

Organismo	Universidad Técnica Particular de Loja		
Proyecto	Aplicación Facturas.		
Entregable	Manual de Usuario		
Autor	Ramos Vásquez, Consuelo		
Director	Jaramillo H. Danilo, Msc.		
Versión/Edición	001	**Fecha Versión**	30/06/2017
Aprobado por		**Fecha Aprobación**	
		Nº Total de Páginas	13

REGISTRO DE CAMBIOS

Versión	Causa del Cambio	Responsable del Cambio	Fecha del Cambio

CONTROL DE DISTRIBUCIÓN

Nombre y Apellidos
Ramos Vásquez Consuelo del Pilar

Contenido

1 DESCRIPCIÓN DEL SISTEMA

1.1 Objeto

La aplicación Facturas permite a las pequeñas y medianas empresas realizar un control de las fechas de pago de las facturas que recibe de los proveedores, desde su dispositivo móvil, ya que la información almacenada en la base de datos se puede consultar los pagos en cualquier lugar y momento, con el principal objetivo de evitar costos por impago de facturas. .

1.2 Alcance

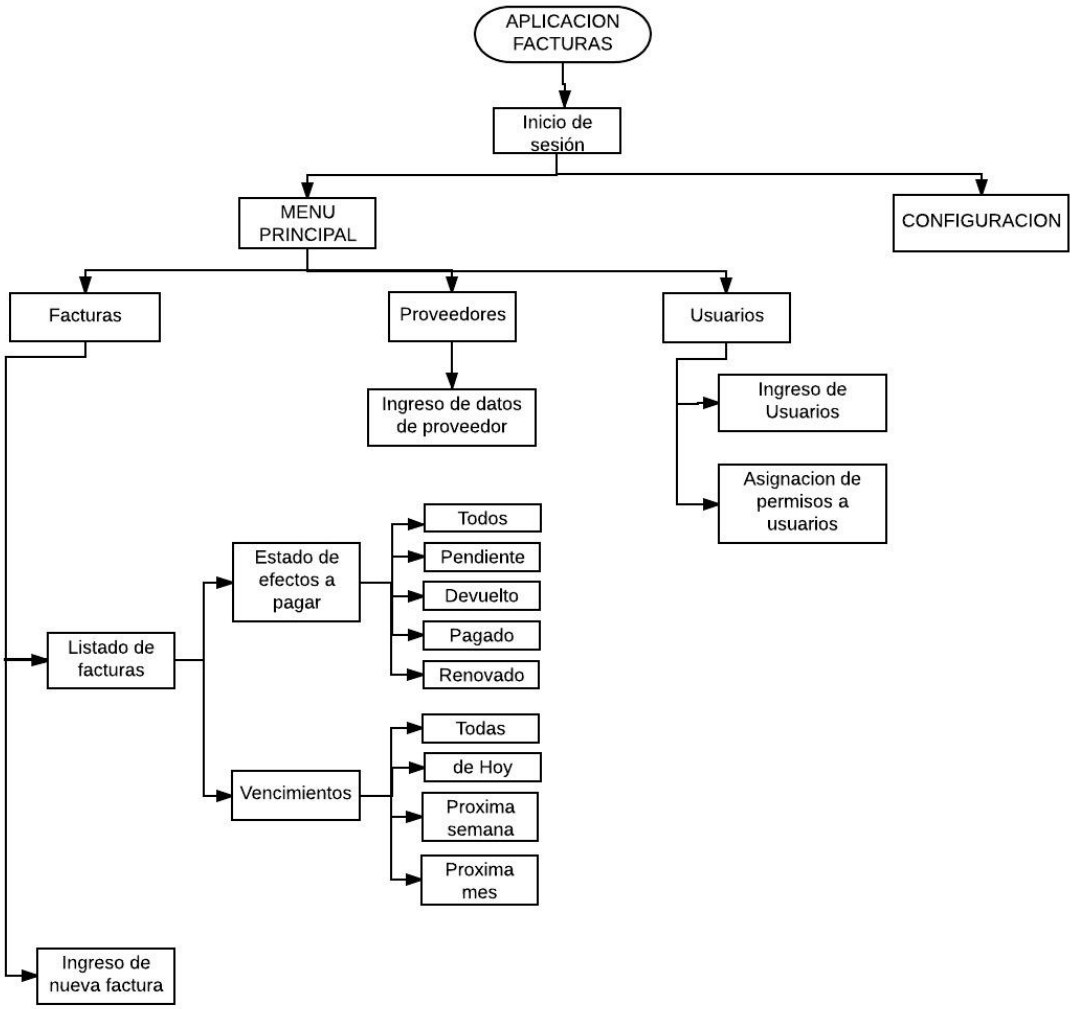
El presente manual detalla las funciones que realiza la aplicación, proporcionando una descripción de cada una de las opciones disponibles.

1.3 Funcionalidad

La presente aplicación presta un servicio de almacenar los datos necesarios de proveedores, facturas y usuarios, para crear un listado de las facturas que recibe la empresa y organizarlas por fecha de pago. Además permite controlar las funciones que tendrán los usuarios con acceso a la aplicación.

2 MAPA DEL SISTEMA

2.1 Diagrama de navegación



3 DESCRIPCIÓN DEL SISTEMA

Al pulsar el icono de facturas, figura1, se accede a la aplicación.

Figura 1. Pantalla inicial

Una vez se accede a la aplicación se visualiza la siguiente pantalla, figura 2, en la cual se debe introducir el usuario, contraseña y el código de seguridad que si visualiza,

Nota: tras cinco veces de intentos fallidos el usuario quedara bloqueado.

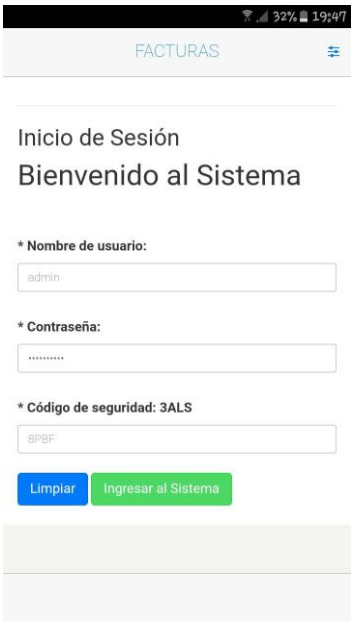


Figura 2. Ingreso al sistema.

Posteriormente se visualiza la pantalla de bienvenido al sistema, figura 3.

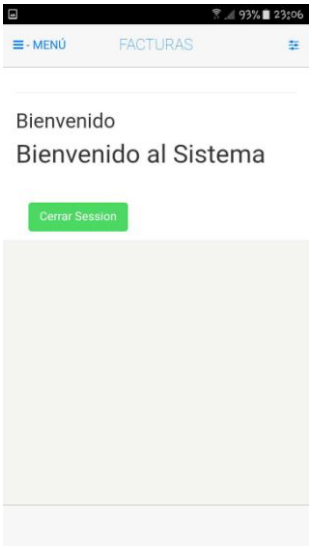


Figura 3. Mensaje de bienvenida.

3.1 Opciones de menú

Una vez accedido a la aplicación se visualiza el menú principal, figura 4, se divide en cuatro opciones:

- Inicio, permite volver a la pantalla inicial.
- Facturas, opción donde se permite ingresar facturas, visualizar listados según criterios que permite seleccionar.
- Proveedores, permite ingresar datos de proveedores.
- Usuarios, opción que permite dar de alta un nuevo usuario y asignar usuarios.

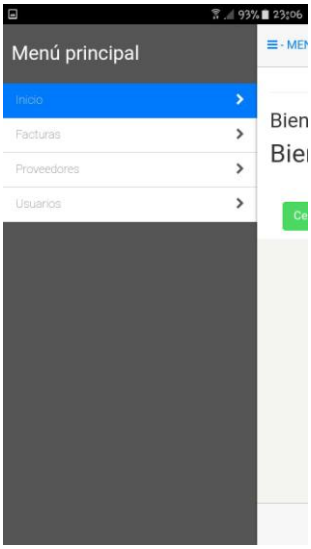


Figura 4. Visualización de menú.

3.1.1 Facturas

Al pulsar la opción de menú de facturas, se visualiza las opciones en que se permite visualizar las facturas introducidas, en la figura 5 y figura 6, se indica las opciones

disponibles.

Figura 5. Menu de facturas.

Figura 6. Menu de facturas.

Al pulsar icono  se abre la pantalla en donde se puede ingresar los datos de la nueva factura, según se pulse la opcion desada se abriran pantallas que permitan escoger y guardar los datos necesarios. En las figuras 7, 8 , 9 , 10, se vizualiza.

Figura 7. Pantalla de ingreso de facturas.

Figura 8. Pantalla de selección de estado de efecto.

Figura 9. Selección de fecha de factura.

Figura 10. Selección de proveedores.

3.1.2 Proveedores

En esta opción permite el ingreso de los datos de proveedores, una vez ingresado los datos, pulsar el botón guardar y los datos se almacenaran. En la figura 11, se indica los datos que son necesarios.

Figura 11. Ingreso de datos de proveedores.

3.1.3 Usuarios

Al pulsar esta opción visualizamos el listado de usuarios existentes, en la figura 12 se visualiza la pantalla. Para añadir un nuevo usuario pulsaremos la opción añadir, donde se abrirá una nueva pantalla (figura 13), y para editar los permisos de cada usuario, se pulsara la opción de permisos en donde se desplegara una pantalla donde seleccionaremos los permisos deseados (figura 14).

Figura 12. Vizualicion de listado de usuarios.

Esta opción permite el registro de nuevos usuarios, en donde se ingresa el nombre, usuario y contraseña, una vez guardado se pueden asignar los permisos necesarios. En la figura 13, se visualiza la pantalla.

Figura 13. Ingreso de usuario.

La pantalla Permisos administrativos, indicada en la figura 14, permite seleccionar los permisos que se puede asignar a cada usuario.

Figura 14. Selección de permisos.