UNIVERSIDAD PUBLICA DE EL ALTO

CARRERA DE INGENIERÍA DE SISTEMAS

PROYECTO DE GRADO

DESARROLLO DE UNA BILLETERA MOVIL
PARA REALIZAR TRANSACCIONES FINANCIERAS
CASO: BANCO DE CRÉDITO DE BOLIVIA “BCP”

Para Optar al Título de Licenciatura en Ingeniería de Sistemas

Mención: MENCIÓN INFORMÁTICA Y COMUNICACIONES

Postulante: Univ. Peter Ciro Alanoca Aruquipa
Tutor Metodológico: Ing. Marisol Arguedas Balladares
Tutor Revisor: Ing. Freddy Alanoca Coarite
Tutor Especialista: Ing. Reynaldo Javier Zeballos Daza

EL ALTO – BOLIVIA

2020

Dedicatoria

Dedico el presente proyecto:

A mis padres.

Con mucho cariño y amor a mis padres: Cirilo y Secundina quienes me brindaron su amor, apoyo incondicional, cariño, disciplina, lágrimas, alegrías, principios y valores.

Me ayudaron en toda mi vida y en los momentos más difíciles, me dieron todo y lo que soy como persona será gracias a ellos.

A mi familia.

A mi compañera Judith y mi hija Celeste… gracias por el sacrificio, tolerancia y esfuerzo pude alcanzar y mi Sueño.

A mi hermana.

Doy gracias a mi querida hermana: Sofia, gracias a ella por los momentos y la vida que compartimos, fueron mi motivación para lograr esta meta.

Agradecimientos

A Dios, por haberme permitido llegar a este momento tan especial, por haberme dado salud, fortaleza y sabiduría para continuar y lograr mis objetivos.

Gracias a la Ing. Marisol Arguedas Balladares. Tutor metodológico, por sus palabras de aliento y guía profesional para la culminación y defensa de este proyecto.

A el Ing. Freddy Alanoca Coarite. Tutor Revisor, por brindarme su apoyo incondicional, tiempo, dedicación y por su comprensión, me permitió culminar del presente trabajo.

A el Ing. Reynaldo Javier Zeballos Daza. Tutor especialista por el voto de confianza y la orientación específica en la realización del proyecto.

Al Sr. Director Ing. David Carlos Mamani Quipe. por el apoyo al frente de la Carrera de Ingeniería de Sistemas y al personal encargado de la administración. Y fundamentalmente a la Universidad Pública de El Alto, mi casa superior de estudios donde adquirí los conocimientos y formación que me impulsan a ser mejor día a día.

Por último, agradecer al Lic. Luis Loza Pari Gerente del área de Negocio del Banco de Crédito. Quien me abrió la puerta para poder demostrar mi capacidad profesional.

A todos ellos… Muchas Gracias.

Resumen

El presente proyecto de grado pretende sitematizar los procesos que la empresa realiza como las transferencias y los pagos de servicios de manera inmediata, ya que estos procesos son de vital importancia para la empresa y es necesario que se tenga que sistematizar las transacciones financieras para usuarios que no tengan una cuenta bancaria. El Proyecto de Grado titulado DESARROLLO DE UNA BILLETERA MOVIL PARA REALIZAR TRANSACCIONES FINANCIERAS CASO: BANCO DE CRÉDITO DE BOLIVIA “BCP” ha sido desarrollado en las oficinas del Banco de Crédito con el objetivo de sistematizar las transacciones financieras presenciales.

Para el desarrollo del proyecto se utilizó la metodología Scrum, que propone un modelo de proceso incremental, basado en iteraciones y revisiones continuas.

Para la conclusión del desarrollo de la aplicación Android se utilizó como herramienta primordial el lenguaje de programación Java, con el gestor de base de datos SQL SERVER y con la ayuda principal del servidor para la función correcta de la aplicación.

INDICE GENERAL

CAPITULO I - MARCO PRELIMINAR 1

1.1 Introducción 2

1.2 Antecedentes 3

1.2.1 Antecedentes institucionales 3

1.2.2 Antecedentes académicos 7

1.2.3 Antecedentes internacionales 8

1.2.4 Antecedentes nacionales 9

1.3 Planteamiento del problema 10

1.3.1 Problema general 10

1.3.2 Problemas específicos 10

1.4 Objetivos 11

1.4.1 Objetivo general 11

1.4.2 Objetivos específicos 11

1.5 Justificación 12

1.5.1 Justificación técnica 12

1.5.2 Justificación económica 12

1.5.3 Justificación social 12

1.6 Metodología 13

1.7 Herramientas 16

1.8 Límites y alcances 19

1.8.1 Límites 19

1.8.2 Alcances 19

1.9 Aportes 20

CAPITULO II - MARCO TEORICO 21

2.1 Introducción 22

2.2 Antecedentes de la institución 22

2.2.1 Misión 24

2.2.2 Visión 24

2.2.3 Estructura 24

2.2.4 Organigrama 25

2.2.5 Conformación 26

2.2.6 Modelamiento del sistema actual 28
2.3 Sistema 32
2.4 Billetera Móvil 32
2.5 Nociones básicas sobre aplicaciones móviles 33
2.6 Versiones disponibles de Android 33
2.6.1 Cupcake: Android Versión 1.5 34
2.6.2 Donut: Android Versión 1.6 34
2.6.3 Eclair: Android Versión 2.0/2.1 34
2.6.4 Froyo: Android Versión 2.2 34
2.6.5 Ginger Bread: Android Versión 2.3 34
2.6.6 Honey Comb: Android Versión 3.0/3.4 34
2.6.7 Ice Cream Sandwich: Android Version 4.0 35
2.6.8 Jelly Bean Android Versión 4.1 35
2.6.9 KitKat Android Versión 4.4 35
2.6.10 Lollipop Android Versión 5.0 35
2.6.11 Mashmallow Android Versión 6.0 36
2.6.12 Nougat Android Versión 7.0/7.1 36
2.6.13 Oreo Android Versión 8.0/8.1 36
2.6.14 Pie Android Versión 9 37
2.6.15 Pie Android Versión 10 37
2.6.16 Pie Android Versión 11 38
2.7 Web Services 38
2.7.1 Características de los Servicios Web 39
2.7.2 Tipos de servicios Web 40
2.7.3 Arquitectura de los Servicios Web 42
2.8 Intranet 43
2.8.1 Características de una Intranet 43
2.8.2 Ventajas y Desventajas de una Intranet 43
2.9 Metodología Ágil 44
2.10 Metodología Scrum 46
2.10.1 Ventajas de Scrum 47
2.10.2 Desventajas de Scrum 47
2.10.3 Diferencias del desarrollo tradicional vs Scrum 47

2.10.4 Objetivos y preferencias de la gestión ágil 48
2.10.5 Visión general del proceso 51
2.10.6 Modelo aplicativo 53
2.10.7 FASE N° 1: Definir backlog del producto 55
2.10.8 FASE N° 2: Planificación del backlog 58
2.10.9 FASE N° 3: Scrum diario 58
2.10.10 FASE N° 4: Revisión del Sprint 60
2.10.11 FASE N° 5: Retrospectiva del Sprint 62
2.11 Arquitectura de software 65
2.12 Arquitectura Model View View Model 66
2.13 Patrón modelo vista controlador 68
2.13.1 Descripción del patrón 68
2.14 Herramientas de desarrollo 70
2.15 Calidad de software – ISO 9125 76
2.15.1 Características 76
2.15.2 Análisis de costos COCOMO II 77
2.16 Pruebas de Caja Blanca 83
2.16.1 Prueba del Camino Básico 83
2.16.2 Representación de un grafo de flujo 85
2.16.3 Calcular la complejidad ciclomática 85
2.16.4 Determinar el conjunto de caminos básicos independientes 85
2.16.5 Derivar los casos de prueba 86
CAPITULO III - MARCO APLICATIVO 87
3.1 Introducción 88
3.2 FASE 1: Definir backlog del producto 88
3.2.1 Historias de usuario 88
3.2.2 Historias identificadas 91
3.3 FASE 2: Planificación del Sprint 97
3.3.1 Primera reunión de planificación de Sprint (SPRINT 1) 99
3.3.2 Segunda reunión de planificación de Sprint (SPRINT 2) 102
3.3.3 Tercera reunión de planificación de Sprint (SPRINT 3) 104
3.3.4 Cuarta reunión de planificación de Sprint (SPRINT 4) 105
3.3.5 Quinta reunión de planificación de Sprint (SPRINT 5) 106

3.4 FASE 3: Scrum diario 107
3.4.1 Comunicación de Sprint Backlogs 107
3.4.2 Trabajando con el cuadro Burndown 109
3.5 FASE 4: Revisión del Sprint 113
3.5.2 Sprint 1 114
3.5.3 Sprint 2 118
3.5.4 Sprint 3 120
3.5.5 Sprint 4 122
3.5.6 Sprint 5 125
3.6 FASE 5: Retrospectiva del Sprint 128
3.7 Pruebas de Software 131
3.7.1 Pruebas de Caja Blanca 131
CAPITULO IV - CALIDAD Y SEGURIDAD 135
4.1 Calidad 136
4.1.1 Usabilidad 136
4.1.2 Mantenibilidad 137
4.1.3 Funcionalidad 138
4.1.4 Confiabilidad 142
4.1.5 Portabilidad 143
4.1.6 Resultados 144
4.2 Seguridad 145
4.2.1 Seguridad de la capa de transporte 145
4.2.2 SSL PINNING 146
4.2.3 Ofuscación del código 147
4.2.4 Seguridad de Base de Datos 147
CAPITULO V – ANALISIS DE COSTOS 149
5.1 Cálculo de costos 150
5.2 Cálculo del Esfuerzo de Desarrollo 152
5.3 Cálculo del Tiempo de Desarrollo 152
5.4 Cálculo de Productividad 152
5.5 Cálculo de Personal Promedio 153
5.6 Costo de desarrollo 153
CAPITULO VI - CONCLUSIONES Y RECOMENDACIONES 154

6.1 Conclusiones 155
6.2 Recomendaciones 155
BIBLIOGRAFIA 157
REFERENCIAS ELECTRÓNICAS 159
ANEXOS 162

INDICE DE FIGURAS

CAPITULO II

Figura Nro. 2. 1: Organigrama Institucional 25

Figura Nro. 2. 2: Diagrama de flujo del proceso de apertura de cuenta 28

Figura Nro. 2. 3: Diagrama de flujo del proceso de transferencias entre cuentas 29

Figura Nro. 2. 4: Diagrama de flujo del proceso de proceso de depósito a cuenta 30

Figura Nro. 2. 5: Diagrama de flujo del proceso de pago de servicios básicos.....31

Figura Nro. 2. 6: Desarrollo tradicional vs. Desarrollo ágil 48

Figura Nro. 2. 7: Visión general del proceso 51

Figura Nro. 2. 8: Modelo aplicativo 54

Figura Nro. 2. 9: Ejemplo de Product Backlog o Pila de Producto 57

Figura Nro. 2. 10: Ejemplo de pizarra de una reunión de retrospectiva Scrum 64

Figura Nro. 2. 11: La arquitectura MVVM 66

Figura Nro. 2. 12: Patrón modelo vista controlador 68

CAPITULO III

Figura Nro. 3. 1: Sprint backlog para el proyecto de Billetera Móvil 108

Figura Nro. 3. 2: Diagrama burndown para el Sprint backlog 1 109

Figura Nro. 3. 3: Diagrama burndown para el Sprint backlog 2 110

Figura Nro. 3. 4: Diagrama burndown para el Sprint backlog 3 111

Figura Nro. 3. 5: Diagrama burndown para el Sprint backlog 4 112

Figura Nro. 3. 6: Diagrama burndown para el Sprint backlog 5 113

Figura Nro. 3. 7: Diagrama de base de datos nivel transaccional 114

Figura Nro. 3. 8: Diagrama de base de datos nivel transaccional e historial 115

Figura Nro. 3. 9: Diagrama de base de datos nivel cliente 116

Figura Nro. 3. 10: Diagrama de base de datos nivel transferencias y pago de servicios.....116

Figura Nro. 3. 11: Captura de endpoints del inicio de sesión en Visual Studio 117

Figura Nro. 3. 12: Captura de pantalla del inicio de sesión en el emulador de Android Studio 117

Figura Nro. 3. 13: Captura de pantalla del registro en el emulador de Android Studio.....118

Figura Nro. 3. 14: Captura de endpoints de recargas y transferencias en Visual Studio 119

Figura Nro. 3. 15: Captura de pantalla de transferencia en el emulador de Android Studio 119

Figura Nro. 3. 16: Captura de pantalla de recargas en el emulador de Android Studio.....120

Figura Nro. 3. 17: Captura de endpoints de transferencias a cuentas BCP en Visual Studio 121

Figura Nro. 3. 18: Captura de pantalla de transferencias a cuentas BCP en el emulador de Android Studio.....122

Figura Nro. 3. 19: Captura de pantalla de transferencias interbancarias en el emulador de Android Studio 122

Figura Nro. 3. 20: Captura de endpoints de movimientos y pago de servicios en Visual Studio 123

Figura Nro. 3. 21: Captura de pantalla de movimientos en el emulador de Android Studio 123

Figura Nro. 3. 22: Captura de pantalla de pagos de servicios en el emulador de Android Studio...124

Figura Nro. 3. 23: Captura de endpoints de depósito y cambio de pin en Visual Studio 125

Figura Nro. 3. 24: Captura de pantalla de cambio de pin en el emulador de Android Studio.....126

Figura Nro. 3. 25: Captura de pantalla de reseteo de pin en el emulador de Android Studio.....126

Figura Nro. 3. 26: Captura de pantalla de depósito en el emulador de Android Studio.....127

Figura Nro. 3. 27: Captura de pantalla del código fuente de inicio de sesión en Android Studio ...132

Figura Nro. 3. 28: Grafo de pruebas de caja blanca inicio de sesión 133

INDICE DE TABLAS

CAPITULO I

Tabla Nro. 1. 1: Tabla comparativa de lenguajes de programación 16

Tabla Nro. 1. 2: Tabla comparativa de gestores de base de datos 17

Tabla Nro. 1. 3: Tabla comparativa de lenguajes de servidores web 18

CAPITULO II

Tabla Nro. 2. 1: Diferencias entre las metodologías 46

Tabla Nro. 2. 2: Herramientas de desarrollo lenguaje de programación C# 70

Tabla Nro. 2. 3: Herramientas de desarrollo lenguaje de programación Java 71

Tabla Nro. 2. 4: Herramientas de desarrollo IDE Android Studio 73

Tabla Nro. 2. 5: Herramientas de desarrollo IDE Visual Studio 74

Tabla Nro. 2. 6: Herramientas de desarrollo gestor de base de datos Microsoft SQL Server 75

Tabla Nro. 2. 7: Estimación de esfuerzo.....79

Tabla Nro. 2. 8: Estimación de costos 82

CAPITULO III

Tabla Nro. 3. 1: Historia de usuario afiliación de una cuenta Billetera.....91

Tabla Nro. 3. 2: Historia de usuario inicio de sesión 91

Tabla Nro. 3. 3: Historia de usuario opción de inicio o bienvenida 92

Tabla Nro. 3. 4: Historia de usuario recargas de crédito de telefónicas 92

Tabla Nro. 3. 5: Historia de usuario transferencias entre cuentas billetera 93

Tabla Nro. 3. 6: Historia de usuario transferencias a cuentas BCP.....93

Tabla Nro. 3. 7: Historia de usuario transferencias a cuentas de otros bancos.....94

Tabla Nro. 3. 8: Historia de usuario estados de transferencias y movimientos.....94

Tabla Nro. 3. 9: Historia de usuario estados de pago de servicios 95

Tabla Nro. 3. 10: Historia de usuario retiro en cajeros automáticos y compra en comercios 95

Tabla Nro. 3. 11: Historia de usuario cambio y reseteo de PIN 96

Tabla Nro. 3. 12: Historia de usuario depósitos en cajeros automáticos para no clientes 96
Tabla Nro. 3. 13: Team Scrum 97
Tabla Nro. 3. 14: Product Backlog 98
Tabla Nro. 3. 15: Sprint Planning días y horas disponibles 100
Tabla Nro. 3. 16: Sprints backlog definidos en la primera reunión de planificación 102
Tabla Nro. 3. 17: Estimación de tiempo para el Sprint 1 102
Tabla Nro. 3. 18: Estimación de tiempo para el Sprint 2 103
Tabla Nro. 3. 19: Estimación de tiempo para el Sprint 3 105
Tabla Nro. 3. 20: Estimación de tiempo para el Sprint 4 106
Tabla Nro. 3. 21: Estimación de tiempo para el Sprint 5 107
Tabla Nro. 3. 22: Estado final de las tareas del Product Backlog del proyecto 129
Tabla Nro. 3. 23: Cierre del proyecto con Scrum 130
Tabla Nro. 3. 24: Casos de prueba para pruebas de caja blanca 134

CAPITULO IV

Tabla Nro. 4. 1: Encuesta de Usabilidad del Sistema 137
Tabla Nro. 4. 2: Entradas para el cálculo de funcionalidad según punto de fusión 139
Tabla Nro. 4. 3: Cuenta Total con Factor de Ponderación Medio 140
Tabla Nro. 4. 4: Ajuste del factor de complejidad 141
Tabla Nro. 4. 5: Resultado de Evaluación de Calidad 144

CAPITULO V

Tabla Nro. 5. 1: Tabla de conversión factor LDC/PF 150
Tabla Nro. 5. 2: Modelo COCOMO 151
Tabla Nro. 5. 3: Ajuste del factor de complejidad 152

CAPITULO I

MARCO PRELIMINAR

1.1 Introducción

En la era de la información, la banca electrónica global está aumentando su importancia rápidamente e influyendo en la manera como los bienes y servicios son comercializados a través de las fronteras. Y es que la aparición de Internet hace posible la transformación de la banca hacia la banca electrónica, haciendo que el sistema de comercio mundial este comenzando una etapa en la cual los bienes y servicios son comercializados utilizando cada vez más medios de entrega electrónicos, reduciendo costos y mejorando la productividad, al igual que las opciones del consumidor.

A pesar de la tendencia a nivel mundial, la banca electrónica en Bolivia sigue rezagado. Las razones pueden ser diversas desde la falta de un adecuado sistema de pago, la desconfianza que genera su uso por la informalidad, la baja penetración del internet en el país y la escasez de una normativa específica, como las principales limitaciones que obstruyen su desarrollo. A pesar de esto, con la globalización, la banca electrónica empieza a popularizarse de manera informal en el país.

La metodología utilizada para llevar a cabo este proyecto Scrum, debido a las condiciones y requerimientos planteados por los solicitantes y que dadas las condiciones, tiempo y recursos se decidió el uso de este método de desarrollo ágil.

La calidad fue avaluada basada en la ISO 9126, que es un estándar internacional para la evaluación de calidad de software.

Como herramientas de desarrollo de la aplicación Android se utilizó como herramienta primordial el lenguaje de programación Java, con el gestor de base de datos SQL SERVER y con la ayuda principal del servidor para la función correcta de la aplicación.

1.2 Antecedentes

1.2.1 Antecedentes institucionales

El Banco de Crédito BCP opera en el Perú desde 1889 como Banco Italiano y a partir de 1941 como Banco de Crédito del Perú. El BCP es una institución sólida cuya vocación y principios la han hecho líder indiscutible del mercado peruano a lo largo de sus más de 115 años de impecable trayectoria.

Desde 1995 forma parte del grupo Credicorp, uno de los conglomerados financieros más importantes de Latinoamérica, que combina los negocios bancarios del BCP y del Atlantic Security Bank con los negocios de seguros de Pacífico Peruano Suiza, la empresa más grande del Perú en su rubro. Las acciones de Credicorp están inscritas en la Bolsa de Valores de Nueva York, lo que le permite acceder con facilidad a los mercados de capitales internacionales.

El Banco de Crédito de Bolivia S.A. es una empresa 100% subsidiaria del Banco de Crédito BCP. Iniciamos nuestras operaciones en el mercado boliviano en 1994 después de que adquirimos el Banco Popular, institución que se ubicaba en el puesto número trece del sistema financiero nacional.

Esta primera apuesta por Bolivia fue muy exitosa, lo que nos alentó a seguir invirtiendo en el país, adquiriendo en 1998 el Banco de La Paz y un año más tarde la cartera del Banco Boliviano Americano. Con estas compras, y en base al empuje de nuestro excelente equipo de profesionales, fuimos ganando la preferencia y confianza de nuestros clientes hasta ubicarnos entre los principales bancos del país.

En el año 2000, comenzaron un proceso de renovación tecnológica sin precedentes con el fin de sentar las bases necesarias para otorgar a sus clientes el mejor servicio de la banca boliviana. Trajeron la más moderna tecnología bancaria disponible, interconectamos nuestros sistemas informáticos con los del Banco de Crédito BCP en Perú e introdujimos el Servimatic, su innovador sistema de filas virtuales, que revolucionó el concepto de atención al cliente no sólo en la banca sino también en otras industrias del país.

Desarrollaron modernos canales electrónicos como la Banca por Teléfono y la Banca por Internet para llevar al Banco a cualquier lugar del mundo donde nuestros clientes están.

Asimismo, invirtieron más de dos millones de dólares para ampliar su red de canales de atención al público, consolidándola como la más grande del país con 45 oficinas y 125 cajeros automáticos a nivel nacional.

En la actualidad, sus esfuerzos han hecho del Banco de Crédito de Bolivia S.A. una institución moderna, innovadora y con excelente calidad de atención. Tienen la cartera más sana y con una de las mayores coberturas del sistema, lo que aunado al sólido respaldo internacional del BCP, les pone en las mejores condiciones para seguir avanzando hacia el liderazgo indiscutible del sistema financiero nacional.

La apuesta del Banco de Crédito por Bolivia es definitiva y por ello han renovado nuestra imagen institucional, haciéndola más moderna y cercana. Su nuevo logotipo es el símbolo de que ponemos nuestro sólido respaldo internacional al servicio del país, con el compromiso de otorgar a sus clientes una experiencia bancaria simple y eficiente.

De ahora en adelante, trabajaran con mayor dedicación, accesibilidad y flexibilidad. Pondrán a disposición del mercado nuevos productos y servicios. Seguirán invirtiendo, seguirán

modernizándonos, seguirán innovando y contribuyendo al desarrollo del país y de sus clientes. Todo esto, con el propósito de seguir siendo el mejor Banco de Bolivia.

Misión

Ofrecer soluciones financieras a personas naturales y jurídicas en Bolivia con la mejor tecnología, calidad y servicio al cliente construyendo relaciones de largo plazo.

Visión

Ser la Primera opción para el Cliente del Sistema Financiero Boliviano.

Principios

- **Satisfacción del cliente**

  Ofrecer a sus clientes una experiencia de servicio positiva a través de nuestros productos, servicios, procesos y atención.

- **Pasión por las metas**

  Trabajar con compromiso y dedicación para exceder nuestras metas y resultados y lograr el desarrollo profesional en el BCP.

- **Eficiencia**

  Cuidar los recursos del BCP como si fueran los propios.

- **Gestión al riesgo**

  Asumir el riesgo como elemento fundamental en nuestro negocio y tomar la responsabilidad de conocerlo, dimensionarlo y gestionarlo.

- **Transparencia**

  Actuar de manera abierta, honesta y correcta con tus compañeros y clientes y brindarles información confiable para establecer con ellos relaciones duraderas.

- **Disposición al cambio**

  Tener una actitud positiva para promover y adoptar los cambios y mejores prácticas.

- **Disciplina**

  Ser ordenado y estructurado para aplicar consistentemente los procesos y modelos de trabajo establecidos.

Valores

- Honestidad
- Compromiso
- Respeto
- Transparencia
- Optimismo
- Actitud Positiva

1.2.2 Antecedentes académicos

Con relación a proyectos similares, se han revisado los siguientes trabajos:

“PAGO ELECTRONICO A TRAVES DE TELEFONOS MOVILES”

El objetivo de este proyecto es ofrecer un mecanismo de pago electrónico a las personas desde cualquier lugar geográfico con cobertura celular, está dividido en cinco capítulos, en el primer capítulo habla de los de los antecedentes del negocio, posterior mete hablamos sobre la situación actual del negocio la forma de procesar las transacciones, proseguimos con la posible tecnología a utilizar para resolver nuestro problema planteado. (ESCUELA SUPERIOR POLITECNICA DEL LITORIAL - ECUADOR 2009)

“SISTEMAS DE PAGO PARA COMERCIO ELECTRONICO “

Este proyecto de investigación tiene los siguientes objetivos generales identificar los principales sistemas de pagos de comercio electrónico utilizados en el modelo Business to Customer, proponer una guía de integración de estos sistemas de pago en un sitio web. También tiene como objetivos específicos comparar alternativas exitosas de pago e identificar similitudes y diferencias, priorizar alternativas de métodos de pago e identificar los procesos generales de integración o contratación de las alternativas elegidas. (CENTRO DE INVESTIGACION EN MATEMATICAS CIMAT – MEXICO 2012)

“SISTEMAS Y ENTORNOS DE PAGO PARA LA ADQUISICION DE CONTENIDOS Y SERVICIOS ELECTRONICOS”

Este trabajo de tesis presenta distintas soluciones. Por un lado, presenta dos soluciones de pago basadas en monedero inteligente que pretenden facilitar el uso de estos dispositivos de una forma progresiva. En primer lugar, facilitando el pago con los monederos existentes y,

en segundo lugar, proponiendo un nuevo monedero que facilite su uso para pagos en la Web. (UNIVERSIDAD DE MURCIA – ESPAÑA 2009)

1.2.3 Antecedentes internacionales

Con relación a aplicaciones similares, se han revisado los siguientes trabajos:

"BIGO"

La primera billetera digital multiemisor del Uruguay es una realidad, que permite realizar pagos utilizando la tecnología contactless o pagos sin contacto.

BIGO permite "digitalizar" en un celular todas las tarjetas de crédito, débito y prepagas Visa, para realizar compras de forma fácil, rápida y segura, simplemente acercando el celular a una terminal POS con tecnología de pagos sin contacto. (URUGUAY)

"SAFRA WALLET"

Para utilizar el servicio no es necesario ser cliente de Banco Safra, y el proceso de registro es gratuito, simple y rápido. No es obligatorio presentar comprobante de ingresos. Con un modelo diferente al de los medios tradicionales, las transferencias, los retiros y los pagos por SafraWallet no tendrán cargo alguno. (BRASIL)

"BIMO"

Se trata de una estrategia con la que la banca espera que se incremente la bancarización en el país. En total 29 entidades forman parte de la plataforma. De ellas, 16 son bancos y el resto son cooperativas y redes de cooperativas. Pablo Narváez, gerente general de Banred, explicó que BIMO está diseñada para personas que ya están bancarizadas, pero también por personas

no tienen cuentas en el sistema financiero, que podrán usar la plataforma para la apertura de una cuenta básica. (ECUADOR)

1.2.4 Antecedentes nacionales

Con relación a aplicaciones similares, se han revisado los siguientes trabajos:

“TIGO MONEY”

Tigo Money es el primer servicio de billetera móvil en Bolivia que revolucionará la manera en que manejas tu dinero.

A través de Tigo Money conviertes tu dinero efectivo en dinero electrónico que se almacena en una cuenta en tu celular. Con este dinero electrónico puedes realizar:

Transferencia a otras cuentas, pagar por bienes o servicios básicos: agua, luz, telefonía, cable, también pagar o recibir sueldos.

“BILLETERA VIVA”

Es un medio de pago casado al número de teléfono celular del cliente, es una alternativa que compite con los canales ya usados habitualmente. En la actualidad el servicio de billetera móvil es ofrecido por el BNB y BCP. Los requisitos para tener una billetera móvil es tener CI vigente y contar con una línea VIVA activa.

“BILLETERA ENTEL”

La estatal Entel Financiera ingresó a la era de las transacciones electrónicas y presentó este lunes el servicio de Billetera Móvil para operaciones bancarias como depósitos y transferencias, cancelación de salarios a trabajadores, bonos y remesas. el gerente general de la Empresa de Servicios de Pago Móvil Entel Financiera, Reynaldo Marconi, informó de la

iniciativa en una conferencia de prensa que ofreció junto con el presidente del Banco Central de Bolivia (BCB).

1.3 Planteamiento del problema

En el Banco de Crédito de Bolivia uno de los principales objetivos es ofrecer soluciones financieras con la mejor tecnología, calidad y servicio al cliente. Dicha tecnología tiene que estar a la vanguardia y exigencias del mercado, por lo tanto, se tiene que estar innovando constantemente para cumplir con los clientes y no clientes del banco construyendo relaciones de largo plazo y por ende pérdidas económicas.

1.3.1 Problema general

Actualmente el Banco de Crédito de Bolivia tiene problemas y deficiencias al momento de ofrecer servicios financieros a personas que no tienen una cuenta bancaria, lo que genera una gran pérdida de clientes para el banco.

1.3.2 Problemas específicos

- Las transferencias o envió de dinero actualmente se realizan con una cuenta bancaria, generando que las personas estén obligadas a abrir una cuenta bancaria en ventanillas.
- Las transferencias a otros bancos para personas sin cuenta bancaria se realizan de manera presencial, generando largas filas y molestia para los clientes e incomodidad.
- Los pagos de servicios como telecomunicaciones, agua, electricidad. para las personas sin cuenta bancaria se realizan de manera presencial, lo que genera que la mayoría de las sucursales estén saturadas.

- Las personas sin cuenta bancaria prefieren utilizar aplicaciones como Tigo Money que ofrecen servicios financieros similares a los que una entidad financiera, lo que genera una gran pérdida de futuros clientes para el banco.
- Las personas que no tienen una cuenta bancaria no pueden portar su dinero en un medio que no sea en efectivo o físico.

1.4 Objetivos

1.4.1 Objetivo general

Desarrollar una aplicación Android que resuelva los problemas de servicios financieros a personas que no tienen una cuenta bancaria, y que cubra las necesidades de los clientes del Banco de Crédito de Bolivia, previo diagnóstico para describir los procesos de pagos y transferencias.

1.4.2 Objetivos específicos

- Sistematizar los envíos de dinero para que los clientes no estén obligados a abrir una cuenta bancaria.
- Sistematizar de transferencias a otros bancos para que disminuya las filas de clientes generadas en plataforma.
- Sistematizar los pagos de servicios para que los clientes realicen sus pagos de manera no presencial.
- Realizar una aplicación que ofrezca todos los servicios de la competencia y que este a la altura del mercado nacional de tal manera que más personas sean clientes del banco.
- Realizar una aplicación que permita portar el dinero de manera digital facilitando el pago electrónico.

1.5 Justificación

1.5.1 Justificación técnica

Se justifica técnicamente porque se aplicarán métodos, técnicas y herramientas de última generación, tales como: Scrum para la ingeniería del proyecto, se aplicarán la combinación de Java, C# y SQL SERVER que conjuntamente tienen ciertas ventajas para el desarrollo de aplicaciones móviles como la programación orientada a objetos además que son estándares que se utilizan en la empresa.

1.5.2 Justificación económica

Con la aplicación móvil, el Banco de Crédito de Bolivia, se beneficia de la siguiente manera:

- La aplicación móvil realizara diversos procesos de manera automática, minimizando costos y tiempo.
- El banco podrá tomar decisiones que afecten la economía de la institución en base a la información recolectada como por ejemplo transferencias, pagos de servicios, etc.
- Todas las operaciones transaccionales no tendrán ninguna comisión.
- La licencia de uso de la aplicación móvil, es libre.

1.5.3 Justificación social

Se beneficiará los clientes y el Banco de Crédito de Bolivia porque los clientes podrán tener acceso a las transacciones y pagos de servicios sin tener que ir a un banco o tener una cuenta bancaria, y el banco podrá utilizar eficientemente la información de las operaciones financieras de los clientes para poder tomar decisiones que afecten la imagen del banco.

1.6 Metodología

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos. (Proyectos Agiles, https://proyectosagiles.org/que-es-scrum/)

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

El proceso

En Scrum un proyecto se ejecuta en ciclos temporales cortos y de duración fija (iteraciones que normalmente son de 2 semanas, aunque en algunos equipos son de 3 y hasta 4 semanas, límite máximo de feedback de producto real y reflexión). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite.

El proceso parte de la lista de objetivos/requisitos priorizada del producto, que actúa como plan del proyecto. En esta lista el cliente (Product Owner) prioriza los objetivos balanceando el valor que le aportan respecto a su coste (que el equipo estima considerando la Definición de Hecho) y quedan repartidos en iteraciones y entregas.

Las actividades que se llevan a cabo en Scrum son las siguientes (los tiempos indicados son para iteraciones de 2 semanas):

Planificación de la iteración

El primer día de la iteración se realiza la reunión de planificación de la iteración. Tiene dos partes:

- **Selección de requisitos (2 horas).** El cliente presenta al equipo la lista de requisitos priorizada del producto o proyecto. El equipo pregunta al cliente las dudas que surgen y selecciona los requisitos más prioritarios que prevé que podrá completar en la iteración, de manera que puedan ser entregados si el cliente lo solicita.
- Planificación de la iteración (2 horas). El equipo elabora la lista de tareas de la iteración necesarias para desarrollar los requisitos seleccionados. La estimación de esfuerzo se hace de manera conjunta y los miembros del equipo se autoasignan las tareas, se autoorganizan para trabajar incluso en parejas (o grupos mayores) con el fin de compartir conocimiento (creando un equipo más resiliente) o para resolver juntos objetivos especialmente complejos.

Ejecución de la iteración

Cada día el equipo realiza una reunión de sincronización (15 minutos), normalmente delante de un tablero físico o pizarra (Scrum Taskboard). El equipo inspecciona el trabajo que el resto está realizando (dependencias entre tareas, progreso hacia el objetivo de la iteración, obstáculos que pueden impedir este objetivo) para poder hacer las adaptaciones necesarias que permitan cumplir con la previsión de objetivos a mostrar al final de la iteración.

Durante la iteración el Facilitador (Scrum Master) se encarga de que el equipo pueda mantener el foco para cumplir con sus objetivos.

- Elimina los obstáculos que el equipo no puede resolver por sí mismo.

- Protege al equipo de interrupciones externas que puedan afectar el objetivo de la iteración o su productividad.

Durante la iteración, el cliente junto con el equipo refina la lista de requisitos (para prepararlos para las siguientes iteraciones) y, si es necesario, cambian o replanifican los objetivos del proyecto (10%-15% del tiempo de la iteración) con el objetivo de maximizar la utilidad de lo que se desarrolla y el retorno de inversión.

Inspección y adaptación

El último día de la iteración se realiza la reunión de revisión de la iteración. Tiene dos partes:

- **Revisión (demostración) (1,5 horas).** El equipo presenta al cliente los requisitos completados en la iteración, en forma de incremento de producto preparado para ser entregado con el mínimo esfuerzo. En función de los resultados mostrados y de los cambios que haya habido en el contexto del proyecto, el cliente realiza las adaptaciones necesarias de manera objetiva, ya desde la primera iteración, replanificando el proyecto.
- **Retrospectiva (1,5 horas).** El equipo analiza cómo ha sido su manera de trabajar y cuáles son los problemas que podrían impedirle progresar adecuadamente, mejorando de manera continua su productividad. El Facilitador se encargará de eliminar o escalar los obstáculos identificados que estén más allá del ámbito de acción del equipo.

1.7 Herramientas

Lenguajes de programación

Nombre	Características	Ventajas	Desventajas	Licencia
Java	Es un lenguaje para la creación de sitios web dinámicos, acrónimo de Java.	El código está bien estructurado y resulta fácil de leer si se conoce bien el lenguaje.	El alojamiento web requiere tener instalado un servidor Tomcat	GNU
ASP	El lenguaje *ASP* (Active Server Pages) se trata de un lenguaje de programación web desarrollado por Microsoft en 1996 para poder crear sitios web dinámicos.	Admite la programación con Visual Basic Script lo que facilita su implementación.	Muchos hostings y alojamientos web no lo soportan por su alto coste. *ASP.NET* necesita tener instalado IIS con el Framework .Net.	EULA
Ruby	Es un lenguaje de programación interpretado, reflexivo y orientado a objetos, creado por el programador japonés Yukihiro "Matz" Matsumoto.	Ruby puede cargar librerías de extensiones dinámicamente si el (Sistema Operativo) lo permite.	Primero se tiene que aprender el lenguaje Ruby.	MIT

Tabla 1.1. Tabla comparativa de lenguajes de programación

Fuente: Elaboración propia

Se seleccionó Java porque es con la que el banco tiene contratos de soporte además que el área de arquitectura pide que sea con tecnología nativa para desarrollo móvil.

Gestores de base de datos

Nombre	Características	Ventajas	Desventajas	Licencia
Microsoft SQL Server	Es un sistema de gestión de bases de datos relacionales basado en el lenguaje Transact-SQL, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea	Seguridad: SQL permite permisos a nivel de servidor, seguridad en tablas, permitir o no lectura, escritura, ejecución; seguridad en los procedimientos almacenados	La principal desventaja de Microsoft SQL SERVER es la enorme cantidad de memoria RAM que utiliza para la instalación y utilización del software.	EULA
Oracle DBMS	Es un sistema de gestión de base de datos relacional (o RDBMS por el acrónimo en inglés de Relational Data Base Management System), fabricado por Oracle Corporation.	Oracle es la base de datos con más orientación hacía INTERNET	Un error frecuente consiste en pensar que basta instalar el Oracle en un servidor y enchufar directamente las aplicaciones clientes. Un Oracle mal configurado puede ser desesperantemente lento.	Privativa
PostgreSQL	Es un Sistema de gestión de bases de datos relacional orientado a objetos y libre, publicado bajo la licencia PostgreSQL, similar a la BSD o la MIT.	Su sintaxis SQL es estándar y fácil de aprender.	En comparación con MySQL es más lento en inserciones y actualizaciones, ya que cuenta con cabeceras de intersección que no tiene MySQL.	PostgreSQL License

Tabla 1.2. Tabla comparativa de gestores de base de datos

Fuente: Elaboración propia

Se selecciona Microsoft SQL Server porque es un sistema de gestión de base de datos relacional, multihilo y multiusuario seguramente el más usado en aplicaciones de Microsoft, la probabilidad

de corromper datos, incluso si los errores no se producen en el propio gestor síno en el sistema en el que está.

Servidor Web

Nombre	Características	Ventajas	Desventajas	Licencia
XAMPP	Es un servidor independiente de plataforma, software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL.	Si alguna vez has intentado instalar Apache, sabes que no es una tarea fácil, sin embargo, con XAMPP todo es diferente.	No se pueden actualizar individualmente las versiones de los programas que instala.	GNU
WAMPSERVER	Es un entorno de desarrollo web para Windows en el cual se podrán crear aplicaciones web con Apache, PHP y base de datos en MySQL.	La instalación modificará los archivos de configuración (*.conf) con la ruta donde finalmente se ubicará el programa.	El desarrollador obtiene poco conocimiento en configuración de Apache, PHP y MySQL.	GNU
Windows Server 2012	Es un sistema operativo destinado a servidores lanzado por Microsoft. Es la versión para servidores de Windows 8 y es el sucesor de Windows Server 2008 R2.	Seguridad avanzada Arquitectura confiable Rendimiento mejorado	Costo alto No hay puente intermedio Sin opción de virtualización Los mensajes tienden a fallar Requiere altos conocimientos	Microsoft CLUF (EULA)

Tabla 1.2. Tabla comparativa de lenguajes de servidores web

Fuente: Elaboración propia

Se seleccionó Windows Server 2012 porque es con la que el banco tiene contratos de soporte además que el área de arquitectura pide que sea con tecnología Microsoft.

1.8 Límites y alcances

1.8.1 Límites

La aplicación móvil se diseñará y desarrollará de acuerdo a los requerimientos y estándares específicos del área de Desarrollo y Soluciones del Banco de Bolivia.

El sistema web se delimitará en los siguientes aspectos:

- El desarrollo de la aplicación no contempla la recarga de dinero a la cuenta billetera.
- El desarrollo de la aplicación no contempla los limites transaccionales.
- La aplicación será desarrollada solo para el sistema operativo Android con la versión 4.2 superior.
- Las transferencias a otras cuentas, pagos de servicios serán a través de servicios web internos del banco.
- Para el depósito y retiro por cajero automáticos serán a través de servicios web internos del banco.

1.8.2 Alcances

En el presente proyecto se proyecta desarrollar los siguientes módulos:

- Módulo de inicio de sesión.
- Módulo de registro.
- Módulo de transferencias.
- Módulo de pago de servicios.
- Módulo de saldo y movimientos.
- Módulo de cambio y reseteo de pin.
- Módulo de retiro de dinero.

1.9 Aportes

El presente proyecto de grado, generará los siguientes aportes:

- Una aplicación móvil que permitirá realizar las operaciones financieras sin la necesidad de una cuenta bancaria o la necesidad de ir presencialmente a una entidad bancaria.
- La aplicación coadyuvará en la seguridad, integridad, disponibilidad y confidencialidad de la información financiera de los clientes.

CAPITULO II

MARCO TEORICO

2.1 Introducción

El marco teórico, que se desarrolla a continuación permitirá situar el proyecto dentro de un conjunto de conocimientos, además ofrecer una conceptualización adecuada de términos y conceptos necesarios para su entendimiento.

2.2 Antecedentes de la institución

El Banco de Crédito BCP opera en el Perú desde 1889 como Banco Italiano y a partir de 1941 como Banco de Crédito del Perú. El BCP es una institución sólida cuya vocación y principios la han hecho líder indiscutible del mercado peruano a lo largo de sus más de 115 años de impecable trayectoria.

Desde 1995 forma parte del grupo Credicorp, uno de los conglomerados financieros más importantes de Latinoamérica, que combina los negocios bancarios del BCP y del Atlantic Security Bank con los negocios de seguros de Pacífico Peruano Suiza, la empresa más grande del Perú en su rubro. Las acciones de Credicorp están inscritas en la Bolsa de Valores de Nueva York, lo que le permite acceder con facilidad a los mercados de capitales internacionales.

El Banco de Crédito de Bolivia S.A. es una empresa 100% subsidiaria del Banco de Crédito BCP. Iniciamos nuestras operaciones en el mercado boliviano en 1994 después de que adquirimos el Banco Popular, institución que se ubicaba en el puesto número trece del sistema financiero nacional.

Esta primera apuesta por Bolivia fue muy exitosa, lo que nos alentó a seguir invirtiendo en el país, adquiriendo en 1998 el Banco de La Paz y un año más tarde la cartera del Banco Boliviano Americano. Con estas compras, y en base al empuje de nuestro excelente equipo de

profesionales, fuimos ganando la preferencia y confianza de nuestros clientes hasta ubicarnos entre los principales bancos del país.

En el año 2000, comenzaron un proceso de renovación tecnológica sin precedentes con el fin de sentar las bases necesarias para otorgar a sus clientes el mejor servicio de la banca boliviana. Trajeron la más moderna tecnología bancaria disponible, interconectamos nuestros sistemas informáticos con los del Banco de Crédito BCP en Perú e introdujimos el Servimatic, su innovador sistema de filas virtuales, que revolucionó el concepto de atención al cliente no sólo en la banca sino también en otras industrias del país.

Desarrollaron modernos canales electrónicos como la Banca por Teléfono y la Banca por Internet para llevar al Banco a cualquier lugar del mundo donde nuestros clientes están.

Asimismo, invirtieron más de dos millones de dólares para ampliar su red de canales de atención al público, consolidándola como la más grande del país con 45 oficinas y 125 cajeros automáticos a nivel nacional.

En la actualidad, sus esfuerzos han hecho del Banco de Crédito de Bolivia S.A. una institución moderna, innovadora y con excelente calidad de atención. Tienen la cartera más sana y con una de las mayores coberturas del sistema, lo que aunado al sólido respaldo internacional del BCP, les pone en las mejores condiciones para seguir avanzando hacia el liderazgo indiscutible del sistema financiero nacional.

La apuesta del Banco de Crédito por Bolivia es definitiva y por ello han renovado nuestra imagen institucional, haciéndola más moderna y cercana. Su nuevo logotipo es el símbolo de que ponemos nuestro sólido respaldo internacional al servicio del país, con el compromiso de otorgar a sus clientes una experiencia bancaria simple y eficiente.

De ahora en adelante, trabajaran con mayor dedicación, accesibilidad y flexibilidad. Pondrán a disposición del mercado nuevos productos y servicios. Seguirán invirtiendo, seguirán modernizándonos, seguirán innovando y contribuyendo al desarrollo del país y de sus clientes. Todo esto, con el propósito de seguir siendo el mejor Banco de Bolivia.

2.2.1 Misión

Ofrecer soluciones financieras a personas naturales y jurídicas en Bolivia con la mejor tecnología, calidad y servicio al cliente construyendo relaciones de largo plazo.

2.2.2 Visión

Ser la Primera opción para el Cliente del Sistema Financiero Boliviano.

2.2.3 Estructura

Conformada en primera instancia por el Directorio y la Gerencia General, quienes ejercen sus funciones en conformidad al Código de Gobierno Corporativo, su respectivo Reglamento Interno del Banco y conforme a las Directrices Básicas para la Gestión de un Buen Gobierno Corporativo emitidas por la Autoridad de Supervisión del Sistema Financiero - ASFI.

2.2.4 Organigrama

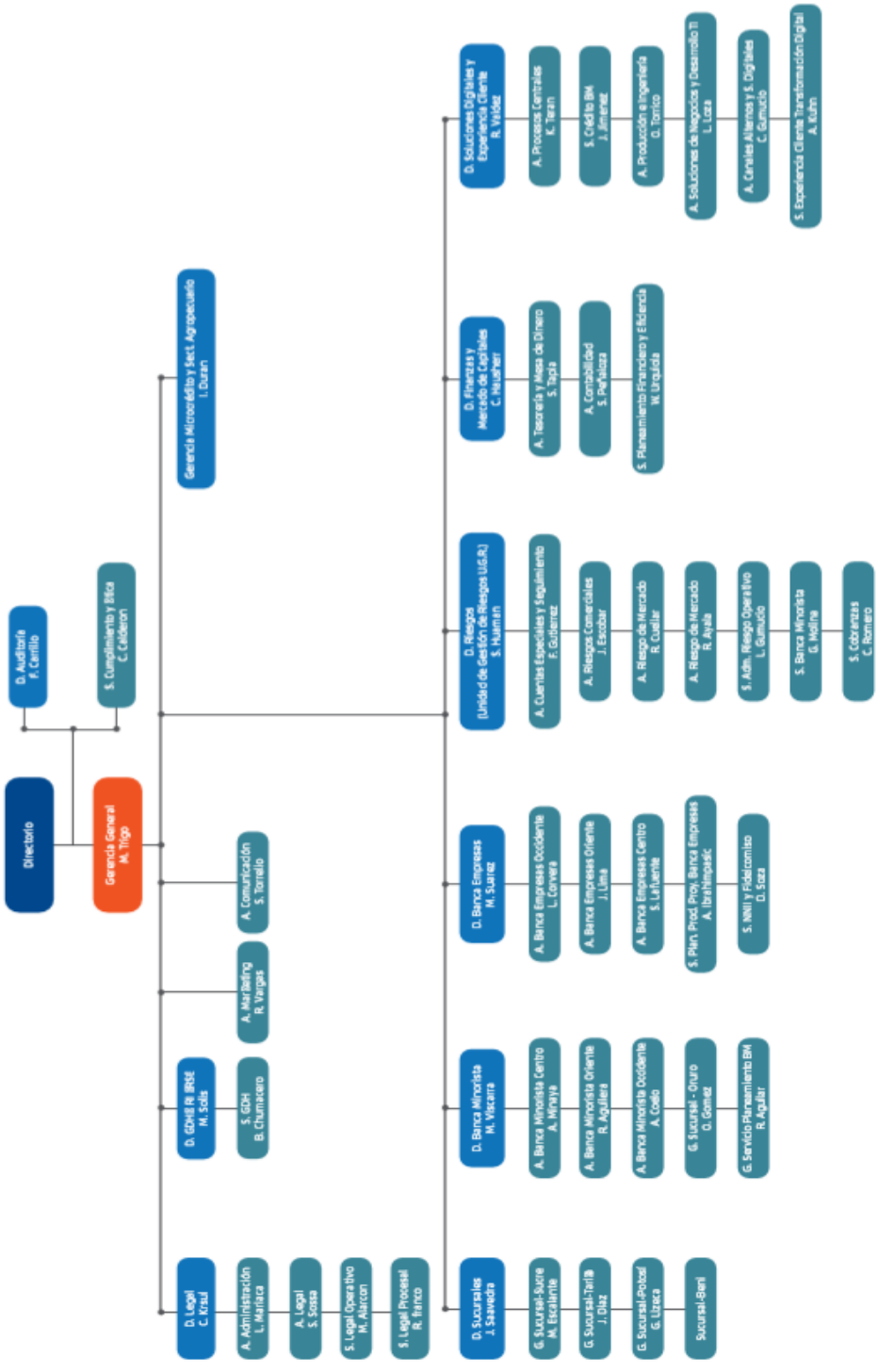


Figura 2.1. Organigrama del Banco de Crédito

Fuente: https://www.bcp.com.bo/

2.2.5 Conformación

El Directorio está conformado por 8 miembros, todos ellos hombres al igual que en la gestión 2019 y todos mayores de 50 años.

El objetivo principal del Directorio es fiscalizar la gestión y velar por la solidez financiera de la entidad. Para ello, se le confieren las siguientes atribuciones y responsabilidades:

- Evaluar, aprobar, dirigir y hacer seguimiento a la estrategia corporativa.
- Vigilar la cultura corporativa, el cumplimiento de los valores y principios éticos comunicados a toda la organización.
- Aprobar los principales planes de acción, el presupuesto y planes de negocios.
- Establecer mecanismos correctivos con base en observaciones y recomendaciones efectuadas, por auditoría interna, externa y/o ASFI.
- Evaluar regularmente las prácticas de gobierno corporativo con las cuales opera, realizando cambios a medida que éstos se hagan necesarios.

La administración queda a cargo de su planta gerencial de la que forma parte la Gerencia General y 10 Gerencias de División:

I. Auditoría
II. Legal
III. Gestión de Desarrollo Humano, Relaciones Institucionales y Responsabilidad Social
IV. Microcrédito y Sector Agropecuario
V. Sucursales
VI. Minorista
VII. Empresas
VIII. Riesgos

IX. Finanzas

X. Sistemas y Procesos

2.2.5.1 Comités corporativos

- Unidad de Gestión de Riesgos (UGR).
- Administración de Riesgo Operativo (ARO).
- Asset and Liability (ALCO).

2.2.5.2 Comités regulatorios

- De Gobierno Corporativo
- Ética
- Cumplimiento
- Auditoría
- Seguridad Física
- Tecnología de la información
- Operativo de tecnología de la información
- Seguridad de Información

2.2.5.3 Comités internos

- Inversiones
- Gestión
- Tasas y Tarifas
- Productividad
- Evaluación de Nuevos Productos
- Eventos PCN (Continuidad de Negocios)
- Créditos (Nacional)

2.2.6 Modelamiento del sistema actual

Los procesos que actualmente se realizan y tienen carácter primario son:

- Proceso de apertura de cuenta
- Proceso de transferencias entre cuentas
- Proceso de depósito a cuenta
- Proceso de pago de servicios básicos

2.2.6.1 Proceso de apertura de cuenta

El siguiente diagrama de flujo representa los pasos que sigue un cliente para aperturar una cuenta.

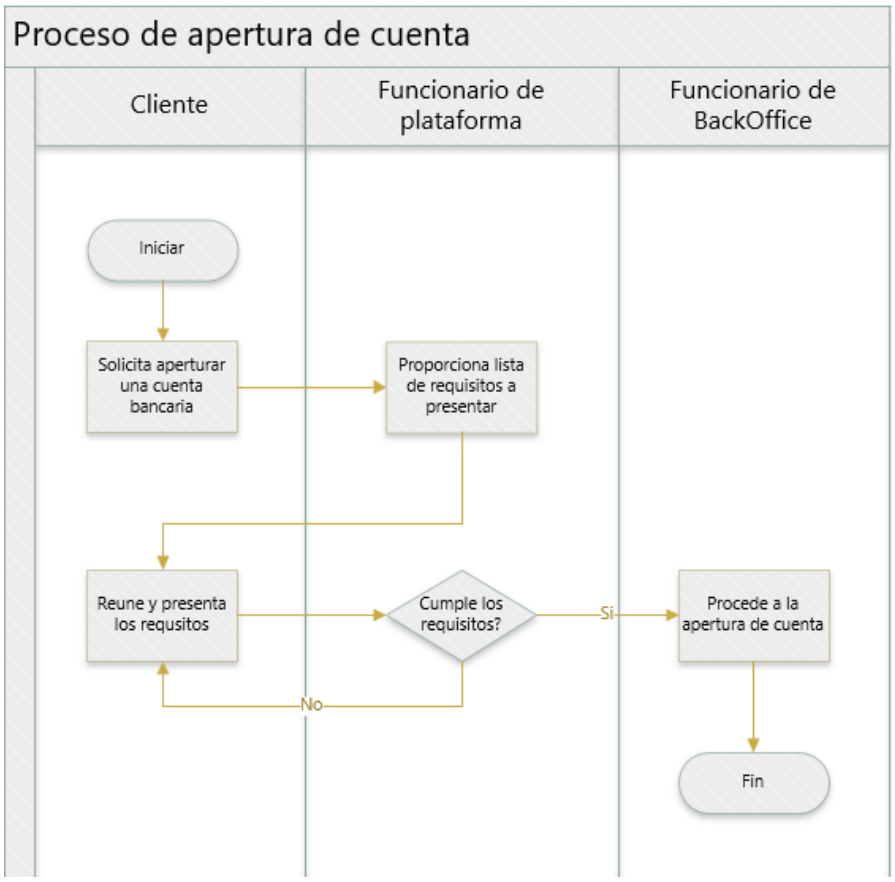


Figura 2.2. Diagrama de flujo del proceso de apertura de cuenta

Fuente: Elaboración propia

2.2.6.2 Proceso de transferencias entre cuentas

El siguiente diagrama de flujo representa los pasos que sigue un cliente para realizar transferencias entre cuentas.

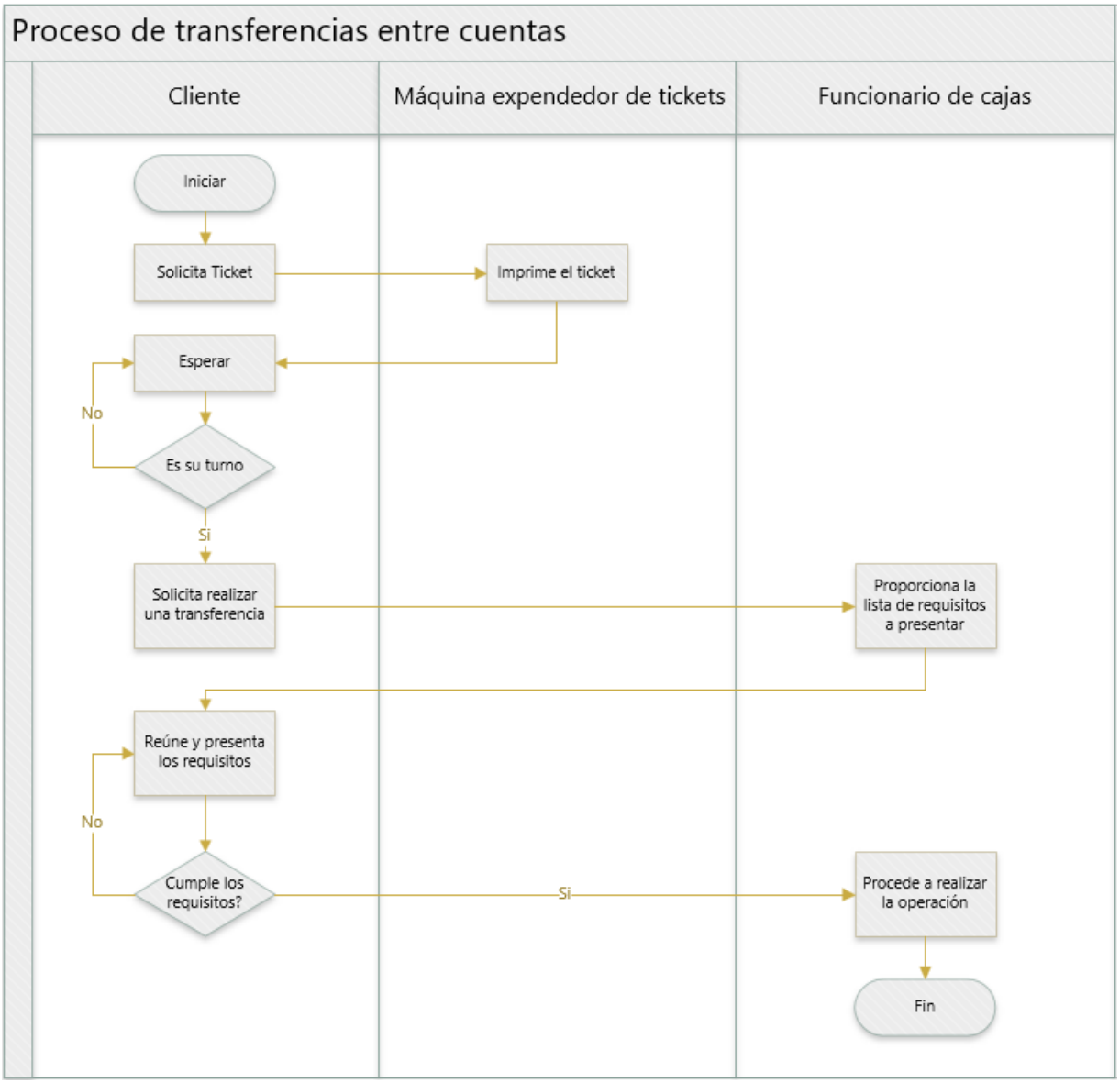


Figura 2.3. Diagrama de flujo del proceso de transferencias entre cuentas

Fuente: Elaboración propia

2.2.6.3 Proceso de depósito a cuenta

El siguiente diagrama de flujo representa los pasos que sigue un cliente para realizar el depósito a cuenta.

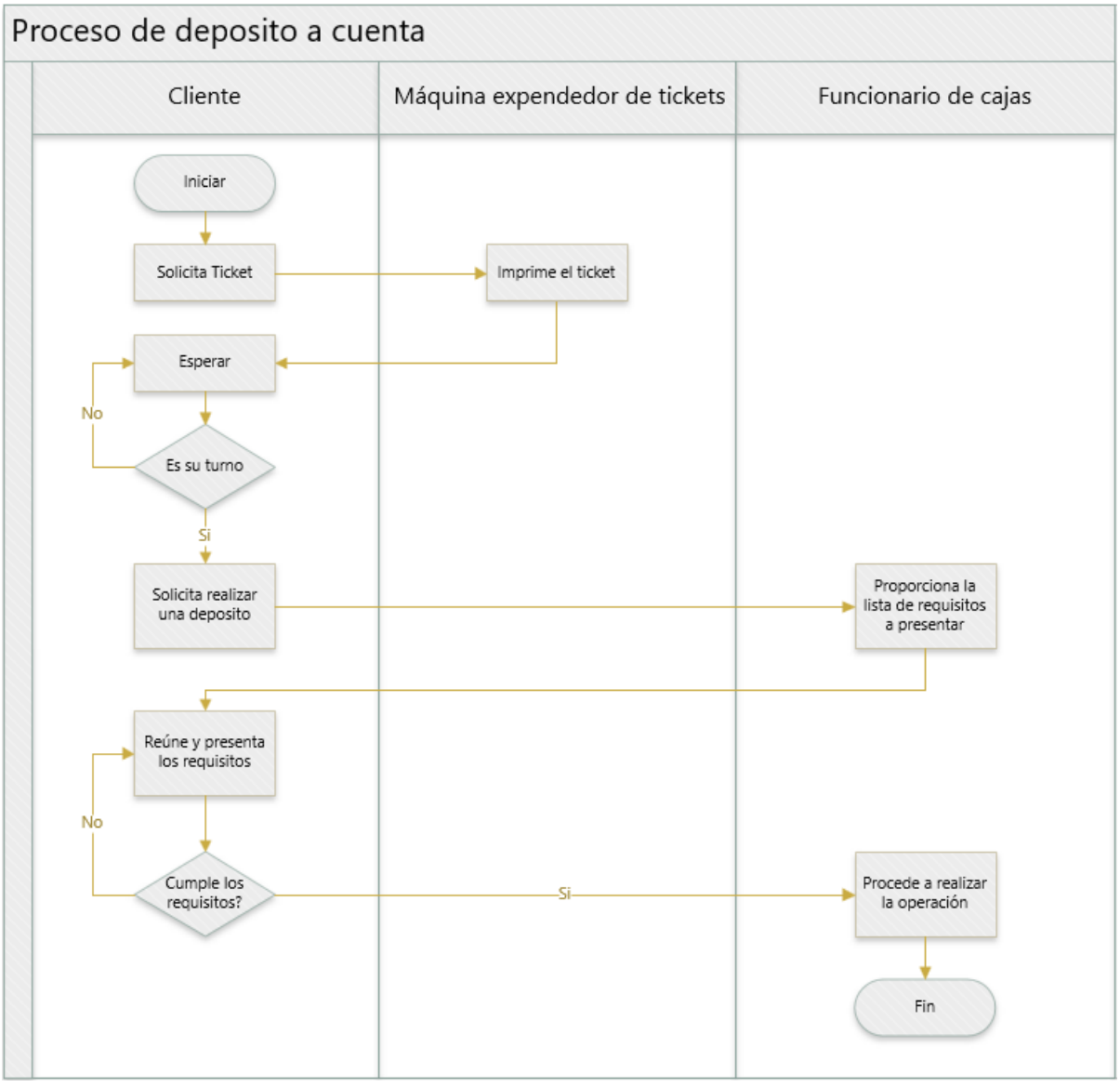


Figura 2.4. Diagrama de flujo del proceso de proceso de depósito a cuenta

Fuente: Elaboración propia

2.2.6.4 Proceso de pago de servicios básicos

El siguiente diagrama de flujo representa los pasos que sigue un cliente para realizar los pagos de servicios básicos.

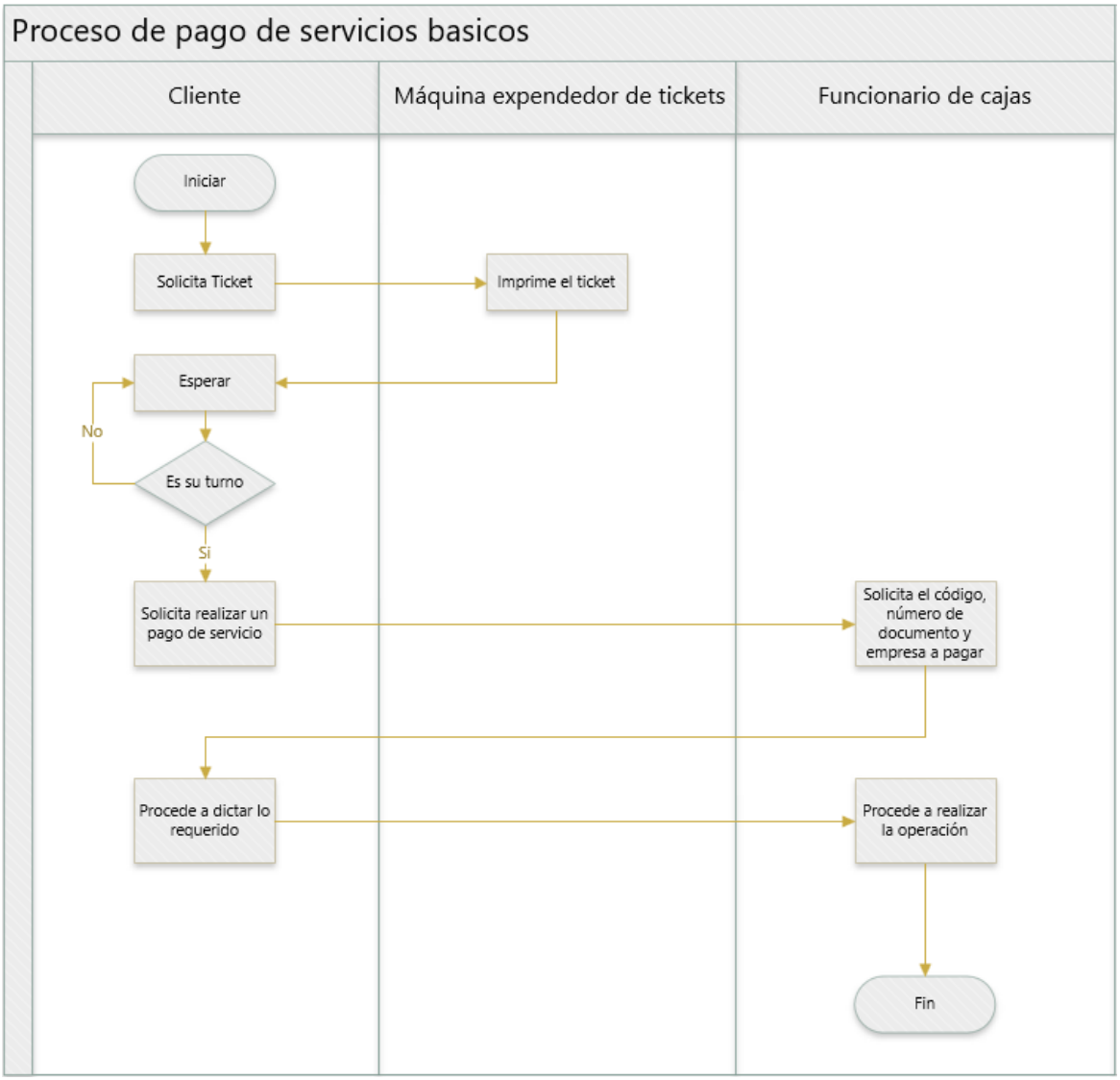


Figura 2.5. Diagrama de flujo del proceso de pago de servicios básicos

Fuente: Elaboración propia

2.3 Sistema

Según la publicación en Wikipedia (https://es.wikipedia.org/wiki/Sistema, fecha 06 de julio 2020), el término sistema (proveniente del latín systēma) se define como "un objeto complejo cuyas partes o componentes se relacionan con al menos algún otro componente". Según el sistemismo, todos los objetos son sistemas o componentes de otro sistema.

Según la Teoría General de Sistemas de Cajizo Perez es un conjunto ordenando de componentes o elementos interrelaciones, interdependientes e interactuantes, que tienen por la finalidad el logro de objetivos determinados en un plan.

Según Idalberto Chiavenato, un sistema puede definirse como un conjunto de elementos dinámicamente relacionales, en interacción que desarrollan una actividad para lograr un objetivo o propósito operando como datos, energía, materia unidos al ambiente que rodea el sistema para suministrar información, energía o materia.

Por lo tanto, un sistema de información es un conjunto de elementos organizados y orientados al tratamiento y administración de datos e información para cubrir una necesidad u objetivo.

2.4 Billetera Móvil

Según el sitio Rankia (https://www.rankia.co/blog/mejores-cdts/4010471-billetera-movil-que-como-funciona-bancos, fecha abril de 2020) la billetera móvil es una aplicación (programa informativo) para dispositivos inteligentes, mediante el cual podemos guardar dinero u asociar otros medios de pago (las tarjetas de crédito y débito), para efectuar o recibir pagos, y realizar transferencias a terceros. La billetera móvil al igual que las tarjetas de crédito y débito está asociada a una cuenta bancaria.

Es por ello que su funcionamiento es similar a las tarjetas de débito o crédito, pero la principal diferencia es que la información en este caso quedará almacenada en el smarthpone o en la nube en lugar de en el chip del plástico.

A diferencia de las apps de "pago móvil", este tipo de aplicaciones no necesariamente deben estar vinculadas a un banco y su infraestructura tecnológica.

2.5 Nociones básicas sobre aplicaciones móviles

Según el artículo de la Comisión Federal de Comercio de Washington (https://www.consumidor.ftc.gov/articulos/s0018-aplicaciones-moviles-que-son-y-como-funcionan, fecha septiembre de 2011) indica que una aplicación móvil es un programa que usted puede descargar y al que puede acceder directamente desde su teléfono o desde algún otro aparato móvil como por ejemplo una tablet o un reproductor MP3.

Se necesita un Smartphone o algún otro aparato móvil con acceso a internet. No todas las aplicaciones funcionan en todos los dispositivos móviles. Cuando se compra uno de estos aparatos móviles Android, Apple, Microsoft y BlackBerry tienen tiendas de aplicaciones que operan en línea en las cuales se pude buscar, descargar e instalar las aplicaciones. Algunos comerciantes minoristas también operan tiendas de aplicaciones en internet. Se debe usar una tienda que le ofrezca las aplicaciones que funcionen con el sistema operativo de su dispositivo móvil.

2.6 Versiones disponibles de Android

El sistema operativo Android, al igual que los propios teléfonos móviles, ha evolucionado rápidamente, acumulando una gran cantidad de versiones, desde la 1.0 para el QWERTY HTC G1, hasta la 11 que acaba de salir al mercado.

2.6.1 Cupcake: Android Versión 1.5

Características: Widgets, teclado QWERTY virtual, copy & paste, captura de videos y poder subirlos a Youtube directamente (Robledo, 2013).

2.6.2 Donut: Android Versión 1.6

Características: Añade a la anterior la mejoría de la interfaz de la cámara, búsqueda por voz, y navegación en Google Maps (Robledo, 2013).

2.6.3 Eclair: Android Versión 2.0/2.1

Características: Mejoras en Google Maps, salvapantallas animado, incluye zoom digital para la cámara, y un nuevo navegador de internet (Robledo, 2013).

2.6.4 Froyo: Android Versión 2.2

Características: Incluye hostpost Wifi, mejora de la memoria, más veloz, Microsoft Exchange y video-llamada (Robledo, 2013).

2.6.5 Ginger Bread: Android Versión 2.3

Características: Mejoras del consumo de batería, el soporte de video online y el teclado virtual, e incluye soporte para pagos mediante NFC2 (Robledo, 2013).

2.6.6 Honey Comb: Android Versión 3.0/3.4

Características: Mejoras para tablets, soporte Flash y Divx, integra Dolphin, multitarea pudiendo cambiar de aplicación dejando las demás en espera en una columna, widgets y homepage personalizable (Robledo, 2013).

2.6.7 Ice Cream Sandwich: Android Version 4.0

Características: Multiplataforma (tablets, teléfonos móviles y netbooks), barras de estado, pantalla principal con soporte para 3D, widgets redimensiónales, soporte usb para teclados, reconocimiento facial y controles para PS3 (Robledo, 2013).

2.6.8 Jelly Bean Android Versión 4.1

En esta versión se hace hincapié en mejorar un punto débil de Android: la fluidez de la interfaz de usuario. Con este propósito se incorporan varias técnicas: sincronismo vertical, triple búfer y aumento de la velocidad del procesador al tocar la pantalla (Robledo, 2013).

2.6.9 KitKat Android Versión 4.4

El principal objetivo de la versión 4.4 es hacer que Android esté disponible en una gama aún más amplia de dispositivos, incluyendo aquellos con tamaños de memoria RAM de solo 512 MB. Para ello, todos los componentes principales de Android han sido recortados para reducir sus requerimientos de memoria, y se ha creado una nueva API que permite adaptar el comportamiento de la aplicación en dispositivos con poca memoria (Robledo, 2013).

2.6.10 Lollipop Android Versión 5.0

La novedad más importante de Lollipop es la extensión de Android a nuevas plataformas, incluyendo Android Wear, Android TV y Android Auto. Hay un cambio significativo en la arquitectura, al utilizar la máquina virtual RT en lugar de Dalvik. Esta novedad ya había sido incorporada en la versión anterior a modo de prueba. ART mejora de forma considerable el tiempo de ejecución del código escrito en Java. Además, se soporta dispositivos de 64bits en procesadores ARM y x86 y MIPS. Muchas aplicaciones del sistema (Chrome, Gmail, …) se han incorporado en código nativo para una ejecución más rápida (Robledo, 2013).

2.6.11 Mashmallow Android Versión 6.0

Una de las novedades más interesantes es el administrador de permisos. Los usuarios podrán conocer o retirar ciertos permisos a cada aplicación. Con esto el sistema da mucha más proyección a la privacidad de los usuarios (Developer Android, https://developer.android.com/about/versions/marshmallow, 2020).

2.6.12 Nougat Android Versión 7.0/7.1

En teléfonos tablets con Android 7.0 los usuarios pueden ejecutar dos apps en paralelo o una encima de la otra en el modo de pantalla dividida. También pueden modificar el tamaño de las apps arrastrando la línea divisora que se encuentra entre ellas.

En los dispositivos con Android TV, las apps pueden habilitar de forma automática el modo "picture-in-picture". Esto le permite continuar mostrando contenido mientras el usuario explora otras apps o interactúa con ellas (Developer Android, https://developer.android.com/about/versions/nougat, 2020).

2.6.13 Oreo Android Versión 8.0/8.1

Presenta canales de notificación que te permiten crear un canal que el usuario puede personalizar para cada tipo de notificación que desees mostrar. Los usuarios pueden posponer las notificaciones para que reaparezcan posteriormente. Las notificaciones vuelven a reaparecer con el mismo nivel de importancia con el que aparecieron por primera vez (Developer Android, https://developer.android.com/about/versions/oreo, 2020).

2.6.14 Pie Android Versión 9

Mejora la privacidad del usuario, Android 9 introduce varios cambios de comportamiento como la limitación obtenida en análisis de Wi-Fi, y nuevas reglas de permisos y grupos de permisos relacionados con llamadas telefónicas. Estos cambios afectan a todas las apps que se ejecuten en Android 9, sin importar la versión de SDK a la que se orienten (Developer Android, https://developer.android.com/about/versions/pie, 2020).

2.6.15 Pie Android Versión 10

Esta versión de Android incluirá un tema oscuro para todo el sistema. Algo que los usuarios de esta plataforma venían pidiendo hace tiempo. Tendrá un modo de activación que dependerá del usuario, lo que hará posible dejarlo activado siempre, desactivado o también activo dependiendo de una hora previamente configurada. Tanto el lanzador de aplicaciones, los ajustes y otras aplicaciones se pondrán en su modo oscuro. Por otra parte, se renovarán los permisos, de manera que se podrá conocer con estadísticas los permisos más usados por las aplicaciones y filtrar por tipo de permisos. Otra novedad es que Android 10 permitirá a las operadoras bloquear las SIM de sus competidores, de tal manera que se podrá bloquear la segunda ranura de los dispositivos con doble SIM. Esta restricción de bloqueo de la SIM se aplica de inmediato y se mantendrá, aunque se reinicie el teléfono o se restablezca la configuración de fábrica. Estos cambios no afectaran en ningún momento a los móviles libres, es decir, que no estén enlazados a ninguna compañía. Otro punto importante es que Android incluirá su propio soporte nativo de reconocimiento facial siguiendo, de esta forma, las pautas de seguridad conocidas en los dispositivos Apple (Developer Android, https://developer.android.com/about/versions/10, 2020).

2.6.16 Pie Android Versión 11

Android 11 es el sistema operativo de Google que se revelará a mediados de 2020. Entre sus nuevas características, se rumorea que Scoped Storage mejoraría su funcionamiento,108de manera que el sistema permitiría que las velocidades de lectura de memoria fuesen mucho más rápidas, mejorando así la seguridad y evitaría la necesidad de otorgar permisos a cada aplicación nueva. Por otra parte, se espera un mejorado Modo Oscuro (característica diseñada para facilitar la lectura, al reducir el contraste, y a ofrecer un consumo de energía menor) para todos los usuarios, mayor integración de la IA y otras optimizaciones en el sistema que garanticen el rendimiento del dispositivo móvil (Developer Android, https://developer.android.com/preview, 2020).

2.7 Web Services

Según el sitio de la Universidad de Alicante (http://www.jtech.ua.es/j2ee/publico/servc-web-2012-13/sesion01-apuntes.html, fecha 26 de junio de 2014) un Servicio Web es un componente al que podemos acceder mediante protocolos Web estándar, utilizando XML para el intercambio de información.

Normalmente nos referimos con Servicio Web a una colección de procedimientos (métodos) a los que podemos llamar desde cualquier lugar de Internet o de nuestra intranet, siendo este mecanismo de invocación totalmente independiente de la plataforma que utilicemos y del lenguaje de programación en el que se haya implementado internamente el servicio (Universidad de Alicante, 2014).

Cuando conectamos con un servidor web desde nuestro navegador, el servidor nos devuelve la página web solicitada, que es un documento que se mostrará en el navegador para que lo

visualice el usuario, pero es difícilmente entendible por una máquina. Podemos ver esto como web para humanos. En contraposición, los Servicios Web ofrecen información con un formato estándar que puede ser entendido fácilmente por una aplicación. En este caso estaríamos ante una web para máquinas (Universidad de Alicante, 2014).

Los servicios Web son componentes de aplicaciones distribuidas que están disponibles de forma externa. Se pueden utilizar para integrar aplicaciones escritas en diferentes lenguajes y que se ejecutan en plataformas diferentes. Los servicios Web son independientes de lenguaje y de la plataforma gracias a que los vendedores han admitido estándares comunes de Servicios Web (Universidad de Alicante, 2014).

El WC3 (World Wide Web Consortium) define un servicio Web como un sistema software diseñado para soportar interacciones máquina a máquina a través de la red. Dicho de otro modo, los servicios Web proporcionan una forma estandar de interoperar entre aplicaciones software que se ejecutan en diferentes plataformas. Por lo tanto, su principal característica su gran interoperabilidad y extensibilidad, así como por proporcionar información fácilmente procesable por las máquinas gracias al uso de XML. Los servicios Web pueden combinarse con muy bajo acoplamiento para conseguir la realización de operaciones complejas. De esta forma, las aplicaciones que proporcionan servicios simples pueden interactuar con otras para "entregar" servicios sofisticados añadidos (Universidad de Alicante, 2014).

2.7.1 Características de los Servicios Web

Las características deseables de un Servicio Web son:

- Un servicio debe poder ser accesible a través de la Web. Para ello debe utilizar protocolos de transporte estándares como HTTP, y codificar los mensajes en un

lenguaje estándar que pueda conocer cualquier cliente que quiera utilizar el servicio (Universidad de Alicante, 2014).

- Un servicio debe contener una descripción de sí mismo. De esta forma, una aplicación podrá saber cuál es la función de un determinado Servicio Web, y cuál es su interfaz, de manera que pueda ser utilizado de forma automática por cualquier aplicación, sin la intervención del usuario. (Universidad de Alicante, 2014)
- Debe poder ser localizado. Deberemos tener algún mecanismo que nos permita encontrar un Servicio Web que realice una determinada función. De esta forma tendremos la posibilidad de que una aplicación localice el servicio que necesite de forma automática, sin tener que conocerlo previamente el usuario (Universidad de Alicante, 2014).

2.7.2 Tipos de servicios Web

A nivel conceptual, un servicio es un componente software proporcionado a través de un endpoint accesible a través de la red. Los servicios productores y consumidores utilizan mensajes para intercambiar información de invocaciones de petición y respuesta en forma de documentos auto-contenidos que hacen muy pocas asunciones sobre las capacidades tecnológicas de cada uno de los receptores (Universidad de Alicante, 2014).

Los servicios pueden interconectarse a través de la red. En una arquitectura orientada a servicios, cualquier interacción punto a punto implica dos endpoints: uno que proporciona un servicio, y otro de lo consume. Es decir, que un endpoint es cada uno de los "elementos", en nuestro caso nos referimos a servicios, que se sitúan en ambos "extremos" de la red que sirve de canal de comunicación entre ellos. Cuando hablamos de servicios Web, un endpoint se especifica mediante una URI (Universidad de Alicante, 2014)

A nivel técnico, los servicios pueden implementarse de varias formas. En este sentido, podemos distinguir dos tipos de servicios Web:

- **Servicios Web SOAP**

  Los servicios Web SOAP, o servicios Web "big", utilizan mensajes XML para intercomunicarse que siguen el estándar SOAP (Simple Object Access Protocol), un lenguaje XML que define la arquitectura y formato de los mensajes. Dichos sistemas normalmente contienen una descripción legible por la máquina de la descripción de las operaciones ofrecidas por el servicio, escrita en WSDL (Web Services Description Language), que es un lenguaje basado en XML para definir las interfaces sintácticamente (Universidad de Alicante, 2014).

- **Web RESTful**

  Los servicios Web RESTful (Representational State Transfer Web Services) son adecuados para escenarios de integración básicos ad-hoc. Dichos servicios Web se suelen integrar mejor con HTTP que los servicios basado en SOAP, ya que no requieren mensajes XML o definciones del servicio en forma de fichero WSDL

  Los servicios Web REST utilizan estándares muy conocidos como HTTP, SML, URI, MIME, y tienen una infraestructura "ligera" que permite que los servicios se construyan utilizando herramientas de forma mínima. Gracias a ello, el desarrollo de servicios RESTful es barato y tiene muy pocas "barreras" para su adopción (Universidad de Alicante, 2014).

2.7.3 Arquitectura de los Servicios Web

Según José Manuel Ortega Candel (Seguridad en Aplicaciones Web, pag. 106) indica que los servicios web presentan una arquitectura orientada a servicios que permite crear una definición abstracta de un servicio, proporcionar una implementación concreta de dicho servicio, publicar y localizar un servicio, seleccionar una instancia de un servicio, y utilizar dicho servicio con una elevada interoperabilidad. Es posible desacoplar la implementación del servicio Web y su uso por parte de un cliente. También es posible desacoplar la implementación del servicio y de cliente. Las implementaciones concretas del servicio pueden desacoplarse a nivel de lógica y transporte. La siguiente figura muestra el diagrama de una arquitectura orientada a servicios.

El proveedor del servicio define la descripción abstracta de dicho servicio utilizando un lenguaje de descripción de Servicios Web (WSDL: Web Services Description Language). A continuación, se crea un Servicio concreto a partir de la descripción abstracta del servicio, produciendo así una descripción concreta del servicio en WSDL. Dicha descripción concreta puede entonces publicarse en un servicio de registro como por ejemplo UDDI (Universal Description, Descovery and Integration). Un cliente de un servicio puede utilizar un servicio de registro para localizar una descripción de un servicio, a partir de la cual podrá seleccionar y utilizar una implementación concreta de dicho servicio (José Manuel Ortega Candel, 2018).

La descripción abstracta se define en un documento WSDL como un PortType. Una instancia concreta de un Servicio se define mediante un elemento port de un WSDL (consistente a su vez en una combinación de un PortType, un binding de codificación y

transporte, más una dirección). Un conjunto de ports definen un elemento service de un WSDL (José Manuel Ortega Candel, 2018).

2.8 Intranet

Según Wikipedia (https://es.wikipedia.org/wiki/Intranet, fecha 3 de abril de 2020) La palabra proviene de la combinación de los términos “intra” (dentro) y “net” de (network) haciendo referencia al concepto de red, así mismo puede definirse como una red privada que pertenece a una institución, organización o empresa que tiene como objetivo conectar, unir, compartir información entre los empleados, estudiantes o personas que forman parte de dicha organización sin tener que recurrir a un medio público en algunos casos inseguro.

Por lo general una intranet consiste en la interconexión de varias redes de área local (LAN) que se desarrolla a partir del protocolo TCP/IP. Gracias a este protocolo, los usuarios pueden acceder a la red de manera remota.

2.8.1 Características de una Intranet

- **Confidencialidad:** Garantizar que los datos no sean comunicados incorrectamente.
- **Integridad: Proteger los datos para evitar cambios no autorizados.**
- **Autentificación:** Tener confianza en la identidad de los usuarios.
- **Verificación:** Comprobar que los mecanismos de seguridad están correctamente implementados.
- **Disponibilidad:** Garantizar que los recursos estén disponibles cuando se necesiten.

2.8.2 Ventajas y Desventajas de una Intranet

Las ventajas de implementar una intranet son:

- Ofrece rapidez y eficiencia en la comunicación con los empleados: La realización de anuncios o comunicados directamente en la intranet es más rápida que otras vías de comunicación y queda archivada para posteriores consultas.
- Permite la conexión y acceso a bases de datos de otros departamentos, a las aplicaciones propias o provistas por terceros y a los recursos físicos.
- No tiene limitaciones horarias o de espacio. Está abierta permanentemente y accesible a todos los empleados incluidos los que están de viaje o fuera de las instalaciones centralizadas.

Entre las desventajas mencionamos que:

- El costo puede ser considerable durante la instalación inicial de la intranet su tamaño y complejidad de la misma determinarán la inversión.
- Existen riesgos de seguridad, las intranets aun con los niveles de seguridad óptimamente implementados tienen la desventaja de ser redes que están expuestas, en caso de que personal filtre o descubra contraseñas se corre el riesgo de perder valiosa información.
- La complejidad del sistema puede inhibir el uso de este. Los empleados pueden percibirlo como algo demasiado difícil de comprender y, por lo tanto, no lo usarán para obtener máxima eficacia.

2.9 Metodología Ágil

Según una publicación en el sitio Proyectos Agiles (https://proyectosagiles.org/desarrollo-iterativo-incremental, fecha julio de 2020) indica que las metodologías agiles son métodos que posee la ingeniería de software basados en el desarrollo iterativo e incremental, en donde los

requisitos y soluciones evolucionan mediante la colaboración de grupos auto organizados y multidisciplinarios.

Por lo tanto, existen muchos métodos de desarrollo ágil, la mayoría minimiza riesgos desarrollando software en lapsos cortos. El software desarrollo en una unidad de tiempos es llamado una iteración, la cual debe durar de una a cuatro semanas. Cada iteración del ciclo de vida incluye: planificación, análisis de requisitos, diseño, codificación, revisión y documentación. Una iteración no debe agregar demasiada funcionalidad para justificar el lanzamiento del producto al mercado, sino que la meta es tener una demo (sin errores) al final de cada iteración. Al final de cada iteración el equipo vuelve a evaluar las prioridades del proyecto.

El ambiente moderno de negocios que genera sistemas basados en computadora y productos de software evoluciona constantemente. La ingeniería de software ágil combina una filosofía con un conjunto de lineamientos de desarrollo.

La filosofía de la metodología ágil pone el énfasis en la satisfacción del cliente y la entrega rápida de software.

Diferencias entre las metodologías:

Metodologías tradicionales	Metodologías agiles
Están basadas en normas provenientes de estándares seguidos por el entorno de desarrollo	Basadas en heurísticas provenientes de prácticas de producción de código
Poseen cierta resistencia y son pocos flexibles a los cambios	Especialmente preparados para cambios durante el proyecto
Proceso mucho más controlado, con numerosas políticas/ normas	Proceso menos controlado, con pocos principios

El cliente interactúa con el equipo de desarrollo mediante reuniones de entrega	El cliente es parte activa en el proceso de desarrollo
Grupos grandes y posiblemente distribuidos donde a cada integrante se le asignaran tareas especificas	Grupos pequeños, con 10 integrantes o menos y trabajando en el mismo sitio en el cual tienen conocimiento sobre el proceso de desarrollo
La arquitectura del software es esencial y se expresa mediante modelos	Menos énfasis en la arquitectura del software
Existe un contrato prefijado	No existe contrato tradicional o al menos es bastante flexible

Tabla 2.1. Diferencias entre las metodologías:

Fuente: Elaboración propia

2.10 Metodología Scrum

Scrum es una metodología ágil de desarrollo de proyectos que toma su nombre a principios de los estudios realizados sobre nuevas prácticas de producción por Hirotaka Takeuchi e Ikujijo Nonaka a mediados de los 80. Aunque surgió como modelo para el desarrollo de productos tecnológicos, también se emplea en entornos que trabajan con requisitos inestables y que requieren rapidez y flexibilidad; situaciones frecuentes en el desarrollo de determinados sistemas de software.

Jeff Sutherland aplicó el modelo Scrum al desarrollo de software en 1993 en EaselCorporation (Empresa que en los macro-juegos de compras y fusiones se integraría en VMARK, luego en Informix y finalmente en Ascential Software Corporation). En 1996 lo presentó junto con Ken Schwaber como proceso formal, también para gestión del desarrollo de software en OOPSLA 96. Más tarde, en 2001 serían dos de los promulgadores del Manifiesto Ágil. En el desarrollo de software Scrum está considerado como modelo ágil por la Agile Alliance.

Por lo tanto, es una metodología de desarrollo muy simple, que requiere trabajo duro porque no se basa en el seguimiento de un plan, sino en la adaptación continua a las circunstancias de la evolución del proyecto.

de desarrollo de aplicaciones y su posterior mantenimiento.

2.10.1 Ventajas de Scrum

Dentro de las ventajas que ofrece la metodología Scrum, tenemos:

- Adecuado manejo de los requerimientos cambiantes.
- Incentiva la motivación del equipo de desarrollo.
- El cliente se haya involucrado con el proyecto en un mayor grado.
- Entrega de un producto funcional al finalizar cada Sprint.
- Visualización del proyecto día a día.
- Permite superar satisfactoriamente los fallos presentados durante el tiempo de vida del proyecto.

2.10.2 Desventajas de Scrum

Cómo cualquier metodología de desarrollo, presenta algunas restricciones durante su aplicación:

- No genera toda la evidencia o documentación de otras metodologías.
- No es apto para todos los proyectos, sobre todo para aquellos en los que intervienen equipos dispersos.
- Es probable que sea necesario complementarlo con otras metodologías como XP.

2.10.3 Diferencias del desarrollo tradicional vs Scrum

En la figura N° 2.6 se aprecia los principales contrastes que diferencian el desarrollo tradicional del desarrollo ágil Scrum.

Figura 2.6. Desarrollo tradicional vs. Desarrollo ágil

Fuente: http://www.navegapolis.net/files/s/NST-010_01.pdf

De la figura de la página anterior, en cuánto al desarrollo ágil no lo realizan equipos diferentes especializados. Es un equipo único, formado por personas muy competentes, con perfiles y conocimientos que cubren las disciplinas necesarias para llevar a cabo el trabajo.

Respecto a las fases, se destaca que en el modelo Scrum no hay fases. En realidad, las fases pasan a ser tareas que se ejecutan cuando se necesitan. No se hace primero el diseño del concepto o los requisitos, más tarde el análisis, luego el desarrollo, etc. Lo que aplicado al software serían las fases de: Requisitos del sistema, requisitos del software, análisis, diseño, construcción, pruebas e integración; y se ejecutarían de forma secuencial, pasan a tareas que se llevan a cabo en el momento que hacen falta. Normalmente a lo largo de pequeñas iteraciones durante todo el desarrollo.

2.10.4 Objetivos y preferencias de la gestión ágil

La gestión ágil de proyectos tiene como objetivo dar garantías a las demandas principales de la industria actual, entre los que tenemos:

- **Valor**

  La gestión ágil se necesita en los mercados rápidos. Su objetivo es dar el mayor valor posible al producto, cuando éste se basa en: Innovación y flexibilidad.

  La innovación, la permanencia de estas empresas depende de su capacidad de innovación continua. Del lanzamiento continuo de novedades, que compiten con los productos de otras empresas que también están en continua innovación.

  La flexibilidad, el producto no sólo es valioso por su valor en el momento de su lanzamiento, sino también por su capacidad de adaptación y evolución a través de actualizaciones y ampliaciones.

- **Reducción del tiempo de salida al mercado**

  En la década de los 90, el tiempo medio de salida al mercado de los nuevos productos en EE.UU. se redujo de 35,5 a 11 meses (Wujec & Muscat, 2002). Este tiempo es un factor competitivo clave en determinados sectores.

  Las estrategias de la gestión ágil para producir resultados en menos tiempo que la gestión tradicional son:

  a) Solapamiento de las fases de desarrollo
  b) Entrega temprana de las primeras partes del producto, que corresponden con las de mayor urgencia para el cliente, de forma que puede lanzar la primera versión en el menor tiempo posible.

- **Agilidad**

Capacidad para producir partes completas del producto en periodos breves de tiempo.

- **Resultados fiables**

El objetivo de la gestión predictiva es ejecutar el trabajo planificado (y conocido de antemano) en el plazo planificado y por el coste previsto.

La gestión ágil no tiene un carácter predictivo o de anticipación. No conoce de antemano el detalle del producto que va a desarrollar, y por eso su objetivo no es fiabilidad en el cumplimiento de los planes, sino en el valor del resultado.

Los procesos de la gestión tradicional son buenos cuando consiguen desarrollar de forma repetible los productos especificados en el tiempo y con los costes previstos.

Los procesos de la gestión ágil son buenos, cuando consiguen entregar de forma temprana y continua un valor innovador.

Las preferencias de la gestión ágil, a diferencia de la tradicional, se muestran en el manifiesto ágil:

a) La capacidad de respuesta al cambio, sobre el seguimiento de un plan.
b) Los productos que funcionan frente a especificaciones y documentaciones innecesarias.
c) La colaboración con el cliente frente a la negociación contractual.
d) A las personas y su interacción por encima de los procesos y las herramientas.

2.10.5 Visión general del proceso

Scrum denomina “Sprint” a cada iteración de desarrollo y recomienda realizarlas con duraciones de 30 días.

En la figura Nº 2.7 se observa una visión general del proceso de desarrollo de software, siguiendo la Metodología Scrum.

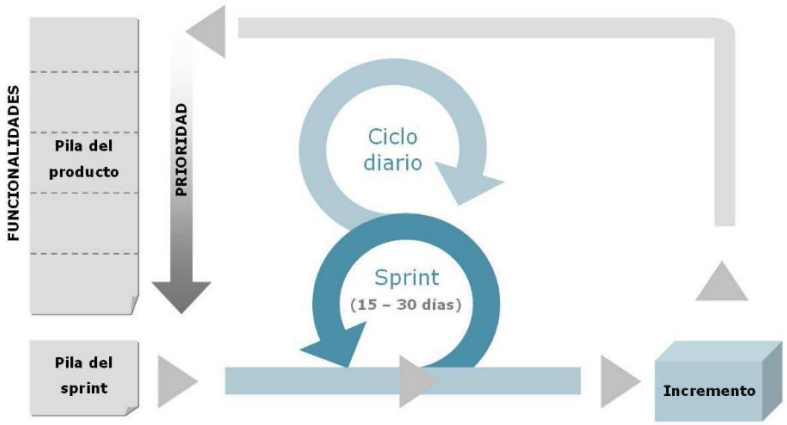


Figura 2.7. Visión general del proceso

Fuente: http://www.navegapolis.net/files/s/NST-010_01.pdf

Del gráfico, es importante considerar que el Sprint es por tanto el núcleo central que proporciona la base de desarrollo iterativo e incremental. Los elementos que conforman el desarrollo Scrum son:

- **Las reuniones**
  a) **Planificación de Sprint:** Jornada de trabajo previa al inicio de cada Sprint en la que se determina cuál va a ser el trabajo y los objetivos que se deben cumplir en esa iteración.
  b) **Reunión diaria:** Breve revisión del equipo del trabajo realizado hasta la fecha y la previsión para el día siguiente.

c) **Revisión de Sprint:** Análisis y revisión del incremento generado.

- **Los artefactos**

Scrum define una pequeña cantidad de artefactos para el seguimiento del proyecto y control de las actividades asociadas al Sprint.

a) Product backlog (Pila del producto)
b) Sprint backlog (Pila del Sprint)
c) Gráfica de progreso

- **Los roles**

Scrum clasifica a todas las personas que intervienen o tienen interés en el desarrollo del proyecto en: Dueño del Producto, Equipo Scrum, Scrum Manager o Scrum Master y "otros interesados".

a) **Equipo Scrum (Scrum Team):** Definido como miembros del equipo de proyecto; se halla conformado por las personas dedicadas al desarrollo y diseño de la aplicación. Normalmente está constituido por un grupo de 3 a 9 individuos.
b) **Dueño del Producto (Product Owner):** Cumple los roles de cliente y de usuario. La persona que asume este rol puede ser tanto un cliente como un miembro del equipo especializado en la lógica del negocio y procesos que se desarrollan; se encarga de proporcionar al equipo Scrum con conocimientos relacionados al negocio. Asimismo, administra los backlogs de los productos, que son un listado de requerimientos pendientes en donde todas las especificaciones del producto están enumeradas, los cuales están a la vista de todos los miembros de la organización.

c) **Scrum Master:** Definido como Director del Proyecto; esta persona tiene diariamente breves reuniones, denominados Scrums Diarios, con el equipo. Asimismo, es el encargado de canalizar la información referente a nuevos requerimientos o modificaciones con la finalidad de disminuir el número de interrupciones a los desarrolladores.

- **Valores**

Scrum es una "carrocería" para dar forma a los principios ágiles. Es una ayuda para organizar a las personas y el flujo de trabajo; como lo pueden ser otras propuestas de formas de trabajo ágil

La carrocería sin motor, sin los valores que dan sentido al desarrollo ágil, no funciona.

a) Delegación de atribuciones (empowerment) al equipo para que pueda auto-organizarse y tomar las decisiones sobre el desarrollo.
b) Respeto entre las personas. Los miembros del equipo deben confiar entre ellos y respetar sus conocimientos y capacidades.
c) Responsabilidad y auto-disciplina (no disciplina impuesta).
d) Trabajo centrado en el desarrollo de lo comprometido
e) Información, transparencia y visibilidad del desarrollo del proyecto.

2.10.6 Modelo aplicativo

En la figura Nº 2.8 permite visualizar el modelo aplicativo Scrum, que presenta la secuencia de pasos metodológicos que se deben seguir para poder aplicar la Metodología Scrum.

Figura 2.8. Modelo aplicativo

Fuente: Rodríguez González, Pilar (2008). Estudio de la Aplicación de Metodologías Ágiles para la Evolución de productos software. Madrid, España. Pág. 17

De la figura anterior, se observa que el Modelo Scrum está conformado por las siguientes fases:

- **FASE N° 1: Definir backlog del producto**
- **FASE N° 2: Planificación del backlog**
- **FASE N° 3: Scrum diario**
- **FASE N° 4: Revisión del Sprint**
- **FASE N° 5: Retrospectiva del Sprint**

Independientemente del tipo de metodología que se utilice, cualquier desarrollo de software parte siempre de un mismo problema: conocer las necesidades de los clientes. Scrum, al igual que el resto de metodologías ágiles, pretende no centrar las tareas de desarrollo en un

conjunto de requisitos formalmente definidos, sino que aboga por la incorporación del cliente como un miembro más del equipo de desarrollo. De este modo, no se considera el proceso de definición de requisitos como un fin dentro del desarrollo del proyecto, sino que los requisitos aparecen implícitamente dentro del contenido de las denominadas historias de usuario. A continuación, se procede a detallar cada fase del modelo aplicativo Scrum:

2.10.7 FASE N° 1: Definir backlog del producto

En esta primera fase, punto 1 de la figura N° 2.8, antes de comenzar el primer Sprint, es necesaria la elaboración del Backlog del Producto o Pila del Producto.

La pila de producto es el corazón de Scrum, es donde empieza todo. Básicamente es una lista priorizada de requisitos, historias o funcionalidades que el cliente desea, descritas en terminología del cliente, Se llama a esto historias de usuario, o a veces simplemente elementos de la pila que, por lo general, incluyen los siguientes campos:

- **ID**

  Un identificador único, simplemente un número auto-incremental. Esto permite no perder la pista a las historias cuando se cambia su nombre.

- **Nombre**

  Una descripción corta de la historia. Por ejemplo, “Ver tu historial de transacciones”. Suficientemente claro como para que el Dueño de Producto comprenda aproximadamente de qué se está hablando, y suficientemente clara como para que se distinga de las otras historias. Normalmente, 2 a 10 palabras.

- **Importancia**

  El ratio de importancia que el Dueño de Producto da a esta historia. Por ejemplo, 10. O 150. Más alto = más importante. Es preferible evitar el término “prioridad” porque típicamente “1” se considera la “máxima prioridad, lo que es muy incómodo si posteriormente se decide que algo es más importante. ¿Qué prioridad se le daría a ese nuevo elemento? ¿Prioridad 0? ¿Prioridad -1?

- **Estimación inicial**

  La valoración inicial del Equipo acerca de cuanto trabajo es necesario para implementar la historia, comparada con otras historias. Las unidades son “puntos de historia” y usualmente corresponde a “días-persona ideales”. Se pregunta al equipo: “si tuvieras el número óptimo de personas para esta historia (ni muchos ni pocos, típicamente 2) y los encerraras en una habitación con cantidad de comida, y trabajan sin distracciones, ¿en cuántos días saldrían con una implementación terminada, demostrable, testeada y liberable?”. Si la respuesta es “con 3 personas encerrados en una habitación nos llevaría 4 días”, entonces la estimación inicial son 12 puntos (3x4). Lo importante no es que las estimaciones absolutas sean correctas (es decir, que una historia de 2 puntos deba durar 2 días), lo importante es que las estimaciones relativas sean correctas (es decir, que una historia de 2 puntos debería durar la mitad que una historia de 4 puntos).

- **Como probarlo**

  Una descripción a alto nivel de cómo se demostrará esta historia en la demo al final del Sprint. Se trata, esencialmente, de una simple especificación de un test: “Haz esto, entonces haz lo otro, y entonces debería ocurrir aquello”.

- **Notas**

  Cualquier otra información, clarificación, referencia a otras fuentes de información, etc. Normalmente muy breve.

La figura N° 2.9 muestra un ejemplo de pila de producto del lado del dueño del producto.

Pila de Producto (ejemplo)					
ID	**Nombre**	**Imp.**	**Est.**	**Como probarlo**	**Notas**
1	Depósito	30	5	Entrar, abrir página de depósito, depositar 10€, ir a página de balance y comprobar que se ha incrementado en 10€	Necesita un diagrama UML. No preocuparse por encriptación aun
2	Ver tu historial de transacciones	10	8	Entrar, ver transacciones. Realizar un depósito de 10€. Ir a transacciones y comprobar que se ha actualizado con el nuevo depósito	Utilizar paginación para no hacer consultas muy grandes a la BB.DD. Diseño similar a la página de usuario.

Figura 2.9. Ejemplo de Product Backlog o Pila de Producto

Fuente: Kniberg, H. (2007). Scrum and XP from the Trenches: How we do scrum. Estados Unidos: Editorial C4Media. Page. 18.

Estos seis campos son los únicos que se utilizan Sprint tras Sprint. Se mantiene esta tabla en un documento con la propiedad compartir habilitado (es decir, muchos usuarios pueden editar simultáneamente la hoja). Oficialmente, el dueño de producto es el propietario del documento, pero no se quiere dejar al resto de usuarios fuera. Muchas veces un desarrollador necesita abrir el documento para clarificar algo o cambiar una estimación. Por la misma razón, no se coloca este documento en el repositorio de control de versiones; en vez de eso, se almacena en una unidad de red compartida.

2.10.8 FASE N° 2: Planificación del backlog

La planificación de Sprint es una reunión crítica, probablemente la más importante de Scrum. Una planificación de Sprint mal ejecutada puede arruinar por completo todo el Sprint. El propósito de la planificación de Sprint es proporcionar al equipo suficiente información como para que puedan trabajar en paz y sin interrupciones durante unas pocas semanas, y para ofrecer al dueño de producto suficiente confianza como para permitírselo.

Una planificación de Sprint produce, concretamente:

- Una meta de Sprint
- Una lista de miembros (y su nivel de dedicación, si no es del 100%)
- Una Pila de Sprint o Sprint Backlog (lista de historias incluidas en el Sprint)
- Una fecha concreta para la Demo del Sprint
- Un lugar y momento definidos para el Scrum Diario

Es importante que el dueño de producto asista a la planificación del Sprint, se sabe que a veces los dueños de producto se resisten a pasar horas con el equipo preparando la planificación de Sprint.

2.10.9 FASE N° 3: Scrum diario

En esta tercera fase, punto 3 de la figura N° 2.8, se deberá definir el sitio y la hora para el Scrum Diario.

Uno de los productos frecuentemente olvidados de la planificación de Sprint es un sitio y una hora determinados para el Scrum Diario. Sin ello, tu Sprint está condenado a un mal

comienzo. El primer Scrum diario es esencialmente el lanzamiento, donde todo el mundo decide por dónde va a empezar a trabajar.

- Desventaja de Scrum por las tardes, cuando se llega al trabajo por la mañana, se tiene que acordar de qué se le dijo a la gente sobre lo que deberían hacer hoy.
- Desventaja de los Scrum por las mañanas, cuando se llegas al trabajo por la mañana, se debe acordar de qué se hizo ayer para informar sobre ello hoy.

El procedimiento por defecto es seleccionar la hora más temprana a la que ningún miembro del equipo vaya a quejarse. Usualmente las 9:00, 9:30 o 10:00a.m. Lo más importante es que sea a una hora a la que todo el equipo acepte con total convencimiento.

Si el tiempo se está agotando. De todos los asuntos que queremos resolver durante la planificación de Sprint, ¿qué abandonar si se queda sin tiempo? Bueno, normalmente se usa la siguiente lista de prioridades:

Prioridad 1: Una meta de Sprint y una fecha para la demo. Esto es lo mínimo que se necesita para comenzar un Sprint. El equipo tiene una meta y una fecha de finalización, y puede trabajar directamente con la pila de producto. Se debe considerar seriamente organizar una nueva reunión de planificación de Sprint mañana mismo, pero si realmente se necesita que el Sprint comience entonces probablemente se pueda hacer con esto.

Prioridad 2: Lista de qué historias ha aceptado terminar el equipo en este Sprint.

Prioridad 3: Una estimación para cada historia del Sprint.

Prioridad 4: “Cómo probarlo”, relleno para cada historia del Sprint.

Prioridad 5: Cálculos de velocidad y recursos, como chequeo de la planificación del Sprint. Incluyendo una lista de los miembros del equipo y sus compromisos (de otra forma, no se podría calcular la velocidad).

Prioridad 6: Un sitio y hora específicos para la realización del Scrum diario. Sólo se necesita un momento para decidirlo, pero si se queda sin tiempo el Scrum Master puede simplemente decidir esto después de la reunión y mandar un correo a todo el mundo.

Prioridad 7: Historias divididas en tareas. Esta división puede sin embargo hacerse diariamente durante los Scrum diarios, pero interferirá levemente el flujo del Sprint.

2.10.10 FASE N° 4: Revisión del Sprint

En esta cuarta fase, punto 4 de la figura N° 2.8, se habla acerca de la Revisión del Sprint o también conocido como Demo de Sprint, que es una parte importante de Scrum que la gente tiende a subestimar.

La revisión del Sprint, involucra la presencia del equipo Scrum, Scrum Master, Product Owner con todas las personas implicadas en el proyecto. La duración máxima de esta reunión es de cuatro horas, y el objetivo es presentar al propietario del producto y a las nuevas funcionalidades implementadas.

Una demo de Sprint bien ejecutada, aunque parezca poco espectacular, tiene un efecto muy profundo:

- El equipo obtiene reconocimiento por sus logros. Se sienten bien.
- Otras personas se enteran de lo que está haciendo el equipo.
- La demo consigue feedback vital de los interesados.

- Las demos son (o deberían ser) un evento social donde diferentes equipos pueden interactuar unos con otros y discutir su trabajo. Esto es muy valioso.
- Hacer una demo fuerza al equipo a acabar realmente las cosas y entregarlas (incluso aunque sea sólo en entorno de pruebas). Sin las demos, se sigue consiguiendo enormes montones de cosas terminadas al 99%. Con las demos puede que se consigan menos cosas terminadas, pero estas están realmente terminadas, lo que es mucho mejor que tener una enorme pila de cosas que están más o menos listas y que se pulirán en el próximo Sprint.
- Si un equipo se ve obligado a hacer una demo de Sprint, incluso aunque no tengan mucho que realmente esté funcionando, la demo será embarazosa. El equipo tartamudeará y tropezará mientras hace la demo y el aplauso de después será frío. La gente sentirá un poco de pena por el equipo, algunos se enfadarán con ellos por hacerles perder el tiempo con una demo pésima. Esto duele, pero el efecto es similar al de una amarga medicina. En el próximo Sprint, el equipo intentará por todos los medios tener cosas terminadas. El equipo sabe que tendrán que hacer una demo pase lo que pase, lo que incrementa significativamente las posibilidades de que haya algo útil que demostrar.

Lista de comprobación para demos de Sprint

- Se debe asegurar de presentar claramente el objetivo del Sprint. Si hay personas en la demo que no saben nada sobre tu producto, tómate un par de minutos para describirlo.
- No se debe perder mucho tiempo preparando la demo, especialmente en llamativas presentaciones. Se debe concentrar en mostrar código funcionando.

- Mantener el paso rápido, es decir, se debe concentrar la preparación en hacer que la demo sea rápida en lugar de bonita.
- Mantener la demo a nivel de negocio, dejar los detalles técnicos aparte. Se debe concentrar en "qué se ha hecho" en lugar de "cómo se ha hecho".
- En la medida de lo posible, se debe dejar que la audiencia pruebe el producto por sí misma.
- No se debe mostrar un montón de pequeños errores solucionados y funcionalidades triviales. Se puede mencionar, pero no mostrar, ya que normalmente se tarda mucho y desvía la atención de las historias más importantes.

Al final de la reunión se interroga individualmente a todos los asistentes para recabar impresiones, sugerencias de cambio y mejora, y su relevancia. Y el propietario del producto trata con los asistentes y con el equipo las posibles modificaciones en el product backlog.

2.10.11 FASE N° 5: Retrospectiva del Sprint

En esta quinta fase, punto 5 de la figura N° 2.8, lo más importante de una Retrospectiva de Sprint es asegurarse de que tienen lugar.

Por alguna razón, los equipos no siempre parecen inclinados a hacer retrospectivas. Sin un empujón, la mayoría de los equipos usualmente se saltarían la retrospectiva y continuarían con el próximo Sprint. Pero, todo el mundo coincide en que las retrospectivas son extremadamente útiles. De hecho, se puede afirmar que la retrospectiva es el segundo evento más importante de Scrum (siendo el primero la reunión de planificación de Sprint), ya que es la mejor oportunidad para mejorar.

Por supuesto, no se necesita una reunión de retrospectiva para conseguir buenas ideas, pero si la idea viene del equipo, es decir, surge durante la retrospectiva, entonces a todo el mundo se le permite contribuir y discutir las ideas. Sin las retrospectivas se notará que el equipo sigue cometiendo los mismos errores una y otra vez.

Para organizar las retrospectivas, el formato general varía un poco, pero normalmente se hace algo como esto:

- Se reserva 1-3 horas, dependiendo de cuánta discusión esperemos.
- Participantes: el Dueño de Producto, el Equipo Scrum y el Scrum Master.
- Se conducen a una reunión cerrada, un rincón cómodo con sofás, el patio del tejado o algún sitio similar. Que se pueda tener una discusión sin interrupciones.
- Normalmente no se hace retrospectivas en la sala del equipo, ya que la atención de la gente suele diluirse.
- Alguien es designado secretario.
- El Scrum Master muestra la pila de Sprint y, con ayuda del equipo, resume el Sprint. Eventos importantes, decisiones, etc.
- Se hace “la ronda”. Cada persona tiene una oportunidad de decir, sin ser interrumpida, qué piensan que ha ido bien, que podría haber ido mejor y que piensan que debería hacerse de forma diferente en el próximo Sprint.
- Se observa la velocidad estimada frente a la real. Si hay una gran diferencia, intentamos analizar por qué.
- Cuando el tiempo casi se ha acabado, el Scrum Master trata de resumir las sugerencias concretas sobre qué puede hacerse mejor en el próximo Sprint.

Las retrospectivas generalmente no están muy estructuradas. No obstante, el tema subyacente es siempre el mismo: “qué se puede hacer mejor el próximo Sprint”.

En la figura N° 2.10, se tiene un ejemplo de pizarra de una retrospectiva reciente.

Del gráfico de la página siguiente, se observan tres columnas de mejoras:

- Bien: si se hiciera un Sprint otra vez, se volvería a hacer estas cosas igual.
- Mejorable: si se hiciera otra vez el Sprint, haríamos estas cosas de forma diferente.
- Mejoras: ideas concretas sobre cómo podemos mejorar en el futuro.

Figura 2.10. Ejemplo de pizarra de una reunión de retrospectiva Scrum

Fuente: Kniberg, H. (2007). Scrum and XP from the Trenches: How we do scrum. Estados Unidos: Editorial C4Media. Page. 70.

Así que las columnas 1 y 2 son una mirada al pasado, mientras que la columna 3 mira al futuro. Después de que el equipo genere todas estas ideas en post-its, utilizan “votación por puntos” para determinar en qué mejoras centrarse el próximo Sprint. Cada miembro del equipo tiene tres imanes y se les invita a votar sobre cualquier mejora en la que les gustaría trabajar en el próximo Sprint. Cada miembro del equipo distribuye los imanes como quiera,

incluso colocando los tres en el mismo elemento. Basándose en esto, se selecciona cinco mejoras de procesos en los que concentrarse, y se evalúa en la siguiente retrospectiva.

Es importante no ser demasiado ambicioso. Se debe concentrar en unas pocas mejoras en cada Sprint.

2.11 Arquitectura de software

Para Kazman (1996), la arquitectura de software alude a "la estructura general de este y a las formas en las que esta da la integridad conceptual a un sistema" En su forma más sencilla, la arquitectura es la estructura de organización de los componentes de un programa (módulos), la forma en la que estos interactúan y la estructura de datos que utilizan. Sin embargo, en un sentido más amplio, los componentes se generalizan para que representen los elementos de un sistema grande y sus interacciones.

Shaw y Garlan describen un conjunto de propiedades que deben especificarse como parte del diseño de la arquitectura:

- Propiedades estructurales. Este aspecto de la representación del diseño arquitectónico define los componentes de un sistema (módulos, objetos, filtros, etc.) y la manera en la que están agrupados e interactúan unos con otros. Por ejemplo, los objetos se agrupan para que encapsulen tanto datos como el procedimiento que los manipula e interactúen invocando métodos.
- Propiedades extra funcionales. La descripción del diseño arquitectónico debe abordar la forma en la que la arquitectura del diseño satisface los requerimientos de desempeño, capacidad, confiabilidad, seguridad y adaptabilidad, así como otras características del sistema.

- Familias de sistemas relacionados. El diseño arquitectónico debe basarse en patrones repetibles que es común encontrar en el diseño de familias de sistemas similares. En esencia el diseño debe tener la capacidad de reutilizar bloques de construcción arquitectónica.

2.12 Arquitectura Model View View Model

La arquitectura MVVM es una arquitectura Model-View-ViewModel que elimina el acoplamiento estrecho entre cada componente. Lo más importante, en esta arquitectura, los niños no tienen la referencia directa al padre, solo tienen la referencia por observables (Microsoft, https://docs.microsoft.com/en-us/archive/msdn-magazine/2010/july/design-patterns-problems-and-solutions-with-model-view-viewmodel, fecha julio de 2010)

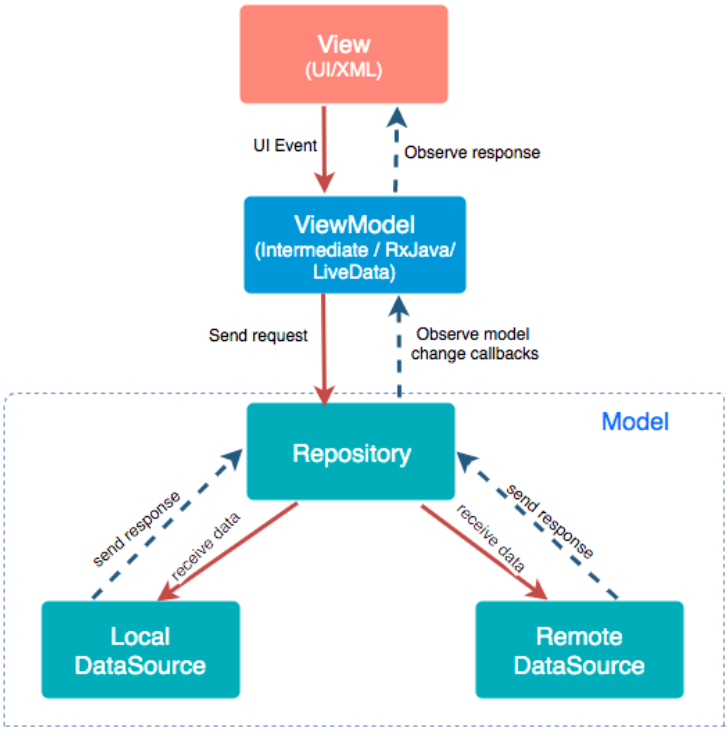


Figura 2.11. La arquitectura MVVM

Fuente: https://docs.microsoft.com/en-us/archive/msdn-magazine/2010/july/design-patterns-problems-and-solutions-with-model-view-viewmode.

- **Modelo**: Representa los datos y la lógica empresarial de la aplicación de Android. Consiste en la lógica de negocios: fuente de datos local y remota, clases de modelos, repositorio.
- **Vista:** Consiste en el código de la interfaz de usuario (actividad, fragmento), XML. Envía la acción del usuario al ViewModel pero no obtiene la respuesta directamente. Para obtener la respuesta, tiene que suscribirse a los observables que ViewModel le expone.
- **ViewModel:** es un puente entre la Vista y el Modelo (lógica de negocios). No tiene ni idea de qué Vista tiene que usar, ya que no tiene una referencia directa a la Vista. Básicamente, el ViewModel no debe ser consciente de la vista con la que está interactuando. Interactúa con el Modelo y expone lo observable que la Vista puede observar.

Esto se trata de MVVM, ahora pasemos a la parte de implementación.

- **Flexibilidad de desarrollo.** Este enfoque mejora la conveniencia del trabajo en equipo, porque mientras un miembro del equipo trabaja con el diseño y la estilización de la pantalla, el otro, al mismo tiempo, describe la lógica de la adquisición y el procesamiento de datos;
- **Pruebas.** Esta estructura simplifica la escritura de prueba y el proceso de creación de objetos simulados. Además, en la mayoría de los casos elimina la necesidad de una prueba de IU automatizada ya que puede envolver ViewModel con pruebas unitarias;
- **Separación lógica.** Debido a la mayor diferenciación, el código se vuelve más flexible y fácil de soportar, sin mencionar su legibilidad. Cada módulo es responsable solo de una función específica.

2.13 Patrón modelo vista controlador

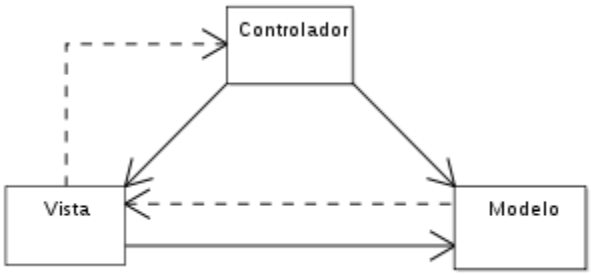


Figura 2.12. Patrón modelo vista controlador

Fuente: https://es.wikipedia.org/wiki/Modelo-vista-controlado.

Modelo Vista Controlador (MVC) es un patrón de arquitectura de software, que separa los datos y principalmente lo que es la lógica de negocio de una aplicación de su representación y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado, define componentes para la representación de la información, y por otro lado para la interacción del usuario. Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento. (Wikipedia, https://es.wikipedia.org/wiki/Modelo-vista-controlador, fecha Julio de 2020)

2.13.1 Descripción del patrón

De manera genérica, los componentes de MVC se podrían definir como sigue:

- **El Modelo:** Es la representación de la información con la cual el sistema opera, por lo tanto, gestiona todos los accesos a dicha información, tantas consultas como actualizaciones, implementando también los privilegios de acceso que se hayan

descrito en las especificaciones de la aplicación (lógica de negocio). Envía a la vista aquella parte de la información que en cada momento se le solicita para que sea mostrada (típicamente a un usuario). Las peticiones de acceso o manipulación de información llegan al modelo a través del controlador.

- **El Controlador:** Responde a eventos (usualmente acciones del usuario) e invoca peticiones al modelo cuando se hace alguna solicitud sobre la información (por ejemplo, editar un documento o un registro en una base de datos). También puede enviar comandos a su 'vista' asociada si se solicita un cambio en la forma en que se presenta el modelo (por ejemplo, desplazamiento o scroll por un documento o por los diferentes registros de una base de datos), por tanto, se podría decir que el controlador hace de intermediario entre la vista y el modelo.
- **La Vista:** Presenta el modelo (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario), por tanto, requiere de dicho modelo la información que debe representar como salida.

2.14 Herramientas de desarrollo

Una herramienta de desarrollo de software es una aplicación informática que usa un programador para crear, depurar, gestionar o mantener un programa. Las herramientas que se utilizarán en el proyecto son:

Herramienta	C#
Definición	C# es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET.
Características	Este lenguaje ha sido influido principalmente por Java, C++, Eiffel, Modula-3 y Pascal, y al mismo tiempo también ha influido a lenguajes como D, F#, Java 5 o Vala.
Ventajas	• Las principales ventajas que presenta el uso C# en comparación con otros lenguajes es su potencia como lenguaje, pero también su flexibilidad. Soporta la mayoría de paradigmas, destacando el paradigma funcional que combinado con el paradigma orientado a objetos hacen del lenguaje uno de los más potentes. • Este lenguaje es muy parecido a Java, y deriva de otros más populares como el C o el C++.

Tabla 2.2. Herramientas de desarrollo lenguaje de programación C#

Fuente: Elaboración propia

Herramienta	Java
Definición	Java es un lenguaje de programación con el que podemos realizar cualquier tipo de programa. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de Internet como en la informática en general
Características	La principal característica de Java es que es independiente de la plataforma (multiplataforma). Esto significa que cuando estás programando en Java. Puedes ejecutar EL MISMO programa en un PC con Windows, otro con Linux, en un Servidor SUN con sistema operativo Solaris, o en un teléfono móvil de última generación.
Ventajas	• El lenguaje Java es orientado a objetos. El paradigma de programación orientada a objetos supuso un gran avance en el desarrollo de aplicaciones, ya que es capaz de acercar la forma de programar a la forma de pensar del ser humano. • En java no existen problemas con la liberacion de memoria en el sistema: En Java decidieron romper con el sistema tradicional de liberación de memoria, haciendo que el programador ya no fuese el responsable de esa tarea. Así, lo único que necesita hacer el programador es solicitar la memoria al sistema.

Tabla 2.3. Herramientas de desarrollo lenguaje de programación Java

Fuente: Elaboración propia

Herramienta	Android Studio
Definición	Android Studio es el entorno de desarrollo integrado oficial para la plataforma Android. Fue anunciado el 16 de mayo de 2013 en la conferencia Google I/O, y reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android. La primera versión estable fue publicada en diciembre de 2014.
Características	**Capacidad de ejecución**. Android Studio está capacitado para ejecutar y depurar el código de la aplicación sin que sea necesario tener que reiniciar la aplicación o reconstruir el archivo de instalación APK. De esta forma se le facilita al desarrollador ver inmediatamente los cambios realizados. **Editor inteligente de código.** Aumenta la productividad de los desarrolladores con funciones tales como el análisis de programación o la refactorización, que inciden en mejorar la calidad del código. **Emulador de funciones**. Android Studio permite al desarrollador que pruebe su aplicación de forma virtual, sin la necesidad de un dispositivo móvil real, con las debidas configuraciones para teléfonos inteligentes, o tabletas, o dispositivos con Android TV o relojes inteligentes.
Ventajas	• Ejecuta las compilaciones de forma muy rápida.

	• Renderizado de layouts en tiempo real, tiene la posibilidad de usar parámetros. • Ejecución en tiempo real de la aplicación y desde el móvil, gracias al emulador. • Tiene la capacidad de asociar archivos y carpetas de forma automática en la aplicación, la eliminación de archivos y la creación de carpetas en valores. • Puede desarrollar cualquier IDE. • No soporta desarrollo para NDK, pero a través de intelliJ sí lo soporta con el plugin

Tabla 2.4. Herramientas de desarrollo IDE Android Studio

Fuente: Elaboración propia

Herramienta	Visual Studio
Definición	Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows. Soporta varios lenguajes de programación, tales como Visual C++, Visual C#, Visual J#, y Visual Basic .NET.
Características	Entre las características que podremos agregar en Visual Studio, encontraremos desde herramientas de Debug hasta opciones para actualización en tiempo real de nuestro código en la vista del navegador y compilación en vivo de los lenguajes que lo requieran (por ejemplo, en el caso de SASS a CSS).

	La mejora en las capacidades de Pruebas Unitarias permite ejecutarlas más rápido independientemente de si lo hacen en el entorno IDE o desde la línea de comandos. Se incluye además un nuevo soporte para diagnosticar y optimizar el sistema a través de las herramientas de pruebas de Visual Studio. Con ellas se podrán ejecutar perfiles durante las pruebas para que ejecuten cargas, prueben procedimientos contra un sistema y registren su comportamiento; y utilizar herramientas integradas para depurar y optimizar.
Ventajas	• Un aspecto fundamental de un editor de código es que podamos utilizarlo con los lenguajes de programación que trabajamos a diario. Visual Studio Code es una herramienta que tiene soporte nativo para gran variedad de lenguajes, entre ellos podemos destacar los principales del desarrollo Web: HTML, CSS, y JavaScript, entre otros. • Otra ventaja interesante es la posibilidad de configurar la vista a nuestro gusto. De esta forma, podremos tener más de un código visible al mismo tiempo, las carpetas de nuestro proyecto y también acceso a la terminal o un detalle de problemas, entre otras posibilidades.

Tabla 2.5. Herramientas de desarrollo IDE Visual Studio

Fuente: Elaboración propia

Herramienta	Microsoft SQL Server
Definición	Microsoft SQL Server es un sistema de gestión de base de datos relacional, desarrollado por la empresa Microsoft. El lenguaje de desarrollo utilizado (por línea de comandos o mediante la interfaz gráfica de Management Studio) es Transact-SQL (TSQL), una implementación del estándar ANSI del lenguaje SQL, utilizado para manipular y recuperar datos (DML), crear tablas y definir relaciones entre ellas (DDL).
Características	• Soporte de transacciones. • Soporta procedimientos almacenados. • Incluye también un entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente. • Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información. • Además, permite administrar información de otros servidores de datos.
Ventajas	• Puede ser útil para manejar y/o obtener datos de la red de redes. • Ofrece una potente forma de unir SQL e Internet. • Utiliza una extensión al SQL estandar, que se denomina Transact SQL.

	• El Transact SQL, soporta la definición, modificación y eliminación de bases de datos, tablas, atributos, índices, etc., es decir, el lenguaje de definición de datos (LDD), así como la consulta, actualización y borrado de tuplas de tablas, es decir, el lenguaje de manipulación de datos (LMD).

Tabla 2.6. Herramientas de desarrollo gestor de base de datos Microsoft SQL Server

Fuente: Elaboración propia

2.15 Calidad de software – ISO 9125

ISO 9126 es un estándar internacional para la evaluación del Software. Esta dividió en cuatro partes las cuales dirigen, respectivamente el modelo de calidad, las métricas externas, métricas internas y la calidad en las métricas de uso. Está siendo reemplazado por el conjunto de normas SQuaRE, ISO 25000:2014, la cual desarrolla los mismos conceptos. Aun así, este estándar continúa siendo el más utilizado. ISO 9126 distingue entre fallo y no conformidad. Un fallo es el cumplimiento de los requisitos especificados. Una distinción similar es la que se establece entre validaciones y verificación (Wikipedia, https://es.wikipedia.org/wiki/ISO/IEC_9126, fecha Marzo de 2020).

2.15.1 Características

El modelo de calidad clasifica la calidad del software en un conjunto estructurado de características y subcaracterísticas.

- **Funcionalidad**: Un conjunto de atributos que se relacionan con la existencia de un conjunto de funciones y sus propiedades específicas. Las funciones son aquellas que satisfacen las necesidades implícitas o explícitas.

- **Fiabilidad**: Un conjunto de atributos relacionados con la capacidad del software de mantener su nivel de prestación bajo condiciones establecidas durante un período establecido.
- **Usabilidad**: Un conjunto de atributos relacionados con el esfuerzo necesario para su uso, y en la valoración individual de tal uso, por un establecido o implicado conjunto de usuarios.
- **Eficiencia**: Conjunto de atributos relacionados con la relación entre el nivel de desempeño del software y la cantidad de recursos necesitados bajo condiciones establecidas.
- **Mantenibilidad**: Conjunto de atributos relacionados con la facilidad de extender, modificar o corregir errores en un sistema software.

2.15.2 Análisis de costos COCOMO II

El modelo original COCOMO (Constructive Cost Model) fue publicado por primera vez en 1981 por Barry Boehm y reflejaba las prácticas en desarrollo de software de ese entonces. En las décadas siguientes las técnicas de desarrollo de software cambiaron drásticamente.

Estos cambios incluyen el gasto de tanto esfuerzo en diseñar y gestionar el proceso de desarrollo software como en la creación del producto software, un giro total desde los mainframes que trabajan con procesos batch nocturnos hacia los sistemas en tiempo real y un énfasis creciente en la reutilización de software ya existente y en la construcción de nuevos sistemas que utilizan componentes software a medida. (Boehm, 1981).

Los años y el avance tecnológico hicieron que la aplicación del modelo COCOMO original empezara a resultar problemática, como solución se determinó reinventar el modelo para aplicarlo nuevamente y así después de muchos años de esfuerzo combinado surge

COCOMO II, un modelo de estimación de coste que refleja los cambios en la práctica de desarrollo de software profesional.

Este nuevo y mejorado COCOMO resultará de gran ayuda para los estimadores profesionales de coste software. El principal cálculo en el modelo COCOMO es el uso de la ecuación del esfuerzo para estimar el número de personas o de meses necesarios para desarrollar el proyecto.

COCOMO define tres modos de desarrollo o tipos de proyectos:

- **Orgánico:** proyectos relativamente sencillos, menores de 50 KDLC líneas de código, en los cuales se tiene experiencia de proyectos similares y se encuentran en entornos estables.
- **Semi-acoplado:** proyectos intermedios en complejidad y tamaño (menores de 300 KDLC), donde la experiencia en este tipo de proyectos es variable, y las restricciones intermedias.
- **Empotrado:** proyectos bastante complejos, en los que apenas se tiene experiencia y se engloban en un entorno de gran innovación técnica. Además, se trabaja con unos requisitos muy restrictivos y de gran volatilidad.

Y por otro lado existen diferentes modelos que define COCOMO:

- **Modelo básico:** Se basa exclusivamente en el tamaño expresado en LDC.
- **Modelo intermedio:** Además del tamaño del programa incluye un conjunto de medidas subjetivas llamadas conductores de costes.
- **Modelo avanzado:** Incluye todo lo del modelo intermedio además del impacto de cada conductor de coste en las distintas fases de desarrollo.

La función básica que utilizan los tres modelos es:

$$\boldsymbol{E = a(K \times l)b \times m(X)}$$

Donde:

- a y b son constantes con valores definidos en cada submodelo.
- Kl es la cantidad de líneas de código, en miles.
- m(X) Es un multiplicador que depende de 15 atributos. El resultado se da en unidades salario/mes y horas-hombre.

Modelo básico. Se utiliza para obtener una primera aproximación rápida del esfuerzo, y hace uso de la siguiente tabla de constantes para calcular distintos aspectos de costes:

MODO	a	b	c	d
Orgánico	2,4	1,05	2,5	0,38
Semilibre	3	1,12	2,5	0,35
Rígido	3,6	1,2	2,5	0,32

Tabla 2.7. Estimación de esfuerzo

Fuente: Grupo de investigación de costos (Beltrán 2018)

Estos valores son para realizar el cálculo de:

- Personas necesarias por mes para llevar adelante este proyecto

$$\boldsymbol{MM = a \times (Klb)}$$

- Tiempo de desarrollo del proyecto

$$\boldsymbol{TDEV = c \times (MMd)}$$

- Personas necesarias para realizar el proyecto

$$\boldsymbol{CosteH = \frac{MM}{TDEV}}$$

- Costo total del proyecto

$$CosteM = CosteH \times salario\ medio\ de\ programadores$$

Se puede observar que a medida que aumenta la complejidad del proyecto (modo), las constantes aumentan de 2.4 a 3.6, que corresponde a un incremento del esfuerzo del personal. Hay que utilizar con mucho cuidado el modelo básico puesto que se obvian muchas características del entorno.

Atributos. Cada atributo se cuantifica para un entorno de proyecto. La escala es muy bajo, bajo, nominal, alto, muy alto, extremadamente alto. Dependiendo de la calificación se asigna un valor para usar de multiplicador en la fórmula (por ejemplo, si para un proyecto el atributo DATA es calificado como muy alto, el resultado de la fórmula debe ser multiplicado por 1000).

El significado de los atributos según el tipo es el siguiente:

Software

- **RELY:** garantía de funcionamiento requerida al software. Indica las posibles consecuencias para el usuario en el caso que existan defectos en el producto. Va desde la sola inconveniencia de corregir un fallo (muy bajo) hasta la posible pérdida de vidas humanas (extremadamente alto, software de alta criticidad).
- **DATA:** tamaño de la base de datos en relación con el tamaño del programa. El valor del modificador se define por la relación: D / K, donde D corresponde al tamaño de la base de datos en bytes y K es el tamaño del programa en cantidad de líneas de código.
- **CPLX:** representa la complejidad del producto.

Hardware

- **TIME:** limitaciones en el porcentaje del uso de la CPU.
- **STOR:** limitaciones en el porcentaje del uso de la memoria.
- **VIRT:** volatilidad de la máquina virtual.
- **TURN:** tiempo de respuesta requerido.

Personal

- **ACAP:** calificación de los analistas.
- **AEXP:** experiencia del personal en aplicaciones similares.
- **PCAP:** calificación de los programadores.
- **VEXP:** experiencia del personal en la máquina virtual.
- **LEXP:** experiencia en el lenguaje de programación a usar.

Proyecto

- **MODP:** uso de prácticas modernas de programación.
- **TOOL:** uso de herramientas de desarrollo de software.
- **SCED:** limitaciones en el cumplimiento de la planificación.

Los valores de los atributos se muestran a continuación

Atributos	**Valor**					
	Muy bajo	Bajo	Nominal	Alto	Muy alto	Extra alto
Atributos de software						
Fiabilidad	0,75	0,88	1,00	1,15	1,40	
Tamaño de Base de datos		0,94	1,00	1,08	1,16	

Complejidad	0,70	0,85	1,00	1,15	1,30	1,65
Atributos de hardware						
Restricciones de tiempo de ejecución			1,00	1,11	1,30	1,66
Restricciones de memoria virtual			1,00	1,06	1,21	1,56
Volatilidad de la máquina virtual		0,87	1,00	1,15	1,30	
Tiempo de respuesta		0,87	1,00	1,07	1,15	
Atributos de personal						
Capacidad de análisis	1,46	1,19	1,00	0,86	0,71	
Experiencia en la aplicación	1,29	1,13	1,00	0,91	0,82	
Calidad de los programadores	1,42	1,17	1,00	0,86	0,70	
Experiencia en la máquina virtual	1,21	1,10	1,00	0,90		
Experiencia en el lenguaje	1,14	1,07	1,00	0,95		
Atributos del proyecto						
Técnicas actualizadas de programación	1,24	1,10	1,00	0,91	0,82	
Utilización de herramientas de software	1,24	1,10	1,00	0,91	0,83	
Restricciones de tiempo de desarrollo	1,23	1,08	1,00	1,04	1,10	

Tabla 2.8. Estimación de costos

Fuente: Grupo de investigación de costos (Beltrán 2018)

2.16 Pruebas de Caja Blanca

Según el sitio E-GOV (https://egov.ufsc.br/portal/conteudo/propuesta-de-procedimiento-para-realizar-pruebas-de-caja-blanca-las-aplicaciones-que-se-des, octubre de 2012) las pruebas de Caja Blanca pretende investigar sobre la estructura interna del código, exceptuando detalles referidos a datos de entrada o salida, para probar la lógica del programa desde el punto de vista algorítmico. Realizan un seguimiento del código fuente según se va ejecutando los casos de prueba, determinándose de manera concreta las instrucciones, bloques, etc. que han sido ejecutados por los casos de prueba.

En las pruebas de Caja Blanca se desarrollan casos de prueba que produzcan la ejecución de cada posible ruta del programa o módulo, considerándose una ruta como una combinación específica de condiciones manejadas por un programa.

Hay que señalar que no todos los errores de software se pueden descubrir verificando todas las rutas de un programa, hay errores que se descubren al integrar unidades del sistema y pueden existir errores que no tengan relación con el código específicamente.

2.16.1 Prueba del Camino Básico

Buscando una mejor comprensión de los contenidos, se hace importante definir primeramente algunos conceptos fundamentales:

Camino: Un camino se puede definir como la ruta de secuencias que se siguen dentro del código de fuente de un programa, un ejemplo es, desde la entrada de valores al sistema hasta la devolución de resultados que arroja, respetando la estructura de código.

Camino Básico: Es una técnica de prueba de Caja Blanca que permite obtener una medida de complejidad lógica para generar un conjunto básico de caminos que se ejecutan por lo menos una vez durante la ejecución del programa.

Camino independiente: El conjunto de caminos independientes se obtiene construyendo el Grafo de Flujo asociado y se calcula su complejidad ciclomática. Por último, se diseñan los casos de prueba y se ejecutan los mismos.

Complejidad: Es proporcional al número de errores en un segmento de código. Se relaciona con el esfuerzo requerido para probar.

Complejidad ciclomática: Es la medida de la complejidad lógica de un módulo "G" y el esfuerzo mínimo necesario para calificarlo. Es el número de rutas lineales independientes de un módulo "G", por lo tanto, es el número mínimo de rutas que deben probarse"

Esta técnica ofrece una gran ventaja con respecto a las otras técnicas, ya que el número mínimo requerido de pruebas se sabe por adelantado y por tanto el proceso de prueba se puede planear y supervisar en mayor detalle.

Los pasos a seguir para aplicar esta técnica son:

- Representar el programa en un grafo de flujo.
- Calcular la complejidad ciclomática.
- Determinar el conjunto básico de caminos independientes.
- Derivar los casos de prueba que fuerzan la ejecución de cada camino.

A continuación, se detallan cada uno de estos pasos.

2.16.2 Representación de un grafo de flujo

El grafo de flujo se utiliza para representar el flujo de control lógico de un programa. Este emplea los tres elementos siguientes:

- Nodos: Representan cero, una o varias sentencias en secuencia. Cada uno comprende como máximo una sentencia de decisión (bifurcación).
- Aristas: Líneas que unen dos nodos.
- Regiones: Áreas delimitadas por aristas y nodos. Cuando se contabilizan las regiones de un programa debe incluirse el área externa como una región más.

2.16.3 Calcular la complejidad ciclomática

La complejidad ciclomática es una métrica del software que proporciona una medida cuantitativa de la complejidad lógica de un programa. Se basa en la representación gráfica del flujo de control del programa, el análisis desprende una medida cuantitativa de la dificultad de prueba y una indicación de la fiabilidad final. Cuando se utiliza en el contexto del método de prueba del camino básico, el valor calculado como complejidad ciclomática define el número de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez.

2.16.4 Determinar el conjunto de caminos básicos independientes

Camino linealmente independiente de otros: Introduce por lo menos un nuevo conjunto de sentencias de proceso o una nueva condición.

Un camino independiente es cualquier camino del programa que introduce, por lo menos, un nuevo conjunto de sentencias de proceso o una condición, respecto a los caminos existentes. En términos del diagrama de flujo, un camino independiente está constituido por lo menos

por una arista que no haya sido recorrida anteriormente a la definición del camino. En la identificación de los distintos caminos de un programa para probar se debe tener en cuenta que cada nuevo camino debe tener el mínimo número de sentencias nuevas o condiciones nuevas respecto a los que ya existen. De esta manera se intenta que el proceso de depuración sea más sencillo.

2.16.5 Derivar los casos de prueba

El último paso es construir los casos de prueba que fuerzan la ejecución de cada camino.

CAPITULO III

MARCO APLICATIVO

3.1 Introducción

El objetivo del marco aplicativo es la puesta en marcha de las actividades de proceso de desarrollo que nos sugiere la metodología Scrum. En este capítulo se describirá el ámbito aplicativo de la investigación, identificando los requerimientos generales tanto de los clientes como la institución y el proceso de desarrollo de los componentes de la aplicación y su adaptación a la metodología aplicada.

3.2 FASE 1: Definir backlog del producto

Durante la fase de creación de la Backlog del producto (Pila de Producto) se capturarán los requerimientos del usuario, los cuales serán ítems de la pila del producto. Si bien se obtendrá el mayor número de requerimientos posible, la priorización de los mismos será realizada por el Propietario del Producto, que en este caso será de el subgerente Lic. Giovanni Terrazas unos de los responsables de la división de Soluciones de Negocio.

3.2.1 Historias de usuario

Para la representación de los requerimientos encontrados, se hará uso de las Historias de Usuario establecidas por la metodología propuesta en el presente proyecto.

Afiliación de una cuenta Billetera

Este módulo realiza la función de afilar o registrar un usuario o cuenta billetera, también realizará el proceso de alta para realizar transacciones financieras.

Inicio de sesión

Este módulo realiza la función de ingresar o iniciar sesión en la aplicación, de esta manera el usuario podrá realizar las operaciones o transacciones financieras que el usuario necesite.

Recargas de crédito de telefónicas

Este módulo realiza la función de recargar de crédito a líneas telefónicas, de esta manera el usuario podrá realizar recargas para realizar llamadas o comprar datos de internet.

Transferencias entre cuentas billetera

El módulo realiza la función de transferir o enviar dinero a otras cuentas billeteras en la cual el usuario podar ingresar el número de cuenta del beneficiario y el monto deseado.

Transferencias a cuentas BCP

El módulo realiza la función de transferir o enviar dinero a cuentas BCP en la cual el usuario podar ingresar el número de cuenta del beneficiario y el monto deseado.

Transferencias a cuentas BCP

El módulo realiza la función de transferir o enviar dinero a cuentas BCP en la cual el usuario podar ingresar el número de cuenta del beneficiario y el monto deseado.

Transferencias a cuentas de otros bancos

El módulo realiza la función de transferir o enviar dinero a cuentas de otros bancos en la cual el usuario podar ingresar el número de cuenta del beneficiario, nombre del beneficiario y el monto deseado.

Estados de transferencias y movimientos

En este módulo se realizará la función de mostrar la información detallada como los movimientos y estado de transacciones así mismo los códigos secretos para realizar los retiros en cajeros automáticos o compras.

Pago de servicios

En este módulo se realizará la función de pagos de servicios para el pago de deudas o facturas pendientes como por ejemplo VIVA, TIGO, ENTEL, EPSAS, YPFB y DELAPAZ.

Retiro en cajeros automáticos y compra en comercios

El módulo se realizará la función de generar un código secreto para que el usuario pueda efectuar el retiro de dinero mediante cajero o también puede realizar el código secreto para realizar una compra en los comercios que cuenten con máquinas POS.

Cambio y reseteo de PIN

Estos módulos deben realizaran la función de cambiar y resetear el pin en caso de que el usuario haya olvidado su pin y también para mantener segura la cuenta billetera.

Depósitos en cajeros automáticos para no clientes

El módulo debe realizara la función generar un código y enviarlo mediante sms para realizar el depósito a una cuenta BCP para las personas no clientes.

3.2.2 Historias identificadas

Afiliación de una cuenta Billetera

N°	1	Historia de usuario		
Usuario/Autor		Giovanni Terrazas	Sprint	1
Área		Soluciones de Negocio	Prioridad	1
Programador		Peter Alanoca	Estimación	20
Nombre de la historia		**Afiliación de una cuenta Billetera**		
Descripción: Como cliente quiero afiliarme o registrarme para tener una cuenta billetera				
Criterio de aceptación: Es necesario que se llenen los siguientes campos: • Nombres • Cedula de identidad • Celular • Correo • PIN Debe validarse el documento de identidad con el servicio de SEGIP Debe validarse el celular mediante sms Debe validarse que el pin y comfirmarse confirme una segunda ves				

Tabla 3.1. Historia de usuario afiliación de una cuenta Billetera

Fuente: Elaboración propia

Inicio de sesión

N°	2	Historia de usuario		
Usuario/Autor		Giovanni Terrazas	Sprint	1
Área		Soluciones de Negocio	Prioridad	2
Programador		Edson Mamani	Estimación	20
Nombre de la historia		**Inicio de sesión**		
Descripción: Como cliente quiero iniciar sesión para tener acceder a la aplicación con mi cuenta billetera				
Criterio de aceptación: Es necesario que se llenen los siguientes campos: • Celular • PIN Debe validarse las credenciales con el servicio web interno de la institución. Tienen que tener visibles la opción de olvido de pin, afiliación y deposito para no clientes.				

Tabla 3.2. Historia de usuario inicio de sesión

Fuente: Elaboración propia

Opción de Inicio o bienvenida

N°	3	Historia de usuario	
Usuario/Autor	Giovanni Terrazas	Sprint	2
Área	Soluciones de Negocio	Prioridad	1
Programador	Peter Alanoca	Estimación	10
Nombre de la historia	**Opción de inicio o bienvenida**		
Descripción: Como cliente quiero que la opción inicio me muestre mi saldo y las ultimas transacciones, también que me de accesos a otras opciones como por ejemplo transferencias para tener la información de mi cuenta al momento de iniciar sesión.			
Validación: Debe mostrarse la siguiente información: • Saldo • Ultimas transacciones. Debe tener acceso a las siguientes opciones: • Transferencias • Recarga de crédito.			

Tabla 3.3. Historia de usuario opción de inicio o bienvenida

Fuente: Elaboración propia

Recargas de crédito de telefónicas

N°	4	Historia de usuario	
Usuario/Autor	Giovanni Terrazas	Sprint	2
Área	Soluciones de Negocio	Prioridad	2
Programadores	Peter Alanoca Edson Mamani	Estimación	10
Nombre de la historia	**Recargas de crédito de telefónicas**		
Descripción: Como cliente necesito recargar crédito a mi línea telefónica para realizar llamadas o compras datos de internet.			
Criterio de aceptación: Debe estar muy visible o distinguirse la empresa telefónica a escoger. Se debe de tener algunos montos predefinidos. Deben validarse tanto los campos: • Número de celular • Monto			

Tabla 3.4. Historia de usuario recargas de crédito de telefónicas

Fuente: Elaboración propia

Transferencias entre cuentas billetera

N°	5	Historia de usuario	
Usuario/Autor	Giovanni Terrazas	Sprint	2
Área	Soluciones de Negocio	Prioridad	3
Programador	Edson Mamani	Estimación	10
Nombre de la historia	**Transferencias entre cuentas billetera**		
Descripción: Como cliente quiero transferir dinero a otras billeteras para pagar o enviar dinero a otra cuenta billetera.			
Criterio de aceptación: Deben validarse tanto los campos: • Número de cuenta • Monto Debe validarse el pin internamente.			

Tabla 3.5. Historia de usuario transferencias entre cuentas billetera

Fuente: Elaboración propia

Transferencias a cuentas BCP

N°	6	Historia de usuario	
Usuario/Autor	Giovanni Terrazas	Sprint	3
Área	Soluciones de Negocio	Prioridad	1
Programador	Peter Alanoca	Estimación	20
Nombre de la historia	**Transferencias a cuentas BCP**		
Descripción: Como cliente quiero transferir dinero a cuentas BCP para pagar o enviar dinero a cuentas BCP.			
Criterio de aceptación: Deben validarse tanto los campos: • Número de cuenta • Monto Debe validarse el pin internamente.			

Tabla 3.6. Historia de usuario transferencias a cuentas BCP

Fuente: Elaboración propia

Transferencias a cuentas de otros bancos

N°	7	Historia de usuario	
Usuario/Autor	Giovanni Terrazas	Sprint	3
Área	Soluciones de Negocio	Prioridad	2
Programador	Edson Mamani	Estimación	20
Nombre de la historia	**Transferencias a cuentas de otros bancos**		
Descripción: Como cliente quiero transferir dinero a cuentas de otros bancos para pagar o enviar dinero a cuentas de otros bancos.			
Criterio de aceptación: Deben validarse tanto los campos: • Número de cuenta • Nombre del beneficiario • Banco destino • Monto Debe validarse el pin internamente. La confirmación de la transferencia será en segundo plano y se podrá ver el estado en la opción de estados de transferencias.			

Tabla 3.7. Historia de usuario transferencias a cuentas de otros bancos

Fuente: Elaboración propia

Estados de transferencias y movimientos

N°	8	Historia de usuario	
Usuario/Autor	Giovanni Terrazas	Sprint	4
Área	Soluciones de Negocio	Prioridad	1
Programador	Peter Alanoca	Estimación	20
Nombre de la historia	**Estados de transferencias y movimientos**		
Descripción: Como cliente quiero consultar mis movimientos, códigos de retiro y estado de transferencias para tener la información financiera actualizada.			
Criterio de aceptación: Deben mostrarse la siguiente información: • Saldo • Ultimas 20 transacciones o movimientos Los movimientos deben estar ordenada por fecha de forma descendiente.			

Tabla 3.8. Historia de usuario estados de transferencias y movimientos

Fuente: Elaboración propia

Pago de servicios

N°	9	Historia de usuario	
Usuario/Autor	Giovanni Terrazas	Sprint	4
Área	Soluciones de Negocio	Prioridad	2
Programador	Edson Mamani	Estimación	20
Nombre de la historia	**Pago de servicios**		
Criterio de aceptación: Como cliente quiero pagar mis deudas o facturas pendientes de servicios básicos o telefónicas para que no se realice el corte de los servicios que se consume.			
Validación: Deben validarse los campos: • Código de servicio • Monto • Empresa Las empresas deben estar categorizadas por rubro. Debe validarse el pin internamente.			

Tabla 3.9. Historia de usuario estados de pago de servicios

Fuente: Elaboración propia

Retiro en cajeros automáticos y compra en comercios

N°	10	Historia de usuario	
Usuario/Autor	Giovanni Terrazas	Sprint	5
Área	Soluciones de Negocio	Prioridad	1
Programador	Peter Alanoca	Estimación	10
Nombre de la historia	**Retiro en cajeros automáticos y compra en comercios**		
Criterio de aceptación: Como cliente quiero realizar retiros en cajeros automáticos y comprar en comercios para poder disponer del dinero de manera física.			
Validación: Debe mostrarse el código secreto para realizar el retiro. El código secreto deberá tener un tiempo de vida de 30 minutos máximo, la validación lo hará el servicio web que genera los códigos. El código debe estar visible en la opción de estados o transferencias. Debe validarse el pin internamente.			

Tabla 3.10. Historia de usuario retiro en cajeros automáticos y compra en comercios

Fuente: Elaboración propia

Cambio y reseteo de PIN

N°	11	Historia de usuario	
Usuario/Autor	Giovanni Terrazas	Sprint	5
Área	Soluciones de Negocio	Prioridad	2
Programadores	Peter Alanoca Edson Mamani	Estimación	10
Nombre de la historia	**Cambio y reseteo de PIN**		
Descripción: Como cliente quiero cambiar o resetear mi pin para mantener segura mi cuenta, también tener un pin nuevo en caso de olvido.			
Criterio de aceptación: Deben validarse los campos: • Número de cuenta • Pin nuevo • Pin actual En el caso de reseteo de pin se tendrá que llamar a banca por teléfono para solicitar un código realizar el reseteo.			

Tabla 3.11. Historia de usuario cambio y reseteo de PIN

Fuente: Elaboración propia

Depósitos en cajeros automáticos para no clientes

N°	12	Historia de usuario	
Usuario/Autor	Giovanni Terrazas	Sprint	5
Área	Soluciones de Negocio	Prioridad	3
Programador	Edson Mamani	Estimación	10
Nombre de la historia	**Depósitos en cajeros automáticos para no clientes**		
Descripción: Como cliente quiero realizar depósitos en cajeros automáticos para no clientes para depositar dinero a cuentas BCP.			
Criterio de aceptación: Deben validarse los campos: • Número de cuenta • Nombre del beneficiario • Celular • Cedula de identidad • Origen y destino de fondos Se envía un código sms para realizar el depósito.			

Tabla 3.12. Historia de usuario depósitos en cajeros automáticos para no clientes

Fuente: Elaboración propia

3.3 FASE 2: Planificación del Sprint

Para llevar a cabo la reunión de planificación de Sprint, previamente el equipo debió asegurarse que el Product Backlog se encuentre bien definido.

El equipo para este proyecto fue conformado de la siguiente manera:

Product Owner	Lic. Giovanni Terrazas
Scrum Master	Peter Alanoca Aruquipa
Desarrollador	Peter Alanoca Aruquipa
Desarrollador	Edson Mamani Apaza

Tabla 3.13. Team Scrum

Fuente: Elaboración propia

La primera reunión de planificación de Sprints, permitirá que el equipo Scrum estructure los Sprints necesarios, además que realice todas las estimaciones iniciales y que verifique las importancias establecidas por el cliente, tal como se muestra en la tabla:

Id	Nombre	Importancia	Estimación	Como probarlo
1	Afiliación de una cuenta Billetera	80	20	Llenar todos los datos y presionar el botón Registrar
2	Inicio de sesión	20	20	Llenar todos los datos y presionar el botón Ingresar
3	Opción de Inicio	80	10	Iniciar sesión será la primera opción visible
4	Recargas de crédito de telefónicas	40	10	Llenar todos los datos y presionar el botón Recargar

5	Transferencias entre cuentas billetera	40	10	Llenar todos los datos y presionar el botón Enviar pago
6	Transferencias a cuentas BCP	40	20	Llenar todos los datos y presionar el botón Enviar pago
7	Transferencias a cuentas de otros bancos	40	20	Llenar todos los datos y presionar el botón Enviar pago
8	Estados de transferencias y movimientos	60	20	Seleccionar la opción de Consultas
9	Pago de servicios	80	20	Seleccionar la opción de Servicios, seleccionar una empresa llenar todos los datos y presionar el botón Pagar
10	Retiro en cajeros automáticos y compra	70	10	Seleccionar la opción de Contraseña y presionar el botón Generar
11	Cambio y reseteo de PIN	40	10	Seleccionar la opción de Cambio de Pin, llenar todos los datos y presionar el botón Cambiar
12	Depósitos en cajeros automáticos para no clientes	40	10	Seleccionar la opción de Deposito, llenar todos los datos y presionar el botón Depositar

Tabla 3.14. Product Backlog

Fuente: Elaboración propia

3.3.1 Primera reunión de planificación de Sprint (SPRINT 1)

Fecha: Jueves 7/02/2019

Lugar: Banco de Crédito calle Colón esquina Mercado. - Piso 3 Sala "8"

Próxima reunión: Viernes 15/03/2019

- **9:00 – 9:30.** El dueño de producto comenta la meta del Sprint y resume la Pila de Producto. Se establece el lugar, fecha y hora para la revisión del Sprint.

  Meta del primer Sprint:

  a) Afiliación de una cuenta Billetera

  b) Inicio de sesión

- **9:30 – 10:00.** El equipo Scrum da estimaciones de tiempo, y divide los elementos tanto como sea necesario de acuerdo a su experiencia. El dueño de producto actualiza las ratios de importancia. Se clarifican los elementos. Para todos los elementos de alta importancia se establece la columna "Cómo probarlo".
- **10:00 – 10:30.** El equipo selecciona las historias que se incluirán en el Sprint. Se realizan cálculos de velocidad para chequear si es factible.
- **10:30 – 11:00.** Se selecciona un lugar y hora para el Scrum Diario. Se continúa dividiendo las historias en tareas.

El Sprint Planning es una reunión crítica, probablemente el evento más importante en Scrum, ya que una reunión de planificación mal ejecutada puede llevar a incumplir un Sprint entero.

El propósito de la reunión de planificación de Sprint es dar al equipo suficiente información para ser capaz de trabajar en paz por cuatro semanas, y proporcionarle al dueño de producto los entregables de la meta de Sprint en la fecha acordada para su revisión y retrospectiva de

ser el caso. Los desarrolladores Peter Alanoca y Edson Mamani estarán en el proyecto enfocados 80 días (640 horas laborales) y líder del equipo Scrum será Peter Alanoca, también que no se tomaran en cuenta las siguientes fechas 4 y 5 de marzo, 19 de abril, 1 de mayo, 20 y 21 de junio y 16 de julio por el tema de feriados.

Consolidando todo ello tenemos:

Desarrollador	Días disponibles	Horas disponibles
Peter Alanoca	100	800
Edson Mamani	100	800
TOTAL	200	1600

Tabla 3.15. Sprint Planning días y horas disponibles

Fuente: Elaboración propia

Lo cual da un total de 640 horas disponibles para el proyecto. Esta estimación se ve reflejada en el cronograma de la propuesta técnica proporcionada al usuario.

La tabla N° 3.16 muestra los cinco Sprints backlogs definidos en la primera reunión de planificación de Sprints.

Sprint	Responsable	Tareas	Horas
SPRINT 1	**Peter Alanoca**	Realizar el análisis y diseño del proyecto	**40**
		Realizar el modelamiento de la base de datos.	**32**
		Creación de tablas y procedimientos almacenados	**24**
		Creación de servicio web	**32**
		Creación de los endpoints de registro	**32**
	Edson Mamani	Consumir los servicios de envio de sms interno	**16**
		Consumir los servicios de Segip	**16**
		Creación del endpoint de login	**28**
		Consumir el servicio de autenticación interno	**28**
		Creación de la interfaz del registro en las aplicaciones móviles	**32**
		Creación de la interfaz del login en las	**40**

		aplicaciones móviles	
			320
Sprint	**Responsable**	**Tareas**	**Horas**
SPRINT 2	**Peter Alanoca**	Creación de tablas y procedimientos almacenados	**32**
		Creación del endpoint de inicio	**32**
		Creación de los endpoints de recargas	**40**
		Consumir los de recargas	**16**
		Creación de los endpoints de transferencias entre cuentas billetera	**40**
	Edson Mamani	Creación de la interfaz de la opción en las aplicaciones móviles	**40**
		Creación de la interfaz de recargas en las aplicaciones móviles	**60**
		Creación de la interfaz de transferencias entre cuentas billetera en las aplicaciones móviles	**60**
			320
Sprint	**Responsable**	**Tareas**	**Horas**
SPRINT 3	**Peter Alanoca**	Creación de tablas y procedimientos almacenados	**40**
		Creación de los endpoints de transferencias a cuenta BCP	**40**
		Consumir el servicio de abonos a cuentas BCP	**40**
		Creación de los endpoints de transferencias a cuentas de otros bancos	**40**
	Edson Mamani	Consumir el servicio de abonos a cuentas de otros bancos	**60**
		Creación de la interfaz de transferencias a cuentas BCP en las aplicaciones móviles	**60**
		Creación de la interfaz de transferencias a de otros bancos en las aplicaciones móviles	**60**
			320
Sprint	**Responsable**	**Tareas**	**Horas**
SPRINT 4	**Peter Alanoca**	Creación de tablas y procedimientos almacenados	**40**
		Creación del endpoint de movimientos	**40**
		Creación del endpoint de estado de transferencias	**44**
		Consumir el servicio de código secreto para retiro	**36**
	Edson Mamani	Creación de los endpoints de pago de servicios	**40**
		Consumir el servicio de pago de servicios	**32**
		Creación de la interfaz de movimientos y estados de transferencias	**32**
		Creación de la interfaz de pagos de servicios en las aplicaciones móviles	**56**
			320
Sprint	**Responsable**	**Tareas**	**Horas**
SPRINT 5	**Peter Alanoca**	Creación de tablas y procedimientos almacenados	**32**
		Creación del endpoint de código secreto para retiro	**32**
		Consumir el servicio de código secreto para retiro	**16**

		Creación del endpoint de cambio de pin	**16**
		Creación del endpoint de reseteo de pin	**32**
		Creación del endpoint de depósito para no clientes	**32**
	Edson Mamani	Consumir el servicio de depósito para no clientes	**24**
		Creación de la interfaz de retiros en cajeros en las aplicaciones móviles	**32**
		Creación de la interfaz de cambio de pin	**32**
		Creación de la interfaz de reseteo de pin	**32**
		Creación de la interfaz de depósitos para no clientes	**40**
			320

Tabla 3.16. Sprints backlog definidos en la primera reunión de planificación

Fuente: Elaboración propia

La tabla N° 3.16 contiene el responsable por cada tarea de cada Sprint, que fue producto de la primera reunión de planificación de Sprints. Donde todos los Sprints tienen la duración de 20 días (4 semanas laborales)

La estimación de tiempo para el Sprint 1, se determinó de la siguiente manera:

Desarrollador	Días disponibles	Horas disponibles
Peter Alanoca	20	160
Edson Mamani	20	160
TOTAL	40	320

Tabla 3.17. Estimación de tiempo para el Sprint 1

Fuente: Elaboración propia

Para el proyecto, considerando que el factor de dedicación para este primer Sprint es igual a: $F_{dedicación} = 10/10 = 1$. Entonces, la velocidad estimada será igual a: $V_{estimada} = 10x1 = 10$.

3.3.2 Segunda reunión de planificación de Sprint (SPRINT 2)

Fecha: Viernes 15/03/2019

Lugar: Banco de Crédito calle Colón esquina Mercado. - Piso 3 Sala “8”

Próxima reunión: Martes 16/04/2019

- **9:00 – 9:30.** El equipo Scrum proporciona al dueño del producto los entregables de producto de las metas de Sprint acordadas en la reunión anterior.
- **9:30 – 10:00.** El dueño de producto verifica las metas de Sprint y valida si es lo que solicitó en el product backlog. En este caso hubo un error en el modelamiento de la base de datos y se deberá hacer la respectiva retrospectiva del Sprint.
- **10:00 – 10:30.** El dueño de producto establece las metas del Sprint para el siguiente Sprint. Se establece el lugar, fecha y hora para la revisión del Sprint. El equipo selecciona las historias que se incluirán en el Sprint.

  Meta de segundo Sprint:

  a) Opción de Inicio

  b) Recargas de crédito de telefónicas

  c) Transferencias entre cuentas billetera
- **10:30 – 11:00.** Se selecciona un lugar y hora para el Scrum Diario. El equipo Scrum continúa dividiendo las historias en tareas.

La estimación de tiempo para el Sprint 2, se determinó de la siguiente manera:

Desarrollador	Días disponibles	Horas disponibles
Peter Alanoca	20	160
Edson Mamani	20	160
TOTAL	40	320

Tabla 3.18. Estimación de tiempo para el Sprint 2

Fuente: Elaboración propia

Para el proyecto, considerando que el factor de dedicación para este primer Sprint es igual a: $F_{dedicación} = 10/10 = 1$. Entonces, la velocidad estimada será igual a: $V_{estimada} = 10x1 = 10$.

3.3.3 Tercera reunión de planificación de Sprint (SPRINT 3)

Fecha: Martes 16/04/2019

Lugar: Banco de Crédito calle Colón esquina Mercado. - Piso 3 Sala "8"

Próxima reunión: Martes 21/05/2019

- **9:00 – 9:30.** El equipo Scrum proporciona al dueño del producto los entregables de producto de las metas de Sprint acordadas en la reunión anterior.
- **9:30 – 10:00.** El dueño de producto verifica las metas de Sprint y valida si es lo que solicitó en el product backlog. En este caso hubo un error en el modelamiento de la base de datos y se deberá hacer la respectiva retrospectiva del Sprint.
- **10:00 – 10:30.** El dueño de producto establece las metas del Sprint para el siguiente Sprint. Se establece el lugar, fecha y hora para la revisión del Sprint. El equipo selecciona las historias que se incluirán en el Sprint.

  Meta del tercer Sprint:

  a) Transferencias a cuentas BCP

  b) Transferencias a cuentas de otros bancos
- **10:30 – 11:00.** Se selecciona un lugar y hora para el Scrum Diario. El equipo Scrum continúa dividiendo las historias en tareas.

La estimación de tiempo para el Sprint 3, se determinó de la siguiente manera:

Desarrollador	Días disponibles	Horas disponibles
Peter Alanoca	20	160
Edson Mamani	20	160
TOTAL	40	320

Tabla 3.19. Estimación de tiempo para el Sprint 3

Fuente: Elaboración propia

Para el proyecto, considerando que el factor de dedicación para este primer Sprint es igual a: $F_{dedicación} = 10/10 = 1$. Entonces, la velocidad estimada será igual a: $V_{estimada} = 10x1 = 10$.

3.3.4 Cuarta reunión de planificación de Sprint (SPRINT 4)

Fecha: Martes 21/05/2019

Lugar: Banco de Crédito calle Colón esquina Mercado. - Piso 3 Sala "8"

Próxima reunión: Martes 24/06/2019

- **9:00 – 9:30.** El equipo Scrum proporciona al dueño del producto los entregables de producto de las metas de Sprint acordadas en la reunión anterior.
- **9:30 – 10:00.** El dueño de producto verifica las metas de Sprint y valida si es lo que solicitó en el product backlog. En este caso hubo un error en el modelamiento de la base de datos y se deberá hacer la respectiva retrospectiva del Sprint.
- **10:00 – 10:30.** El dueño de producto establece las metas del Sprint para el siguiente Sprint. Se establece el lugar, fecha y hora para la revisión del Sprint. El equipo selecciona las historias que se incluirán en el Sprint.

  Meta de cuarto Sprint:

  a) Estados de transferencias y movimientos

b) Pago de servicios

- **10:30 – 11:00.** Se selecciona un lugar y hora para el Scrum Diario. El equipo Scrum continúa dividiendo las historias en tareas.

La estimación de tiempo para el Sprint 4, se determinó de la siguiente manera:

Desarrollador	Días disponibles	Horas disponibles
Peter Alanoca	20	160
Edson Mamani	20	160
TOTAL	40	320

Tabla 3.20. Estimación de tiempo para el Sprint 4

Fuente: Elaboración propia

Para el proyecto, considerando que el factor de dedicación para este primer Sprint es igual a: $F_{dedicación} = 10/10 = 1$. Entonces, la velocidad estimada será igual a: $V_{estimada} = 10x1 = 10$.

3.3.5 Quinta reunión de planificación de Sprint (SPRINT 5)

Fecha: Martes 24/06/2019

Lugar: Banco de Crédito calle Colón esquina Mercado. - Piso 3 Sala "8"

Próxima reunión: Viernes 26/07/2019

- **9:00 – 9:30.** El equipo Scrum proporciona al dueño del producto los entregables de producto de las metas de Sprint acordadas en la reunión anterior.
- **9:30 – 10:00.** El dueño de producto verifica las metas de Sprint y valida si es lo que solicitó en el product backlog. En este caso hubo un error en el modelamiento de la base de datos y se deberá hacer la respectiva retrospectiva del Sprint.

- **10:00 – 10:30.** El dueño de producto establece las metas del Sprint para el siguiente Sprint. Se establece el lugar, fecha y hora para la revisión del Sprint. El equipo selecciona las historias que se incluirán en el Sprint.

  Meta del quinto Sprint:

  a) Retiro en cajeros automáticos y compra en comercios

  b) Depósitos en cajeros automáticos para no clientes

- **10:30 – 11:00.** Se selecciona un lugar y hora para el Scrum Diario. El equipo Scrum continúa dividiendo las historias en tareas.

La estimación de tiempo para el Sprint 5, se determinó de la siguiente manera:

Desarrollador	Días disponibles	Horas disponibles
Peter Alanoca	20	160
Edson Mamani	20	160
TOTAL	40	320

Tabla 3.21. Estimación de tiempo para el Sprint 5

Fuente: Elaboración propia

Para el proyecto, considerando que el factor de dedicación para este primer Sprint es igual a: $F_{dedicación} = 10/10 = 1$. Entonces, la velocidad estimada será igual a: $V_{estimada} = 10x1 = 10$.

3.4 FASE 3: Scrum diario

3.4.1 Comunicación de Sprint Backlogs

Para poder comunicar el avance de cada uno de los cinco Sprints backlogs, se realizan los scrum diarios o reuniones diarias, en donde participan el Scrum Master y el Equipo Scrum principalmente para verificar y evaluar el avance realizado por los responsables de las tareas

asignadas. La finalidad de ello es que ninguna tarea sea un cuello de botella que impida la culminación del proyecto.

Sobre una gran pizarra, y con la ayuda de post-its y plumones, se construyó la tabla de tareas para el proyecto y se comunicaron los avances de los Sprints backlogs (Sprint 1, Sprint 2, Sprint 3, Sprint 4 y Sprint 5). Tal como se muestra en el gráfico N° 3.1.

Del gráfico de la página siguiente, se observa que es importante la comunicación de las tareas que se vienen realizando para ver el comportamiento del gráfico burndown a lo largo del proyecto, y tener así la perspectiva de si el avance es óptimo.

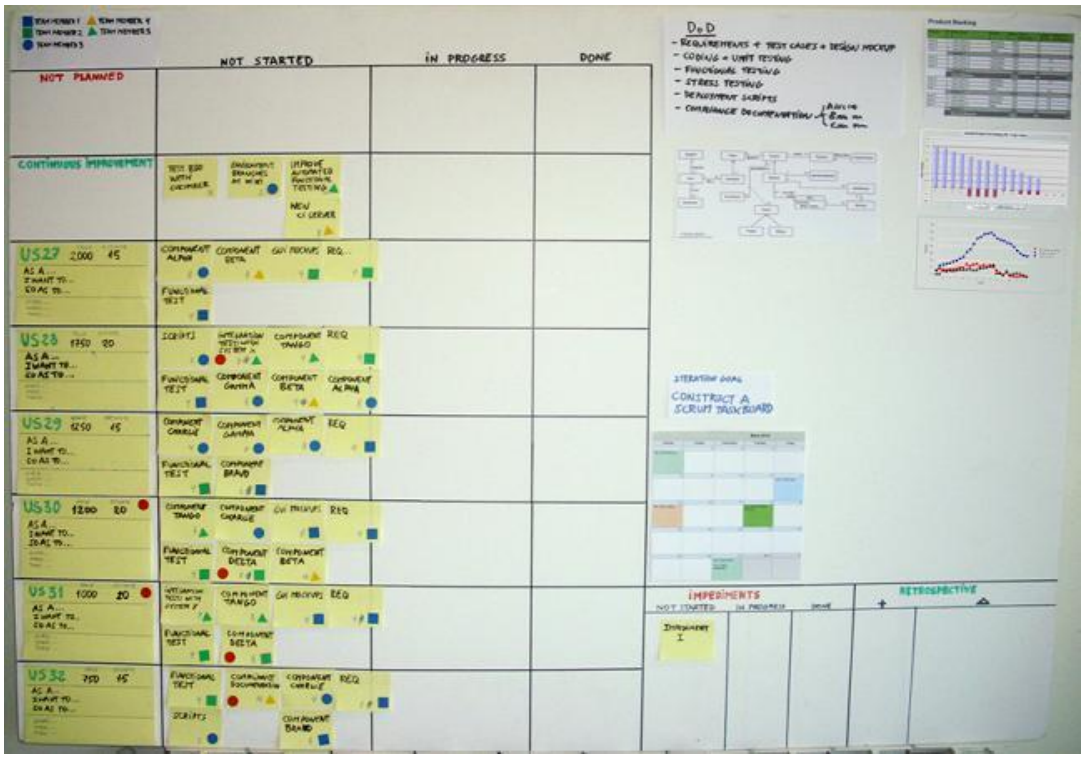


Figura 3.1. Sprint backlog para el proyecto de Billetera Móvil

Fuente: Elaboración propia

Del gráfico de la página siguiente, se observa que es importante la comunicación de las tareas que se vienen realizando para ver el comportamiento del gráfico burndown a lo largo del proyecto, y tener así la perspectiva de si el avance es óptimo.

3.4.2 Trabajando con el cuadro Burndown

La figura N° 3.2 muestra el cuadro burndown o gráfica de progreso para el Sprint backlog 1, el cual tuvo una duración de 20 días.

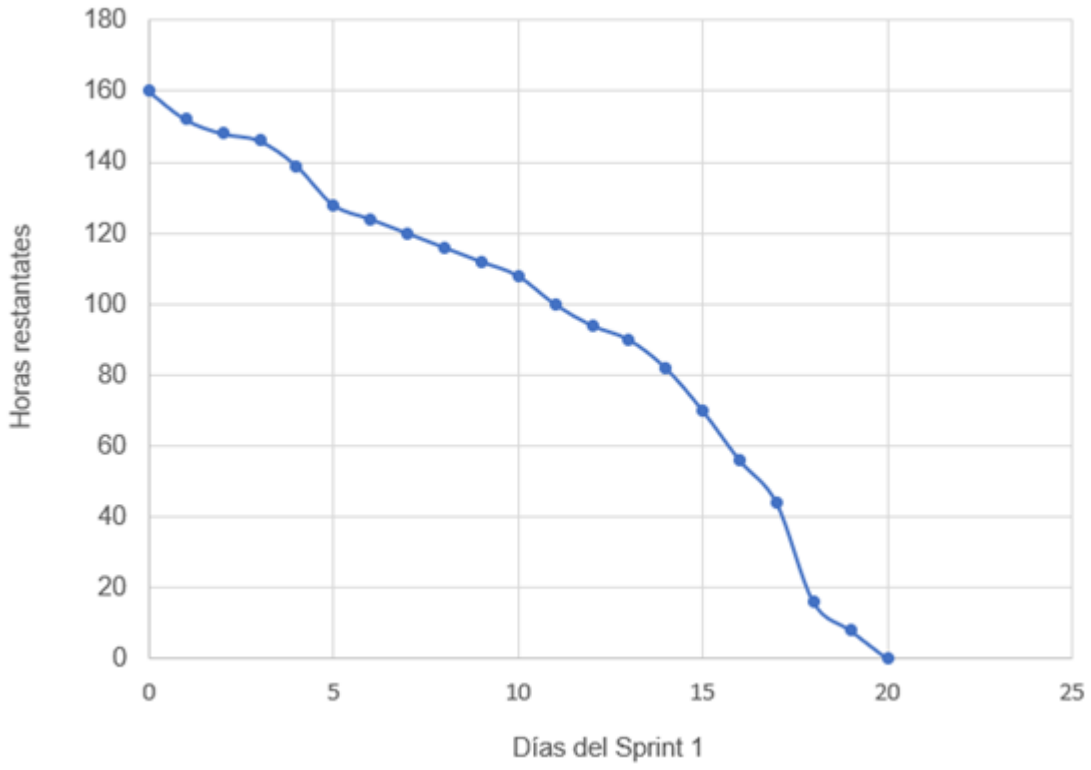


Figura 3.2. Diagrama burndown para el Sprint backlog 1

Fuente: Elaboración propia

Del gráfico se puede observar el comportamiento de la gráfica lineal, la cual tiende al decrecimiento, ello debido a las reuniones diarias que permiten la evaluación de avances y la

minimización de retrasos. De esta manera se va culminando el Sprint 1 hasta pasar al siguiente Sprint.

La figura N° 3.3 muestra el cuadro burndown o gráfica de progreso para el Sprint backlog 2, el cual tuvo una duración de 20 días.

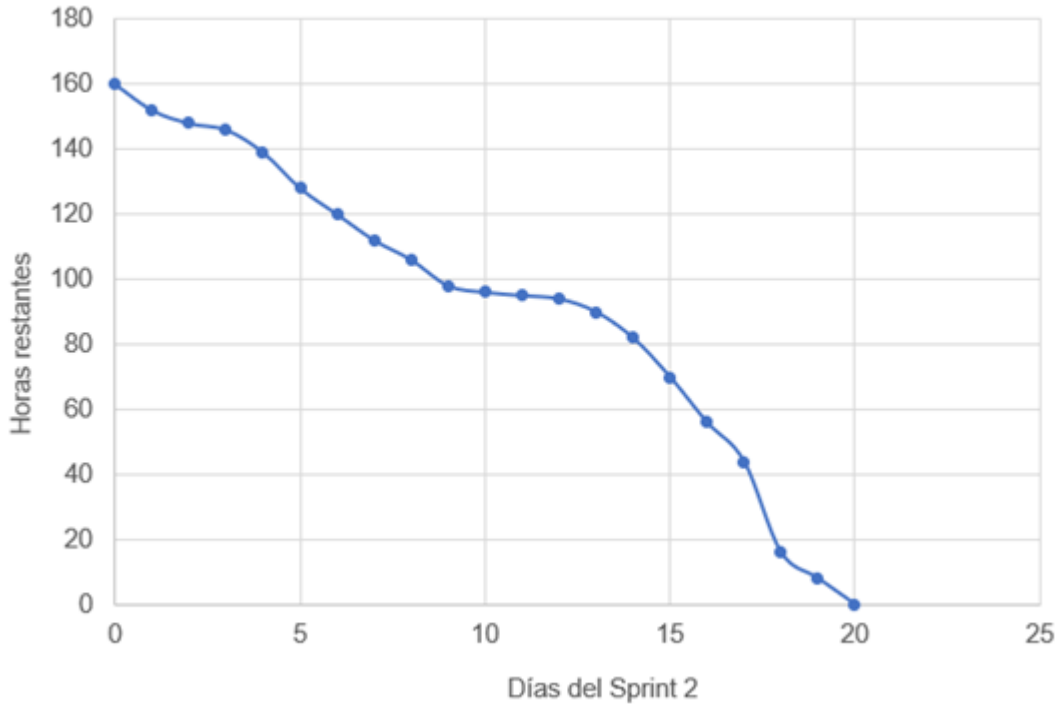


Figura 3.3. Diagrama burndown para el Sprint backlog 2

Fuente: Elaboración propia

Del gráfico se puede observar el comportamiento de la gráfica lineal, la cual tiende al decrecimiento.

La figura N° 3.4 muestra el cuadro burndown o gráfica de progreso para el Sprint backlog 3, el cual tuvo una duración de 20 días.

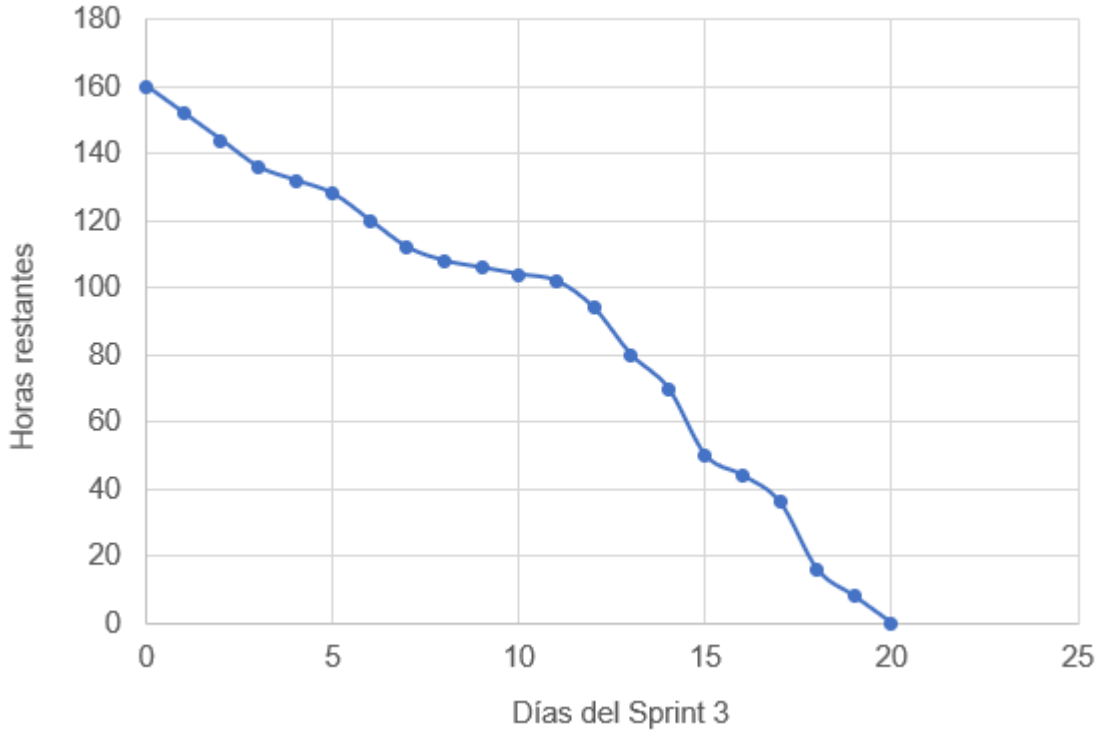


Figura 3.4. Diagrama burndown para el Sprint backlog 3

Fuente: Elaboración propia

Del gráfico se puede observar el comportamiento de la gráfica lineal, la cual tiende al decrecimiento, ello debido a las reuniones diarias que permiten la evaluación de avances y la minimización de retrasos. De esta manera se va culminando el Sprint 4 hasta pasar al siguiente Sprint.

La figura N° 3.5 muestra el cuadro burndown o gráfica de progreso para el Sprint backlog 4, el cual tuvo una duración de 20 días.

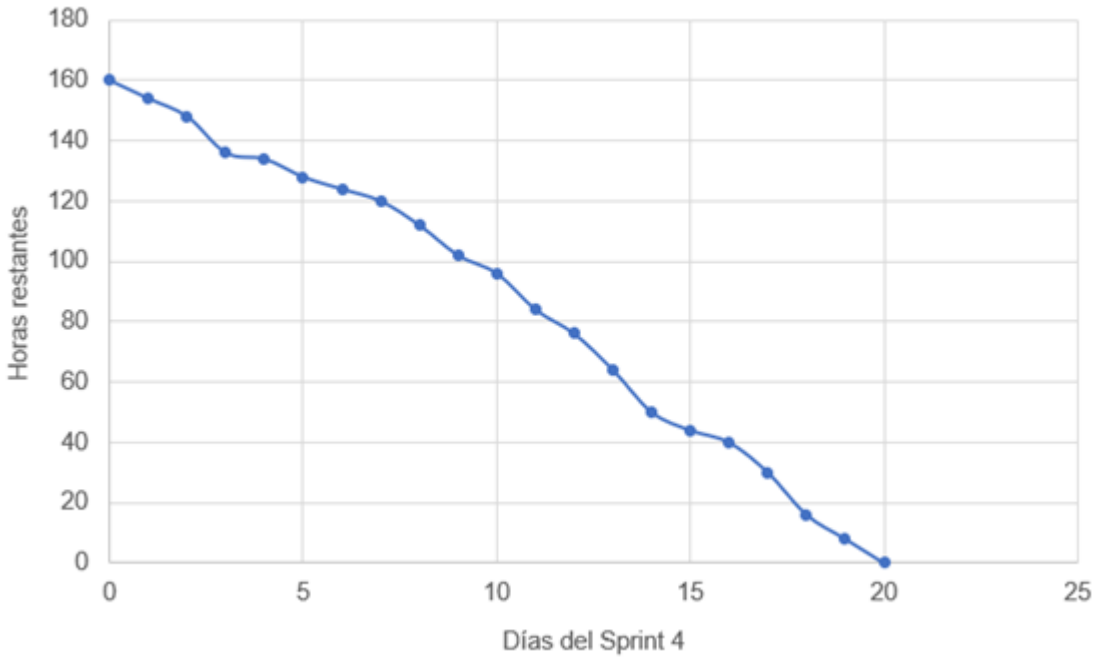


Figura 3.5. Diagrama burndown para el Sprint backlog 4

Fuente: Elaboración propia

Del gráfico se puede observar el comportamiento de la gráfica lineal, la cual tiende al decrecimiento, ello debido a las reuniones diarias que permiten la evaluación de avances y la minimización de retrasos. De esta manera se va culminando el Sprint 4 hasta pasar al siguiente Sprint.

La figura N° 3.6 muestra el cuadro burndown o gráfica de progreso para el Sprint backlog 5, el cual tuvo una duración de 20 días.

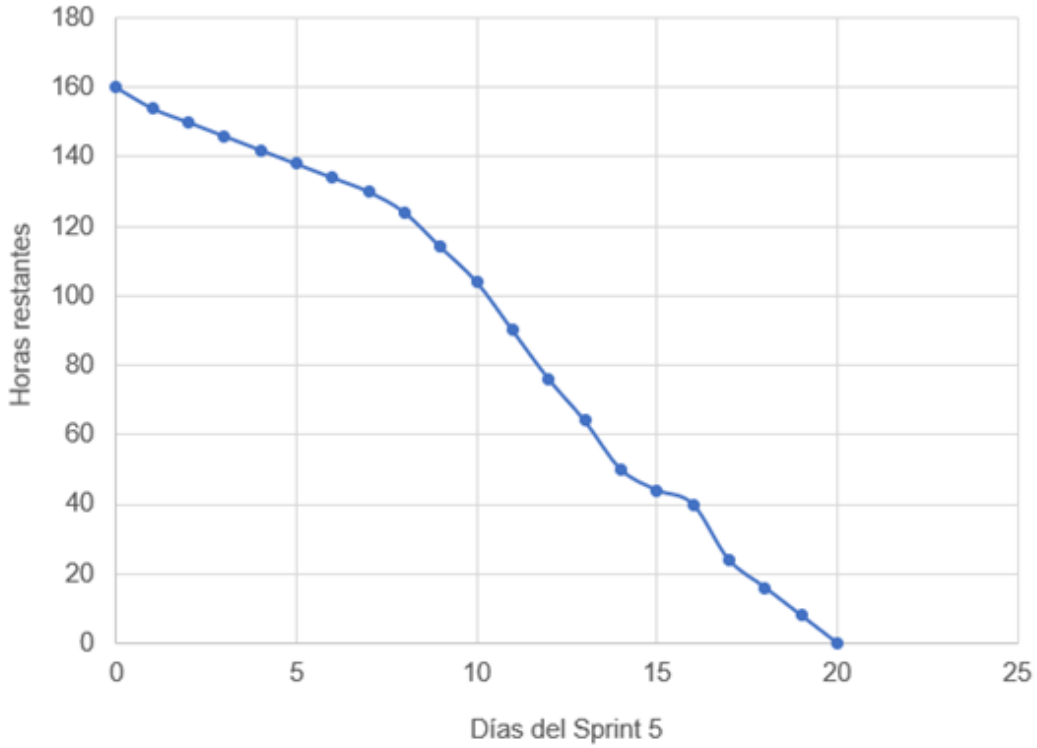


Figura 3.6. Diagrama burndown para el Sprint backlog 5

Fuente: Elaboración propia

Del gráfico se puede observar el comportamiento de la gráfica lineal, la cual tiende al decrecimiento.

3.5 FASE 4: Revisión del Sprint

3.5.1 Planificación de entregas

Los entregables de cada Sprint, se basan inicialmente en el product backlog definido en la primera etapa de la metodología, la Definición del Product Backlog. Asimismo, se basan en las tareas establecidas en el Sprint Backlog, definidas en la segunda etapa de Planificación de Sprints.

3.5.2 Sprint 1

Para el Sprint 1, se tenían las siguientes metas de Sprint:

- Afiliación de una cuenta Billetera
- Inicio de sesión

La primera meta del Sprint 1, realizar el análisis y diseño del proyecto fue concluida en su totalidad sin mayor inconveniente, y en cuanto a la segunda meta del Sprint 1; inicialmente se cometió un error en el modelamiento de la base de datos de nombre BD_Billetera, el cual fue corregido.

La figura N° 3.7, N° 3.8, N° 3.9 y N° 3.10 muestra el diagrama de base de datos producto del modelamiento de datos inicial, realizado en Microsoft SQL Server 2012 R2

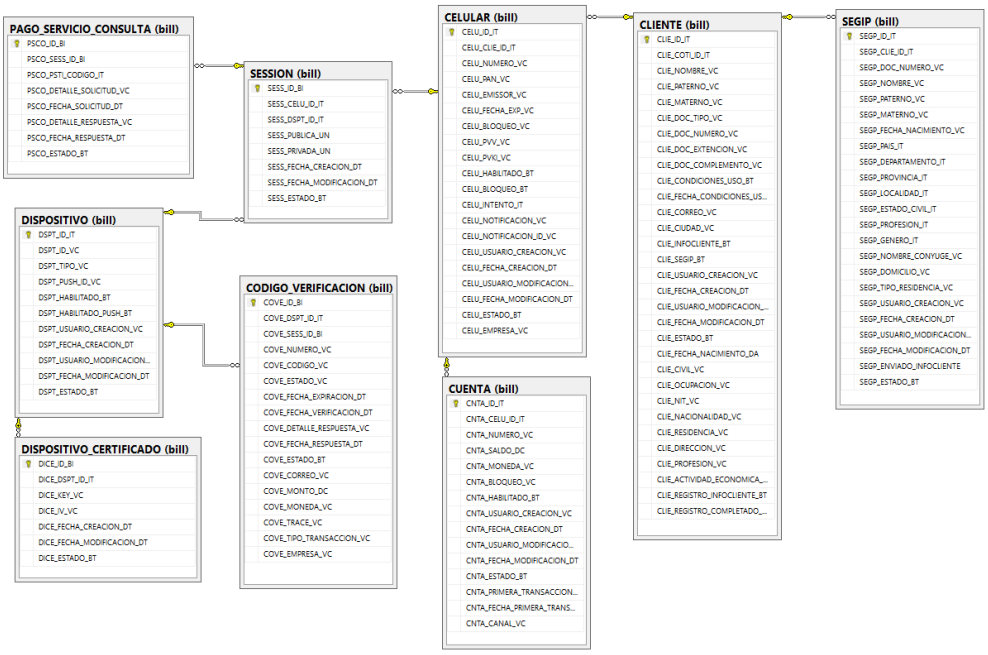


Figura 3.7. Diagrama de base de datos nivel transaccional

Fuente: Elaboración propia

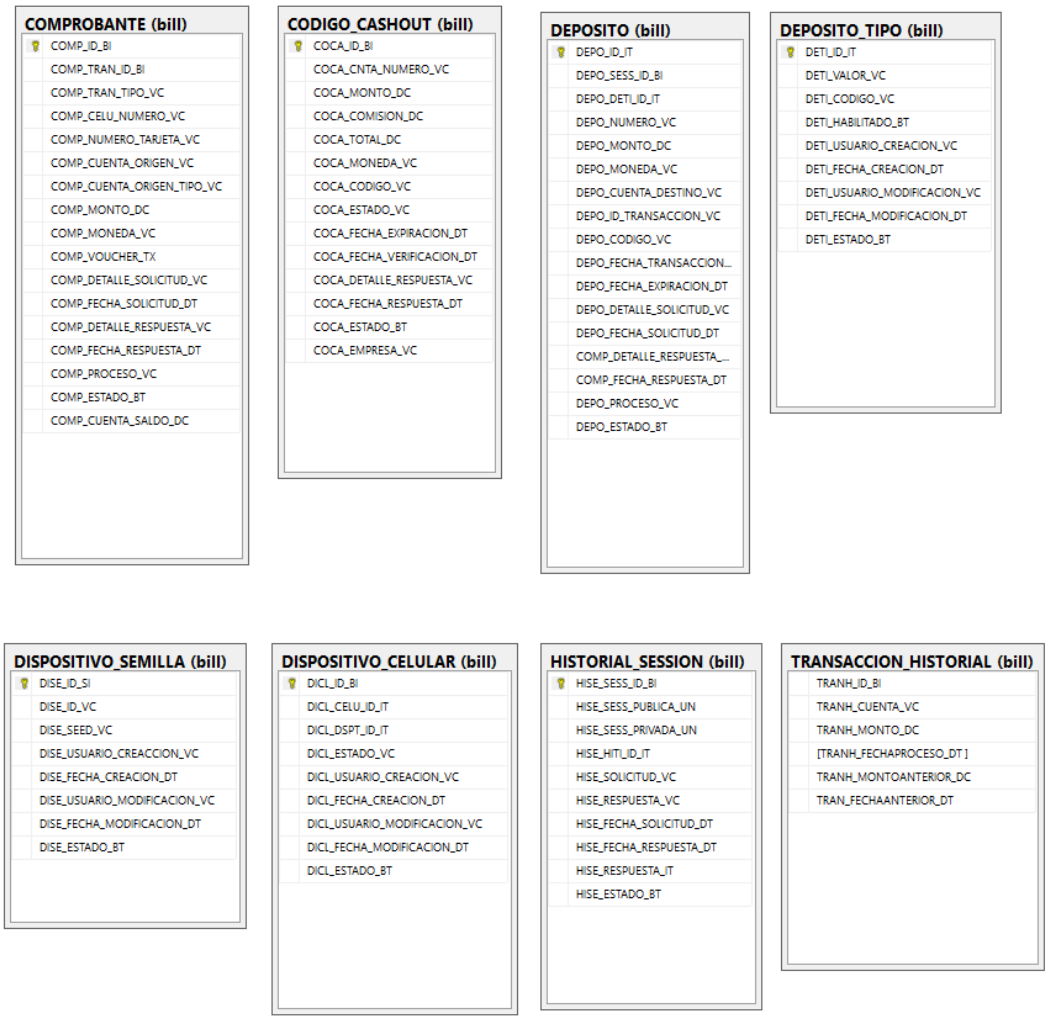


Figura 3.8. Diagrama de base de datos nivel transaccional e historial

Fuente: Elaboración propia

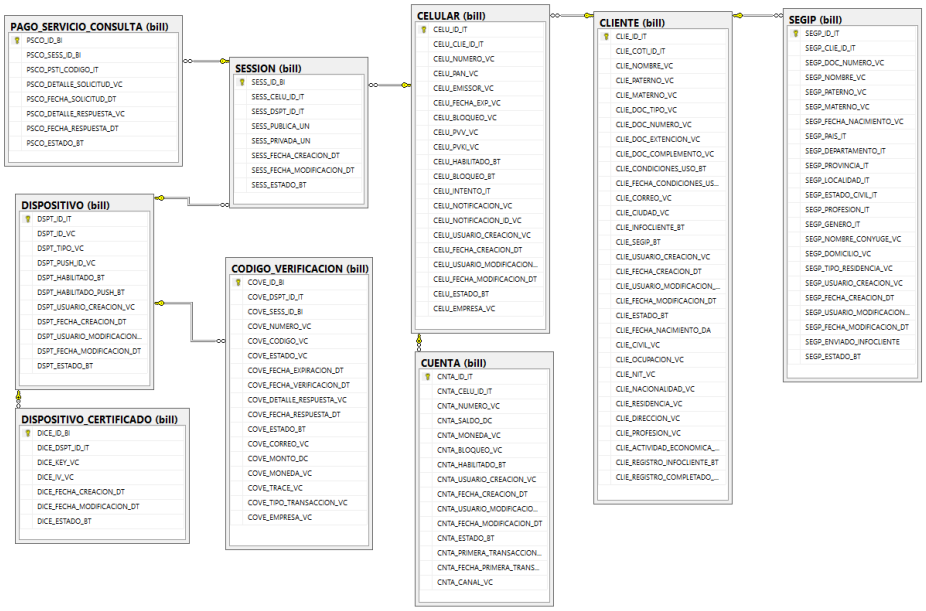


Figura 3.9. Diagrama de base de datos nivel cliente

Fuente: Elaboración propia

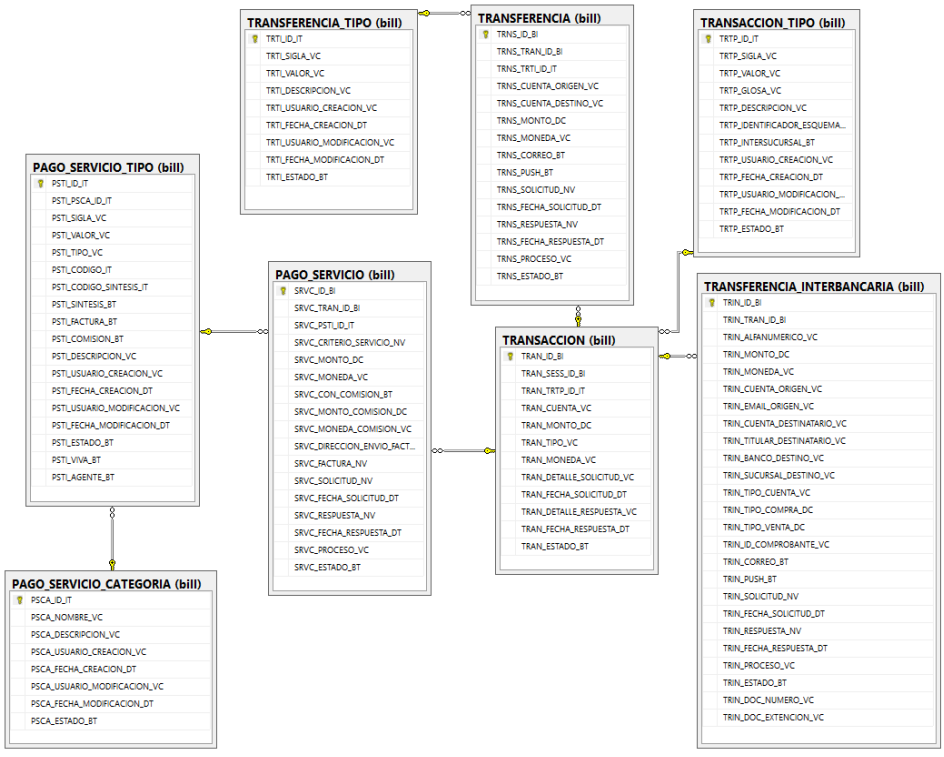
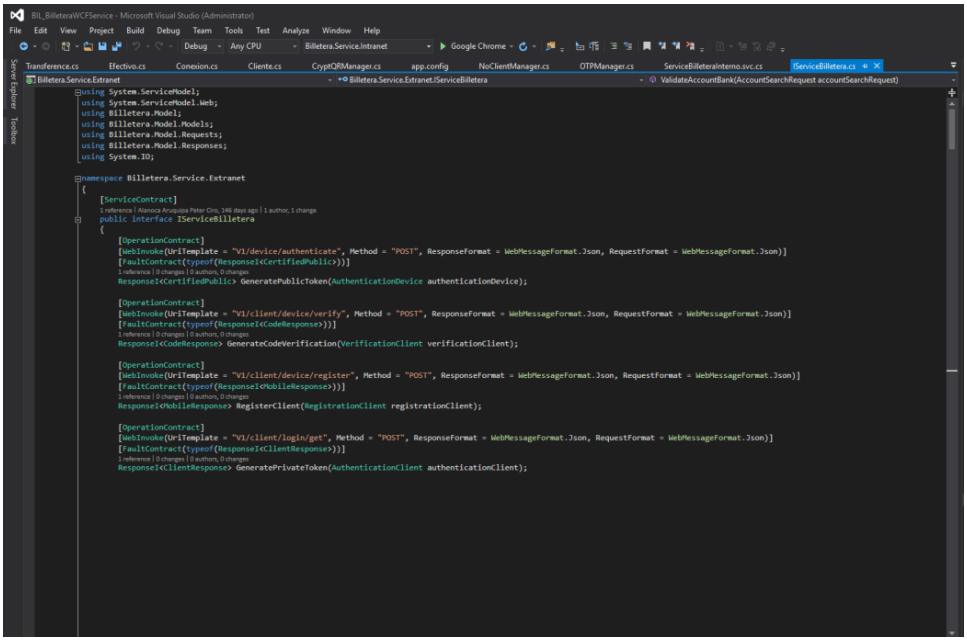


Figura 3.10. Diagrama de base de datos nivel transferencias y pago de servicios

Fuente: Elaboración propia

La figura N° 3.11 muestra los endpoints del inicio de sesión y la afiliación o registro, realizado en Visual Studio.



```
using System.ServiceModel;
using System.ServiceModel.Web;
using Billetera.Model;
using Billetera.Model.Models;
using Billetera.Model.Requests;
using Billetera.Model.Responses;
using System.IO;

namespace Billetera.Service.Extranet
{
    [ServiceContract]
    public interface IServiceBilletera
    {
        [OperationContract]
        [WebInvoke(UriTemplate = "V1/device/authenticate", Method = "POST", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]
        [FaultContract(typeof(ResponseI<CertifiedPublic>))]
        ResponseI<CertifiedPublic> GeneratePublicToken(AuthenticationDevice authenticationDevice);

        [OperationContract]
        [WebInvoke(UriTemplate = "V1/client/device/verify", Method = "POST", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]
        [FaultContract(typeof(ResponseI<CodeResponse>))]
        ResponseI<CodeResponse> GenerateCodeVerification(VerificationClient verificationClient);

        [OperationContract]
        [WebInvoke(UriTemplate = "V1/client/device/register", Method = "POST", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]
        [FaultContract(typeof(ResponseI<MobileResponse>))]
        ResponseI<MobileResponse> RegisterClient(RegistrationClient registrationClient);

        [OperationContract]
        [WebInvoke(UriTemplate = "V1/client/login/get", Method = "POST", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]
        [FaultContract(typeof(ResponseI<ClientResponse>))]
        ResponseI<ClientResponse> GeneratePrivateToken(AuthenticationClient authenticationClient);
```

Figura 3.11. Captura de endpoints del inicio de sesión en Visual Studio

Fuente: Elaboración propia

La figura N° 3.12 muestra la pantalla de inicio de sesión, realizado en Android Studio.

Figura 3.12. Captura de pantalla del inicio de sesión en el emulador de Android Studio

Fuente: Elaboración propia

La figura N° 3.13 muestra las pantallas del registro o afiliación, realizado en Android Studio.

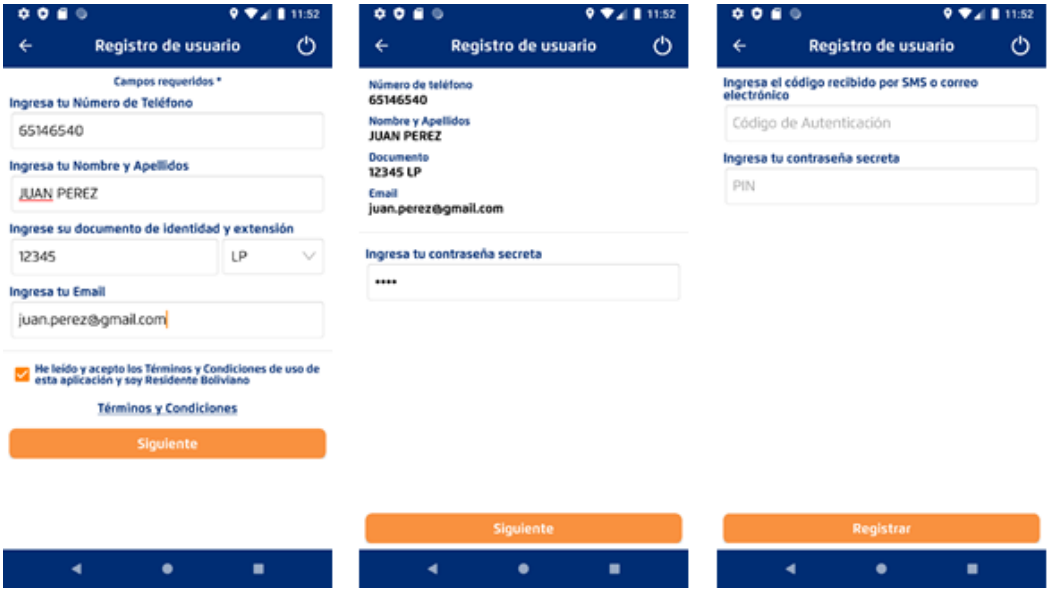


Figura 3.13. Captura de pantalla del registro en el emulador de Android Studio

Fuente: Elaboración propia

3.5.3 Sprint 2

Para el Sprint 2, se tenían las siguientes metas de Sprint:

- Recargas de crédito de telefónicas
- Transferencias entre cuentas billetera

Tanto la primera meta como la segunda meta del Sprint 2 se desarrollaron correctamente, tal como se muestran en las figuras N° 3.14, N° 3.15 y N° 3.16.

La figura N° 3.14 muestra los endpoints de recargas y transferencias entre cuentas billeteras, realizado en Visual Studio.

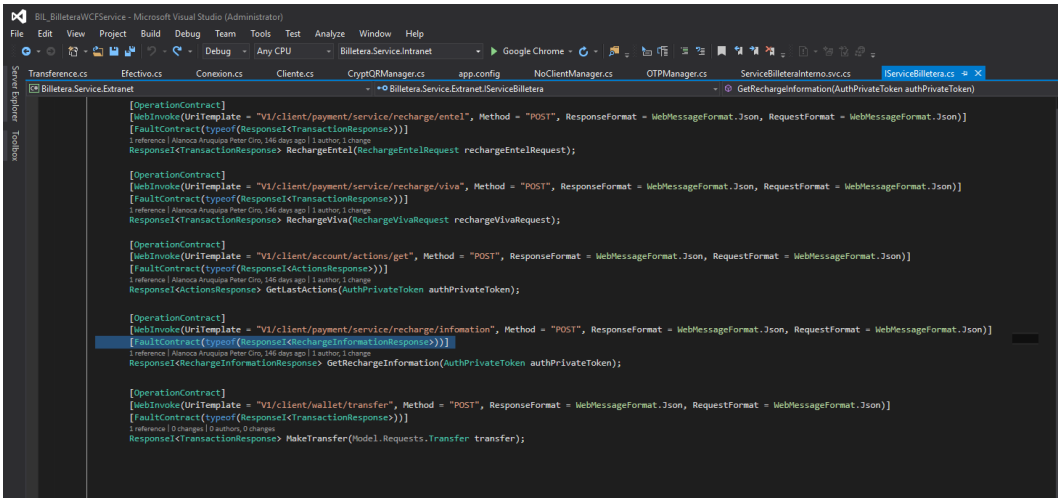


Figura 3.14. Captura de endpoints de recargas y transferencias en Visual Studio

Fuente: Elaboración propia

La figura N° 3.15 muestra las pantallas de transferencias entre cuentas billetera, realizado en Android Studio.

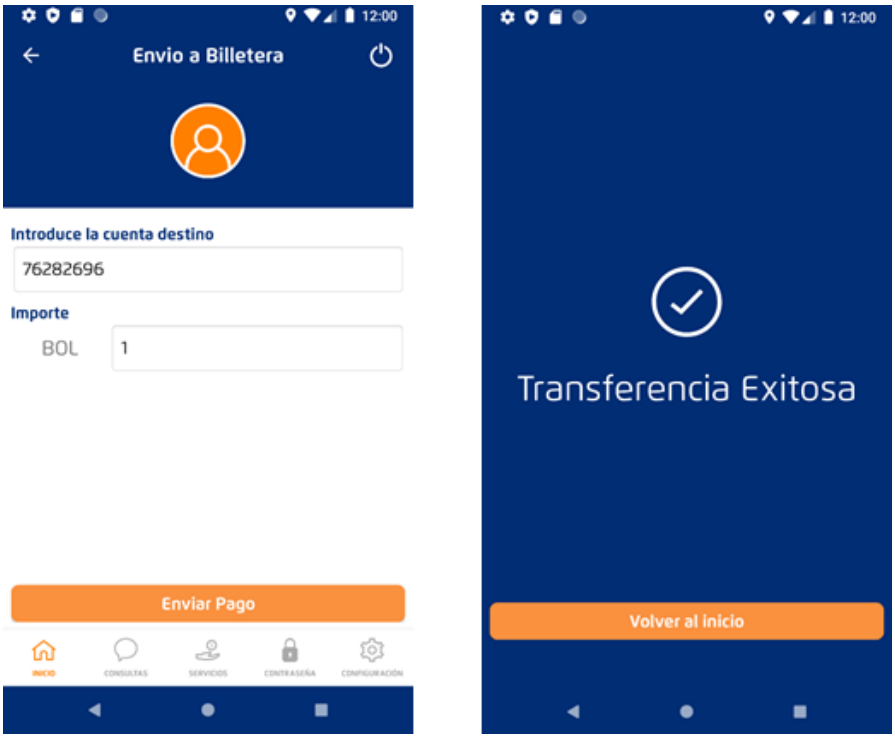


Figura 3.15. Captura de pantalla de transferencia en el emulador de Android Studio

Fuente: Elaboración propia

La figura N° 3.16 muestra las pantallas de recargas, realizado en Android Studio.

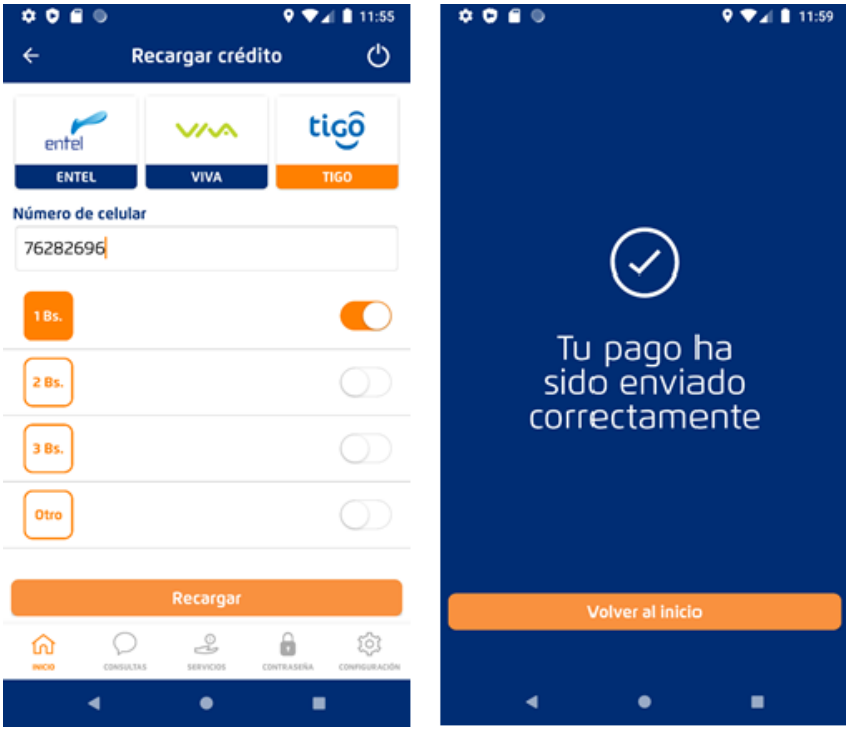


Figura 3.16. Captura de pantalla de recargas en el emulador de Android Studio

Fuente: Elaboración propia

3.5.4 Sprint 3

Para el Sprint 3, se tenían las siguientes metas de Sprint:

- Transferencias a cuentas BCP
- Transferencias a cuentas de otros bancos

Tanto la primera meta como la segunda meta del Sprint 3 se desarrollaron correctamente, tal como se muestra en las figuras N° 3.17, N° 3.18 y N° 3.19.

La figura N° 3.17 muestra los endpoints de transferencias a cuentas BCP y transferencias a cuentas de otros bancos, realizado en Visual Studio.

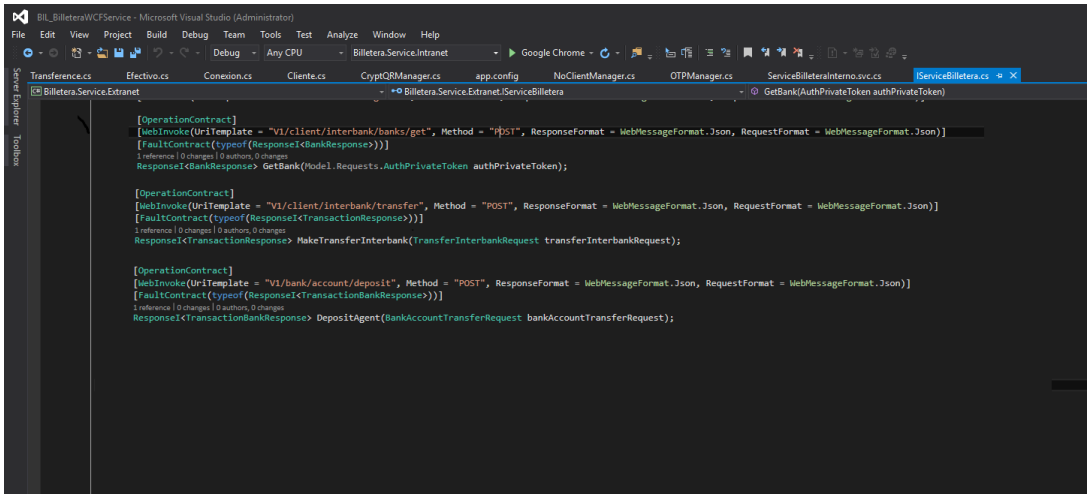


Figura 3.17. Captura de endpoints de transferencias a cuentas BCP en Visual Studio

Fuente: Elaboración propia

La figura N° 3.18 muestra las pantallas de transferencias a cuentas BCP, realizado en Android Studio.

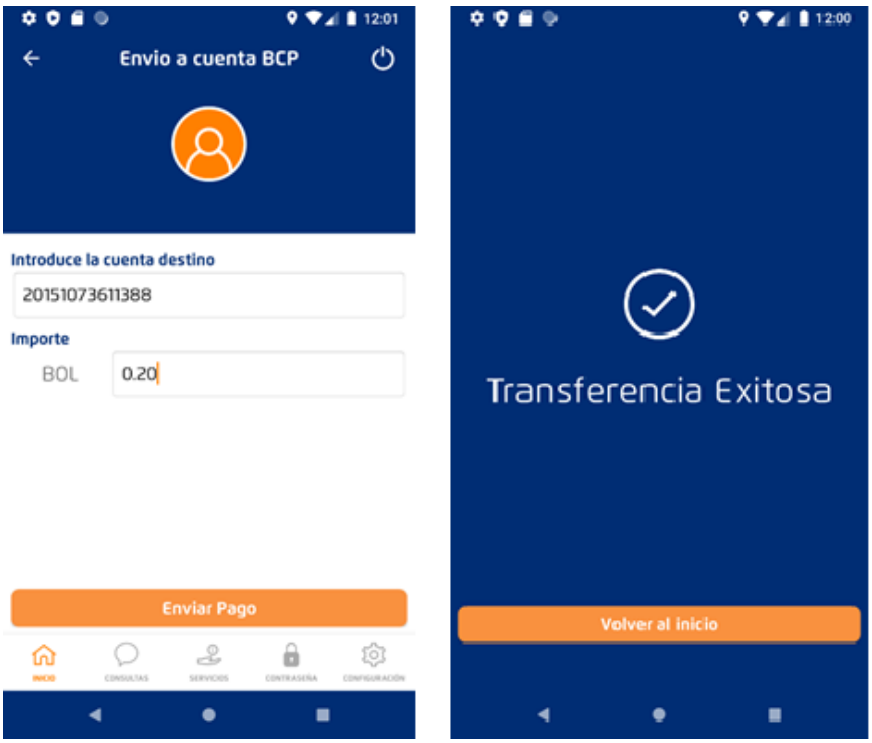


Figura 3.18. Captura de pantalla de transferencias a cuentas BCP en el emulador de Android Studio

Fuente: Elaboración propia

La figura N° 3.19 muestra las pantallas de transferencias a otros bancos, realizado en Android Studio.

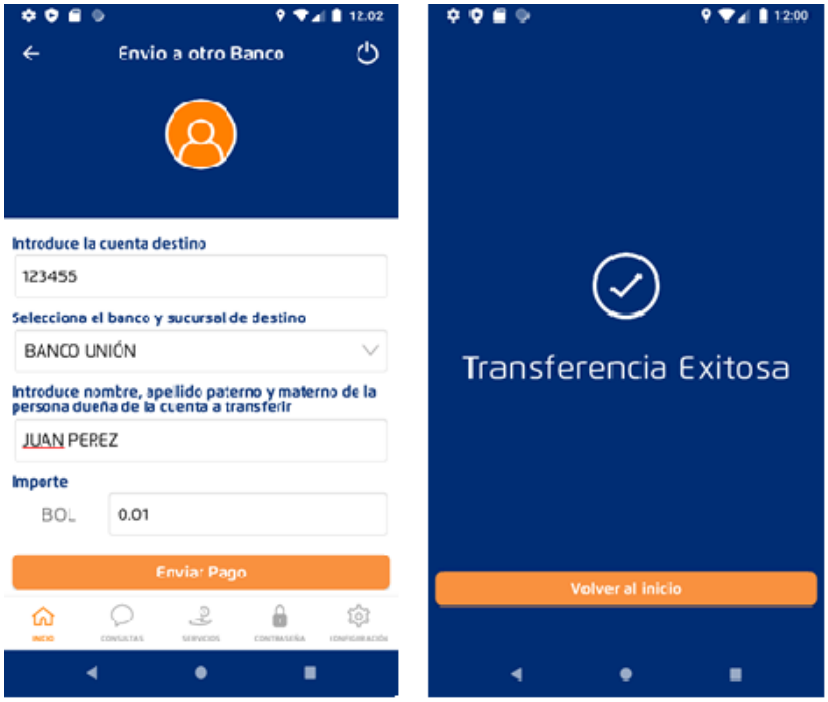


Figura 3.19. Captura de pantalla de transferencias interbancarias en el emulador de Android Studio

Fuente: Elaboración propia

3.5.5 Sprint 4

Para el Sprint 4, se tenían las siguientes metas de Sprint:

- Estados de transferencias y movimientos
- Pago de servicios

Tanto la primera meta como la segunda meta del Sprint 4 se desarrollaron correctamente, tal como se muestra en las figuras N° 3.20, N° 3.21 y N° 3.22.

La figura N° 3.20 muestra los endpoints estados, movimientos y pagos de servicios, realizado en Visual Studio.

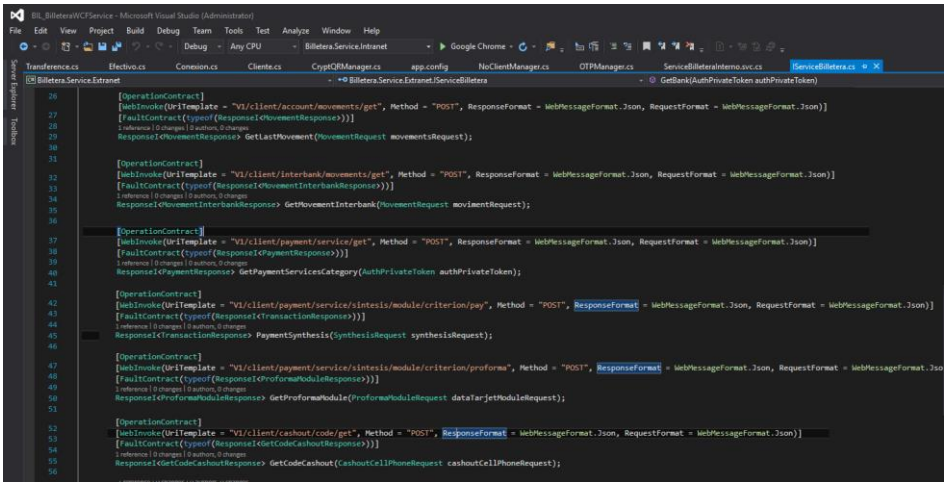


Figura 3.20. Captura de endpoints de movimientos y pago de servicios en Visual Studio

Fuente: Elaboración propia

La figura N° 3.21 muestra las pantallas de estados de transferencias y movimientos, realizado en Android Studio.

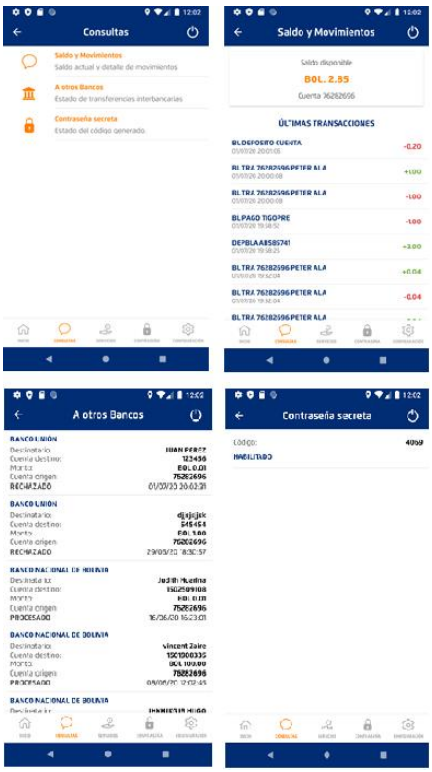


Figura 3.21. Captura de pantalla de movimientos en el emulador de Android Studio

Fuente: Elaboración propia

La figura N° 3.22 muestra las pantallas de pago de servicios, realizado en Android Studio.

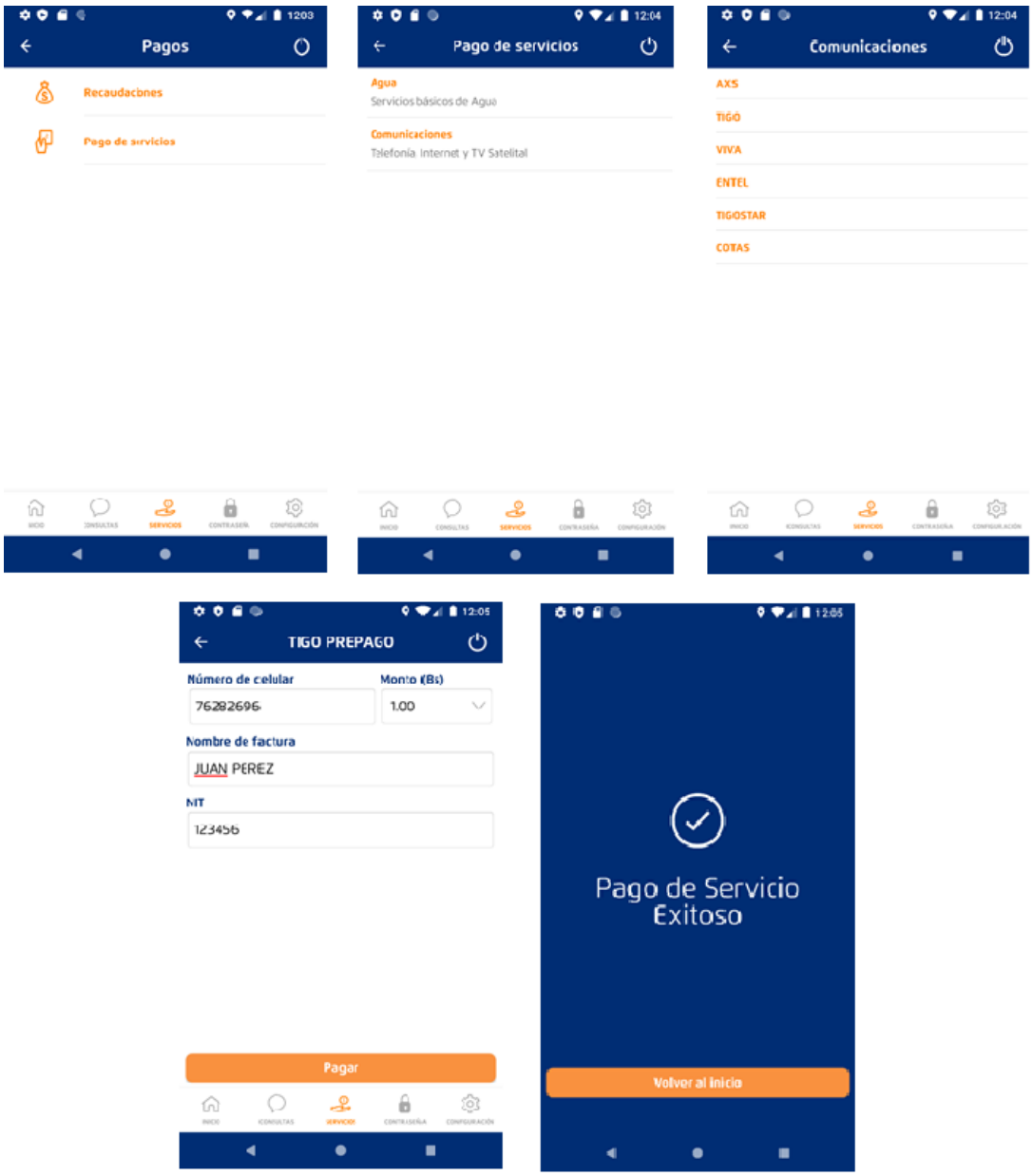


Figura 3.22. Captura de pantalla de pagos de servicios en el emulador de Android Studio

Fuente: Elaboración propia

3.5.6 Sprint 5

Para el Sprint 5, se tenían las siguientes metas de Sprint:

- Retiro en cajeros automáticos y compra en comercios
- Cambio y reseteo de PIN
- Depósitos en cajeros automáticos para no clientes

Tanto la primera meta como la segunda meta del Sprint 5 se desarrollaron correctamente, tal como se muestra en las figuras del N° 3.23 al N° 3.26.

La figura N° 3.23 muestra los endpoints retiros, cambio, reseteo de pin y depositos, realizado en Visual Studio.

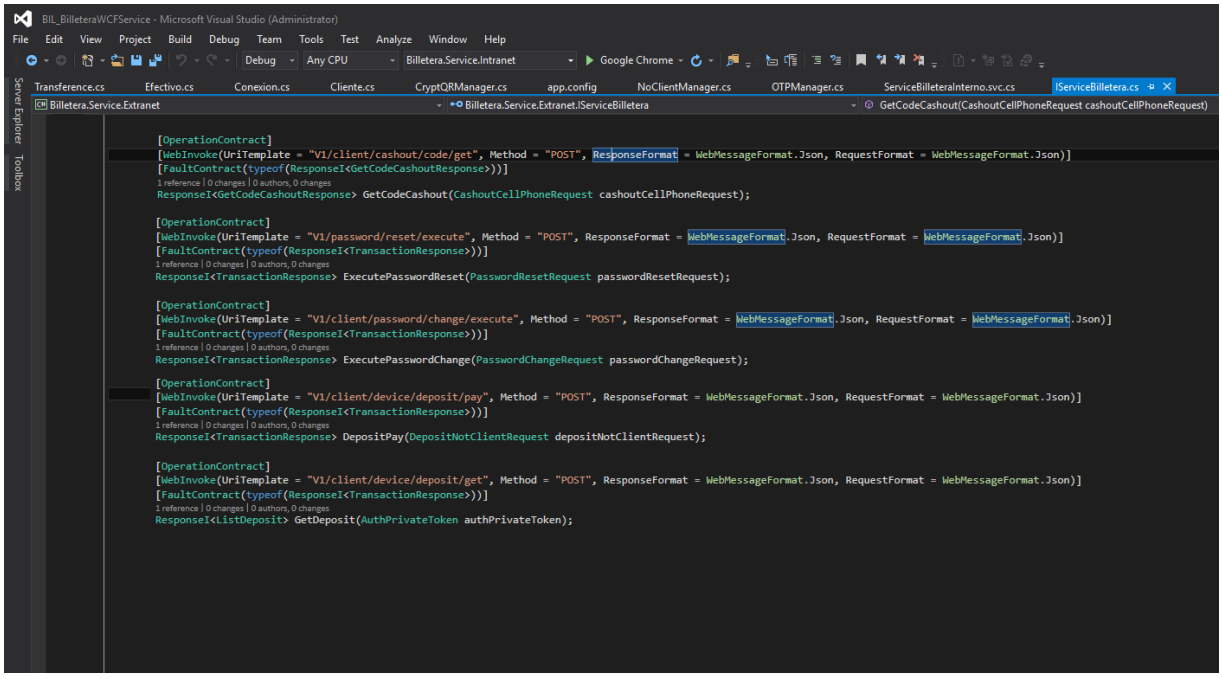


Figura 3.23. Captura de endpoints de depósito y cambio de pin en Visual Studio

Fuente: Elaboración propia

La figura N° 3.24 muestra el cambio de pin, realizado en Android Studio.

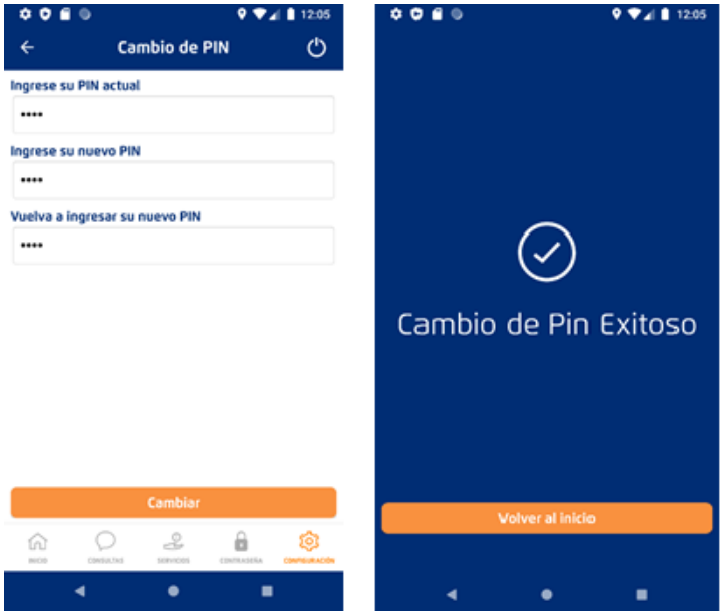


Figura 3.24. Captura de pantalla de cambio de pin en el emulador de Android Studio

Fuente: Elaboración propia

La figura N° 3.25 muestra las pantallas de reseteo de pin, realizado en Android Studio.

Figura 3.25. Captura de pantalla de reseteo de pin en el emulador de Android Studio

Fuente: Elaboración propia

La figura N° 3.26 muestra las pantallas de depósito, realizado en Android Studio.

Figura 3.26. Captura de pantalla de depósito en el emulador de Android Studio

Fuente: Elaboración propia

3.6 FASE 5: Retrospectiva del Sprint

En esta etapa se debe realizar la retrospectiva de cada uno de los tres Sprints definidos en la fase de planificación de Sprints, siempre y cuando el cliente y/o dueño de producto establezca que el entregable proporcionado por el equipo Scrum no es lo que se solicitó al inicio del proyecto.

En el presente estudio las retrospectivas para los Sprint 1, Sprint 2, Sprint 3, Sprint 4 y Sprint 5, fueron satisfactorias.

Finalmente, la tabla N° 3.22 muestra el estado final de los ítems del product backlog.

Id	Nombre	Importancia	Estimación	Como probarlo	Estado Final
1	Afiliación de una cuenta Billetera	80	20	Llenar todos los datos y presionar el botón Registrar	REALIZADO
2	Inicio de sesión	20	20	Llenar todos los datos y presionar el botón Ingresar	REALIZADO
3	Opción de Inicio	80	10	Iniciar sesión será la primera opción visible	REALIZADO
4	Recargas de crédito de telefónicas	40	10	Llenar todos los datos y presionar el botón Recargar	REALIZADO
5	Transferencias entre cuentas billetera	40	10	Llenar todos los datos y presionar el botón Enviar pago	REALIZADO
6	Transferencias a cuentas BCP	40	20	Llenar todos los datos y presionar el botón Enviar pago	REALIZADO
7	Transferencias a cuentas de otros bancos	40	20	Llenar todos los datos y presionar el botón Enviar pago	REALIZADO
8	Estados de transferencias y movimientos	60	20	Seleccionar la opción de Consultas	REALIZADO
9	Pago de servicios	80	20	Seleccionar la opción de Servicios, seleccionar una empresa llenar todos	REALIZADO

				los datos y presionar el botón Pagar	
10	Retiro en cajeros automáticos y compra en comercios	70	10	Seleccionar la opción de Contraseña y presionar el botón Generar	REALIZADO
11	Cambio y reseteo de PIN	40	10	Seleccionar la opción de Cambio de Pin, llenar todos los datos y presionar el botón Cambiar	REALIZADO
12	Depósitos en cajeros automáticos para no clientes	40	10	Seleccionar la opción de Deposito, llenar todos los datos y presionar el botón Depositar	REALIZADO

Tabla 3.22. Estado final de las tareas del Product Backlog del proyecto

Fuente: Elaboración propia

La tabla refleja que la aplicación correcta de Scrum, permitió culminar cada historia de usuario planteada al inicio del proyecto.

Una vez levantadas cada una de las observaciones realizadas por el dueño del producto a los entregables desarrollados por el equipo Scrum, se procede a realizar el cierre del proyecto. Tarea que generalmente le corresponde al Scrum Master en coordinación con el dueño del producto.

La tabla N° 3.23 muestra el estado de las tareas definidas en la etapa de Planificación de Sprint.

Sprint	Responsable	Tareas	Status	Status and Review
SPRINT 1	**Peter Alanoca**	Realizar el análisis y diseño del proyecto	OK	Done
		Realizar el modelamiento de la base de datos.	OK	Done
		Creación de tablas y procedimientos almacenados	OK	Done
		Creación de servicio web	OK	Done
		Creación de los endpoints de registro	OK	Done
	Edson Mamani	Consumir los servicios de envio de sms interno	OK	Done
		Consumir los servicios de Segip	OK	Done
		Creación del endpoint de login	OK	Done

		Consumir el servicio de autenticación interno	OK	Done
		Creación de la interfaz del registro en las aplicaciones móviles	OK	Done
		Creación de la interfaz del login en las aplicaciones móviles	OK	Done
SPRINT 2	**Peter Alanoca**	Creación de tablas y procedimientos almacenados	OK	Done
		Creación del endpoint de inicio	OK	Done
		Creación de los endpoints de recargas	OK	Done
		Consumir los de recargas	OK	Done
		Creación de los endpoints de transferencias entre cuentas billetera	OK	Done
	Edson Mamani	Creación de la interfaz de la opción en las aplicaciones móviles	OK	Done
		Creación de la interfaz de recargas en las aplicaciones móviles	OK	Done
		Creación de la interfaz de transferencias entre cuentas billetera en las aplicaciones móviles	OK	Done
SPRINT 3	**Peter Alanoca**	Creación de tablas y procedimientos almacenados	OK	Done
		Creación de los endpoints de transferencias a cuenta BCP	OK	Done
		Consumir el servicio de abonos a cuentas BCP	OK	Done
		Creación de los endpoints de transferencias a cuentas de otros bancos	OK	Done
	Edson Mamani	Consumir el servicio de abonos a cuentas de otros bancos	OK	Done
		Creación de la interfaz de transferencias a cuentas BCP en las aplicaciones móviles	OK	Done
		Creación de la interfaz de transferencias a de otros en las aplicaciones móviles	OK	Done
SPRINT 4	**Peter Alanoca**	Creación de tablas y procedimientos almacenados	OK	Done
		Creación del endpoint de movimientos	OK	Done
		Creación del endpoint de estado de transferencias	OK	Done
		Consumir el servicio de código secreto para retiro	OK	Done
	Edson Mamani	Creación de los endpoints de pago de servicios	OK	Done
		Consumir el servicio de pago de servicios	OK	Done
		Creación de la interfaz de movimientos y estados de transferencias	OK	Done
		Creación de la interfaz de pagos de servicios en las aplicaciones móviles	OK	Done
SPRINT 5	**Peter Alanoca**	Creación de tablas y procedimientos almacenados	OK	Done
		Creación del endpoint de código secreto para retiro	OK	Done
		Consumir el servicio de código secreto para retiro	OK	Done
		Creación del endpoint de cambio de pin	OK	Done
		Creación del endpoint de reseteo de pin	OK	Done
		Creación del endpoint de depósito para no clientes	OK	Done
	Edson Mamani	Consumir el servicio de depósito para no clientes	OK	Done
		Creación de la interfaz de retiros en cajeros en las aplicaciones móviles	OK	Done
		Creación de la interfaz de cambio de pin	OK	Done
		Creación de la interfaz de reseteo de pin	OK	Done

		Creación de la interfaz de depósitos para no cliente	OK	Done

Tabla 3.23. Cierre del proyecto con Scrum

Fuente: Elaboración propia

La tabla muestra el estado actual de las tareas asignadas a los responsables, las cuales tienen el estado final de “Hecho” (Done) al cierre del proyecto.

En este capítulo se realizó la Intervención Metodológica, siguiendo lineamientos establecidos en el modelo aplicativo. Se aplicó cada una de las fases, partiendo por la primera denominada Definición del backlog del producto, siguiendo por la Planificación del Sprint, después el Scrum diario, continuando con la Revisión del Sprint y finalmente realizando la Retrospectiva del Sprint. Todo este proceso culmina con la producción de un incremento operativo del software verificado y validado por el cliente, obteniéndose una versión terminada del producto (producto esperado).

3.7 Pruebas de Software

Se tratará de encontrar todo posible error durante un proceso antes que se entre en aplicación, mediante las siguientes pruebas realizadas.

3.7.1 Pruebas de Caja Blanca

Esta prueba se aplica en los procesos más relevantes del modelo de pronóstico de demanda y el sistema en cual se realiza su aplicación siendo técnica de prueba del camino básico con grafos, complejidad ciclo matica y derivación de casos de prueba.

De acuerdo a ello la complejidad del flujo es V (G) y puede calcularse en tres alternativas:

Identificar el número de regiones del grafo de flujo, aplicando un segmento de código

fuente.

V(G) = A – N + 2, donde:

A: Es el número de aristas

N: es el número de nodos.

V (G) = P + 1, donde P es números de nodos predicado

PROCESO DE INICIO DE SESION

```
private void sendAuthenticacionClientRequest() {
    String message = "";
    String mobileNumber = editTextPhoneNumber.getText().toString();
    String pin = editTextPin.getText().toString();
    if (!Validation.isNumberCellPhone(mobileNumber)) { 1
    2 message = "Ingresa tu número de Teléfono";
    }
    if (Validation.isEmptyOrNull(pin)) { 3
    4 message = "Ingresa tu PIN";
    }
    if (Validation.isEmptyOrNull(message)) { 5
        authenticationClient = new AuthenticationClient(getAuthToken());
    6 authenticationClient.MobileNumber = mobileNumber;
        authenticationClient.Pin = getCertificate().encryptToBytes(pin);
        authenticationClient.CertifiedId = getCertificate().Id;
        authenticationClient.Version = BuildConfig.VERSION_NAME;
        authenticationClient.Application = BuildConfig.APPLICATION_ID;
        authenticationClient.Send(getContext(), RC_AUTHENTICATION_USER,  iServiceCallback: this);
        buttonLogin.setEnabled(false);
        showDialog(LoadingDialog.newInstance("Iniciando sesión... "),  requestCode: 0);
    } else {
        showToastError(message);
    7 buttonLogin.setEnabled(true);
    }
}
```

Figura 3.27. Captura de pantalla del código fuente de inicio de sesión en Android Studio

Fuente: Elaboración propia

De la figura N. 3.27 podemos generar los siguientes nodos.

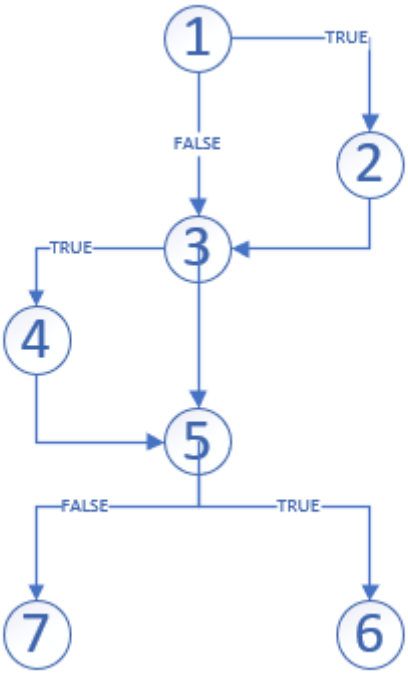


Figura 3.28. Grafo de pruebas de caja blanca inicio de sesión

Fuente: Elaboración propia

Para las pruebas de caja blanca se proporcionará una medición cuantitativa de la complejidad lógica del programa, de la figura N. 3.28 se tomarán las siguientes variables:

A = 8

N = 7

$$V(G) = 8 - 7 + 2 = 3$$

$$V(G) = 3$$

Por lo tanto, tenemos una complejidad ciclo matica de 3 (8 aristas y 7 nodos).

Casos de prueba

Nº	CAMINO	ENTRADA	SALIDA
1	1, 3, 5, 6	MobileNumber = True, Pin = True	Enviar solicitud
2	1, 2, 3, 5, 7	MobileNumber = False, Pin = True	Mostrar mensaje “Ingresa tu número de Teléfono”
3	1, 3, 4, 5, 7	MobileNumber = True, Pin = False	Mostrar mensaje “Ingresa tu PIN”
4	1, 2, 3, 4, 5, 7	MobileNumber = False, Pin = False	Mostrar mensaje “Ingresa tu PIN”

Tabla 3.24. Casos de prueba para pruebas de caja blanca

Fuente: Elaboración propia

CAPITULO IV

CALIDAD Y SEGURIDAD

4.1 Calidad

En esta etapa se aplicará la calidad de software ISO 9126, mismo que propone una jerarquía de atributos de calidad como la de un producto o servicio para satisfacer las necesidades del usuario, que siempre buscará esta cualidad en todos los productos, no solamente de equipos sino también de programas.

El estándar ISO-9126 establece que cualquier componente de la calidad del software puede ser descrito en términos de una o más de seis características básicas, las cuales son: funcionalidad, confiabilidad, usabilidad, eficiencia, mantenibilidad y portabilidad.

4.1.1 Usabilidad

La usabilidad consiste en la evaluación del esfuerzo necesario que el usuario invertirá para usar el sistema, en base a su comprensión y estructura lógica que el sistema tiene.

Esta comprensión por parte de los usuarios con relación al sistema evalúa los siguientes casos:

- Comprensibilidad
- Facilidad de Aprender
- Operabilidad

Se realizan encuestas a los usuarios finales sobre el manejo, la compresión, y la facilidad de aprender el sistema para medir la usabilidad según la siguiente tabla:

Preguntas	Respuestas		Porcentaje %
	SI	NO	
¿El acceso al sistema es complicado?	0	10	100%
¿Son compresibles las respuestas del sistema?	1	9	90%
¿Son complicadas los procesos que realiza el sistema?	10	0	100%
¿El sistema tiene interfaces entendibles?	9	1	90%
¿La interfaz del sistema es agradable a la vista?	10	0	100%
¿Son satisfactorias las respuestas que el sistema devuelve?	8	2	80%
¿El sistema reduce su tiempo de trabajo?	9	1	90%
¿Es difícil aprender a manejar el sistema?	1	9	90%
¿El sistema satisface las necesidades que usted requiere?	9	1	90%
¿Utiliza el sistema con facilidad?	9	1	90%
PROMEDIO			92%

Tabla 4.1. Encuesta de Usabilidad del Sistema

Fuente: Elaboración propia

4.1.2 Mantenibilidad

Para hallar la mantenibilidad del sistema se utiliza el índice de madurez de software (IMS), que proporciona una indicación de la estabilidad de un producto de software (basado en los cambios que ocurren con cada versión del producto).

Se determina la siguiente fórmula para hallar el (IMS):

$$IMS = \frac{Mt - (Fc + Fa + FE)}{Mt}$$

Donde:

Mt: Numero de módulos total de la versión actual.

Fc: Numero de módulos de la versión actual que se modificaron.

Fa: Numero de módulos de la versión actual que se agregaron.

FE: Numero de módulos de la versión anterior que se eliminaron.

$$IMS = \frac{12 - (1 + 0 + 0)}{12}$$

$$IMS = 91.6\%$$

El resultado de 91.6% indica que el sistema no requiere de mantenimiento inmediato ni a corto plazo.

4.1.3 Funcionalidad

La funcionalidad examina si el sistema satisface los requisitos funcionales esperados. El objetivo es revelar problemas y errores en lo que concierne a la funcionalidad del sistema y su conformidad al comportamiento, expresado o deseado por el usuario.

Haremos uso de cinco características de dominios de información y se proporcionan las cuentas en la posición apropiada de la tabla. Los valores de los dominios de información, y se definen de la siguiente manera:

- **Número de entradas de usuario.** Se cuenta cada entrada de usuario que proporciona diferentes datos orientados a la aplicación. Las entradas se deberían diferenciar de las peticiones, las cuales se cuentan de forma separada.
- **Número de entradas de usuario.** Se cuenta cada entrada de usuario que proporciona diferentes datos orientados a la aplicación. Las entradas se deberían diferenciar de las peticiones, las cuales se cuentan de forma separada.
- **Número de salidas de usuario.** En este contexto la salida de usuario se refiere a informes, pantallas, mensajes de error, etc. Los elementos de datos particulares dentro de un informe no se cuentan de forma separada.

- **Número de peticiones de usuario.** Una petición se define como una entrada interactiva que produce la generación de alguna respuesta del software inmediata en forma de salida interactiva. Se cuenta cada petición por separado.
- **Número de archivos.** Se cuenta cada archivo maestro lógico (se refiere a un grupo lógico de datos que puede ser parte de una gran base de datos o un archivo independiente).
- **Número de interfaces externas.** Se cuentan todas las interfaces legibles por la maquina (por ejemplo: archivos de datos de disco) que se utilizan para transmitir información a oro sistema [Pressman, 2005].

Parámetros de entrada	Cuenta
Número de entradas de usuario	2
Número de salidas de usuario	2
Número de peticiones de usuario	3
Número de archivos	1
Número de interfaces externas	0

Tabla 4.2. Entradas para el cálculo de funcionalidad según punto de fusión

Fuente: Elaboración propia

Los puntos fusión se calculan con la ayuda de la tabla 4.2 considerando los factores de ponderación medio.

Parámetros de entrada	Cuenta	Factor Ponderación Medio	Totales
Número de entradas de usuario	2	8	16

Número de salidas de usuario	2	8	16
Número de peticiones de usuario	3	4	12
Número de archivos	1	10	10
Número de interfaces externas	0	7	0

Tabla 4.3. Cuenta Total con Factor de Ponderación Medio

Fuente: Elaboración propia

Calcularemos la relación para calcular el punto de fusión:

PF = CUENTA TOTAL * (Grado de Confiabilidad + Tasa de Error * ∑ F)

Donde:

PF = Medida de Confiabilidad

CUENTA TOTAL = Es la suma del valor de las entradas, salidas, peticiones, interfaces externas y archivos.

Grado de Confiabilidad = Confiabilidad estimada del sistema.

Tasa de Error = Probabilidad subjetiva estimada del dominio de la información este error es del 1%

F = Son los valores de ajuste de complejidad que toman los valores de la tabla 4.4 que dan respuesta a la tabla 4.3.

IMPORTANCIA	SIN IMPORTANCIA	INCREMENTAL	MODERADO	MEDIO	SIGNIFICATIVO	ESENCIAL
1.- ¿Requiere el sistema copias de seguridad y es de fácil recuperación?						X
2.- ¿Se requiere comunicación de datos?						X
3. ¿Existe funciones de procesos distribuidos?					X	
4. ¿Es critico el rendimiento?					X	
5.- ¿El sistema web será ejecutado en un sistema operativo actual?					X	
6.- ¿Se requiere una entrada interactiva para el sistema?					X	
7.- ¿Se requiere que el sistema tenga entradas a datos con múltiples ventanas?				X		
8.- ¿Se actualiza los archivos de forma interactiva?			X			
9.- ¿Son complejas las entradas, salidas, los archivos o las peticiones?				X		
10.- ¿Es complejo el procesamiento interno del sistema?					X	
11.- ¿se ha diseñado el código para ser reutilizado?				X		
12.- ¿Se ha diseñado el sistema para facilitar al usuario el trabajo?					X	
Cuenta Total $\sum(fi)$ =	45					

Tabla 4.4. Ajuste del factor de complejidad

Fuente: Elaboración propia

Hallamos el punto de valor máximo para comparar los valores del sistema.

PF = CUENTA TOTAL * (Grado de Confiabilidad + Tasa de Error * ∑ F)

$$PF = 54 * (0{,}91 + 0{,}01 \times 45)$$

$$PF = 73{,}44$$

El punto función ideal al 100 % de los factores en la ecuación tiene el valor de 48, a continuación, se halla el PF para obtener la estimación con el resultado antes obtenido

$$PF = 54 * (0.91 + 0{,}01 \times 48)$$

$$PF = 75{,}06$$

Con ambos resultados se calcula el porcentaje de funcionalidad del sistema:

$$PF\ real = \frac{PF\ obtenida}{PF\ ideal} \times 100$$

$$PF\ real = \frac{48{,}96}{50{,}04} \times 100$$

$$PF\ real = \ 97{,}84\%$$

El resultado obtenido muestra que el sistema tiene una funcionalidad o utilidad del 97,84% para la organización, lo que indica que él es sistema cumple con los requisitos funcionales de forma satisfactoria.

4.1.4 Confiabilidad

Aquí se agrupan un conjunto de atributos que se refieren a la capacidad del software de mantener su nivel de ejecución bajo condiciones normales en un periodo de tiempo establecido. [Valle, 2009]

La función siguiente muestra el nivel de confiabilidad del sistema:

$$\boldsymbol{F(t) = (Funcionalidad) * e^{-\lambda t}}$$

Se observa el trabajo hasta que se observa un fallo en un instante t, la función es la siguiente.

Probabilidad de hallar una falla: $\boldsymbol{P(T \leq t) = F(t)}$

Probabilidad de hallar una falla: $\boldsymbol{P(T > t) = 1 - F(t)}$

Valor de Funcionalidad previo = **97,84%**

λ = 0.01 (es decir 1 error en cada 6 ejecuciones)

$t = 6\ meses$

Hallamos la confiabilidad del sistema

$$F(12) = 97{,}84 \times \ \mathrm{e}^{-\frac{1}{6}\times 12}$$

$$F(12) = 13{,}24\%$$

La probabilidad de hallar una falla es de un 13,24% durante los próximos 12 meses.

Por lo tanto, la probabilidad de no hallar un error es del 86.76%

4.1.5 Portabilidad

La portabilidad de refiere a la habilidad del software de ser transferido de un ambiente a otro, se considera los siguientes aspectos.

- **Adaptabilidad.** Aquí se evalúa la capacidad de adaptar el software a diferentes ambientes sin la necesidad de aplicarle modificaciones.
- **Facilidad de instalación.** Es el esfuerzo necesario para instalar el software en un ambiente determinado.
- **Conformidad**. Permite evaluar si el sistema se adhiere a estándares o convenciones relativas a portabilidad.
- **Capacidad de reemplazo.** Hace referencia a la oportunidad y el esfuerzo usado en sustituir el software por otro producto con funciones similares.

Se puede medir en los siguientes niveles:

Nivel de Hardware

- **Tamaño pantalla**

Tablet: 10.1" (255.8mm)

Smartphone: >= 5’’ (por comodidad del usuario)

- **Velocidad CPU**

  Quad Core 1.2GHz

- **Conexión**

  Wi-Fi y 4G (servicio de datos) para poder trabajar con servicio de datos.

Nivel de Software

- **Versión Android**

  5.X, preferiblemente con Android 6 como mínimo.

4.1.6 Resultados

El factor de calidad total está directamente relacionado con el grado de satisfacción con el usuario que ingresa al sistema.

Características	Resultado
Usabilidad	92%
Funcionalidad	97,84%
Mantenibilidad	91,6%
Confiabilidad	86.76%
Evaluación Total de Calidad	**92.05%**

Tabla 4.5. Resultado de Evaluación de Calidad

Fuente: Elaboración propia

4.2 Seguridad

El crecimiento exponencial que ha tenido el Internet, trae consigo amenazas a la privacidad de información. Mientras más se conecta el mundo, la necesidad de seguridad en los procedimientos utilizados para compartir la información toma mayor notoriedad y se vuelve más importante.

Por lo tanto, para el proyecto se tomó las siguientes configuraciones:

4.2.1 Seguridad de la capa de transporte

Seguridad de la capa de transporte (en inglés: Transport Layer Security o TLS) y su antecesor Secure Sockets Layer (SSL; en español capa de puertos seguros) son protocolos criptográficos, que proporcionan comunicaciones seguras por una red, comúnmente Internet.

Se usan certificados X.509 y por lo tanto criptografía asimétrica para autentificar a la contraparte con quien se están comunicando, y para intercambiar una llave simétrica. Esta sesión es luego usada para cifrar el flujo de datos entre las partes. Esto permite la confidencialidad del dato/mensaje, códigos de autenticación de mensajes para integridad y como un producto lateral, autenticación del mensaje. Varias versiones del protocolo están en aplicaciones ampliamente utilizadas como navegación web, correo electrónico, fax por Internet, mensajería instantánea y voz-sobre-IP (VoIP). Una propiedad importante en este contexto es forward secrecy, para que la clave de corta vida de la sesión no pueda ser descubierta a partir de la clave asimétrica de largo plazo (Wikipedia, https://es.wikipedia.org/wiki/Seguridad_de_la_capa_de_transporte).

Por lo tanto, para el proyecto se decidió usar TLS 1.2 por su robustez en los algoritmos

de encriptación y validación de certificados.

4.2.2 SSL PINNING

En un caso de uso de SSL típico, el servidor se configura con un certificado que contiene una clave pública y una privada coincidente. Como parte del protocolo de enlace entre un servidor y un cliente SSL, el servidor confirma que tiene la clave privada mediante la firma de su certificado con criptografía de clave pública.

No obstante, cualquiera puede generar claves privadas y certificados propios, de modo que un protocolo de enlace simple no comprueba respecto del servidor más que el hecho de que este último reconoce la clave privada que coincide con la clave pública del certificado. Una manera de resolver este problema es exigir que el cliente tenga un conjunto de uno o más certificados en los que confíe. Si el certificado no se encuentra en ese conjunto, no se confiará en el servidor.

Este enfoque simple tiene varias desventajas. Los servidores deberían poder actualizar a claves más seguras con el tiempo ("rotación de claves"), que reemplacen la clave pública en el certificado por una nueva. Lamentablemente, ahora se debe actualizar la app cliente debido a lo que en esencia representa un cambio de configuración del servidor, lo cual resulta problemático si el servidor no está bajo el control del desarrollador de la app; por ejemplo, si se trata de un servicio web de terceros. Este enfoque también presenta problemas si la app debe comunicarse con servidores arbitrarios, como un navegador web o una app de correo electrónico (Android Developers, https://developer.android.com/training/articles/security-ssl).

4.2.3 Ofuscación del código

La ofuscación de código consiste en reordenar o alterar las instrucciones de un programa para que, aunque realice la misma función, sea más difícil su comprensión. En consecuencia, la ofuscación es la herramienta más importante para evitar la ingeniería inversa.

Para que tu app sea lo más pequeña posible, debes habilitar la reducción en tu compilación de lanzamiento a fin de quitar el código y los recursos que no se usan. Cuando habilitas esta opción, también te beneficias de la ofuscación, que acorta los nombres de las clases y los miembros de tu app, y la optimización, que aplica estrategias más agresivas a fin de reducir aún más el tamaño de tu app. En esta página, se describe cómo R8 realiza estas tareas en tiempo de compilación en tu proyecto y cómo puedes personalizarlas (Android Developers, https://developer.android.com/studio/build/shrink-code).

4.2.4 Seguridad de Base de Datos

Para mejorar la seguridad en Microsoft SQL Server proporcionando a los usuarios de bases de datos sólo los privilegios requeridos para acceder a los objetos de la base de datos. Puede eliminar los permisos de administración de scripts de base de datos y colocar permisos en objetos de base de datos (Microsoft, https://docs.microsoft.com/es-es/dotnet/framework/data/adonet/sql/authorization-and-permissions-in-sql-server).

Principio de los privilegios mínimos

Desarrollar una aplicación utilizando un enfoque basado en cuenta de usuario de privilegios mínimos (LUA) constituye una parte importante de una estrategia de defensa

exhaustiva contra las amenazas a la seguridad. El enfoque LUA garantiza que los usuarios siguen el principio de los privilegios mínimos y siempre inician sesión con cuentas de usuario limitadas. Las tareas administrativas se realizan utilizando roles fijos del servidor y el uso del rol fijo del servidor sysadmin es muy restringido.

Cuando conceda permisos a usuarios de base de datos, siga siempre el principio de los privilegios mínimos. Otorgue a usuarios y roles los mínimos permisos necesarios para que puedan realizar una tarea concreta.

Permisos basados en roles

Otorgar permisos a roles en lugar de a usuarios simplifica la administración de la seguridad. Los conjuntos de permisos asignados a roles los heredan todos los miembros del rol. Es más fácil agregar o quitar usuarios de un rol que volver a crear conjuntos de permisos distintos para cada usuario. Los roles se pueden anidar. Sin embargo, la existencia de demasiados niveles de anidamiento puede reducir el rendimiento. También puede agregar usuarios a roles fijos de bases de datos para simplificar los permisos de asignación.

Puede conceder permisos en el nivel de esquema. Los usuarios heredan automáticamente los permisos en todos los objetos nuevos creados en el esquema; no es necesario otorgar permisos cuando se crean objetos nuevos.

CAPITULO V

ANALISIS DE COSTOS

5.1 Cálculo de costos

Para realizar la estimación de costos del software desarrollado se utiliza el método algorítmico de aproximación COCOMO II, orientado a los puntos de función ajustados. Para este fin, en primer lugar, se calcula el punto función.

Datos presentes por puntos de función ya obtenidos:

- Puntos de función sin ajustar: $\boldsymbol{PFSA}$ **= 54**
- Punto de función ajustado $\boldsymbol{PFA}$ **= 73,44**

Cálculo de esfuerzo

Se obtiene mediante la formula

$$LDC = PFA \times (\text{factor } \frac{LDF}{PF})$$

Donde:

- **LDC** = Líneas de Código
- **PFA** = Puntos de función ajustados
- **Factor LCD/PF** = Factor Líneas de código por punto función

Lenguaje	Factor LDC/PF
C	128
Java	**53**
Lenguajes OO	19
C++	53
JavaScript	47

Tabla 5.1. Tabla de conversión factor LDC/PF

Fuente: http://www.qsm.com/resources/function-point-languages-table

Utilizamos el factor para el lenguaje orientado a objetos y reemplazamos los valores.

$$LDC = 73{,}44 \times 53$$

$$LDC = 3892{,}32$$

Posterior a ello se convierte LDC a KLDC dividiendo el resultado entre 1000.

$$KLDC = \frac{3892{,}32}{1000}$$

$$KLDC = 3{,}89$$

Así pues, en nuestro caso el tipo orgánico será el más apropiado ya que el número de líneas de código es menor a los 50 KLDC, y además el proyecto no es muy complicado, por consiguiente, los coeficientes que usaremos serán las siguientes:

Proyecto Software	a	e	c	d
Organico	3.2	1,05	2,5	0,38
Semi-acoplado	3,0	1,12	2,5	0,35
Empotrado	2,8	1,20	2,5	0,32

Tabla 5.2. Modelo COCOMO

Fuente: Pressman, 1999

Calcular la variable FAE

CONDUCTORES DE COSTE	VALORACIÓN					
	Muy bajo	Bajo	Nominal	Alto	Muy alto	Extra alto
Fiabilidad requerida del software	0,75	0,88	1,00	**1,15**	1,40	-
Tamaño de la base de datos	-	0,94	1,00	**1,08**	1,16	-
Complejidad del producto	0,70	0,85	1,00	**1,15**	1,30	1,65
Restricciones del tiempo de ejecución	-	-	1,00	**1,11**	1,30	1,66
Restricciones del almacenamiento principal	-	-	**1,00**	1,06	1,21	1,56
Volatilidad de la máquina virtual	-	0,87	**1,00**	1,15	1,30	-

Tiempo de respuesta del ordenador	-	0,87	1,00	**1,07**	1,15	-
Capacidad del analista	1,46	1,19	1,00	**0,86**	0,71	-
Experiencia en la aplicación	1,29	1,13	1,00	0,91	**0,82**	-
Capacidad de los programadores	1,42	1,17	1,00	0,86	**0,70**	-
Experiencia en S.O. utilizado	1,21	1,10	1,00	**0,90**	-	-
Experiencia en el lenguaje de programación	1,14	1,07	1,00	**0,95**	-	-
Prácticas de programación modernas	1,24	1,10	1,00	**0,91**	0,82	-
Utilización de herramientas software	1,24	1,10	**1,00**	0,91	0,83	-
Limitaciones de planificación del proyecto	1,23	1,08	**1,00**	1,04	1,10	-

Tabla 5.3. Ajuste del factor de complejidad

Fuente: Pressman, 1999

$$FAE = 1{,}55 * 1{,}08 * 1{,}15 * 1{,}11 * 1 * 1 * 1{,}07 * 0{,}86 * 0{,}82 * 0{,}70 * 0{,}90 * 0{,}95 * 0{,}91 * 1 * 1$$

$$FAE = 0{,}6$$

5.2 Cálculo del Esfuerzo de Desarrollo

$$E = a \times (KLDC)^e \times FAE = 3{,}2 \times 3{,}89^{1{,}05} \times 0{,}6$$

$$E = 7{,}99\ personas\ /mes$$

5.3 Cálculo del Tiempo de Desarrollo

$$T = c(Esfuerzo)^d = 2{,}5 \times 7{,}99^{0.38}$$

$$T = 5\ meses\ Es\ el\ tiempo\ que\ tomará\ desarrollar\ el\ proyecto$$

5.4 Cálculo de Productividad

$$PR = \frac{LDC}{Esfuerzo}$$

$$PR = \frac{3892{,}32}{7{,}99}$$

$$PR = 487{,}15 \frac{LDC}{Persona} mes$$

5.5 Cálculo de Personal Promedio

$$P = \frac{E}{T}$$

$$P = \frac{7{,}99}{5}$$

$$P = 1{,}6$$

Según estas cifras será necesario un equipo de 2 personas trabajando alrededor de 5 meses.

5.6 Costo de desarrollo

Para el presente proyecto se ha establecido el salario de un desarrollador el cuál es 5000 Bs.

El costo total de desarrollo del proyecto, se calcula mediante:

$$Costo\ de\ desarrollo = nro.de\ programadores * Tiempo\ estimado * Salario$$

$$Costo\ de\ desarrollo = 2 * 5 * 5000$$

$$Costo\ de\ desarrollo = 50000\ Bolivianos$$

Por tanto, el costo total será de 50000 bolivianos.

CAPITULO VI

CONCLUSIONES Y RECOMENDACIONES

6.1 Conclusiones

Habiendo cumplido los requerimientos establecidos por la empresa, se ha logrado alcanzar el objetivo planteado por haber terminado el desarrollo de una aplicación android que resuelve los problemas de servicios financieros al Banco de Crédito “BCP”.

Por tanto, se llegan las siguientes conclusiones según a los objetivos específicos y general:

- Se logro desarrollar una aplicación android que resuelve los problemas de servicios financieros a personas que no tienen una cuenta bancaria.
- Se realizo la sistematización de los envíos de dinero para que los clientes no estén obligados a abrir una cuenta bancaria.
- Se realizo la sistematización de las transferencias a otros bancos para que disminuya las filas de clientes generadas en plataforma.
- Se realizo la sistematización de los pagos de servicios para que los clientes realicen sus pagos de manera no presencial.
- Se logro realizar una aplicación que ofrezca todos los servicios de la competencia y que está a la altura del mercado nacional.

6.2 Recomendaciones

A partir del presente trabajo se propone las siguientes recomendaciones, con el fin de buscar el mejoramiento del sistema.

- Se recomienda realizar un sistema alterno orientado al administrador para monitorear las transacciones financieras.
- Se recomienda usar herramientas para seguimientos de fallas en tiempo real que permita informar las fallas a nivel aplicación.

- Se recomienda realizar un módulo de facturación hacia el cliente.
- Se recomienda hacer pruebas de estrés a los servicios en caso ser implementado.
- Se recomienda crear un canal o área de soporte para realizar el seguimiento de casos o para atención al cliente.

BIBLIOGRAFIA

- Rodríguez González, Pilar (2008). Estudio de la Aplicación de Metodologías Ágiles para la Evolución de productos software. Madrid, España.
- Kniberg, H. (2007). Scrum and XP from the Trenches: How we do scrum. Estados Unidos: Editorial C4Media.
- PRESSMAN, R. (2010) “Ingeniería de software, un enfoque práctico”, 7ma. Edición.
- KENNETH E., JULIE E. (2005), “Análisis y diseño de sistemas”, 6ta. edición.
- LARMAN C. (2003), “Introducción al análisis y diseño orientado a objetos y al proceso unificado.”, 1ra. edición.
- SOMMERVILLE IAN, (2005). “Ingeniería de software”, Séptima edición.
- GIMSON LORAINE, (2015). “Metodologías ágiles y desarrollo basado en conocimiento”.
- BECK KENT, ANDRES C. (2012). “Extremme programming explained”, 2da edición
- Cáceres E. (2014), “Análisis y diseño de sistemas de información”

REFERENCIAS ELECTRÓNICAS

- Sonia Rodríguez Ruiz. Definición de Intranet

  Recuperado de http://www.masadelante.com/faqs/intranet

- Berrio Taniles, Redes Informáticas

  Recuperado de http://redesinformaticasipodd.blogspot.com/2014/11/5-internet- intranet-extranet-definicion.html

- e-intelligent, Ventajas de una Intranet

  Recuperado de https://www.e-intelligent.es/es/blog/ventajas-de-una-intranet-corporativa-para-una-empresa-o-pyme

- Cevallos Karla, Metodología de Desarrollo Ágil

  Recuperado de https://ingsotfwarekarlacevallos.wordpress.com/2015/05/08/metodologia-de- desarrollo-agil-xp-y-scrum/

- Felipe Camacho, diferencias entre las metodologías ágiles y tradicionales

  Recuperado de http://diferenciasmetodologias.blogspot.com/2016/04/las- diferencias-entre-las-metodologias_5.html

- Sergio Alberto Castro Acuña, Programación extrema

  Recuperado de https://iswugaps2extremeprogramming.wordpress.com/2015/09/14/roles/

- Juan Pavón Mestras, patrón Modelo-Vista-Controlador

  Recuperado de https://www.fdi.ucm.es/profesor/jpavon/poo/2.14.mvc.pdf

- Citlali G. Nieves, Juan P. Ucán-Pech, Víctor H. Menéndez, Método en Caso de Estudio

  Recuperado de http://sistemas.unla.edu.ar/sistemas/redisla/ReLAIS/relais-v2- n3-137-143.pdf

- Lic. Pinciroli F, Necesidad del empleo de herramientas estándares en XP

  Recuperado de http://sedici.unlp.edu.ar/bitstream/handle/10915/23050/Documento_completo.pdf?sequence=1

  Monografias.com, Estado del arte de metodologías, herramientas y lenguajes para el desarrollo de aplicaciones Web

  Recuperado de https://www.monografias.com/trabajos98/estado-del-arte- metodologias-herramientas-y-lenguajes-desarrollo-aplicaciones-web/estado- del-arte-metodologias-herramientas-y-lenguajes-desarrollo-aplicaciones- web2.shtml

- Wikipedia.org, Red y definiciones

  Recuperado de https://es.wikipedia.org

- Maria Eugenia Arevalo Lizardo, Características de las metodologías ágiles

  Recuperado de https://arevalomaria.wordpress.com/

- Desarrollo tradicional vs. Desarrollo ágil

  Recuperado de http://www.navegapolis.net/files/s/NST-010_01.pdf

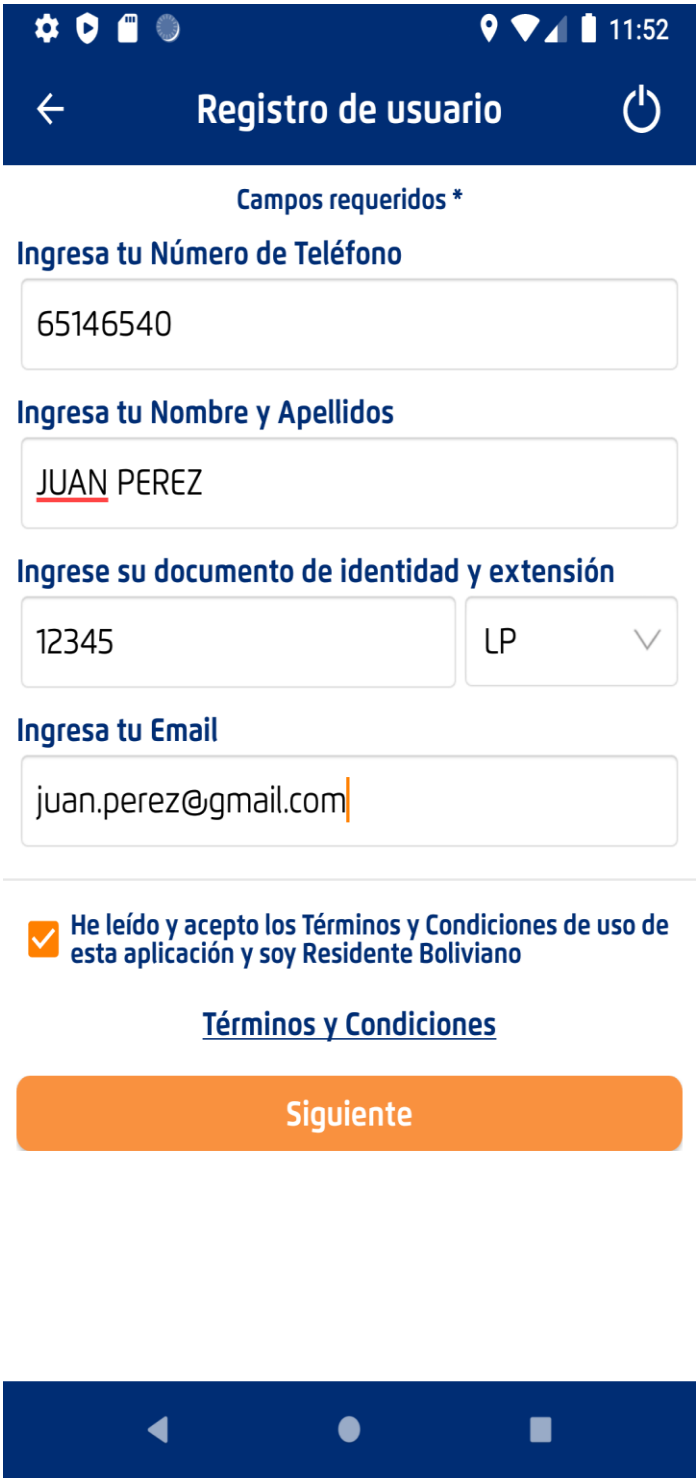
ANEXOS

MANUAL DE USUARIO

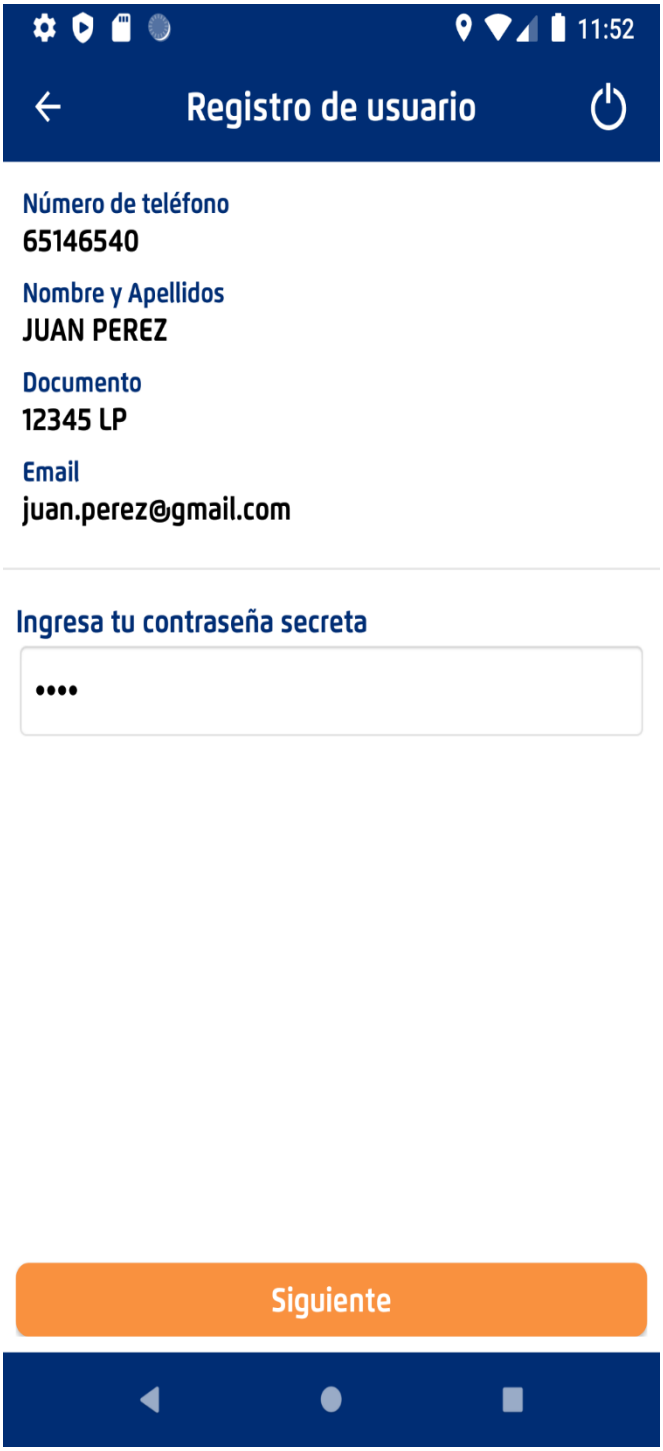
BILLETERA "BCP" REGISTRO DE USUARIO

"CREAR BILLETERA"

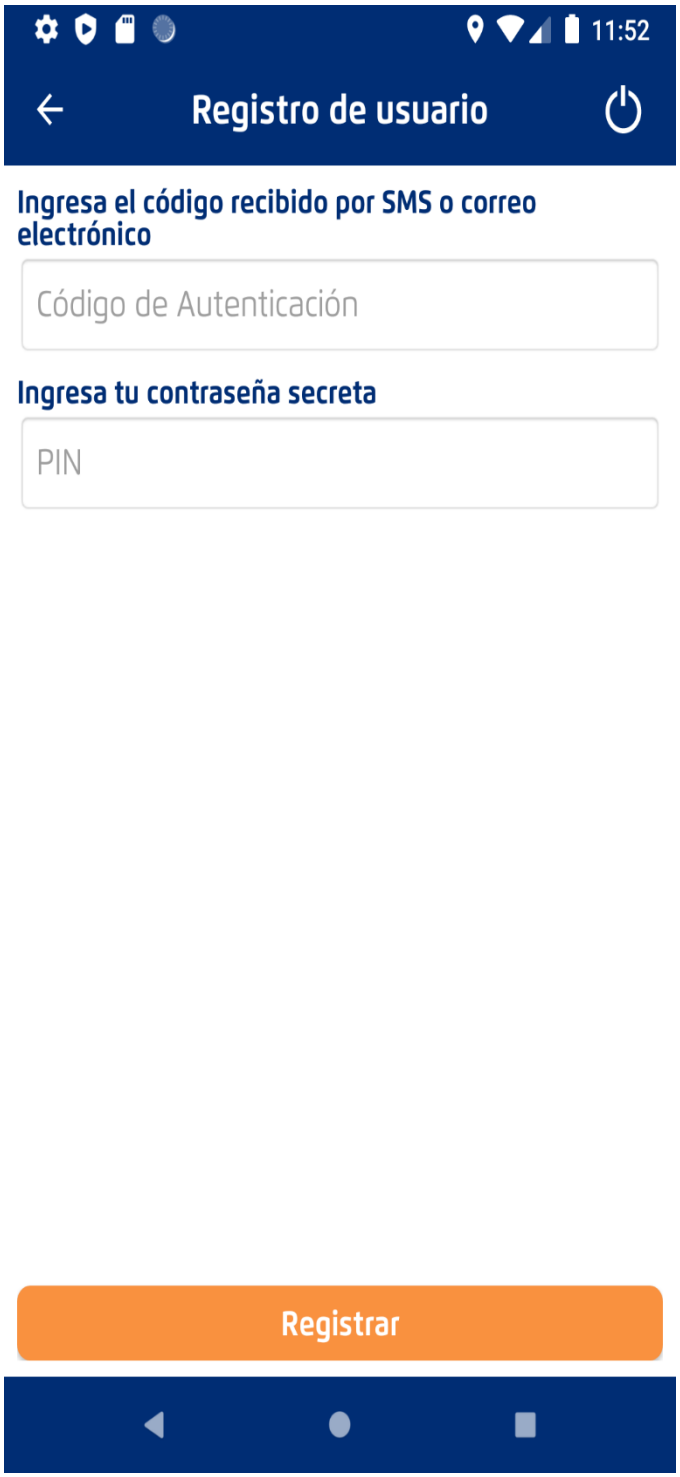
Pantalla principal presionar la opción “**Crear Billetera” afiliación del cliente**



Llenar los dados personales en el formulario de afiliación (**Solo se llena por única vez)** y presionar la opción **siguiente**



Ingrese una contraseña de cuatro dígitos y **siguiente.**

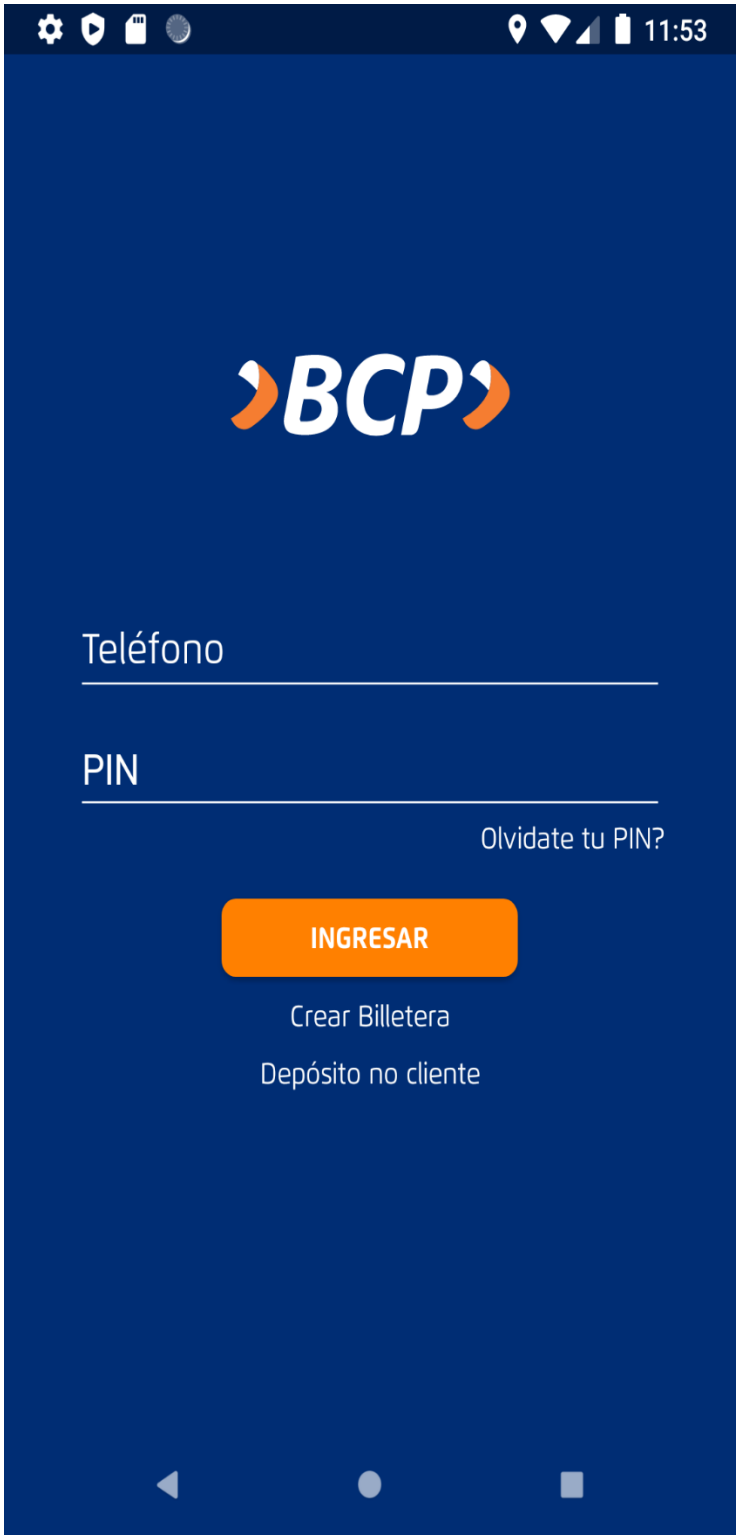


Inserte el código recibido por SMS o correo y el "**Pin"** de cuatro dígitos creado anteriormente, presionar **Registrar.**

BILLETERA "BCP"

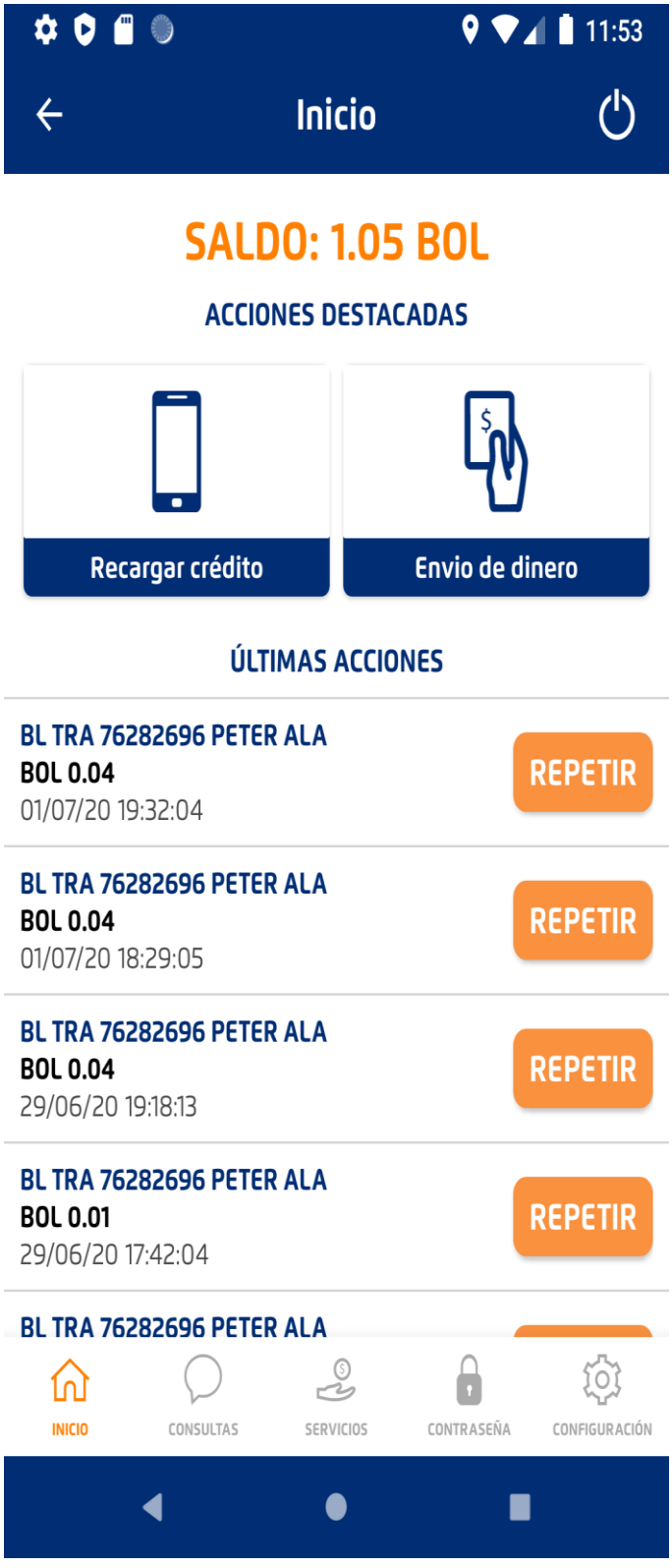
INICIO

"RECARGAR CREDITO"



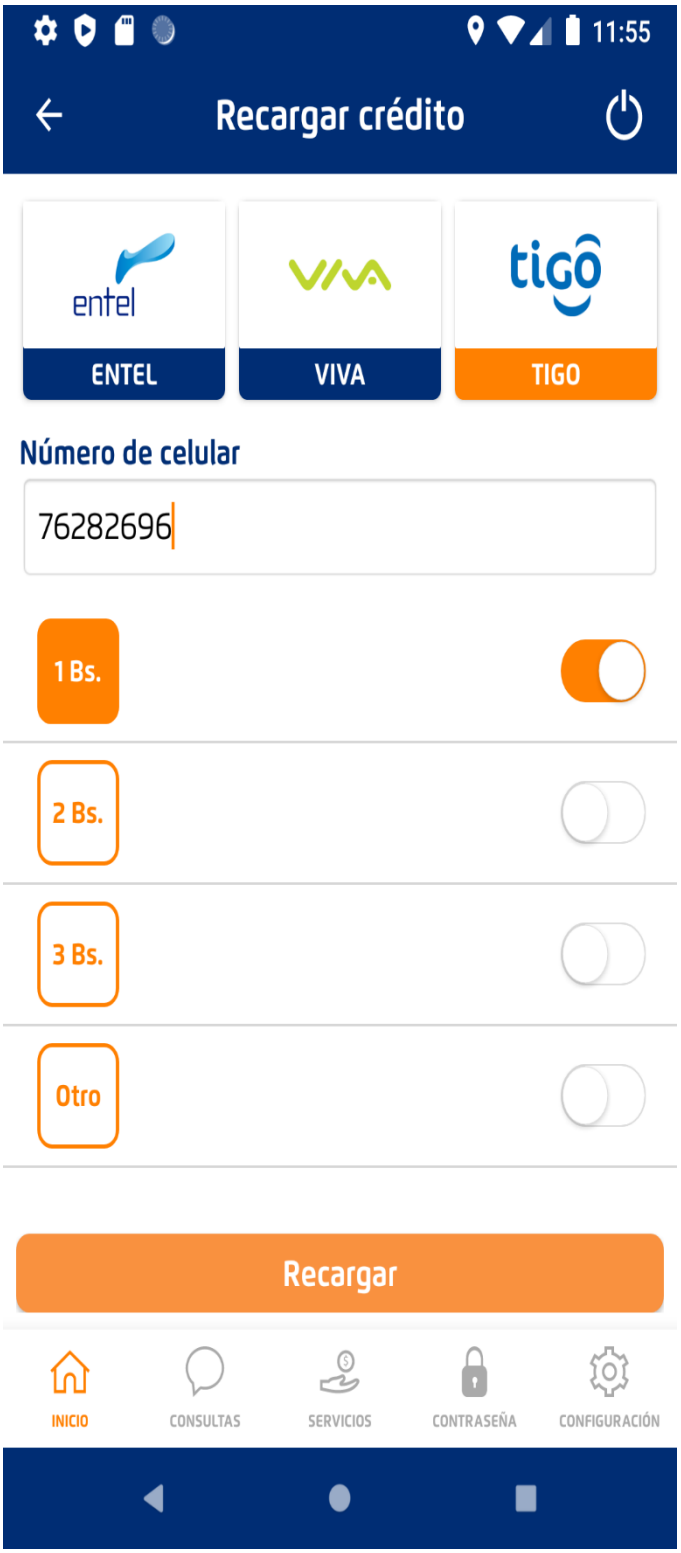
Llenar el **Número De Teléfono**, el **Pin** creado anteriormente y presionar la opción “**ingresar**

le

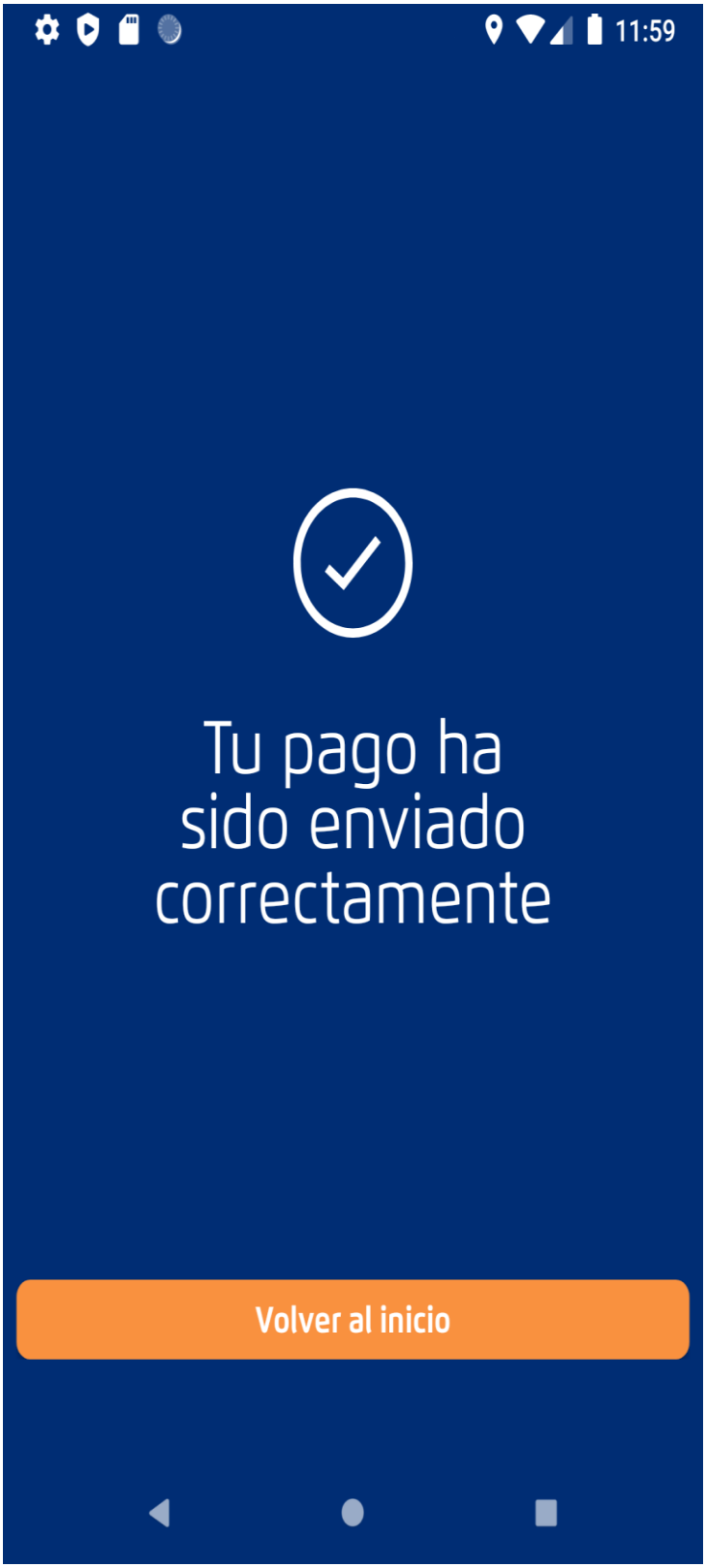


Una vez iniciada sesión en la parte de inicio se observará: el **saldo con el que cuenta**, los **últimos movimientos** que se realiza y dos opciones de **recarga de crédito** y **envió de crédito.**

Ingrese a la opción **“recarga de Crédito”**



Ingrese su número de celular, seleccione el monto recarga que requiera **(ej. 1Bs),** selecciones a la compañía que pertenece su equipo de celular y presione la opción **Recargar**

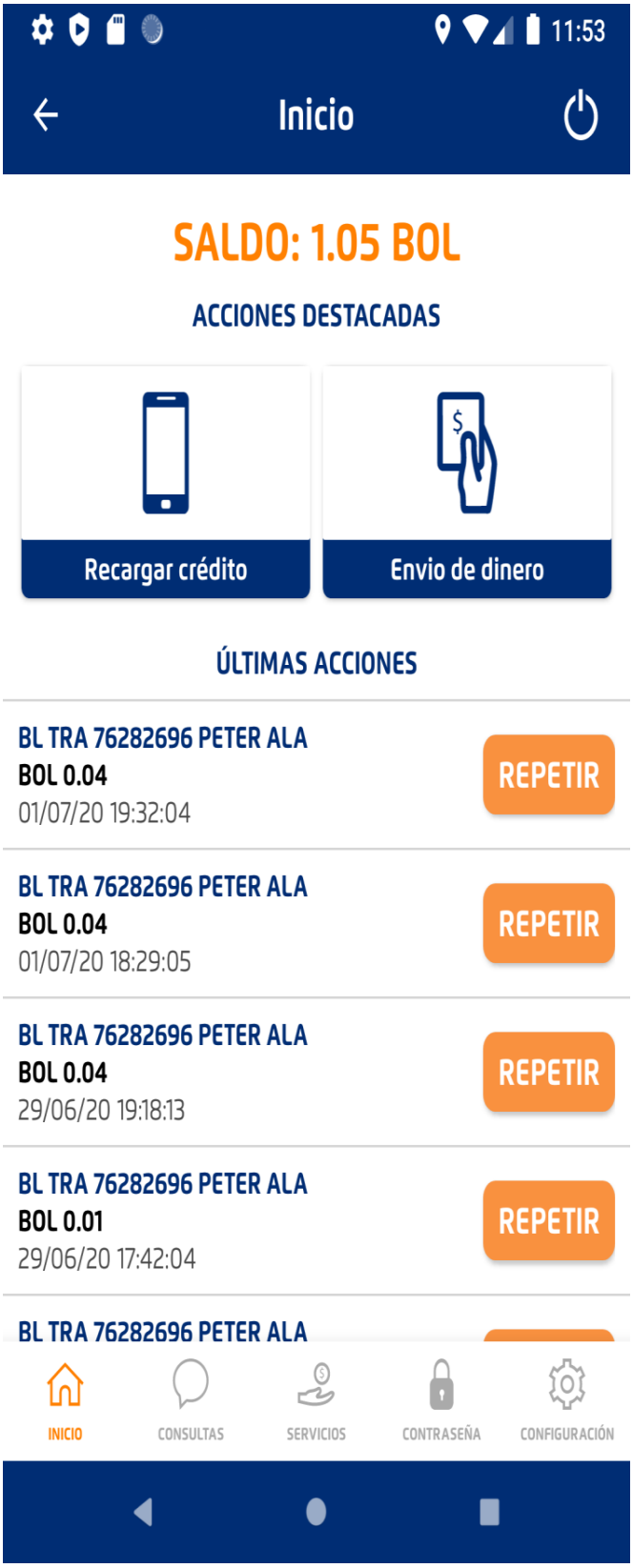


Una vez realizado la recarga “**Volver Al Inicio”.**

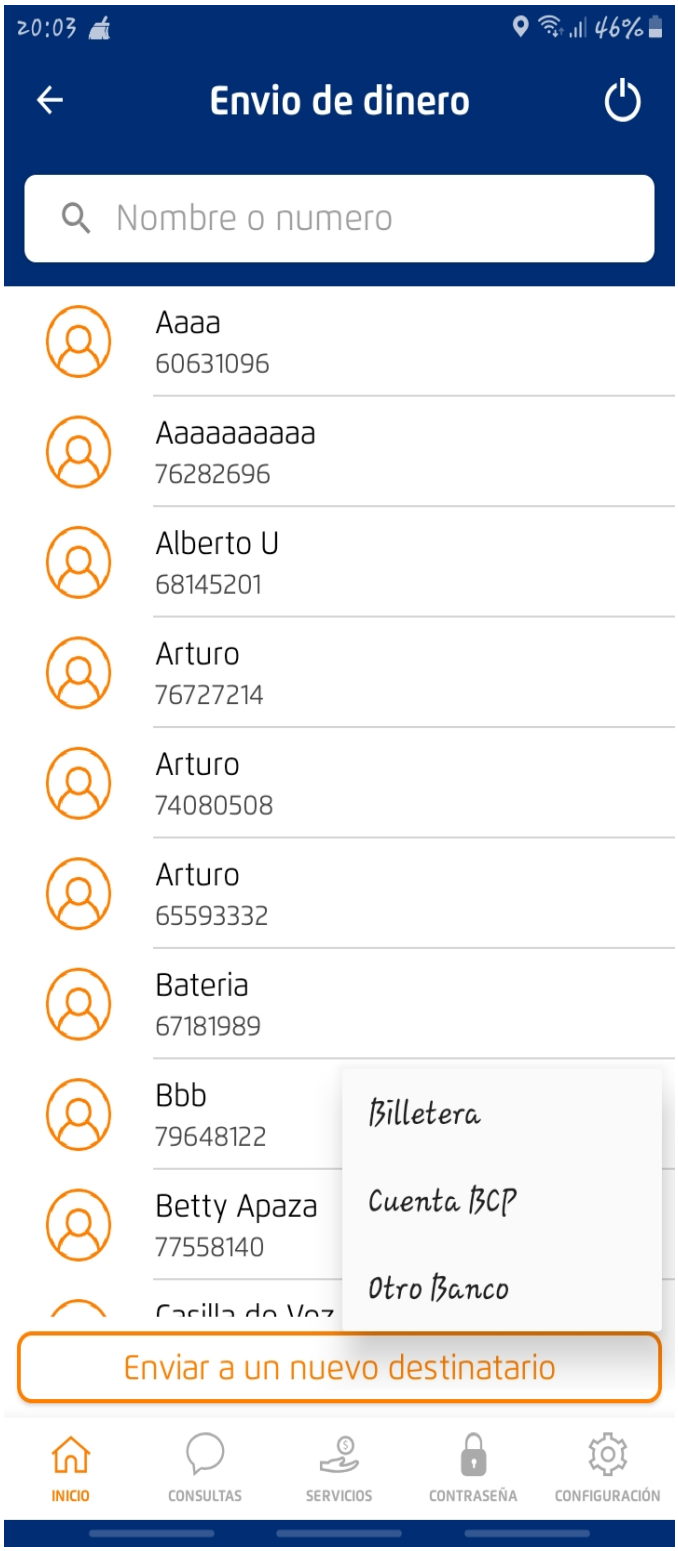
BILLETERA "BCP"

INICIO

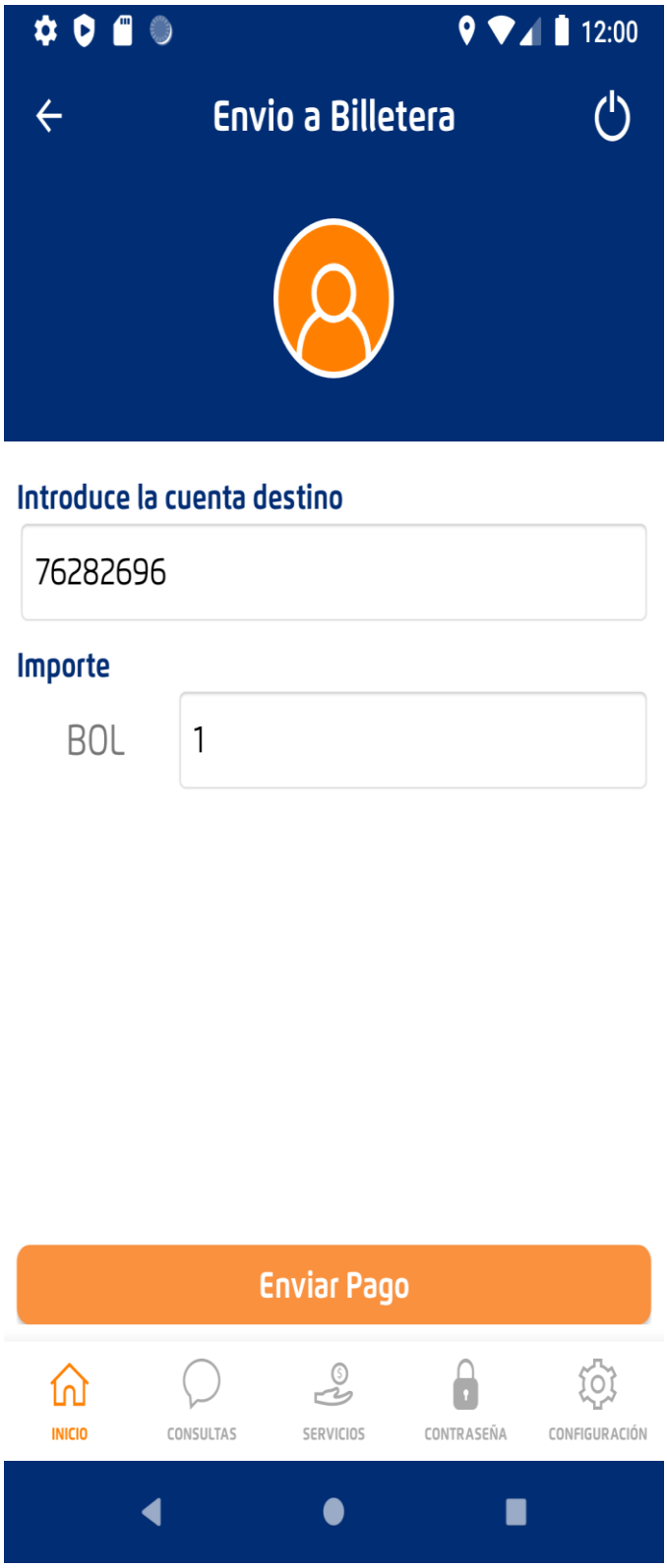
"ENVIO DE DINERO"



Presiona la opción "**Envió de dinero**"

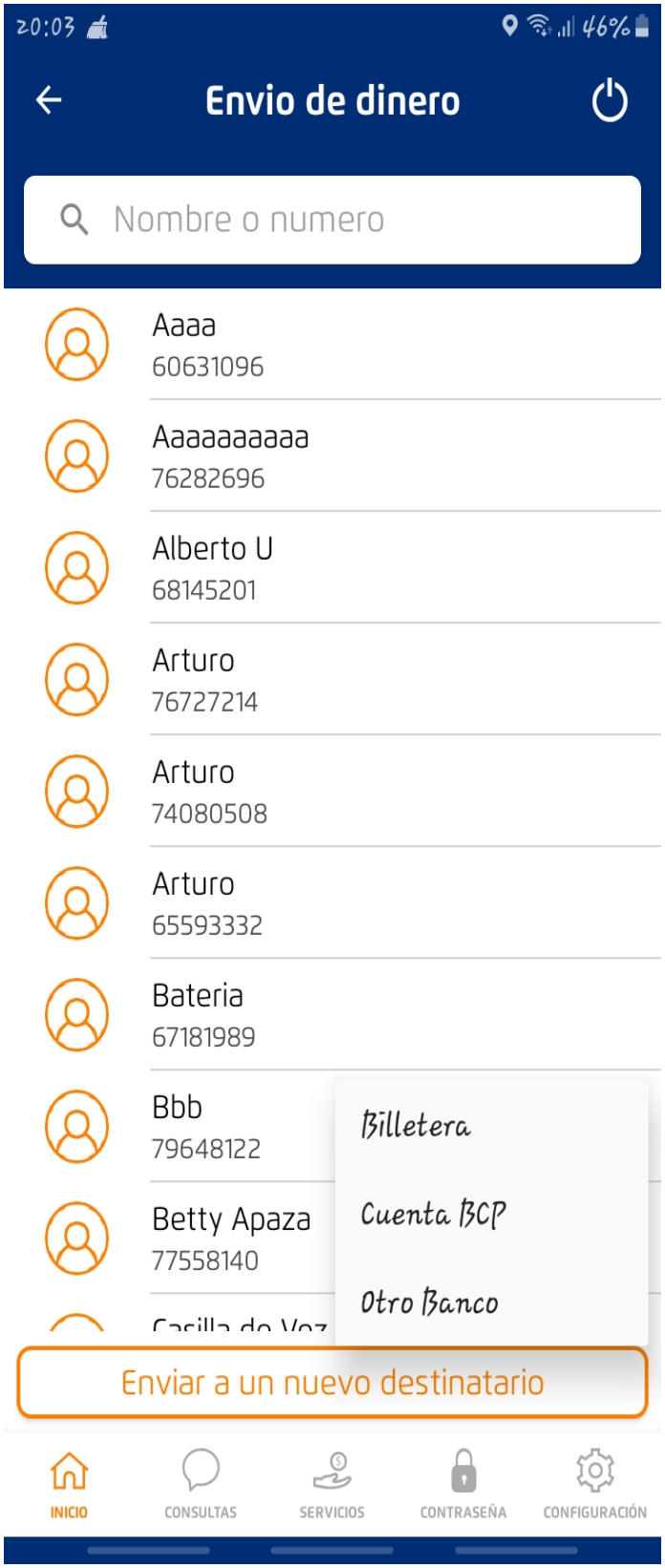


Presionar la opción "**Envió a un nuevo destinatario"** y luego "**Billetera"**

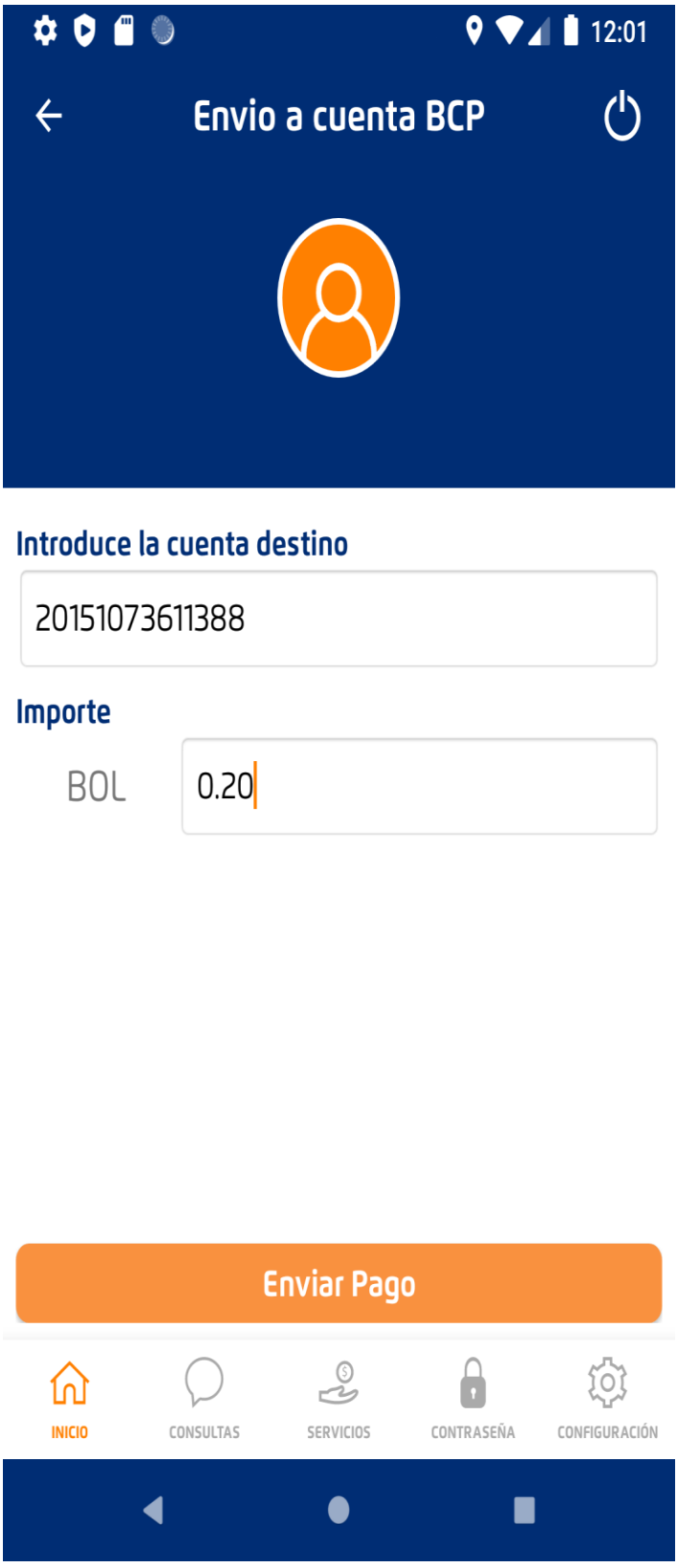


En los espacios inserte el número de celular (Ej.76282696) a destino y posteriormente el importe a enviar (Ej. 1 Bs), presionar "**Enviar Pago"**

Presionar **“Volver a Inicio”**



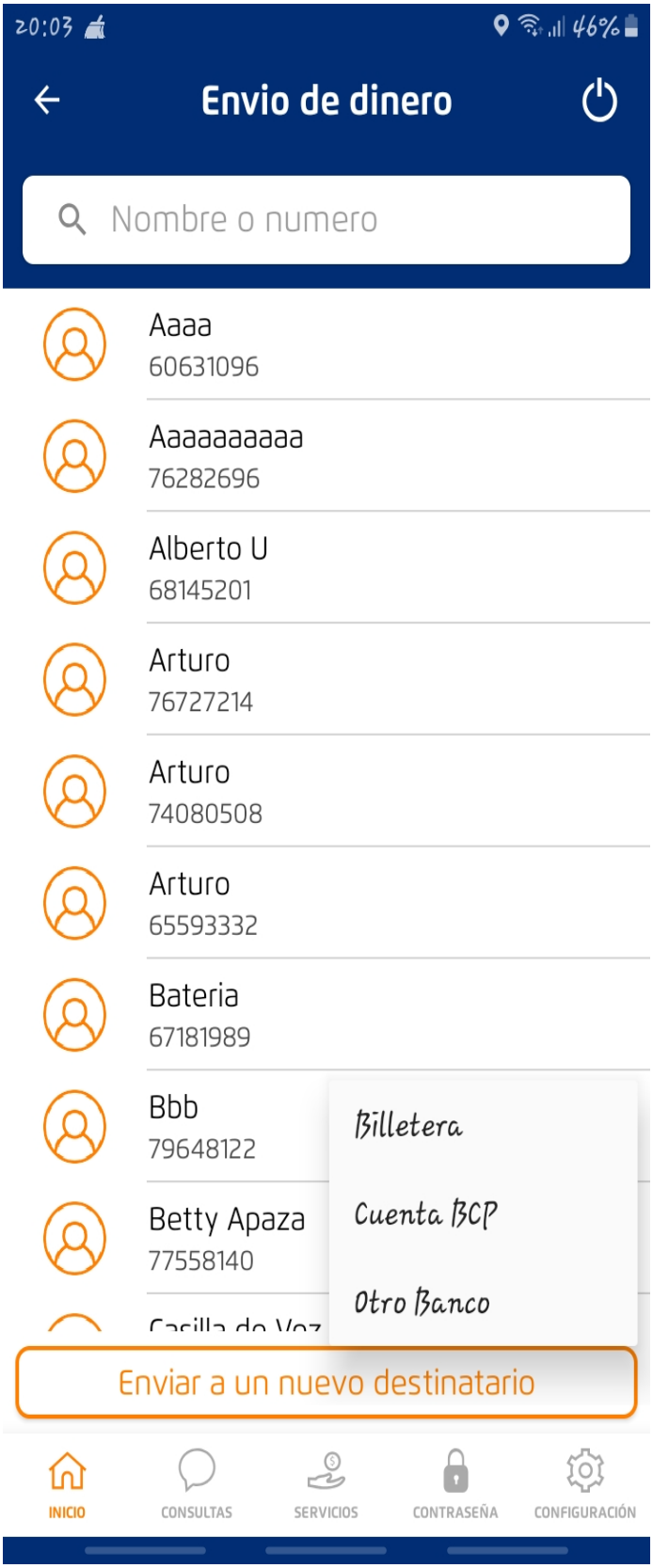
Presionar la opción **"Cuenta BCP"**



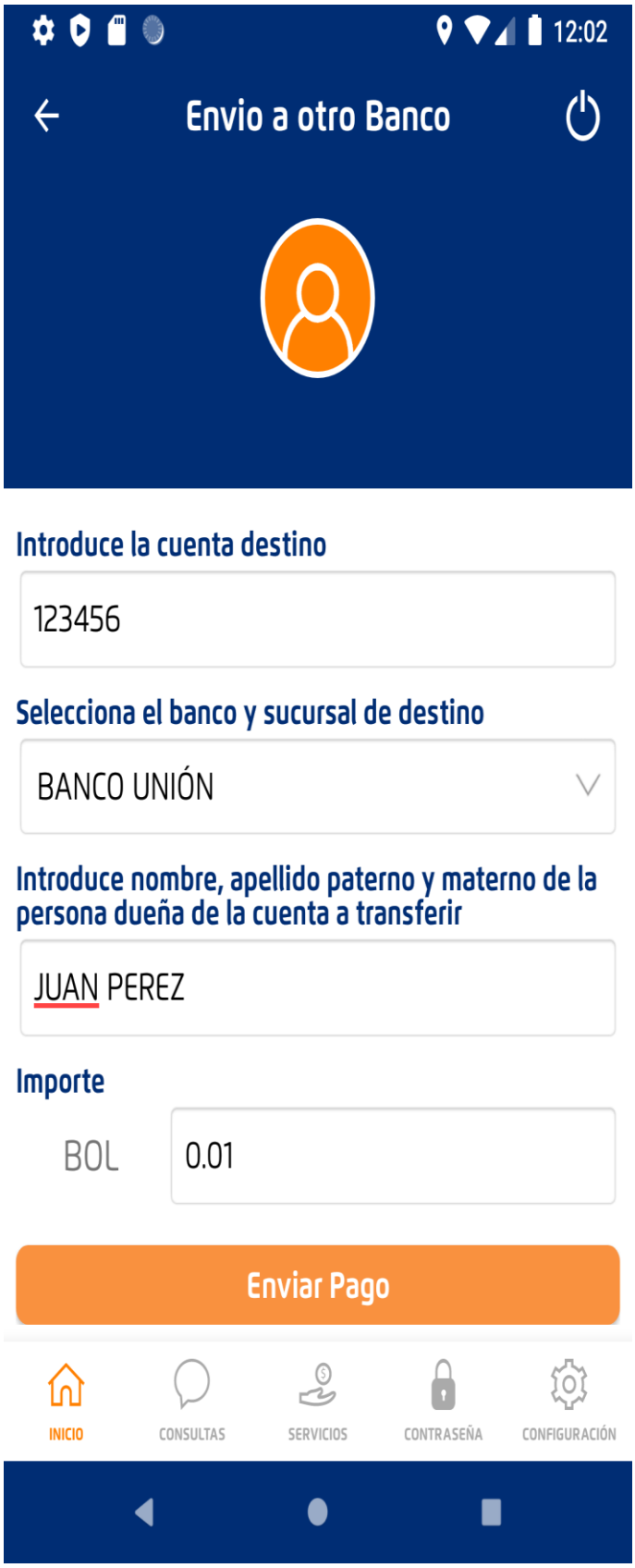
Introducir el número de cuenta BCP (Ej. 20151073611388), y el importe a enviar (Ej. 0.20).

Presionar **Enviar Pago**

Hacer clic en “**volver a inicio**”



Presionar la opción **"otros Bancos"**

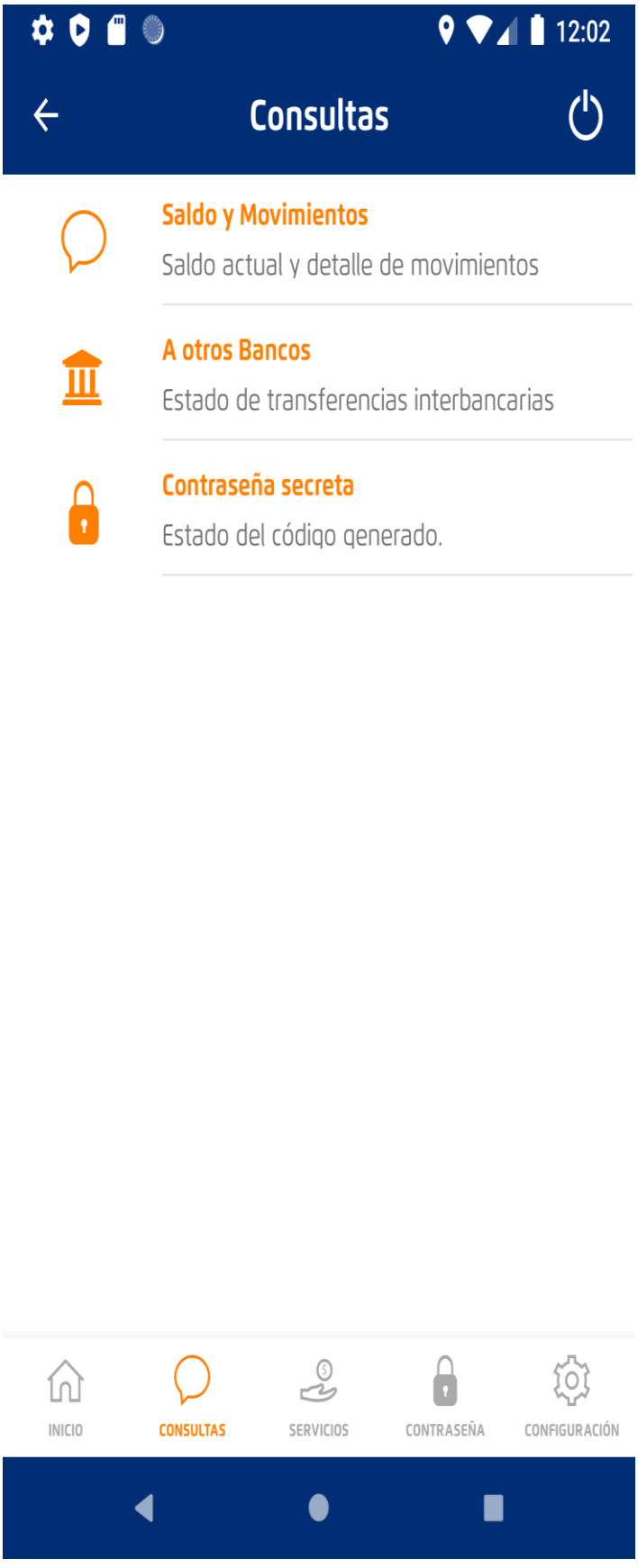


Introducir el número de cuenta a destino (Ej. 123456). Seleccione la entidad financiera a la que pertenece el número de cuenta (Ej. Banco Unión). El nombre al que pertenece la cuenta (Ej. Juan Pérez) y el monto que desee enviar (Ej. 0.01), por ultimo presionar **(Enviar pago)**

Presionar la opción **“volver a Inicio”**

BILLETERA "BCP"

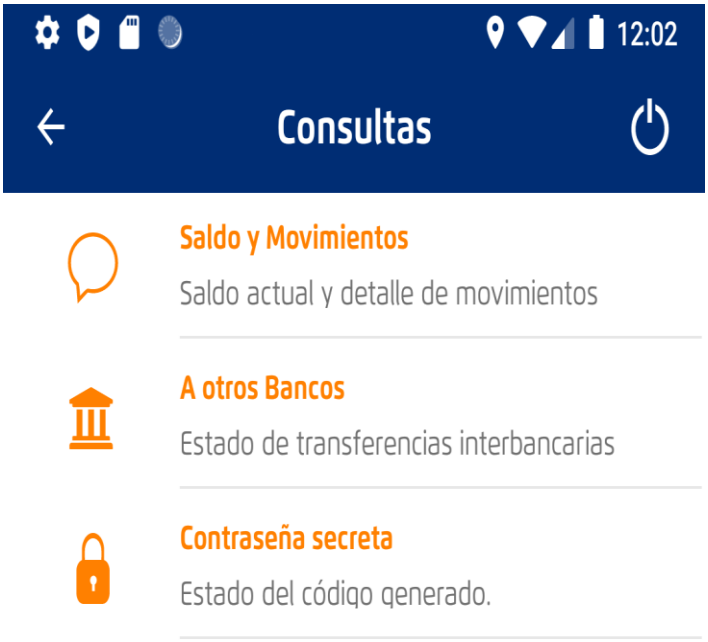
CONSULTAS



Presionar la opción **“Consultas”** y **“Saldo y Movimientos”**

En esta imagen se observará los últimos movimientos que se realizó y el saldo disponible (Ej. 2.85).

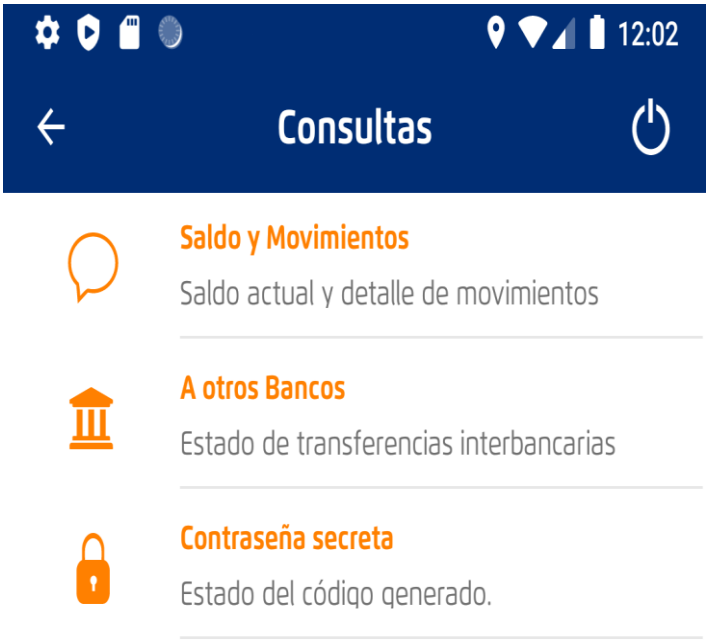
Para ir a tras presionar



Presionar la opción “**Otros Bancos”**

En esta opción se observa los movimientos o transacciones que se realizó a otros bancos

Para ir atrás hacer clic en



Presionar la Opción **“Contraseña Secreta”**

La primera imagen aparece “Código Habilitado” (Ej. 4069) lo que afirma que la contraseña está habilitada.

En la segunda indica que la contraseña esta inhabilitada.

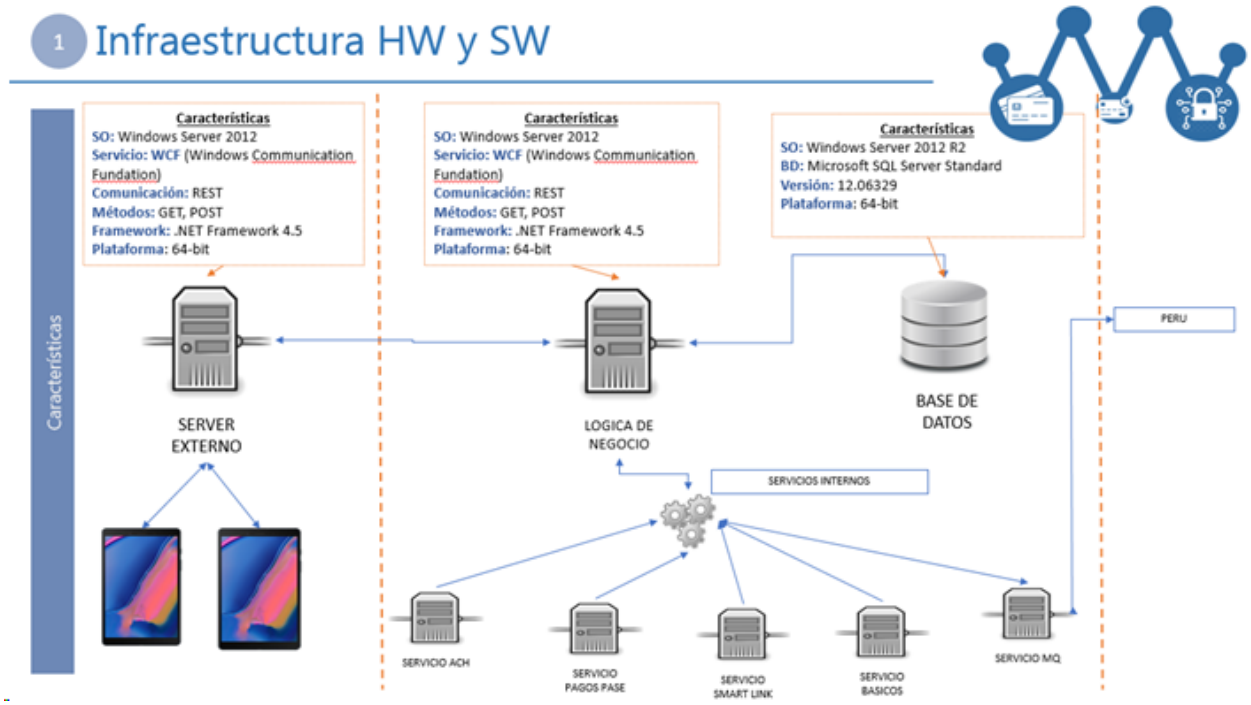
Para volver atrás presionar


12:03
Pagos
Recaudaciones
Pago de servicios
INICIO
CONSULTAS
SERVICIOS
CONTRASEÑA
CONFIGURACIÓN

MANUAL

TECNICO

ARQUITECTURA



MODELADO

DIAGRAMA E/R

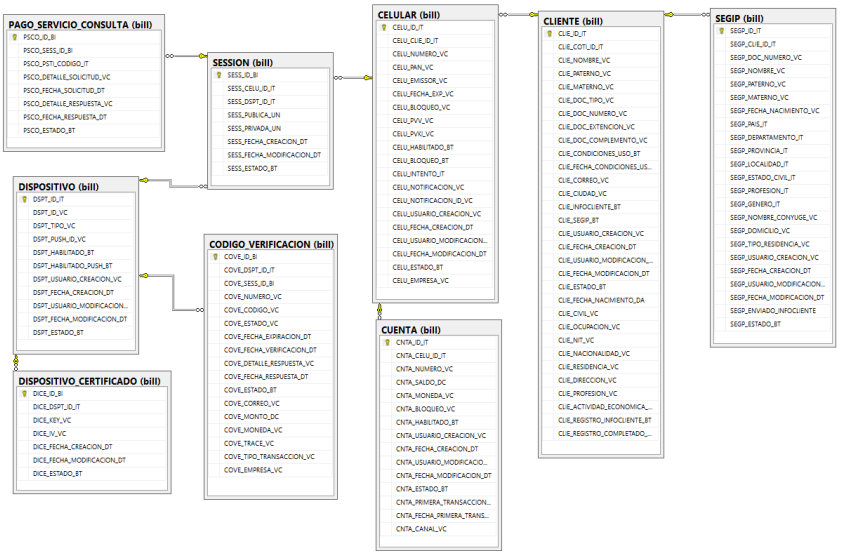


Diagrama de base de datos nivel transaccional

Fuente: Elaboración propia

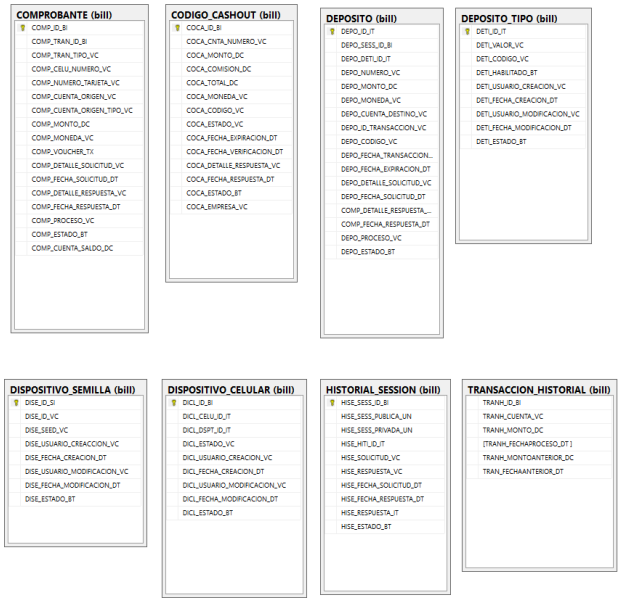


Diagrama de base de datos nivel transaccional e historial

Fuente: Elaboración propia

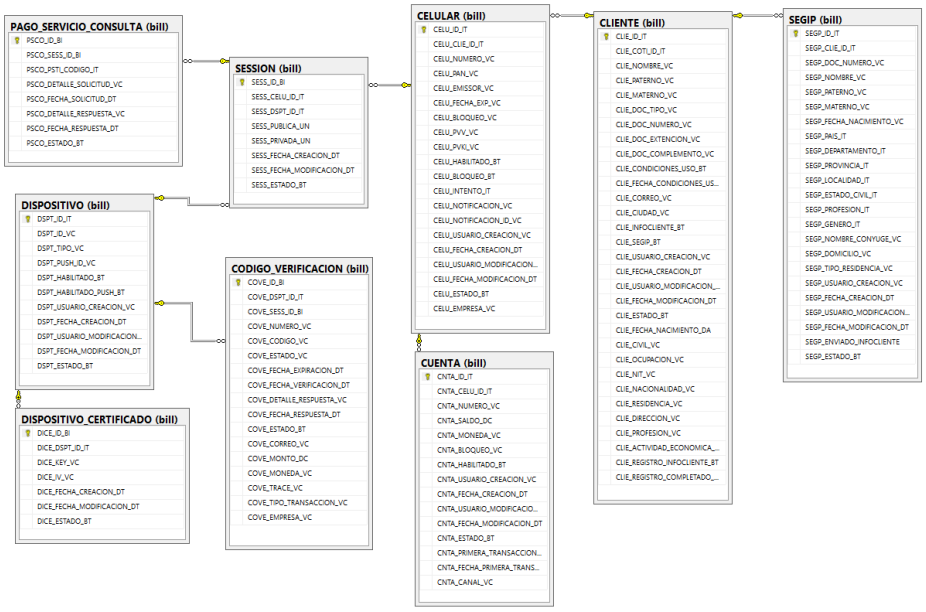


Diagrama de base de datos nivel cliente

Fuente: Elaboración propia

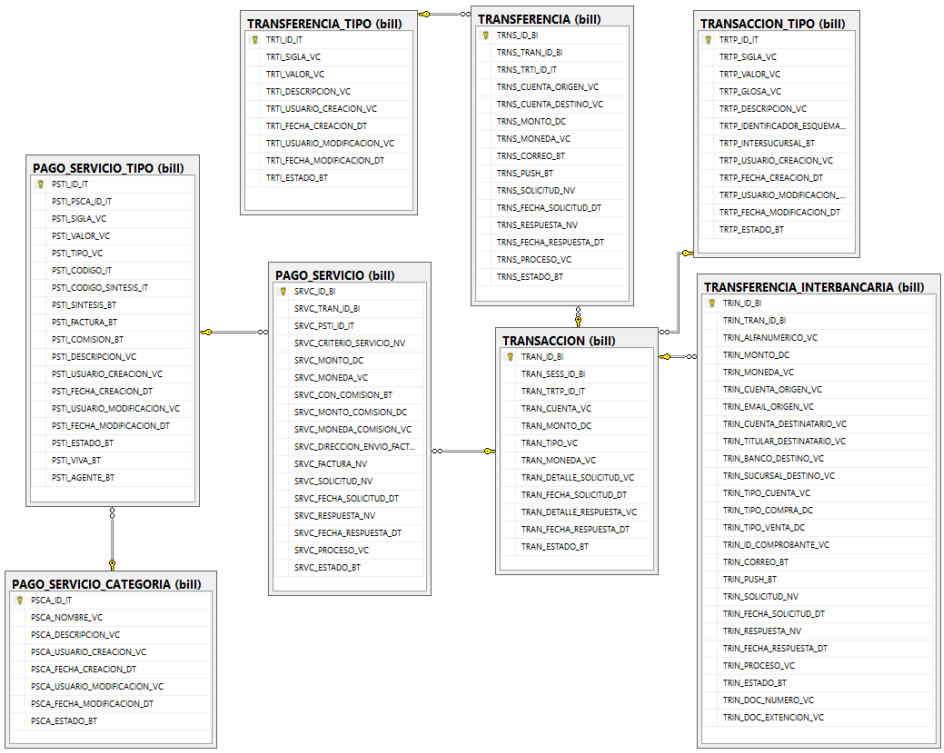
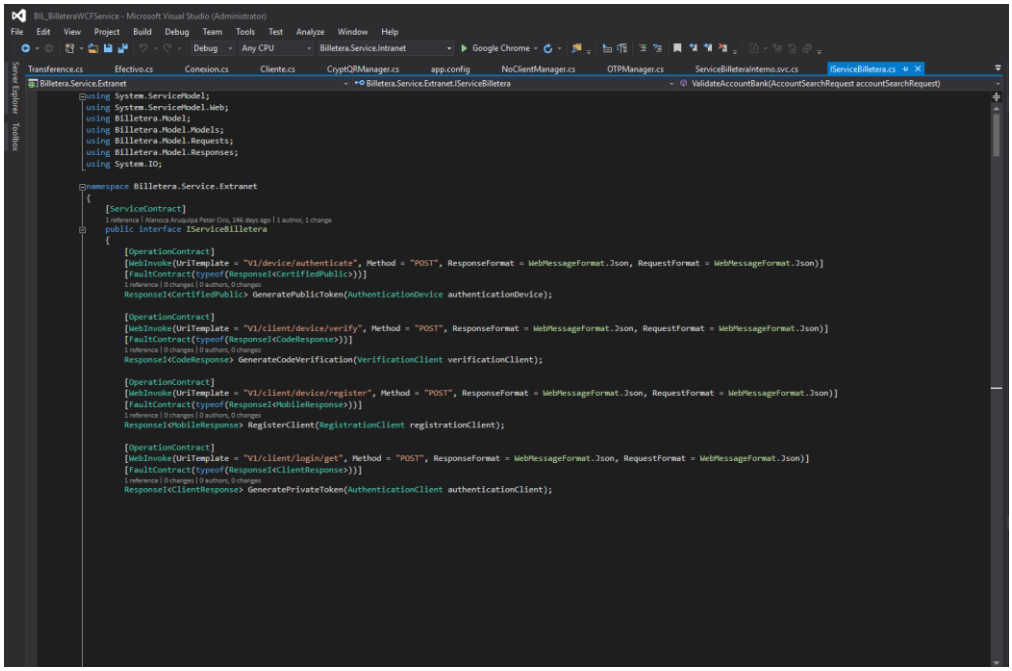


Diagrama de base de datos nivel transferencias y pago de servicios

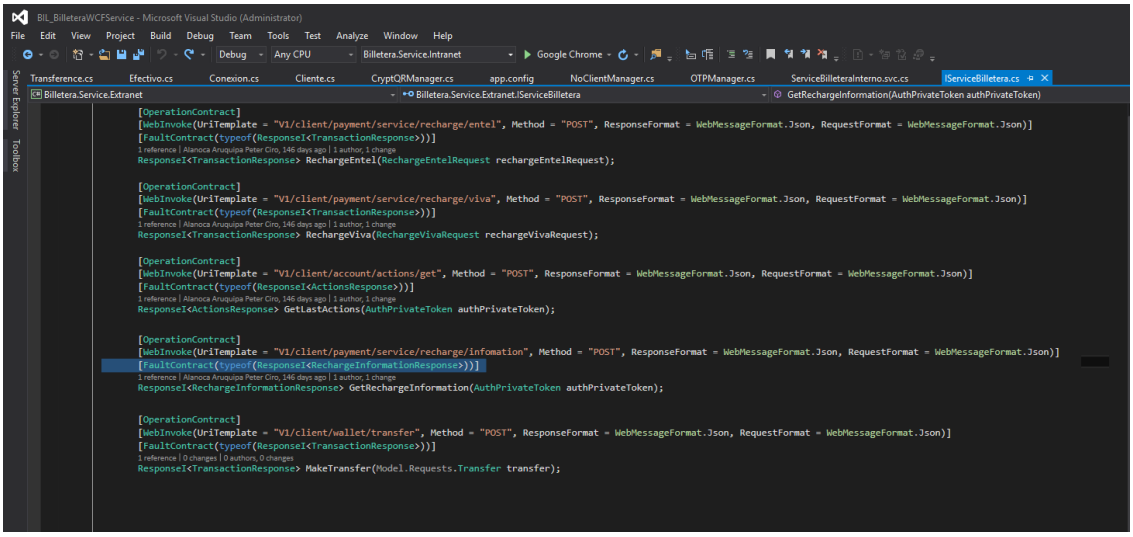
Fuente: Elaboración propia

SERVICIOS WEB



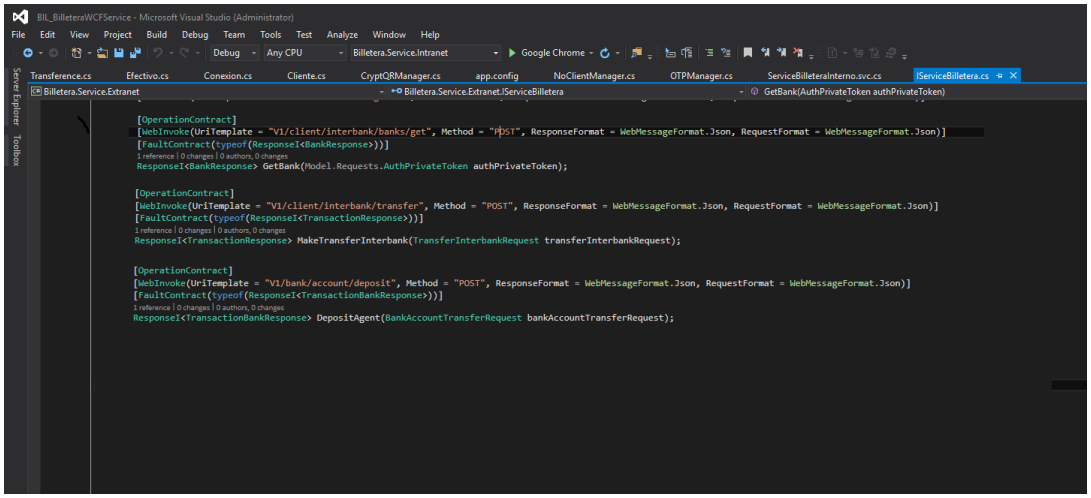
Captura de endpoints del inicio de sesión en Visual Studio

Fuente: Elaboración propia



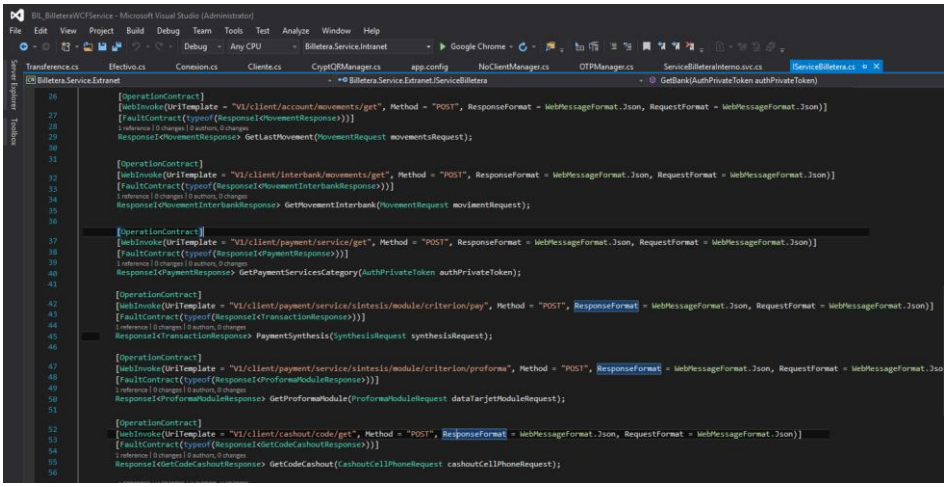
Captura de endpoints de recargas y transferencias en Visual Studio

Fuente: Elaboración propia



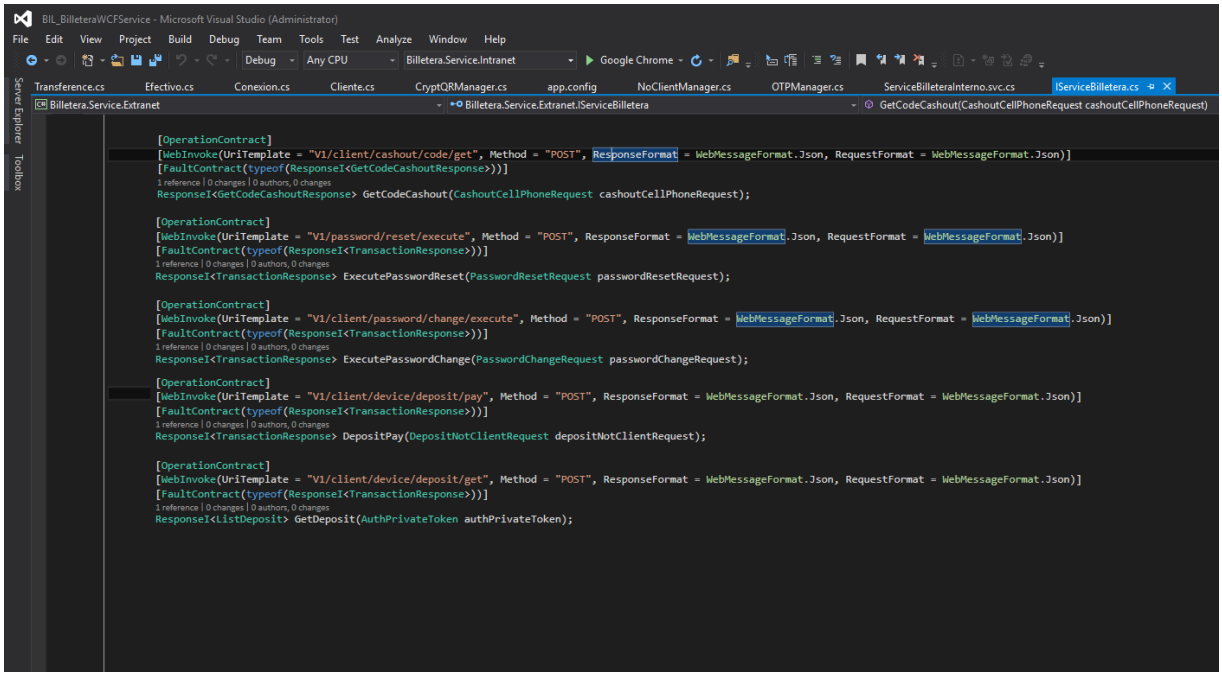
Captura de endpoints de transferencias a cuentas BCP en Visual Studio

Fuente: Elaboración propia



Captura de endpoints de movimientos y pago de servicios en Visual Studio

Fuente: Elaboración propia



Captura de endpoints de depósito y cambio de pin en Visual Studio

Fuente: Elaboración propia

MANUAL DE INSTALACIÓN

Historia de las Revisiones

Las siguientes tablas describen la historia de modificación del documento (Plantilla o Entregable del proyecto) para propósitos de rastreo. Únicamente el cambio que produzca una nueva versión en Harvest deberá ser mostrado en estas tablas.

Historia de modificaciones de la plantilla del MIS

Versión	Fecha	Modificaciones	Modificado por:
1.0	2020/07/15	Creación del documento	Peter Alanoca Aruquipa

Entregable del proyecto (La versión es la misma que se indica en la carátula y es la que el cambio genera en Harvest)

Historia de modificaciones del MIS dentro del proyecto.

Versión (V.X)	Fecha yymmdd	Código cambio (de SCE)	Modificaciones	Modificado por

SERVIDORES INVOLUCRADOS

SERVIDOR	**FUNCIÓN**
BTBBMD01	Servidor de Base de Datos SQL Server 2012
BTBBBW00	Servidor de Servicio Web
SERVER14	Servidor Web

SERVIDOR BTBBMD01

1. **CONSIDERACIONES INICIALES**
   - **1.1. BACKUP PREVIO A LA INSTALACIÓN**
2. **COMUNICACIÓN CON OTROS SERVIDORES**
3. **CREACIÓN DE USUARIOS de dominio**
4. **COPIA DE ARCHIVOS**
5. **CREACIÓN DE DIRECTORIOS COMPARTIDOS**
6. **INSTALACIÓN DE APLICATIVOS**
7. **INSTALACIÓN DE SERVICIOS**
8. **CONFIGURACIÓN DE SERVICIOS**
9. **CAMBIOS EN EL REGISTRO DEL SISTEMA**
10. **CONFIGURACIÓN DE SERVIDOR DE BASES DE DATOS**
11. **PROGRAMACIÓN DE TAREAS DE BASE DE DATOS**
12. **CONFIGURACIÓN DE SERVIDORES REMOTOS DE BASE DE DATOS**
13. **CREACIÓN DE DSN**
14. **REGISTRO COMPONENTES EN EL REGISTRO DEL SISTEMA**
15. **CREACIÓN DE PAQUETES TRANSACCIONALES**
16. **CREACIÓN DE REPORTES**
17. **REGISTRO DE COMPONENTES EN EL SERVIDOR TRANSACCIONAL**
18. **GENERAR ARCHIVOS PARA EXPORTAR LOS PAQUETES INSTALADOS**
19. **INSTALACIÓN DE COMPONENTES REMOTOS**
20. **REGISTRO DE COMPONENTES PARA COMUNICACIÓN CON HOST**
21. **CONFIGURACIÓN DEL SERVIDOR WEB**
22. **ARCHIVO DE RESOLUCIÓN DE NOMBRES (EJEM. LMHOSTS/HOSTS)**
23. **PROGRAMACIÓN DE TAREAS DEL SERVIDOR**
24. **RESPALDO Y RECUPERACIÓN DE DATOS (BACKUPS)**
25. **OTRAS CONFIGURACIONES**
26. **ROLLBACK**
27. **TAREAS COORDINADAS CON HOST**

CONFIGURACIÓN POR SERVIDOR BTBBMD01

1. CONSIDERACIONES INICIALES

No hay impacto en esta sección.

2. BACKUP PREVIO A LA INSTALACIÓN

Realizar un back up de la base de datos

No hay impacto en esta sección.

3. COMUNICACIÓN CON OTROS SERVIDORES

No hay impacto en esta sección.

4. USUARIOS GENÉRICOS (Dominio, SIX, Servidor, etc.)

El usuario de la base de datos BD_Billetera con el cual se realiza la conexión es el siguiente:

Certificación	Producción
UsrBD_BILLETERA	UsrBD_BILLETERA

5. COPIA DE ARCHIVOS

No hay impacto en esta sección.

6. CREACIÓN DE DIRECTORIOS COMPARTIDOS

No hay impacto en esta sección.

7. INSTALACIÓN DE APLICATIVOS

No hay impacto en esta sección.

8. INSTALACIÓN DE SERVICIOS

No hay impacto en esta sección.

9. CONFIGURACIÓN DE SERVICIOS

No hay impacto en esta sección.

CAMBIOS EN EL REGISTRO DEL SISTEMA

No hay impacto en esta sección.

10. CONFIGURACIÓN DE SERVIDOR DE BASES DE DATOS

Ingresar a Microsoft SQL Server 2008 → SQL Server Management Studio, conectarse al gestor con los siguientes datos:

Server type: DataBase Engine

Server name: BTBBMD01 (Producción) y BTBBMD01 (Certificación)

Authentication: Ingrese con una cuenta administradora y asegúrese que la misma pertenezca al grupo sysadmin del SQL Server.

Abrir la siguiente ruta del congelado:

\5.Instaladores\2.Servidor\1.Scripts\1.Carga Inicial

Ejecutar el siguiente script de modificación de la estructura de datos.

1. Carga Inicial BD_Billetera.sql

11. PROGRAMACIÓN DE TAREAS DE BASE DE DATOS

No hay impacto en esta sección.

12. CONFIGURACIÓN DE SERVIDORES REMOTOS DE BASE DE DATOS

No hay impacto en esta sección.

13. CREACIÓN DE DSN

No hay impacto en esta sección.

14. REGISTRO COMPONENTES EN EL REGISTRO DEL SISTEMA

No hay impacto en esta sección.

15. CREACIÓN DE PAQUETES TRANSACCIONALES

No hay impacto en esta sección.

16. CREACIÓN DE REPORTES

No hay impacto en esta sección.

17. REGISTRO DE COMPONENTES EN EL SERVIDOR TRANSACCIONAL

No hay impacto en esta sección.

18. GENERAR ARCHIVOS PARA EXPORTAR LOS PAQUETES INSTALADOS

No hay impacto en esta sección.

19. INSTALACIÓN DE COMPONENTES REMOTOS

No hay impacto en esta sección.

20. REGISTRO DE COMPONENTES PARA COMUNICACIÓN CON HOST

No hay impacto en esta sección.

21. CONFIGURACIÓN DEL SERVIDOR WEB

No hay impacto en esta sección.

22. ARCHIVO DE RESOLUCIÓN DE NOMBRES (EJEM. LMHOSTS/HOSTS)

No hay impacto en esta sección.

23. PROGRAMACIÓN DE TAREAS DEL SERVIDOR

No hay impacto en esta sección.

24. RESPALDO Y RECUPERACIÓN DE DATOS (BACKUPS).

No hay impacto en esta sección.

25. OTRAS CONFIGURACIONES

No hay impacto en esta sección.

26. ROLLBACK

Ingresar a Microsoft SQL Server 2008 → SQL Server Management Studio, conectarse al gestor con los siguientes datos:

Server type: DataBase Engine

Server name: BTBBMD01 (Producción) y BTBBMD01 (Certificación)

Authentication: Ingrese con una cuenta administradora y asegúrese que la misma pertenezca al grupo sysadmin del SQL Server.

27. TAREAS COORDINADAS CON HOST

No hay impacto en esta sección.

SERVIDOR BTBBBW00

1. **CONSIDERACIONES INICIALES**
2. **COMUNICACIÓN CON OTROS SERVIDORES**
3. **CREACIÓN DE USUARIOS de dominio**
4. **COPIA DE ARCHIVOS**
5. **CREACIÓN DE DIRECTORIOS COMPARTIDOS**
6. **INSTALACIÓN DE APLICATIVOS**
7. **INSTALACIÓN DE SERVICIOS**
8. **CONFIGURACIÓN DE SERVICIOS**
9. **CAMBIOS EN EL REGISTRO DEL SISTEMA**
10. **CONFIGURACIÓN DE SERVIDOR DE BASES DE DATOS**
11. **PROGRAMACIÓN DE TAREAS DE BASE DE DATOS**
12. **CONFIGURACIÓN DE SERVIDORES REMOTOS DE BASE DE DATOS**
13. **CREACIÓN DE DSN**
14. **REGISTRO COMPONENTES EN EL REGISTRO DEL SISTEMA**
15. **CREACIÓN DE PAQUETES TRANSACCIONALES**
16. **CREACIÓN DE REPORTES**
17. **REGISTRO DE COMPONENTES EN EL SERVIDOR TRANSACCIONAL**
18. **GENERAR ARCHIVOS PARA EXPORTAR LOS PAQUETES INSTALADOS**
19. **INSTALACIÓN DE COMPONENTES REMOTOS**
20. **REGISTRO DE COMPONENTES PARA COMUNICACIÓN CON HOST**
21. **CONFIGURACIÓN DEL SERVIDOR WEB**
22. **ARCHIVO DE RESOLUCIÓN DE NOMBRES (EJEM. LMHOSTS/HOSTS)**
23. **PROGRAMACIÓN DE TAREAS DEL SERVIDOR**
24. **RESPALDO Y RECUPERACIÓN DE DATOS (BACKUPS)**
25. **OTRAS CONFIGURACIONES**
26. **ROLLBACK**
27. **TAREAS COORDINADAS CON HOST**

CONFIGURACIÓN POR SERVIDOR BTBBBW00

1. CONSIDERACIONES INICIALES

No hay impacto esta sección.

2. COPIA DE ARCHIVOS

Archivos de los publicados

Se debe realizar la copia de los archivos del publicado de los servicios web

Realizar la copia del contenido de la carpeta:

WS_Billetera

De la siguiente ruta del congelado:

Congelado\5.Instaladores\2.Servidor\3.Instaladores\WS_Billetera

Copiar el contenido a la siguiente ruta del servidor:

C:\inetpub\wwwroot\Billetera\WS_Billetera

Nota: En caso de que las carpetas existan, realizar un reemplazo de las mismas. Caso contrario crearlas

3. CONFIGURACIÓN DE SERVIDOR DE BASES DE DATOS

No hay impacto en esta sección.

4. PROGRAMACIÓN DE TAREAS DE BASE DE DATOS

No hay impacto en esta sección.

5. CONFIGURACIÓN DE SERVIDORES REMOTOS DE BASE DE DATOS

No hay impacto en esta sección.

6. CREACIÓN DE DSN

No hay impacto en esta sección.

7. REGISTRO COMPONENTES EN EL REGISTRO DEL SISTEMA

No hay impacto en esta sección.

8. CREACIÓN DE PAQUETES TRANSACCIONALES

No hay impacto en esta sección.

9. CREACIÓN DE REPORTES

No hay impacto en esta sección.

10. REGISTRO DE COMPONENTES EN EL SERVIDOR TRANSACCIONAL

No hay impacto en esta sección.

11. GENERAR ARCHIVOS PARA EXPORTAR LOS PAQUETES INSTALADOS

No hay impacto en esta sección.

12. INSTALACIÓN DE COMPONENTES REMOTOS

No hay impacto en esta sección.

13. REGISTRO DE COMPONENTES PARA COMUNICACIÓN CON HOST

No hay impacto en esta sección.

14. CONFIGURACIÓN DEL SERVIDOR WEB

Ingresar a la carpeta bin del back up del *Billetera*

…\WS_Billetera\bin

Seleccionar los archivos y copiar:

Ingresar a la siguiente ruta del servidor y pegar los archivos seleccionados:

C:\inetpub\wwwroot\Billetera\WS_Billetera\bin

En la sección de **Application Pool**, seleccionar **App_Billetera**, reiniciar el Application Pool.

15. ARCHIVO DE RESOLUCIÓN DE NOMBRES (EJEM. LMHOSTS/HOSTS)

No hay impacto en esta sección.

16. PROGRAMACIÓN DE TAREAS DEL SERVIDOR

No hay impacto en esta sección.

17. RESPALDO Y RECUPERACIÓN DE DATOS (BACKUPS).

No hay impacto en esta sección.

18. OTRAS CONFIGURACIONES

No hay impacto en esta sección.

19. TAREAS COORDINADAS CON HOST

No hay impacto en esta sección.

DOCUMENTACIÓN

DECLARACIÓN JURADA

Mediante la presente declaro de manera pública que la propuesta de investigación titulada "DESARROLLO DE UNA BILLETERA MOVIL PARA REALIZAR TRANSACCIONES FINANCIERAS CASI: BANCO DE CRÉDITO" es de mi autoría y no constituye una copia o replica de trabajos similares elaborados con carácter previo.

Autorizo la publicación del resumen de mi propuesta en Internet y me comprometo a responder todos los cuestionamientos que se desprendan de su lectura.

La Paz, julio de 2020

Peter Ciro Alanoca Aruquipa

CI: 7031708 LP

AVAL DE CONFORMIDAD

El Alto, julio de 2020

Señores:

HONORABLE CONSEJO DE CARRERA

CARRERA INGENIERÍA DE SISTEMAS

Presente.-

REF. AVAL DE CONFORMIDAD

Distinguida autoridad:

Mediante la presente tengo a bien comunicarle mi conformidad del proyecto de grado DESARROLLO DE UNA BILLETERA MOVIL PARA REALIZAR TRANSACCIONES FINANCIERAS CASO: BANCO DE CRÉDITO DE BOLIVIA "BCP" que propone el postulante Peter Ciro Alanoca Aruquipa, con cédula de identidad 7031708 LP. Para su defensa pública, evaluación correspondiente a la materia Taller de Licenciatura II, de acuerdo a reglamento vigente de la Carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto.

Sin otro particular, me despido con las consideraciones más distinguidas.

Atentamente,

Ing. Marisol Arguedas Balladares

DOCENTE TALLER DE LICENCIATURA II

AVAL DE CONFORMIDAD

El Alto, julio de 2020

Señores:

HONORABLE CONSEJO DE CARRERA

CARRERA INGENIERÍA DE SISTEMAS

Presente. -

REF. AVAL DE CONFORMIDAD

Distinguida autoridad:

Mediante la presente tengo a bien comunicarle mi conformidad del proyecto de grado DESARROLLO DE UNA BILLETERA MOVIL PARA REALIZAR TRANSACCIONES FINANCIERAS CASO: BANCO DE CRÉDITO DE BOLIVIA "BCP" que propone el postulante Peter Ciro Alanoca Aruquipa, con cédula de identidad 7031708 LP. para su defensa pública, evaluación correspondiente a la materia Taller de Licenciatura II, de acuerdo a reglamento vigente de la Carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto.

Sin otro particular, reciba saludos cordiales.

Atentamente,

Lic. Freddy Alanoca Coarite

TUTOR REVISOR

AVAL DE CONFORMIDAD

El Alto, julio de 2020

Señores:

HONORABLE CONSEJO DE CARRERA

CARRERA INGENIERÍA DE SISTEMAS

Presente. -

REF. AVAL DE CONFORMIDAD

Distinguida autoridad:

Mediante la presente tengo a bien comunicarle mi conformidad del proyecto de grado DESARROLLO DE UNA BILLETERA MOVIL PARA REALIZAR TRANSACCIONES FINANCIERAS CASO: BANCO DE CRÉDITO DE BOLIVIA "BCP" que propone el postulante Peter Ciro Alanoca Aruquipa, con cédula de identidad 7031708 LP. Para su defensa pública, evaluación correspondiente a la materia Taller de Licenciatura II, de acuerdo a reglamento vigente de la Carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto.

Sin otro particular, reciba saludos cordiales.

Atentamente,

Ing. Reynaldo Javier Zeballos Daza

TUTOR ESPECIALISTA

El Alto, julio de 2020

AVAL DE CONFORMIDAD

Señor(a):

Ing. Marisol Arguedas Balladares

TUTOR METODOLÓGICO TALLER II

Presente. -

REF. AVAL DE CONFORMIDAD

Distinguida Licenciada:

Remito a ustedes el ejemplar para la defensa de Proyecto de Grado titulado: DESARROLLO DE UNA BILLETERA MOVIL PARA REALIZAR TRANSACCIONES FINANCIERAS CASO: BANCO DE CREDITO DE BOLIVIA "BCP", perteneciente al universitario Peter Ciro Alanoca Aruquipa con cedula de identidad 7031708 LP con RU: 12006408 para la emisión de la Resolución respectiva.

Asimismo, me cabe informar que el mencionado proyecto cuenta con el aval del área de Soluciones de Negocio que ha manifestado su conformidad, en nota adjunta en los ejemplares, y en mi condición de gerente del área de soluciones de negocio, expreso mi aval de conformidad para que el mencionado universitario pueda proseguir con su trabajo para optar al grado de Licenciatura en Ingeniería de Sistemas.

Con este motivo saludo a usted.

Atentamente.

Lic. Luis Loza Pari
Gerente de Soluciones de Negocio