UNIVERSIDAD SAN PEDRO

FACULTAD DE INGENIERÍA

ESCUELA ACADÉMICO PROFESIONAL INGENIERÍA INFORMÁTICA Y DE SISTEMAS

Aplicación móvil multiplataforma de integración con ERP para la visualización y aprobación de solicitudes de dinero en el grupo MOLICOM

"Tesis para obtener el título profesional de ingeniero en informática y de sistemas"

Autor:
Víctor Manuel Cárdenas León

Asesor:
Oscar Arquímedes Ascón Valdivia

CHIMBOTE – PERÚ

2019

INDICE

Tema **Página Nº.**

Palabras clave

Tema	Aplicación Móvil
Especialidad	Ingeniería de Software

Keywords

Theme	Mobile Application
Especialty	Software Engineering

Líneas de Investigación:

Área: Ingeniería y Tecnología.

Sub Área: Ingeniería Eléctrica, Electrónica e Informática.

Disciplina: Ingeniería de Sistemas y Comunicaciones.

Aplicación móvil multiplataforma de integración con ERP para la visualización y aprobación de solicitudes de dinero en el Grupo Molicom.

RESUMEN

La presente investigación tuvo como propósito la implementación de una aplicación móvil que permita visualizar y aprobar documentos de solicitudes de dinero. Existe en la empresa un sistema ERP donde el usuario aprobador puede realizarlo, la falta de atención inmediata retrasa los procesos administrativos.

El tipo de investigación para el presente proyecto es a nivel de estudio descriptivo y de orientación aplicada experimental de corte transversal, se basa en el desarrollo de una aplicación móvil multiplataforma que funcione en distintas versiones de los sistemas operativos Android y iOS. Se utilizara el lenguaje de programación de Visual Studio con herramienta Xamarin.Forms para el desarrollo del Cliente y con una tecnología Web Api del lado del servidor de servicios web. Además, una base de datos en SQL Server donde se obtendrá y actualizará la información.

Como resultado del proyecto, se brinda la solución que facilite a los usuarios aprobadores obtener la información y la pronta aprobación de un documento de solicitud de dinero mediante una aplicación móvil, así como también dejar un marco de trabajo mediante encapsulación de código nativo, servicios, estándares y despliegue multiplataforma (Android y iOS).

ABSTRACT

The purpose of this research was to implement a mobile application that allows you to view and approve money request documents. There is an ERP system in the company where the approving user can do it, the lack of immediate attention delays administrative processes.

The type of research for this project is at the level of descriptive study and experimental applied cross-sectional guidance, it is based on the development of a multiplatform mobile application that works on different versions of Android and iOS operating systems. The Visual Studio programming language will be used with Xamarin.Forms tool for Customer development and with a Web Api technology on the web services server side. In addition, a database in SQL Server where the information will be obtained and updated.

As a result of the project, the solution is provided that facilitates approving users to obtain the information and prompt approval of a money request document through a mobile application, as well as leaving a framework through encapsulation of native code, services, standards and cross-platform deployment (Android and iOS)

.

I. INTRODUCCIÓN

Este problema se suscitó en toda la organización, pero con el pasar del tiempo se vinieron mejorando los procesos de aprobación de solicitud de dinero; es así que la empresa del Grupo Molicom, desea aprovechar la tecnología de los dispositivos móviles y emplearla para su beneficio, por lo que requiere una aplicación que permita visualizar la información para ser aprobada y así agilitar el proceso de aprobación de solicitud de dinero.

Es así que en nuestro medio, por ejemplo, contamos con antecedentes los cuales se han abordado los trabajos más relevantes a esta investigación:

Alexander Ocsa, Gustavo Suero, Jose Herrera, Klinge Villalba (2014) con la tesis, "PROPUESTA PARA EL DISEÑO Y DESARROLLO DE APLICACIONES M-LEARNING: CASO, APPS DE HISTORIA DEL PERÚ COMO OBJETOS DE APRENDIZAJE MOVILES".

El objetivo del proyecto presento el análisis diseño de una aplicación MLearning basado en tópicos de interacción humano computador, diseño centrado en el usuario, desarrollo multiplataforma, se usó la metodología de Software Educativo MeISE, el cual propone un ciclo de vida dividido en dos etapas. Se desea mejorar la portabilidad en el despliegue y reducir la complejidad desarrollo mediante la encapsulación de código nativo, servicios y estándares; para conseguir una aplicación educativa móvil de alta calidad, estandarizadas a TIN CAN API, y despliegue multiplataforma (iOS, Android y Windows).

Como resultado se obtuvo dos casos de aplicación: Una App interactiva en formato comic sobre la fundación del imperio incaico y una App en formato de libro interactivo sobre los principales atractivos turísticos del Perú. Las aplicaciones presentadas tienen alta demanda cognitiva para los estudiantes, para el caso del libro interactivo orientado al desarrollo de capacidades y en el caso del comic al desarrollo de los niveles de comprensión lectora, estos aspectos son requisitos que se tomaron en cuenta para elaborar dichas aplicaciones.

Emerson Damir Cadena Navarrete (2015) con la tesis, "DESARROLLO DE UNA APLICACIÓN MULTIPLATAFORMA Y DISTRIBUIDA PARA DISPOSITIVOS MOVILES QUE PERMITA AUTOMATIZAR EL MANEJO DE INFORMACION DEL TRANSPORTE INTERPROVINCIAL DE LA CIUDAD DE QUITO".

El objetivo del proyecto consiste en el desarrollo de un prototipo que permita manejar de forma adecuada la información del transporte interprovincial de la ciudad de Quito, con este desarrollo se propone una solución que facilite a los usuarios obtener información del servicio que presentan a los terminales de Quito.

Mediante el uso del lenguaje UML se definieron los casos de uso y el diagrama de clases.

El proyecto se basó en el desarrollo de una aplicación multiplataforma que funciono en distintas versiones de los sistemas operativos Android y Windows Phone, se utilizó el lenguaje de programación C# para el desarrollo del cliente y del

servidor con una base de datos como repositorio de la información, la parte principal de la aplicación pudo utilizarse para aplicaciones en otras plataformas.

Como resultado se obtuvo facilitar el acceso al servicio que presentan los terminales interprovinciales mediante la aplicación para los sistemas Windows Phone y Android con la cual permite manejar de forma adecuada la información del transporte interprovincial para los diferentes tipos de usuarios.

Gustavo Hernando Pum Tejada (2016) con la tesis, "SISTEMA MOVIL DE DETECCION DE CAIDAS PARA ADULTOS MAYORES USANDO BEACONS".

El objetivo del proyecto propone un sistema móvil de bajo costo, diseñado principalmente para Smartphone, el cual junto con unos dispositivos inalámbricos llamados Beacons nos permite realizar la detección de caídas.

El prototipo implementado fue desarrollado utilizando una metodología ágil, con iteraciones cortas y teniendo en cuenta las buenas prácticas de programación, en especial en lo referente al lenguaje de programación C# (Microsoft Inc., 2016)

La solución se dio mediante el diseño e implementación de un sistema ubicuo de detección de caídas para adultos mayores que permita la detección y respuesta rápida, sin requerir la constante supervisión.

Esto es posible a través de una serie de algoritmos diseñados exclusivamente para el sistema propuesto, el sistema tiene dos modos: Paciente y Cuidador.

El Paciente es la persona en riesgo de sufrir una caída y El Cuidador, la persona que será notificada sobre cualquier emergencia que sea detectada por el sistema. Se desarrolló una app prototipo para Android y iOS para probar su precisión.

José Marcos Cueva Rodríguez (2016) con la tesis "IMPLEMENTACIÓN DE UNA APLICACIÓN PARA PROCESOS DE CALIFICACIONES DEL COLEGIO DE BACHILLERATO "CIUDAD DE LOYOLA"".

El objeto del proyecto es desarrollar una Aplicación Web para el Colegio de Bachillerato "Ciudad de Loyola" Para el desarrollo de la Aplicación se utilizó la arquitectura Cliente-Servidor por capas. El sistema se desarrolló en su totalidad en el lenguaje de programación C#.NET. Para la creación de reportes se usó Reporting Services y cada reporte se muestra en una página web haciendo uso de un visualizador ReportViewer. La Capa de Servicios Web, se desarrolló usando la plantilla de proyectos Web API, que permite crear servicios REST. La aplicación de consultas de notas se desarrolló en Xamarin.Forms.

Todo esto bajo las normativas de la metodología de desarrollo SCRUM. Gracias al uso de la Aplicación Web y Móvil en el entorno del Colegio, se optimizará la realización de los procesos de matriculación y registro de calificaciones para el Colegio de Bachillerato "Ciudad de Loyola".

Como resultado se obtuvo la automatización de los procesos de registro de empleados, estudiantes y representantes; también se optimizó la gestión y administración de los registros generados a raíz del proceso de matriculación; se automatizó también, el proceso de registro de calificaciones, y se dispuso un

servicio web disponible para estudiantes, a fin de facilitar la recuperación de calificaciones a través de dispositivos móviles.

José Carbo Vite (2017) con la tesis, “DESARROLLO DE UNA APLICACIÓN MÓVIL CON LA HERRAMIENTA XAMARIN STUDIO PARA EL APOYO Y SOPORTE DE LOS PACIENTES DE ENFERMEDADES RENALES CRÓNICAS DE LA UNIDAD DE HEMODIÁLISIS DIALRÍOS”.

El objetivo del proyecto fue implementar la aplicación móvil con la herramienta Xamarin Studio para el apoyo y soporte de los pacientes de enfermedades renales crónicas en la unidad de hemodiálisis Dialríos.

En donde fue desarrollada en Visual Studio 2015 con la herramienta Xamarin la cual permitió programar un solo código en C# reutilizando gran pate del mismo, está disponible para las plataforma Android e iOS. Para el desarrollo del proyecto se usará el modelo en cascada, sólo aplicable cuando están totalmente cerrados los requisitos y no van a cambiar, no hay retroalimentación entre las fases o etapas en que se divide el proyecto, por lo que cada fase o etapas se va cerrando en forma secuencial, todo el proceso del proyecto está fijado por fecha límite y presupuesto.

Como resultado se obtuvo la aplicación móvil dialdiet “dieta de dialríos” provee a los pacientes de enfermedades renales crónica de la unidad de hemodiálisis dialríos de una guía de alimentación dietética nutricional y consejos de acuerdo a su enfermedad.

La presente investigación se justifica científicamente, ya que conlleva conocimientos selectivos y sistematizados para explicar racionalmente los procesos de desarrollo en una aplicación móvil multiplataforma para aprobaciones de solicitud de dinero en el Grupo Molicom y contribuir a la mejora de la actualización de la información, así como facilitar a las personas que laboran en dicha organización.

La investigación se justifica socialmente ya que las diferentes líneas de negocio ofrecidos por el Grupo Molicom, son las que tienen mayor demanda en el Perú, cabe mencionar que es importante destacar que la elaboración y desarrollo de la investigación permitirá agilizar los procesos administrativos en el área de Finanzas y Tesorería, evitando así tiempo de perdida innecesario en la demora de entrega de dinero para algún fin en la empresa, de esta manera presentaremos en nuestra sociedad a una entidad competitiva, capaz de brindar a sus colaboradores soluciones a sus necesidades. Esto con el fin de su seguridad, funcionalidad y productividad. Y de esta manera poder ser una entidad capaz de satisfacer las necesidades de sus proveedores de una forma ágil, versátil y eficiente.

Actualmente el Grupo Molicom está apostando por la tecnología, brindando a las diferentes áreas de la empresa mejores estrategias tecnológicas a sus problemas, para crear soluciones nuevas e innovadoras. Es por ello que la Gerencia de Sistemas ha decidido que las jefaturas nacionales y Gerencia Financiera Central; que son los aprobadores del documento de solicitud de dinero, se ponga en marcha un proceso de automatización lo cual permita agilizar el proceso de aprobación.

La aprobación de solicitud de dinero se realiza mediante un sistema ERP. Este proceso tiene 3 niveles de aprobación, inicia desde que un usuario administrativo de las diferentes sucursales genera la solicitud de dinero, hasta que pueda ser aprobado por las distintas jefaturas o gerencias, el proceso de aprobación demora alrededor de 1 a 2 días, la demora se debe a que la gerencia financiera o jefaturas aprobadoras por sus diferentes actividades fuera de oficina, normalmente tienen que apersonarse para ingresar al sistema y realizar la aprobación, lo que causa que el proceso no sea ágil y las solicitudes de dinero se demoren mucho más de lo que deberían.

Por lo tanto se ha visto en la necesidad de proponer una solución que facilite a la gerencia obtener la información de manera oportuna.

La solución propuesta se basa en el desarrollo de una aplicación móvil multiplataforma que funcione en distintas versiones de los sistemas operativos Android y iOS. Se utiliza el lenguaje de programación de Visual Studio con herramienta Xamarin.Forms para el desarrollo del Cliente y con una tecnología Web Api del lado del Servidor de servicios web. Además, una base de datos en SQL Server donde se obtendrá y actualizará la información.

El presente proyecto por lo tanto presenta ofrecer una solución a la problemática planteada.

En la presente investigación se presenta el siguiente problema:

¿En qué medida una aplicación móvil de aprobación puede mejorar en el proceso de la solicitud de dinero dentro del GRUPO MOLICOM?

Para poder tener una mejor respuesta a nuestra interrogante, se detalla los aspectos teóricos que permiten cumplir con el proyecto. Se describe los conceptos para el diseño y el desarrollo de software, además de las principales características de las herramientas que se utiliza y se incluye conceptos generales del tema.

Metodologías ágiles según Abrahamsson (2002), ante la demanda creciente que experimentó la industria del software por metodologías más "ligeras", las metodologías ágiles comienzan a aparecer en la década de los noventa, diferenciándose de las metodologías tradicionales por su naturaleza incremental, adaptable y de trabajo en equipo.

Se consolidan como metodologías con la aparición del "manifiesto de desarrollo de software ágil" (K. Beck, Mike Beedle, Alistair Cockburn, Martin Fowler, Jim Highsmith, Robert C. Martin, Ken Schwaber, Jeff Sutherland, 2001). Según este manifiesto, la máxima prioridad está en satisfacer a los clientes a través de la entrega temprana y continuada de software funcional que aporte valor real al negocio, para lo cual la comunicación entre los desarrolladores, sus interacciones, la colaboración con el cliente y la respuesta ante el cambio, son los pilares para conseguirlo.

La Figura muestra las claves del manifiesto y el cambio de prioridad en los 4 enunciados que lo componen (por ejemplo, individuos e interacciones antes que los procesos y las herramientas).

Figura 01. Claves y enunciados del manifiesto ágil.
Fuente: Manifiesto ágil, K. Beck, Mike Beedle, Alistair Cockburn, Martin Fowler, Jim Highsmith, Robert C. Martin, Ken Schwaber, Jeff Sutherland, 2001.

La repercusión de las metodologías y prácticas ágiles ha tenido un alcance mundial según Dave West (2010) y como resultado de esta investigación del estado del arte, se han identificado un total de 47 metodologías generales o específicas en determinadas prácticas, de las cuales algunas son resultado de la combinación de otras (véase Figura 02).

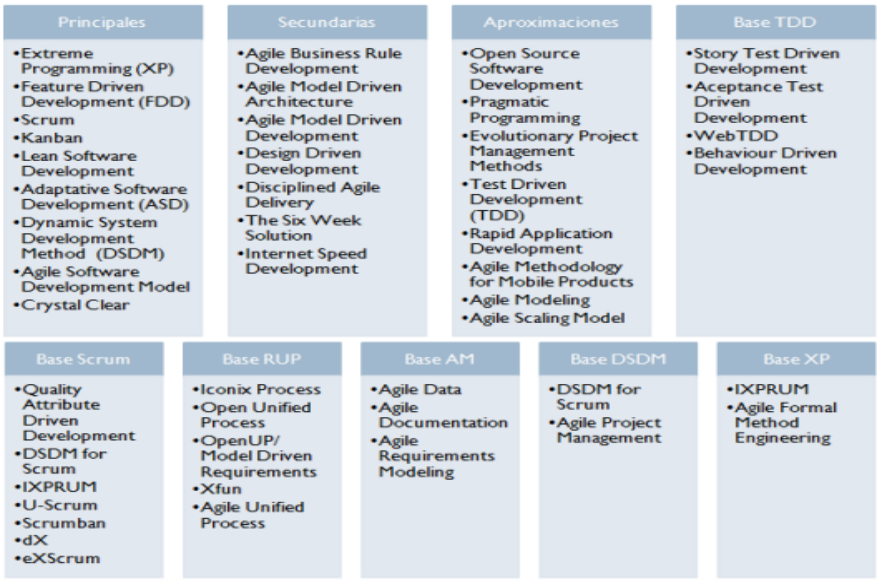

Figura 02. Metodologías ágiles identificadas en la investigación del estado del arte.
Fuente: Agile Development: Mainstream Adoption Has Changed Agility, 2010.

A pesar del número elevado de propuestas existentes, ha sido “Scrum”, el método ágil usado más común. Si realizamos una verificación cruzada de las estadísticas del tipo de métodos ágiles utilizados (Scrum, Kanban, XP, Scrumban, etc.) con las estadísticas del tipo de técnicas ágiles utilizadas (sincronización diaria, iteraciones, retrospectivas, etc.) podemos concluir que más del 85% de las organizaciones utilizan el marco Scrum como base, según la última encuesta realizada por “Versión One Inc.” publicada en el año 2018 (Versión One Inc., 2018).

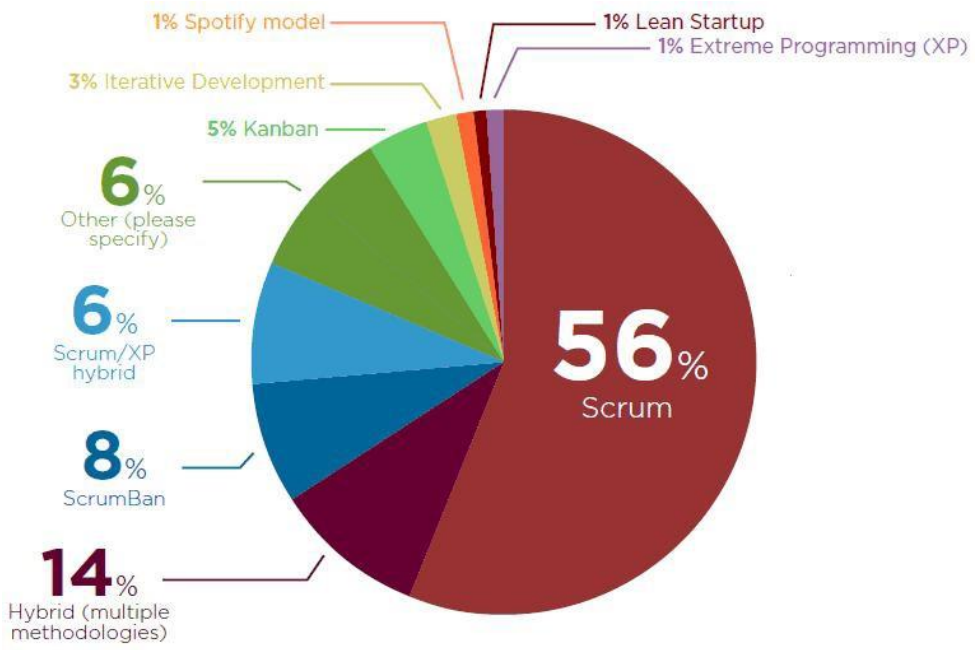

Figura 03. Marcos de Metodologías ágiles más usadas en las organizaciones.
Fuente: 12th. Anual State of Agile Development Survey, Version One Inc., 2018.

A continuación, se presenta un resumen de la metodología ágil Extreme Programming (XP) siguiendo la descripción y modelo utilizado en (Abrahamsson,

(2002), con el fin de cubrir distintas tipologías y coberturas del ciclo de vida de un proyecto de desarrollo de software.

Extreme Programming (XP) según (Beck, 2000), es un compendio de buenas prácticas acumuladas año tras año en el desarrollo del software, que fueron aglutinadas y ordenadas en una nueva metodología.

Refuerza la importancia de responder a los cambios del cliente de manera efectiva, incluso cuando el proyecto está muy avanzado, y de igual manera, valora altamente el trabajo en equipo entre los desarrolladores, clientes y gerentes (alineamiento perfecto con el manifiesto ágil).

Propone entregas frecuentes en ciclos de desarrollo cortos para mejorar la productividad e introducir puntos de verificación en los cuales el cliente pueda incorporar nuevos requisitos.

Dispone de 5 valores sobre los cuales gira su filosofía (Beck, Andres, 2004): comunicación, simplicidad, realimentación, coraje y respeto. Sus prácticas están agrupadas en 4 áreas:

- Realimentación: programación en parejas, juego de planificación, desarrollo guiado por pruebas (test driven development), el equipo como conjunto.
- Proceso continuo: integración continua, refactorización, entregas pequeñas.
- Compartir el aprendizaje: estándares de codificación, propiedad colectiva, diseño simple, metáfora del sistema.
- Bienestar del desarrollador: paz sostenible.

Resumen del proceso

El proceso está organizado en 6 fases principales:

- **Fase “Exploración”**

  En esta fase el cliente escribe las tarjetas de las historias de usuario que desea sean desarrolladas para la primera (o siguiente) entrega. Al mismo tiempo el equipo de desarrollo se familiariza con la tecnología, herramientas y prácticas que serán utilizadas durante la realización del proyecto. Puede tener una duración de semanas o meses según las necesidades del proyecto y el conocimiento del equipo en los ítems mencionados.

- **Fase “Planificación”**

  En esta fase el equipo de desarrollo estima las historias de usuario que serán incluidas en la siguiente entrega prevista. Esta fase solo dura un par de días y la ventana de tiempo para la siguiente entrega no debe exceder los dos meses.

- **Fase “Iteraciones”**

  Incluye todas las iteraciones necesarias para estabilizar el producto hasta realizar la siguiente entrega. Cada iteración está compuesta por un subconjunto de las historias de usuario previstas y tiene una duración de entre una o dos semanas. Paralelamente se crean las pruebas funcionales que luego se ejecutan al finalizar cada iteración. Al finalizar la última iteración, el producto está listo para salir a producción.

- **Fase “Producción”**

Requiere una verificación adicional a nivel funcional y de rendimiento.

En esta fase se podrían identificar nuevos cambios para ser abordados en la siguiente iteración o postergados para la fase de mantenimiento.

- **Fase "Mantenimiento"**

  En el momento que se realiza la primera entrega, el proyecto entra en un modelo de desarrollo y mantenimiento en paralelo. Durante el mantenimiento se reduce la velocidad de desarrollo del equipo porque también deben realizar tareas de soporte al cliente. Si la carga de mantenimiento es grande, incluso puede requerir incorporar más gente al proyecto y cambiar la estructura del equipo.

- **Fase "Muerte"**

  Esta fase se alcanza cuando el cliente no tiene más historias de usuario que implementar, y cuando aspectos como la estabilidad y el rendimiento han sido también satisfechos. Es el momento en el que se produce la documentación final que requiera el proyecto. También podría ocurrir en caso de que el proyecto no alcance las expectativas previstas o el trabajo de desarrollo o mantenimiento supera el presupuesto disponible.

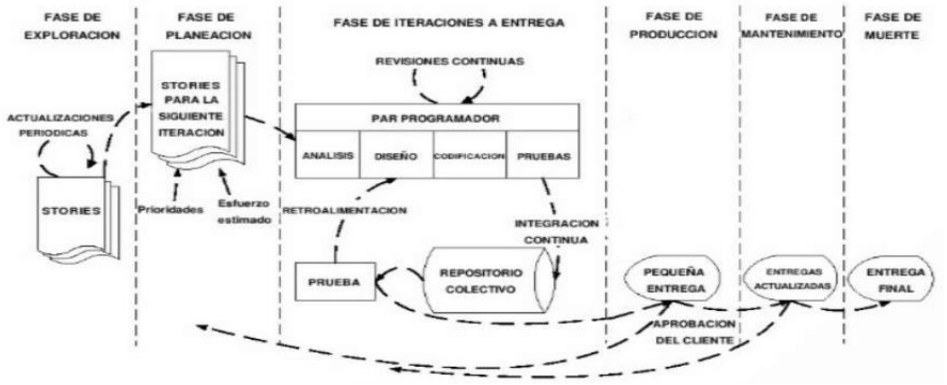

Figura 04. Ciclo de vida del proceso XP.
Fuente: Abrahamsson, (2002), Agile Software Development Methods.

Conceptos generales

ASP.NET Web API, según Lakshmiraghavan (2013), es un Framework de .NET para construir servicios Web API, basados en HTTP y arquitectura REST.

Los Servicios HTTP, desplegados sobre la arquitectura API Web pueden ser consumidos por diferentes tipos de clientes:

- Browsers
- Aplicaciones móviles
- Aplicaciones Desktop

Los Servicios Web API, son ligeros y buenos para dispositivos que tienen un ancho de banda limitado, como los teléfonos inteligentes.

Protocolo HTTP (Hypertext Transfer Protocol)

El protocolo de transferencia de hipertextos (HTTP) según Stallings W. (2004), es el protocolo base de *world wide web* (WWW) y se puede utilizar en cualquier aplicación cliente-servidor que suponga la utilización de hipertextos. El nombre es más bien confuso ya que HTTP no es un protocolo para transferir hipertexto; en lugar de eso es un protocolo para transferir información con la eficiencia necesaria para hacer que el hipertexto salte. Los datos transferidos por el protocolo pueden ser texto propiamente dicho, hipertexto, audio, imágenes o cualquier información accesible a través de Internet.

Rest (Representational State Transfer)

Rest según Flanders (2009), es una arquitectura para construir servicios en aplicaciones en red mediante el acceso a recursos. Está principalmente enfocado al uso de la web y al protocolo HTTP para realizar peticiones.

Características Rest:

- Cliente Servidor
- Basado en estados.
- Consumibles y basado en verbos.
- Basado en capas
- Opera bajo demanda

Formatos para intercambiar información

Para intercambiar información entre distintos sistemas es necesario establecer un formato de comunicación que defina la manera en que los datos son representados. Existen dos alternativas principales que permiten transmitir y compartir información: XML y JSON, para nuestro proyecto se usó JSON.

Json (JavaScript Object Notation)

Es un formato ligero para el intercambio de datos, está estandarizado por las normas RFC 7159 (RFC, Request for Comments) y ECMA-404. Se basa en una estructura atributo/valor para transmitir datos. Es comprensible para humanos y es interpretado y generado fácilmente por las máquinas, constituye una alternativa al uso de XML. Json (2005). Recuperado de http://json.org/

Ambiente de desarrollo

Está conformado por todos los programas y herramientas necesarios para crear, desarrollar, mantener e implementar el proyecto. A continuación se detalla los elementos para el desarrollo de la aplicación.

Xamarin.Forms según Reynolds (2014). Es un conjunto de herramientas de interfaz de usuario multiplataforma que permite a los desarrolladores crear fácilmente diseños de interfaz de usuario nativas que pueden ser compartidos a través de Android, iOS y Windows Phone.

Como framework permite a los desarrolladores crear interfaces de usuario rápidamente. Proporciona su propia abstracción para la interfaz de usuario que se representa utilizando controles nativos en iOS, Android, Windows o Windows Phone, lo que significa que las aplicaciones pueden compartir una gran parte de su código de interfaz de usuario y todavía conservar el aspecto nativo de la plataforma de destino.

Características de Xamarin

- Compartir código: Además de compartir un mismo lenguaje y entorno de desarrollo, podemos utilizar un mismo patrón de desarrollo.
- Completa cobertura de las APIs de iOS y Android: Tenemos todas las APIs disponibles con C#, cualquier cosa que se pueda hacer con Objective - C/Swift o Java, se puede hacer con C# y Xamarin.
- Aplicaciones nativas: Las aplicaciones desarrolladas con Xamarin son 100% nativas.

- Siempre actualizado: Xamarin suele añadir soporte el mismo día del lanzamiento oficial de una actualización.
- Open source: Tras la compra de Xamarin por parte de Microsoft, pasó a ser Open Source gratuito.

Figura 05. Compartir código con Xamarin.Forms.
Fuente: Díaz Concha, 2016 Xamarin.Forms en Acción, Aplicaciones para Acción. https://docs.google.com/viewerng/viewer?url=http://rclibros.es/wp-content/uploads/2017/05/capitulo_9788494465093.pdf&hl=es

Xamarin tradicional vs. Xamarin Forms

Xamarin tradicional: Se puede compartir toda la lógica de la aplicación entre las diferentes plataformas, a excepción de la interfaz de usuario, la cual será independiente para cada una de las mismas. En cambio, Xamarin forms añade una capa de abstracción sobre la UI que permite compartir, además de la lógica de negocio, la interfaz de usuario, aumentando consigo la reutilización de código.

Xamarin forms: La capa de abstracción que añade Xamarin forms a la UI nos facilita la tarea de crear interfaces de usuarios nativas compartidas, ya que cada uno

de los elementos de dicha abstracción son mapeados a elementos propios de cada una de las plataformas.

Tabla 01:
Diferencias Xamarin.Forms vs Xamarin

Xamarin.Forms	Xamarin.iOS / Xamarin.Android
• App de negocio. • App de datos. • App que requieran poca Funcionalidad específica de cada plataforma. • App donde compartir el código sea más importante que una IU sofisticada.	• Apps que requieran interacción muy especializada. • Apps que usen muchas APIs específicas de la plataforma. • Apps en donde la IU sea más importante que compartir el código.

Fuente: Díaz Concha, 2016 Xamarin Forms en Acción, Aplicaciones para Acción. https://docs.google.com/viewerng/viewer?url=http://rclibros.es/wp-content/uploads/2017/05/capitulo_9788494465093.pdf&hl=es

Las interfaces en Xamarin Forms se pueden definir tanto con código C# desde Code Behind, como con XAML, mi recomendación es utilizar este último para aprovechar su enfoque de separación de responsabilidades entre diseño y codificación.

XAML (eXtensible Application Markup Language) es un lenguaje declarativo basado en XML y pensado para escribir la interfaz gráfica de una aplicación de forma textual y ordenada, aparece por primera vez en la versión 3.0 del Framework de .NET.

Una de las características más importantes de XAML es que todos los elementos que definamos en este son instanciados por el CLR y quedan accesibles como objetos desde código, sin necesidad de realizar de nuevo la declaración de los mismos en Code Behind, gracias al mecanismo de las clases parciales. Xamarin

(2016), Introducing Xamarin Studio. Recuperado de https://docs.microsoft.com/es-es/xamarin/xamarin-forms/get-started/index

Plataforma Android

Según Olson, S., Hunter, J., Horgen, B., y Kenny, G. (2012). La plataforma Android necesita los siguientes elementos:

JDK (Java Development Kit)

Es un software que permite la creación de programas Java, está compuesto de herramientas como la máquina virtual de Java, el compilador de Java y las utilidades JAR (Java Archive) y documentación de Java.

Android SDK (SDK, Software Development Kit)

Es un conjunto de herramientas y bibliotecas necesarias para desarrollar aplicaciones Android, permite escoger los paquetes adecuados de acuerdo a la versión de Android requerida.

AVD Manager (AVD, Android Virtual Devices)

Es una de las herramientas del Android SDK que proporciona una interfaz gráfica para la creación de dispositivos virtuales Android de acuerdo a las versiones específicas del API instaladas. Los dispositivos virtuales son usados por el emulador de Android, incluido en el SDK, para desarrollar y probar las aplicaciones sin un dispositivo físico pero con igual funcionamiento.

Los dispositivos virtuales son genéricos con la versión de Android pero también existen implementaciones personalizadas por fabricantes de dispositivos reales que se pueden utilizar para probar aplicaciones. Existen alternativas de emuladores Android desarrollados por otras compañías, como Genymotion, que mejoran la velocidad de los emuladores originales para probar las aplicaciones.

Xamarin.Android (Mono for Android)

Es un conjunto de herramientas que permiten crear aplicaciones nativas para Android basadas en el lenguaje C#. Se basan en las implementaciones open source Mono del .NET Framework permitiendo usar las bibliotecas de clases base de .NET y además tiene enlaces para comunicarse con el API de Android. Puede ser usado por Xamarin Studio o Visual Studio.

Plataforma para iOS

Según Díaz Concha, (2016) La plataforma para iOS necesita los siguientes elementos:

Xamarin.iOS para Visual Studio

Xamarin.iOS para Visual Studio permite escribir y probar aplicaciones de iOS en equipos Windows con un equipo Mac en red que proporciona el servicio de compilación e implementación.

Logra que los desarrolladores puedan crear, compilar y depurar aplicaciones de iOS en un equipo de Windows mediante el IDE de Visual Studio. No puede hacer esto solo, no se puede crear aplicaciones de iOS sin el compilador de Apple y no se

pueden implementar sin los certificados y las herramientas de firma de código de Apple. Esto significa que una instalación de Xamarin.iOS para Visual Studio requiere una conexión a un equipo de Mac OS X para poder realizar estas tareas. Una vez configuradas, las herramientas de Xamarin harán que el proceso sea lo más directo posible.

Mac con iOS sierra 10.12 o superior

Se necesita un equipo Mac para compilar archivos IPA y no es posible implementar aplicaciones en un dispositivo sin certificados de Apple ni herramientas de firma de código. Además, el simulador de iOS solo se puede usar en un equipo Mac.

Usar un equipo Mac solo como host de compilación. En este caso simplemente se conectaría a la misma red que una máquina de Windows con las herramientas necesarias instaladas.

iOS SDK (Software Development Kit)

Contiene el código, la información y las herramientas necesarias para desarrollar, probar, ejecutar, depurar y ajustar las apps para el iOS. Dentro de este kit encontramos tres aplicaciones fundamentales:

- **Xcode:** Contiene un conjunto de herramientas para el desarrollo de las aplicaciones. Permite: editar, depurar y compilar el código fuente.
- **Interface Builder:** Permite la creación de interfaces gráfica y su vinculación con Xcode. (A partir de Xcode 4, interface builder está incorporado en la misma interfaz de Xcode)

- **iOS Simulator:** Ejecuta las aplicaciones desarrolladas en un emulador del dispositivo.

Aplicaciones multiplataforma en C#

Técnicas para usar código compartido

Según Shackles, G., (2012). C# es un lenguaje que puede ejecutarse sobre varias plataformas, es por ello que se pueden construir aplicaciones con la misma funcionalidad en diferentes entornos. Una ventaja de utilizar C# es que permite compartir código que no se relaciona directamente con la interfaz. Es decir, la lógica del negocio, que por lo general es la parte central y funcional de una aplicación puede ser escrita una vez y utilizarse el mismo código fuente a través de distintas aplicaciones. El código de la interfaz de usuario es difícil de compartir ya que cada plataforma define sus propios elementos visuales de manera única, por tanto se prefiere que la interfaz sea específica de cada plataforma, de esta manera se puede hacer uso también de las características que brinda cada ambiente.

Existen varias técnicas que permiten compartir código entre distintas aplicaciones, estas técnicas están basadas en características del lenguaje, patrones de diseño, características del compilador y funcionalidades del ambiente de desarrollo. A continuación se describen las técnicas más importantes admitidas.

Arquitectura monocross

La arquitectura multiplataforma según Olson, S., Hunter, J., Horgen, B., y Kenny, G., (2012), permite la portabilidad de código pero no en su totalidad, los

principales aspectos de conflicto son la navegación, el flujo de trabajo y en especial la interfaz de usuario. Además las aplicaciones deben aprovechar las capacidades de la interfaz de cada plataforma para mejorar la experiencia del usuario. Por tanto, es necesario separar la aplicación en distintos niveles o capas en el código. Por una parte, el código compartido, que permita incluir la mayor cantidad de código donde se ubica la lógica del negocio y el acceso a los datos. Por otra parte, el código personalizado, que involucra principalmente la interfaz de usuario y aspectos específicos de cada plataforma permitiendo una amplia personalización de la aplicación.

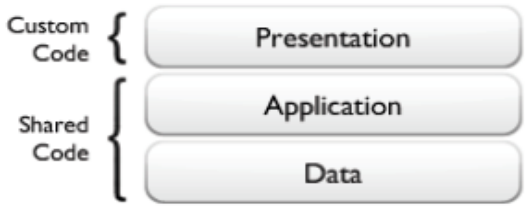

Figura 06. Separación del código compartido y código personalizado
Fuente: Monocross. (2015, Febrero) Monocross Model. Recuperado de http://monocross.net/portfolio/monocross-model.

Modelo-Vista-Controlador (MVC)

Según Mollericona (2014), El modelo vista controlador, es un patrón de arquitectura de software que aporta separando los datos de las aplicaciones de la lógica y de la interfaz de usuario obteniendo así tres componentes relacionados entre sí. Si bien la POO ayuda a clasificar aplicaciones, el MVC ayudará a separarlo por componentes.

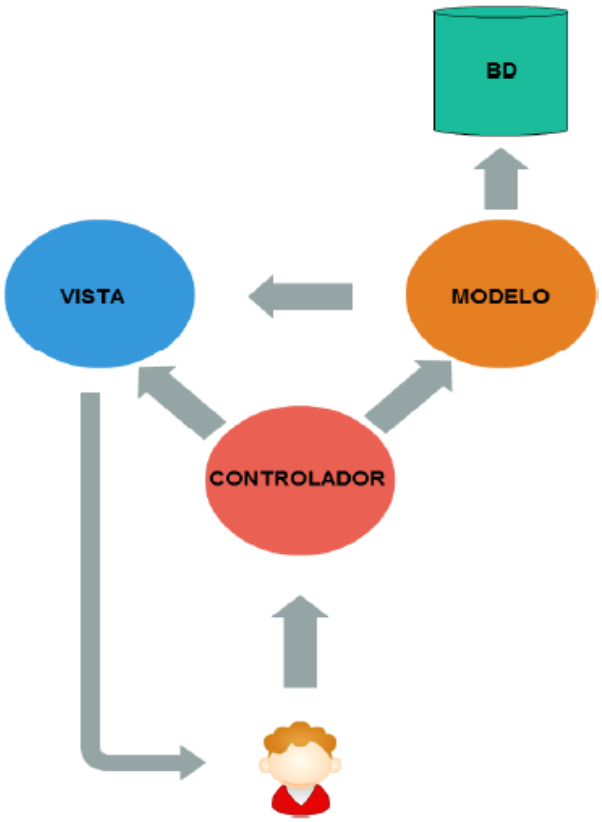

Figura 07. Diagrama Modelo Vista Controlador.
Fuente: Mollericona. (2014, Agosto). Recuperado de http://www.edsonmm.com/modelo-vista-controlador-php/Mollericona.

A continuación, paso a explicarte en que consiste cada una de estas capas:

Modelo: Es todo lo referente a la gestión de la información y la interacción con los datos de nuestra aplicación (comúnmente bases de datos) este modelo realizará acceso a dicha información como también podrá realizar actualizaciones y depuraciones de los datos gestionados. Toda petición de acceso a la información siempre pasará por esta capa.

Controlador: Este es el puente entre la vista y el modelo ya que el usuario solicitará información mediante la vista y este pasará por el controlador para

posteriormente realizar la petición al modelo, habitualmente es llamado la capa de lógica del negocio.

Vista: Esta capa mostrará la información formateada y ordenada, es el resultado de todo lo que el modelo interaccione con los datos, este lo muestra mediante la interfaz de usuario, habitualmente llamado la capa de presentación.

Modelo-Vista-Vista-Controlador (MVVM)

Según Microsoft (2017), MVVM es un patrón arquitectónico. Es una especialización del patrón de modelo de presentación introducido por Martin Fowler. También está relacionado con el patrón modelo – vista – controlador (MVC) y el patrón modelo – vista – presentador (MVP).

Una aplicación que usa MVVM separa la lógica empresarial, la interfaz de usuario y el comportamiento de la presentación.

- Los modelos representan el estado y las operaciones de los objetos empresariales que tu aplicación manipula. Por ejemplo, Hilo lee y modifica archivos de imagen, por lo que tiene sentido que los tipos de datos de los archivos de imagen y las operaciones que se realizan con archivos de imagen formen parte del modelo de Hilo.
- Las vistas contienen elementos de la interfaz de usuario, e incluyen todo el código que implementa la experiencia del usuario de la aplicación. Una vista define la estructura, el diseño y la apariencia de lo que el usuario ve en pantalla. Cuadrículas, páginas, botones y cuadros de texto son algunos ejemplos de los elementos que los objetos de vista administran.

- Los modelos de vista encapsulan el estado, las acciones y las operaciones de la aplicación. Un modelo de vista sirve como nivel de desacoplamiento entre el modelo y la vista. Proporciona los datos con un formato que la vista pueda utilizar y actualiza el modelo para que la vista no tenga que interactuar con el modelo. Los modelos de vista responden a los comandos y desencadenan eventos. También actúan como orígenes de cualquier dato que las vistas muestran. Los modelos de vista se crean específicamente para admitir una vista. Puedes pensar en un modelo de vista como en la aplicación menos la interfaz de usuario.
- Las relaciones entre una vista, un modelo de vista y un modelo son:

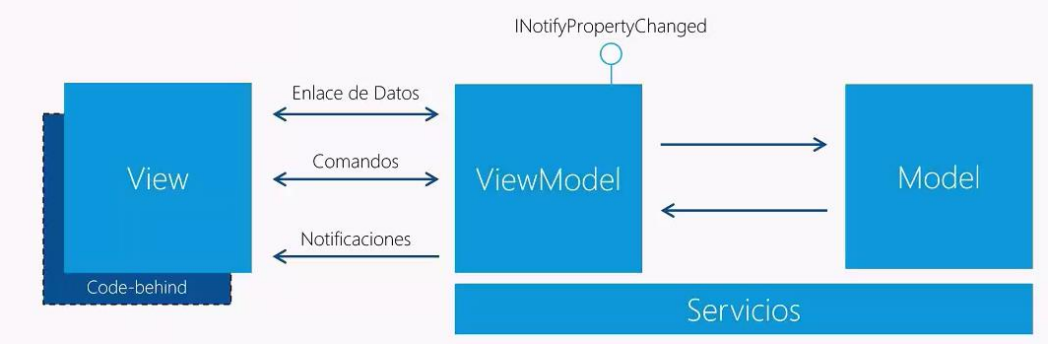

Figura 08. Modelo-Vista-Vista-Controlador
Fuente: Microsoft. (2017, Diciembre). Recuperado de https://msdn.microsoft.com/es-xl/library/windows/apps/jj160324.aspx

Finalmente, un modelo puede tener distintas representaciones de vistas, por tanto el contenedor asocia la vista con una perspectiva, que es una cadena que indica el uso de la vista y que se especifica por el controlador para presentar la vista adecuada al usuario.

La hipótesis de investigación es implícita por ser un estudio de alcance descriptivo, con el cual se determinó los procesos y se aplicaron herramientas informáticas para el desarrollo de una aplicación móvil.

La presente investigación tiene como objetivo general la implementación de una Aplicación móvil multiplataforma de integración con ERP para la visualización y aprobación de solicitudes de dinero en el grupo MOLICOM, el cual se llevara a cabo mediante los diferentes objetivos específicos detallados a continuación:

- Recopilar y analizar información sobre los principales problemas que se generan en el proceso de documento de la solicitud de dinero, para así poder establecer posibles soluciones con la aplicación móvil multiplataforma.

- Aplicar la metodología XP (Extreme Programming) para el desarrollo de la aplicación móvil.

- Construir la aplicación móvil basada en Xamarin.Forms y como gestor de base de datos el SQL Server.

- Implantar la aplicación móvil en las plataformas Android e iPhone.

II. METODOLOGÍA

El tipo de investigación de la presente tesis de acuerdo a la orientación es aplicada porque se aplica teorías que tuvieron éxito, las cuales nos ayudaran a desarrollar nuestro proyecto de investigación, se utiliza la metodología XP, la cual es muy útil y eficaz en el desarrollo de la aplicación móvil.

El presente trabajo se basa como nivel de investigación explicativa porque se explica como la variable independiente influye en la dependiente, dando un resultado favorable.

El diseño de la investigación será experimental con diseño pre – experimental porque demostrará la hipótesis a través de métodos experimentales, ya que es aquel diseño que tiene como objetivo indagar estímulo o tratamiento, y después se realiza una medición de una o más variables para observar cual es el nivel del grupo en éstas.

Grupo único con pre-prueba y post-prueba y grupo control.

Ge	O1	X	O2

Donde:

Ge: Grupo Experimental: Es el grupo al que se le aplicará el estímulo (aplicación móvil).

O1: Datos de la Pre-Prueba para los indicadores de la variable dependiente: Mediciones Inicial.

X: Aplicación Móvil: Estímulo o condición experimental.

O2: Datos de la Post-Prueba para los indicadores de la variable independiente una vez implementado la aplicación móvil.

El diseño de investigación trata de un grupo (Ge) conformado por un número representativo de aprobadores de solicitud de dinero, a quienes se le aplica una medición previa de los indicadores a ser estudiados (O1), después se implementará la Aplicación Móvil (X), para mejorar el proceso de aprobación de la solicitud de dinero y finalmente se aplicará una nueva medición de los indicadores (O2). Se espera que los valores O2 sean mejores que los valores O1. Las dos variables están constituidas de forma intencional pero representativa estadísticamente. Tanto en ausencia como la presencia de la Aplicación Móvil propuesto.

La población que se consideró para el proyecto, son las jefaturas administrativas nacionales y gerencia financiera central, quienes serán los trabajadores activos aprobadores y que hacen un total de 35 trabajadores, para luego el personal de tesorería que son los trabajadores inactivos quienes esperan la aprobación final, para realizar el pago en el Grupo Molicom.

Hallando el tamaño de la muestra de cada universo independientemente, utilizando la siguiente fórmula:

$$n = \frac{Z^2 pqN}{NE^2 + Z^2 pq}$$

Donde:

n = tamaño de la muestra.

Z = nivel de confianza.

p = variabilidad positiva.

q = variabilidad negativa.

N = tamaño de la población.

E = precisión de error.

Para nuestro estudio tomaremos:

Porcentaje de confianza de 95%

Error un 5%

Variabilidad negativa = variabilidad positiva = 0.5; pues no existen estudios anteriores de la investigación, además: la variabilidad negativa + variabilidad positiva = 1.

Entonces hallando el valor de Z, con un porcentaje de confianza de 0.95:

$$P(-Z < z < Z) = 0.95$$

Utilizando la tabla de distribución normal nos da un valor de:

$$P(-1.96 < z < 1.96) = 0.95$$

N = 35

Reemplazando en fórmula de la muestra:

$$n = \frac{(1.96)^2(0.5)(0.5)35}{500(0.05)^2 + (1.96)^2(0.5)(0.5)}$$

$$n = 32$$

Esto quiere decir que el tamaño de la muestra es de 32 trabajadores del Grupo Molicom.

Técnicas e Instrumentos de Investigación

Observación Directa:

La observación es una técnica que consiste en observar atentamente el fenómeno, hecho o caso, tomar información y registrarla para su posterior análisis.

Es un elemento fundamental de todo proceso investigativo; en ella se apoya el investigador para obtener el mayor número de datos.

Es directa cuando el investigador se pone en contacto personalmente con el hecho o fenómeno que trata de investigar.

Técnica de Encuesta:

Es uno de los métodos más utilizados en la investigación de mercados porque permite obtener amplia información de fuente primaria. Usa como instrumento el cuestionario, es un instrumento muy utilizado para recolectar los datos, consiste en un conjunto de preguntas respecto a una o más variables a medir.

Las técnicas e instrumentos en el presente trabajo de investigación fueron:

Tabla 02:
Técnicas e Instrumentos de la Investigación en el Grupo Molicom.

TÉCNICAS	**INSTRUMENTOS**	**ACCION**
Observación Directa	**Ficha de Observación:** **Anexo III:** Cuadro de observación del proceso de	Se realiza a las jefaturas nacionales de la empresa.

	aprobación de solicitud de dinero.	
Aplicación de Encuestas	**Encuestas:** **Anexo I:** Encuesta utilizada para levantamiento de información. **Anexo II:** Encuesta de satisfacción del trabajador.	Utilizado para conocer el estado del sistema de trabajo que utilizan actualmente.

Fuente: Elaboración Propia.

Procesamiento y Análisis de Información

El análisis de los datos se realizará a través del procedimiento de estadística descriptiva. Para realizar el procesamiento de la recolección de datos se procederá a tabular en una matriz de datos, de ser necesario codificando para aplicar el software MS Excel, encontrando promedios, varianza, correlación y pruebas de hipótesis.

La metodología de desarrollo de Software denominado XP (Programación extrema) consta de 4 fases, las cuales son:

- **Planificación del proyecto;** La Metodología XP plantea la planificación como un diálogo continuo entre las partes involucradas en el proyecto, incluyendo al cliente, a los programadores y a los coordinadores.
- **Diseño;** La Metodología XP hace especial énfasis en los diseños simples y claros. Los conceptos más importantes de diseño en esta metodología son los siguientes: Simplicidad, Soluciones, Recodificación y metáforas.
- **Codificación;** En esta fase se revisan temas como disponibilidad del cliente durante todo el proyecto, considerando el uso de estándares en la

programación, programación dirigida por las pruebas, programación en pares e integraciones permanentes.

- **Pruebas;** Todos los módulos deben de pasar las pruebas unitarias antes de ser liberados o publicados, la detección y corrección de errores, permite generar nuevas pruebas para verificar que el error haya sido resuelto.

III RESULTADOS

Con respecto al primer objetivo específico de la investigación se realizaron las reuniones de planificación para determinar los requerimientos funcionales y no funcionales en el proceso de solicitud de dinero en el sistema informático para que sea implementado en una aplicación móvil.

FASE I: Planificación

a. Roles

Tabla 03:
Roles de la programación extrema

Rol	Responsable
Gestor (Big Boss)	Víctor Cárdenas León
Encargado de seguimiento (Tracker)	Víctor Cárdenas León
Encargado de Pruebas (Tester)	Usuarios del Grupo Molicom
Programador	Víctor Cárdenas León
Entrenador (Coach)	Víctor Cárdenas León
Cliente	Usuarios del Grupo Molicom

Fuente: Elaboración propia.

b. Reunión de planificación

Durante las reuniones de planificación se trató las historias de usuario una a una y definiendo la prioridad para cada una en las 3 iteraciones. Los resultados obtenidos de la reunión de planificación son las historias de usuario que se listan, que incluyen su estimación, descripción y prioridad.

Esfuerzo i, donde i=1, 2, 3,... n semana(s)

Tabla 04:
Detalles de Historias de Usuario.

N° H.U.	Nombre	Prioridad	Esfuerzo	Tareas
1	Autenticar Usuario.	Alta	3	• Lectura de datos y procesado de datos. • Comprobación de resultados en la base de datos y la interfaz del usuario.
2	Administrar selección de Empresa.	Media	1	• Lectura de datos y procesado de selección de empresa. • Comprobación de resultados en la base de datos y la interfaz de solicitud de dinero.
3	Solicitud de dinero pendiente de aprobación.	Media	3	• Lectura de datos y procesado de la solicitud de dinero. • Comprobación de resultados en la base de datos y en la interfaz de solicitud de dinero.

				• Crear consulta SQL que actualice estado de la solicitud de dinero que fue aprobada.
4	Detalle de la Solicitud de dinero.	Media	1	• Lectura de datos y procesado de la solicitud de dinero. • Comprobación de resultados en la base de datos y en la interfaz de detalle de la solicitud de dinero.

Fuente: Elaboración propia.

Con respecto al segundo objetivo específico de la investigación se aplicaron las historias de usuario y las tarjetas CRC para realizar el análisis y diseño de la aplicación móvil.

c. Historias de Usuario

Tabla 05:
Historias de Usuario – Autenticar usuario.

HISTORIA DE USUARIO	
Número: 01	**Nombre:** Autenticar usuario
Usuario: Usuario	
Iteración Asignada: 1	
Prioridad : Alta	**Riesgo en Desarrollo:** Alta

Programador responsable: Cárdenas León Víctor
Descripción: El usuario tendrá acceso a un formulario LOGIN, el cual comprobará, si los datos introducidos corresponden al usuario real; previo ingreso hacia el sistema. **La Interfaz tendrá las siguientes características.** - Un formulario LOGIN. - Campos de texto con una breve descripción, de la información a ingresarse. - Un campo CONTRASEÑA que mostrará puntos en lugar de caracteres. **Eventos al presionar el botón Ingresar.** - Se verificará que todos los campos de texto estén llenos, caso contrario el sistema mostrará un mensaje de alerta, correspondiente al campo requerido, sin afectar los campos llenos. **Acciones en la selección Recordar usuario en el dispositivo.** - Una vez que se ingresa a la aplicación móvil, luego se vuelva a ingresar con esa opción se recordara el usuario.
Observaciones: Solo permite acceder a la aplicación móvil, si los usuarios previamente son registrados en el sistema integrado.

Fuente: Elaboración propia.

Tabla 06:
Historias de Usuario – Administrar selección de empresa.

HISTORIA DE USUARIO	
Número: 02	**Nombre:** Administrar selección de Empresa
Usuario: Usuario	
Iteración Asignada: 2	
Prioridad : Media	**Riesgo en Desarrollo:** Media
Programador responsable: Cárdenas León Víctor	
Descripción: El usuario tendrá opción a seleccionar la empresa por la cual va a poder visualizar las solicitudes de dinero pendientes por empresa.	
Observaciones: El usuario aprobador debe tener acceso en el sistema ERP a las aprobaciones por empresa.	

Fuente: Elaboración propia.

Tabla 07:
Historias de Usuario – Solicitud de dinero pendiente de aprobación.

HISTORIA DE USUARIO	
Número: 03	**Nombre:** Solicitud de dinero pendiente de aprobación
Usuario: Usuario	
Iteración Asignada: 2	
Prioridad : Media	**Riesgo en Desarrollo:** Media
Programador responsable: Cárdenas León Víctor	

Descripción: El usuario aprobador accederá a la visualización de solicitudes pendientes por aprobar compuesta por una cabecera, cuerpo y pie de acuerdo a la empresa seleccionada. **Estructura y funcionalidades de los formularios:** - **Cabecera:** tendrá campos de texto con los datos de acceso, como usuario y empresa, también tendrá un campo de búsqueda por número de solicitud. - **Cuerpo:** Tendrá campos de texto con una breve descripción de la información obtenida por cada solicitud de dinero pendiente de aprobar, también tendrá campo de selección para aprobar solicitud de dinero. - **Pie:** Constará de un campo de selección de todas las solicitudes de dinero y un botón Aprobar. **Características generales** - El listado de solicitud de dinero esta ordenado por fecha de emisión de manera descendente. - Se Utilizará colores sencillos para el diseño y maquetado. - Los campos de texto dispondrán de una breve descripción de la información. - Si no se selecciona ninguna solicitud de dinero para aprobar y se presiona el botón Aprobar mostrara alerta "Seleccione por lo menos una solicitud"
Observaciones: - La empresa debe ser escogida por el usuario para que muestre las solicitudes pendientes de aprobación por empresa seleccionada. - Se verificará que la solicitud de dinero seleccionada a aprobar este pendiente de aprobación y que el usuario tiene acceso a aprobarla. - Si se sobrepasa el tiempo de inactividad permitido que es 10 minutos, mostrará alerta y te regresara al Login. - Cada vez que se apruebe una solicitud de dinero en la aplicación móvil, se debe observar aprobado en el Sistema Integrado ERP.

Fuente: Elaboración propia.

Tabla 08:
Historias de Usuario – Detalle de solicitud de dinero.

HISTORIA DE USUARIO	
Número: 04	**Nombre:** Detalle de Solicitud de dinero
Usuario: Usuario	
Iteración Asignada: 3	
Prioridad : Media	**Riesgo en Desarrollo:** Media
Programador responsable: Cárdenas León Víctor	
Descripción: mostrará detalle de la solicitud de dinero seleccionada, la interfaz está compuesta por su cabecera y detalle.	

Estructura y funcionalidades de los formularios: - **Cabecera:** tendrá campos de texto con los datos de la solicitud de dinero, como empresa, numero de solicitud, centro de responsabilidad, beneficiario, responsable, fecha de emisión, total y símbolo de moneda. - **Detalle:** mostrará el detalle de los documentos de gasto del beneficiario. **Características generales** - Los campos de texto como son la serie, documento, moneda e importe se mostrara de color azul. - Los campos de texto dispondrán de una breve descripción de la información.
Observaciones:

Fuente: Elaboración propia.

d. Velocidad del proyecto

Durante el desarrollo la velocidad del proyecto se mantuvo casi constante, a pesar de que no todas las historias de usuario tenían un diferente nivel de dificultad y por lo tanto el número de horas también. Por esto se encontró que mientras en la segunda y tercera iteración se trabajaron menos horas semanales en comparación con la primera iteración, también fue donde más tareas de historias de usuario se realizaron. El motivo de este resultado fue el nivel de dificultad de la primera iteración y por lo tanto el número de horas requeridas en la primera iteración fueron mayores en todo el proyecto. Esto se resume en la tabla N° 09.

Tabla 09:
Velocidad del proyecto

	Iteración 1	**Iteración 2**	**Iteración 3**
Historias de Usuario	1	2	1
Semanas	4	1	1
Horas semanales	16	12	12
Total de Horas x Semana	**64**	**12**	**12**

Fuente: Elaboración propia.

Por lo cual la velocidad (promedio) del proyecto estaría dado por:

(1+2+1) / 3 = 1.3 hu/iteración.

Un problema que se presentó con la velocidad del proyecto fue el refactoring, ya que en la tercera iteración surgieron varias recomendaciones por parte del cliente que no se había considerado dentro de la media de velocidad.

e. Entregas funcionales

Debido a que las iteraciones tenían una duración de alrededor de 1 mes, fue al término de este plazo que se realizaron las entregas, las cuales siempre fueron funcionales, lo que quiere decir que al momento de la entrega estaban en condiciones para que pase a producción.

Tabla 10:
Fechas de entregas funcionales

Iteración	Fecha	Duración
Primera	20/10/2018	1:30 horas
Segunda	11/11/2018	1:00 hora
Tercera	15/12/2018	1:00 hora

Fuente: Elaboración propia.

En las reuniones con los usuarios finales se hizo la entrega y explicación de cómo usar correctamente las funcionalidades de la aplicación móvil, buscando la aprobación del usuario final y sus observaciones para el refactoring.

FASE II: Diseño

a. Simplicidad

XP sugiere que el diseño debe ser sencillo y que solo se deben crear diagramas útiles, por lo que se utilizó la recomendación de XP de solo invertir el tiempo

necesario en la elaboración de diagramas y en un correcto diseño de interfaz gráfica. Para la interfaz de usuario, se invirtió mucho tiempo en su diseño, por lo que solo se ubicaron los elementos tal como los definió el usuario. Como consecuencia el cliente se mostró conforme con la apariencia visual de la aplicación móvil.

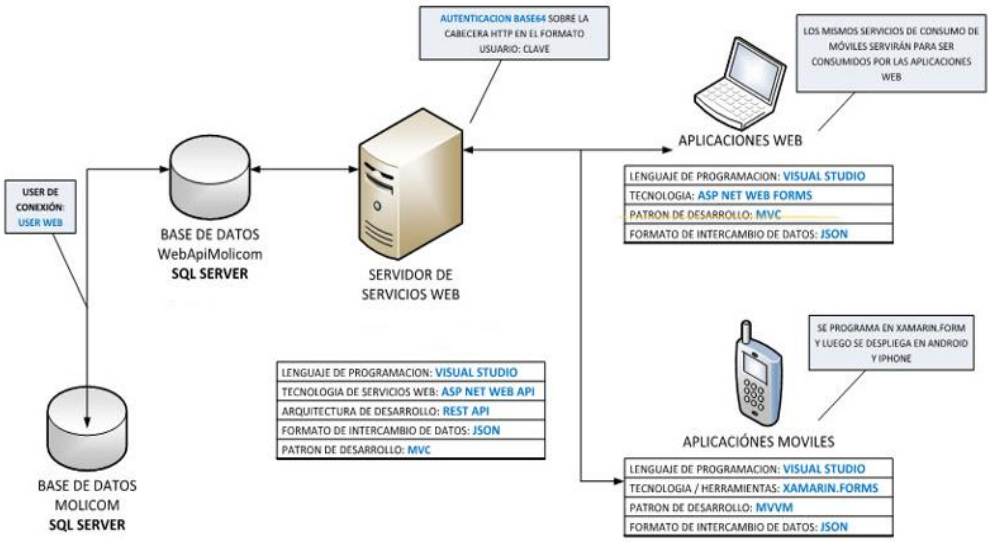

Figura 09. Arquitectura general servicios Web y Aplicaciones móviles.
Fuente: Elaboración propia.

b. Diagrama de clases.

Nos representa registro de la información de los elementos que intervienen en el problema y sus relaciones.

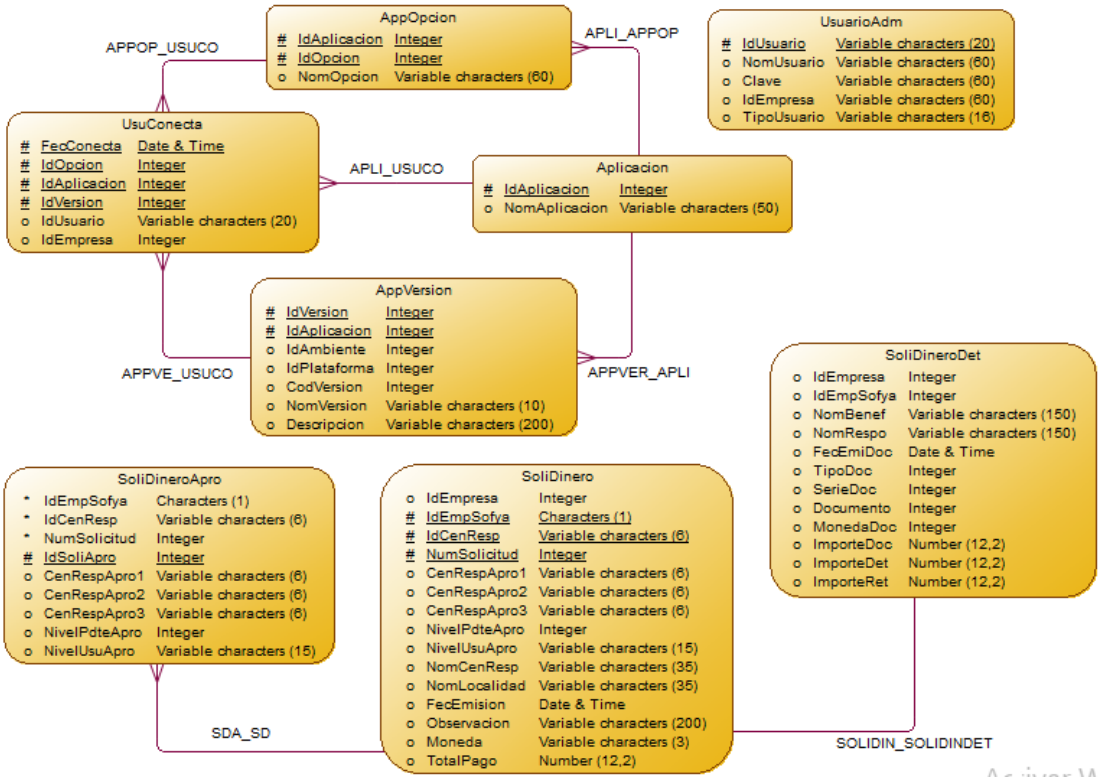

Figura 10. Diagrama de clases del proceso de aprobación de una SD en Aplicación móvil.
Fuente: Elaboración propia.

c. Tarjetas Clase Responsabilidad Colaboración

Tabla 11:
Tarjeta CRC – Conexión de usuario a la App.

TARJETA CRC		
Número: 01	**Escenario:** Auditoria de eventos que realiza el usuario.	
Nombre CRC: UsuConecta		
Responsabilidades - Guarda auditoria de usuario de acuerdo a las acciones que realice en la aplicación móvil.	**Colaboradores** - Usuario.	**Métodos** - Guardar auditoria de usuario.
Observaciones: Los usuarios registrados son aquellos que podrán acceder a la aplicación móvil.		

Fuente: Elaboración propia.

Tabla 12:
Tarjeta CRC – Opciones de la App.

TARJETA CRC		
Número: 02	**Escenario:** Mantenimiento de Opciones que tiene la aplicación móvil.	
Nombre CRC: AppOpcion		
Responsabilidades - Guardar las opciones de la aplicación.	**Colaboradores** - Aplicación.	**Métodos** - Obtener opción. - Guardar opción.
Observaciones: muestra las opciones de la aplicación.		

Fuente: Elaboración propia.

Tabla 13:
Tarjeta CRC – Aplicación de la App.

TARJETA CRC		
Número: 03	**Escenario:** Mantenimiento de Aplicación.	
Nombre CRC: Aplicación		
Responsabilidades - Guardar nombre de la Aplicación.	**Colaboradores** - Opción. - Versión.	**Métodos** - Obtener Aplicación. - Guardar Aplicación.
Observaciones: mostrara el nombre de la aplicación.		

Fuente: Elaboración propia.

Tabla 14:
Tarjeta CRC – Versión de la App.

TARJETA CRC		
Número: 04	**Escenario:** Mantenimiento de Versión de Android y Iphone.	
Nombre CRC: AppVersion		
Responsabilidades - Guarda la versión de la Aplicación por plataforma en Android e iPhone.	**Colaboradores** - Aplicación. - Versión.	**Métodos** - Obtener Versión. - Guardar Versión.
Observaciones: Se ingresa las versiones de cada plataforma en la tabla de a la base de datos mediante un insert.		

Fuente: Elaboración propia.

Tabla 15:
Tarjeta CRC – Mantenimiento de Usuario Acceso

TARJETA CRC		
Número: 05	**Escenario:** Acceso a usuarios	
Nombre CRC: UsuarioAdm		
Responsabilidades - Guarda los usuarios que tienen acceso a la aplicación.	**Colaboradores**	**Métodos** - Obtener Usuario. - Guardar Usuario.
Observaciones: Se ingresa los usuarios a la tabla de a la base de datos mediante un insert.		

Fuente: Elaboración propia.

Tabla 16:
Tarjeta CRC – Consulta de solicitud de dinero pendiente.

TARJETA CRC		
Número: 06	**Escenario:** Consulta solicitud de dinero pendiente.	
Nombre CRC: SoliDinero		
Responsabilidades - Obtener solicitudes de dinero pendientes de aprobación.	**Colaboradores**	**Métodos** - Consultar solicitudes de dinero pendientes. - Validar Solicitud de dinero.
Observaciones: Las solicitudes de dinero mostradas son aquellas que el usuario tiene pendientes por aprobar.		

Fuente: Elaboración propia.

Tabla 17:
Tarjeta CRC – Detalle de solicitud de dinero.

TARJETA CRC		
Número: 07	**Escenario:** Detalle de solicitud la dinero.	
Nombre CRC: SoliDineroDet		
Responsabilidades - Obtener el detalle de la solicitud de dinero seleccionada.	**Colaboradores** - SoliDinero.	**Métodos** - Consultar detalle de solicitudes de dinero.

Observaciones: El detalle de las solicitudes de dinero mostradas son aquellas que el usuario selecciono.

Fuente: Elaboración propia.

Tabla 18:
Tarjeta CRC – solicitud de dinero Aprobada.

TARJETA CRC		
Número: 08	**Escenario:** Solicitud de dinero aprobada.	
Nombre CRC: SoliDineroApro		
Responsabilidades - Obtener solicitudes de dinero pendientes de aprobación. - Guardar solicitud aprobada.	**Colaboradores** - SoliDinero**.**	**Métodos** - Guardar solicitudes de dinero pendientes. - Validad solicitud de dinero. - Actualizar Solicitud de dinero.
Observaciones: Las solicitudes de dinero mostradas son aquellas que el usuario aprobó y serán actualizadas en la base de datos del sistema integrado.		

Fuente: Elaboración propia.

d. Refactoring

Durante el desarrollo de la aplicación, se observó que surgieron situaciones que no fueron tomadas en cuenta al inicio del proyecto por lo que la forma de superar estos incidentes es con la refactorización, en la cual se buscan mejorar la codificación, pero manteniendo la funcionalidad y tratando de conservar la simplicidad del código.

Una de estas situaciones se refirió a la decisión que se tomó de no crear la función de seleccionar la empresa luego del login de usuario y las medidas tomadas para superar este error no empleo más de 4 horas para su solución.

Con respecto al tercer objetivo específico de la investigación se realizó la codificación y pruebas de la aplicación móvil.

FASE III: Codificación

a. Cliente siempre presente

En el caso de estudio, como no siempre se tenía al usuario final en contacto con el equipo de desarrollo, se optó por contactarse vía telefónica para los días en los cuales no se podía contar con la presencia del usuario final para poder solucionar dudas respecto a las historias de usuario en desarrollo. Si bien no se cumple con lo señalado por la metodología, fue suficiente para lograr una buena comunicación con el usuario final.

b. Estándares en el código

Los estándares son una buena práctica para el desarrollo de software el cual no solo se debe utilizar con la metodología XP sino también al aplicar otra metodología. Al aplicar estándares se buscó facilitar la comprensión en el código para el desarrollo.

- Estándares en la Base de Datos:
  - Los nombres de las tablas se escribieron en minúscula.
  - Los nombres de los campos se escribieron en minúscula.
- Estándares en el código:

- Los nombres de los elementos visuales tienen el mismo nombre e identificación.

- El código debe estar tabulado correctamente

FASE IV: Pruebas

La metodología XP se centra en la ejecución de pruebas a lo largo del proyecto, con el fin de asegurar la realización de lo planificado al inicio de cada iteración. En este proceso participó el equipo de desarrollo junto con el usuario final con sus aportes sobre todo en las pruebas de aceptación. Las pruebas están estrechamente relacionadas con la planificación de iteraciones.

Pruebas de aceptación

XP sugiere que se deben diseñar con base a los requerimientos capturados de las historias de usuario, para lo cual cada una de las historias de usuarios seleccionadas deberá tener una prueba de aceptación. Estas pruebas son de caja negra porque representan el resultado de una determinada transacción en el sistema.

Estas pruebas fueron diseñadas por el usuario final, pero con el apoyo de los programadores para poder guiar a los usuarios finales en un correcto diseño de las pruebas y que al final se valide la funcionalidad de la mejor manera. Para que una historia de usuario se considere aprobada, deberá pasar todas las pruebas de aceptación elaboradas para dicha historia.

Casos de prueba

Según XP se debe realizar un caso de prueba por cada historia de usuario, los cuales fueron ejecutados al final de cada iteración según las historias implementadas en los planes de entrega.

A continuación se detallan las pruebas de aceptación para ello se procedió a seleccionar dos historia de usuario que fueron realizados sobre la aplicación móvil y con integración de todos los módulos.

a. Especificación de Prueba: Autenticación de Usuario

Descripción

En esta historia hay que comprobar el registro de datos del usuario en la base de datos, un nuevo usuario debe estar registrado en el Sistema Integrado ERP, así como se debe validar la conexión al servidor de servicios. Si al introducir un dato del usuario que no es correcto se indica al usuario el incidente y no se permite el ingreso. También se debe comprobar, en el caso de que la validación del usuario sea correcta, el registro de ingreso del usuario serán almacenados en la base de datos.

Introducción correcta de Usuario.

Descripción

El usuario deberá ingresar su usuario y clave del sistema integrado ERP, una vez ingresado se le mostrará un listado con las empresas del grupo donde podrá seleccionar y se procederá con el listado de solicitudes de dinero pendientes de aprobación.

Condiciones de ejecución

El usuario deberá estar dado de alta (registrado) en el sistema integrado ERP.

Entrada

- Aparecerá un formulario donde deberá ingresar los datos de usuario y contraseña.
- Luego se pulsara el botón “Ingresar”.
- Si los datos son correctos mostrara listado de empresa a seleccionar.
- En el listado principal se mostrara las solicitudes pendientes de aprobación.

Resultado esperado

Tras la introducción de autenticación de usuario, si el procesado ha sido correcto, en la base de datos aparecerán los datos de los usuarios conectados a la aplicación móvil.

Evaluación de la prueba

Prueba satisfactoria.

Introducción de usuario con errores

Descripción

Si al confirmar los datos ingresados ocurre algún error se indicará al usuario del error de procesado (muestra el mensaje: usuario incorrecto o contraseña incorrecta), también si no hubiera una correcta conexión con el servidor de servicios o de base de datos mostrara una alerta.

Condiciones de ejecución

El usuario deberá estar dado de alta en el sistema.

Entrada

- Aparecerá un formulario donde deberá ingresar los datos de usuario y contraseña pero intencionalmente ingresa un campo vacío.
- Después pulsa el botón "Ingresar”.
- En el caso de que ocurra algún error en la validación de los datos mostrará un mensaje indicando que debe llenar todos los campos obligatorios.
- El proceso de introducción de usuario se considera como finalizado.
- Los datos quedan en la vista para que vuelva a ser llenado.

Resultado esperado

Los usuarios incorrectos no son introducidos en la base de datos y no permite el acceso a la aplicación móvil.

Evaluación de la prueba

Prueba satisfactoria.

V. Resultados de cada interacción

a. Primera Iteración

Plan de entrega

Consta de 1 historias de usuario y de las tareas que se deben realizar para cada historia, las cuales se resumen en la tabla 19.

Tabla 19:
Cronograma de Actividades, Iteración I.

Plan de Entrega **Historia de usuario**	**Tareas**
Autenticar usuario	- Configuración de un entorno de desarrollo web. - Creación de scripts utilizando lenguaje SQL. - MAPEO OBJETO-RELACIONAL de las entidades. - Seguridad de autenticación de usuario en los servicios web. - Desarrollo de interfaz de login de usuario.

Fuente: Elaboración propia.

1. Configuración de un entorno de desarrollo web.

Tabla 20:
Autenticación de usuario - Tarea N° 1.

Número Tarea: 01	**Numero de Historia:** 1
Nombre de Tarea: Configuración de un entorno de desarrollo web.	
Tipo de tarea: General	**Tiempo estimado:** 8 horas
Fecha inicio: 06/08/18	**Fecha fin:** 06/08/18
Programador responsable: Cárdenas León Víctor	
Descripción: La presente tarea describe el proceso de instalación y configuración de un entorno de desarrollo web, utilizando una arquitectura Api web. **Componentes a Implementar:** Implementar un servicio web utilizando los servicios IIS dentro de un sistema window server, esta característica permite convertir su máquina en un servidor web, IIS ofrece soporte para los siguientes protocolos: HTTP, HTTPS, FTP, FTPS, SMTP y NNTP. Antes de activar el modulo IIS se tiene que configurar el framework primero. Luego se tiene que configurar el administrador del servidor (Server Manager) agregar roles y características del Web Server (IIS). Se puede probar si fue exitosa la instalación y está activo el IIS, de la siguiente manera dentro del servidor habrá el explorador: http://localhost/ Si la instalación es correcta, se mostrará la página principal de Microsoft IIS. Abrir el Administrador del servidor. En el menú seleccione la opción "Herramientas" (Tools) y haga clic en Administrador de Internet Information Services (IIS), para luego indicar el nombre del sitio web, especifique la ruta en la cual posee los archivos del sitio web.	

Plataforma de desarrollo: Windows Server Standard, Service Pack 2. **Servicios:** Web Server IIS. **Tecnología:** ASP NET WEBAPI

Fuente: Elaboración propia.

2. Creación de scripts utilizando lenguaje SQL

Tabla 21:
Autenticación de usuario - Tarea N° 2.

Número Tarea: 02	**Numero de Historia:** 1
Nombre de Tarea: Creación de scripts utilizando lenguaje SQL.	
Tipo de tarea: Administrador de base de datos	**Tiempo estimado:** 8 horas
Fecha inicio: 07/08/18	**Fecha fin:** 07/08/18
Programador responsable: Cárdenas León Víctor	
Descripción: La presente tarea describe el proceso de creación de vistas mediante script SQL, se creó script para obtener las tablas SoliDinero, SoliDineroDet, SoliDineroApro, trayendo datos mediante un Select de las tablas de la base de datos origen para la creación de vistas en la base de datos móvil, así poder mapear las entidades al servicio web para que puedan ser usados.	

Fuente: Elaboración propia.

3. MAPEO OBJETO-RELACIONAL de las entidades.

Tabla 22:
Autenticación de usuario - Tarea N° 3.

Número Tarea: 03	**Numero de Historia:** 1
Nombre de Tarea: mapeo objeto-relación de las entidades.	
Tipo de tarea: General	**Tiempo estimado:** 8 horas
Fecha inicio: 08/08/18	**Fecha fin:** 09/08/18
Programador responsable: Cárdenas León Víctor	
Descripción: La presente tarea describe el proceso de mapeo de las entidades, aplicando el paradigma de la PROGRAMACIÓN ORIENTADA A OBJETOS, para poder reutilizar el código fuente y acceder hacia la información de una manera rápida y eficiente. Convertimos nuestra tabla UsuarioAdm, en su respectivo objeto encapsulado, utilizando una clase, para poder instanciarla y acceder a sus respectivos atributos y métodos. Así como las tablas SoliDinero, SoliDineroApro y SoliDineroDet.	

Fuente: Elaboración propia.

4. Seguridad de autenticación de usuario en los servicios web.

Tabla 23:
Autenticación de usuario - Tarea N° 4.

Número Tarea: 04	**Numero de Historia:** 1
Nombre de Tarea: Seguridad de autenticación de usuario en los servicios web.	
Tipo de tarea: General	**Tiempo estimado:** 60 horas
Fecha inicio: 09/08/18	**Fecha fin:** 22/08/18
Programador responsable: Cárdenas León Víctor	
Descripción: La presente tarea describe el proceso de seguridad de acceso a la aplicación móvil de un usuario, usando el lenguaje de programación C# y el lenguaje de marcado XAML el cual es usado para crear las interfaz de usuario, se crea una división explícita entre interfaz gráfica y lógica, o Vista y Controlador en términos de la arquitectura MVC. Se crea controles de seguridad para la autenticación, desencriptado y decodificación del usuario. También se crea controlador UsuarioAdmController.cs, que es el que va obtener el usuario usando GET a la entidad UsuarioAdm, cada vez que se obtiene un usuario se hace un llamado antes a su validación de autorización.	

Fuente: Elaboración propia.

5. Desarrollo de interfaz de usuario.

Tabla 24:
Autenticación de usuario - Tarea N° 5.

Número Tarea: 05	**Numero de Historia:** 1
Nombre de Tarea: desarrollo de interfaz de usuario	
Tipo de tarea: General	**Tiempo estimado:** 40 horas
Fecha inicio: 24/08/18	**Fecha fin:** 31/08/18
Programador responsable: Cárdenas León Víctor	

Descripción:
La presente tarea describe el proceso de desarrollo de interfaz de la usuario, mediante patrón de diseño MVVM(Model-View-ViewModel), se desarrolló LoginView.xaml usando View(Vista) es una página Xaml que debe tener su correspondiente clase ViewModel LoginViewModel.cs contiene la lógica del negocio, debe estar asociado a un View para notificarle los cambios.
 	
Modelo:
Son las clases que representan el modelo del negocio, tiene como entidad la tabla UsuarioAdm. Sirven para crear un enlace entre la capa de negocios y la de datos. El modelo es el siguiente:
 	
`public class UsuarioAdm`
`{`	

```
        // Propiedades inherentes de entidad

        public string IdUsuario { get; set; }

        public string NomUsuario { get; set; }

        public int IdEmpresa { get; set; }

        public string Empresa { get; set; }

        public string TipoUsuario { get; set; }

        // Propiedades para traslado de información

        public string Clave { get; set; }

        public int SupervisaGrupo { get; set; }

        public List<UsuarioAdm> ListaUsuario { get; set; }
    }
```

Existe una capa adicional, llamada CAPA DE SERVICIOS, que es una capa adicional de la lógica del negocio, y contiene: servicios de almacenamiento en base de datos local, consumo de Api.
Entre nuestra aplicación móvil y la API REST se creó una clase dentro de la carpeta SERVICES llamada **ApiService.cs**, la cual representa a nuestra capa de datos y al punto central por el cual cualquier módulo que quiera leer o escribir a la BD debe pasar. El procedimiento estándar de comunicación consiste en **Serializar** y **Deserializar.**

Se hace un llamado al servicio Api realiza una deserialización del formato JSON y envía lista de usuario.

Fuente: Elaboración propia.

Resultados de la Primera Iteración.

Figura 10. Interfaz Login de usuario.
Fuente: Elaboración propia.

b. Segunda Iteración

Plan de entrega

Consta de 2 historias de usuario y de las tareas que se deben realizar para cada historia, las cuales se resumen en la tabla 19.

Tabla 25:
Cronograma de Actividades, Iteración II.

Plan de Entrega Historia de usuario	**Tareas**

Administrar selección de Empresa.	- Desarrollo de interfaz de administración de empresa. - Menú de aplicación móvil.
Solicitud de dinero pendiente de aprobación.	- Desarrollo de servicio web para las solicitudes de dinero pendientes de aprobación. - Desarrollo de interfaz de solicitud de dinero pendiente de aprobación.

Fuente: Elaboración propia.

Historia 2 - Administrar selección de Empresa.

1. Desarrollo de interfaz de administración de empresa.

Tabla 26:
Administrar selección de empresa - Tarea N° 1.

Número Tarea: 01	**Numero de Historia:** 2
Nombre de Tarea: desarrollo de interfaz de administración de empresa	
Tipo de tarea: General	**Tiempo estimado:** 8 horas
Fecha inicio: 17/09/18	**Fecha fin:** 17/09/18
Programador responsable: Cárdenas León Víctor	
Descripción: La presente tarea describe el proceso de desarrollo para la pantalla de selección de empresa, luego del acceso de usuario. Crear un listado de selección, el cual nos permitirá mostrar la información de las solicitudes pendientes por aprobar por empresa, para la pantalla de selección de empresa, se trae los datos de usuario para luego mostrar los nombres de la empresa en un arreglo dentro de un método DisplayActionSheet, que es la hoja de acción de visualización en formularios de Xamarin.	

Fuente: Elaboración propia.

2. Menú de la aplicación móvil.

Tabla 27:
Administrar selección de empresa - Tarea N° 2.

Número Tarea: 02	**Numero de Historia:** 2
Nombre de Tarea: Menú de la aplicación móvil.	
Tipo de tarea: General	**Tiempo estimado:** 16 horas
Fecha inicio: 18/09/18	**Fecha fin:** 19/09/18

Programador responsable: Cárdenas León Víctor

Descripción:
La presente tarea describe el proceso de desarrollo del menú de la aplicación móvil, mediante patrón de diseño MVVM(Model-View-ViewModel), se desarrolló un View(Vista) MenuView.xaml y MainView.xaml que crea una propiedad de tipo MasterDetailPage, que luego define cual será la pantalla Detail dentro del Menú, que debe tener su correspondiente clase ViewModel MainViewModel contiene la lógica del negocio, debe estar asociado a un View para notificarle los cambios.
 	
Modelo: El modelo es el siguiente:
 	
`public class Menu`
`{`
`// Propiedades items de menú`
 	
`public string Icono { get; set; }`
 	
`public string Titulo { get; set; }`
 	
`public string NombrePage { get; set; }`
`}`

Fuente: Elaboración propia.

Historia 3 - Solicitud de dinero pendiente de aprobación.

3. Desarrollo de servicio web para las solicitudes pendientes de aprobación.

Tabla 28:
Solicitud de dinero pendiente de aprobación - Tarea N° 1.

Número Tarea: 01	**Numero de Historia:** 3
Nombre de Tarea: Desarrollo de servicio web para las solicitudes de dinero pendientes de aprobación.	
Tipo de tarea: General	**Tiempo estimado:** 32 horas
Fecha inicio: 20/09/18	**Fecha fin:** 21/09/18
Programador responsable: Cárdenas León Víctor	
Descripción: La presente tarea describe el proceso de desarrollo de servicios web de la solicitud de dinero, Representational State Transfer (REST) es una arquitectura que define una serie de principios sobre los cuales podemos diseñar servicios web, utiliza HTTP como medio de transporte y JSON como lenguaje interoperacional. JSON es un formato abierto expresado en texto plano para transmitir objetos representados como pares de atributos y llaves. El prototipo implementado en este trabajo utiliza Servicios REST para la comunicación entre cliente y servidor.	

Para probar la ejecución del controlador SoliDinero (servicio web api) se usó POSTMAN, que debe ser asociado a una cuenta Gmail y se puede gestionar una ejecución GET y POST.

Configuración Accept:
type : Basic Auth
Haders : Content Type = Aplication/Json

Ejemplo: IP/puerto/controlador
http://128.1.41.2:8084/api/SoliDinero?idempresa=1&idCenResp=72112&idSoliDinero=1286

Los formatos son usados por el servidor para solicitar y responder mensajes, estos formatos son llamados “Media Type Formatters”.

Una vez ejecutado el comando, podemos acceder a una representación gráfica de nuestra API REST a través de un navegador, Para acceder al contenido de nuestra base de datos basta con especificarlo en la URL.

Ejemplo: texto plano en el lenguaje JSON del servicio SoliDineroApro.


```
[
{
"IdEmp":2,
"IdSerie":1,
"NumSoliCot":2101,
"Nano":2012,
"NumCorrel":15,
"UsuApro":"VICTORCL"
}
]
```


Fuente: Elaboración propia.

4. Desarrollo de interfaz de solicitud pendiente de aprobación.

Tabla 29:
Solicitud de dinero pendiente de aprobación - Tarea N° 2.

Número Tarea: 02	**Numero de Historia:** 3
Nombre de Tarea: Desarrollo de interfaz de solicitud de dinero pendiente de aprobación.	
Tipo de tarea: General	**Tiempo estimado:** 32 horas
Fecha inicio: 19/09/18	**Fecha fin:** 26/09/18
Programador responsable: Cárdenas León Víctor	
Descripción:	

La presente tarea describe el proceso de desarrollo de interfaz de la solicitud de dinero, mediante patrón de diseño MVVM(Model-View-ViewModel), se desarrolló SoliDineroView.xaml usando View(Vista) es una página Xaml que debe tener su correspondiente clase ViewModel SoliDineroViewModel.cs contiene la lógica del negocio, debe estar asociado a un View para notificarle los cambios.

Modelo:
Son las clases que representan el modelo del negocio, tiene como entidad la tabla SoliDinero y SoliDineroApro. Sirven para crear un enlace entre la capa de negocios y la de datos. Los modelos son los siguientes:

```
public class SoliDinero
    {
        public int IdEmpresa { get; set; }

        public string IdEmpSofya { get; set; }

        public string IdCenResp { get; set; }

        public int NumSolicitud { get; set; }

        public string CenRespApro1 { get; set; }

        public string CenRespApro2 { get; set; }

        public string CenRespApro3 { get; set; }

        public int NivelPdteApro { get; set; }

        public string NivelUsuApro { get; set; }

        public string NomCenResp { get; set; }

        public string NomLocalidad { get; set; }

        public DateTime FecEmision { get; set; }

        public string Observacion { get; set; }

        public string Moneda { get; set; }

        public double TotalPago { get; set; }

        // Propiedades computadas

        public string CentRespCompleto => $"{IdCenResp:00000} -
{NomCenResp}";

        // Propiedades de selección

        public bool IsSelected { get; set; }

        // Propiedades para traslado de información
```

```
        public string IdUsuario { get; set; }

        public string Clave { get; set; }

        public string Empresa { get; set; }

    }

public class SoliDineroApro
    {
        public string IdEmpSofya { get; set; }

        public string IdCenResp { get; set; }

        public int NumSolicitud { get; set; }

        public string CenRespApro1 { get; set; }

        public string CenRespApro2 { get; set; }

        public string CenRespApro3 { get; set; }

        public int NivelPdteApro { get; set; }

        public string NivelUsuApro { get; set; }
    }
```

Existe una capa adicional, llamada CAPA DE SERVICIOS, que es una capa adicional de la lógica del negocio, y contiene: servicios de almacenamiento en base de datos local, consumo de Api.

Entre nuestra aplicación móvil y la API REST se creó una clase dentro de la carpeta SERVICES llamada **ApiService.cs**, la cual representa a nuestra capa de datos y al punto central por el cual cualquier módulo que quiera leer o escribir a la BD debe pasar. El procedimiento estándar de comunicación consiste en **Serializar** y **Deserializar.**

La acción de leer **GetSoliDinero()** realiza una deserialización ya que muestra información, obtenida a través de una acción HTTP **GET** dirigida a nuestra API REST, se encuentra en formato JSON y queremos convertirla a una representación en memoria, más específicamente, a una lista con un de objetos del tipo **SoliDinero** para poder manipularla dentro de nuestro programa.

Para **SoliDineroApro** se aplicó un POST enviando un listado a la tabla SoliDineroApro donde se implementó un disparador (Trigger) en la base de datos móvil que actualiza el estado de la solicitud de dinero a aprobado en la base de datos origen.

Fuente: Elaboración propia.

Resultados de la Segunda Iteración

Figura 11. Administrar selección de empresa.
Fuente: Elaboración propia.

Figura 12. Menú de aplicación móvil.
Fuente: Elaboración propia.

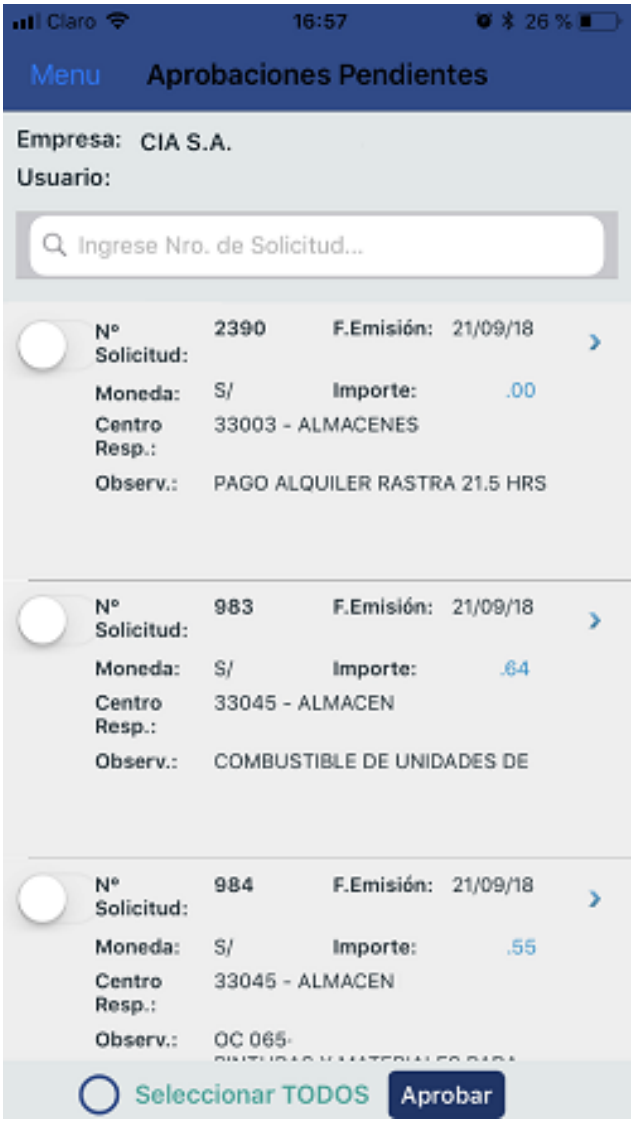

Figura 13. Listado de solicitud de dinero pendientes.
Fuente: Elaboración propia.

c. Tercera Iteración

Plan de entrega

Consta de 1 historias de usuario y de las tareas que se deben realizar para cada historia, las cuales se resumen en la tabla 30.

Tabla 30:
Cronograma de Actividades, Iteración III.

Plan de Entrega Historia de usuario	**Tareas**

Detalle de Solicitud de dinero.	- Desarrollo de servicio web para el detalle de solicitud de dinero pendiente de aprobación. - Desarrollo de interfaz de detalle de la solicitud de dinero. - Publicación de servicios web.

Fuente: Elaboración propia.

1. Desarrollo de servicio web para las solicitudes pendientes de aprobación.

Tabla 31:
Solicitud de dinero pendiente de aprobación - Tarea N° 2.

Número Tarea: 01	**Numero de Historia:** 3
Nombre de Tarea: Desarrollo de servicios web para el detalle de solicitud de dinero pendiente de aprobación.	
Tipo de tarea: General	**Tiempo estimado:** 32 horas
Fecha inicio: 27/09/18	**Fecha fin:** 31/09/18
Programador responsable: Cárdenas León Víctor	
Descripción: La presente tarea describe el proceso de desarrollo de servicios web del detalle de la solicitud de dinero, Representational State Transfer (REST) es una arquitectura que define una serie de principios sobre los cuales podemos diseñar servicios web, utiliza HTTP como medio de transporte y JSON como lenguaje interoperacional. JSON es un formato abierto expresado en texto plano para transmitir objetos representados como pares de atributos y llaves. El prototipo implementado en este trabajo utiliza Servicios REST para la comunicación entre cliente y servidor. Para probar la ejecución del controlador SoliDineroDet (servicio web api) se usó POSTMAN, que debe ser asociado a una cuenta Gmail y se puede gestionar una ejecución GET y POST. Configuración Accept: type : Basic Auth Haders : Content Type = Aplication/Json Ejemplo: IP/puerto/controlador http://128.1.41.2:8084/api/SoliDineroDet?idempresa=1&idCenResp=72112&idSoliDinero=1286 Los formatos son usados por el servidor para solicitar y responder mensajes, estos formatos son llamados “Media Type Formatters”.	

Una vez ejecutado el comando, podemos acceder a una representación gráfica de nuestra API REST a través de un navegador, Para acceder al contenido de nuestra base de datos basta con especificarlo en la URL.
Ejemplo: texto plano en el lenguaje JSON.


```
[
{
"IdEmp":2,
"IdSerie":1,
"NumSoliCot":2101,
"Nano":2012,
"NumCorrel":15,
"UsuApro":"VICTORCL"
}
]
```


Fuente: Elaboración propia.

2. Desarrollo de interfaz de detalle de la solicitud de dinero.

Tabla 32:
Autenticación de usuario - Tarea N° 2.

Número Tarea: 02	**Numero de Historia:** 3
Nombre de Tarea: Desarrollo de interfaz de detalle de la solicitud de dinero.	
Tipo de tarea: General	**Tiempo estimado:** 32 horas
Fecha inicio: 01/10/18	**Fecha fin:** 04/10/18
Programador responsable: Cárdenas León Víctor	

Descripción:
La presente tarea describe el proceso de desarrollo de interfaz del detalle de la solicitud de dinero, mediante patrón de diseño MVVM(Model-View-ViewModel), se desarrolló SoliDineroDetView.xaml usando View(Vista) es una página Xaml que debe tener su correspondiente clase ViewModel SoliDineroDetViewModel contiene la lógica del negocio, debe estar asociado a un View para notificarle los cambios.
 	
Modelo:
Son las clases que representan el modelo del negocio, tiene como entidad la tabla SoliDineroDet. Sirven para crear un enlace entre la capa de negocios y la de datos. El modelo es el siguiente:
 	
`public class SoliDineroDet`
`{`
`public int IdEmpresa { get; set; }`
 	
`public string IdEmpSofya { get; set; }`	

```
        public string IdCenResp { get; set; }

        public int NumSolicitud { get; set; }

        public string NomBenef { get; set; }

        public string NomRespo { get; set; }

        public DateTime? FecEmiDoc { get; set; }

        public string TipoDoc { get; set; }

        public string SerieDoc { get; set; }

        public string Documento { get; set; }

        public string MonedaDoc { get; set; }

        public double ImporteDoc { get; set; }

        public double ImporteDet { get; set; }

        public double ImporteRet { get; set; }

    }
```

Existe una capa adicional, llamada CAPA DE SERVICIOS, que es una capa adicional de la lógica del negocio, y contiene: servicios de almacenamiento en base de datos local, consumo de Api

Para la comunicación entre nuestra aplicación móvil y la API REST se creó una clase dentro de la carpeta SERVICES llamada **ApiService.cs**, la cual representa a nuestra capa de datos y al punto central por el cual cualquier módulo que quiera leer o escribir a la BD debe pasar. El procedimiento estándar de comunicación consiste en **Serializar** y **Deserializar.**

La acción de leer **GetSoliDineroDet()** realiza una deserialización ya que muestra información, obtenida a través de una acción HTTP **GET** dirigida a nuestra API REST, se encuentra en formato JSON y queremos convertirla a una representación en memoria, más específicamente, a una lista de objetos del tipo **SoliDineroDet** para poder manipularla dentro de nuestro programa. En la acción de escribir hacemos lo inverso: Tenemos un objeto del tipo **SoliDineroDet** el cual es serializado a formato JSON y enviado a través de una acción HTTP **POST** a nuestra API REST.

Fuente: Elaboración propia.

3. Publicación de servicios web.

Tabla 33:
Autenticación de usuario - Tarea N° 5.

Número Tarea: 03	**Numero de Historia:** 3
Nombre de Tarea: Publicación de servicios web	
Tipo de tarea: General	**Tiempo estimado:** 8 horas
Fecha inicio: 27/08/18	**Fecha fin:** 23/08/18
Programador responsable: Cárdenas León Víctor	
Descripción: La presente tarea describe el proceso de publicación de los servicios web desde la aplicación móvil. - Crear una carpeta en una ruta de la maquina donde se alojara los archivos de publicación. - Abrir servidor configurador WebApi, copiar archivos publicados en la ruta C:\inetpub\wwwroot del servidor. - Abrir Internet Information Services IIS. - Configurar un nuevo SiteWeb dentro de Site(Add Site Web), seleccionar un puerto local 8084 para la publicación. - Configurar el nuevo Site con WebApiAdm entrar a Advanced Settings, Application Pool = ASP.NET v4.0. - Ejecutar en el explorador http://128.1.41.2:8084/api/SoliDinero?idempresa=1&idCenResp=72112&idSoliDinero=1286, si es correcto tiene que llamar a la seguridad de los servicios web.	

Fuente: Elaboración propia.

Resultados de la tercera Iteración

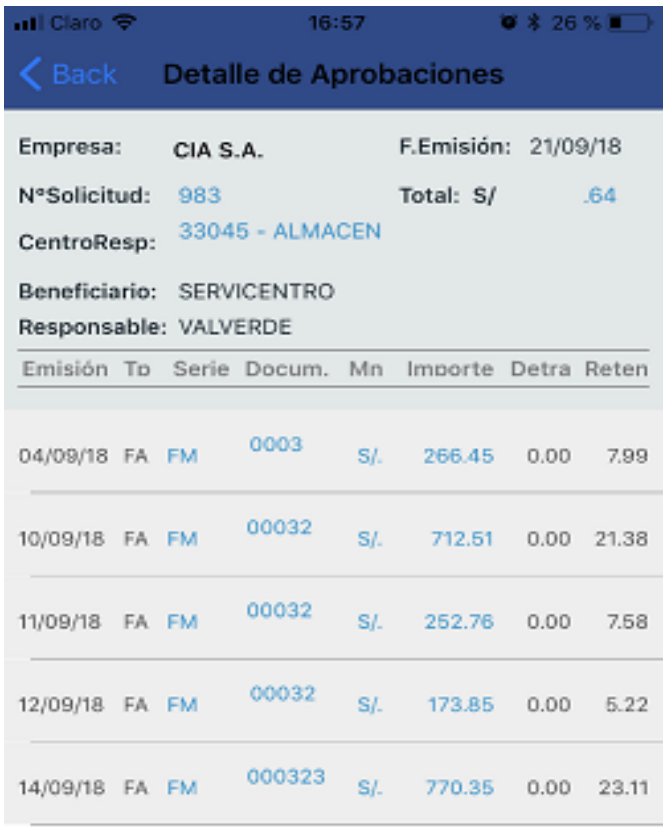

Figura 13. Listado de detalle de solicitud de dinero.
Fuente: Elaboración propia.

Con respecto al cuarto objetivo específico de la investigación se realizó la implementación y publicación de la aplicación móvil.

En esta parte se expone la etapa de implementación, la cual está separada en dos: El trabajo de difusión previo al lanzamiento y la etapa de distribución, donde se comienza a tener contacto con los usuarios. En el primer punto se detalla el trabajo de difusión, constituido por un conjunto de acciones que tienen como objetivo principal dar a conocer el proyecto e invitar a los trabajadores a participar en él. Posteriormente se muestra el periodo de testeo en el cual se muestra el feedback de los testers,

consistente en recomendaciones y reporte de errores. Finalmente se expone el proceso de lanzamiento de la aplicación, distinguiendo las principales dificultades y soluciones del proceso.

Lanzamiento de la Aplicación móvil

Durante el mes de Diciembre se lanzó un correo con un manual de usuario, cuya finalidad es dar a conocer la idea dentro del lugar de trabajo y captar a los trabajadores interesados. Consta de una breve descripción escrita de la idea, la visualización de pantallas con descripción de opciones, también se indica a los interesados de participar de la versión beta de la aplicación. Posterior a su instalación.

Descarga de la Aplicación Móvil para la Distribución

La distribución de la aplicación se realizará a través de las tiendas de aplicaciones. Se determinó que la aplicación estará disponible para las plataformas iOS y Android, por lo cual las descargas se realizarán desde la App Store y Google Play respectivamente.

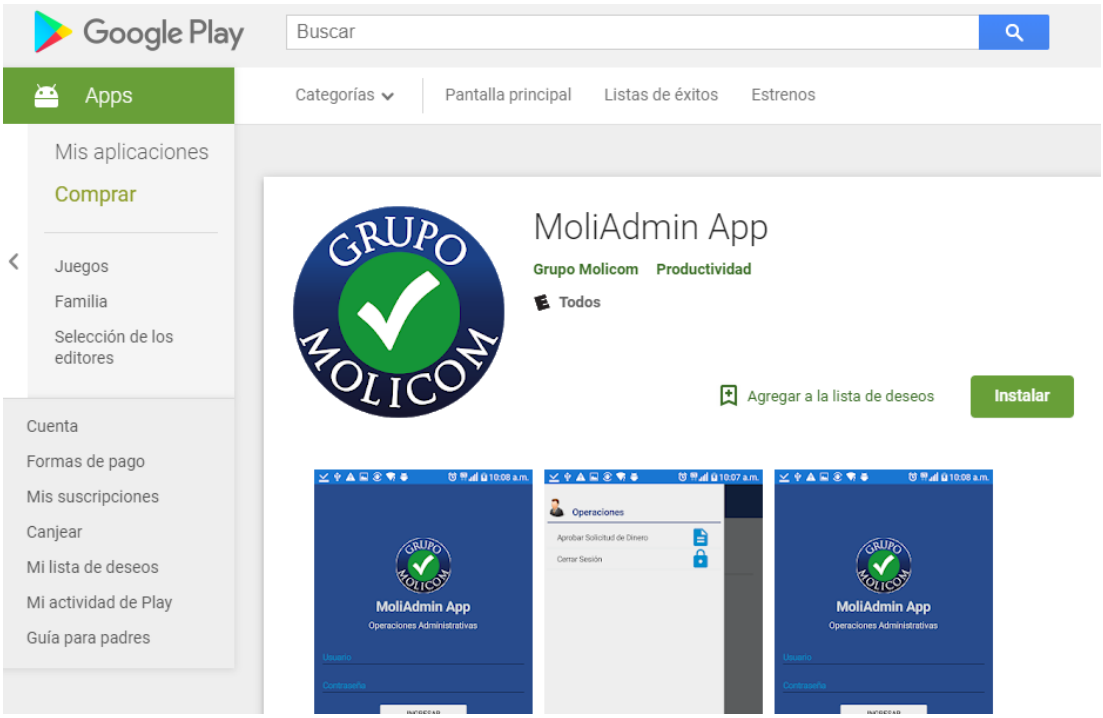

Figura 14. Captura de pantalla de la página en Google Play.
Fuente: Pagina Google Play.

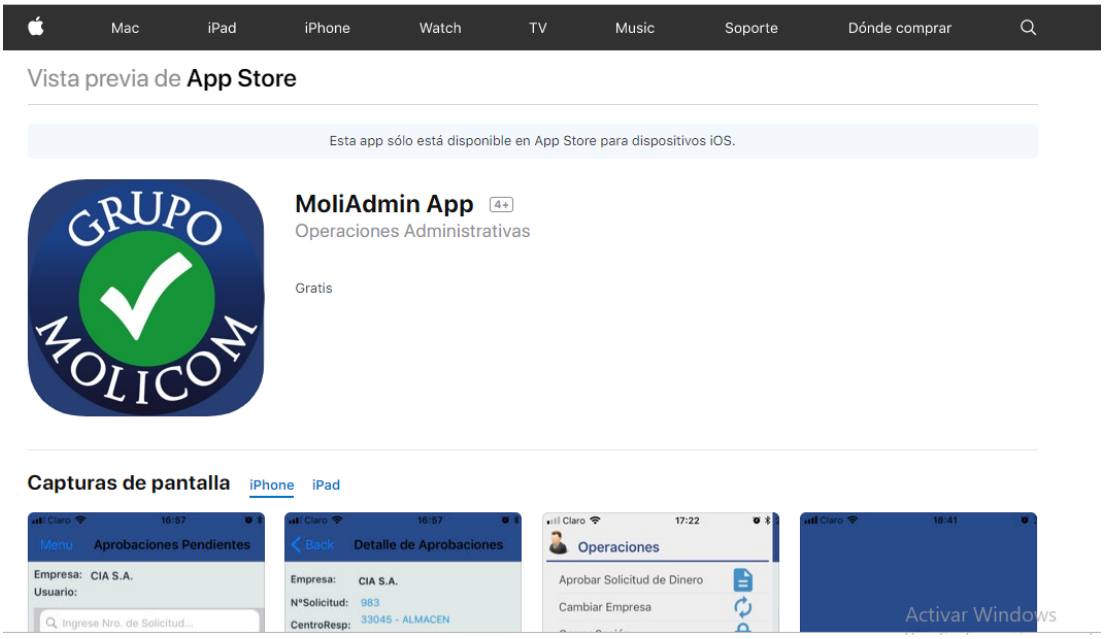

Figura 15. Captura de pantalla de la página en App Store.
Fuente: Pagina App Store.

Implementación de la Aplicación móvil.

La implementación corresponde a la etapa en la cual la aplicación puede ser descargada por usuarios externos al equipo de desarrollo del proyecto. Esta se realiza en dos etapas, Testeo y Distribución. La primera etapa corresponde al periodo en el cual la aplicación es liberada para pocos usuarios con el objetivo de evaluar su funcionamiento y encontrar posibles errores. La etapa de distribución comienza con el lanzamiento de la aplicación en las tiendas de aplicaciones, para la libre descarga de los usuarios.

Google Play dispone de una plataforma web para los desarrolladores de aplicaciones, llamada Developer Console. Esta plataforma busca facilitar la publicación y distribución de las aplicaciones Android. Ofrece a los desarrolladores la posibilidad de ver estadísticas de las descargas de la aplicación, la evaluación y comentarios de los usuarios que la han descargado, gestionar la información de la aplicación que aparece dentro de la tienda y poner a disposición testeo y distribución.

Play Store dispone de una plataforma web App Store Connect para los desarrolladores con la posibilidad de instalar, probar y distribuir las apps para iOS.

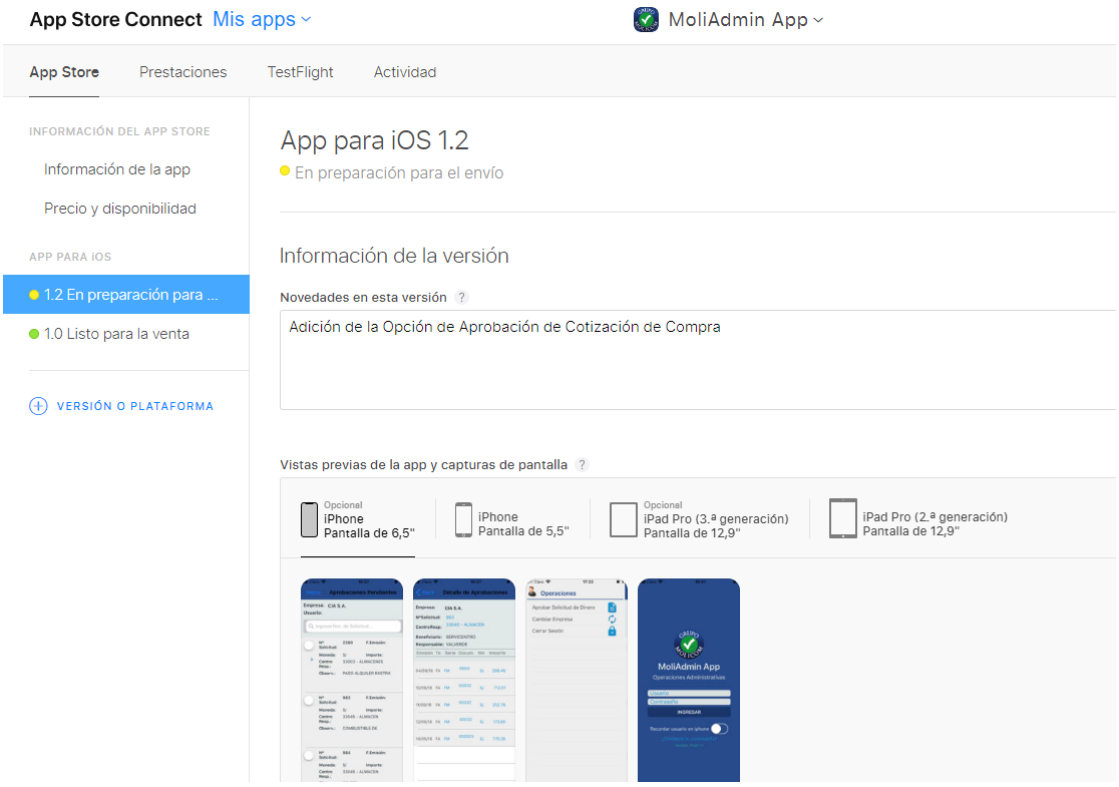

Figura 16. Interfaz de la plataforma App Store Connect.
Fuente: App Store Connect.

Figura 17. Interfaz de la plataforma App Store Connect.
Fuente: App Store Connect.

Testeo de aplicación para dispositivos Android

El testeo de la aplicación se realizó durante un periodo de una semana utilizando la una plataforma beta, que el usuario pudo descargarlo, que entrega la plataforma de Google Play, las pruebas en iPhone se probó en un equipo iPhone donde se realizó pruebas de usuario. Las recomendaciones y errores de funcionamiento que se recibieron son los siguientes:

Recomendaciones:

- Agrandar el botón Aprobar, en iPhone se vio muy pequeño.
- Realizar en la interfaz de login de usuario, opción recordar usuario.

Errores Reportados:

- Leve lentitud en mostrar los datos.
- Se cuelga sin mostrar mensajes de error.

Luego de recibir el feedback de los testers se procedió a reparar los errores de funcionamiento, pues son el principal objetivo de esta instancia. Tras reparar los errores, se evaluó cuál de las recomendaciones ofrecían una mejor relación costo-beneficio y posibilitaban cumplir con los plazos estipulados para el proyecto. Luego se obtuvo por realizar las recomendación ya que no llevo mucho tiempo en realizarlo.

El periodo de Testeo concluyó una vez que no se reportaron errores de funcionamiento, dando paso a la etapa de distribución.

Lanzamiento y distribución de la aplicación para dispositivos Android y iPhone

La distribución corresponde a la etapa en la cual la aplicación está disponible para la libre descarga de los usuarios. El lanzamiento de la aplicación para Android a través de Play Store, se realizó el día 15 de Diciembre, dando inicio a un periodo de aproximadamente de cuatro meses, donde se realizó la medición de datos y recopilación de información desde los usuarios finales, luego en Enero se realizó el lanzamiento a App Store.

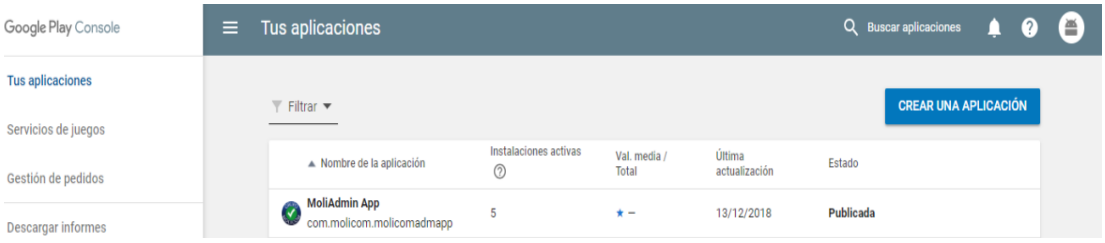

Figura 18. Interfaz de la plataforma App Store Connect.
Fuente: App Store Connect.

Figura 19. Interfaz de la plataforma App Store Connect.
Fuente: App Store Connect.

Los resultados obtenidos y su análisis se presentarán en el siguiente capítulo.

IV. ANÁLISIS Y DISCUSIÓN

Después de obtener resultados a través de las encuestas pre y post realizadas a los trabajadores del Grupo Molicom se logró entender que redujo los tiempos empleados en los registros del proceso en comparación a los realizados mediante un sistema integrado es por ello que llegando a la conclusión se coincide con la Tesis de Emerson Damir Cadena Navarrete (2015) Con la tesis: "DESARROLLO DE UNA APLICACIÓN MULTIPLATAFORMA Y DISTRIBUIDA PARA DISPOSITIVOS MOVILES QUE PERMITA AUTOMATIZAR EL MANEJO DE INFORMACION DEL TRANSPORTE INTERPROVINCIAL DE LA CIUDAD DE QUITO", donde especifica también que por medio de una aplicación móvil donde se hace uso de la tecnología de información brinda un soporte de gran utilidad, como es el caso del Grupo Molicom donde se logró a través de la automatización automatizar procesos, minimizar costos o acceder a las nuevas oportunidades que se presentan al incorporar nuevas tecnologías en la empresa.

Así mismo se pudo determinar el promedio de desempeño de las actividades del proceso de aprobación de solicitud de dinero con un de los trabajadores se encuentran satisfechos ante sus requerimientos por ello que se coincide con la tesis de José Marcos Cueva Rodríguez (2016) Con la tesis: "IMPLEMENTACIÓN DE UNA APLICACIÓN PARA PROCESOS DE CALIFICACIONES DEL COLEGIO DE BACHILLERATO "CIUDAD DE LOYOLA"". Con respecto a los resultados obtenidos por los autores con respecto al desarrollo de la aplicación móvil podemos decir que concordamos con dichos resultados puesto que gracias a la automación a realizar se lograra mejores tiempos de respuesta en las actividades asociadas.

Así mismo un 80% de los trabajadores consideran que realizar el proceso de aprobación de solicitud de dinero mediante sistemas de información es una pérdida de tiempo y sólo un 20% se encuentra satisfechos dado que se requiere de un tiempo adecuado para realizarlo, es por ello que se coincide con la de tesis de José Carbo Vite (2017) con la tesis: "DESARROLLO DE UNA APLICACIÓN MÓVIL CON LA HERRAMIENTA XAMARIN STUDIO PARA EL APOYO Y SOPORTE DE LOS PACIENTES DE ENFERMEDADES RENALES CRÓNICAS DE LA UNIDAD DE HEMODIÁLISIS DIALRÍOS". Nuestra posición con respecto a esta tesis nos dice que también estamos de acuerdo porque también hemos logrado disminuir el tiempo de atención con respecto al sistema de información.

Así mismo se determinó que un 80 % de los trabajadores considera excelente el desarrollo de un aplicativo móvil y solo un 20% está satisfecho con este desarrollo ya que solucionará la problemática que la empresa afronta ya que se tendrá información actualizada de las aprobaciones de solicitud de dinero, con resultados inmediatos ante requerimientos minimizando tiempos. Es por ello que se coincide con la tesis de Gustavo Hernando Pum Tejada (2016) con la tesis, "SISTEMA MOVIL DE DETECCION DE CAIDAS PARA ADULTOS MAYORES USANDO BEACONS" ya que explica que el principal problema que se tiene la empresa es el manejo de respuestas rápidas. El proyecto de investigación es necesario para poder resolver los problemas que tiene en respuestas rápidas apoyándose en la tecnología la empresa empezando con la aprobación de documentos de solicitud de dinero mediante un aplicativo móvil.

Y se coincide también con la tesis Alexander Ocsa, Gustavo Suero, Jose Herrera, Klinge Villalba (2014) con la tesis, “PROPUESTA PARA EL DISEÑO Y DESARROLLO DE APLICACIONES M-LEARNING: CASO, APPS DE HISTORIA DEL PERÚ COMO OBJETOS DE APRENDIZAJE MOVILES”. En este sentido, el objetivo general del presente estudio es proponer una aplicación móvil de visualización y aprobación de documentos de solicitud de dinero para mejorar la gestión de aprobaciones a fin de generar recomendaciones concretas que contribuyan a optimizar la gestión de aprobaciones, tomando en consideración que los tiempos de respuesta, son vitales para la operatividad dentro del grupo Molicom, operaciones que se requieren de pronta atención y tener la información disponible de una manera eficiente y organizada.

V. CONCLUSIONES Y RECOMENDACIONES

➢ Conclusiones

- Con el uso de las encuestas se logró obtener y comprender las necesidades de los interesados de la empresa para poder establecer los requerimientos funcionales y no funcionales a considerar para la aplicación móvil.

- La utilización de la metodología de desarrollo de software (XP) logró reducir el tiempo de desarrollo de la aplicación, ya que mantiene al mínimo la documentación y prioriza la interacción con el usuario.

- Se concluye que se construyó la aplicación móvil usando la herramienta Xamarin.forms y utilizando para la base de datos SQL Server, también se logró la construcción del web service con el patrón de diseño MVC. Además, se logró introducir de manera práctica el patrón de diseño MVVM como se observa en la parte de desarrollo de esta tesis, cumpliendo con el objetivo de aplicar este patrón al desarrollo de una aplicación móvil multiplataforma.

- Según los resultados obtenidos, la implementación de la aplicación móvil redujo el tiempo empleado en el proceso de aprobación de solicitudes de dinero, gracias a que el usuario ahora dispondrá de una aplicación cuando él lo necesite.

- **Recomendaciones**

  - Para la etapa de recolección de información se recomienda anotar todo aquello que se crea es de importancia, por mínimo o insignificante que parezca, siempre hay algo que pueda hacer la diferencia.

  - Es importante que las metodologías y técnicas utilizadas se adapten al tipo de proyecto y sus características para permitir un desarrollo más rápido y de mejor calidad del producto.

  - Existen diferentes herramientas para la implementación de un determinado producto de software, por tanto se recomienda definir claramente en un proyecto que tipo de tecnología utilizar para evitar pérdida de tiempo y recursos.

  - Se recomienda establecer medidas de seguridad que disminuyan la vulnerabilidad de la aplicación contra ataques imprevistos que puedan perjudicar su adecuado desempeño y la integridad de la información que esta procesa. Es por ello que se recomienda tomar en consideración criterios seguridad adicionales.

AGRADECIMIENTOS

Agradezco de manera muy especial por su esfuerzo, dedicación y colaboración, al Ing. Oscar Arquímedes Ascón Valdivia asesor de tesis.

Al Grupo Molicom, por darme la apertura, confianza y la ayuda necesaria para el desarrollo del proyecto.

REFERENCIAS BIBLIOGRÁFICAS

LIBROS

Abrahamsson. (2002). P. et al., "*Agile Software Development Methods: Review and Analysis."* VTT Publications.

K. Beck, Mike Beedle, Alistair Cockburn, Martin Fowler, Jim Highsmith, Robert C.Martin, Ken Schwaber, and Jeff Sutherland. (2001). *"Manifiesto ágil,"*.

K. Beck. (2000). *Extreme programming eXplained*: *embrace change. Reading, MA: Addison-Wesley.*

K. Beck and C. Andres, (2004). *Extreme Programming Explained: Embrace Change*, 2nd Edition, 2nd edition. Boston, MA: Addison-Wesley.

Lakshmiraghavan, B. (2013). Pro ASP.NET Web API Security, 1st ed. New York: Apress.

Olson, S., Hunter, J., Horgen, B., y Kenny, G. (2012). *Professional Cross-Platform Mobile Development in C#,* 1st ed. Indianápolis: John Wiley & Sons.

Olson, S., Hunter, J., Horgen, B., & Goers, K. (2011). *Professional Cross-Platform Mobile Development In C#*. (Wrox, Ed.) (First Edit., p. 388).

Pressman, Roger. (2005). *Ingeniería del Software*. España: McGraw Hill.

P.Abrahamsson, J. Warsta, M. T. Siponen, and J. Ronkainen. (2003). "*New directions on agile methods: a comparative analysis*," in Proceedings of the 25th International Conference on Software Engineering, Washington, DC, USA.

Shackles, G. (2012). *Mobile Development with C#*, 1st ed. California: O'Reilly
Media.

Stallings, W. (2004). *Comunicaciones y Redes de Computadores*, Séptima ed. Madrid: Person.

T. G. Dave West. (2010). *"Agile Development: Mainstream Adoption Has Changed Agility".*

Version One Inc. (2018*). "12th. Anual State of Agile Development Survey."* April.

PÁGINAS WEB

Díaz Concha, (2016). *Xamarin Forms en Acción, Aplicaciones para Acción*. https://docs.google.com/viewerng/viewer?url=http://rclibros.es/wp-content/uploads/2017/05/capitulo_9788494465093.pdf&hl=es

Monocross. (2015, Febrero). *Monocross Model*. http://monocross.net/portfolio/monocross-model

Microsoft. (2017, Diciembre). *Usar el patrón modelo-vista-modelo de vista (MVVM) en Hilo (aplicaciones de la Tienda Windows con C++ y XAML)*. https://msdn.microsoft.com/es-xl/library/windows/apps/jj160324.aspx

Reynolds. (2014, Septiembre). *Implementación de una aplicación para procesos de calificaciones del colegio de bachillerato "Ciudad de Loyola"*. http://repositorio.uide.edu.ec/bitstream/37000/1424/1/T-UIDE-0597.pdf

Xamarin. (2016, Diciembre). *Introducing Xamarin Studio*. https://docs.microsoft.com/es-es/xamarin/xamarin-forms/get-started/index

ANEXOS

Anexo I: Encuesta utilizada para levantamiento de información.

Encuesta – Aplicativo Móvil (Pre Test)

OBJETIVO: Realizar un estudio para determinar la factibilidad del desarrollo de la Aplicación Móvil de Aprobación de Solicitud de Dinero.

Gracias por compartir este momento con nosotros, por favor responda las siguientes preguntas.

---

1. Si deseas realizar una aprobación de Solicitudes de Dinero pendiente por aprobar. ¿Cuánto tiempo utilizas? (Por favor indicar en minutos).
   ……

2. ¿Del 1 al 4 como calificarías el proceso de Aprobación de solicitud de dinero?

   Excelente ( ) Bueno ( ) Regular ( ) Pésimo ( )

3. ¿Del 1 al 3, califique si es fácil o difícil realizar el proceso de aprobación de Solicitud de Dinero?

   Fácil (3) Regular (2) Difícil (1)

4. ¿Usarías una aplicación móvil para Aprobar un documento de Solicitud de Dinero?

   SI ( ) NO ( )

5. ¿Qué tipo de Dispositivo móvil dispone?

   Smartphone ( ) Tablet ( ) iPod ( ) iPad ( )

6. ¿Cuál es el Sistema Operativo de su Dispositivo móvil?

   Android ( ) iOS ( )

Anexo II: Encuesta de satisfacción del trabajador.

Encuesta – Aplicativo Móvil (Post Test)

OBJETIVO: Realizar un estudio para determinar la factibilidad del desarrollo de La Aplicación Móvil de Aprobación de Solicitud de Dinero.

Gracias por compartir este momento con nosotros, por favor responda las siguientes preguntas.

---

1. ¿Cuánto tiempo utilizas para realizar la aprobación de un documento de solicitud de dinero utilizando la aplicación Móvil? (Por favor indicar en minutos.)
   ……

2. ¿Del 1 al 4 cómo calificaría el nuevo proceso para aprobar solicitudes de dinero en la aplicación móvil?

   Excelente (4) Bueno (3) Regular (2) Pésimo (1)

3. ¿Del 1 al 3, califique si es fácil o difícil realizar el proceso de aprobación de un documento de solicitud de dinero?

   Fácil (3) Regular (2) Difícil (1)

Anexo III: Cuadro de observación del proceso de consulta de solicitudes de dinero pendientes de aprobación.

Fecha y Hora	**Nombres y Apellidos de Trabajador**	**Número de veces que ingresa para aprobar solicitudes de dinero en el día**	**Tiempo empleado para consultar las solicitudes de dinero pendientes de aprobación**	**Nivel de complejidad al momento de consultar solitudes de dinero pendientes de aprobación**

Anexo IV: ANALISIS E INTERPRETACION

Después de obtener resultados a través de las encuestas realizadas, antes de la Aplicación Móvil (Pre-Prueba) y otra después de la Aplicación Móvil (Post Prueba) a los trabajadores del Grupo Molicom, dando un total de 35 encuestados, los cuales contestaron una encuesta de manera electrónica, la cual cuenta con preguntas de opción múltiple, estos datos se representan en gráficas de barras con su respectivo análisis donde se interpretan los resultados de cada pregunta realizada a los encuestados. A continuación mostramos los resultados de las encuestas realizadas y las conclusiones que se obtiene a partir de los mismos, tomando como punto de referencia lo desarrollado en capítulos anteriores.

A continuación, se presenta las medidas de los KPIs para la Pre-Prueba y Post-Prueba.

Tabla 34:
Resultados de pre-prueba y post-prueba para los KPI1, KPI2, KPI3, KPI4, KPI5.

	KPI1: Tiempo empleado para aprobar una SD pendiente.		KPI2: Número de veces que el usuario puede consultar una SD al día.		KPI3: Tiempo empleado para consultar una SD.		KPI4: Nivel de usabilidad en el proceso de aprobación de la SD.		KPI5: Nivel de satisfacción del usuario.	
#	Pre-Prueba	Post-Prueba	Pre-Prueba	Post-Prueba	Pre-Prueba	Post-Prueba	Pre-Prueba	Post-Prueba	Pre-Prueba	Post-Prueba
1	14	2	2	4	2	1	Fácil	Fácil	Bueno	Excelente
2	25	5	1	5	4	1	Fácil	Fácil	Excelente	Excelente
3	30	1	1	4	1	1	Regular	Fácil	Regular	Bueno
4	11	1	1	4	2	1	Fácil	Fácil	Bueno	Excelente
5	25	5	2	2	5	2	Regular	Fácil	Bueno	Bueno
6	10	3	1	5	5	1	Fácil	Fácil	Bueno	Bueno
7	18	2	1	4	2	2	Fácil	Fácil	Bueno	Excelente
8	3	1	1	4	2	1	Fácil	Fácil	Bueno	Excelente
9	28	2	2	3	3	2	Regular	Fácil	Regular	Bueno
10	12	5	1	3	5	1	Regular	Fácil	Regular	Bueno
11	27	5	2	4	1	1	Fácil	Fácil	Bueno	Excelente
12	9	2	2	5	1	1	Difícil	Regular	Regular	Regular

13	19	2	2	5	5	2	Fácil	Fácil	Bueno	Excelente
14	30	1	1	4	1	2	Regular	Regular	Regular	Regular
15	10	4	1	1	3	2	Fácil	Fácil	Bueno	Bueno
16	16	4	1	1	3	2	Fácil	Fácil	Bueno	Excelente
17	13	4	2	2	2	1	Fácil	Fácil	Bueno	Excelente
18	20	4	1	4	4	2	Fácil	Fácil	Excelente	Excelente
19	16	3	2	2	5	2	Regular	Regular	Excelente	Bueno
20	18	3	2	5	3	1	Regular	Regular	Bueno	Excelente
21	20	2	2	2	2	1	Fácil	Fácil	Bueno	Excelente
22	7	2	2	3	3	2	Regular	Regular	Bueno	Bueno
23	19	2	2	5	2	1	Fácil	Fácil	Excelente	Bueno
24	11	4	1	3	5	2	Fácil	Fácil	Excelente	Regular
25	24	3	1	5	3	2	Regular	Regular	Regular	Regular
26	8	5	1	5	3	1	Fácil	Fácil	Bueno	Excelente
27	11	4	2	5	5	1	Fácil	Fácil	Excelente	Excelente
28	26	4	2	4	5	1	Fácil	Fácil	Bueno	Excelente
29	21	4	1	1	5	1	Fácil	Fácil	Bueno	Excelente
30	22	4	1	1	5	1	Fácil	Fácil	Bueno	Excelente
31	24	3	1	5	3	2	Regular	Regular	Regular	Regular
32	8	5	1	5	3	1	Fácil	Fácil	Bueno	Excelente
33	11	4	2	5	5	1	Fácil	Fácil	Excelente	Excelente
34	26	4	2	4	5	1	Fácil	Fácil	Bueno	Excelente
35	21	4	1	1	5	1	Fácil	Fácil	Bueno	Excelente

Fuente: Elaboración propia.

El tratamiento estadístico que se inició desde evaluar la confiabilidad de los instrumentos y se proyectó hacia la organización de los datos recolectados para su tratamiento, ha sido realizado tomando como herramienta informática de apoyo el programa Excel.

A continuación, presentamos los cuadros y figuras que explican la percepción de los usuarios respecto a la relación entre las variables de estudio. Acopiada la información relacionada a las variables, se procedió de la manera convencional.

1. Tiempo empleado para aprobar una solicitud de dinero pendiente.

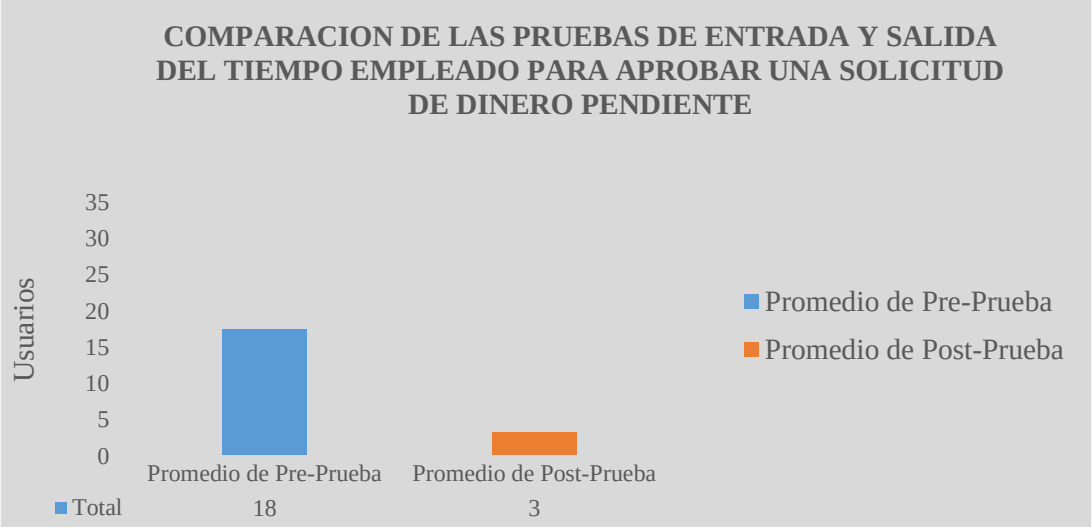

Figura 20. Resumen de grafico KPI1.
Fuente: Elaboración propia.

Interpretación:

El grafico nos muestra la opinión de los trabajadores en este caso el usuario, se obtuvo como media los minutos empleados para aprobar una solicitud de dinero pendiente en el Grupo Molicom, en la Preprueba muestra el valor de 18 minutos, mientras en la Postprueba el valor fue de 3 minutos; esto indica una gran diferencia antes y después de la implementación de la aplicación móvil, por lo tanto, la comparación de medias se considera adecuada, ya que los datos no son mayores y menores con respecto a la media, es decir son muy compactos.

2. Número de veces que el usuario puede consultar una solicitud de dinero al día.

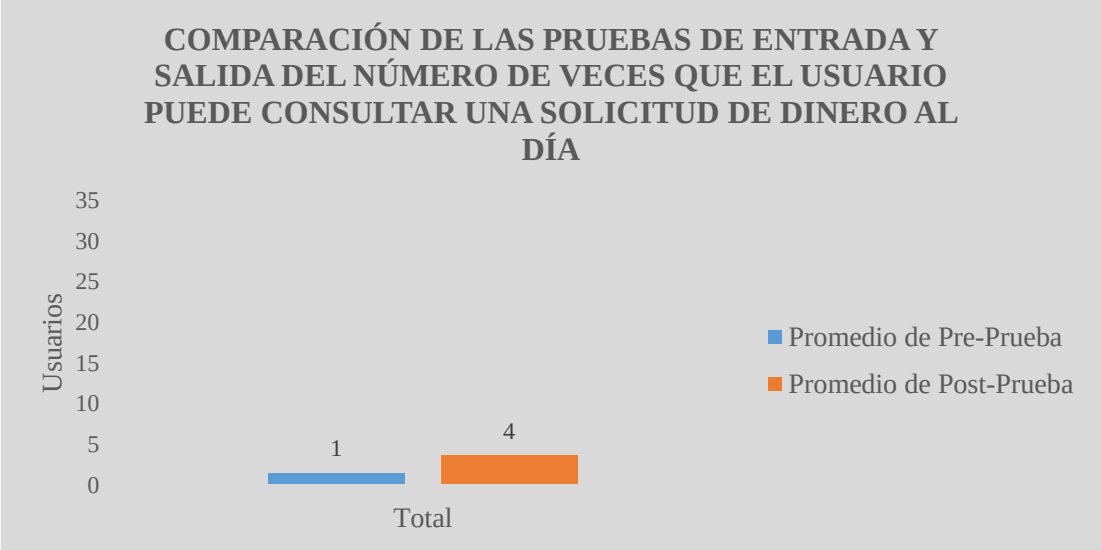

Figura 21. Resumen de grafico KPI2.
Fuente: Elaboración propia.

Interpretación:

Se obtuvo como media el número de veces que el usuario puede consultar solicitudes de dinero en la aplicación móvil como pendientes de aprobación, en la Preprueba muestra el valor de 1 consulta, mientras en la Postprueba el valor fue de 4 consultas; esto indica una gran diferencia antes y después de la implementación de la aplicación móvil.

Por lo tanto la comparación de medias no se considera adecuada, ya que los datos son mayores y menores con respecto a la media, es decir son muy dispersos.

3. Tiempo empleado para consultar una solicitud de dinero.

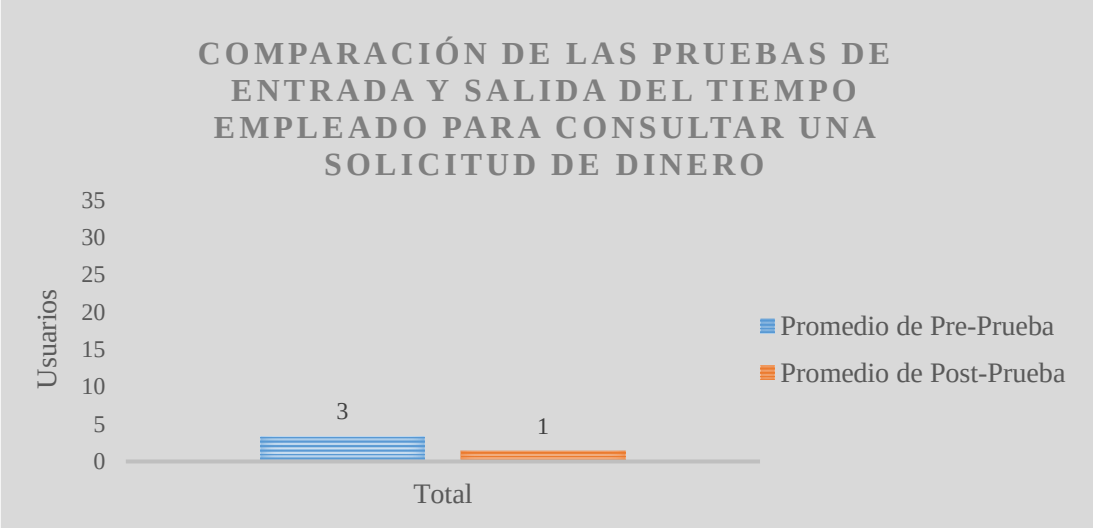

Figura 22. Resumen de grafico KPI3.
Fuente: Elaboración propia.

Interpretación:

Se obtuvo como media los minutos empleados para consultar el horario de salida del próximo tren de una estación, en la Preprueba muestra el valor de 3 minutos, mientras en la Postprueba el valor fue de 1 minuto; esto indica una gran diferencia antes y después de la implementación de la aplicación móvil; asimismo, los valores mínimos de los minutos, 3 minutos antes y 1 minuto después.

Se demuestra variabilidad con respecto a los datos si difiere en gran medida, por lo tanto, la comparación de medias no se considera adecuada, ya que los datos son mayores y menores con respecto a la media, es decir son muy dispersos.

4. Nivel de usabilidad en el proceso de aprobación de la solicitud de dinero.

PRE-PRUEBA	
Estado	Frecuencia
Fácil	24
Regular	10
Difícil	1
Total	**35**

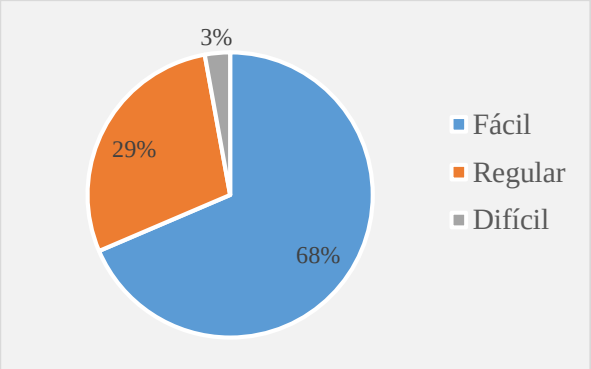

Figura 23. Pre - Prueba KPI4.
Fuente: Elaboración propia.

Interpretación de Pre-Prueba KPI4:

- Sólo el 3% de las veces de Nivel de usabilidad del proceso fueron catalogadas como difíciles.

- El 29% de las veces de Nivel de usabilidad del proceso fueron catalogadas como Regulares.

- Se determina que el 68% de las veces de Nivel de usabilidad del proceso fueron catalogadas como Fáciles.

POST-PRUEBA	
Estado	Frecuencia
Fácil	28
Regular	7
Difícil	0
Total	**35**

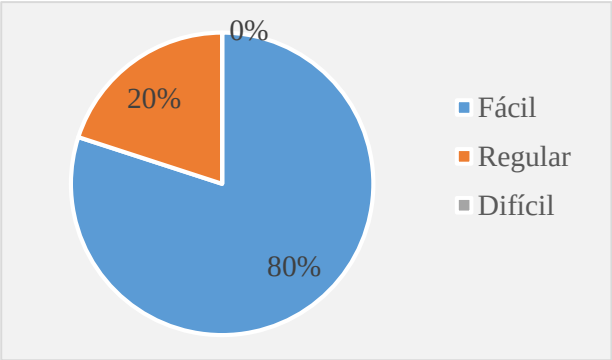

Figura 24. Pre - Prueba KPI4.
Fuente: Elaboración propia.

Interpretación de Post-Prueba KPI4:

- Sólo el 0% de las veces de Nivel de usabilidad del proceso fueron catalogadas como Difíciles.
- El 20% de las veces de Nivel de usabilidad del proceso fueron catalogadas como Regulares.
- Se determina que el 80% de las veces de Nivel de usabilidad del proceso fueron catalogadas como Fáciles.

5. Nivel de satisfacción del usuario.

PRE-PRUEBA	
Estado	Frecuencia
Bueno	21
Excelente	7
Regular	7
Pésimo	0
Total	**35**

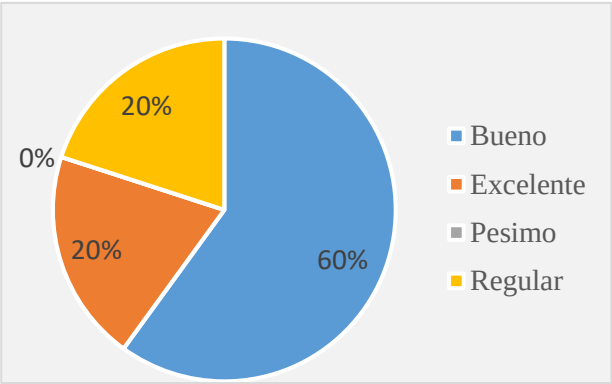

Figura 25. Pre - Prueba KPI5.
Fuente: Elaboración propia.

Interpretación de Pre-Prueba KPI5:

- El 0% de las veces de Nivel de Satisfacción del usuario fueron catalogadas como Pésimas.

- El 20% de las veces de Nivel de Satisfacción del usuario fueron catalogadas como Regulares.

- El 60% de las veces de Nivel de Satisfacción del usuario fueron catalogadas como Buenas.

- Se determina que el 20% de las veces de Nivel de Satisfacción del usuario fueron catalogadas como Excelentes.

POST-PRUEBA	
Estado	**Usuario**
Bueno	9
Excelente	21
Regular	5
Pésimo	0
Total	**35**

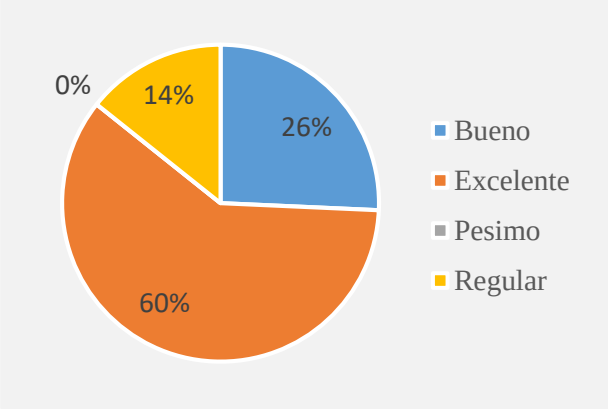

Figura 26. Post - Prueba KPI5.
Fuente: Elaboración propia.

Interpretación de Post-Prueba KPI5:

- El 0% de las veces de Nivel de Satisfacción del usuario fueron catalogadas como Pésimas.

- El 14% de las veces de Nivel de Satisfacción del usuario fueron catalogadas como Regulares.

- El 26% de las veces de Nivel de Satisfacción del usuario fueron catalogadas como Buenas.

- Se determina que el 60% de las veces de Nivel de Satisfacción del usuario fueron catalogadas como Excelentes.